UCL

COMP2009
Software Engineering

Projects and Process
USDP and Agile

UCL

Projects

- A development project needs a *Process* framework to describe:
  - Structure
  - Sequence of activities
  - Management tasks and goals
- A process must be:
  - planned
  - documented
  - understood by all participants
  - feasible
  - repeatable

UCL

What is a development project?

- Dictionary definitions (Longmans):
  - "A specific plan or design"
  - "A planned undertaking"
  - "A large undertaking, e.g., a public works scheme"
- Key points are *planning* and *size* of task.

UCL

Jobs v. Projects

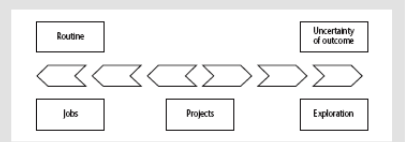


Figure 1.1 Activities most likely to benefit from project management

- *Job* - repetition of very well-defined and well understood tasks with very little uncertainty.
- *Exploration* - e.g., research, finding a cure for common cold; the outcome is very uncertain.
- *Project* - in between!

UCL

Project Characteristics

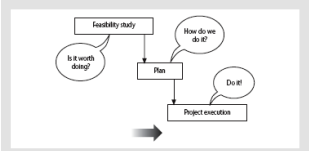
- Non-routine
- *Planned*
- Aiming at a specific outcome
- Carried out for clearly identified client/customer
- Involves a number of specialisms
- Made up of *different phases*
- *Constrained* by time and resources
- Large and/or complex

UCL

Are Software Projects different from other projects?

- Not really, but...
  - Invisibility
    - Progress not visible or easy to measure compared to physical artefacts such as an aeroplane or bridge.
  - Complexity
    - Large software systems more complex than other engineered products per £ spent.
  - Conformity
    - Physical systems (cement, steel) conform to well understood behaviours and constraints. Software doesn't and conforming to requirements is hard.
  - Flexibility
    - Software can be changed very easily and there is an expectation that software will be changed rather than the context it is used in.
    - All make software a lot more problematic to build than other engineered artefacts.

High-Level Project View



- **Feasibility study**
  - Is project technically feasible and worthwhile from a business point of view?
- **Planning**
  - Only done if project is feasible
- **Execution**
  - Implement plan, but plan may be changed as we go along

Setting Objectives

- Answering the question "What do we do to have a success?"
- Need a *project authority.*
  - Sets project scope
  - Allocates and approves costs
- Authority can be one person or a group of people.
  - Project Manager
  - Project Management Board
  - Steering Committee
- Scale of project defines size of authority.

Objectives

- *Informally*, the objective of a project can be defined by completing the statement:

  The project will be regarded as a success if……..

  Rather like *post-conditions* for the project

- Focus on *what* will be put in place, rather than *how* activities will be carried out.

Objectives Should be SMART

- **S** – specific
  - concrete and well-defined.
- **M** – measurable
  - satisfaction of the objective can be objectively judged.
- **A** – achievable
  - it is within the power of the individual or group concerned to meet the target.
- **R** – relevant
  - the objective must relevant to the true purpose of the project.
- **T** – time constrained
  - there is defined point in time by which the objective should be achieved.

Measures of Effectiveness

- How do we know that the goal or objective has been achieved?
- By a practical test, that can be objectively assessed.
- e.g. for user satisfaction with a software product:
  - Repeat business – they buy further products from us.
  - Number of complaints – if low.

Stakeholders

- These are people who have a stake or interest in the project.
- In general, they could be users/clients or developers/ implementers.

- They can be:
  - Within the project team.
  - Outside the project team, but within the same organisation.
  - Outside both the project team and the organisation.
  - Diverse.
  - Many people.

Management Control

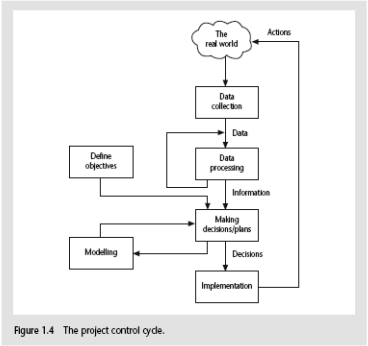


Figure 1.4 The project control cycle.

Management Control (2)

- Data – the raw details.
  - e.g. '6,000 documents processed at location X'
- Information – the data is processed to produce something that is meaningful and useful.
  - e.g. 'productivity is 100 documents a day'.
- Comparison with objectives/goals
  - e.g. we will not meet target of processing all documents by 31st March.

Management Control (3)

- Modelling – working out the probable outcomes of various decisions.
  - e.g. if we employ two more staff at location X how quickly can we get the documents processed?
- Implementation – carrying out the remedial actions that have been decided upon.

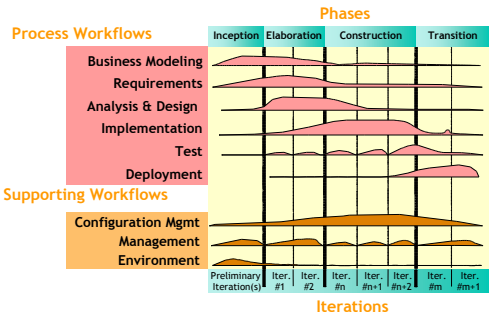
Unified Software Development Process (USDP or just UP) Jacobsen, 1999

- USDP is the development process associated with the Unified Modeling Language (UML)
  - It is *use-case driven* and risk-confronting
  - It is *architecture-centric*
  - It is *iterative* and *incremental*
  - It is free!
- USDP is based on an iterative incremental model.
- Each iteration delivers a part of the system.
- Provides a structural framework for a software development project.

USDP for your project…

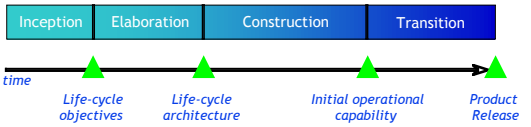
- USDP is a *generic* software engineering process. It has to be customised (instantiated) for your project:
  - In-house standards.
  - Document templates.
  - Tools.
  - Databases.
  - Lifecycle modifications.
- RUP, the Rational Unified Process, is a version of USDP marketed and owned by IBM Software.
- RUP also has to be instantiated for your project.
- Note, UML is a notation not a process.
  - USDP can use UML

Overall Structure of the USDP Lifecycle



Adapted from [Jacobson 1999] RATIONAL

USDP Lifecycle Phases and Milestones



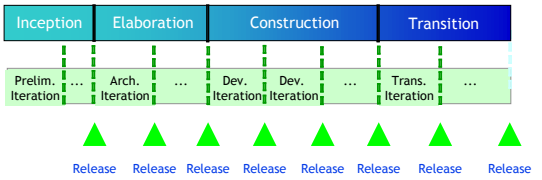
• **Inception**	Define scope of project and develop business case
• **Elaboration**	Plan project, specify features and baseline architecture
• **Construction**	Build product
• **Transition**	Transition product to its users

Adapted from [Booch 1999] RATIONAL

USDP Milestone Acceptance Criteria

- **Lifecycle objectives**: system scope, key requirements, outline architecture, risk assessment, business case, feasibility, agreed project objectives with stakeholders.
- **Lifecycle architecture**: executable architectural baseline, updated risk assessment, project plan to support bidding process, business case verified against plan, continued stakeholder agreement, detailed requirements and use cases.
- **Initial operational capability**: product ready for beta test in user environment.
- **Product release**: completed beta and acceptance tests, defects fixed with fixes in the user community.

USDP Phases and Iterations



An iteration is a sequence of activities with an established plan and evaluation criteria, resulting in an executable release

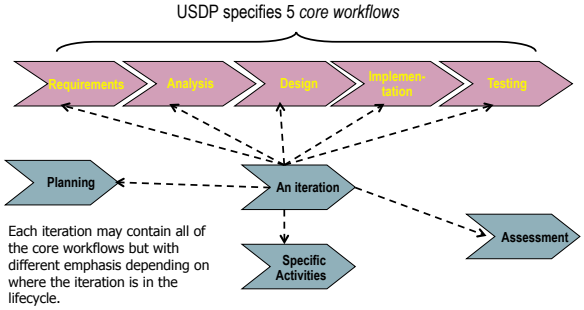
Adapted from [Booch 1999] RATIONAL

USDP Iterations

- *Iteration* is key to USDP
- Each iteration is like a mini-project.
  - Results in an *increment.*
- 5 core workflows during each iteration.
  - Requirements ⇒ analysis ⇒design ⇒ implementation ⇒ test
- Final product release is the result of a sequence of many iterations.

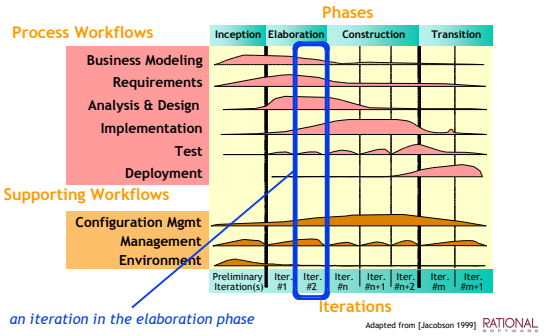
Iteration Workflows

USDP specifies 5 *core workflows*



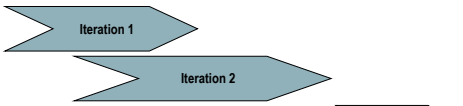
Each iteration may contain all of the core workflows but with different emphasis depending on where the iteration is in the lifecycle.

USDP Phases, Iterations and Workflows



Adapted from [Jacobson 1999] RATIONAL

Iterations may overlap



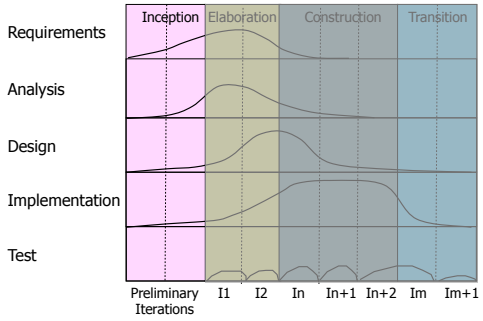
In order to allow parallel development and flexible working in large teams, iterations can, and often do, overlap.
In the example above, Iteration 1 overlaps significantly with iteration 2.

This requires careful planning.

USDP Increments

- Each iteration results in the release of various artefacts, which together are called a *baseline*.
- Baselines assist with review and approvals procedures.
- An *increment* is the difference between two successive baselines.

Inception



Inception - Goals

- Establish feasibility of the project.
- Create a business case.
- Capture key requirements.
- Scope the system.
- Identify critical risks.
- Create proof of concept prototype.
- Decide whether project is feasible and should proceed further.

Inception - Focus

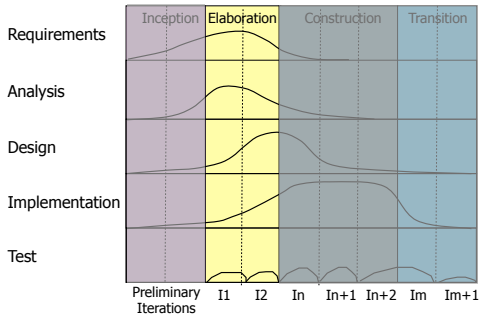
- Requirements – establish business case, scope and core requirements.
- Analysis – establish feasibility.
- Design – design proof of concept or technical prototypes.
- Implementation – build the proof of concept prototype.
- Test – not generally applicable.
  - But do want feedback from stakeholders.

N.B. The bars indicate approximately the relative amount of resource needed

Life Cycle Objectives

- Conditions of satisfaction:
  - System scope has been defined.
  - Key requirements for the system have been captured. These have been defined and agreed with the stakeholders.
  - An architectural vision exists. This is just a sketch at this stage.
  - A Risk Assessment.
  - A Business Case.
  - Project feasibility is confirmed.
  - The stakeholders agree on the objectives of the project.

Elaboration



Elaboration - Goals

- Establish an executable architectural baseline.
- Refine Risk Assessment.
- Define quality attributes (defect rates etc.).
- Capture use-cases around 80% of the functional requirements.
- Create a detailed plan for the construction phase.
- Formulate a bid which includes resources, time, equipment, staff and cost.

How many use-cases?

- Goal is to find sufficient use-cases to allow us to build a system.
- Aim to identify about 80% of the use-cases based on a consideration of functional requirements.
  - The other 20% will come out in later phases if important.
- Aim to model in detail between 40% to 80% of the set of identified use-cases now.
- For each use-case modelled in detail, only a small fraction of the possible scenarios may need to be modelled.

Model just enough use-cases to capture the information you need.

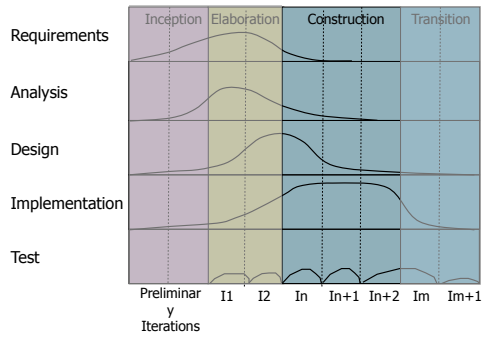
Elaboration - Focus

- Requirements – refine system scope and requirements.
- Analysis – establish what to build.
- Design – create a stable architecture.
- Implementation – build the architectural baseline.
- Test – test the architectural baseline.

Life Cycle Objectives

- Conditions of satisfaction:
  - A resilient, robust executable architectural baseline has been created.
  - The Risk Assessment has been updated.
  - A project plan has been created to enable a realistic bid to be formulated.
  - The business case has been verified against the plan.
  - The stakeholders agree to continue.

Construction



Construction - Goals

- Completing use-case identification, description and realisation.
- Finish analysis, design, implementation and testing.
- Maintain the integrity of the system architecture.
- Create a working system.

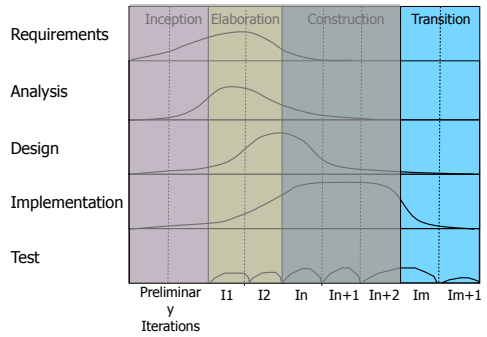
Construction - Focus

- Requirements – uncover any requirements that had been missed.
- Analysis – finish the analysis model.
- Design – finish the design model.
- Implementation – build the Initial Operational Capability.
- Test – test the Initial Operational Capability.

Life Cycle Objectives

- Conditions of satisfaction:
  - The product is ready for beta testing in the user environment.

Transition



Transition - Goals

- Correct defects.
- Prepare the users site for the new software.
- Tailor the software to operate at the users site.
- Modify software if unforeseen problems arise.
- Create user manuals and other documentation.
- Provide customer consultancy.
- Training.
- Conduct post project review.

Transition - Focus

- Requirements – not applicable.
- Analysis – not applicable.
- Design – modify the design if problems emerge in beta testing.
- Implementation – tailor the software for the users site and correct problems uncovered in beta testing.
- Test – beta testing and acceptance testing at the users site.

Life Cycle Objectives

- Product release
- Conditions of satisfaction:
  - Beta testing, acceptance testing and defect repair are finished.
  - The product is released into the user community.

Agile Methods

- Over the past decade there has been much focus on how to improve the development process.
- Agile Methods have emerged as a widely used approach:
  - Focus on iterations, teamwork, collaboration, and process adaptability throughout the life-cycle of the project.
  - Minimal planning and overheads (e.g., short, stand-up meetings).
    - Relies on skilled, professional approach.
    - Minimal != low standards or sloppy work.
  - Code-centred approach.
    - Code embodies the design.
  - Very strong focus on delivering value.

The Agile Manifesto

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

- See http://agilemanifesto.org/

Extreme Programming (Beck 1999)

- A *disciplined*, iterative, *agile* approach to software systems development.
- Some key XP practices
  - Test-driven development
  - The tests are the system specification
  - Releases are as small and frequent as possible
  - Pair programming
  - Collective code ownership
  - Coding standards
  - Continuous integration
  - Frequent refactoring of code
  - Onsite customer is a member of the development team
  - 40-hour work week

When to use XP

- XP works best when
  - Requirements are changing rapidly
  - Projects are high-risk with new challenges
  - Development can be carried out by small groups (2-10 developers)
  - Automated testing is possible
  - Direct customer involvement is possible

Remember that the process must be matched to the problem!

Key Points

- USDP is the iterative and incremental software engineering process for the UML.
- USDP has four phases:
  - Inception, Elaboration, Construction, Transition.
- Each phase may have one or more iterations.
- Each iteration has five core workflows.
  - Requirements, Analysis, Design, Implementation,Test.
- Agile Methods.