Agile Development

Topic 5:

Modeling, Requirements Definition & Prioritization

Topic Coverage

This topic will cover:

- Meanings and examples of modelling;
- Modelling perspectives;
- Modelling within the lifecycle;
- Modelling tips
- What is a requirement?
- Defining the requirements;
- Requirements in the Agile approach lifecycle;
- Requirements and modelling;
- The Requirement Lifecycle.

Exercise 1

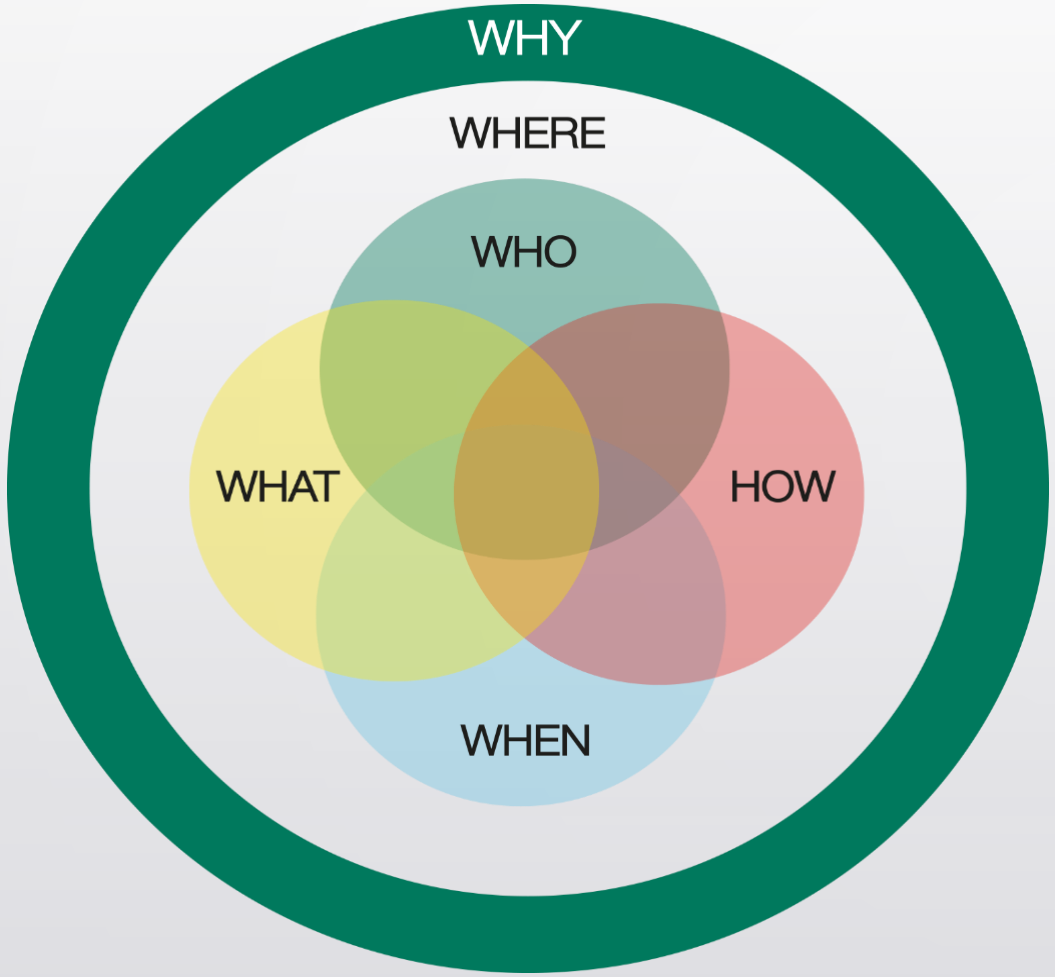
- A model is often a diagram. So, what diagrams do you know/use in a development project?

What is a Model?

A model is:

- A description or analogy (to help visualise something that cannot be directly observed);
- A small but precise replica of an item or product;
- A pattern or figure of an item or product to be created.
- Many industries use models (and prototypes) to establish requirements, confirm expectations and test the achievability of objectives. Examples of models include: storyboards; diagrams; scale models (prototypes); and working software (prototypes).

Modelling Perspectives



WHAT – The information

HOW – The functions, features and processes

WHERE – The locations at which the business operates

WHO – The people: customers, users, stakeholders

WHEN –The events of importance (times and scheduling)

WHY – The business objectives and strategy

Modelling – User Perspective

WHEN
Business Events
time and
scheduling

WHY
Rationale

WHAT?
Data

Technology may have its own:
HOW:
Processes, Inputs, Outputs

WHAT:
Data and Relationships

WHEN:
System Events

WHAT
Business
Procedures

WHO
People
and Tasks

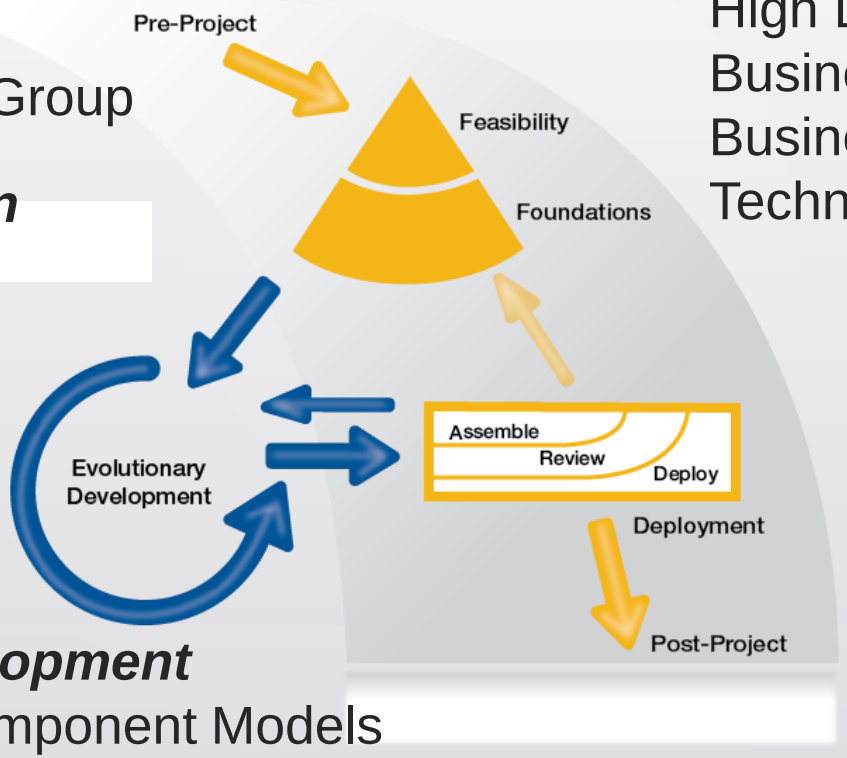
WHERE
Locations and Links

Modelling within the Lifecycle

Feasibility
Scope and Enterprise Model
Business Sponsor
Wider Stakeholder Group

Exploration/Design
Detailed System
Models
Solution
Development Team

Engineering/Development
Technology and Component Models
Solution Development Team



Foundations
High Level System Models
Business Sponsor
Business Visionary
Technical Co-ordinator

Deployment
Functioning, Tested,
Documented System
End Users
Operations

Some User-centered Techniques - 1

User Analysis

- identify user population for the proposed system (job roles, skill levels)

Usability Analysis

- determine characteristics of user interface (non-functional requirements)

Task Modelling

- identify business events (user tasks)

Task Scenario Definition
(& Use Cases)

- identify instances of task execution for a user

Some User-centered Techniques - 2

User Conceptual Modelling
(user object modelling)

- provide a map of the system from the users' perspective

GUI Design

- user interface to support identified tasks

User Interface Prototyping

- provide animated view of proposed system

Modelling Tips

Key points to consider are:

- It is easily understood by user and developer;
- It supports the process of incremental refinement;
- Models produced must enhance communication;
- It must fit easily within the Agile framework.

Summary of Modelling

- **What is modelling?**
  - It is a visualisation process to build focus within a project.
- **Modelling perspectives**
  - When, Why, What, Who, Where and How?
- **Modelling within the lifecycle**
  - Used at all stages by differing stakeholders for varied reasons to ensure project objectives are met.
- **Modelling tips**
  - Make it count! Modelling needs to be easy to understand and fit within all aspects of the life cycle. It will greatly enhance communication.

Exercise: What is a Requirement?

- List as many words or phrases as you can that mean "requirement".

What is a Requirement?

In simple terms, a requirement is a feature, a function, a service or a constraint. For example, requirements can mean the following:

- **Feature/Function** – element of the planned product
- **Service** – a service the project needs to ensure it delivers
- **Constraint** – something that will act as a barrier during the project that needs to be planned for and suitable work arounds achieved.

Defining the Requirements

- It is important to define a requirement along with its acceptance criteria as measurable targets at all levels. It is vital to give each requirement a unique ID, and keep the following details: Source; Owner; Business Benefit; and Priority.

- Two types of requirements
  - Functional
  - Non-functional

Functional Requirements

Functional requirement is ***"what"***, not ***"how".*** It is important to make a requirement SMART: Specific; Measurable; Achievable; Realistic and Timely.

It is also important to consider the wording of a functional requirement. For example, ***"We need the ability to …" or "As a … I need… in order to …".***

Functional requirements should not be in conflict with, or overlap, with other requirements.

Group Exercise: Functional Requirements

- In small groups, consider the requirement; “set up a personal bank account”, Create some SMART requirements based on this from the perspectives of:
  - The customer
  - The bank manager

Non-Functional Requirements – 1

- Non-functional requirements are about "how well" we do the functional requirements. They are features that ensure a finished products work effectively and fits in well with the company brand, such as:

  - Security
  - Availability
  - Portability
  - Maintainability

Non-Functional Requirements – 2

- External interfaces
- Design constraints
- Performance
- Response time

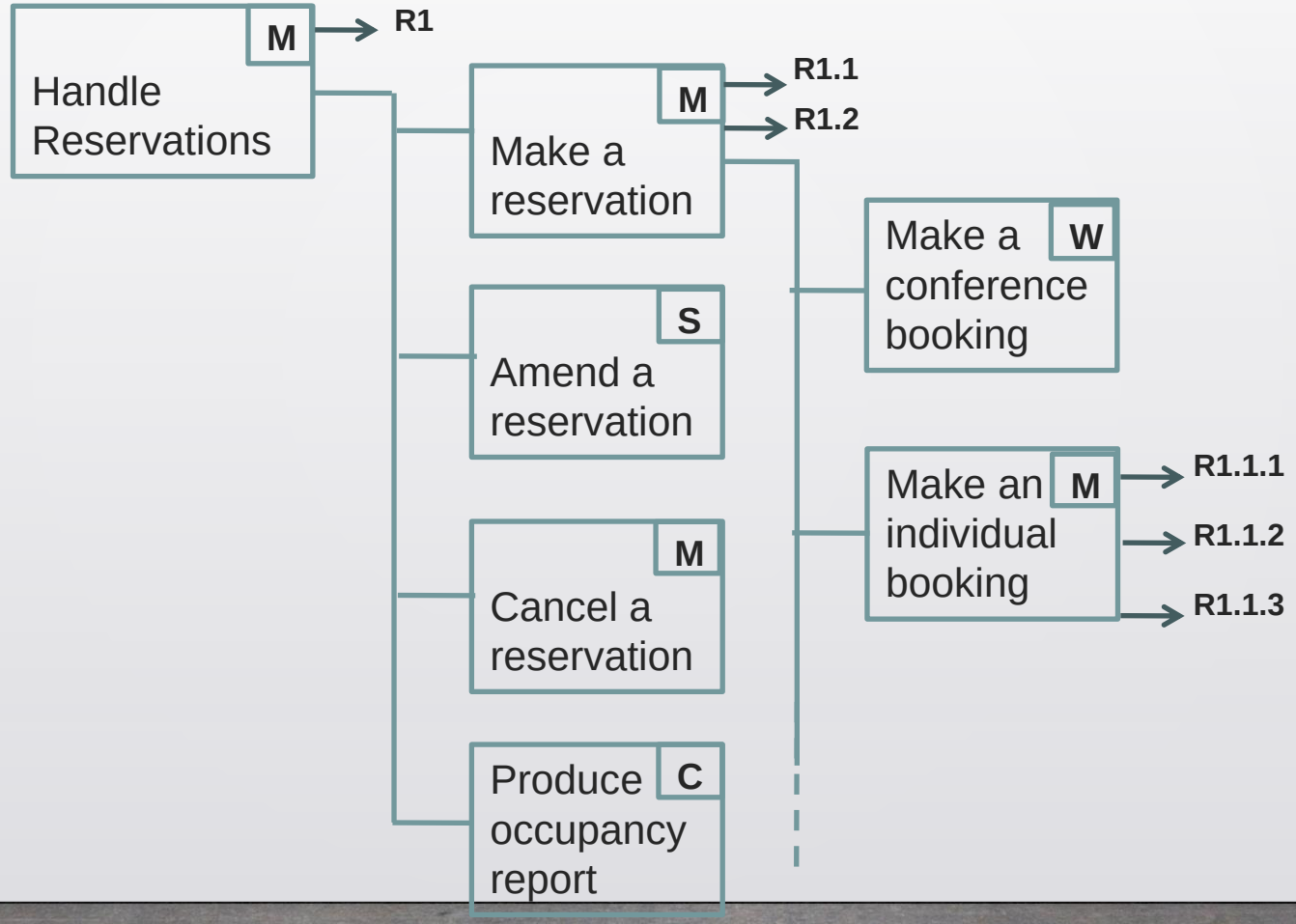
They may be global or related to just one functional requirement. It is likely that they will require extra time in the plan.

Structure of Requirements

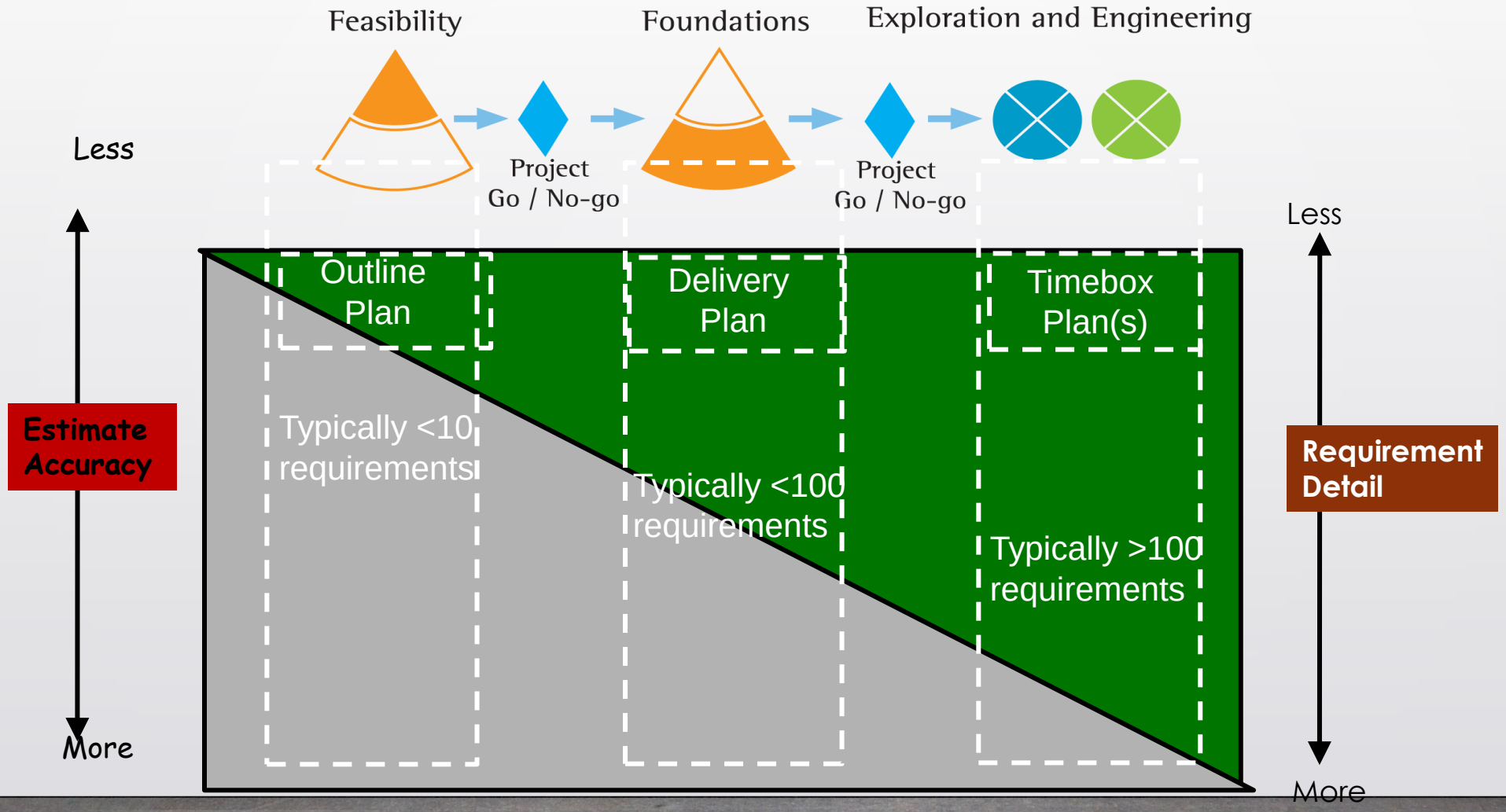
Requirement Decomposition

- Feasibility – A very high level set of requirements is established.
- Foundations – A high level set of prioritised requirements is established (a PRL) and User stories.
- Exploration and Engineering – Each requirement may decompose into more detailed requirements, and, at some point, it may not need to be written down, but evolved as part of iterative development (prototyping).

Requirements and Modelling



Requirements in the DSDM Lifecycle



What is MoSCoW?

Must haves are:

- Fundamental to system;
- Without them, system unworkable/useless;
- Minimum usable subset;
- Guaranteed to be developed.

Should haves include:

- Important requirements;
- Would be mandatory, but workaround exists;
- System will still be useful/usable without them.

What is MoSCoW? – 1

Could haves would add business benefit and are more easily left out of this increment than **Should Haves**. **Won't haves** can be valuable requirements but can wait until a later increment.

Note: All prioritisation is with respect to a clear project objective

The Requirement Lifecycle

Each requirement must be subject to:

- **Elicitation** – Workshops, model-building, interviews, observation, scenarios.
- **Analysis** – Realistic? Ambiguous? Combination of requirements? Aligned with business?
- **Validation** – Prototypes, reviews, models, testing.
- **Management** – Traceability, stability, change management.

Summary of Requirement Definition

- What is a requirement?
  - A function, feature, service or constraint
- Functional and non-functional requirements
  - Functional = *what* not *how*
  - Non-Functional = *how well*
- Requirements in the DSDM lifecycle
  - Requirements increase in number and detail throughout the lifecycle

Summary of Prioritisation

- Must have
- Should have
- Could have
- The requirement life cycle
  - Elicitation
  - Analysis
  - Validation
  - management

More reading Resources

- Chapter 12: Modeling

https://www.agilebusiness.org/page/ProjectFramework_12_Modelling(Last accessed 9th October 2020)

- Chapter 15: Requirements and User Stories https://www.agilebusiness.org/page/ProjectFramework_15_RequirementsandUserStories(Last accessed 9th October 2020)
- Chapter 10: MoSCoW Prioritisation https://www.agilebusiness.org/page/ProjectFramework_10_MoSCoWPrioritisation (Last accessed 9th October 2020)

End of topic ☺

Any Questions?