Project and System Level REQUIREMENTS Specification

Main Outline:

• How to communicate your organisation's 'real' requirements, and your customer's most critical improvement requirements, in an unambiguous, clear, measurable, and testable way.

• a complete method for tackling all the critical and real stakeholder requirements for a project, at all levels of consideration for IT Projects.

Background:

In 2011 several THE BANK IT projects were examined with regard to their most critical top level requirements – the main value delivery improvements to The Bank . In no cases were the top level requirements satisfactorily specified. The top IT management conclusion was that all of our projects need much clearer, quantified, specification of these critical few project objectives. "It is so simple and obvious. Why don't we do it?"

We believe that the most critical requirements are few in quantity, but that it is critical to specify them extremely well, and to get them 'right'. Then we need to adjust them as value delivery experience dictates – and to then adjust them to optimize ability to exploit limited remaining project resources (budget, deadline).

THE BANK has published a standard for doing this, 'Evo', which includes a unique method for requirement specification called 'Planguage' (Planning Language). It is not only strong as a requirements specification language, but also strong because it is intimately coupled to the corresponding design, quality control, and project management disciplines. It will help direct the underlying development processes (Agile or not) to focus on delivering what The Bank values and really expects for its project investment.

The requirement specification method in Planguage is particularly strong in capturing the top-level critical requirements – performance and quality requirements - 'quantitatively'.

Methods of 'fainter heart', focus on 'functional requirements' (and their cousins, 'use cases') and in fact simply specify (poor) *design* (like 'password') when they are completely missing the main point (in this example the specification of the degrees of 'security' they need, the *real* requirement).

Planguage is able to *integrate* multiple performance requirements, together with multiple resource budgets, together with multiple constraints. 'Integrate' means to be able to determine the current priority of *any* requirement at *any* time in the project – so that the project gets the greatest total *value delivered* for a given set of resources.

Planguage with Evo is able to keep intimate track of relationships between all requirements, all levels of requirements, and their corresponding designs and stakeholders. Planguage is able to also keep track of all types of risks, dependencies, assumptions and uncertainties – both in requirements and in the related designs and project steps for delivering the requirements. Planguage is able to make sure that Agile software projects are kept focused on delivering value to The Bank , not just code to testers.

Planguage is based on well-defined rules for specification, on numeric process entry and exit conditions, on well-defined engineering processes and well-defined concepts (655 glossary concepts are formally defined).

Planguage has proven suitable in practice for all kinds of IT banking projects, for small to extremely large. Planguage will be a powerful addition to your current development standards, and to the necessary toolkit of a business analyst.

Summary of Course Advantages:

• This is the most *advanced* and *comprehensive* course on requirements specification in the world.

• this is the course for those of you who know you must have the *state-of-the –art* requirements methods, because you run *big projects* and cannot afford sloppy requirements to threaten their success.

• This requirements method is also distinguished from others by its ability to *integrate* multiple quantified *quality* and cost requirements, with functions and constraints.

• This method permits and encourages *detailed* specification of requirements, not just a simple ambiguous and vague statement.

• We encourage the detailed *background* specifications (not just the *core* requirement itself) about a requirement, so that the requirement specification can serve *many necessary purposes* economically such as *risk management, prioritization, estimation, design, quality control, and project management*

• This course is based on the extremely *well-defined* methods in the 'Competitive Engineering' book. You get 'ready made' standards and processes, if you want.
• These methods are spreading, as a standard, in leading financial institutions.

Course Contents:

• practical examples of Planguage for requirements
• the various requirements concepts defined *deeply* and exemplified
• requirements templates (to make standards practical)
• design constraint templates (a type of required design or architecture)
• how to quantify *any* qualitative requirement *(like intuitiveness or adaptability or security) – this is the key ability that most all other 'requirements' courses do not teach!*
• advanced scale of measure specification methods *(a 'scale' is more than units)*
• how to measure a requirement level numerically *(meters and tests for quality)*
• standards for requirements (rules, processes, templates, glossary)
• principles for requirements *(help you to tackle new problems better)*
• quality control of requirements: *measuring* requirement conformance to standards *(reviews, inspections, agile reviews)*
• estimating the quantified impact of a design on requirements
• evolutionary project management and how it integrates with requirements
• training requirements writers
• changing requirements culture
• expected results from requirements culture improvement
• a policy for improved requirements
• instructor-led workshop: participant input requirement problems solved by Gilb
• how to give information that determines priorities of requirements *(example Wish/Goal/Fail and Qualifiers)*
• how to include requirement information about risks and uncertainties
• how to include requirement information about traceability (up and down)
• overview: how Evo and Planguage relate to Agile development (Scrum and XP Practices)
• overview: the artifacts and roles of Evo and Planguage
• overview: the objectives of this training course

Course Objectives:
• this course will allow you to walk away with practical ability to improve the most critical project requirements
• we hope that participants will walk away with an independent ability to *evaluate* and select subsets of these methods for their professional purposes
• we hope that most participants will *choose to study* these methods further after the course (for example using the 'Competitive Engineering' Handbook, and many other papers and cases we make available)
• we hope that participants will choose to adopt these methods in their work, quickly in current projects

Course Limitations: (there is no silver bullet!)
• You will not become an overnight *expert*. You *will* become a determined student, evangelist and practitioner of these exciting methods. You will be given a certification of course attendance, with an option to progress to Practitioner, and to Maestro levels.
• the participant will only become well qualified with months of practice, back at work (as with most courses)
• consequently, many of the varied course lecture subjects, that should *ideally* each have many times more student time allocated, for proper training, will be 'exposed' (and *well* documented) – but we cannot take the time to exercise, give feedback, and even to lecture in the detail we would like to, if we had time. This is a subject for later coaching and personal study, which we can offer.
• we will prioritize discussion, in class, and in breaks, around questions that help *evaluate* the methods, and *compare* the method with other practices

Intended for:
• people who write requirements (Bas), and their managers
• project managers and their managers
• consultants, engineering/IT methods teachers

Bio:
Tom Gilb and Kai Gilb have developed the requirements methods they teach, personally, with the help of many professional friends and clients. The methods have been developed over decades of practice all over the world in both small companies and projects and the largest companies and projects.

Tom is the author of nine books and hundreds of papers on these and related subjects. His 9th book 'Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage' is a substantial definition of these requirements ideas – and full electronic copies will be given to course participants.
His ideas on requirements are the acknowledged basis for CMMI level 4 (quantification, as initially developed at IBM from 1980).
Tom has guest lectured at universities all over UK, Europe, China, India, USA, Korea – and has been a keynote speaker at dozens of technical conferences internationally. He is not 'academic' – but is intensely practical. There are very many organizations and individuals who use some of his requirement methods, IBM and HP were two early corporate adopters. As of October 2011 over 20,000 engineers at Intel have been voluntarily trained in the 'Planguage' requirements methods – and growing.

Kai Gilb has partnered with Tom in development of these ideas and practicing them at clients for over 15 years. He has written his own book (see Evo at gilb.com), and 2 electronic tools to support the requirements ideas. He will co-teach and be available with Tom for informal discussions about the methods.

A fair sample of Gilb papers and slides will be found for free download at their website www.gilb.com. We invite you to convince yourself that these teachings are indeed the most advanced and powerful on the market.

Here are the course modules in a suggested timing. Each module is about 1 hour of the classroom day. We assume classroom day is 9 to 5PM

Day 1 *(for a more detailed view see later in this brochure)*

- 0. Overview: Evo & Planguage in relation to Agile Methods
- 1. practical examples of Planguage for requirements (case studies)
- 2. the various requirements concepts defined *deeply* and exemplified
- 3. requirements templates (to make standards practical)
    - design constraint templates (a type of required design or architecture)
- 4. how to quantify any qualitative requirement (like intuitiveness or adaptability or security) – this is the key ability that most all other 'requirements' courses do not teach!
- 5. advanced scale of measure specification methods *(a 'scale' is more than units)*
- 6. how to measure a requirement level numerically *(meters and tests for quality)*

Day 2

- 0. Tips for analyzing project plans to find the 'real' value requirements.
- 1. standards for requirements (rules, processes, templates, glossary)
- 2. principles for requirements (help you to tackle new problems better)
- 3. quality control of requirements: *measuring* requirement conformance to standards *(reviews, inspections, agile reviews)*
- 4. how to give information that determines priorities of requirements (example Wish/Goal/Fail and Qualifiers)
- 5. how to include requirement information about risks and uncertainties
- 6. how to include requirement information about traceability (up and down)

•

Day 3

- 1. estimating the quantified impact of a design on requirements
- 2. evolutionary project management and how it integrates with requirements. The Evo cycle and how it relates to Agile iteration.
- 3. training requirements writers: how to train colleagues and yourself

• 4. changing requirements culture: how to change your culture of requirements
• 5. expected results from requirements culture improvement: how to measure or know that things are working well
• 6. a policy for improved requirements: summary of main guidelines for value driven projects, and value requirements.
• 7. instructor-led workshop: participant input requirement problems solved by Gilb
•

Teaching Process:
1. Lectures with slides to present ideas (50%)
    - Basic Theory (Principles, Standards, Rules, Templates)
    - Case studies (as far as possible from THE BANK and banking)
    - Examples of practice (as far as possible from THE BANK and banking)
2. Questions and discussion
3. Participant exercises (small groups 2 to 4), followed up by Instructors, and experienced THE BANK assistants (if available)
4. Substantial digital documentation, a library of books, papers, cases

Option: an examination, theory and practical

Handouts

14/05/2012 19:57

Main Course slides:

- Top Level Requirements (this course)

Books:

Competitive Engineering

- And Full Glossary
- Chapter by Chapter Version for convenience

Kai Gilb, **Evo** book

Slide sets

- Top Level Objectives Requirements Cases (8 real cases)
- Intel Planguage Slides
- More!

Papers on Requirements (by Gilb)

- Spec QC (2009 Testing Experience)
- Quantifying Security
- Managing Priorities
- Rich Requirements Specs
- Real Requirements
- Quantifying Stakeholder Values
- Managing Project Risks in Requirements, Design and Development using Planguage
- Requirements for Outsourcing
- Requirement Relationships
- Making Metrics More Practical in Systems Engineering: Fundamental Principles for Failure and for Success
- Value Delivery in Systems Engineering
- Estimation: A Paradigm Shift Toward Dynamic Design-to-Cost and Radical Management (as published SQA 2011)
- What's fundamentally wrong? Improving our approach towards capturing value in requirements specification
- Quantifying Management Bullshit: forcing IT Stakeholders to reveal the value they really want from your IT Project. (Core 2010)
- User Stories: A Skeptical View (2010 Agile Record)
- Agile Aspects of Planguage for Cost-Effective Engineering (Agile Record Version, January 2011)

-

Case Studies (used as appropriate in teaching slides)

- Confirmit: Handout papers and slides
- Anonymous Bank Examples (D J C S F L)
- Aircraft
- Electronics (T, E, As, Av)
- Oil Business
- Mobile Phone Business (N T)
- Shipping (U)
- Healthcare
- Medicine (S)
- Defence
- Top Level Objectives Requirements Cases (8 real cases)
-

Other Peoples Papers and slides

- Confirmit: Handout papers and slides
- Intel Planguage sides, Eric Simmons
-

Standards:

- Templates
- Full Glossary
- Rules
- Principles
- Processes

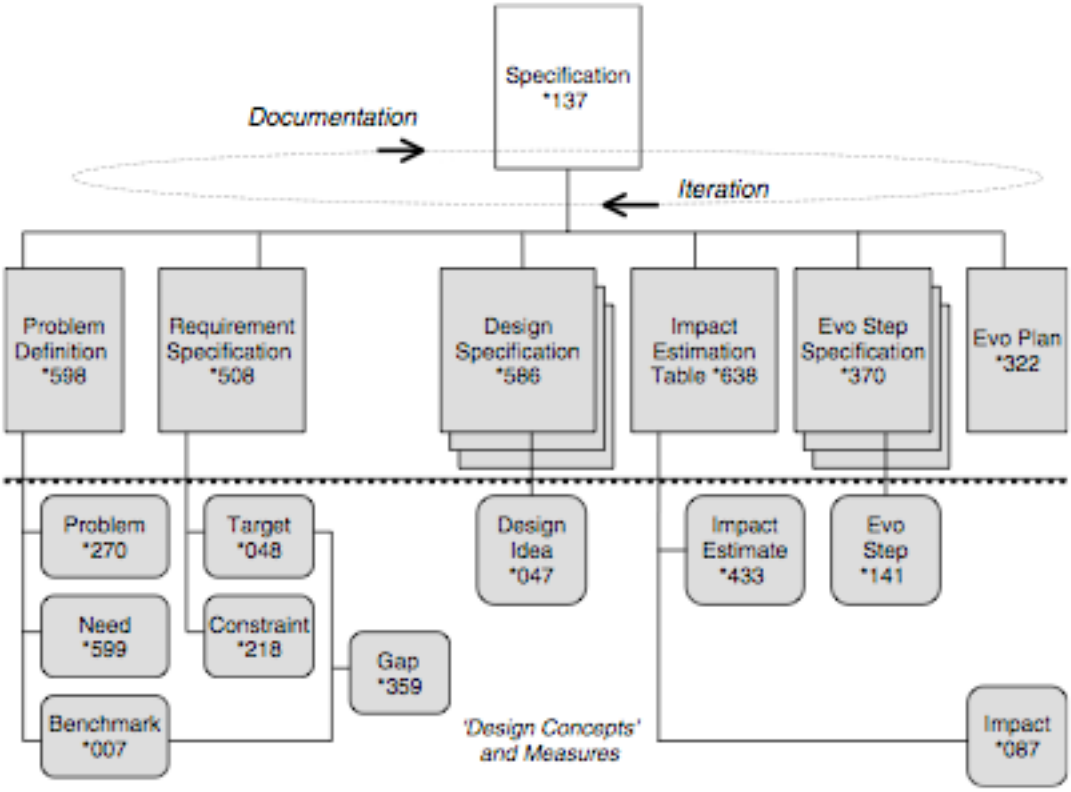
Richard Smith Citigroup 2011

- http://rsbatechnology.co.uk/blog:8
- Back in 2004, I was employed by a large investment bank in their FX e-commerce IT department as a business analyst.
- The wider IT organisation used a complex waterfall-based project methodology that required use of an intranet application to manage and report progress.
- However, it's main failings were that it almost **totally missed the ability to track delivery of actual value improvements to a project's stakeholders**, and **the ability to react to changes in requirements and priority for the project's duration.**
- The toolset generated lots of charts and stats that provided **the illusion of risk control**. but actually provided very little help to the analysts, developers and testers actually doing the work at the coal face.
- The proof is in the pudding;
- I have **used Evo** (albeit in disguise sometimes) on two large, high-risk projects in front-office investment banking businesses, and several smaller tasks.
- On the largest critical project, the original business functions & performance objective **requirements document, which included no design, essentially remained unchanged** over the 14 months the project took to deliver,
- but the detailed **designs** (of the GUI, business logic, performance characteristics) **changed** many many times, guided by lessons learnt and **feedback** gained by delivering a succession of early deliveries to real users.
- In the end, the new system responsible for 10s of USD billions of notional risk, **successfully went live over over one weekend for 800 users worldwide**, and **was seen as a big success by the sponsoring stakeholders.**
- " I attended a 3-day course with you and Kai whilst at Citigroup in 2006"

Day 1

- 0. **Overview**: Evo & Planguage in relation to Agile Methods
    - Evo roles: Product Owner, Project Manager, Agile Developer, Designer, Architect
    - Planguage Tools: Templates, Rules, Process, Principles, Glossary definitions, Software support, books, papers, slides, cases.
    - The Evo cycle; stakeholders, value, solutions, decompose, develop, deliver, measure, learn)
    - Test question: how is Evo different from Scrum?
    - Test Question: How is Planguage different from Uses Cases or User Stories?
    - 

- 1. practical examples of Planguage for requirements (case studies)
    - Before and after examples of top level requirements
    - Where to find the hidden requirements
    - When are critical requirement 'implied' not stated directly
    - Case studies from our own organization (if available)
    - Exercise: translate one nice sounding requirement to a defined scale of measure



- 2. the various requirements concepts defined *deeply* and exemplified
  - Requirement: stakeholder valued future state
  - The 20 or so different classes of Requirement
  - Value requirements defined
  - Test question: list *non* value requirements (types, or actual case)

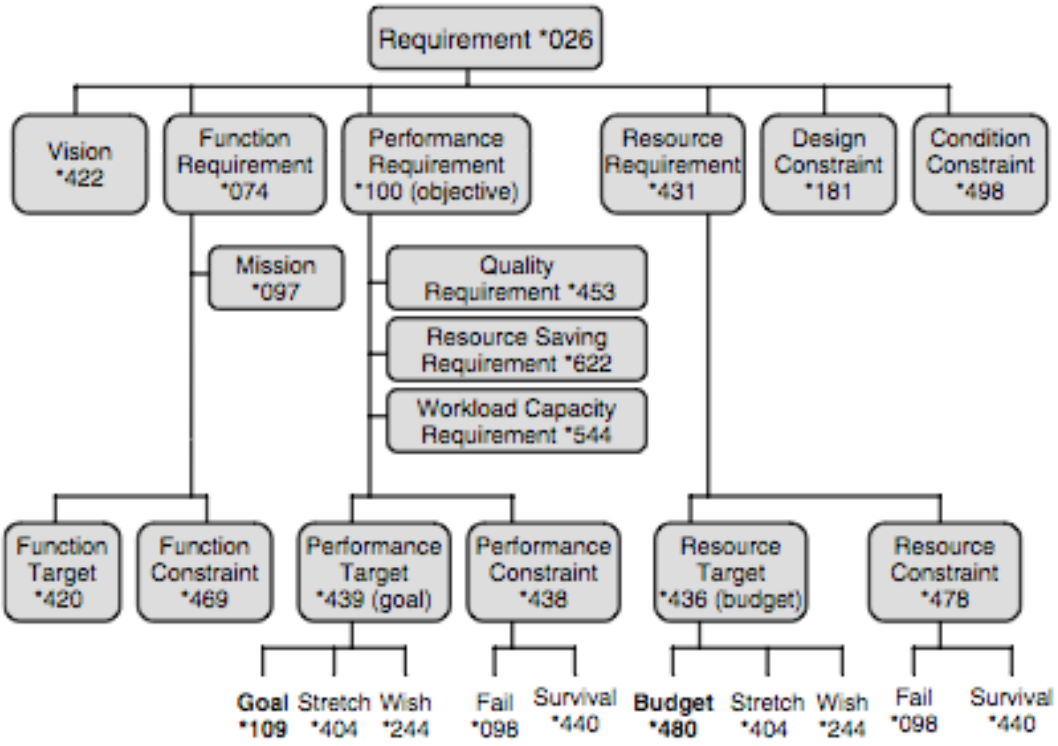


Figure G20
Requirement Concepts.

- 3. requirements templates (to make standards practical)
  - Simple Templates, Template types, Full templates
  - Templates with hints
  - Templates relation to Rules
  - Design constraint templates (a type of required design or architecture)
  - Exercise: suggest at least 4 additions to a basic template, justify.
- 4. how to quantify *any* qualitative requirement (up to 2 hours)
  - (like intuitiveness or adaptability or security) –
  - this is the key ability that most all other 'requirements' courses do not teach!
  - The uses of and differences between Tag, Ambition, Scale, Meter
  - The Value Levels: Benchmarks, Targets, Constraints
  - Fundamental Levels: Past, Goal
  - Advanced Value Scale Levels, Wish, Ideal, Record, Stretch, Tolerable/Fail/Survival

- Qualifiers: how to express the where, who, when if 'conditions' of any level
- Sources: direct info about who or what decides the levels
- Uncertainty notation: ±, 3 -> 6, ?, ??
- Exercise: write one spec using all tools, from a template.

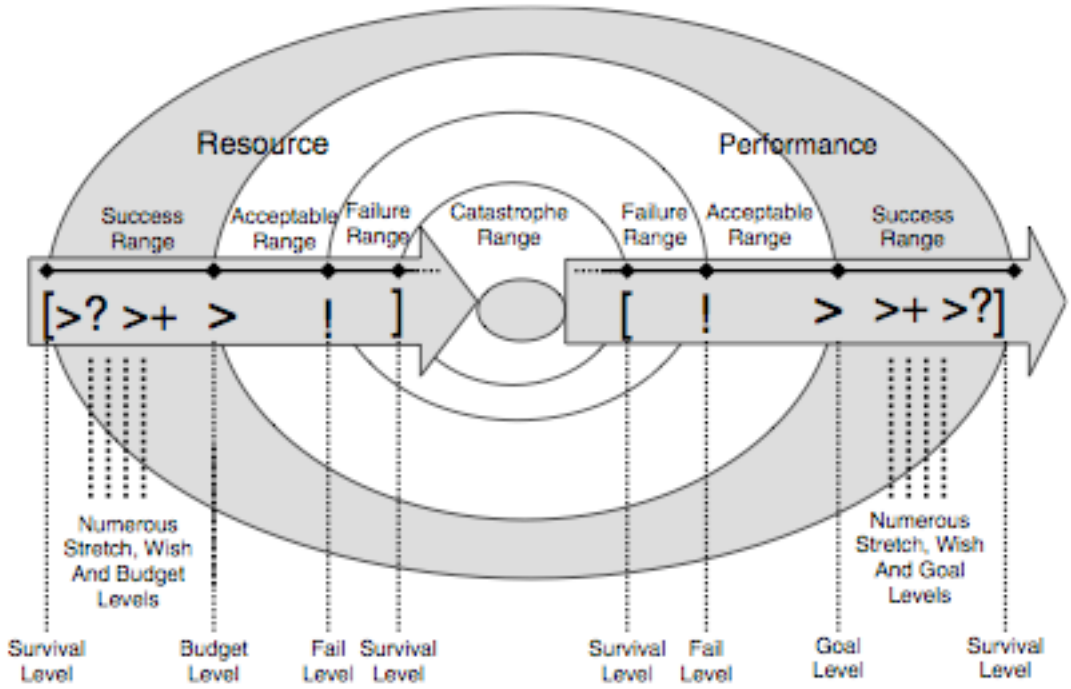
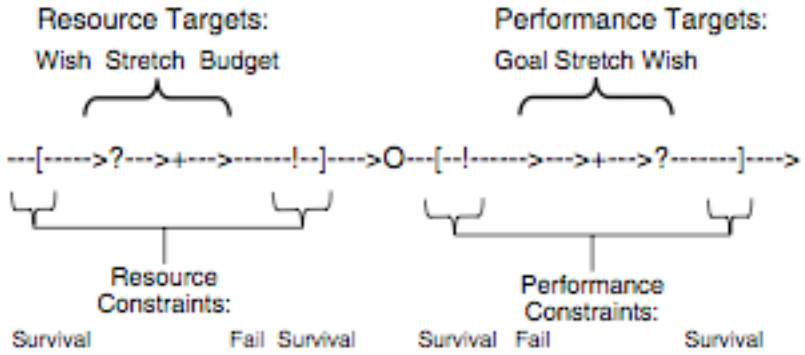


Figure G19
The 'doughnut' diagram indicating different range concepts.

- 5. advanced scale of measure specification methods *(a 'scale' is more than units) less than 30 minutes*
  - The components of a Scale of measure; Units, Relations, Background Environment
  - Scale Parameters: a reusable scale tool. A method for forcing us to consider all real cases.
  - Exercise: define 2 Scales with 3 parameters or more

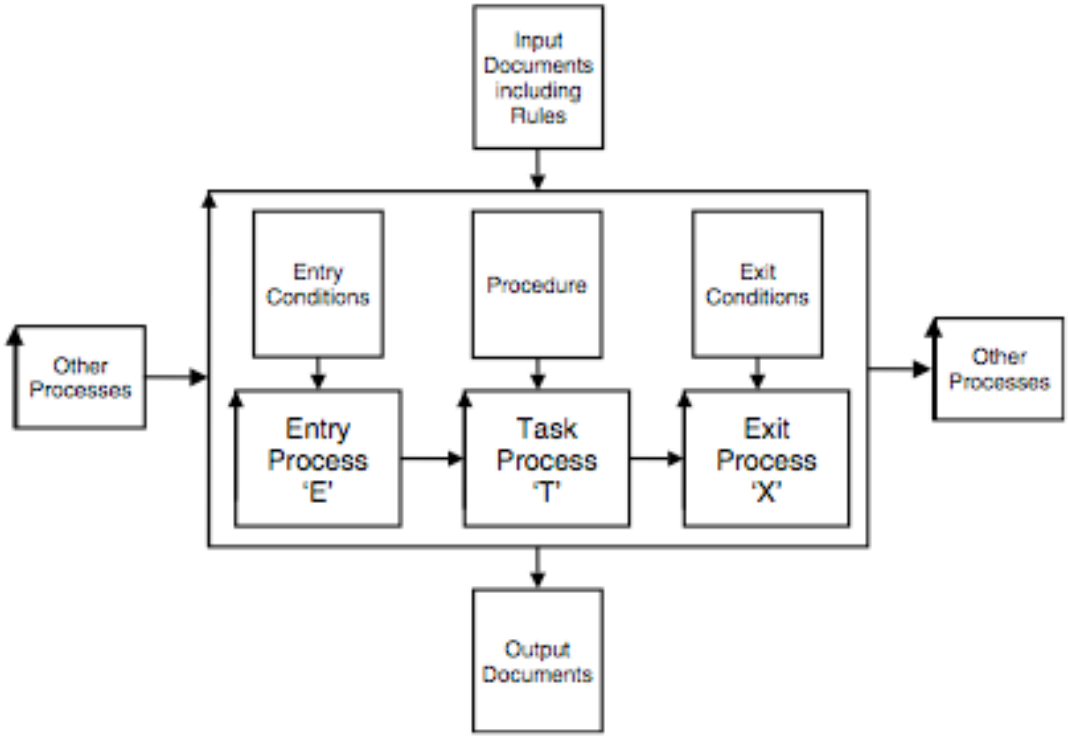


- 6. how to measure a requirement level numerically *(meters and tests for quality)*
    - The possibility of zero to several different Meters (Test processes to measure along the defined scale)
    - How to find practical Meters
    - The idea of sketching Meters roughly, for later test process planning
    - Understanding the cost and value attributes of meters
    - Designing a meter to fit your needs
    - Exercise: define 2 meters for a given Scale definition

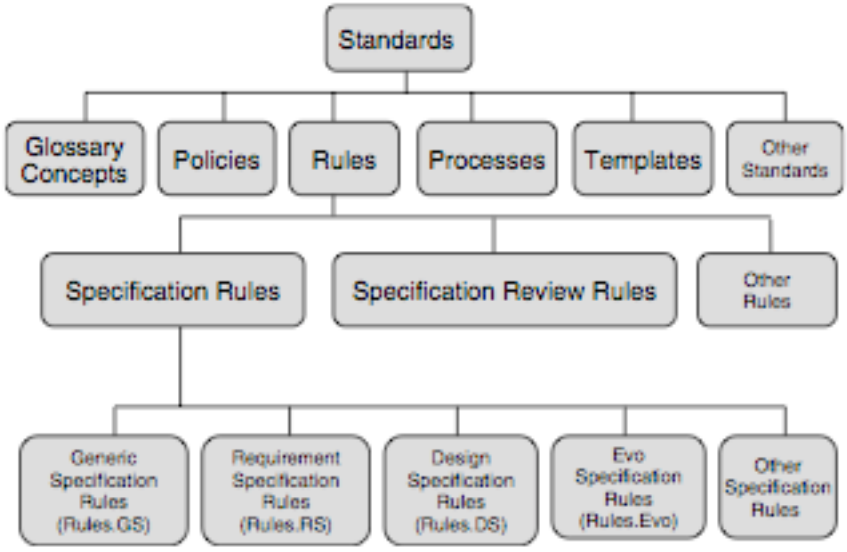
Day 2

- 0. Tips for analyzing raw project plans to find the 'real' value requirements.
    - Classes of source documents (slides, meeting notes, formal plans, higher level plans than your project)
    - Key words to look for (Objectives, Critical, Improved, Reduced, etc)
    - Link Words (by, through, in order to, using): Ends+Means
    - Levels of concerns (Keeney) : Fundamental, Strategic, Means Objectives
    - Deriving main and secondary objectives from stated 'means'
    - Confirming your requirements hypothesis with stakeholders
    - Documenting related requirements levels, and stakeholders in Planguage (Impacts, Impacted by, Design Suggestion, Stakeholders, Issues)
    - Exercise: draw a map of at least 4 levels of concern from an example (handout or your real case)
- 1. **standards for requirements** (rules, processes, templates, glossary, Exit Levels)

- Specification **Rules**: the Planguage set (look into the CE Handbook), A Gilb private collection of rules for your tool library , tailored rules, the concept of a defect in specs, Your Organisation's Rules to date, Rules for Rules (max 1 page!)
- **Processes**: The Planguage Set in CE Handbook, a Gilb private collection of Processes, rules for processes, Process Structure (EPVX). Distinction between process and rules.
- **Entry/Exit levels**: the defect density accepted for beginning a process or completing one.



- **Templates**: forms to fill out specifications, with hints, tailors, partly automated, very automated and integrated
- **Glossary**: The official Planguage Glossary, tailoring to local use, adding concepts and adding terms.
- Exercise: quiz: 10 examples, you tell us what is the type of standard.



- 2. principles for requirements (help you to tackle new problems better)
    - General Principles for Requirements
    - Principles for Quality Requirements
    - Principles for Design Constraints
    - Principles for Agile Projects
    - Exercise: Which 10 principles should a Business Analyst respect?
- 3. **quality control** of requirements: *measuring* requirement conformance to standards *(reviews, inspections, agile reviews)*
    - *Classes of reviews (Inspection, Pair programming)*
    - *Agile Spec QC: for requirements at BA level*
    - *A Case study (as instruction to exercise)*
    - *Exercise: measure major defects in a set of requirements.*
- 4. how to give information that determines **priorities** of requirements (example Wish/Goal/Fail and Qualifiers)
    - Dynamic Prioritization: the natural principle for sleeping, eating, breathing priority.
    - Computing current priorities (red, green, yellow signals)
    - Priority information in Planguage statements (level, conditions)
    - Worst practices of prioritization: weights, declarations by management
    - Exercise: list 10 Planguage elements that give information about relative priority of a requirement
- 5. how to include requirement information about **risks** and uncertainties

- why we need to integrate risk information in requirements
- Planguage as a rich language for capturing info about risks
- Planguage Risk Tactics: notes, relations, sources, assertions, ranges, qualifiers, Spec QC vs Rules
- Planguage details: analyze all Planguage parameters as risk tools.
- Impact Estimation Tables as a risk mapping tactic: Credibility Index, Credibility and uncertainty prioritization.

• 6. how to include requirement information about **traceability** (up and down)

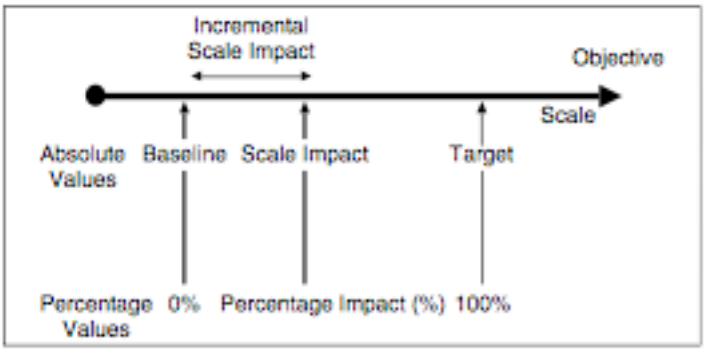
- The catalog of Planguage parameters that show relationships and allow traceability and auditability
- Qualifiers for relating to groups, places and conditions.
- Multiple Levels of Impact Estimation Tables (Org., stakeholders, IT system) showing cause and effects in a system

•

Day 3

• 1. **estimating** the quantified impact of a design on requirements

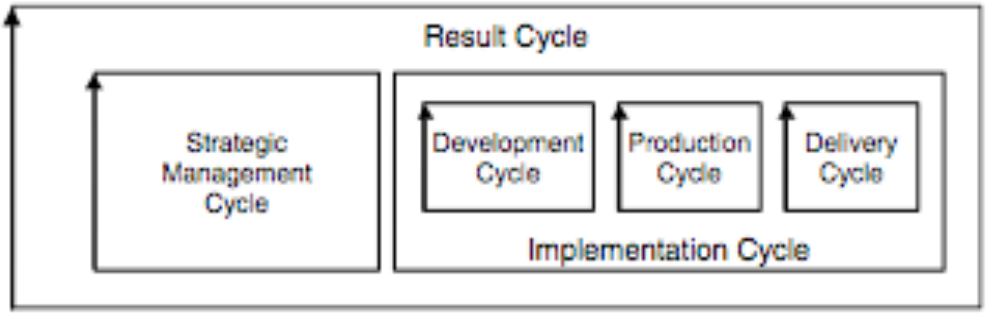
- a value requirement level is only valid if we can find design to meet it
- we need to estimate impacts on all values, not just one dimension like 'security'
- we need to make sure that acceptable designs have acceptable estimated costs n time and money
- how do we make an estimate when we know too little about suggested designs, strategies and architectures
- dynamic estimation, and dynamic measurement of value requirements.



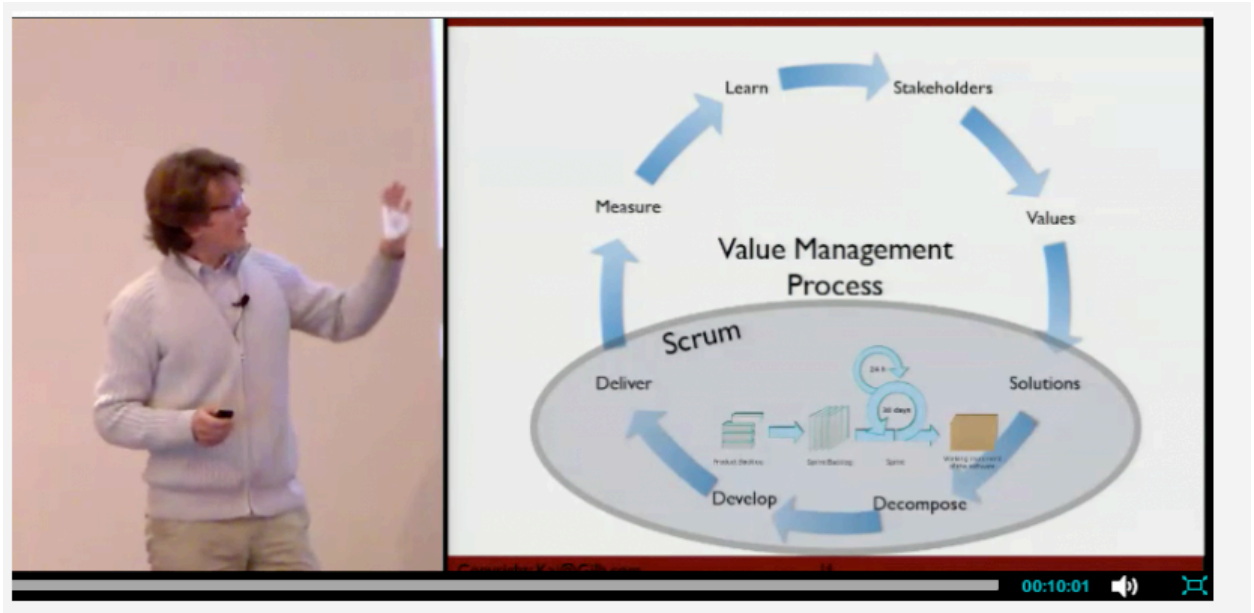
-

• 2. evolutionary project management and how it integrates with requirements. The **Evo cycle** and how it relates to Agile iteration.

- The stages of the Evo cycle, and relationship to Agile development
- The role of requirements in each stage of the cycle



-
-



-

- 3. training requirements writers: how to train colleagues and yourself
    - Self study opportunities
    - Learn by Experimentation: try specification variations yourself
    - Coaching opportunities
    - Learning by delivery and feedback
    - Consulting with stakeholders about their real needs
    -
- 4. **changing** requirements **culture**: how to change your culture of requirements
    - How to change requirement standards
    - How to experiment with better standards
    - How to prove that these methods serve The Bank
- 5. expected results from requirements culture improvement: how to measure or know that things are working well
    - Using Spec QC to measure requirement writers ability to follow standards, and to motivate them to learn the standards in practice
    - Using evolutionary delivery streams of value to stakeholders to 'sell' the method rapidly
    - Using Impact Estimation tables as a cooperative communication tool about requirements, designs, and priorities.
- 6. a policy for improved requirements: summary of main guidelines for value driven projects, and value requirements.
    - A Project Management Policy
    - A Requirements Policy
    - A Design and Architecture Policy
    - A Risk Policy
- 7. instructor-led workshop: participant input requirement problems solved by Gilb
    - We will take suggestions from the class, or a volunteer 'client' and demonstrate how an 'expert' interprets and specifies requirements.

8. Class Photo, Diplomas,