Introduction to

Requirements Gathering

Prepared for:

St. Edwards University

Analysis, Modeling and Design MCIS6310 – Dr. David Franke

Outline

1. Overview of Requirements Management
   - Who is involved?
   - How do we approach the space?
   - What outputs do we create?

2. Structuring Requirements Information
   - How do we categorize requirements?
   - What is the hierarchy of requirements?

3. Managing the Requirements
   - How do we gather, maintain and use requirements?

Outline

1. Overview of Requirements Management
   - Who is involved?
   - How do we approach the space?
   - What outputs do we create?
2. Structuring Requirements Information
   - How do we categorize requirements?
   - What is the hierarchy of requirements?
3. Managing the Requirements
   - How do we gather, maintain and use requirements?

Who Is Involved?

- Primarily two groups are involved in designing, developing and deploying a system or solution
  - Stakeholders of the system
    - Sponsor / Customer
    - User
  - Creators of the system
    - Strategy (What do we build?)
    - Execution (How do we build it?)
    - Subject matter experts (SME) / BA
    - Product and program mgr / BA
    - Developers
    - Quality engineers

Stakeholders

- Sponsor / Customer
  - Funds the project
  - Benefits from the project
  - Great contact for that “vision thing”
- User
  - Uses the solution after deployment
  - Great source of validation
  - Great source of post-deployment feedback

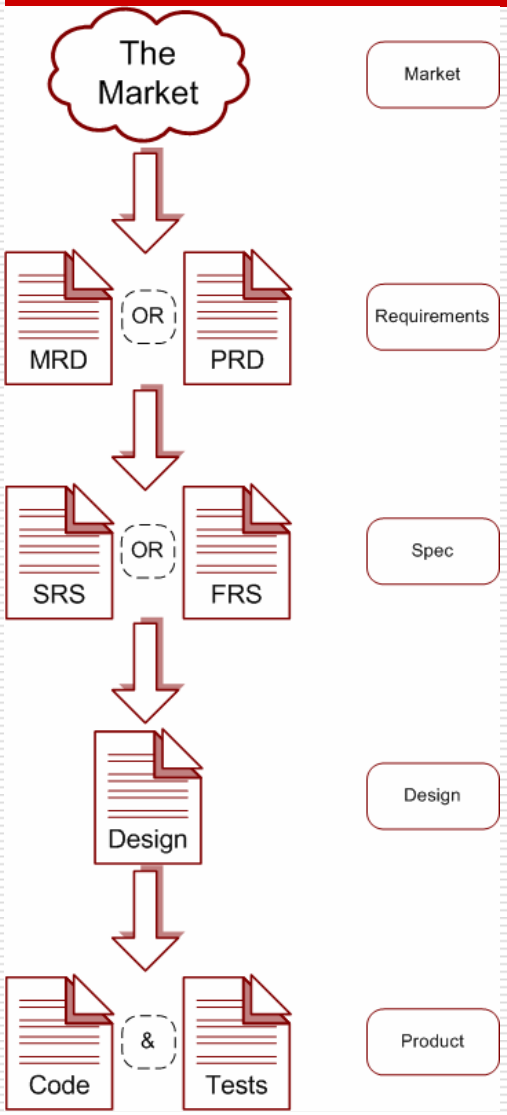
Creators - Strategy

- SME (subject matter expert) / BA (domain)
  - Domain expertise, problem definition
- Product Manager
  - Prioritization and definition of market requirements
  - Defines vision to achieve benefits
    - Outbound (marketing, sales support)
    - Inbound (product definition, release planning)
- Requirements / Program Manager / BA (software)
  - Defines what will be delivered and when
  - Validates that requirements will achieve vision/goals
  - Coordinates activities of all parties
  - Manages documentation and communication

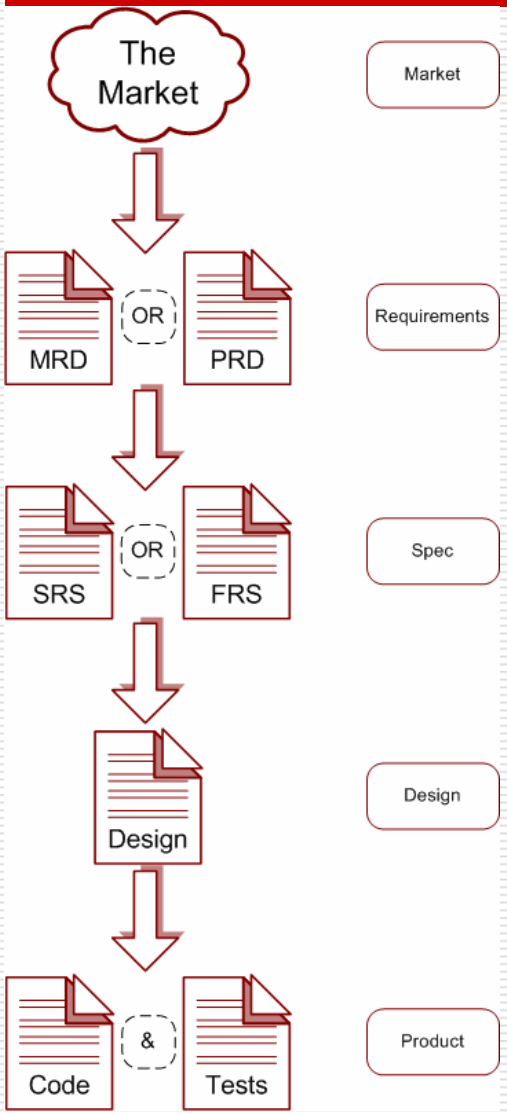
Creators - Execution

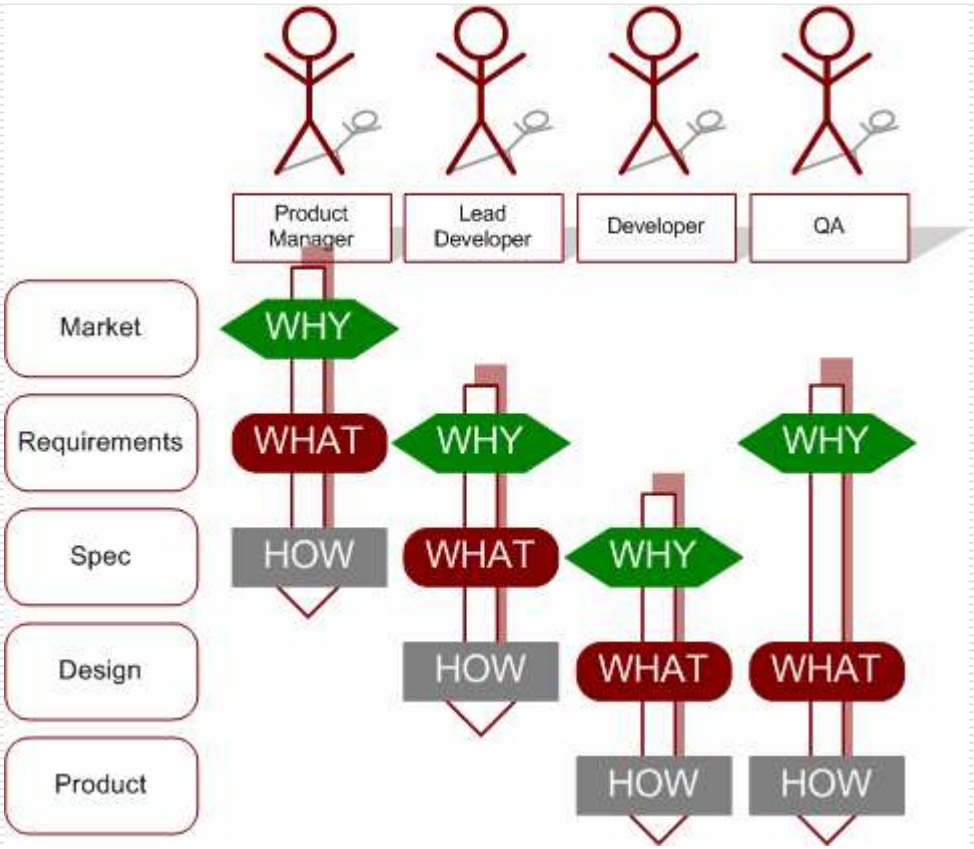
- Project Manager / Development Manager
  - Manages the execution of deliverables
- Designer & Architect
  - Interface Design and Program Design
- Developer
  - Responsible for delivering components of solution
- Quality Engineer
  - Responsible for assuring quality of solution

Graphical View of Creation



Graphical View of Creation





Example Requirement

- Market
  - Music sales are dropping with 18-35 yr olds.
- MRD
  - Online music sales are growing at double the rate of in-store sales decline. Create a music catalog with sample clips for ordering music online.
- PRD
  - We need to sell songs and albums online for downloading and media purchase.
  - We need to enable customers to preview songs before purchasing.
  - We need customers to be able to find available music.
- SRS
  - The system shall present all available media in a searchable catalog.
  - Sample downloads will be available from product pages when samples are available.
  - Downloadable products will be available from product pages when available.
  - …

Crossing the Line

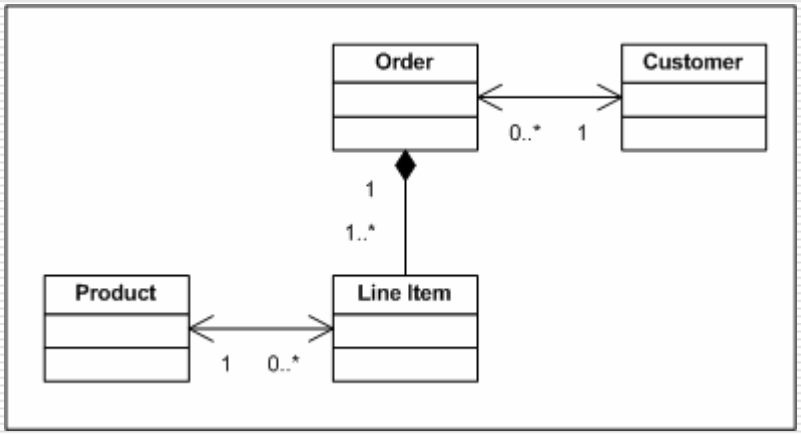
- Difference between requirements and design – simple to define, hard to live by
- If the language talks about implementation, then it's probably design
  - "Users select their language from a dropdown list"
  - "Users can specify their default language"
  - "Search all records created in the database between the specified dates"
  - "Search all records with a creation date between the specified dates"
- Reality gets in the way sometimes
  - "Populate the list of customers from <external system>"
  - "Populate the list of customers from TBL1.CUST_ID not TBL2.CUST_ID in <external system>"

Requirements Frameworks (1)

- Karl Wiegers methodology
  - Framework for describing requirements hierarchically (Goal>Use Case>Functional requirement>Test)
- User centered design (UCD)
  - Software features and functions are driven (formally) from use cases, scenarios, or scripts.
  - Relation to value (profits) is informal
- Interaction Design
  - Identifies primary users of the system
  - Designs solution specifically for them
  - Applies psychology, human factors and others

Requirements Frameworks (2)

- OOA/OOD (Object oriented analysis/design)
  - Technique using UML to present analysis findings
  - Build two object models –
    - one that describes the system (e.g. Customers have Orders)
    - one that describes the solution (class Customer has a collection of Order objects)



Artifacts

- MRD / PRD for strategy
- Use cases bridge PRD to FRS
- SRS / FRS for tactics
- OOA/OOD diagrams
- Stakeholder documents
  - Release schedule / Product roadmap
  - ROI analysis / SOW
  - Status updates

- Screenshots? Technically no.

Outline

1. Overview of Requirements Management
   - Who is involved?
   - How do we approach the space?
   - What outputs do we create?

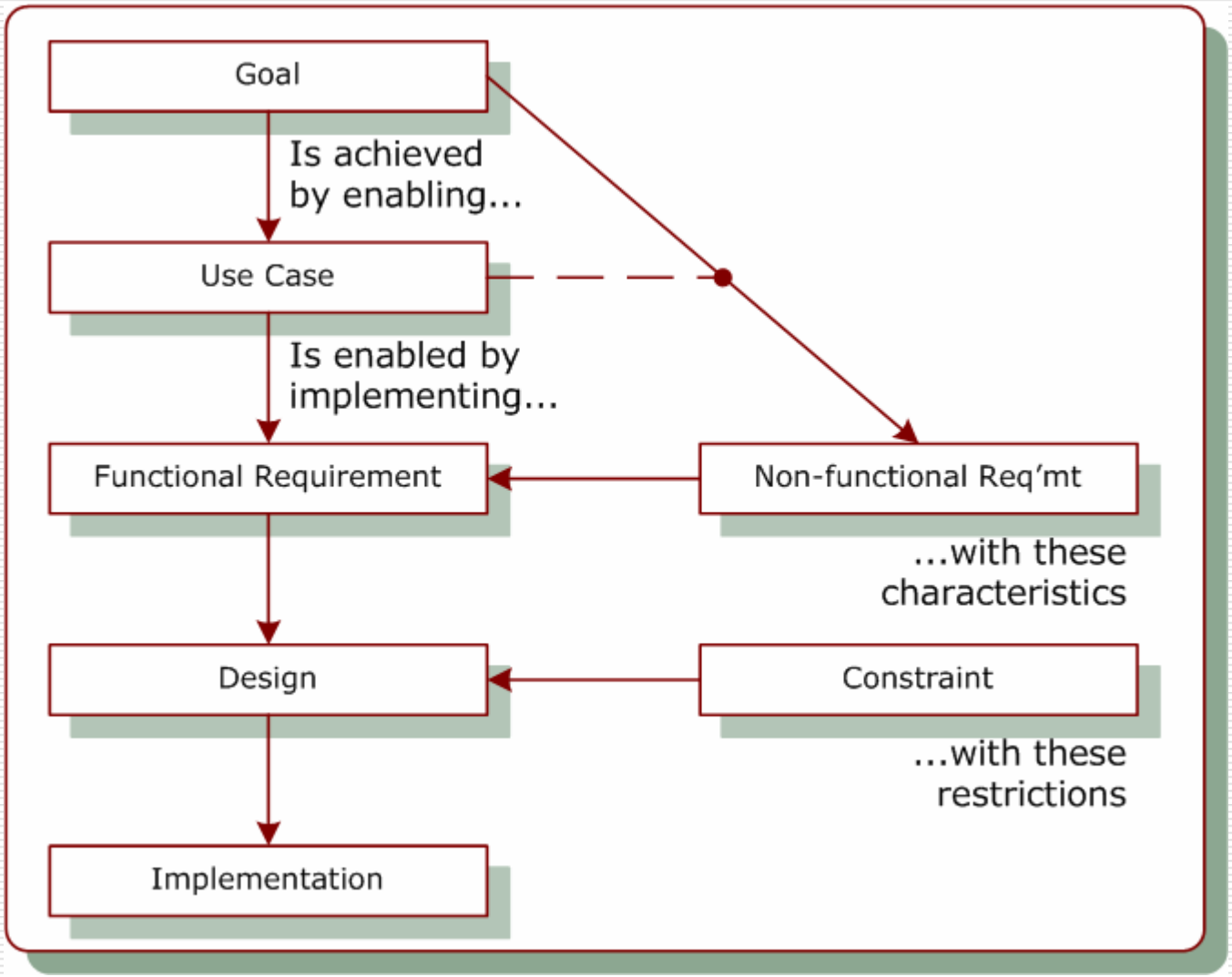
2. Structuring Requirements Information
   - How do we categorize requirements?
   - What is the hierarchy of requirements?

3. Managing the Requirements
   - How do we gather, maintain and use requirements?

Structured Requirements

- Goals
  - What is the ROI? The objective?
- Use Cases
  - How will users interact with the system?
- Functional requirements
  - What, precisely must the system do?
- Non-functional requirements
  - Characterize *how* software performs
  - Constraints (must use J2EE)
    - Limits design flexibility in exchange for conformance

Types of Requirements



Goals

- Stakeholders fund projects to achieve a specific objective
- Increased Profits
  - Increased Profits from cost reductions
  - Increased Profits from increased revenue
  - Increased Profits from improved branding
  - Increased Profits from increased user adoption
- There's a theme

Use cases – Informal

- Narrative description of what the user can do with the system (from the user's perspective) and some detail about how it is accomplished.
  - After logging in, the user can edit his personal profile information. The user will be able to view all of his changes prior to committing them or canceling the changes. The system will provide feedback that the changes were saved or cancelled, as appropriate.
  - The user can create a hot-sheet showing the products with the most sales for a specified period. The user can specify the period of interest, the product areas she is interested in, and the minimum dollar amount to use to limit the length of the report. The system will present the results of all products meeting the specified criteria.

Use cases - Formal

- Detailed step-by-step descriptions of how objectives are achieved.
  - Preconditions – Required state to initiate
  - Normal course – The most common sequence
  - Alternate courses – More than one way to do it
  - Exception courses – What happens when something goes wrong or is unexpected (error handling, for example)
  - Post-conditions – "Contract" for results of completion

Functional requirements

- Defining precisely *what* is to be built
- Primary consumers are developers and testers.
- Precise about behavior, ambiguous about implementation choices

Non-functional requirements

- Defining precisely *how* the application must perform
  - Must use server's clock when time-stamping transactions
  - Must support internationalization
  - Batch processing must complete in <8 hrs
  - Search must return some results within 3 seconds, all results within 20 seconds

Constraints

- Specify the environment or implementation details that restrict *design* choices
  - Must use an existing Oracle database
  - Must run on Linux machines
  - Must be XHTML only
  - Must be Managed code (.NET)
  - Must meet section 508 Accessibility standards

Glossary of Terms

- Provides a *Rosetta Stone* of domain-specific and esoteric knowledge.
  - Definitions of financial terms
    - (e.g. Net30, EBITDA)
  - Definitions of business structures
    - (e.g. divisions, product lines, fiscal year)
  - Definitions of engineering terms
    - (e.g. Erlang calculations, QoS, load balancing)
- Enables efficient, unambiguous communication from stakeholders and SMEs to other team members

Decomposition

- Solely an organizational technique for combining requirements
- X aggregates X1, X2, X3
  - User Account Management (X)
    - Create new account (X1)
    - Modify existing account (X2)
    - Delete existing account (X3)
- Priority of sub-requirements may vary.
- Delivery schedule may vary.

Outline

1. Overview of Requirements Management
   - Who is involved?
   - How do we approach the space?
   - What outputs do we create?

2. Structuring Requirements Information
   - How do we categorize requirements?
   - What is the hierarchy of requirements?

3. Managing the Requirements
   - How do we gather, maintain and use requirements?

Elicitation

- Verification of documented requirements
  - Active Listening is the key
  - Initial documentation is iterative
    - Synthesis of inputs from stakeholders
    - Rigorous exploration of the domain
  - Document what *won't* be implemented

- Iteration
  - Requirements *will* change once the stakeholder can visualize the solution
    - Show screenshots, mockups, prototypes
    - Validate interpretation of requirements

Validation

- Validation of interpretation of requirement
  - Use traceability to assure that supporting requirements will achieve parent-objective
- Happens at all levels within requirements
  - Will we achieve 10% growth if we allow users to do A,B and C but not D?
  - If we implement L,M,N, can user do B?
- Implementation
  - Will this design meet the specifics of N?
- Testing
  - Will testing X,Y,Z assure that L is properly implemented?

Baselining

- On date X, stakeholders / sponsors / customers agree – these *are* the requirements
- Any changes after date X have ripple effects
  - Must validate proposed changes in context of non-functional requirements
  - Must revalidate changed requirements
  - Must re-evaluate implementation design
  - Must revalidate scoping and schedule of implementation and testing
  - Must revalidate schedule of delivery
- Use traceability to manage the propagation, identify the ripples and dependencies

Traceability

- Goals are supported by use cases
- Use cases are supported by functional requirements
- Functional requirements may be supported by other functional requirements
- Non-functional requirements *affect* functional requirements

Scheduling

- Different things for different people
  - Stakeholders need to know when they can achieve X (ROI, product launch, etc)
  - Users want to know when they can perform action Y
  - Developers need to know when functional requirement Z is needed
- Create different documents
  - Product roadmap showing areas of functionality
  - Release schedule showing when use cases are enabled
  - Delivery schedule showing when each functional requirement will be delivered
- Requires coordination with development / project managers responsible for implementation and testing

Management systems

- Enterprise Solutions
  - Planning / Strategic
    - FeaturePlan
  - Execution / Management
    - Borland CaliberRM
    - Telelogic DOORS
    - IBM (Rational) RequisitePro
- Home-grown
  - MS Office, +emails, IMs + post-it notes
  - Wiki
  - Wiki + Bugtracking
  - Focus on Prototypes instead of docs (iRise)

Good references

- Karl Wiegers
  - http://www.processimpact.com/
  - http://www.processimpact.com/goodies.shtml#reqs
  - Software Requirements, 2nd Edition
- Blogs I like about requirements
  - http://tynerblain.com/blog
  - http://requirements.seilevel.com/blog/atom.xml
  - http://feeds.feedburner.com/MichaelHighTechPM
  - http://feeds.feedburner.com/FromStartToEnd

Thank you

- Thanks very much for the time you spent with me today. If you have any questions about the content, or about applying this stuff in the real world, please don't hesitate to contact me

- **scott@tynerblain.com**