Communications Software Engineering (Z01)

Wolfgang Emmerich
Dept. of Computer Science
University College London

How to reach me?

Pearson Building, 402

www.cs.ucl.ac.uk/staff/w.emmerich

020 7679 4413

Organization of the Course

- *15 Lectures this week*
  - *every morning from 9-12 (G23)*
- *Coursework: Group Project in Z15*
- *Coursework mark is course mark!*
- *No written exam paper*

Aims and Objectives

- *In the short run:*
  - *Provide Foundation for Group Project*
- *In the long run:*
  - *Achieve proficiency in modern software engineering notations, methods & tools*
    - *Formal Methods: Process Algebra (FSP)*
    - *Unified Modeling Language*
    - *Use-case centred requirements elicitation*
    - *Object-oriented design for distributed systems*
- *Enable you to define an appropriate software process for a distributed development project*

Outline of the Course

- *Introduction*
- *Software Process*
- *Requirements Analysis*
- *Intro to Formal Methods*
- *Process Algebra (FSP)*
- *Condition Synchronization*
- *Starvation and Deadlocks*
- *Liveness and Progress*
- *Design Model*
- *Distributed Objects*
- *Design Patterns*
- *Project Management*
- *CASE*

Literature

[JCJO92] *I. Jacobson, M.Christerson, P.Jonsson and G.Oevergaard: Object-Oriented Software Engineering - A Use Case Driven Approach, Addison-Wesley, 1992.*

[BRJ99] *G. Booch, J. Rumbaugh and I. Jacobson. The Unified Modeling Language User Guide. Addison-Wesley, 1999*

[MK99] *J. Magee and J. Kramer. Concurrency: State Models and Java Programs. John Wiley & Sons. 1999*

[Emm99] *Engineering Distributed Objects. John Wiley & Sons. 2000. http://www.cs.ucl.ac.uk/staff/emmerich/book. To appear.*

Z02: Communications Software Engineering Introduction

Wolfgang Emmerich

What do you know about SE?

On a blank sheet of paper, write down three to five concepts, which you believe are essential to Software Engineering!

Outline

- *What is Software Engineering?*
- *Software Development Processes*
- *Software Process Case Study*

Software Engineering

- *Ian Sommerville:*
  - *Software engineering is concerned with methods, tools and techniques for developing and managing the process of creating and evolving software products;*
- *Oxford English Dictionary:*
  - *the professional development, production, and management of system software;*

Why Software Engineering?

- *The economies of ALL developed nations are dependent on software*
- *More and more systems are software controlled*
- *Software engineering is concerned with theories, methods and tools for professional software development*
- *Software engineering expenditure represents a significant fraction of GNP in all developed countries*

Software Costs

- *Software costs often dominate system costs. The costs of software on a PC are often greater than the hardware cost*
- *Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times the development costs*
- *Software engineering is concerned with cost-effective software development*

Software Products

- *Generic products*
  - *Stand-alone systems which are produced by a development organisation and sold on the open market to any customer*
- *Bespoke (customised) products*
  - *Systems which are commissioned by a specific customer and developed specially by some contractor*
- *Most software expenditure is on generic products but most development effort is on bespoke systems*

Software Product Attributes

- *Maintainability: SW should be evolvable to meet changing requirements*
- *Dependability: SW should not cause physical/economic damage upon failure*
- *Efficiency: SW should not make wasteful use of system resources*
- *Usability: SW should have an appropriate user interface and documentation*

Outline

- *What is Software Engineering?*
- *Software Development Processes*
- *Software Process Case Study*

Software Development Process

- *Structured set of activities required to develop a software system*
  - *Requirements Analysis*
  - *Design*
  - *Implementation*
  - *Validation*
  - *Evolution*
- *Activities vary depending on organisation and type of system being developed*
- *Must be modelled to be managed*

Process characteristics

- *Understandability: Is the process defined and understandable*
- *Visibility: Is the process progress externally visible*
- *Supportability: Can the process be supported by CASE tools*
- *Acceptability: Is the process acceptable to those involved in it*

Process characteristics

- *Reliability: Are process errors discovered before they result in product errors*
- *Robustness: Can the process continue in spite of unexpected problems*
- *Maintainability: Can the process evolve to meet changing organisational needs*
- *Rapidity: How quickly can the system be produced*

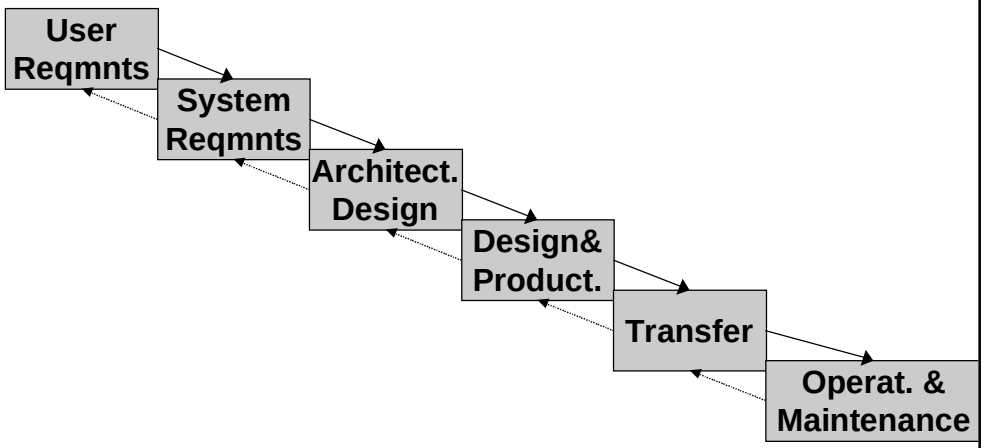
Software process models

- *Normally, specifications are incomplete/anomalous*
- *Very blurred distinction between specification, design and manufacture*
- *No physical realisation of the system for testing*
- *Software does not wear out - maintenance does not mean component replacement*

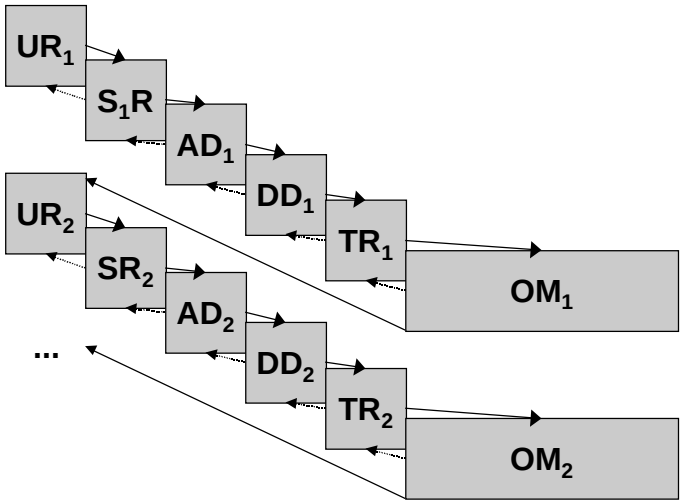
Generic software process models

- *The waterfall model*
  - *Separate and distinct phases of specification and development*
- *Evolutionary development*
  - *Interleaved specification and development*
- *Incremental development*
  - *Development of increments*
- *Formal transformation*
  - *A mathematical system model is formally transformed to an implementation*

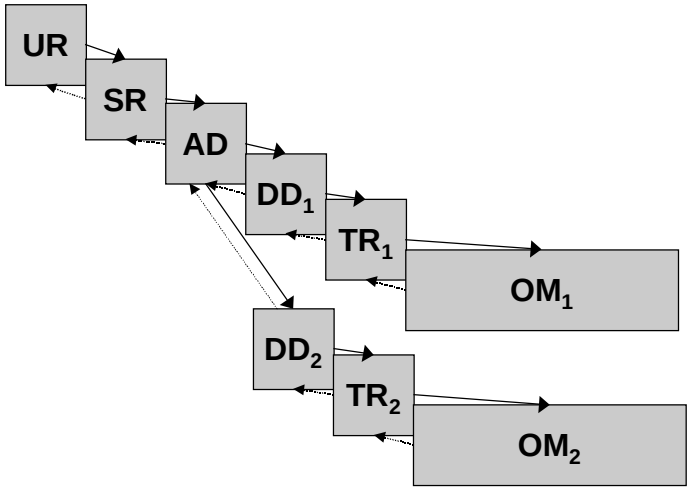
Waterfall Approach



Evolutionary Development



Incremental Development



Risk management

- *Perhaps the principal task of a manager is to minimise risk*
- *The 'risk' inherent in an activity is a measure of the uncertainty of the outcome of that activity*
- *High-risk activities cause schedule and cost overruns*
- *Risk is related to the amount and quality of available information. The less information, the higher the risk*

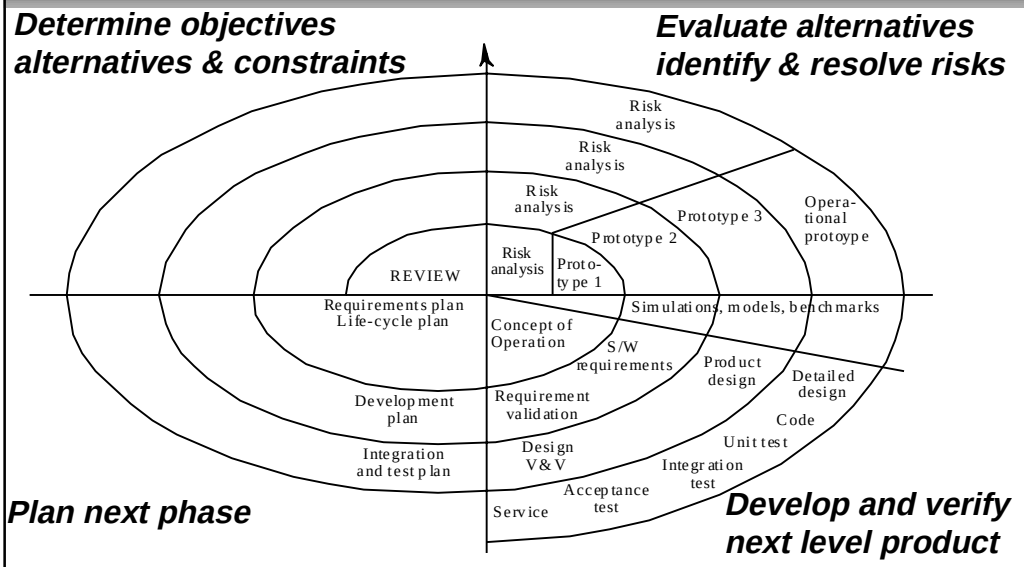
Process model risk problems

- *Waterfall*
  - *High risk for new systems because of specification and design problems*
  - *Low risk for well-understood developments using familiar technology*
- *Evolutionary*
  - *Low risk for new applications because specification and program stay in step*
  - *High risk because of lack of process visibility*

Hybrid process models

- *Large systems are usually made up of several sub-systems*
- *The same process model need not be used for all subsystems*
- *Prototyping for high-risk specifications*
- *Waterfall model for well-understood developments*

Boehm's Spiral model



Phases of the Spiral Model

- *Objective setting: Specific objectives for the project phase are identified*
- *Risk assessment and reduction: Key risks are identified, analysed and information is sought to reduce these risks*
- *Development and validation: Find model for next development phase*
- *Planning: Review project and plans next round of the spiral*

Template for a spiral round

- *Objectives*
- *Constraints*
- *Alternatives*
- *Risks*
- *Risk resolution*
- *Results*
- *Plans*
- *Commitment*

Spiral model flexibility

- *Well-understood systems (low technical risk) - Waterfall model. Risk analysis phase is relatively cheap*
- *Stable requirements and formal specification. Safety criticality - Formal transformation model*
- *High UI risk, incomplete specification - prototyping model*
- *Hybrid models accommodated for different parts of the project*

Spiral model advantages

- *Focuses attention on reuse options*
- *Focuses attention on early error elimination*
- *Puts quality objectives up front*
- *Integrates development and maintenance*
- *Provides a framework for hardware/software development*

Spiral model problems

- *Contractual development often specifies process model and deliverables in advance*
- *Requires risk assessment expertise*
- *Needs refinement for general use*

Outline

- *What is Software Engineering?*
- *Software Development Processes*
- *Software Process Case Study*

What are Standards?

“ Standards are documented agreements containing technical specifications or other precise criteria to be used consistently as rules, guidelines, or definitions of characteristics, to ensure that materials, products, processes and services are fit for their purpose.” [ISO 1997]

Software Engineering Standards

- *Normative and informative reference defining how to develop software or software intensive systems*
- *Document centred*
- *Scope for adoption to*
  - *organisation /or*
  - *project needs*

Overview of SE Standards

- *Int. Software Engineering Standards*
  - *PSS-05 (ESA)*
  - *ISO-12207*
- *Important American Standards*
  - *DoD Mil-Std 2915*
  - *IEEE 1074-1995*
- *Software Process Improvement Standards*
  - *ISO 15504*
  - *SEI Capability Maturity Model*

PSS-05

PSS-05 defines practices for:

- *production phases,*
- *software lifecycle and*
- *management phases.*

A PSS-05 practice can be:

- *mandatory (“shall”),*
- *recommended (“should”) and*
- *guiding (“may”).*

PSS-05 Production Phases

- *User requirements*
- *Software requirements*
- *Architectural design*
- *Detailed design & production of code*
- *Transfer of software to operations*
- *Operations and maintenance*

PSS-05 Production Phases

PSS-05 practices determine for each phase:

- *Input documents*
- *Activities to be conducted*
- *Output documents*

PSS-05 Example Practices for SR Phase

- *For incremental delivery, each software requirement shall include a measure of priority so that the developer can decide the production schedule.*
- *Critical functions should be identified.*
- *The SRD shall be compiled according to the table of contents provided in Appendix C.*

PSS-05 Software Lifecycle

- *Three lifecycle approaches are prescribed*
- *Process Engineer can select one of*
  - *Waterfall*
  - *Incremental Delivery*
  - *Evolutionary Development*

PSS-05 Management Phases

- *Software project management (SPM)*
- *Software configuration management (SCM)*
- *Software verification and validation (SVV)*
- *Software quality assurance (SQA)*

PSS-05 Document Templates

- *Standard includes a dozen document templates*
- *Documents have to conform to structure of templates*
- *PSS-05 templates are based on IEEE Stds. 730, 828, 829, 830, 1012, 1016, 1058 and 1063.*

PSS-05 Template Sample: SRD

1 Introduction
2 General Description
2.1 Relation to current projects
Describe the relationship to other projects
...
3 Specific Requirements
List the specific requirements, with attributes. Subsections may be regrouped around high-level functions.
3.1 Functional requirements
3.2 Performance requirements
3.2 Interface requirements
3.3 Operational requirements
3.5 Resource requirements
3.6 Verification requirements
...
...

Key Points

- *Software engineering is concerned with the theories, methods and tools for developing, managing and evolving software products*
- *The software process consists of those activities involved in software development*

Key points

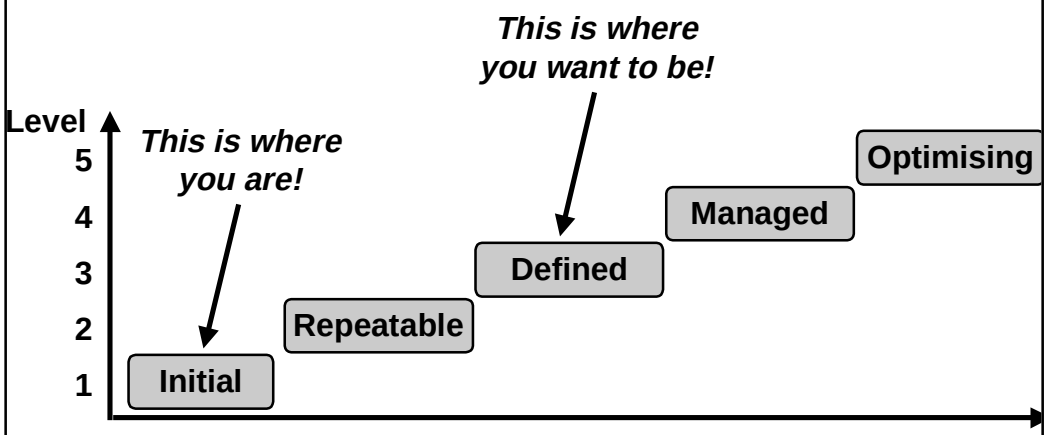
- *The waterfall model considers each process activity as a discrete phase*
- *Evolutionary development considers process activities as concurrent*
- *The spiral process model is risk-driven*
- *Software engineering standards determine refine software processes*

Why Define Process?

- *Introduction of good practice*
- *Tailoring towards Bank & Project*
- *Agreement on development process*
- *Basis for quality control*
- *Basis for process improvement*
- *Supports recruitment of internal / contracting of external developers*
- *Improves collaboration / embedding of contractors*
- *Records current experience for future use*

Capability Maturity Model

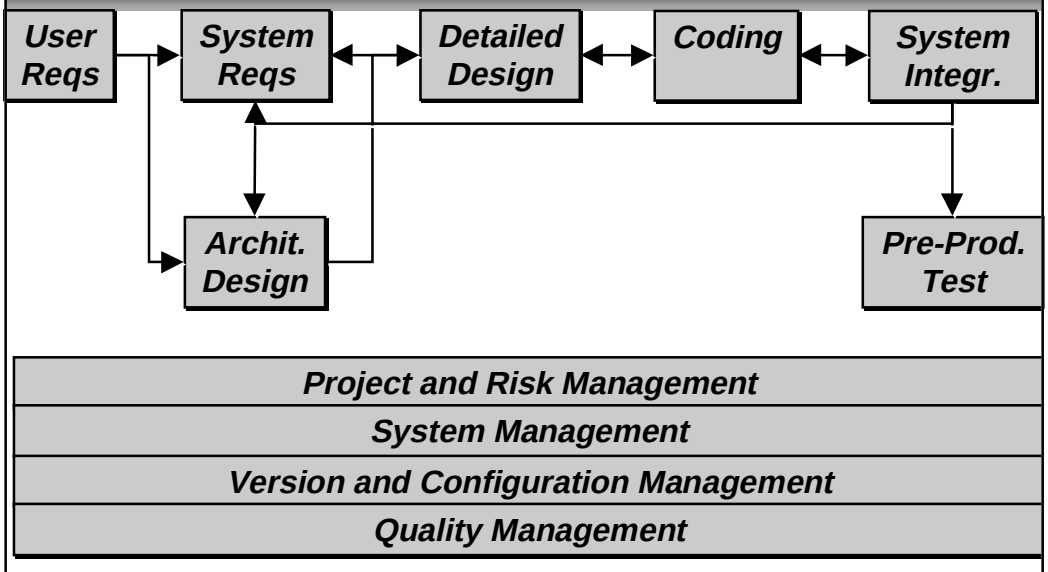
- *Developed by Watts Humphrey at the SEI*
- *US DoD requires Level 3 maturity of contractors*



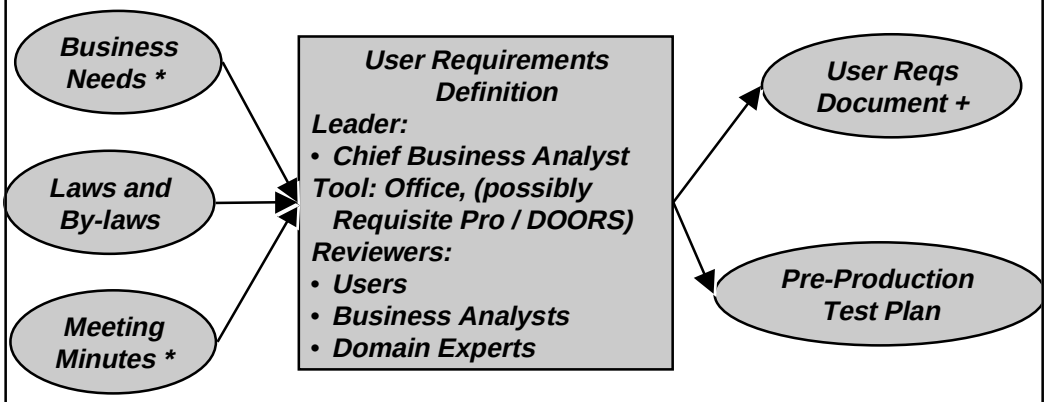
What needs to be defined?

- **Activities**
  - Leader
  - Reviewers
  - Resulting Deliverables
  - Dependencies to/from other activities
- **Deliverables**
  - Purpose
  - Content
  - Owner
  - Tools to produce them
- **Roles**
  - Reporting structure
  - Responsibilities
  - Skills

Stages of Development Process



User Requirements Def.



+ To be completed after MRS goes into production
** Optional (at this stage of the project)*

User Reqs Document Contents

Owner: Chief Business Analyst

1 Introduction
- 1.1 Purpose of the Document
- 1.2 Scope of the software
- 1.3 References

2 General Description
- 2.1 Product Perspective
- 2.2 General Capabilities
- 2.3 General Constraints
- 2.4 User Characteristics
- 2.5 Operational Environment
- 2.6 Assumptions and Dependencies
- 2.7 Business Process

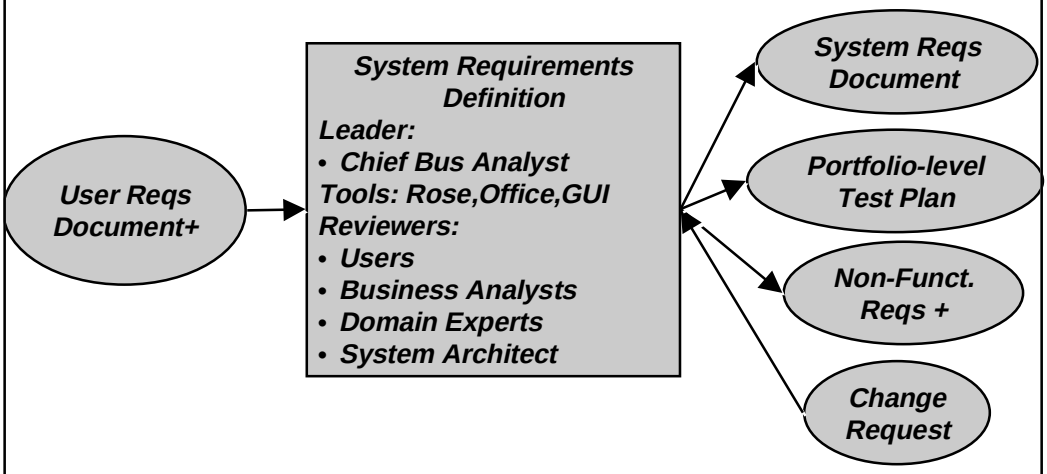
3 Specific Requirements
- 3.1 Capability Requirements
- 3.2 Constraint Requirements

4 Glossary of definitions, acronyms and abbreviations

Pre-Production Test Plan

- *Owner: Chief Business Analyst*
- *Purpose of pre-production test:*
  - *Test from users' perspective*
  - *Test from operating perspective*
  - *Test integration with external systems*
  - *Test functionality of the entire system*
  - *Test in (simulated) deployment environment*
- *Plan tests to be run before production*

System Requirements Def.



System Reqs Document

Owner: Chief Business Analyst

1 UML Use Case Model (Rose) +

2 UML Interaction Diagrams (Rose) +

- *UML Sequence Diagrams*
- *UML Collaboration Diagrams*

3 UML Analysis Class Diagram (Rose) +

Analysis Document

*4 User Interface Prototypes (GUI Tools, Office) **

5 Specification of Report Format (Office)

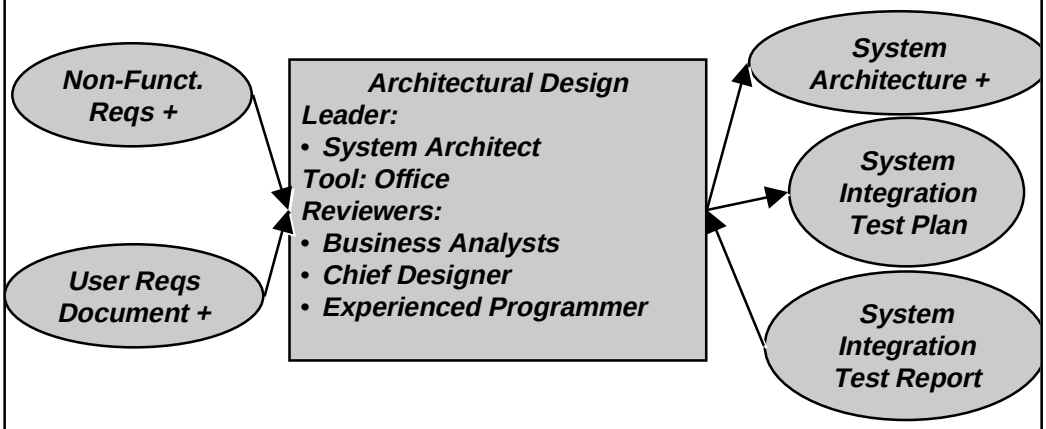
6 User Manual (Office)

- *User Reference Manual*
- *User's Guide*

Non-Functional Requirements

- *Owner: Chief Business Analyst*
- *Capacity Requirements*
- *Performance Requirements*
- *Accuracy Requirements*
- *Constraint Requirements*
  - *Communication Interfaces*
  - *Hardware Interfaces*
  - *Interfaces to external software components*
  - *HCI Requirements*
  - *Adaptability Requirements*
  - *Availability Requirements*
  - *Portability Requirements*
  - *Security Requirements*
  - *Compliance to Standards*
  - *Resources*
  - *Time scales*

Architectural Design



System Architecture

Owner: System Architect

1 Architecture Overview and Rationale

2 Services provided by Modules

3 Resources used by Modules
(disk, memory, CPU time, real-time)

4 Dependencies between Modules
- *Control flow*
- *Data flow*
- *Synchronization*

5 Rationale of chosen implementation languages

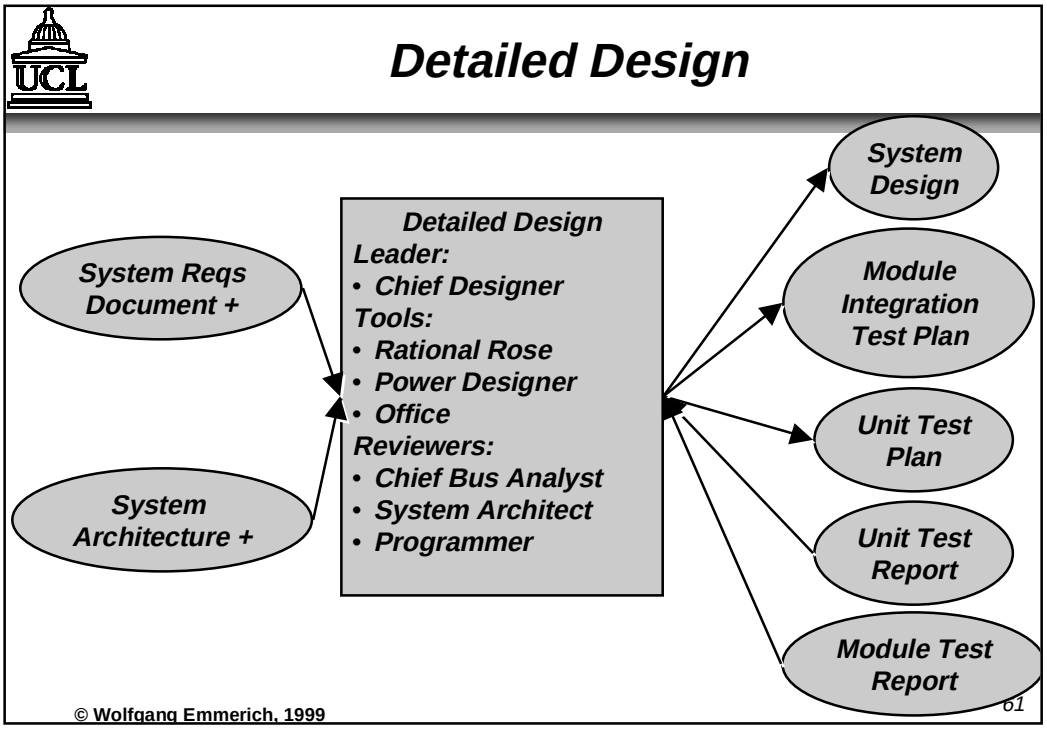
6 Traceability to non-functional requirements?

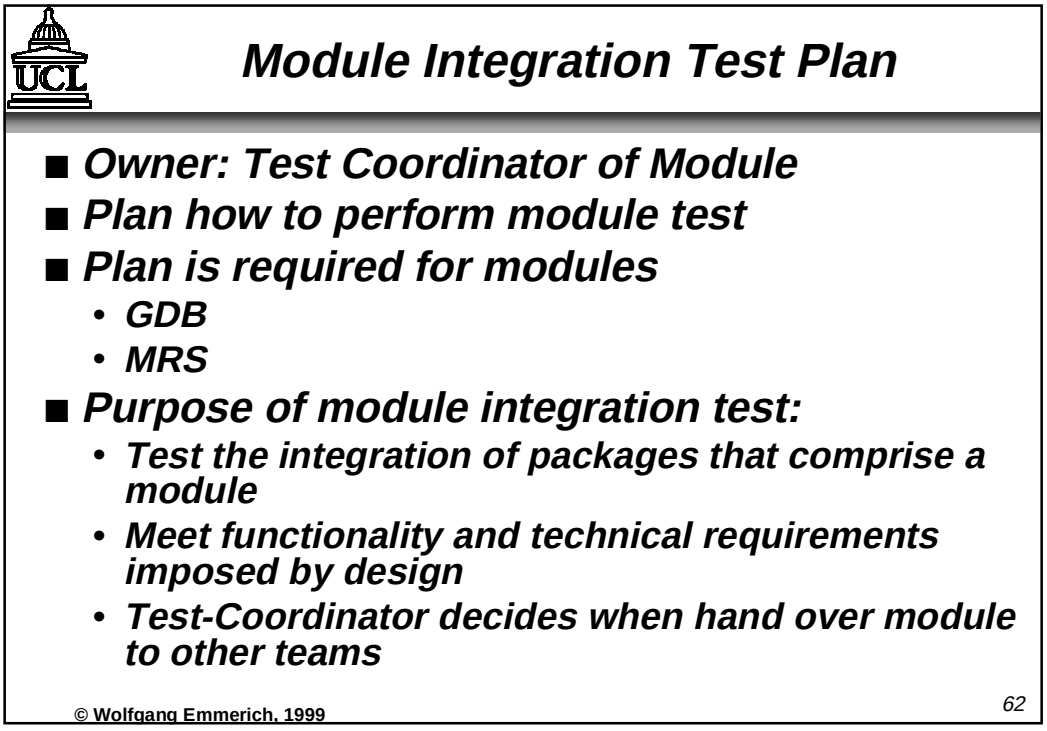
Portfolio-level Test Plan

- *Owner: Chief Business Analyst*
- *Purpose of portfolio-level test*
  - *Tests correctness of Value at Risk Figures based on trade portfolios*
  - *Compare Exact Pricing VaR computed by MRS/CAD and MRS/CAD light (spreadsheet)*
  - *Test hierarchical simulation and variance/covariance VaR by comparison to Exact Pricing*
  - *Test Back-testing and stress-testing*
- *Plan how to perform portfolio-level test*

System Integration Test Plan

- *Owner: Test Manager*
- *Plan how to test integrated system*
- *Purpose of System Integration Test:*
  - *Validate that modules work together*
  - *Trace how trades are processed through the system*
  - *Validate that non-functional requirements are met*


Detailed Design
System Reqs Document +
System Architecture +
Detailed Design
Leader:
• Chief Designer
Tools:
• Rational Rose
• Power Designer
• Office
Reviewers:
• Chief Bus Analyst
• System Architect
• Programmer
System Design
Module Integration Test Plan
Unit Test Plan
Unit Test Report
Module Test Report
© Wolfgang Emmerich, 1999
61


Module Integration Test Plan
■ Owner: Test Coordinator of Module
■ Plan how to perform module test
■ Plan is required for modules
• GDB
• MRS
■ Purpose of module integration test:
• Test the integration of packages that comprise a module
• Meet functionality and technical requirements imposed by design
• Test-Coordinator decides when hand over module to other teams
© Wolfgang Emmerich, 1999
62

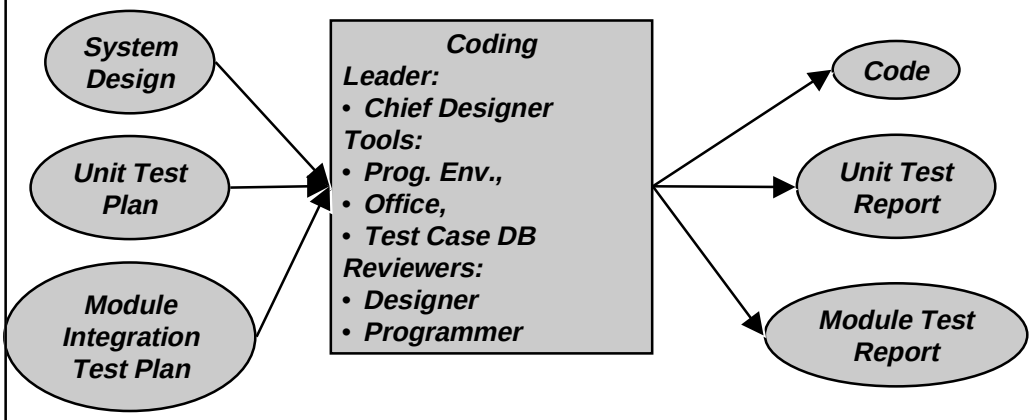
Unit Test Plan

- *Owner: Test Coordinator of resp. Module*
- *Plan how to perform unit tests*
- *Plan is required for each unit (UML package in MRS and GDB, classes in SAI)*
- *Purpose of Unit Test*
  - *Black/Box Test in MRS and GDB*
  - *White/Box Test in SAI*
  - *Test functionality exported by unit*
  - *Test functionality and behaviour of design*
  - *Test behaviour with invalid input parameters*
  - *Test coordinator decides when to hand over unit to other teams*

System Design

- *Owner: Chief Designer*
- *Design Overview*
- *Detailed Design: UML Class Diagrams (Rose) and Design Paper (Soda)*
  - *Broken down into packages*
  - *Responsibility delegation for each package*
- *UML State Diagrams for critical classes (Rose)*
- *UML Interaction Diagrams for critical scenarios (Rose)*
  - *UML Sequence Diagrams*
  - *UML Collaboration Diagrams*
- *UML Component Diagram (Rose)*
- *Data Model (Power Designer)*

Coding



Code

Owner: Responsible for Package

1 Implementations of Classes

- ***C++ Code***
- ***Documentation***
- ***Rationale for Algorithms chosen***

2 SQL schemas

3 Makefiles

4 Scripts, libraries, executables, configfiles

Unit Test Report

Owner: Test Coordinator of Module

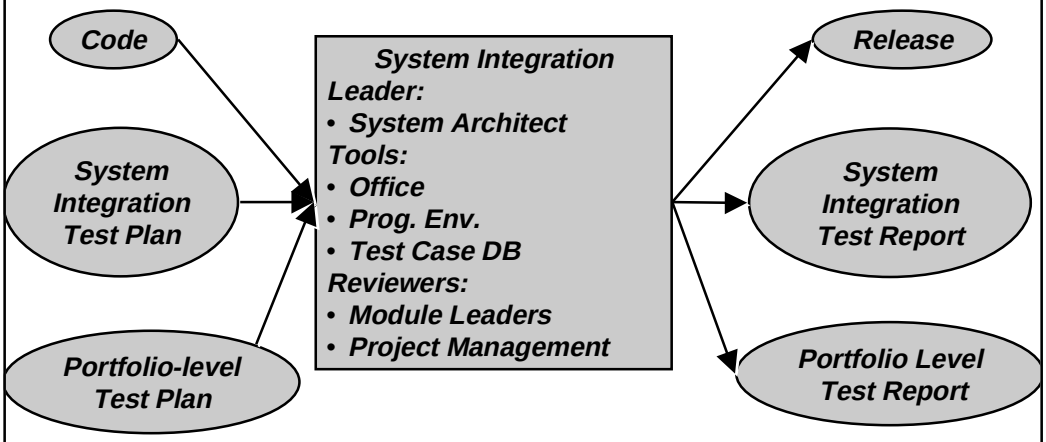
1 Test Plan (From Detailed Design)

2 Test Architecture

3 Test Case Description

4 Test Reports

System Integration



Release

- *Owner: System Architect*
- *Executable Increment*
- *Release Notes*
- *Package for Release Installation*
- *Installation Manual*

System Integration Test Report

Owner: Test Manager

1 Test Plan (from Architectural Design)

2 Test Architecture

3 Test Case Descriptions

4 Test Reports

Portfolio-level Test Report

Owner: Chief Business Analyst

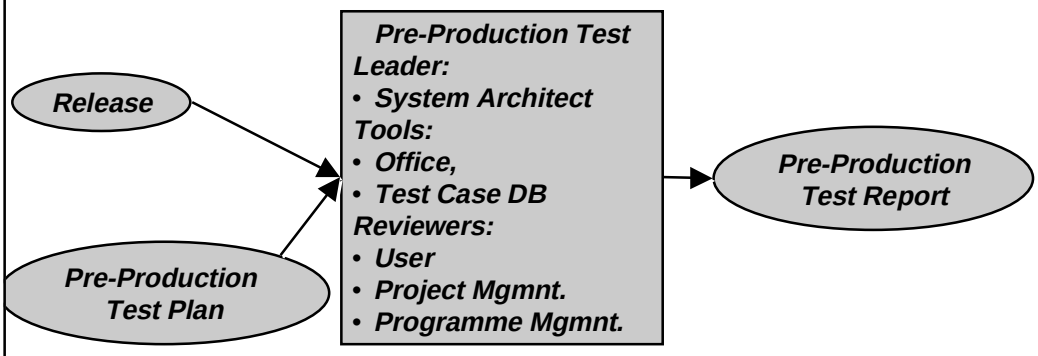
1 Test Plan (from System Requirements)

2 Test Architecture

3 Test Case Descriptions

4 Test Reports

Pre-Production Test



Pre-Production Test Report

Owner: System Architect

1 Test Plan (from User Requirements)

2 Test Architecture

3 Test Case Descriptions

4 Test Reports

1 Test Plan

1 Introduction (summary of items and features to be tested)
2 Test Items (list items to be tested)
3 Features to be tested (features to be tested and why)
4 Features not to be tested (features untested and why)
5 Approach (outline how tests will be carried out)
6 Item pass/fail Criteria (Specify when tests pass or fail)
7 Suspension criteria and resumption requirements
8 Test deliverables (before start and after end)
9 Testing tasks (tasks needed to prepare and carry out test)
10 Environmental needs (properties of test environment)
11 Responsibilities (authorize test, perform test, check)
12 Staffing and training needs (skills of staff,training needs)
13 Schedule (summarize when test activities will be done)
14 Risks and contingencies (risk assumptions, mitigate)
15 Approvals (specify who must approve this plan)

2 Test Architecture

For each test architecture

2.n.1 Test Design Identifier

Give a unique identifier to the test design

2.n.2 Features to be tested

List the features to be tested

2.n.3 Approach refinements

Describe how the tests will be done

2.n.4 Test case identifications

List the specific test cases.

2.n.5 Feature pass/fail criteria

Specify criteria for passing or failing a test

3 Test Case Specification

For each test case:

3.n.1 Test case identifier

Give a unique identifier for the case

3.n.2 Test items

List the items to be tested

3.n.3 Input specifications

Describe the input for the case

3.n.4 Output specifications

Describe the output required from case

3.n.5 Environmental needs

Describe the test environment

3.n.6 Special procedural requirements

Describe special constraints on this case

3.n.7 Inter case dependencies

Test cases that must precede this case

4 Test Reports

For each execution of a test procedure

4.n.1 Test report identifier

Give a unique identifier for the test report

4.n.2 Description

List the items being tested

4.n.3 Activity and event entries

Identify the test procedure.

Say when the test was done, who did it and who witnessed it.

Describe the environmental conditions.

Describe what happened.

Describe where the outputs of the test procedure are kept.

Project & Risk Management

- ***Owner: Project Manager***
- ***Activities:***
  - ***Organising the Project***
    - ***define roles of team members***
    - ***define team structure***
    - ***document responsibilities of team members***
  - ***Risk Management***
    - ***quality and stability of user requirements***
    - ***level of definition & stability of external interfaces***
    - ***adequacy and availability of resources***
    - ***availability and quality of tools***
    - ***staff training and experience***
    - ***definition of responsibilities***
    - ***short time scales***
    - ***technical novelty of the project***

System Management

- *Owner: System Manager*
- *Activities:*
  - *Maintain Hardware/Software Platform*
  - *Maintain Network*
  - *Maintain Tool Infrastructures*
  - *Help desk*
  - *Administer Databases*
  - *Administer and Backup File stores*
  - *Administer Developer Accounts*

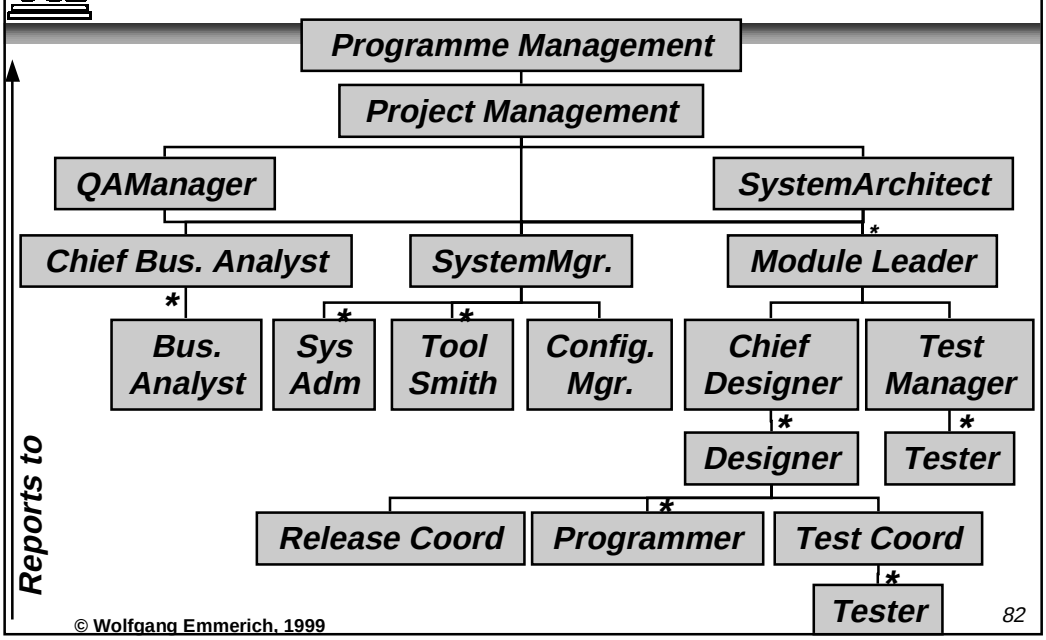
Version & Configuration Mgmt.

- *Owner: Configuration Manager*
- *Activities*
  - *Administer configurations of all deliverables (not just the code!)*
  - *Identify configuration items*
  - *...*

Quality Management

- *Owner: Quality Manager*
- *Activities:*
  - *Develop and maintain quality plan*
  - *Monitor compliance to*
    - *quality plan*
    - *this process definition*
  - *By means of*
    - *Audits*
    - *Reviews*
    - *Inspections*

Roles



Programme Management

- **Responsibilities**
  - Agree on project aims and objectives
  - Agree on project plan (schedule, costs, quality, resources)
  - Receive monthly reports from project management
  - Decide on staffing resources
  - Agree / Reject changes to project plan
- **Skills**
  - Very strong leadership
  - Diplomacy
  - Communication skills

Project Management

- **Responsibilities:**
  - Planning scope, resources, deadlines, quality
  - Determining priorities
  - Reviewing Module leaders' plans
  - Liasing with related departments
  - Reporting of serious unresolved problems to Programme Management
- **Skills:**
  - Strong leadership
  - Technical and domain knowledge
  - Substantial project management experience

System Architect

- ***Responsibilities:***
  - ***Own and enforce the System Architecture***
  - ***Assess technical risks***
  - ***Define content of successive iterations***
  - ***Mediate conflicts between module teams***
  - ***Technical liaison with related projects***
  - ***Consultancy***
- ***Skills:***
  - ***Experience: Problem domain & OO SE***
  - ***Vision, Leadership, Communication***
  - ***Proactive and goal-oriented***
  - ***Risk taker***

Chief Business Analyst

- ***Responsibilities:***
  - ***Advise and support the Architect***
  - ***Mentor and lead Business Analysts***
  - ***Arrange and attend Analysis reviews***
  - ***Define key interfaces***
- ***Skills***
  - ***Leadership: manage Business Analysts***
  - ***Communication***
  - ***Proactive and goal oriented***
  - ***Strong analysis skills and domain knowledge***
  - ***Experience in testing business functionality***

Business Analyst

- ***Responsibilities:***
  - ***Work with Chief Business Analyst***
  - ***Define interfaces and identify classes, packages, patterns***
  - ***Testing***
- ***Skills***
  - ***Communication skills***
  - ***Excellent analysis skills (finding abstractions)***
  - ***Excellent business domain knowledge***
  - ***Some programming skills***

Quality Assurance Manager

- ***Responsibilities***
  - ***Own and enforce Project Standards***
  - ***Work with Project Manager and Architect***
  - ***Organise and attend reviews***
  - ***Own and enforce testing and metrication***
- ***Skills***
  - ***Knowledge of testing procedures***
  - ***Strong interpersonal skills***
  - ***Some analysis/design/coding skills***

Module Leader

- *Responsibilities:*
- *Planning of modules with respect to*
  - *Schedule, resources, quality, contents*
  - *Leader of module team*
  - *Responsible for QA and Documentation*
  - *Reporting of serious unresolved problems*
- *Skills:*
  - *Strong leadership*
  - *Project management experience*

Configuration Manager

- *Responsibilities*
  - *Owns set up and operation of the whole CM environment*
  - *Ensures CM standards are adhered to*
  - *Provides CM training/tools to team members*
- *Skills*
  - *Detailed knowledge of the CM system*
  - *Good communication skills*
  - *Risk aversive*
  - *Reliable and trustworthy*

Toolsmith

- ***Responsibilities***
  - ***Buy or build the best software tools to maximise team productivity***
- ***Skills***
  - ***Intimate knowledge of available tools***
  - ***Good communication skills***
  - ***Excellent coding Skills***
  - ***Inventive, proactive and goal-oriented***
  - ***Likes to tinker***

Chief Designer

- ***Responsibilities***
  - ***Manage team of Designers***
  - ***Co-ordinate all design activities***
  - ***Work with System Architect and Chief Business Analyst to keep design and analysis models in step***
- ***Skills***
  - ***Leadership: manage Designers***
  - ***Good communication skills***
  - ***Proactive and goal oriented***
  - ***Strong design and programming skills***

Designer

- ***Responsibilities***
  - ***Mentor and lead 3 to 4 programmers***
  - ***Detailed class design***
  - ***Owns one or more packages from detailed design through to implementation***
- ***Skills***
  - ***Leadership of small teams***
  - ***Good communication skills***
  - ***Excellent design and programming skills***

Test Manager

- ***Responsibilities***
  - ***Creates test plan for system testing***
  - ***Creates testing standards and works with QA Manager to enforce them***
  - ***Manages a team of testers***
- ***Skills***
  - ***Leadership***
  - ***Excellent communication skills***
  - ***Experience in testing***

Tester

- ***Responsibilities***
  - ***Unit and system testing***
  - ***Designing test strategies***
  - ***Track and document all tests***
- ***Skills***
  - ***Knowledge of testing***
  - ***Programming skills***

Programmer

- ***Responsibilities***
  - ***Implement the design model***
  - ***Tactical class design***
  - ***Class-level testing***
  - ***Participate in code walkthroughs***
- ***Skills***
  - ***Good coding skills and likes to code!***
  - ***Familiar with basic OOA/OOD principles***
  - ***Understands the modelling language***
  - ***Perhaps has a specialisation e.g. GUI***

Release Coordinator

- **Responsibilities**
  - Works with Users, Project Manager and System Architect to create a release plan
  - Monitors and maintains the plan
  - Advises the project manager on any non-technical risks related to the release plan
- **Skills**
  - Good communication and interpersonal skills
  - Good domain knowledge
  - Project planning skills

Test Coordinator

- **Responsibilities**
  - Create test plan for module and unit tests
  - Works with QA manager to enforce test plans
  - May manage more testers
- **Skills**
  - Excellent communication skills
  - Experience in testing

Summary of Benefits

- *Introduce proven process into MRS*
- *Indicate documentation contents as basis for quality control*
- *Assist in identifying problems/risks early*
- *Guide staffing of development team*
- *Identify how to migrate to this process*
- *Record current experience for future use*