

Adaptive Development Software Development Model

Adaptive Development Software Development Model	1
Introduction	1
The product development cycle	3
Coffee meetings	4
Prototyping.....	5
Create user stories	6
High level planning	6
Initial release planning meeting	6
Release development	7
Review meeting	7
The iteration cycle.....	7
Iteration planning meeting	9
Daily development cycle.....	9
Field trial and customer acceptance	9
Celebrations	10
The daily development cycle	10
The stand up meeting	11
Fix customer issues and failing tests	11
Brainstorm, prototype and design	12
Test and develop	12
Check-in.....	12
Automated testing	12

Introduction

This document assumes some familiarity with agile and XP concepts such as ‘user stories’ and velocity. The most detail is provided where our process differs or extends the

iterative model. The assumption is also made that the document ‘ Adaptive Development Tenets’ has been read.

We follow an iterative development process that is similar to the one described at <http://www.extremeprogramming.org>, with a few customizations and modifications.

This software model focuses on short iterations and getting working software to the customer in a short timeframe. It allows new requirements to be introduced late into the development life cycle and uses the concept of just -in-time development where features are added at the time when they are needed most.

This is not a prescriptive model that should be followed to the letter to ensure success. Instead, the model should be tailored to the specific problem domain . For example is the customer available on-site and are you building a rocket ship guidance system or a news widget? Also, the flows provided in the diagrams in this document are not necessarily sequential. In most cases it is OK to jump backwards or up a level, for example, it is OK to go back to the release planning stage if the customer requirements (user stories) change suddenly mid-way through the iteration.

Besides the customer we do not indicate who would be responsible for each phase. This would differ from company to company. Most importantly, developers that will be responsible for developing the product are involved for as much of the process as possible, since it is on their decisions and development that the ultimate success of the project depends.

The development process can be broken into three cycles: -

- The product development cycle
- The iteration cycle
- The daily development cycle

As discussed in this document, each of these cycles is a more detailed view of a step in the previous cycle.

The product development cycle

The product development cycle describes the high level cycle governing product development.

Refer to the following diagram which is described in detail below :

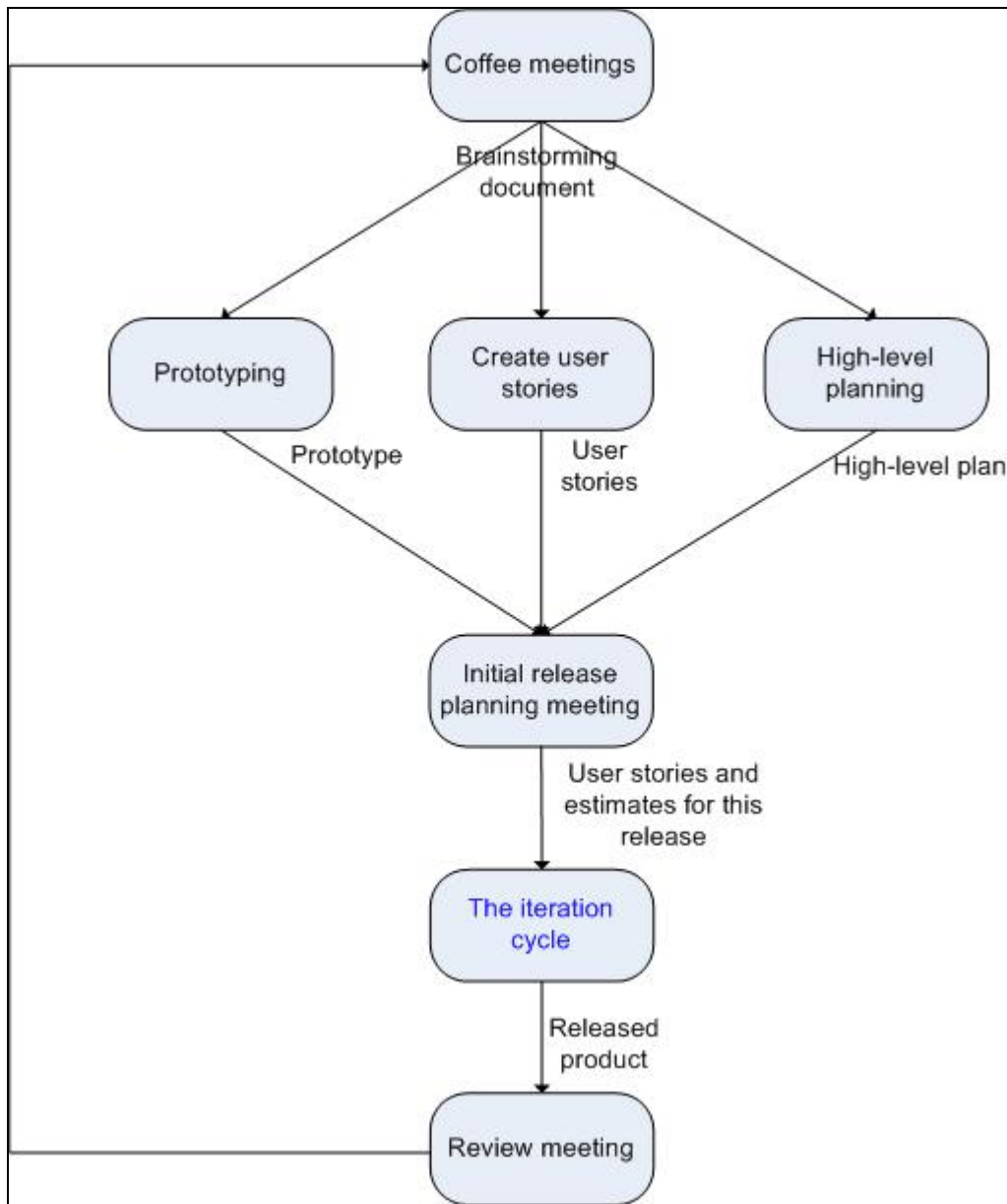


Figure 1

Coffee meetings

‘Coffee meetings’ is a term that describes all the informal discussion between senior stakeholders to discuss the product. At this point the ideas behind the new product (or new development) are still in the fledgling stage. This includes: -

- The first contact by a customer.
- The initial emails and discussions determining whether it is even sensible to take the process to the next stage. For example, if the customer is really looking for a different skill set than what can be offered, or requires a development style that is not supported (e.g. requires all developers on-site all the time and this is not a possibility) then the process will not get past this stage.
- Some high level discussion of roughly how much development time is available for the customer and what sort of cost they can expect. Often, this is left quite late into the negotiation since: -
 1. Available hours and costing are touchy subjects.
 2. It is not easy to give an exact estimate. Both are often dependent on a number of factors such as how interesting the work is and negotiations with existing customers.

However, it is beneficial to give a rough indication of availability and costing otherwise a lot of time and effort can be spent negotiating a deal when there is a fundamental incompatibility that could have been determined early on.

The deliverable from this phase is a brainstorming document that captures the thoughts, ideas, minutes and emails from the discussions and negotiations.

Prototyping

A prototype is quick, pragmatic way to: -

- Determine whether an approach is even possible.
- Quickly discover unknowns.
- Compare different technologies.
- Provide more confident estimates.

Prototyping is discussed in detail in the document – ‘Adaptive Development Tenets’.

Deliverable: prototypes.

Create user stories

User stories are the requirements written in the customer's words. User stories should be light in detail and should only provide enough information to make an estimate of how much work would be required.

Deliverable: user stories.

High level planning

This involves: -

- What process to follow, e.g. the duration of the iterations and the frequency of field trials.
- Level of customer interaction, for example, will the customer be on -site?
- Tools to be used, e.g. the development environment, source control, bug tracking, documentation, Wikis, intranets etc.
- A high level metaphor for the design (if there is one that naturally lends itself to the product being developed), e.g. a conveyor -driven production system.

Deliverable: high level plan.

Initial release planning meeting

This is the first release planning meeting. During this meeting, the developers estimate the user stories in terms of ideal programming weeks / hours. You should record the estimates in terms of tokens rather than weeks or days, since the actual time implied by an estimate changes as the project progresses.

The customer and the developers then negotiate which user stories will be in this release.

Deliverable: Release plan

Release development

This includes the development iterations and is described in more detail later in this document.

Review meeting

Once the final iteration is complete and the customer is happily using the product, it is time for a review meeting. This is an important part of the process since it allows time to consider what went right / wrong in the process and how it can be improved in the future. Even though an adaptive development process consistently monitors and corrects itself based on feedback, it is worthwhile to take some time away from the project and apply the benefit of hindsight.

There is no deliverable as such besides modifying current practices, policies and tools according to the lessons learned.

After this we return to the conceptual stage of the next release and the coffee meeting process.

The iteration cycle

This is the iteration cycle shown in the blue text in figure 1 in the product development cycle above. Each iteration cycle is between 1 and 3 weeks. Each iteration results working software that the customer has accepted. The workflow for this cycle is shown in the diagram below:

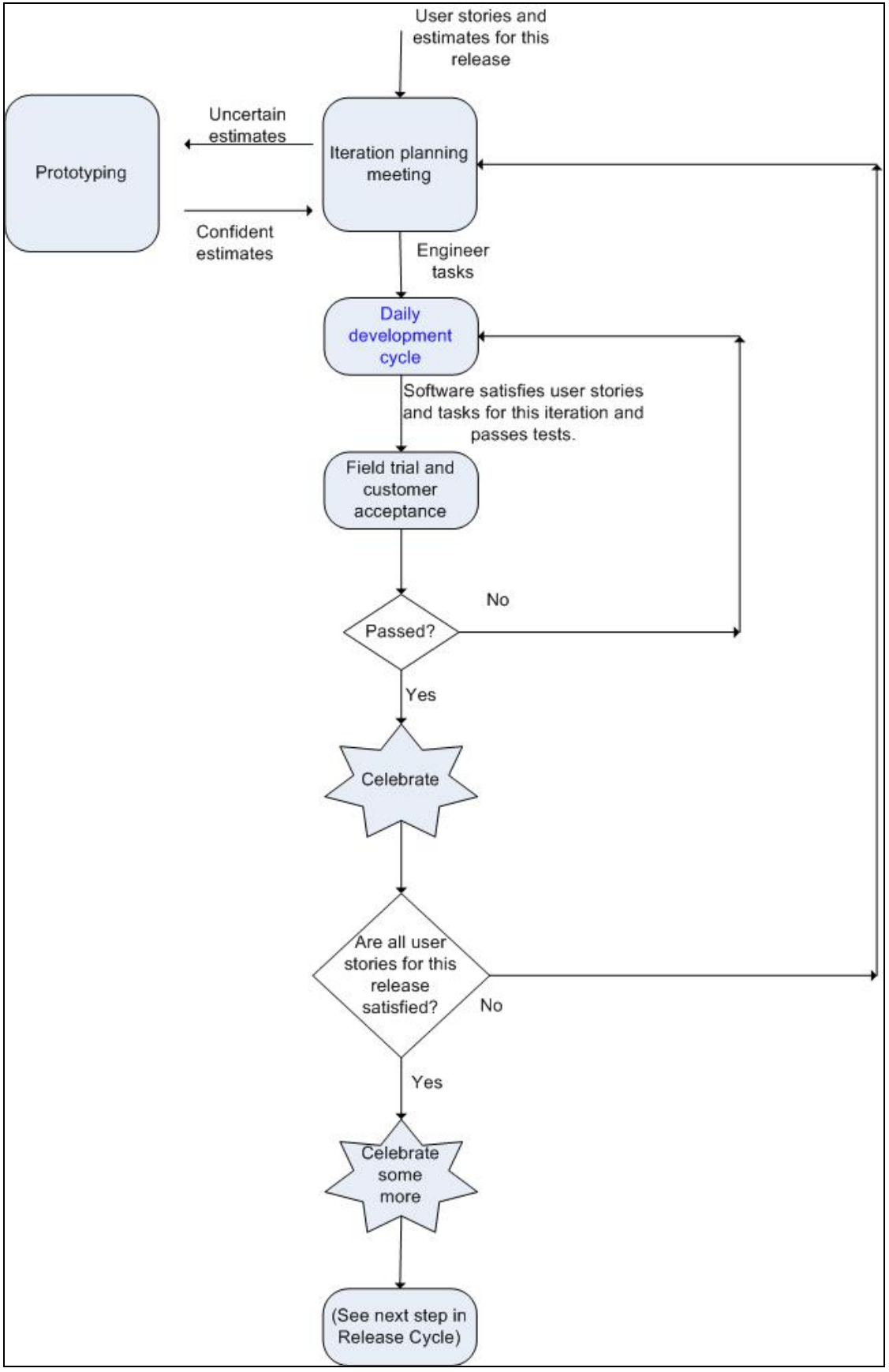


Figure 2

Iteration planning meeting

During the iteration planning meeting, the high priority user stories are broken down into tasks and assigned to engineers. The number of tasks will be determined by the current velocity. For the first iteration choose a small set of tasks and prototypes as an initial phase to provide an initial idea of velocity for the next iteration.

Deliverable: iteration plan (tasks assigned to engineers)

Daily development cycle

This includes the daily development activity and is described in more detail in the next section of this document.

Deliverable: Working software that satisfies the tasks for this iteration and passes testing.

Field trial and customer acceptance

A field trial is where the customer tries the software in a live scenario. Field trials are extremely important; it is only at this stage that you know whether the product solves the true business need. Often, even the customer does not know what they want the product to do and even though they are happy when they try out the product in the lab in a simulated environment, they may actually find that the product does not meet the true business requirements (the requirements that even the customer did not anticipate) until it is running live.

In some cases, the field trial will be limited: -

- The customer is very sensitive to new software going on site.
- The software only satisfies a small subset of the requirements so it would not make much sense running it live.

There are several options in these cases: -

- Do a limited field trial. Instead of having all 100 intended users running the new software, restrict the trial to just 10 users.
- Create a simulated field trial that is as close the real thing as possible where the intended users (rather than the main customer contact) use the software.

Bugs mean that one of the requirements (user stories / tasks) are not satisfied, resolve these before starting the next iteration. Any shortcomings of the product that are not part of the user stories or tasks for this iteration should be discussed, prioritized and assigned (or delayed) during the next iteration meeting.

Celebrations

It is important to take a breather and enjoy some success before diving into the next stage. For example, if an iteration has been accomplished, the team might go out and grab a coffee or lunch.

The daily development cycle

This is the daily development cycle shown in the blue text in figure 2 in the product development cycle above. It covers the typical activities on a daily basis. The workflow for this cycle is shown in the diagram below:

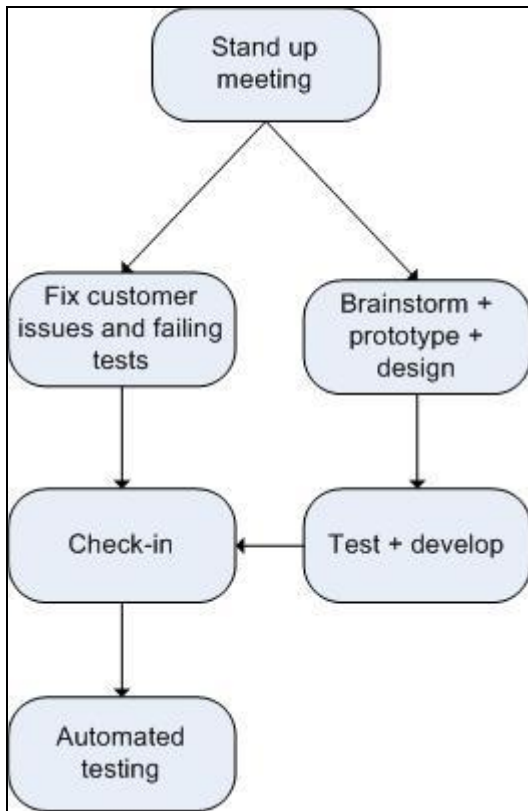


Figure 3

The stand up meeting

- Standing meeting.
- Away from emails, phones and other disturbances.
- Short and to the point.
- Each developer should discuss what they did yesterday, what they will do today and if there is anything stopping them from getting the job done.
- Record the minutes from the meeting as an audit trail for the next day.

Fix customer issues and failing tests

Priority should be given to fixing known bugs over developing new features. Where a bug is found that does not have a test: -

1. Add a new test and make sure that it reproduces the bug.
2. Fix the bug.
3. Run the test and make sure that the test now passes.

Brainstorm, prototype and design

The developers should use lean and effective documentation and design methods.

Test and develop

Sometimes the testing and development is split between developers or between teams, but there should be at least an equal amount of effort put into testing as there is in development.

Check-in

Do regular check-ins, do not allow developer code to get out sync.

Automated testing

After each check-in the automated build should be run. Note that a failing test is just as bad as a compile time bug.