

## **Agile Development (XP)**

### How well do you manage a project?

- Badly?
- Hope and pray?
- With great difficulty?
- Dictate and Motivate?

### Mismanagement

- Produces wrong output
- Inferior quality
- Late
- Working 80 hour weeks

### The date is chosen

- For some reason other than its actual time to develop.
- The date is frozen, project specifications are never frozen.

### Trade offs

- Good (Quality)
- Fast (Time to market)
- Cheap (Cost effectiveness)
- Done

To manage projects we need data!

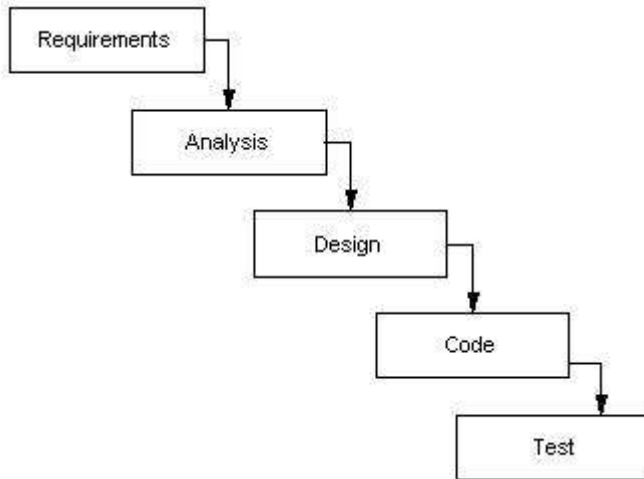
### Agile development uses weekly management.

Points are assigned to the tasks involved, determined by an estimated degree of difficulty. For example, the total points for project A is 500. If the team gets 20 points done, then 480 points remain until the project is finished. This provides a burn down chart like accountants use. It can give management an average number of points per week which can estimate the actual project completion date.

## Iterative Development

This is not a new concept. IBM had a lot of success in the 1960's working with NASA.

In the 1970's most projects took on the waterfall model as shown below:



(Argued as incorrect)

The idea behind the waterfall model may be "measure twice; cut once", and those opposed to the waterfall model argue that this idea tends to fall apart when the problem being measured is constantly changing due to requirement modifications and new realizations about the problem itself.

## Key motivations for iterative development

- Iterative development is lower risk
- Early risk mitigation and discovery
- Accommodates and provokes early change
- Manageable complexity
- Confidence and satisfaction from early, repeated success
- Early partial product
- Relevant progress tracking; better predictability
- Higher quality; less defects
- Final product better matches true client desires
- Early and regular process improvement
- Communication and engagement required
- Prototyping and feedback required

## Using iterative development

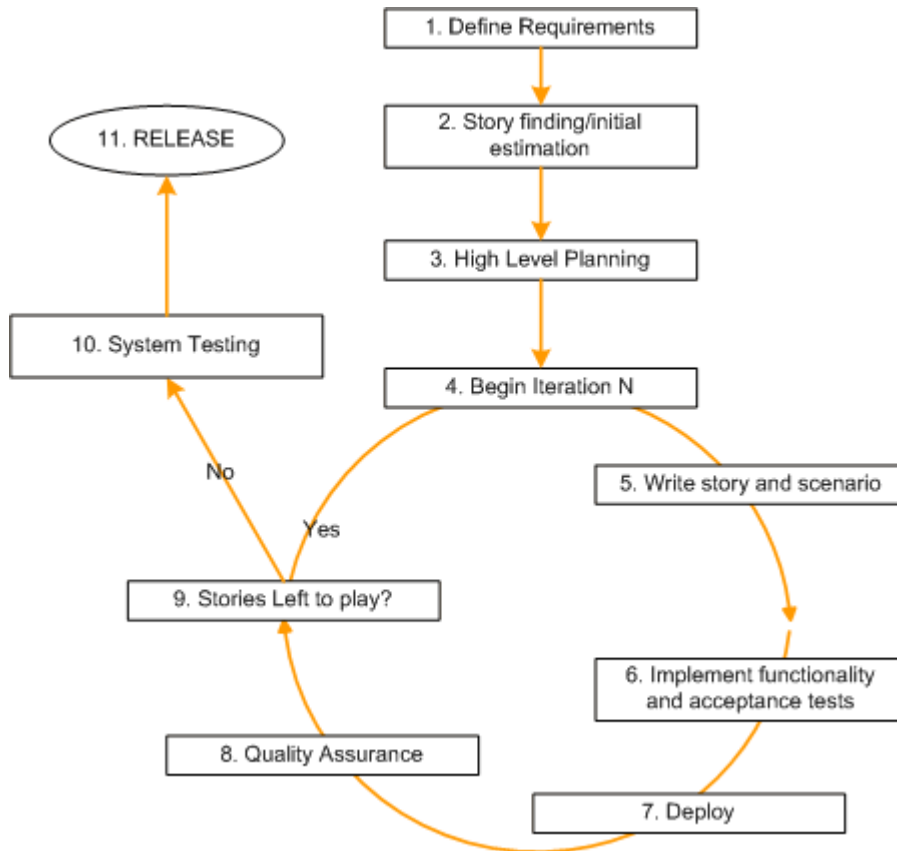
Create a set of bullet items.

List of iterations and what will be done. Say n features per week.

Calibrate the plan as time goes along.

Recalculate the date of completion based on the first week on iterative results.

This is similar to writing a theme paper. Start with a rough draft and keeping rewriting until the final draft.



(Correct)

Write down features on index cards. Anyone can write down a feature. Estimate the cards. These are only relative estimates.

Designing an ATM cash machine system is one example.

Assign points based on the perceived degree of difficulty

Login = 3pts

Logout = 1 pts

Deposit = 5pts

Withdraw = 6pts

Etc.

Pick the cards you think are most important that may add up to 60 points. It's possible you may add more cards and it's also possible you may throw away cards. The goal is the find the highest return on investment.

Repeat this step every week, until done. You'll find that some cards will never be done because you found out later it wasn't needed after all. Slowly the undone pile will become the done pile. The key theme is "release early, release often".

If you cannot get 60 points done per week, you'll know after the first week about how many points you can do. This number of points can now be **tracked**. This will help determine the project's **velocity** and will be more precise over time. Simply stated, velocity is how fast this project is being completed.

In agile development, done means "tested". The agile development process allows for quality assurance during the process. **Acceptance tests** are black box system **tests**. Each **acceptance test** represents some expected result from the system.

Acceptance testing writes the requirements and can be done by anyone. Write a few tests up front per iteration. This allows you to confirm that n # of points are done and working per week.

The iterative project has more interaction with the client and potential project failure can be determined at an early stage, thus reducing risk and minimizing financial loss.

**Related Sources:**

<http://www.objectmentor.com/>

<http://www.extremeprogramming.org/>

<http://en.wikipedia.org/>

mallsop – course notes 2005