System Development Methodologies Cheat Sheet

by Natalie Moore (NatalieMoore) via cheatography.com/19119/cs/9758/

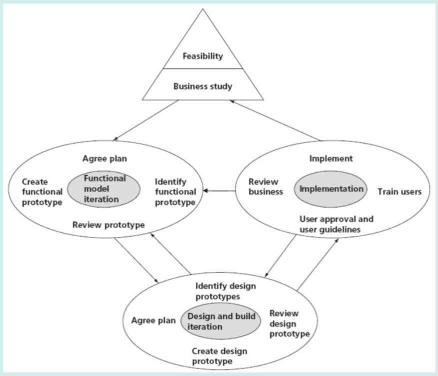
Key aspects of a methodology

Should promote activity which is:
- Purposeful
- Controlled
- Rigorous

and produce results that are:
- Meaningful
- Reproducible
- Relevant

Remember that a methodology is only a means to an end and not an end in itself

Dynamic Systems Development Method Image


Feasibility
Business study
Agree plan
Create functional prototype
Functional model iteration
Identify functional prototype
Review prototype
Implement
Review business
Implementation
Train users
User approval and user guidelines
Identify design prototypes
Agree plan
Design and build iteration
Review design prototype
Create design prototype

Extreme Programming (XP) Methodology

For creating software within a very unstable environment

Allows flexibility within the modelling process

Main goal to lower the cost of change in software requirements

XP Core Practices

- Fine scale feedback
  - Test driven development
  - Planning game
  - Whole team
  - Pair programming
- Continuous process rather than batch
  - Continuous Integration
  - Design Improvement
  - Small Releases
- Shared understanding
  - Simple design
  - System metaphor
  - Collective code ownership
  - Coding standards or coding conventions
- Programmer welfare
  - Sustainable pace (i.e. forty hour week)

Corollary practices

Interaction between developers and customers is good. Therefore, an XP team is supposed to have a customer on site, who specifies and prioritizes work for the team, and who can answer questions as soon as they arise. (In practice, this role is sometimes fulfilled by a customer proxy.)

If learning is good, take it to extremes: Reduce the length of development and feedback cycles. Test early.

Simple code is more likely to work. Therefore, extreme programmers only write code to meet actual needs at the present time in a project, and go to some lengths to reduce complexity and duplication in their code.

If simple code is good, re-write code when it becomes complex.

Code reviews are good. Therefore XP programmers work in pairs, sharing one screen and keyboard (which also improves communication) so that all code is reviewed as it is written.

Testing code is good. Therefore, in XP, tests are written before the code is written. The code is considered complete when it passes the tests (but then it needs refactoring to remove complexity). The system is periodically, or immediately tested using all pre-existing automated tests to assure that it works. See test-driven development.

What can go wrong?

Deadlines

Equipment

Requirements

Vendors

Re-use - Search for ‘Short-Cuts’

Re-use, clone, develop

Packages (modify/tailor)

Share code/libraries/buy-in

Build new interfaces

Umbrella systems

Leverage existing systems/experience

Top tips

Avoid Poor Estimating and/or Scheduling

Avoid Ineffective Stakeholder Management

Avoid Insufficient Risk Management

Avoid Insufficient Planning

Avoid Shortchanging Quality Assurance

Avoid Weak Personnel and/or Team Issues

Avoid Insufficient Project Sponsorship

Why do we have different types?

Degrees of the problem domain, hard and soft

Peoples particular mind sets

Easier to use hard approaches

Soft provide greater insight

System Development Methodologies Cheat Sheet

by Natalie Moore (NatalieMoore) via cheatography.com/19119/cs/9758/

Dynamic Systems Development Method (DSDM)

Principles:

- Active user involvement is imperative
- Focus is on frequent delivery of products
- DSDM teams must be empowered to make decisions
- Fitness for business purpose is the essential criterion for acceptance of deliverables
- Development is iterative and incremental
- All changes during development are reversible
- Requirements are baselined at a high level
- Testing is integrated throughout lifecycle
- Collaborative and co-operative approach between stakeholders is essential

Key Features

- Deliver quickly and often (timeboxing)
- Critical functionality delivered (MoSCoW)
- Joint Application Development workshops (JAD)
- Prototyping/tools used to validate user requirements
- Re-use
- Extreme Programming (user stories, paired programing, focus on communication and teamwork)
- Requirements not fully defined before development
- Culture change

Benefits - overcomes:

Dynamic Systems Development Method (DSDM) (cont)

- Long iteration of refinement/agreement
- Contacting key parties/arranging meetings
- Cycle of meetings
- Resolving different views/perspectives
- Changing requirements during process
- Loss of momentum/commitment

Deliver on time to budget

Does not cut important corners

Results from practical experience

Systems Development Life Cycle (SDLC) Methodology

Conceptual model used in project management

Water fall was original

Describes the stages involved in systems dev project

Documentation is crucial

Documentation is done in parallel with the development process

Most important factor project success may be how closely the plan was followed

Steps:

1. If there is an existing system its deficiencies are identified. Iterview users, consult with support personnel
2. New system requirements are defined
   - addressing any deficiencies
   - with specific proposals for improvement
3. System is designed, plans created for:
   - hardware
   - operating systems
   - programming
   - security issues

Systems Development Life Cycle (SDLC) Methodology (cont)

4. System is developed
   - Components and programs must be obtained and installed.
   - Users must be trained
   - Performance must be tested. Adjustments must be made at this stage.
5. System is deployed
   - might be phased in
   - shut down the old system and implement the new system all at once
6. Monitor and maintain
   - Evaluate system post deploy
   - Maintenance must be kept up
   - Users should be kept up to date

Benefits

Clear project objectives.

Stable project requirements

Progress of system is measurable

Strict sign-off requirements

Disadvantages

Time consuming

Little room for iteration

Difficulty responding to changes

Prototyping

Prototyping/CASE tools

Ability to deliver quickly

Experiment/try out ideas

Project management

Encourages re-use

Version control

Automates testing/system cut-over

Prototype is the technical specification, a repository provides self documentation

System Development Methodologies Cheat Sheet

by Natalie Moore (NatalieMoore) via cheatography.com/19119/cs/9758/

Rapid Application Development (RAD)

Different versions exist

DSDM (Dynamic Systems Development Method)

MOSCOW Rules

Must have – without these features the project is not viable (min. CSFs)

Should have – to gain maximum benefit, these features will be delivered

Could have – if time and resources allow these features will be delivered

Won’t have – these features will not be delivered

Waterfall (a.k.a. Traditional) Methodology

Version of the systems development life cycle model

Classic approach

Rigid and linear

Distinct goals for each phase of development

Each phase is completed before the next one is started

There is no turning back

Benefits

- allows for departmentalization
- allows for managerial control
- Deadlines
- In theory project will be delivered on time due to planning and process

Problems

- often falls short of expectations
- does not embrace the inevitable changes and revisions that become necessary with most projects
- Once an application is in the testing stage, it is very difficult to go back and change something that was not thought of in the concept stage

Scrum Methodology

Agile method

Goal is to dramatically improve productivity in teams previously paralyzed by heavier, process-laden methodologies

Characterized by

- A living backlog of prioritized work to be done
- Completion of a largely fixed set of backlog items in a series of short iterations or sprints
- A brief daily meeting (called a scrum), at which progress is explained, upcoming work is described, and obstacles are raised
- A brief planning session in which the backlog items for the sprint will be defined
- A brief heartbeat retrospective, at which all team members reflect about the past sprint
- Facilitated by a scrum master

Scrum master role

- Primary job is to remove impediments of team to deliver sprint goal
- Not leader of team (team is self-organizing)
- Acts as productivity buffer between team and destabilizing influences

Benefits:

Enables creation of self-organizing teams

Encourages verbal communication across team members and across disciplines

Adopts an empirical approach - accepting the problem cannot be fully understood or defined, focusing instead on maximizing the team's ability to respond in an agile manner to emerging challenges.

Practitioner Advice

Senior management should support the project whole heartedly

Detailed planning should be undertaken

Project management principles should be applied

The key business objectives should be identified and kept in focus

Requirements should be evolved over time by the use of prototypes.

Organisational politics must be considered and navigated

Ensure adequate investment is made

Develop realistic implementation timescales

Use an appropriate development approach/method for the context

Identify users and stakeholders and involve them

Ensure you have skilled, proactive IT/IS people

Changing requirements should be recognised as a normal occurrence and systems need to be put in place to facilitate this

The technology itself should be proven

Any contractors should be managed as if they were an internal team

Problems should be evaluated and where relevant resolved as they are encountered and not ignored

As a last resort, if serious problems are encountered, project timescale should be delayed, rather than risk a disaster

By **Natalie Moore** (NatalieMoore)

cheatography.com/nataliemoore/
www.jchmedia.com/

Published 8th November, 2016.
Last updated 8th November, 2016.