BTC-202

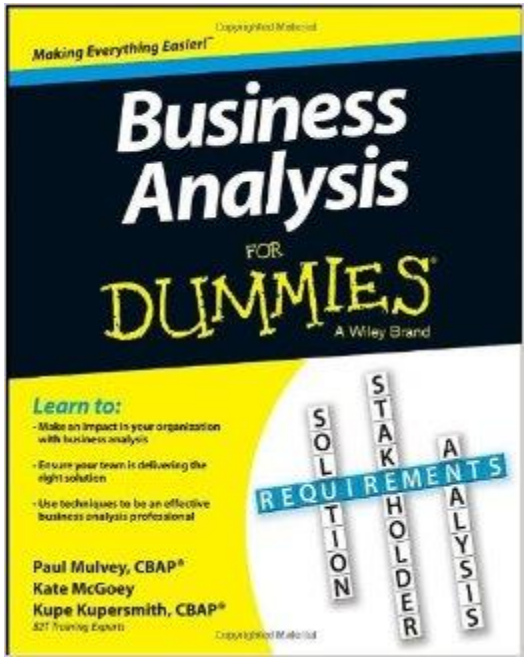
BUSINESS ANALYSIS FOR THE IT PROFESSIONAL

Courseware

- ***Business Analysis*** **-** Doug Hoff
- ***CBAP/CCBA Certified Business Analysis Study Guide*** – Susan Weese & Terri Wagner

Other books

- **Business Analysis For Dummies** by Kupe Kupersmith, Paul Mulvey, Kate McGoey. July 2013 – recommended
  - $13 used Amazon



Other books

- ***Writing Effective Use Cases*** by Alistair Cockburn, Addison-Wesley, 2001 – somewhat difficult but seminal work

Web sites

- **Course repo**
  - http://github.com/doughoff
- **Modern Analyst**
  - http://modernanalyst.com/

CBAP

- CBAP - Certified Business Analysis Professional is the title of a person who has passed the test on the BABoK
- The Business Analysis Body of Knowledge is written by the IIBA.
- IIBA Kansas City meets monthly
- Version 2.0
- This course is based on the BABOK documentation.

Intro to Business Analysis

What drives business?

- Value
- Value becomes goals for strategic value
  - Goals result in value for the **business, not the user**
- Value becomes scope for project value

What is analysis?

- To **understand** current business value
- To **understand** future business value
- To advise and **communicate** improvements
  - downstream – programmers, testers, stakeholders
  - to management

"I like things to be done my way but by somebody else."

How do we begin?

- Be honest
  - You will lie to yourself so you don't have to work
- Question everything for quality
  - Don't assume anything is the best

Scope is subject to change

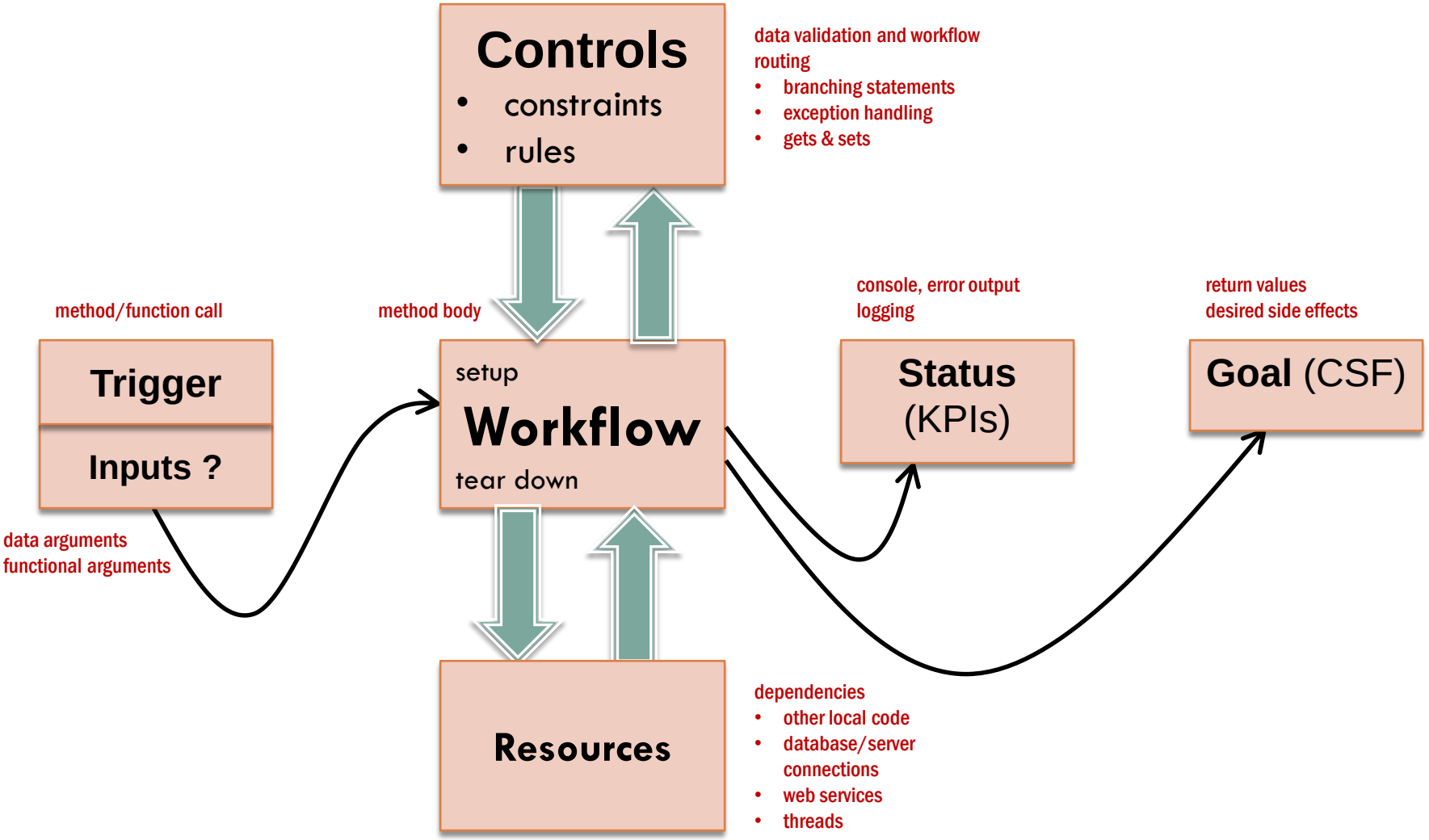
- Workflow and rules from
  - Changing market
  - Changing costs
  - Changing customers
  - Changing competition
  - Changing constraints
- Data usually stays the same
  - but changing constraints

Scope components

- Workflow / process
  - Role of initiator
  - trigger
  - steps
  - rules
  - relationships (cardinality, dependencies)
  - status checks, goal
- Data
  - inputs, outputs
  - descriptions
  - rules
  - relationships (cardinality, dependencies)

A process/service model

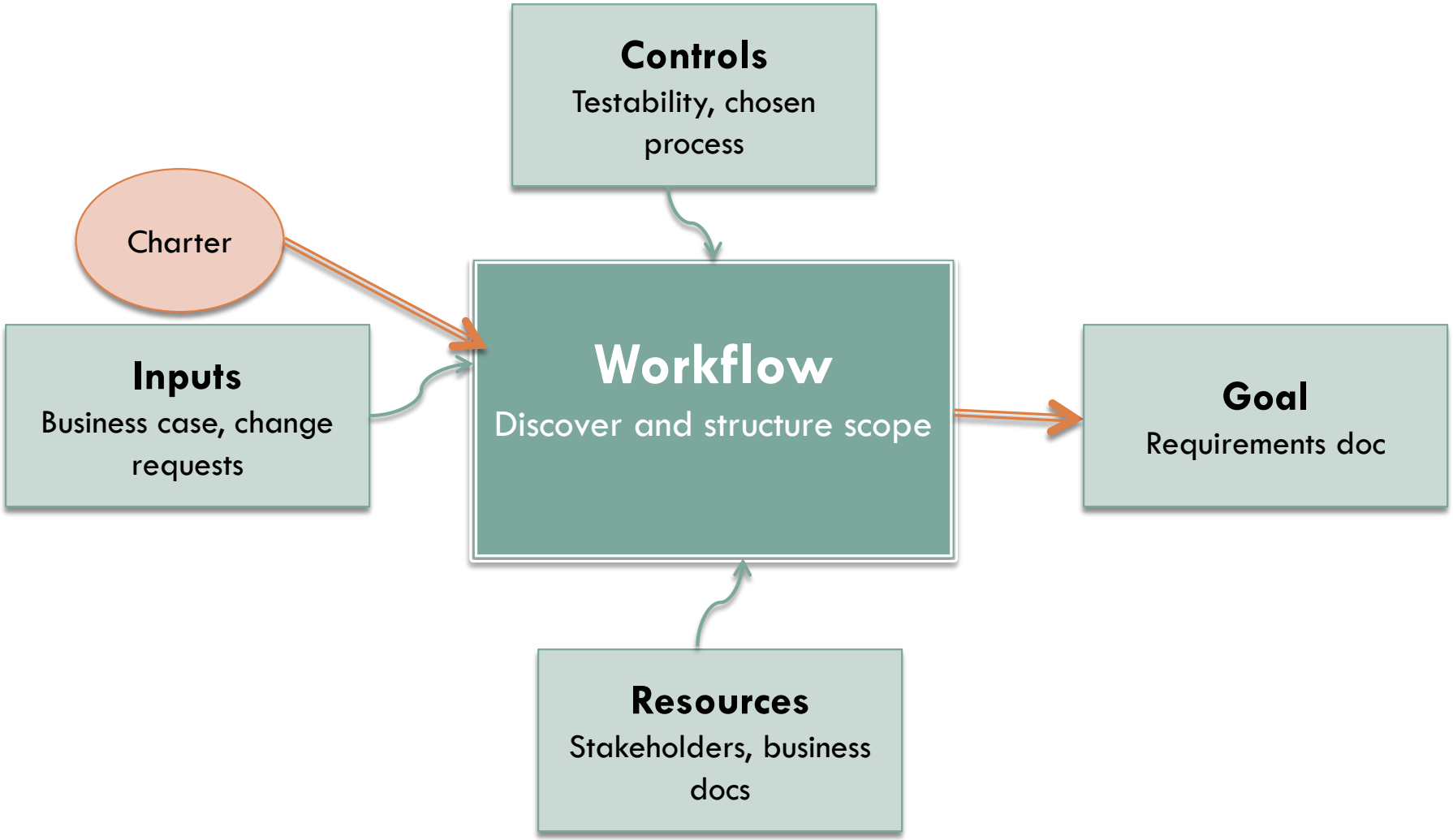
the process parts in computer language



Process maturity levels

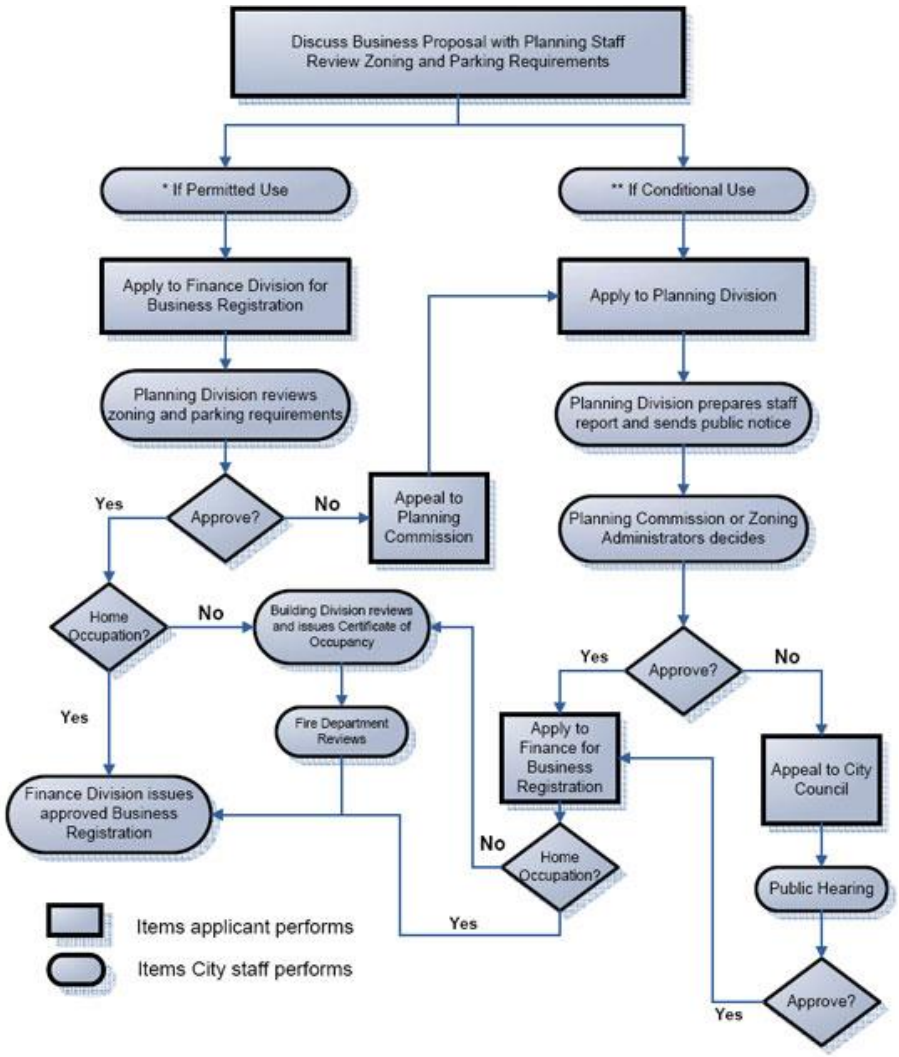
- 1 – individuals, firefighters, driven crazy
  - problems without solutions – 20%
- 2 – collaborators, cops, control driven
  - rules without standard process – 40%
- 3 – protocols, partners, process driven
  - standards without context – 20%
- 4 – optimizing, scientists, data driven
  - silos without working with people – <10%
- 5 – mastering, alphas, quality driven
  - people and relationships – <1%

BA process in project



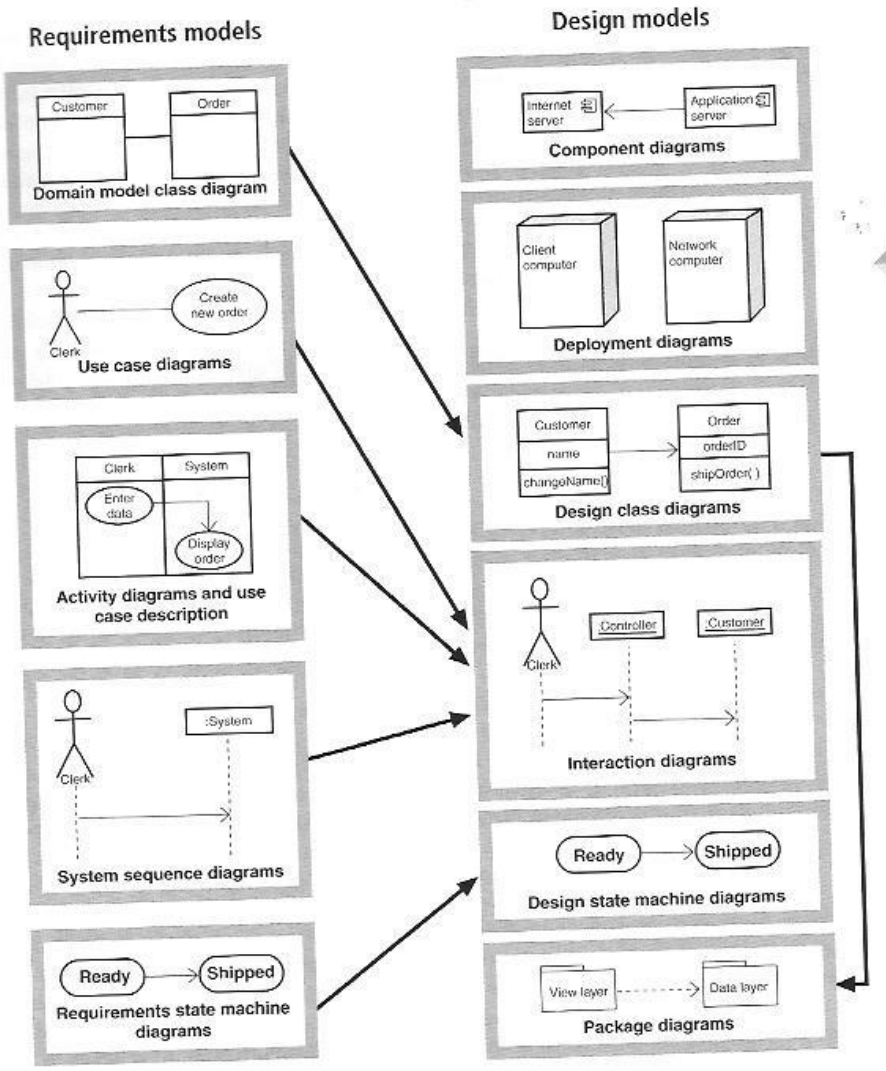
BA data in project, low detail

- Scope
  - Data – dictionary
  - Workflow – flowchart, informal use cases
- Structure
  - Requirements categories
  - System architecture



BA data in project, better detail

- Scope
  - Data – data dictionary, ERDs, class diagrams
  - Workflow – use case detail and diagrams
- Structure
  - success criteria for use cases
  - grouped use cases
  - includes/extends use cases
  - system level sequence



Requirements analysis cycle

- Model the current business state (baseline)
  - Select model types
  - Create, update and verify models
- Model the project's scope
  - Select model types
  - Create, update and verify models
  - Obtain agreement from sponsor and project manager

Requirements analysis cycle

- Model the project's scope in detail
  - Select model types
  - Create, refine and verify models
  - Structure detailed models to hide detail
- Prioritize requirements
  - At goal level
  - Agree on priority metrics
  - Apply metrics

User involvement

- Keep stakeholders involved
  - By any meeting possible
- Lack of involvement = possible lack of support

Problem identification

A problem well stated is half solved
- Charles Kettering

Intro

- Understand the problem / root causes
  - Gain agreement on the problem definition.
- Identify stakeholders and users.
- Identify the constraints to be imposed on the solution in the design.

Problem statement

- The sales pitch to sell the project
- The elevator pitch / marketing style:
  - for <customers> who have <reason> our <idea> so that <benefits> unlike <currently / competition>
- Pixar pitch
  - once upon a time… every day… one day… because of that… and … until finally …
- Focusing question
  - How can <we> do <idea> for <customer> so that <benefit>
- Twitter pitch
  - <idea> #<benefit> e.g. 1,000 songs in your pocket #Ipod

Problem statement

- the **standard format of problem writing** is
  1. “this <problem>…
  2. affects <all these people>…
  3. with <unhappy specific symptoms, actual effects, not causes> for the business...
  4. but our solution would benefit us by<better business results, overall improvements>…”
     - Don’t just negate 3. to get 4.

Exercise - problem statement

- One register with a long line is open at a busy grocery store.
- Oil stains are appearing in the garage on the concrete floor.
- I have less money to spend this year than I did last year.
- Competition and budget cuts are forcing us to eliminate valuable staff people.
- I lose too much time by driving to and from work.

Getting to the root causes

- Problem solving – look for actionable causes
  - What was it that created the effect we can do something about?
  - Finds missing or failed processes
- Requirement analysis – look for motives
  - What about your need helps the business?
  - Larger scope requirements
- Start low level, go up the chain
  - Ask for functional requirements.
  - Ask for related non-functional requirements.
  - Finally, ask for the business goal that should be at the root.

Five Whys

- By asking why five times and answering it each time, we can get to the real cause of the problem.
  - Taiichi Ohno, "Toyota Production System", 1988
  - Lean manufacturing
  - Origins of 5-S, Kanban, Kaizen, Lean, Poka-yoke, SMED, and 1-piece flow.
- A brainstorming technique that leads to a threshold of what you know.
  - Test assumption at the threshold and then brainstorm again.

Ishikawa's 7 basic quality tools

- **Cause-and-effect diagram** - Ishikawa or fishbone chart
- **Check sheet:** A structured, prepared form for collecting and analyzing data
- **Control charts:** Graphs used to study how a process changes over time.
- **Histogram:** Frequency distributions
- **Pareto chart:** Shows on a bar graph which factors are more significant
- **Scatter diagram:** Graphs pairs of numerical data
- **Stratification:** A technique that separates data gathered from a variety of sources so that patterns can be seen (some lists replace "stratification" with "flowchart" or "run chart").

- http://asq.org/learn-about-quality/seven-basic-quality-tools/overview/overview.html

Root requirements

- Requirements are often expressed as needs with a solution especially by non-technical people
- Asking **why** they want a solution will drive back to multiple real requirements
  - Functional
  - Non-functional
- It may be possible to ask **why** a few times to get to the real requirement
  - Technical constraints will be an intermediate stage

Exercise - Root requirements

- System identification (the car, the program, the meal, etc.)
- Functional requirements
  - Usually just one, maybe two (or none)
  - Says what the system will do – the car will …
  - A system button will cause this to occur

- Non-functional requirements
  - Many can be applied
  - Alter how the functional will work (adjectives, adverbs)
  - Says how the domain does things
- Need/solution/design = sum of requirements

Identify stakeholders and users

- Document them
- Questions to drive out details
  - Who are the users of the system?
  - Who is the customer (buyer) of the system?
  - Who else will be affected by the outputs of the system?
  - Who will evaluate and bless the system at delivery and deployment?
  - Who will maintain the system?
  - Are there other internal / external users with needs?

Identify constraints

- Constraints reduce scope **before** project starts
- **Project** constraints
  - Time affecting scope
  - Budget affecting scope
- **Scope - Analysis** / **Design** constraints
  - Technical – OS, database, training
    - Architectural – interfaces, data, processes
  - Cultural, political, legal
  - Environmental

Constraints

- **Risks** are constraints that are not 100% firm.
  - Project management details risks and how to mitigate or bypass negative risks
  - Most constraints are negotiable bounded by a system of values.
- **Assumptions** are risks taken to move ahead with design.
  - Tie back to risk analysis

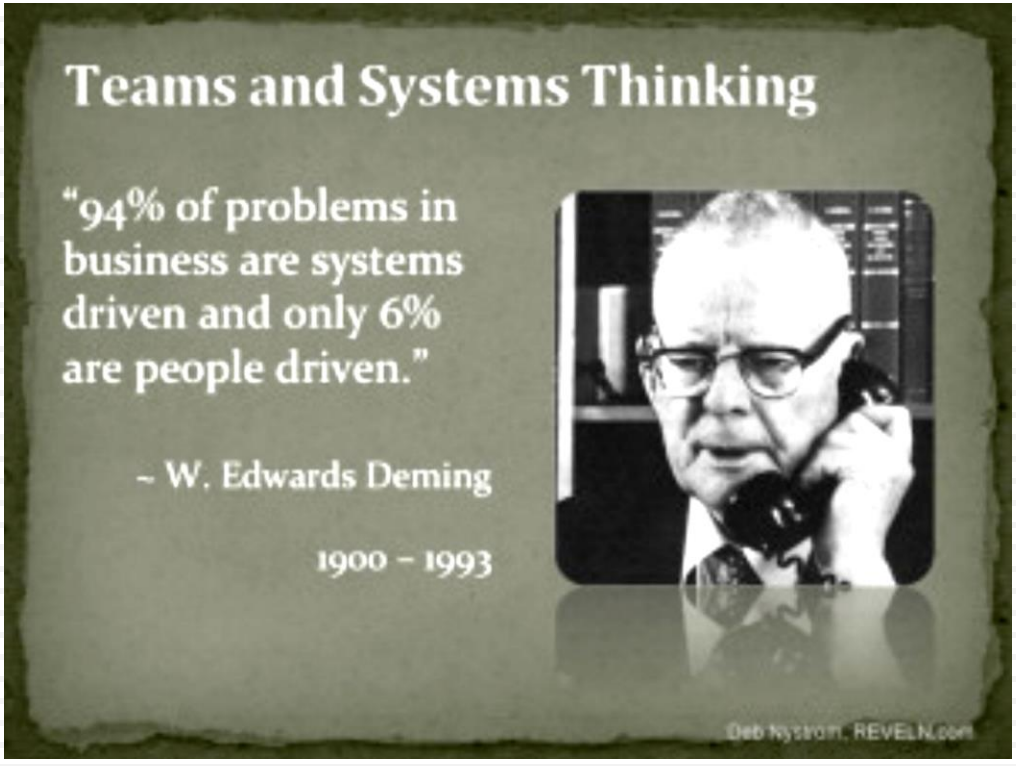
Constraints & system boundary

- Constraints – before project starts – enterprise analysis
  - Executive / sponsor identifies
  - PM manages
  - BA consults
- Boundary – project analysis
  - PM decides
  - BA consults

Exercise – constraint type

1. Access to creeks is limited to footpaths without motor vehicle access.
2. Recurrent funding for the project will be met from state growth stimulus funding.
3. Logging or removal of woody debris is allowed on-site after getting property owner's permission and necessary local regulatory permits.
4. Records must be kept of salmonid species strains used in production.
5. Restoration of the habitat debris must be made in early spring.

Enterprise analysis



Enterprise analysis

- High level business management
  - defining business goals
  - business case development supported by cost/benefit analysis and feasibility studies
  - strategic documentation
  - architectural planning
    - business process modeling (BPM)
  - architectural support and monitoring

Enterprise analysis terms

- **Vision** – a target in the future for success, the highest level business requirement
- **Values** – The output of the organization. Cultural, ethical, moral.

Enterprise analysis terms

- **Mission** – seen from the customer's perspective, the value from the vision in concrete answers. What, how and who.
- **Goals** – The output of the planning, groupings of projects.
- **Objectives** – the projects of the organization that are funded, have a concrete time line and have an expected value.

Activity

- Write down your
  - company's organizational reasons for existence
  - goals and objectives
  - accomplishment of those
  - and how it needs to change

SWOT analysis

- A tool for strategic thinking
  - Internal- people, processes, providers, innovation
    - Strengths
    - Weaknesses
  - External
    - Opportunity – a problem to solve
    - Threat – a negative risk

Walt Disney
SWOT Analysis

Project management

The Project plan

- An overview of all of the other project management documents
- The strategy
- Does not equate with a schedule made by Microsoft Project

Project phases

Business language

Initiation – strategy, identification of opportunity of value and constraints

Elicitation – collection of needs for processes involved in value improvement

Analysis – structuring of needs for value improvement

Technical language

Design – combining needs and constraints to create a plan

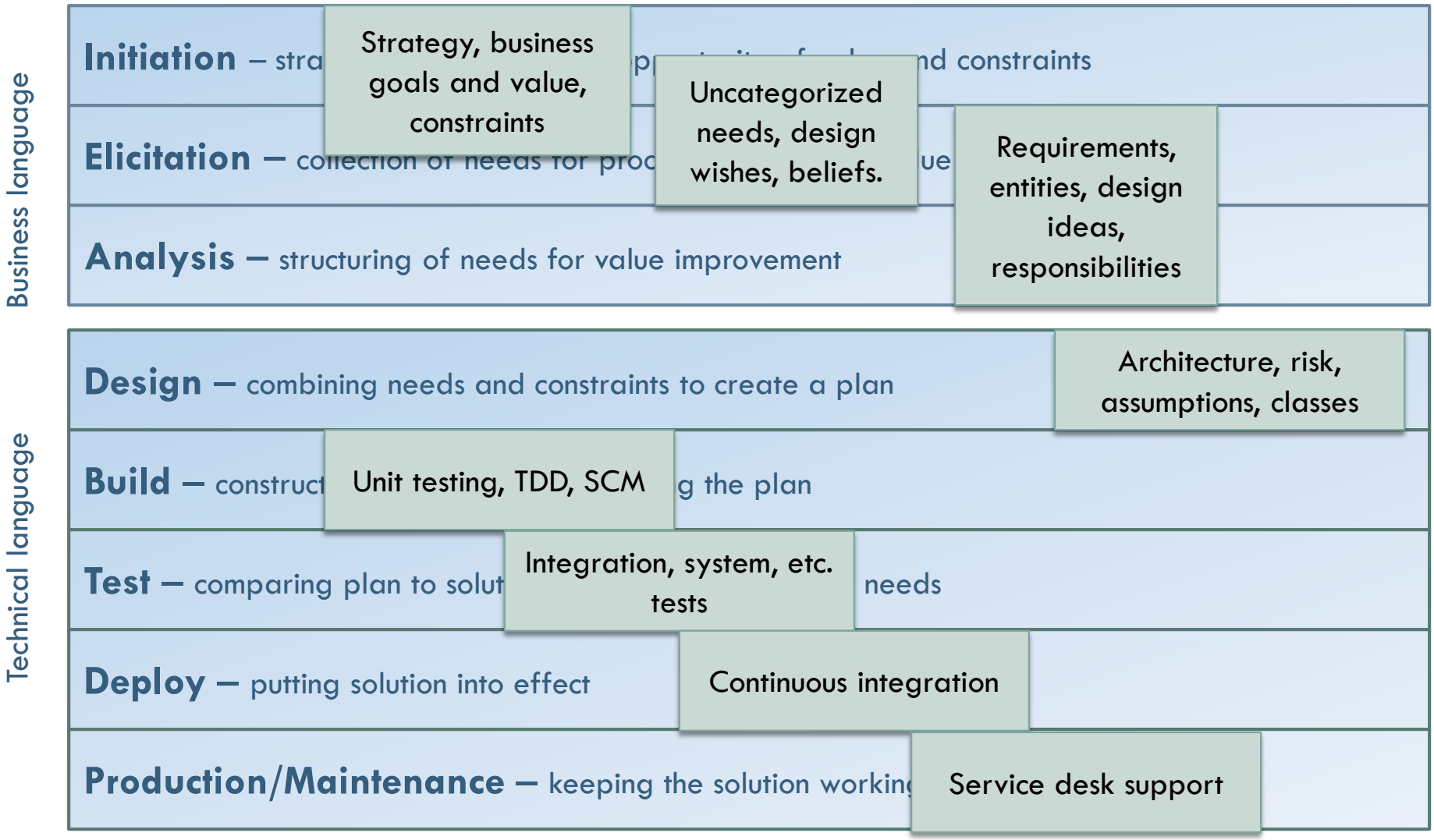
Build – constructing the solution by following the plan

Test – validation, comparing plan to solution and comparing plan to needs

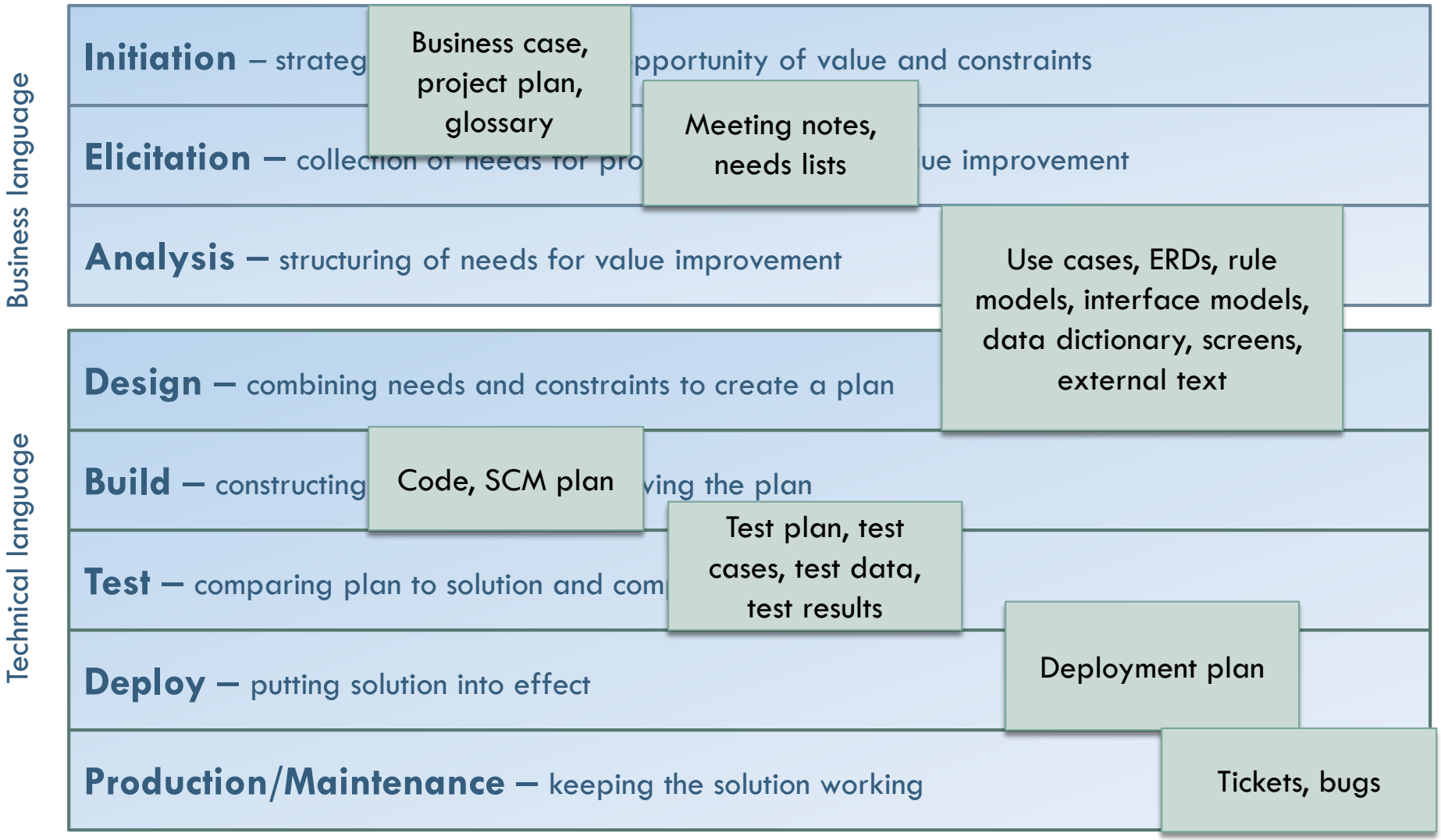
Deploy – putting solution into effect, training

Production/Maintenance – keeping the solution working

Project concepts



Project deliverables



Pre-charter documents

- Business case
  - Market for service, resources available, use of resources for service, value of service to company, tie to vision.
- Optional parts of business case or individual documents
  - feasibility study
  - top-level architecture
  - business requirements (goals)
  - project strategy plan
  - operations concept document

Project types

- System only - software only
- Business only - role and process changes
- Maintenance - update to previous project
- Combination - software with deployment, software with process change, software with any other business task, and can include updates.

Service delivery types

- In-house
- Product development
- Time and material
- COTS
- Tender process
- Contract development
- Sub-contracting
  - Out-sourcing

Requirements management plan

- Part of the project management plan
- Goals
  - to ensure a common understanding of the project
  - complete documentation as necessary

Requirements management plan

- Documents
  - people involved
  - tool usage
  - process of gathering
  - process of structuring
  - process of changes to requirements
  - quality assurance

Change management – Project tasks

- Minimize impact / disruption
- Optimize risk exposure
- Control changes through standardized methods and procedures
- Document changes

Change management – BA tasks

- Create a baseline
  - Business
    - Current business processes, data, business rules.
    - Used to provide plan to return back if change fails.
  - Requirements
    - Current set of requirements (1st draft of requirement statements) used to track changes.

Change management – BA tasks

- **Control change**
  - Evaluate
  - Accept
  - Integrate
- **Trace**
  - Identify
  - Document

Scope creep

- Projects will not enlarge if all requirements are gathered
  - Caveat: market demands can change
- The BA is tasked with gathering ALL the requirements.
- The BA should gather all the requirements as early in the project as possible.

Kanban

- Simple Toyota card system to communicate demand and capacity resulting in a just-in-time production process

Kanban boards

- Visualize your work
  - the sticky note swim lane board
- Limit work in progress – swim lane limit
- Manage flow – using lead times on notes
- Make process policies explicit
- Implement feedback loops
- Improve collaboratively, evolve experimentally

Lean

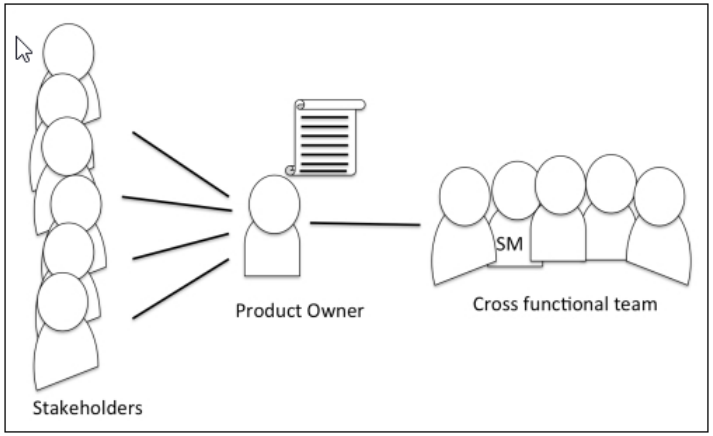
- a Toyota production practice considering resource use for any goal other than the creation of value for the end customer to be wasteful, and thus a target for elimination
- 3 stages
  - resource usage efficiency
  - flow efficiency
  - optimizing value

Agile manifesto

- http://www.agilemanifesto.org/
- http://agilemanifesto.org/principles.html
- We value
  - **individuals and interactions** over processes and tools
  - **customer collaboration** over contract negotiation
  - **working software** over comprehensive documentation
  - **responding to change** over following a plan

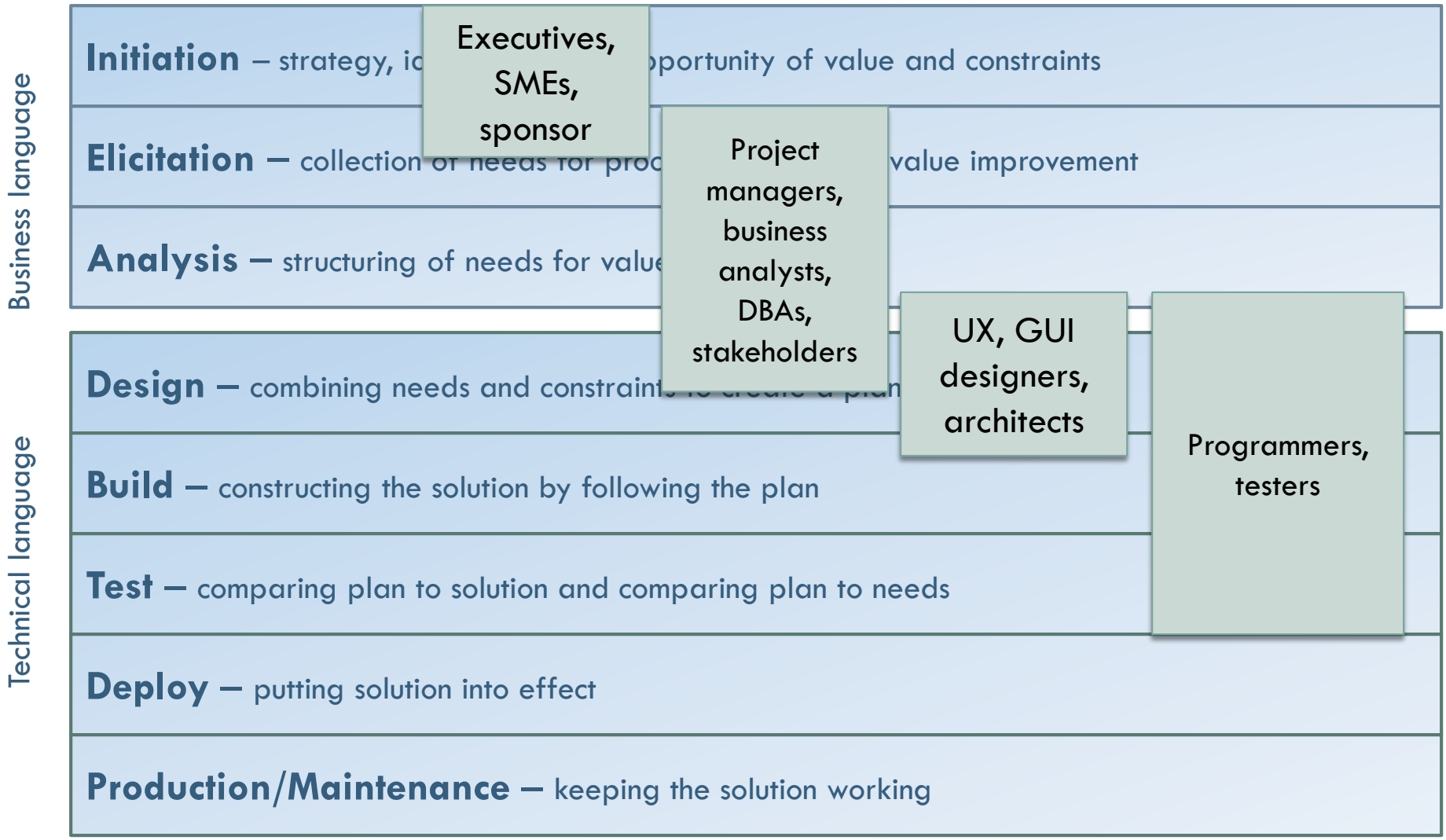
Scrum

- An Agile/Lean process framework
- Stakeholders
- Product owner (business analyst)
- Cross-functional team + Scrum Master (PM)
- Sprints of 1-4 weeks

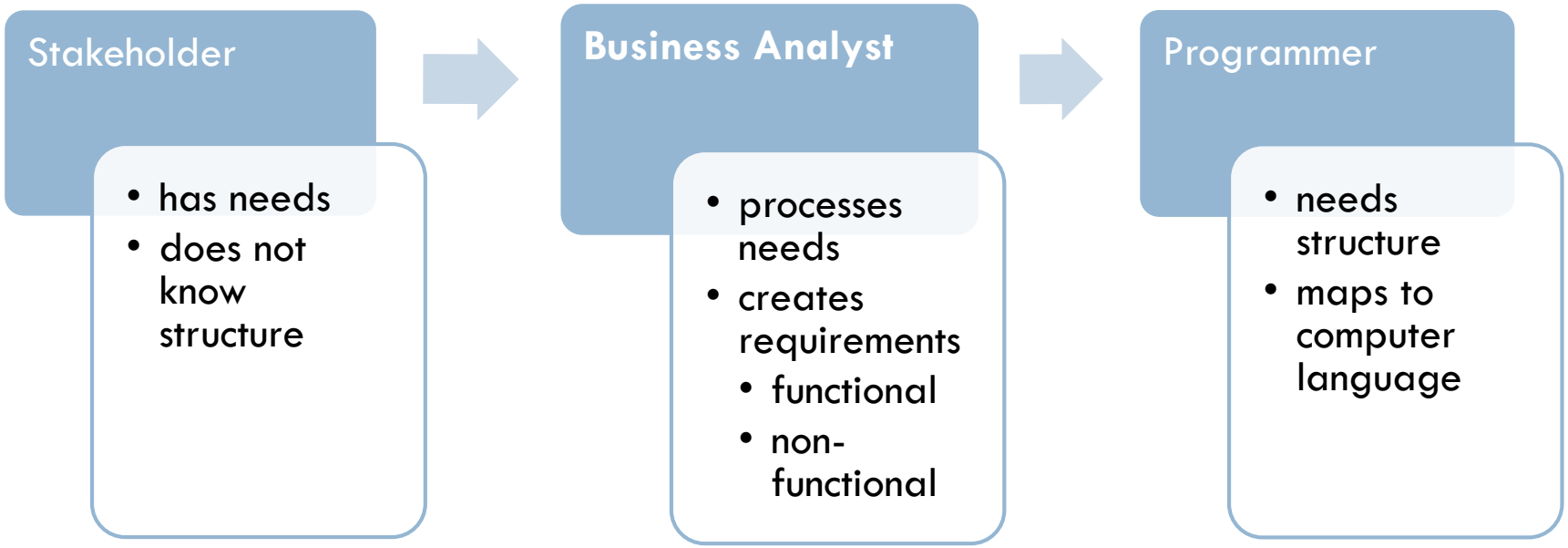


Roles

Project roles



The process by role



The stakeholder role

- Provides processes, business rules, and data used for business
- Is currently involved in a business process to give value to the enterprise
  - This process could be improved.
  - This process could be automated.
- Understanding this process usually requires assistance from a process oriented thinker

The stakeholder role

- Types
  - Sponsor, product champion
  - Direct user, indirect user
  - Advisor, provider
- Stakeholder profiles
  - Role, responsibilities, interests, success criteria, concerns, technical proficiency, work environment

The business analyst role

- Understands and communicates scope of business solutions
- High-level metrics
  - **understanding**
    - do you understand what's going on?
  - **communication**
    - do people do the right thing after a document is distributed?

The business analyst role

- Gathers high-level requirements of the business
  - requires skills in communication, modeling, technical writing, & processes
- Translates them into usable models for the IT professional.
  - requires skills in communication, modeling, technical writing, processes, and analysis
- Works on a team well with
  - any business stakeholder, project managers, data base administrators and software developers

The project manager role

- **Decides** project goals and leads members to those goals
  - Business scope is documented by the BA but project scope is decided by the PM.
- **Manages**
  - scope
  - time
  - budget

Blame chain

- Business
  - Trial & error
  - Tradition
  - "The market is crazy."

- Analyst
  - Document what they tell you.
  - Give them what they want
  - "Users don't know what they want."

- Coder
  - Code what you're told.
  - Make the schedule and PM happy.
  - "Analysts don't understand what I need."

- Tester
  - Test until the deadline.
  - Follow what everybody else did.
  - "I tried to find as many bugs as I could."

Quality chain

- Business
  - Customers have real needs.
  - "What can I do to help them solve their problems?"
- Analyst
  - I have to communicate needs clearly so they can be tested.
  - "How does what the customer does now help them?"
- Coder
  - I have to make everyone's job easier including mine.
  - "How can I translate requirements simply to code?"
- Tester
  - The customer needs the best app possible.
  - "How can I guarantee quality?"
  - "How can I let everyone know how well they did?"

Development

Our customer for software projects

Understand your programmer

- Communication is more successful when
  - you understand the language
  - you share goals
  - you listen
- Software has constraints
  - Analysis should be aware of them
  - Design must respect them

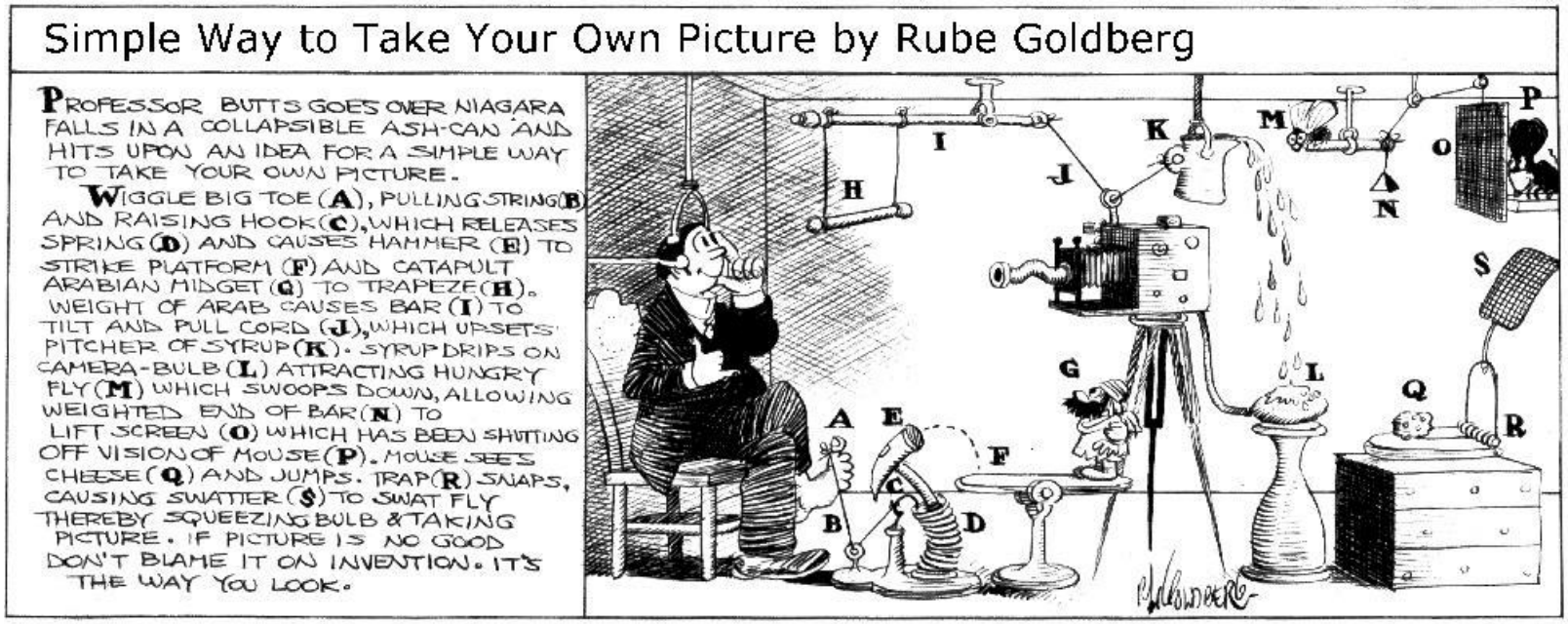
What does a programmer need?

- Scope
  - Workflow
  - Data used by workflow

- Structure
  - Code units
  - Sufficient detail

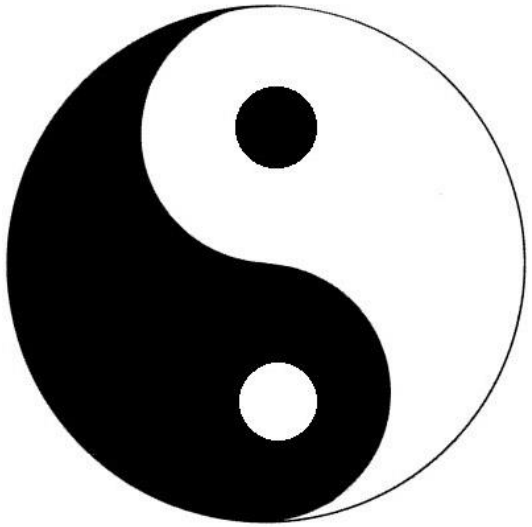
What makes good software? Scope

- Understandable
  - low number of abstractions
- Simple
  - low number of logic paths to test



What makes software better?

- More structural weakness as change increases
- As one increases, the other decreases
- Maintainability
  - low coupling
  - the ability to change some code without affecting other code
- Reusability
  - high cohesiveness
  - the better library

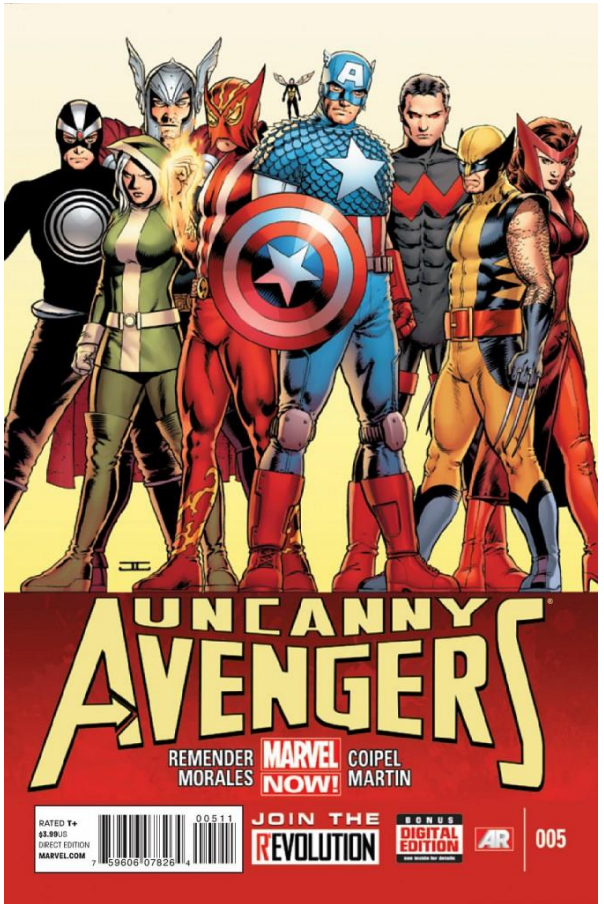


A real software development process

- User wants stuff
- Analysts write down what they say
- Programmers don't have enough detail and talk to users
- **Programmers try to figure out how to code it** by defining scope and structure
- Programmers code the best they can
- Testers don't understand what the code does and try to figure out how to write a test case
- Testers test the best they can

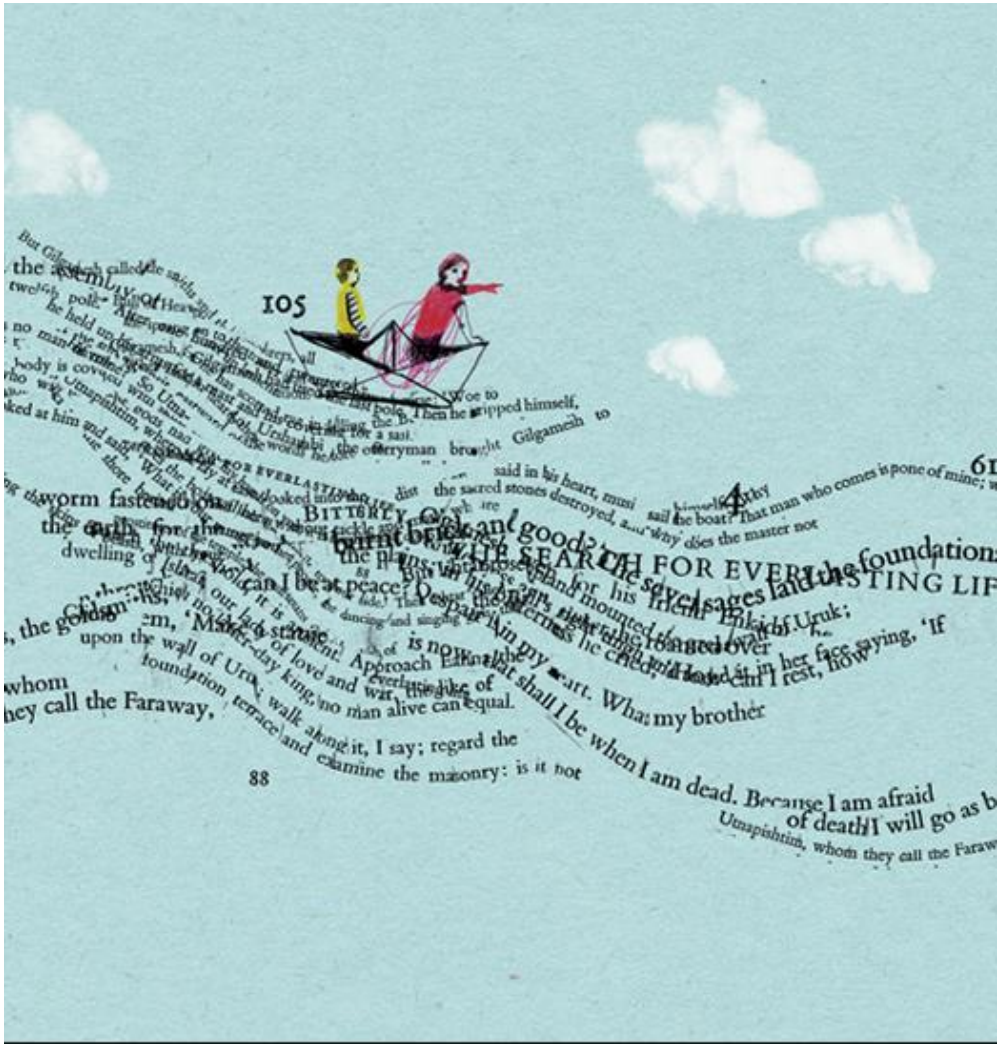
A better software development process

- User wants stuff
- Analyst writes down scope.
- **Analyst structures scope and makes it testable.**
- Programmers translate scope into code units and test it.
- Testers use testable scope to write test cases and test.



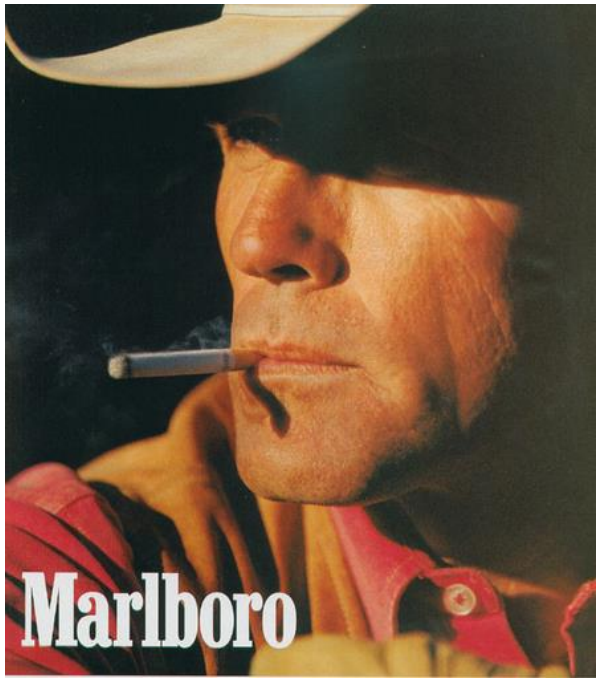
Analysis from a programmer's view

- Write lots of comments so other programmers understand it.



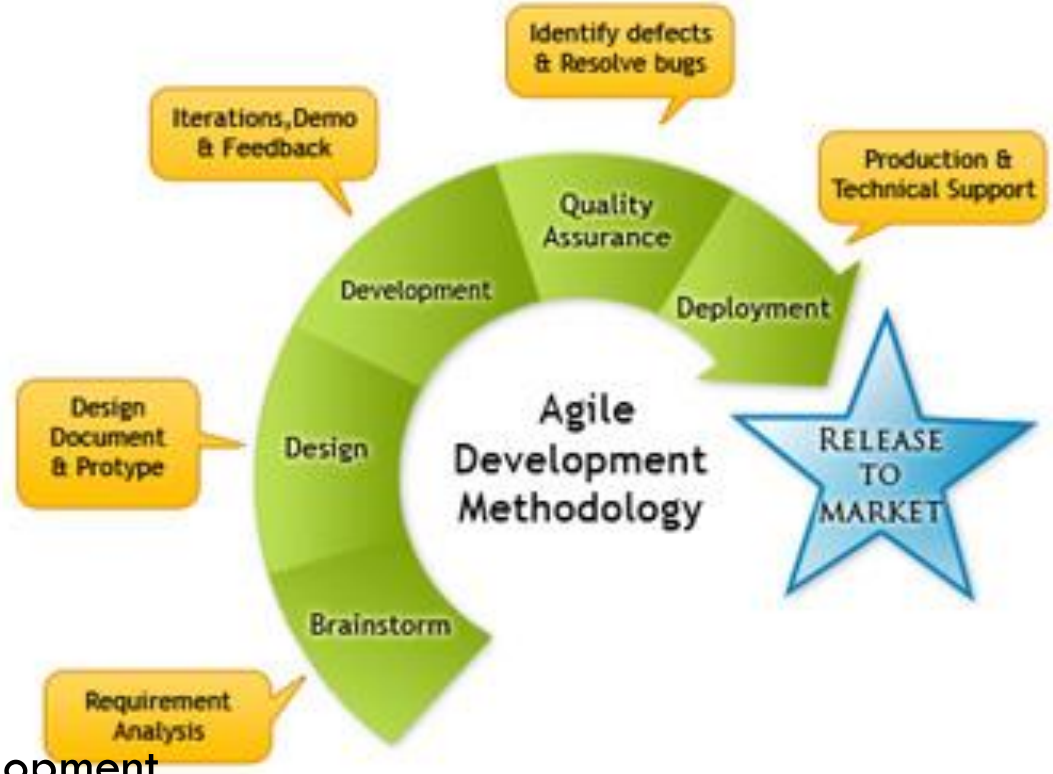
What a programmer is trained for

- Solve all problems
- Don't ask for directions
- Design processes to overcome weaknesses
- Do it all without a team



Analysis processes for programmers

- XP – Extreme Programming
- FDD – Feature Driven Development
- MDD – Model Driven Development
- DDD – Data Driven Development
- TDD – Test Driven Development
- BDD – Behavior Driven Development
- CRC cards - Class, responsibility, collaborators
- AM – Agile Modeling

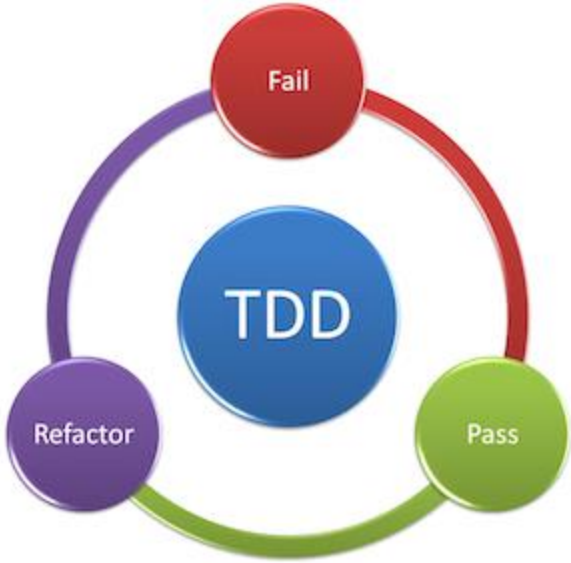


TDD

- a unit testing approach made popular by Kent Beck
  - created Extreme Programming
  - original Agile Manifesto member
  - Smalltalk
  - CRC cards
  - JUnit testing framework that spawned NUnit and others
  - http://www.threeriversinstitute.org/blog/

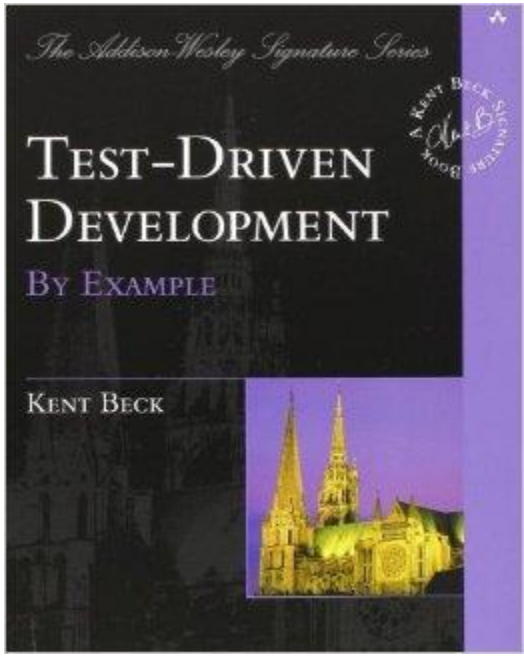
TDD - The process

- Red
  - the test fails
- Green
  - the test works at the minimal level
- Refactor
  - gradually make the code work correctly
- Repeat



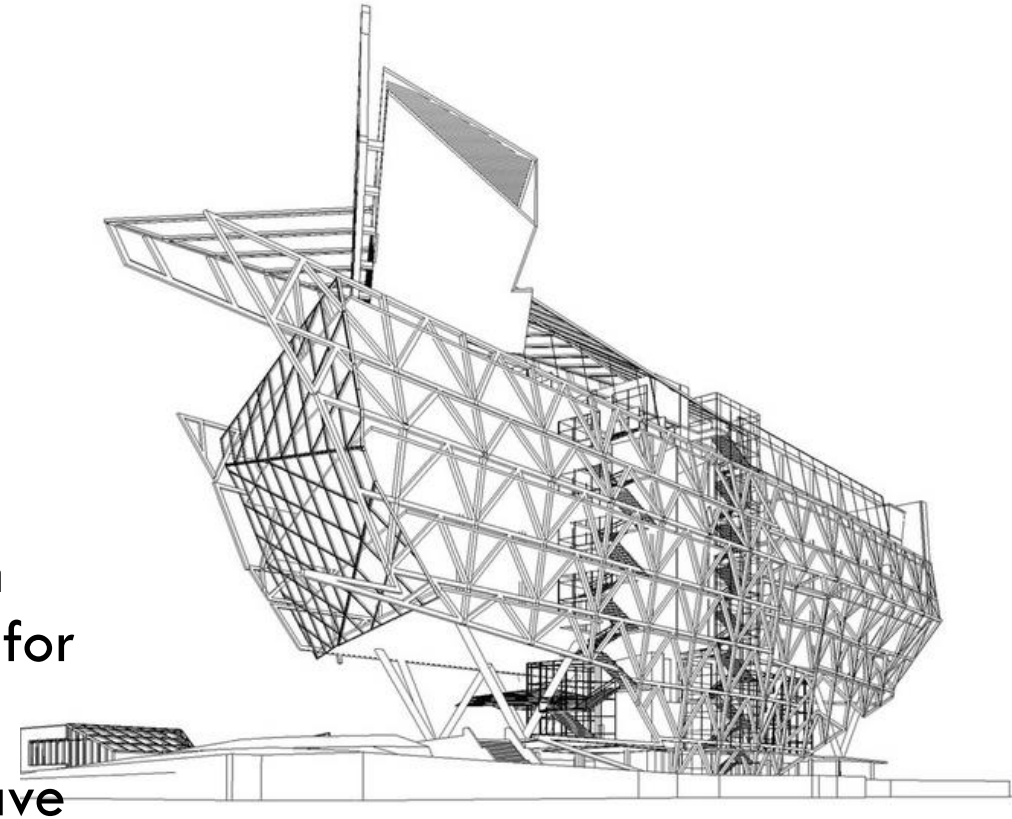
TDD - Implementing tests

- Pick a test that you know will work and can teach you something.
- Make your small-scale test work
- Reintroduce the larger scale test
- Make the larger test work quickly using the mechanism demonstrated by the smaller test
- Notice potential problems and note them on a to-do list instead of addressing them immediately



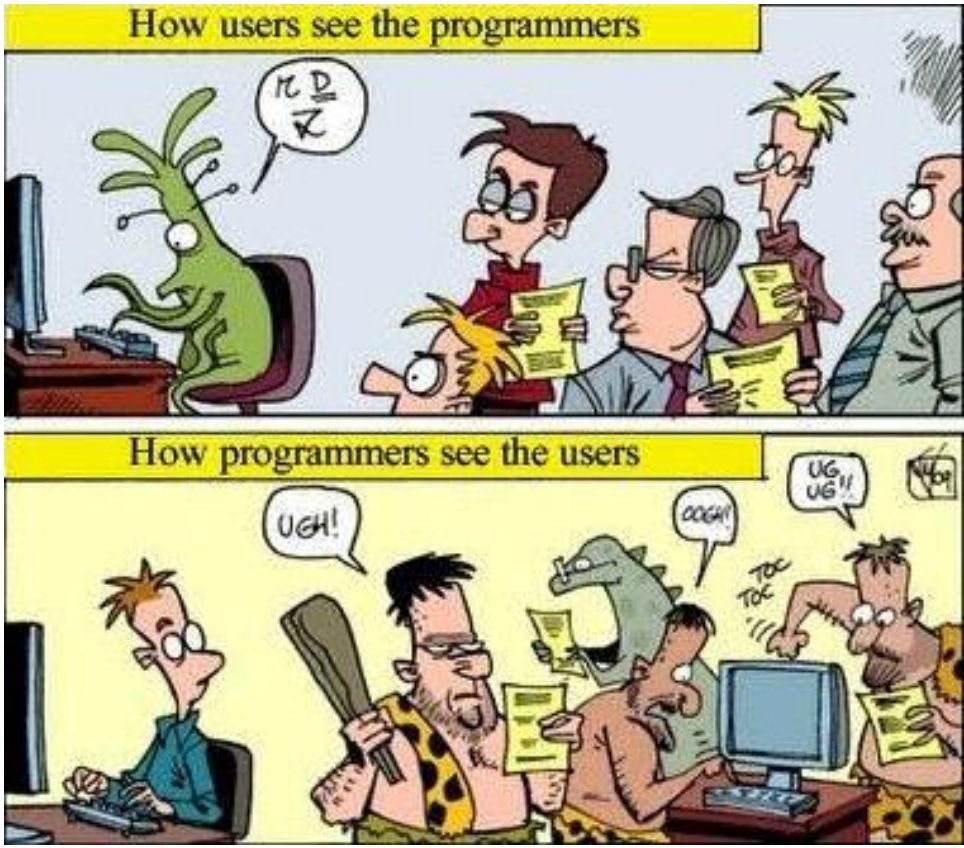
Takeaway

- Create better structured models
- Drive more detail in to workflow
- If you don't then
  - your models aren't that usable downstream even though they may be OK for business
  - your programmer will have to be better at analysis than you in order to write the code



Takeaway

- Talk to programmers about
  - what will be useful on a project
  - what was useful on previous projects
  - what models communicate best
  - what models provide better understanding



Resources

- 1975 – The Mythical Man-Month https://en.wikipedia.org/wiki/The_Mythical_Man-Month

Requirements

Requirements development

- Three steps
  - Elicit – acquire and document user expectations
  - Specify – define statements
  - Analyze – structure and add interface, priority, etc.
- Testing
  - Validation – checklists
  - Verification – get feedback
- Iterate as necessary

Requirements

- Part of the problem domain
  - may not be part of the final project scope
- A need is not a requirement until it is rewritten
  - needs are often not testable

Exercise

- The faulty accounting application
  - Ask about the reasonable expectations (of value) of the buyer matching their needs.
- Issues:
  - A. discount field causes crash without a value
  - B. database system slows after being in use without a way to fix it
  - C. business system does not produce credit memos.

Business requirements

- Highest level of requirement
- Found at the strategic level
  - In the business case
- Two to three per major project
- Sometimes confused with any kind of business language requirements or non-system projects.
- Breaks down into high-level user requirements which show how value will be created
  - User requirements are **business language** requirements also.
  - User requirements are found at the analysis level.

Requirements levels of traceability

- Strategic – business requirements
  - high level - mission, overall business model
  - low level - business case, project goals
- Analysis – user requirements
  - high level – achieves value or specific goal
  - low level – performs a task
- Design - technical requirements
  - using constraints to design a solution
  - models that turn into code, databases, web pages…

Metrics

- Measure through the process
  - the status of each requirement
  - the cumulative number of changes (including the cumulative number proposed, open, approved, and incorporated into the baseline)
- Project management
  - effort and funds expended

Interface analysis

High-level system modeling

Systems, messages, data

- Systems = rooms in house – kitchen, TV room, kid's room
- Messages = come get dinner, come get my dirty plate, come get dessert, come get my dirty bowl.
- Data = food, beverage, china, silverware, napkins

Systems models

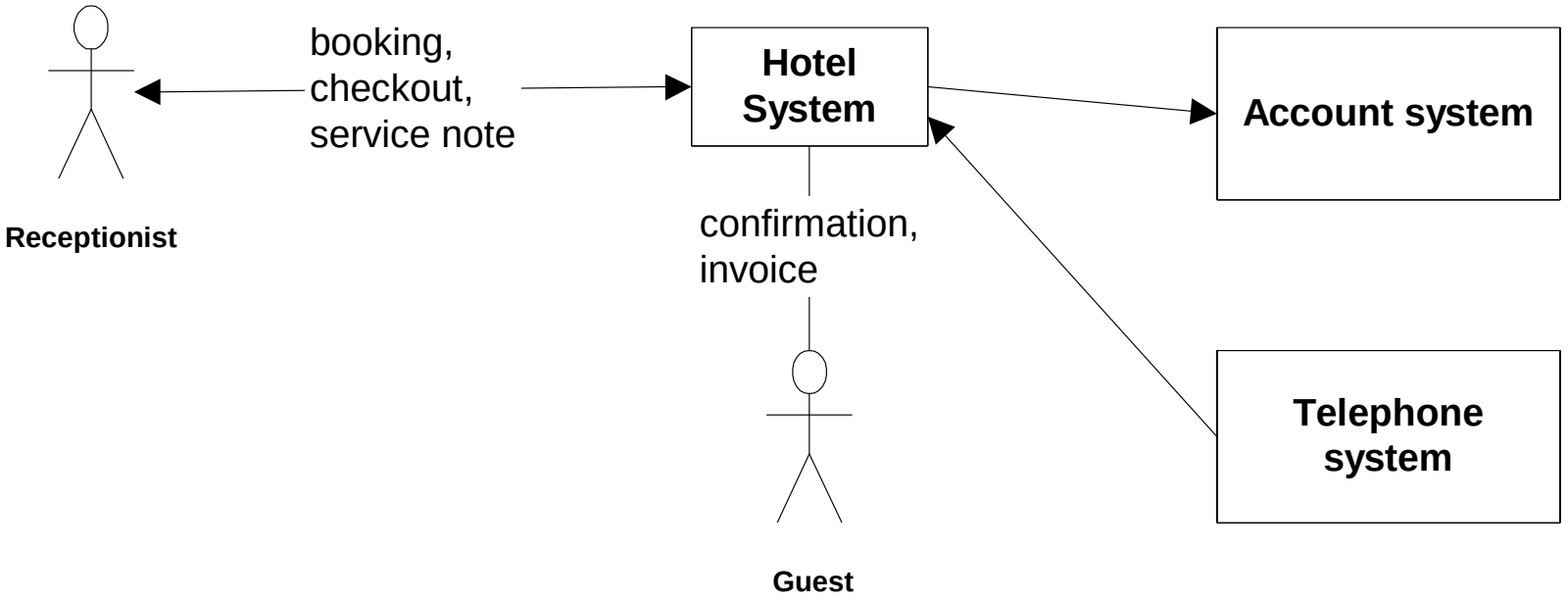
- Focused on either
  - Data payloads
  - Process triggers
- An interface (API) describes all process interactions of one system
- A data interface describes data definitions shared between systems

Data flow

- Data flow interface analysis
  - Who will supply, use, or remove information from the system?
  - Who will operate the system?
  - Who will perform any system maintenance?
  - Where will the system be used?
  - Where does the system get its information?
  - What other external systems will interact with the system?

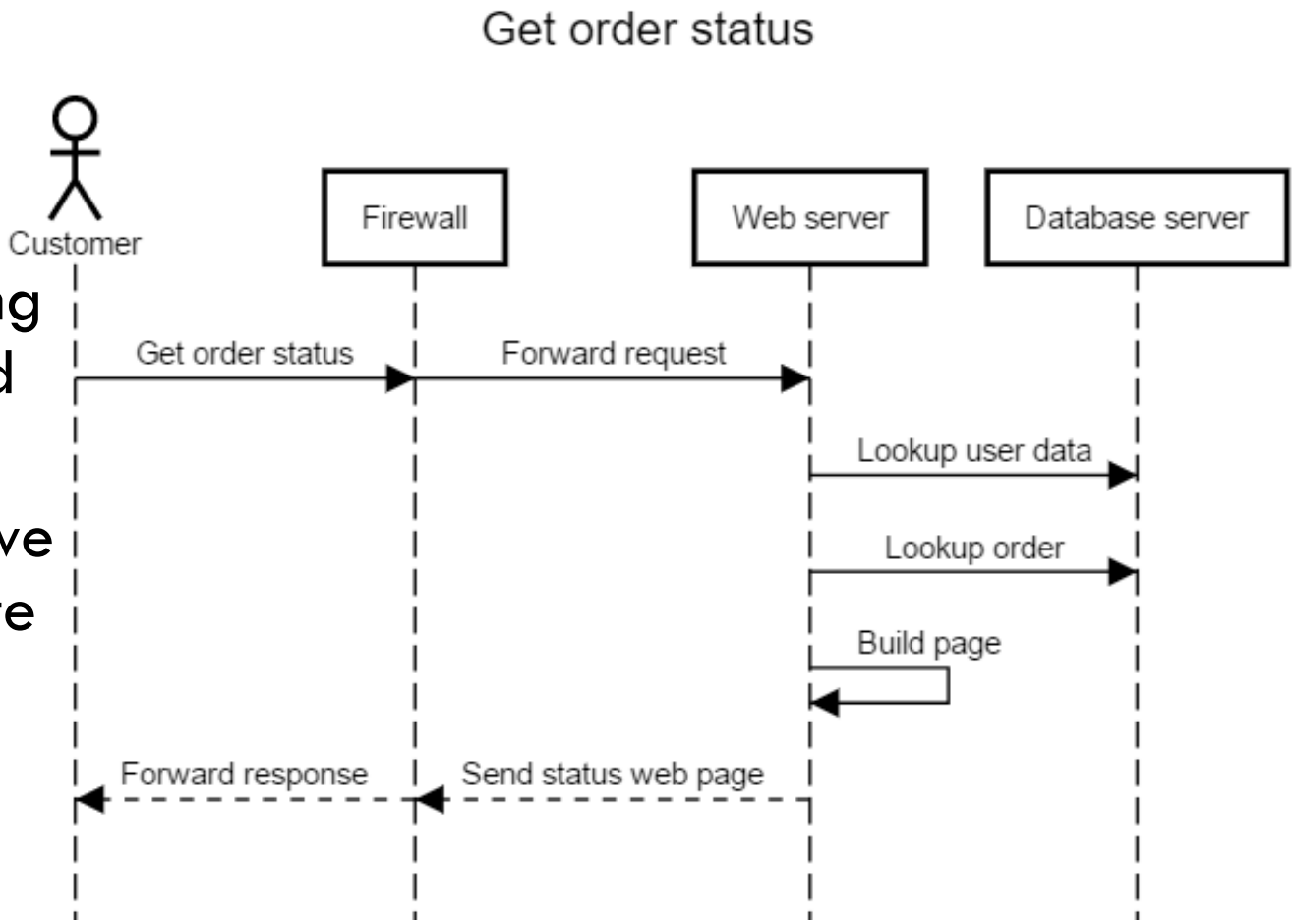
Context diagram

- Shows system in environment with external entities that provide and receive information or materials to and from the system.



System-level sequence diagrams

- Sequence diagrams show
  - participating systems and roles
  - how often we communicate
  - events in time order



Sequence diagrams - tools

- Text driven – best!
  - * http://sequencediagram.org
  - https://www.websequencediagrams.com/
- Drag and drop objects
  - Visio
  - https://www.gliffy.com
  - https://creately.com/

Exercise - System level sequence diagram

- ATM
  - Withdraw Cash sequence of tasks only
    - Label the diagram Withdraw Cash
  - No need to show multiple user to ATM requests
    - Just show last one before needing to send a message to another system
    - Draw a symbol to communicate the skipping of steps

Requirements elicitation

Getting words from people who don't know

Elicitation process

- Requirements gathering is not a specific technique that you apply to a project.
  - it's a set of tools you apply as needed
- Requirements cannot be gathered completely.

Barriers & objections

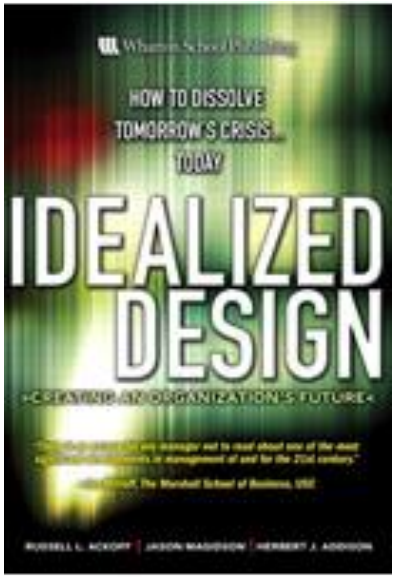
- It's why you have a job.
- Excuse rebuttals can be summed up by
  - you are not the end user
  - the end user must be talked to
  - the project must do what the business wants
  - the project must fulfill the business goals and not your personal ones
- **Reading** - Overcoming objections

Survey / questionnaire

- Beginning of phase
- Email
- No more than five questions
- Open-ended questions
- No attribution
- Expect 10 - 50% participation even with management support

Focus groups

- IBM style and others
  - Introduction
  - Bad experiences
  - **The future without constraint**
  - Issues with that
  - Prioritize issues
  - Review



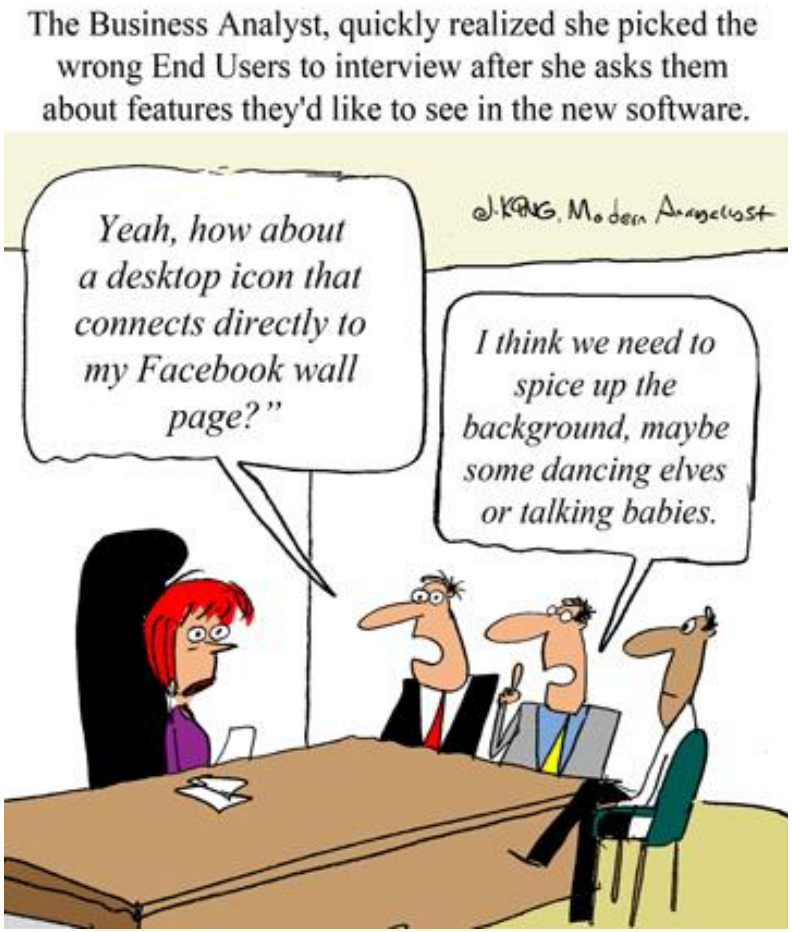
- Listen or read:
  - http://knowledge.wharton.upenn.edu/article/idealized-design-how-bell-labs-imagined-and-created-the-telephone-system-of-the-future/

Individual / Group interview

- **#1**
- One of the best ways to get information and buy-in
- Cross-cut stakeholders
- No prejudice in questions

Individual / Group interview

- Prepare for an interview
  - Research
  - Assign a scribe
- Listen without bias
  - Don't ask why do you do that, ask what it helps.
  - Don't ignore useless detail
  - Don't repeat your questions
- Use other techniques



Individual / Group interview

- Form
  - email 3 days before interview
  - user profile
  - problem definition
  - background
  - open -ended confirmations
- After
  - review with scribe and write top points

Nobody said Business Analysis will be easy!

Brainstorming

- Use brainstorming to get quick lists of thoughts.
- Mostly used with other analysts.
- Don't criticize or discuss the thoughts of yourself or others until the creative part is over.
- All ideas are written down
- Judgment comes after the storm

Requirements workshops

- Free for all
- Any kind of activity
- Needs structure
  - Agenda, roles, right people, small group, ground rules, defined deliverables
- Also called
  - JAD session, design, domain, modeling, stakeholder or facilitated workshop.
- Watch video: The Expert - https://www.youtube.com/watch?v=BKorP55Aqvg

External stakeholders

- Similar companies
- Ask suppliers

Other

- Reviews
- Pilot experiments
  - More of a test

Exercise:

- What are the principles or tools recommended in this article?
- http://speckyboy.com/2015/10/21/guidelines-for-better-client-feedback-on-web-design/

Elicitation without discussion

Grasping the status quo, getting a baseline

Document analysis

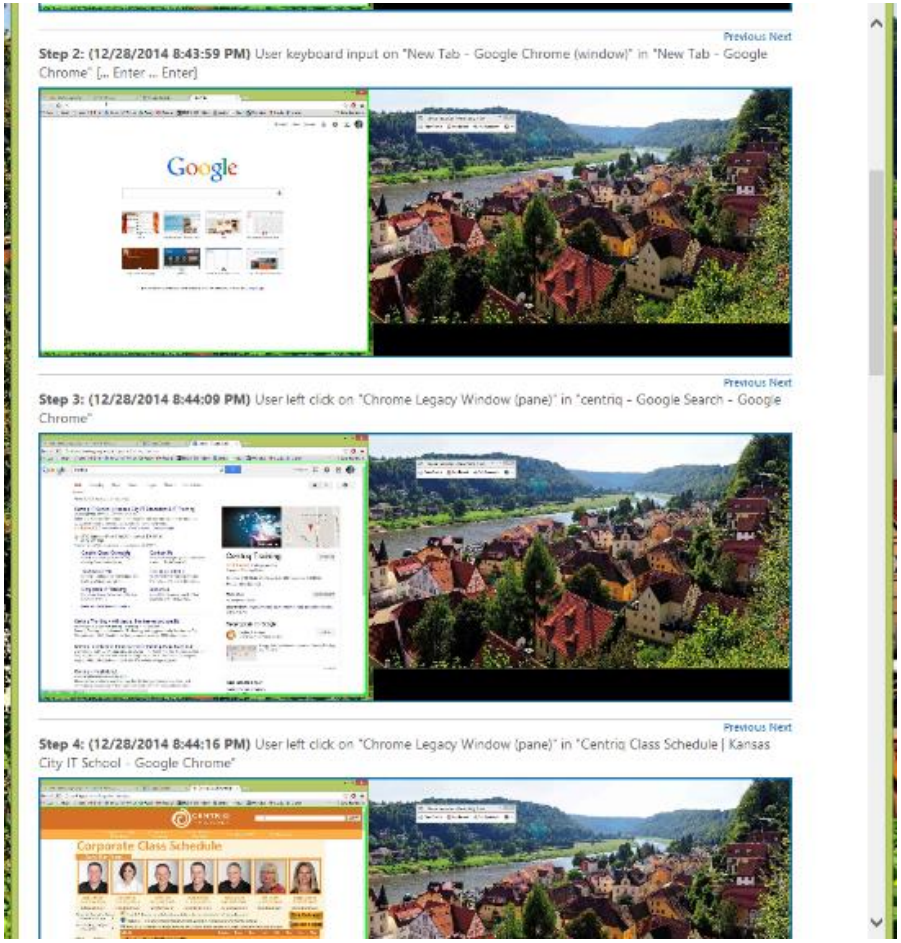
- #3
- Beginning of phase
- Great for offline study and no impact on stakeholders
- Pre-interview

State analysis / stapling

- Become the data / form
- Go around and have the data change by different processes / departments
- Record the activity
- Create a state chart model from this

Observation / task demo

- No words, just action
- Task analysis - The fake crisis, mock emergency
- WebEx demo of current process
  - Record screen activity
  - Record audio from phone
- PSR - Problem Step Recorder in Windows



Elicitation by design

Prototyping

- **#2 Sketching / wireframing**
  - Analysis to design information will be elicited
  - Prototyping for eliciting needs, not
    - Prototypes for analysis
    - Prototypes for design
  - Types
    - Screens
    - Reports

Prototyping

- Sketch on
  - Whiteboard – camera to capture
  - Paper – 8 ½ X 11” sheets (one per screen)
  - Wireframing apps - when users are remote
    - Sometimes reusable
    - Balsamiq, Axure
    - Visio…

Exercise - Prototyping

- Use landscape orientation
- Identify
  - Text areas (with name of data type) boxes or greeked
  - Active areas (links)
  - Images (with name)
- Use the entire box as the screen.
  - Not the check input, keyboard, cash dispenser…
- Follow the requirements!

Elicitation in Agile PM

Agile / Scrum

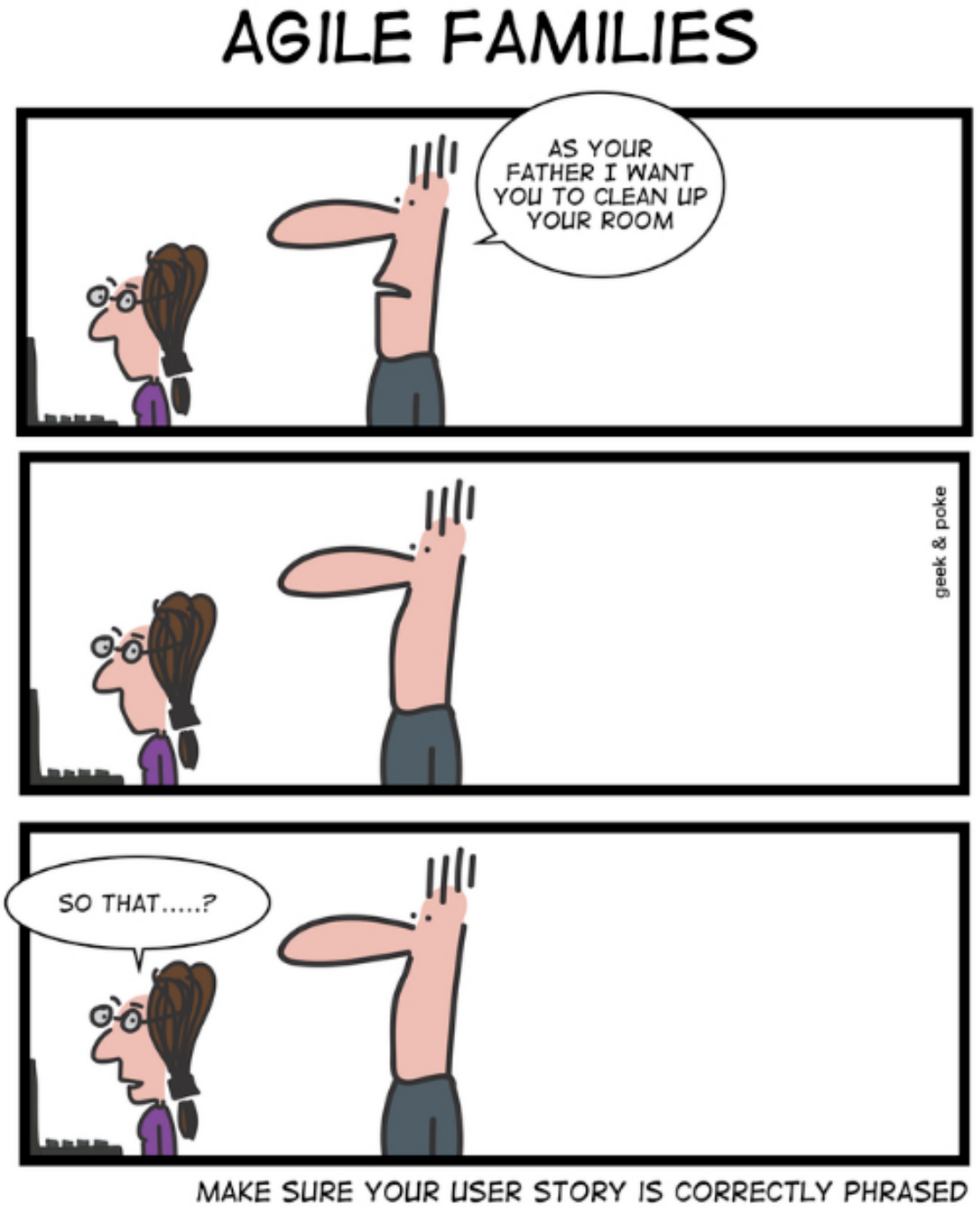
- Agile (principles), Scrum (implementation of Agile)
- Common tasks
  - Create user stories and put into product backlog
  - Roll up stories into themes
  - Prioritize on a wall
  - Work out details continuously with stakeholders
  - Complete requirement just before development
  - Use change control
  - Re-prioritize as necessary

XP story telling

- A requirement elicitation technique
  - not a final product
  - conflicts are OK
  - uses note cards
- **As a** <some user role>**, I need** the system to <high-level functional requirement> **so that** I can <get value>.
  - detail out with use case structure
- Used in Agile projects
  - *Minimize output, and maximize outcome and impact.*

User Stories

- User stories express **expectations as value** for the stakeholder.
- Use cases define **goals as value** for the business.
- User stories create work packages to manage and conversation points during the project.



User story mapping

- Backbone (functional goals), body (tasks).

Requirements Tools

Tools

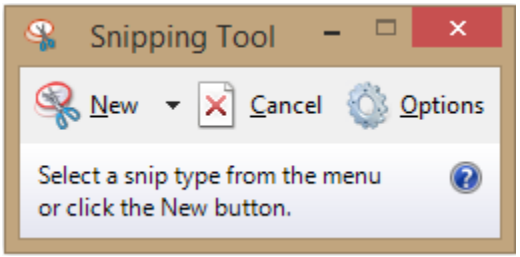
- Modeling
  - Pen and paper
  - Word
  - Visio
- Requirements
  - Word
  - Excel
- Requirements management
  - will have traceability

Major vendors

- **IBM Rational** and the RequisitePro product,
- **Telelogic** and its Doors/Enterprise Requirements Suite (ERS)
- **Borland** Software Corp. with StarTeam and CaliberRM (via its purchase of StarBase)
- **Computer Associates** (CA) International Inc. (through a partnership with Integrated Chipware Inc. and its RTM Workshop product).
- **HP** – Quality Center

Other tools

- Zoomit – zoom, draw
- Snagit $/ Camtasia $
- Snipping tool
- Greenshot
  - http://getgreenshot.org/
- OneNote
- Xmind, Mindmaster



Technical writing

Writing is a craft

- The best writing style is one you don't notice.
- The best writing is read.
- Get better by practice.

Write for humans

- Look for ways to make it easier.
- Stress the important points.
- Is it clear?
- Is it worth your reader's time?
- Is it enjoyable?

Tech writing tips

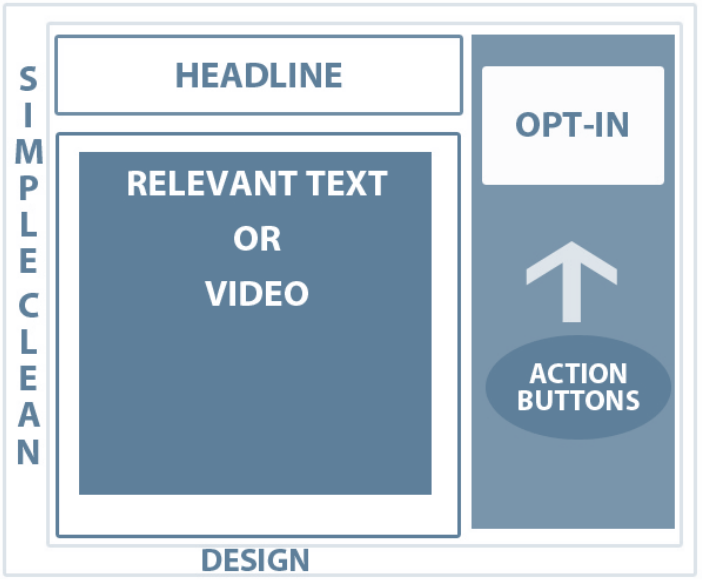
- Create lists
  - people jump to them and continue reading
  - I sell apples, bananas, cherries and grapes.
- Form follows content
  - Don’t assume the template will write the document for you.
  - Do you need an appendix?

Tech writing tips

- Organize details
  - Do you need another type of document?
- Reinforcement
  - Text, graphics, charts…
  - Charts, graphics, text…
  - Graphics, text, charts…

Tech writing tips

- Use relevant text and images
  - The accelerating availability of molecular sequences, particularly the sequences of entire genomes, has transformed both the theory and practice of experimental biology. Colors are important: Stick with Blues and Greens or pastels. Red is normally a “Stop” color and should not be used for the overall design. Red is great for drawing attention to the Call to action so reserve the Red for just this one item. Knowledge of the biological role of such a shared protein in one organism can certainly illuminate, and often provide strong inference of, its role in other organisms.

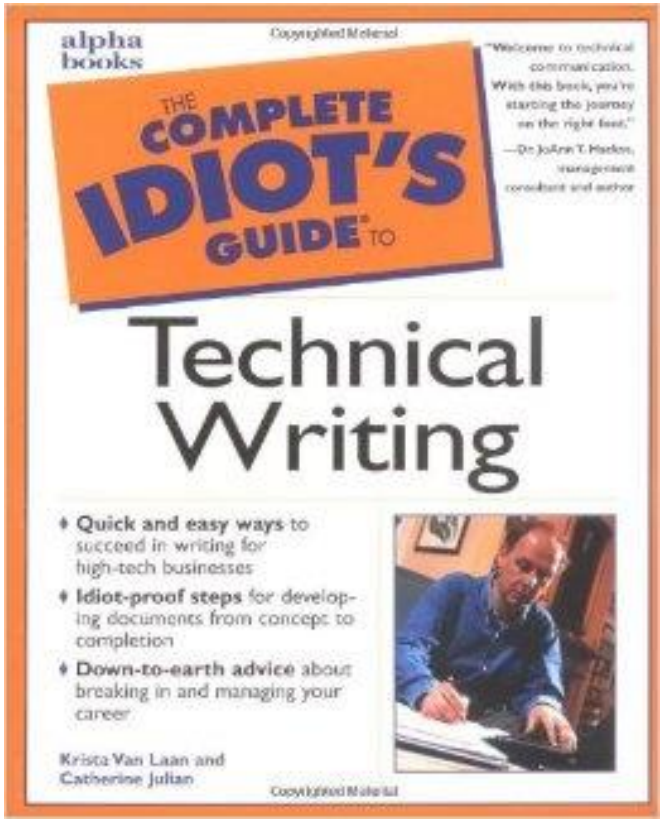


Tech writing tips

- Completeness counts
- Use business terminology
  - Anything heard in a manager's office not describing automation
- Use consistent terminology
  - glossary
- Assume a friendly reader
  - …anybody as long as they work here and have eyes to read this and are authorized to get a copy during work hours and aren't on lunch break or vacation

Books, recommended

- **Complete Idiot's Guide to Technical Writing**, Krista Van Laan and Catherine Julian, Alpha Books, 2001



Requirements Documentation

A report structure - business

- Introductory parts – includes business case info, goals
- Constraints/ high level requirements - system limits, what to deliver, the interfaces, system sequence diagram
- Functional requirements – function lists, feature requirements, process descriptions (SRS), operation, maintenance, conversions, installations, training, documentation
- Non-functional requirements
  - Data requirements – data models, data dictionary
  - Business rules
  - System, quality, SLA

Requirements documentation

- Use cases
  - Name and index number
  - Actors
  - Pre-conditions
  - Flow of events
  - Post-conditions
  - Alternative flows
    - Errors, optional paths
- Add to appendix if not in current structure

Requirements documentation

- Glossary
  - Dictionary of common terms relevant to project
  - Can be enterprise wide but should be extracted for each project
  - Business terms
  - Assign a responsible analyst

Requirements documentation

- Report structure - technical requirements (TRD)
  - Specifications tied to the business requirements
  - Screen shots
  - Architecture

Analysis documents

- Business processes
- System constraints (strategic analysis)
  - Economic
  - Political
  - Technical
  - System
  - Environmental
  - Project management – time and money

Documentation

- IEEE Standard 830-1998 / 29148-2011
  - More structured
- The Volere Requirements Specification Template
  - Even more structured and now for sale

Additional report sections

- Document info, overview
- Expectations, preferences
- Invariants (something doesn't change)
- Platform
- Global characteristics
- Design constraints
- Likely changes (risks or assumptions)

Interim/secondary report types

- Traceability
- Compliance matrix
- Priorities
- Requirements validation/testing matrix
- Risk
- Unsatisfied/unallocated requirements

Design tips

- Group logically
- Use sequences
- Graphic design principles
  - Emphasis
  - Contrast
  - Repetition
  - Alignment
  - Proximity

Requirements statements

The process of writing traditional requirement documents.

Requirement quality criteria

- **#1 testable**
- **#2 complete**
- Consistent, correct, unambiguous, ranked, modifiable,, and traceable (IEEE)
- Feasible, independent, necessary, non-redundant, terse, understandable, measurable/specific

Testability

- Testing is
  - What programmers/testers do, so give them a testable statement
  - Ensures quality

What's a good metric for scope?

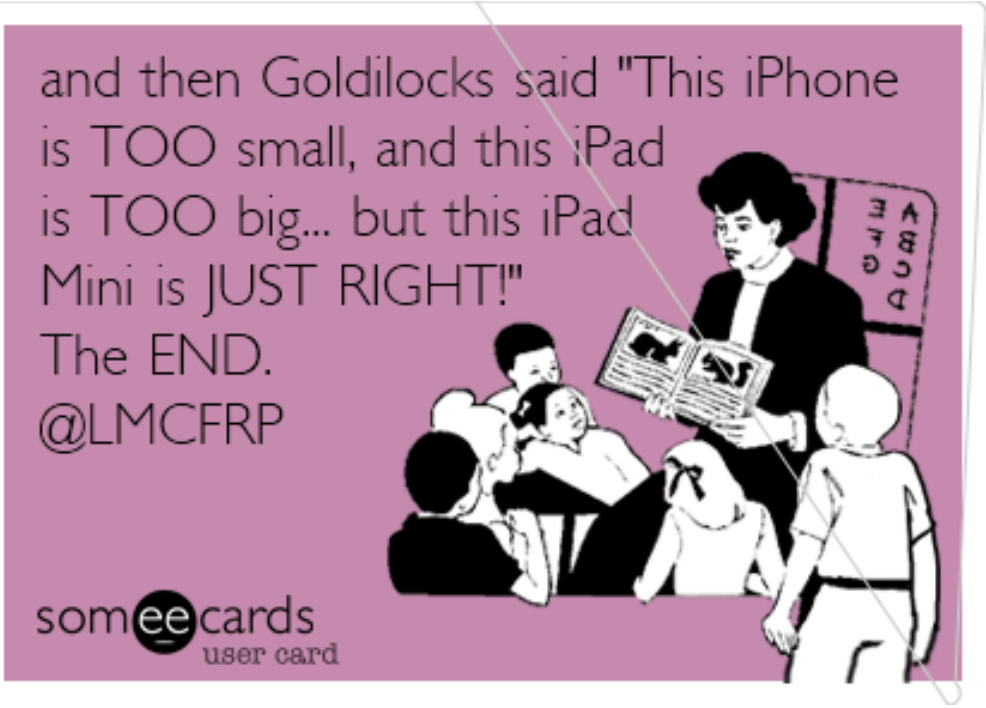
- Goals
  - A deliverable of measurable value
- Goal level use cases
- User stories
  - *As a <type of user>, I want <some **goal**> so that <some reason>*
- Work package
  - The work defined at the lowest level of the work breakdown structure for which cost and duration can be estimated and managed. -- PMBOK® Guide
- CSFs, MVP

How can scope be organized?

- Goal level chunks
- Bigger
  - Qualities resulting from many goals
  - Objectives
  - Business goals
- Smaller
  - Processes dependent for value on other processes
  - KPIs
  - Multiple non-goal chunks

Scope examples – goal driven

- Too small
  - Log in
  - Log out
  - Search
- Too big
  - Manage accounts
- Just right
  - Deliver package
  - Adjust account
  - Edit personal data



Requirements documentation

- Required data
  - ID and date
  - **statement**
  - Originator
  - Priority (goal level and above, business value not personal)

Requirements documentation

- Optional
  - Project
  - System / subsystem
  - Date updated
  - Cross-references
    - Business rules, data, prompts & menu text, designs
  - Level
  - Tracing
  - Index

Requirements documentation

- Optional
  - Purpose
  - Explanations
  - Examples of ways to meet
  - Stability
  - Complexity
  - Stakeholders' interests
  - Guarantees (minimal & maximum success)
    - Same as post-conditions

Prioritization

- Value to project management
  - Selection of scope based on budget and schedule.
- When to prioritize
  - Early
  - Progressively
- How to prioritize
  - It's not how important to the stakeholder it is, it's about the business
  - Don't ask the stakeholder

Prioritization – what to use

- Essential
  - Scope of use (impact)
    - how much of the business will it improve?
    - how many of the staff will it help?
    - Externally equated to target market
  - Business value (urgency)
    - how much do you wish the business had it now?
    - how bad will the business look if it fails in the future?
    - what level of person is asking for it?
    - Externally equated to price willing to pay
- Optional
  - Anything else important to the business

Prioritization

- Using weighted averages
  - "I'll give it a 9.27"
  - No units = no metrics = no standard
  - Use for understanding but not communications
- Other categories
  - Negative risk = Impact * Urgency * probability of failure
  - ROI
  - Satisfaction

Requirements levels

- **Goal driven scenario**
  - What sequence of steps leading to a goal will give value to the business?
  - Scope like PM's WBS: 3 - 10 days of work
  - Lower level manager
  - Target for initial requirements document
- **Group of goals**
  - What broad grouping of goals do you want the system or <role> to do?
  - manage, handle, control, do, work with, take care of
  - Higher level manager

Functional requirement levels

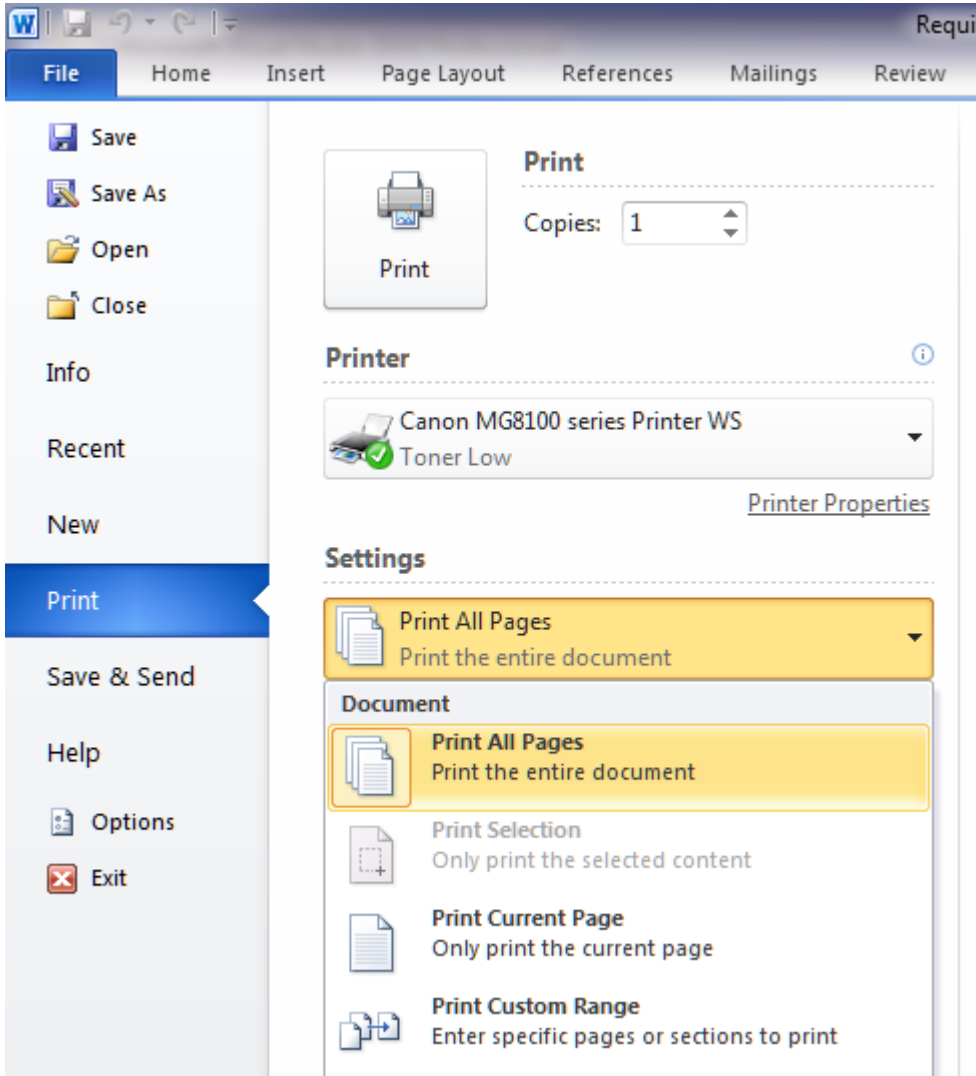
- **Partial scenario** / group of tasks
  - What are the individual or named processes in the scenario ?
  - no or little business value by itself
  - Staff
- **Task**
  - What are the specific actions that need to happen that are the basic steps of the scenarios?
  - Staff SME
- **Design "requirement"**
  - an idea about how it should be built
  - Record as a design recommendation

Deciding the level

- Use words “roll-up” and “drill-down” to talk about relationships between levels with business language.
- Grouping of goals
  - goal
    - partial goal
    - task
    - partial goal
      - task
      - Task
- Can you define a requirement (at a higher level) that summarizes a group of requirements?
- Can you define a requirement (at a lower level) that is a part of the requirement?

Deciding the level

- Similar to a menu
- manage **File**s
  - **Print** file
    - Choose copies
    - Choose printer
    - Choose settings – Print custom range
      - from pages: ?
      - to pages: ?



Traceability

- All higher level requirements have lower levels
- All lower level requirements have higher level
- Traceability matrix
  - Assigns codes
  - Tracks relationships

ID	USER REQUIREMENTS	FORWARD TRACEABILITY
U2	Users shall process retirement claims.	S10, S11, S12
U3	Users shall process survivor claims.	S13

Exercise

- Order these requirements by granularity
- **W1** – The system will count hits (files requested) for each web page.
- **W2** – The system shall create a report that shows the number of people to visit web pages.
- **W3** – The system will manage marketing information on the web site.
- **W4** – The system will report on file tracking information throughout the web site.
- **W5** – The system will use a hit counter made up of individual digits.

Exercise

- Order these requirements by granularity
- **H1** – The system shall check in a guest.
- **H2** – The system shall validate a password.
- **H3** – The system shall log in a clerk to the system.
- **H4** – The system will manage hotel guests' accounts.
- **H5** – The system will report on occupancy levels.

Too little info

- Clues to a higher level requirement, business rule, or lack of knowledge
  - Modifying phrases
  - Vague verbs
  - Most adjectives
  - Most adverbs
  - Time without units
  - Unnamed things
  - Pronouns without reference
  - Passive voice
  - Assumptions and comparisons

TMI

- Conjunctions
  - Split into separate statements
- A grouping of data items
  - move to the data dictionary
- Other systems
  - Phrase with responsible system and move to a sub-project
- Useless
  - negative

Functional requirements

The requirement with a verb.

The functional requirement

- The functional requirement statement contains
  - A responsible party/noun
  - The action/verb to be done
  - A description of the things/direct object which the verb acts on.
- What **strong specific verb** is used to describe what the system or <role> does?

Functional requirements

- Organized by
  - a function or feature list
  - a function description
  - an activity diagram by function
  - a user interface prototype
  - a time-ordered work task list (use cases)

Verb style

- Present tense
- Imperative
- Modal
  - The system shall …
  - The system shall have the ability to…

Statement opening

- A system functional requirement starts with “the <system> shall …”
- A business functional requirement starts with “the <role> shall …”

Requirement detail

- Analysis level
  - A description of what to do without mention of any automation
- Design level
  - A description with a particular system / architecture / communication protocol in mind.
  - Valid when knowing your constraints (web site)

Grouping related statements

- Good idea for defining drill down statements
- Subnumbering
- Better to place in another document as a
  - Glossary item
  - Data requirement – data dictionary

Grouping by system

- High level requirements allocated to a component (subsystem) help establish
  - Project management resource goals for
    - Hardware
    - Software
    - People
  - Interfaces between components

Grouping by ID

- Use the requirement ID to add coded info about the statement
  - SF – system functional
  - PM – project management
  - DAL – data access layer

Workflow description

- Statements **MUST** be of the same granularity within a process description.
- Order gives context to the statement.
  - Events are ordered in event lists.
  - Functional requirements are ordered in use cases
  - Business tasks are ordered in use cases or process flows
- A spreadsheet is not a requirements list.

Process map

- Swimlane diagram, cross-functional process map, Line of visibility model, task on arrow
- A workflow (functional requirement sequence) diagram with trigger and specific steps with inputs, outputs, decision points, and business rules.

Event-response table

- Defines trigger events
  - Business - <role> does something
  - Temporal – system does something at/every <time>
  - Signal - <hardware device> signals sensor
    - Not so good: sensor detects event
- Describes the response
  - Starts a use case
- May have
  - Frequency, delivery method

Non-functional requirements

The requirement without a verb.

Non-functional requirements

- What adjective or adverb describes how all/a subset of the functional parts should behave?
- What details do not affect what the functionality of the system or <role> does?

Non-functional requirement types

- Security
- SLAs (expectations of performance)
- Quality or integration
- Data dictionary – data rules
- Process rules
- Interfaces
- Design recommendations
- Prompts, menus, and messages

Data analysis

CRUD

- The basic atomic (lowest granularity) data operations
  - Create
  - Read
  - Update
  - Delete

CRUD analysis

- Use a CRUD analysis chart when working with entities to discover more requirements
  - That may be missing
  - That may not be complete
  - That may be misplaced

CRUD analysis

CRUD - process vs. entity	Entity type									Comment
	Categories	Subcategories	Products	Product variants	Product related	Specials	Orders	Order items	Order status	
Planning	crud	crud								
Marketing	r	r	rud	rud	rud	rud	r	r	r	
Selling	r	r	r	r	r	r	cru	cru	cru	
Compensation						r	r	r	ru	
Shipping							r	r	ru	
Operations							r	r	ru	
Purchasing			crud	crud	crud	crud				
Forecasting						r	r	r	r	
Receiving			ru	ru	ru	r				
Ordering			ru	ru	ru	r	r	r	r	
Research	r	r	r	r	r	r	r	r	r	

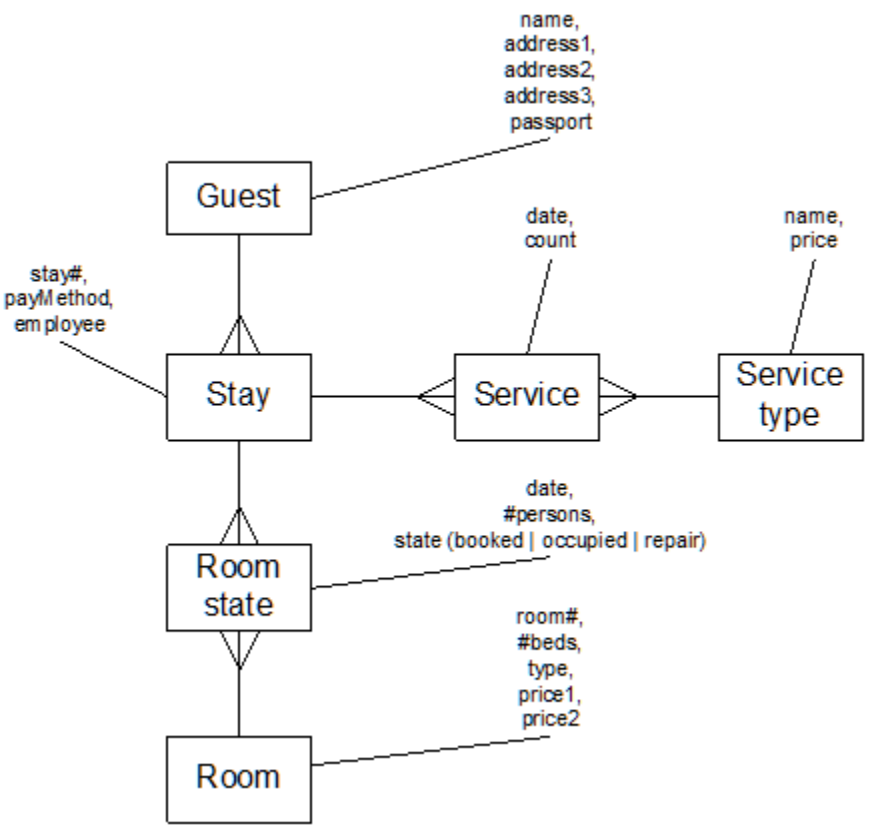
Data dictionary

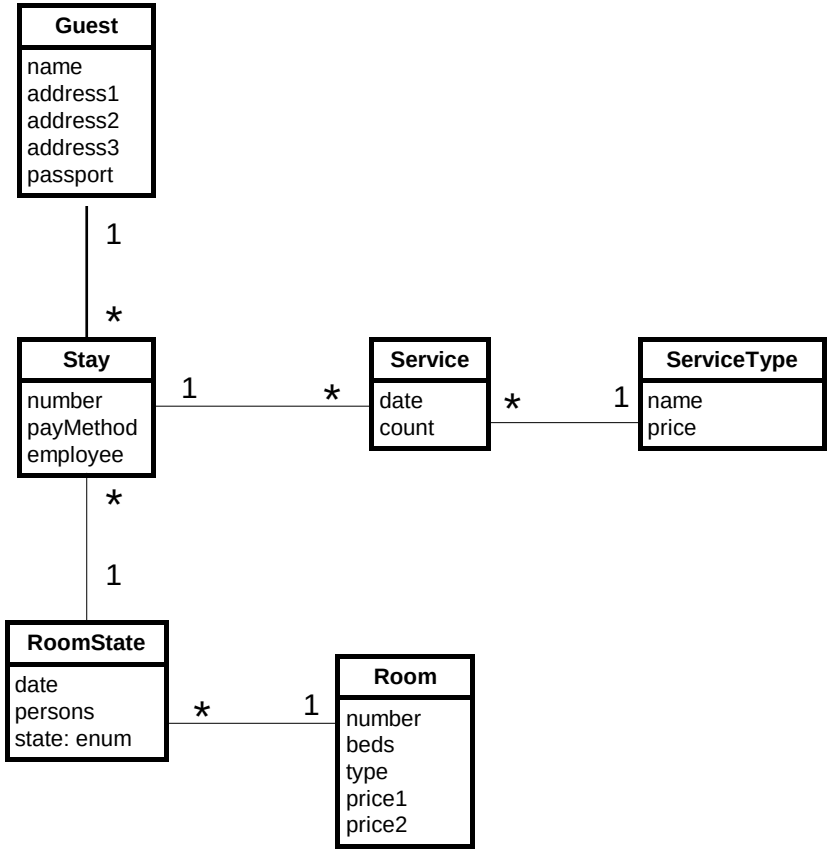
- Business level language
- Know, track, remember, report on..
- Describes
  - Names of individual types of data
    - Name, address, city, state, zip, phone, …
  - Constraints / rules
    - Validation
    - Dependency
  - Related entities
    - Customer, internal rep, …
  - Examples

Affinity diagram

- created in the 1960s by Japanese anthropologist Jiro Kawakita
- Process
  - record ideas/observations and spread on surface
  - place ideas side by side **silently**
  - discuss patterns, shape, motives to move, promote/create a heading idea to group
    - group groups if possible

ERD vs Entity diagram





Data relationship diagrams

- Read two different directions from one entity to another
  - A* --- B
  - A is related to only one B
  - B is related to many As.
- Role
  - Describes how one is related to another entity
  - Default – has a
  - Other – maintains, uses, tracks, owns, sells, delivers, buys, creates, deletes, manages

Data diagram process

- Entity diagram - A high-level UML class diagram
- Discover all of the nouns in a use case.
- Place simple data under complex data
  - **Simple** (fields): Numbers, dates, text, flags
  - **Complex** (entities): People, places, things, roles, events
- Show a line from complex to complex data based on a report where that first complex datum will
  - Need to report on the other datum
  - Need to know about the other datum
  - Needs to be tracked with this other datum

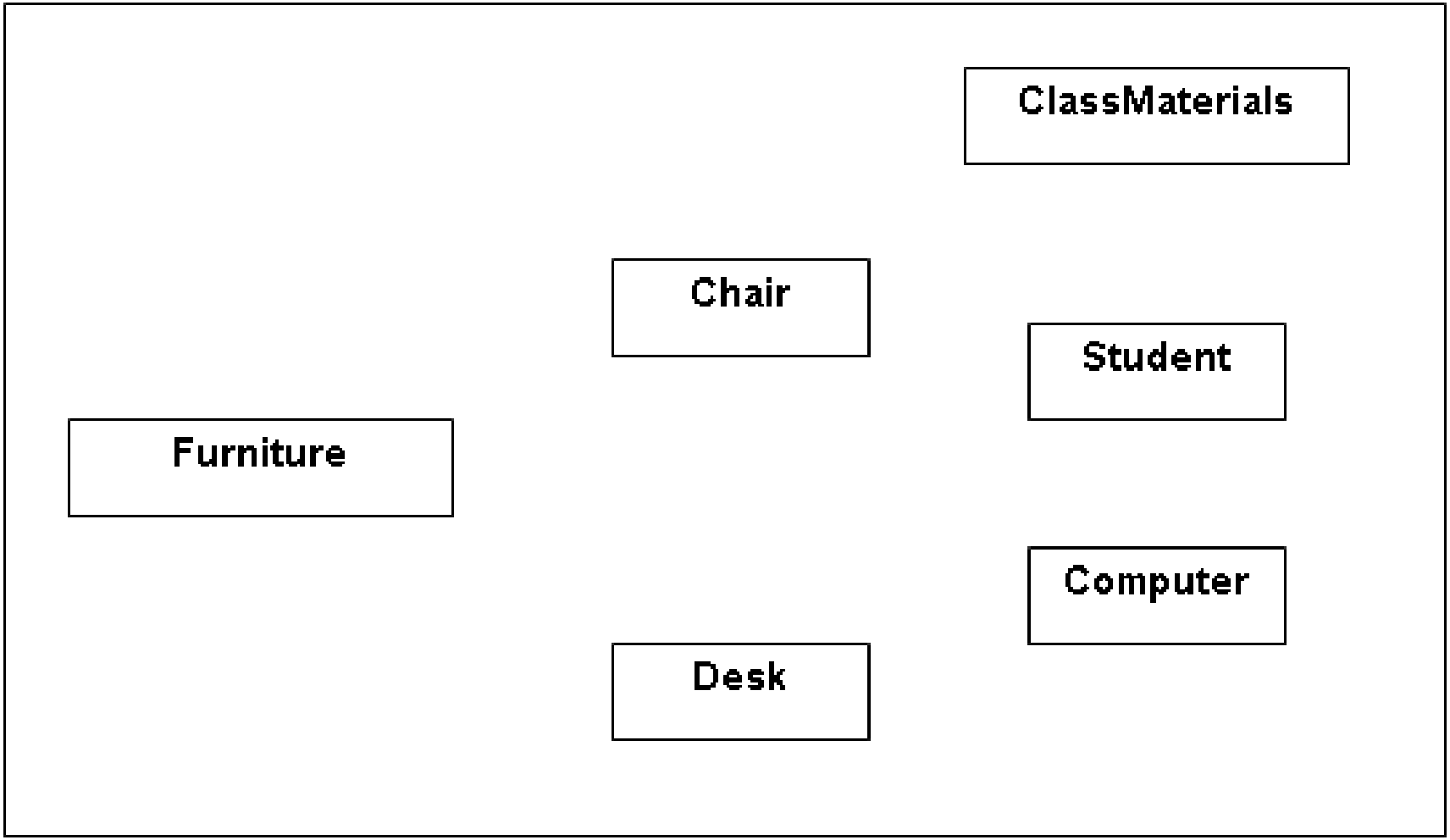
Class diagram

- Show cardinality on the line
  - The first datum will need many (*) of the other datum
  - The * datum will go back and relate to one or more of the first datum.
- Show role on the line
  - A name is common to describe the role
    - Person ---------- has-a ----------- Car
    - Person ---------- sells ------------ Car
- Show direction if needed
  - Dog ---------- <- owns ------------ Person
  - Dog -------- eats from bowl at ->---------- Person

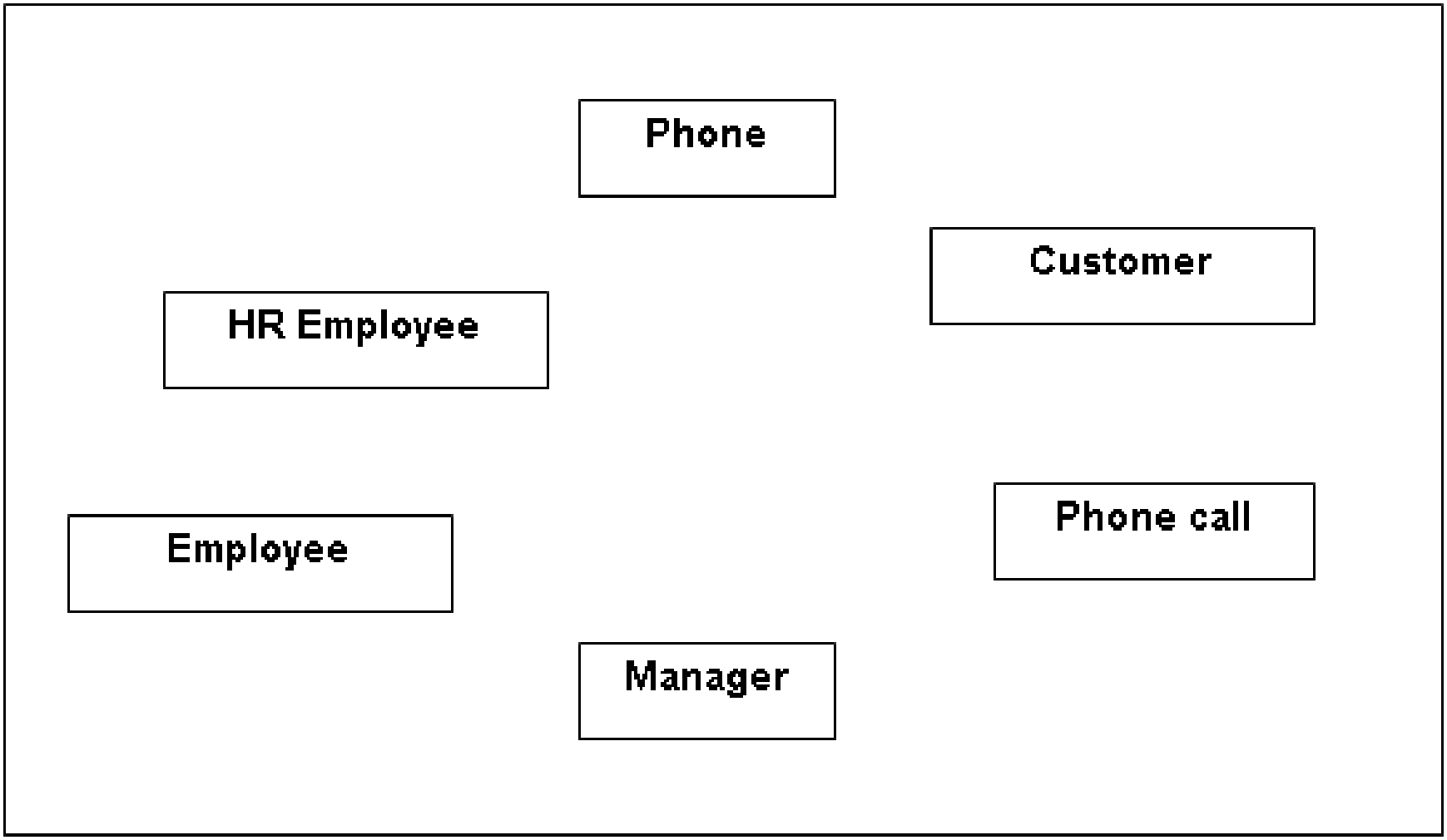
Exercise

- Create the data entity diagram for the ATM system
  - Discover the entities in brainstorming
  - Analyze the entities by
    - Placing simple data types under complex data types
    - Associating complex data types to other complex data types
    - Using cardinality on the associations.

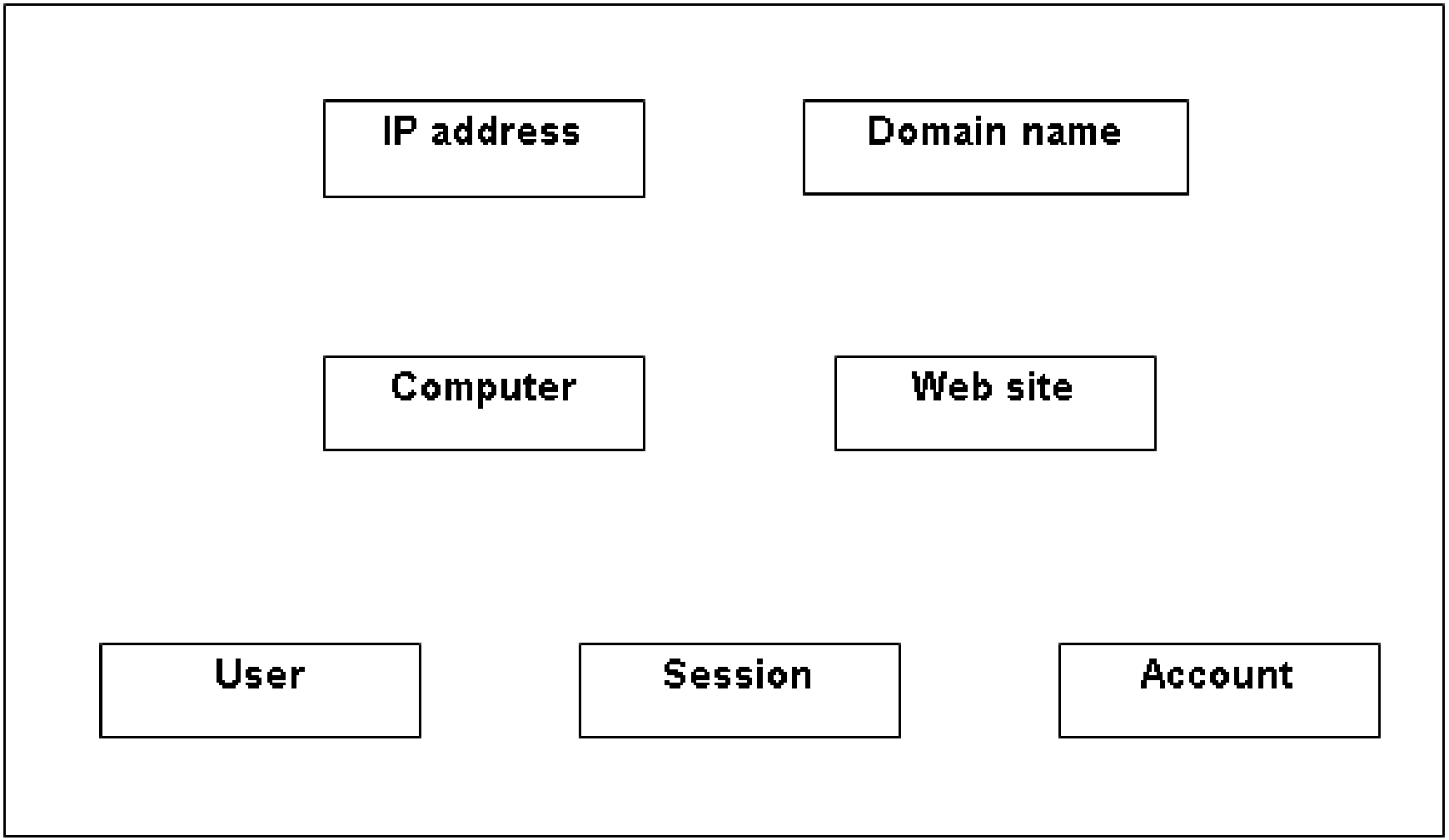
Domain model exercise



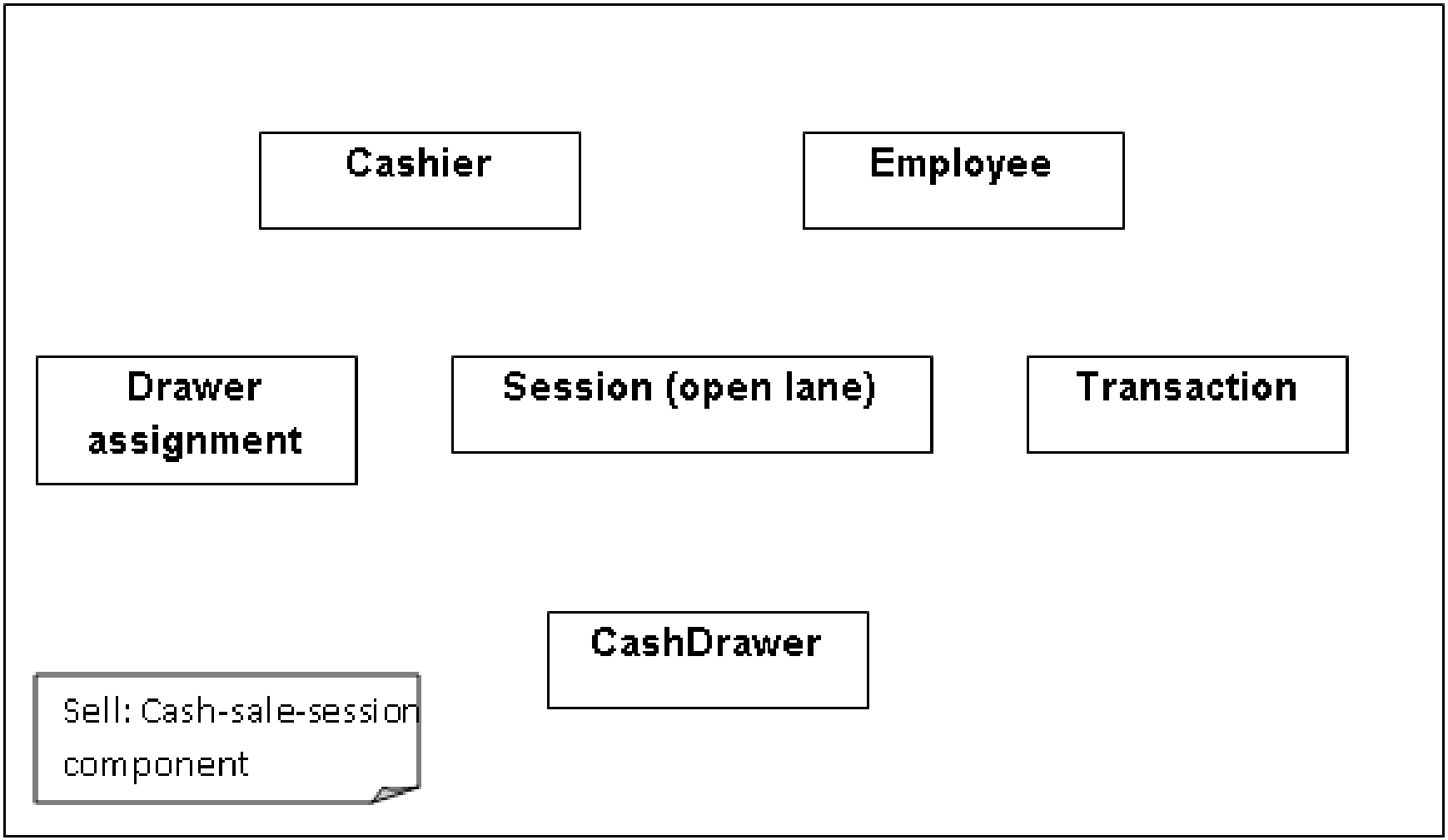
Domain model exercise



Domain model exercise



Domain model exercise



Exercise - data entity diagram

- Manage CDs
  - The system will create, edit, delete, and view the CD data for for title, artists, id number, and tracks. Each track will be show the title and time, artist, composer, label and category. The system will create a sorted report of the CDs.
- Manage artists
  - The system will track individuals in different bands and show the dates associated with that band membership. Individual artist information will show hometown and some description about that artist.
- Manage borrowed CDs
  - The system can track when another person borrowed a CD and how much time they have to return it. The system can create a report for 'overdue' CDs.

Exercise - data entity diagram

- Manage categories
  - The system can add, delete, or update styles with a small description and a date for when that style was popular.
- Manage purchases
  - The system will record the price paid for each CD, the current value of a collection and the store that the CD was purchased.
- Manage tags
  - The system will start with a common set of tags but the user will be able to add, delete, or edit it as needed. Tags will describe categories for the web site, popular genres and key artists.

Exercise 1

- A local University has a need to store grade information for all of its students for all of its classes. Design a logical database structure for this need by identifying all of the entities, attributes, and relationships. This is Phase 1 of this project for the University. Concentrate only on the necessary structures for the grading system.

Exercise 2

- A technical support team tracks equipment for and service calls at each employee's desk including monitors, PCs, and printers. Design a database for them with entities, attributes, and relationships.
  - Each piece of equipment will only be assigned to a single employee
  - When evaluating a service call, record the location for that piece of equipment and document the service tech that was sent to fix it
  - We may be adding other types of hardware in the future like smartphones, tablet PCs, cameras, etc..

Exercise 3

- Your Company uses the following order form to take orders for their products. Design the sales data with all of the entities, attributes, and relationships.
  - The “Bill To” and “Ship To” customers might be different.
  - Shipping charges will be calculated by the application, but the total shipping cost will be recorded.
  - The tax rate will be determined by the application, but the rate will still be recorded.

ORDER FORM

Order Number: 234112
Order Date: September 5, 2000

Bill to: Mary Thompson
1123 Monroe Street
Overland Park, Kansas 66212
913-555-3459

Ship to: Mary Thompson
1123 Monroe Street
Overland Park, Kansas 66212
913-555-3459

Product Number	Product Description	Qty	Price Each	Total Price
100-C02314	36" Ceramic Bird Bath	1	$ 39.99	$39.99
100-C02315	20' Garden Hose	2	$15.99	$31.98
100-B11002	50' Garden Hose	2	$25.99	$51.98
200-A00039	Wave Sprinkler	3	$13.99	$41.97
200-A00119	48" Patio Bench	1	$89.99	$89.99
215-B20020	40" Tomato Cage	4	$5.99	$23.96
399-A00234	50lbs Cedar Mulch Bag	10	$15.99	$150.99

Merchandise Total:	$430.86
Shipping Charge:	$45.00
Sales Tax (7%):	$33.31
Order Total:	**$509.17**

Comments:

Exercise 4

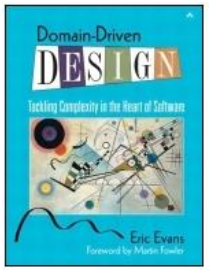
- The AutoGuy dealership sells Toyotas, Hondas and Kias. They keep information about the manufacturers on file as well as the information on the cars. Car info can contain such things as list price, the purchase price from the manufacturer, model name and series. Sales information is recorded about who bought the cars, what they bought, and the amount they paid. Buyer information is stored for future mailings. Develop a model for the dealership based on these facts.

Exercise 5

- You want to develop a system to catalog your media at home. You have various CDs, videotapes, DVDs, and a few records lying around. Of course you store more than one recording on a piece of media and some can appear on multiple items as well as being performed by different singers. You tend to let people borrow them so it is going to be important to track who you lent them to and when they should be returned. Develop a model for a proposed system that will manage your collection.

Resources

- **Domain-Driven Design: Tackling Complexity in the Heart of Software** by Eric Evans. Addison-Wesley Professional, Aug 2003



- **Analysis Patterns** by Martin Fowler
- For programmers
  - Larman, Craig 1998. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design.* Prentice Hall PTR. (get the 2nd version not the 3rd)

Rule analysis

"I'm here because my boss said we should use more decisions tables for our project. What types of decision tables do you sell?"

Business policies

- Derived from business goals
  - Increase yearly repeat business by 25% over last year
- Business policy name
  - Provide discounts to loyal customers

Business rules

- The **functional** part of a requirement
  - The system shall print a report …
- May be modified with a **rule** part
  - **If**/**when** sales are > 100,000 then using rate chart DF3
  - Use rate chart DF3 **when** sales are > 100,000
  - On Thursday
  - When I say so

Business rules

- Process rules = decisions
- Data rules = validations
- Made up of criteria /variables and result/output
  - Criteria – customer type
    - Loyal = purchased items twice in the last 50 weeks
  - Criteria – additional product purchase
    - Product = web hosting
  - Result – discount on additional product
    - Discount = 15%

Business rule quality

- Describes one decision
- Reusable
- Is not a task in a workflow
- Described by measurable facts
  - Criteria - unit and quantity

Rules as boundaries

- Constraints can reduce scope **before** project starts.
  - Can mention automation and allow design into analysis docs.
  - All project level rules
- **Business rules** are active during operations of one project
  - **Workflow** based on variables that produce an output.
  - No mention of any automation.
  - Typically discovered in elicitation.

Rule classification

- Data – Inference (by lookup)
  - KS state tax rate = 5.3%
- Data - Calculation (formula)
  - KS total tax rate = KS state tax + municipality tax rate
- Workflow – restriction (boundary definition)
  - Password must be six characters (reenter)
- Workflow – extension
  - Earnings in KC, MO requires an additional tax form.

Rule writing

Criteria	Criteria	Criteria	Criteria	Result
Client	**Amount due**	**Average purchase**	**Last purchase**	**Type of e-mail**
Region 1 client	<10	100	<50	Follow-up
Region 1 client	<10	100	>50	Follow-up and discount coupon
Region 1 client	>10	100	<50	Follow-up
Region 1 client	>10	100	>50	Follow-up and small gift coupon
Region 2 client	Etc.			

Exercise – rule writing

- Create one rule of your choice with multiple criteria.

Exercise – rewriting bad requirements

- All customers tendered through a retail counter site will be charged counter rates.
- Break-down
  - Start with responsible party: **The clerk/system shall…**
  - Use strong verb for functional: **charge/quote**
  - Find the direct object: **customer/ the clerk / anyone**
- Capture the rules in another document.
- Rethink if it means what they said.
  - **The system shall quote counter rates.**
  - The counter site shall request a counter rate.
  - The clerk shall quote counter rates to customers.
  - The clerk shall charge the shipping rate to the customer.
  - The POS system shall charge the customer's account.

Testing


OUR GOAL IS TO WRITE BUG-FREE SOFTWARE. I'LL PAY A TEN-DOLLAR BONUS FOR EVERY BUG YOU FIND AND FIX.
S. Adams E-mail: SCOTTADAMS@AOL.COM
YAHOO!
WE'RE RICH
YES!!! YES!!! YES!!!
11/3 © 1995 United Feature Syndicate, Inc.(NYC)
I HOPE THIS DRIVES THE RIGHT BEHAVIOR.
I'M GONNA WRITE ME A NEW MINIVAN THIS AFTER-NOON!

Verification

- Correctness – internal view
  - All during the project, KPIs
- Did I write down what the programmers didn't know about?
- Did I get what I know I wanted but didn't say?
- Did we use the right processes along the way?
  - Was the software built correctly? (quality)

Verification tools

- Retrospective
  - After an iteration
- Lessons learned / Post mortem
  - After a project end

Validation

- Correctness – external view
  - The software was correctly built (checklist)
- Did the stakeholder get what they said they wanted?
  - The software was built so that it's usable
- Validation tools – tracing code to requirements

Use case basics

A time-ordered list of workflow tasks.

History

- Ivar Jacobson in the late 1960s
  - Much of the odd terminology comes from translating his Swedish
- Now associated with OO Analysis & Design
- UML diagram
  - But no fixed template for text organization

Definition

- A repeatable
- ordered sequence of tasks
- by an initiating party
- to support a business goal (provides value)

Use cases

- Three types
  - Business - Tasks for employee roles
  - System - Tasks for a system under development or maintenance
  - Blended – Tasks intertwine
- No good distinction in a diagram except to put domain box around system use cases.

What is a use case?

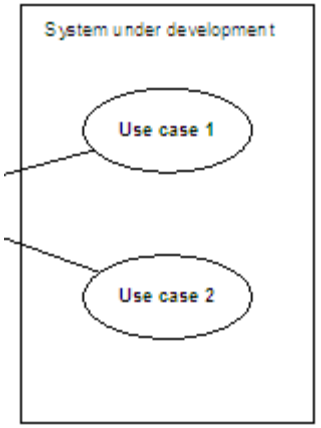
- Focuses on functional requirements
- Non-functional requirements are captured on a case by case basis
  - when important to that specific functional one.
- Requirements must be of the same granularity.

Use cases vs. traditional lists

- The jigsaw puzzle.
  - Needs - puzzle pieces unordered
  - Define the scope – identify the edge pieces
  - The data base – number each puzzle piece
  - The use case – assemble small complete groups of pieces by color
  - The goal – put all the pieces together.
- Small – medium projects – use cases & DD core
- Large projects – also requirements program for reporting, linking to docs, code, graphics, etc.

Domain

- A business term for the **scope** of the
  - Related tasks for employee roles
  - Related tasks for the system under development
- Also related to
  - Business units
  - Subsystems
- Marked with an option box around the use case symbols
- Without constraints



Document styles

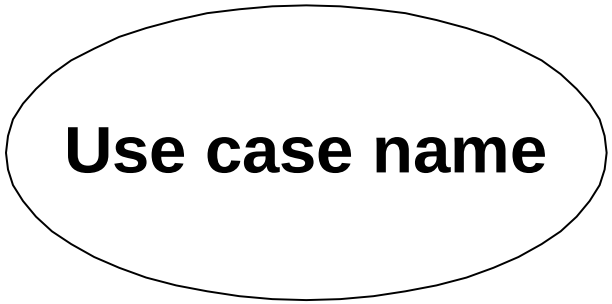
- Informal – the story
  - An elevator speech
  - Use for a table of contents
- Formal – all the facts
  - When it's important to be clear
- Mix 'n' match

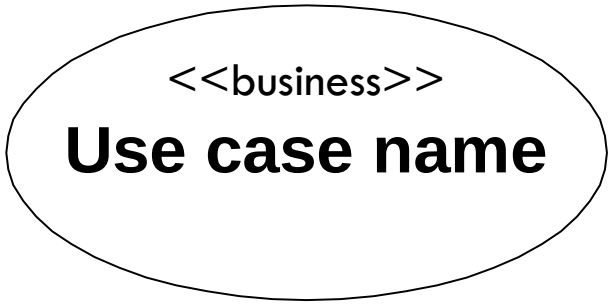
The process

- Gather user needs – elicitation tools
- Analyst only meetings
  - Find actors
  - Find use case names
  - Diagram use case relationships
  - Write use case details
    - Increase detail when important
    - Invite SMEs when unclear
  - Structure if complex
- Review use case with users

Use case names

- Use verb-noun clause syntax
- Use domain words
  - The mapmaker's rule
- Don't goldplate
  - Keep notes to ask users later though



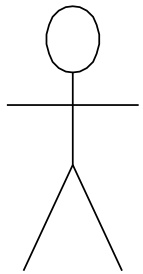


Use case levels

- Aim to deliver business value – a goal
  - Cockburn – user goal level
- Groups of goals will come out easily
  - Break them into individual value parts
  - Use SMEs to help
- Partial goals will distract you
  - Look for what they are a part of
- Groups can organize partials or goals

Actors

- Should have been called roles.
- Actors initiate a use case.
- Actor roles enforce the ability to do processes
  - Actors describe security group names that have **permission** sets.
  - For any two actors, one will have a unique use case that the other doesn't do.



Actor name

The actor table

- Identifies and classifies system users by roles and responsibilities
- Includes
  - Names of actual stakeholders (people/systems)
  - Description
  - Related job titles
  - Location
  - Level of expertise
  - Domain expertise
  - Frequency of use

Personas – Actors in design

- Used by Alan Cooper for interaction design and copied by software industry after 1998.
- Defined by goals, revealed through roles in analysis.
- Has explicit personality traits to represent the "actor"

Exercise: Actor table

- Use **user** and **admin** (easy)
- Or use all different actor names (hard)
  - Each must have a unique use case that no one else can do.
- Or something in between.

Use case validation

- Does your complete scenario meet the walk-away test?
  - Walk up, do it, walk away
- The business gains **value**
  - What was the **goal** that was achieved?
- Does the system return to the **same state** it was in before you started the use case?
  - I can do it, then you can do it.
  - If not, then it's part (task, function) of a bigger use case
- Strong actionable verb
  - Vague verbs indicate a **group** of use cases

Use case validation

- Can the **system** functionality be applied to a phone interface?
  - If you talk about a GUI then that's design.
- If you talk about **how** something is done, that's a rule.

Too much in a use case

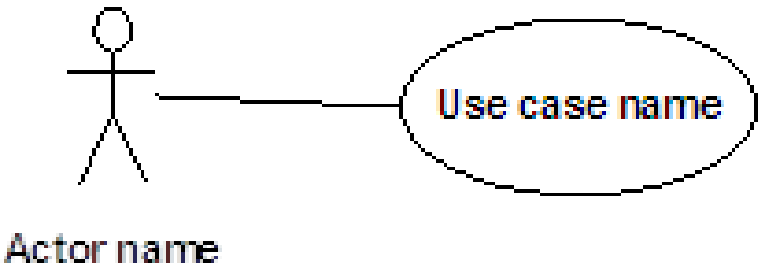
- Any use of conditional logic for workflow will indicate separate use cases when
  - The outcome is different
  - Steps are skipped
  - Steps are not always included
  - A rule is used to alter workflow

ATM exercises

- Brainstorming
  - Round-robin questions first, then more open discussion
  - No criticism
  - Then apply the rules after you run out of ideas
  - Humorous ideas don't end the session, they sometimes spark new results.
- Actors table – candidates
- Use case summary – candidate and confirmed
- Actors table – confirmed

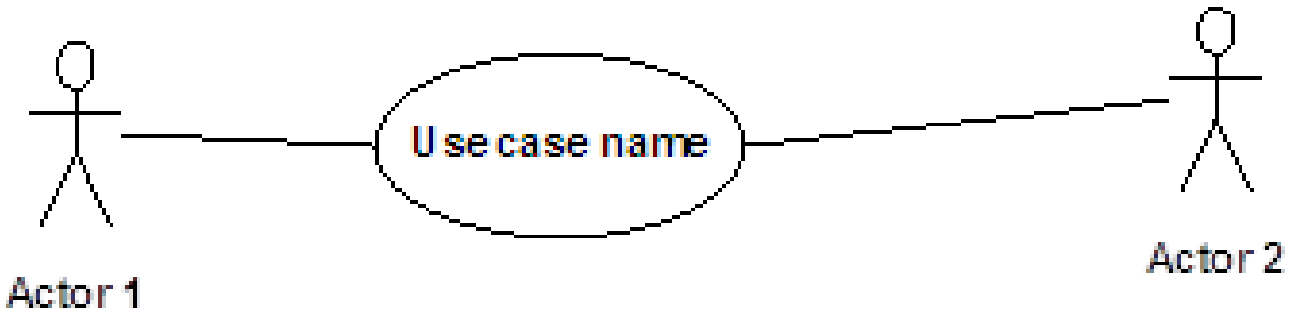
Use case diagrams

- Only to show scope, granularity and triggers!
  - No sequences.
- Split diagrams into readable sections.
- Only show actors who initiate use cases on diagrams.
- Multiple domains may be necessary.



Use case diagrams

- Place actor on either side, but no directionality.
  - Not a flow or sequence diagram, but a scope diagram.
- Keep lines from crossing when possible
- **Exercise** – use case diagram



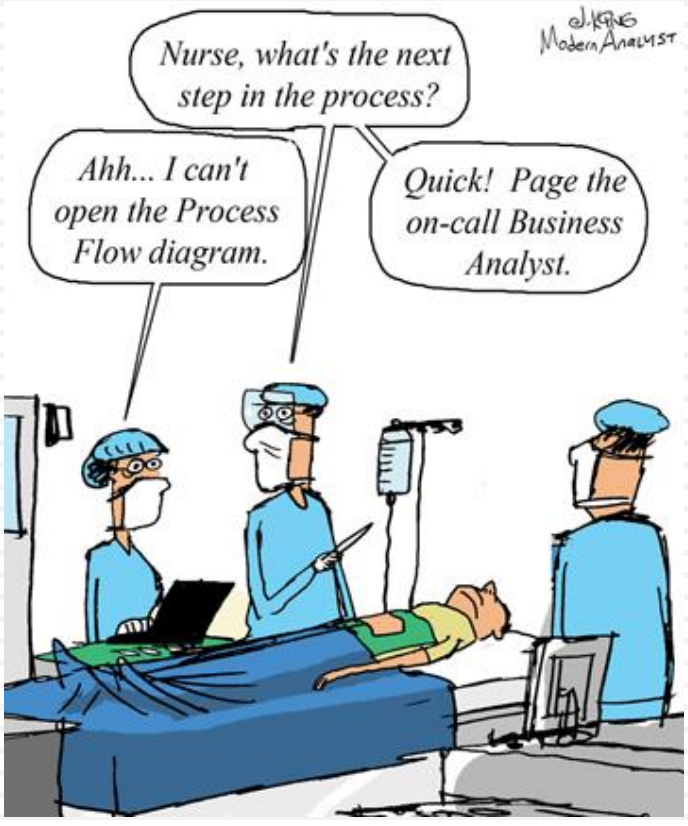
Use case diagram tools

- Text driven
  - https://yuml.me/diagram/usecase/draw

Use case prioritization

- Well-scoped use cases can be used by project managers to estimate project size.
- Priorities can help meet schedule and budget
- **Exercise** - prioritization

Use case detail



Use case detail

- Always a “happy path”
  - A success scenario
  - Problems will be captured later
- No conditionals
  - No if-then-else statements
  - Multiple partial sequences (loops) should be expressed as optional parts.
- Detail level
  - as much detail as possible without design
- No design (without constraints)
  - e.g. button click, submit buttons or anything that connects system to hardware, software, tools, or materials

Use case detail

- The course of events
  - The use case starts when the actor …
  - Response: The system …
- Possibly multiple actors could initiate the use case
- Numbering
  - Group one or more statements/tasks together
  - Smaller increments are better when you need to start in the middle at a specific spot due to an error.
  - Start with system does… usually.

Using references

- Move out the details that are not functional
  - Small detailed parts are OK for clarity.
    - Sub points, mark the type
  - Use the specific document to capture reusable or complex rules, designs, etc.
    - Rules
    - Data dictionary
    - Designs - menus, screens
    - Externalized text – prompts, error messages
  - Use character style to show rules & data dictionary items

Pre- and post-conditions

- Constraints
- Pre-conditions
  - block the use case from doing the first step.
  - validate the state of the software before anything happens specific to that use case.
  - A log on is not a pre-condition in a system use case
- Post-conditions restate the important points connected to the goal.
  - Optional usually
  - Minimal vs maximum success metrics

Special requirements

- Put things like SLAs and location or time needs in a special category.
- Non-functional requirements that are specific to this use case should be documented with the use case.
- Admin people can understand why a requirement should be met.
- Notes

Project documents

- Project plan
  - Systems involved
  - Actor summary
- Data dictionary
- Requirements - Use cases
- Optional
  - Designs or Layouts
  - External text
  - Business rules

Use case peer reviews

- Completeness check – role playing game
  - Person = bank, ATM, database, other roles
  - Data represented on Post-it notes / listed on pad
  - Designs sketched on paper, hold up when active
- Testability check -play out differently
  - Use data range boundary values
  - Substitute more extreme values
  - Repeat more times
  - Make environment worse

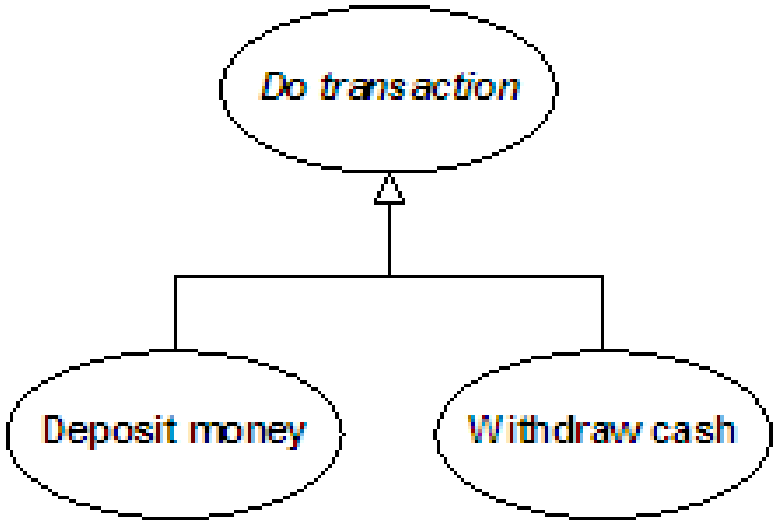
Exercise – use case detail

- Create pre-conditions / post-conditions if any
- Do course of events for Withdraw Cash
- Create sections and reference for
  - Data dictionary [DD] or just bold text
  - Business rules [R#] or subpoint – and bold text
  - External text [T#] – internationalization – il8n
  - Designs [D#] – prototypes
- Create alternative flows
- Create error flows

Use case structuring

Groups

- Use grouping when it cleans up a use case diagram and makes it easier to understand.
- Can be groups of goals or groups of partials
- No use case details are made, just a summary.

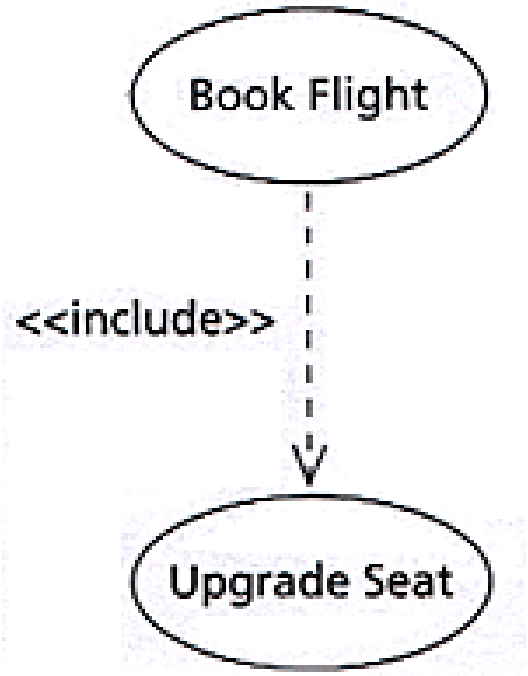


Include and extends

- **Include** is required (dashed arrow goes to what is required to be done)
- **Extends** is optional (dashed arrow goes from what can be done to what it must be a part of)
- Do not use these until all scenario level use cases have been detailed!

Includes

- Use special includes when
  - users expect that name (log in)
  - options are important to see
- Shown as use case ellipse with a dependency arrow and stereotype (category surrounded with <<European quotes>>).
- Include is a partial goal level, not goal level.



Extends / alternative flows

- Two types
  - **Extension** points return back where you came from after an optional set of steps. <<extends>>
  - **Failure** points stop the use case, return you to a different point, or fix the problem and let you continue.
- Write in your choice of styles (informal, formal)
  - **Bad thing happens** (13, 15) – try to fix and return to 12.
- Include a return point or end the use case.

Include vs. different use case

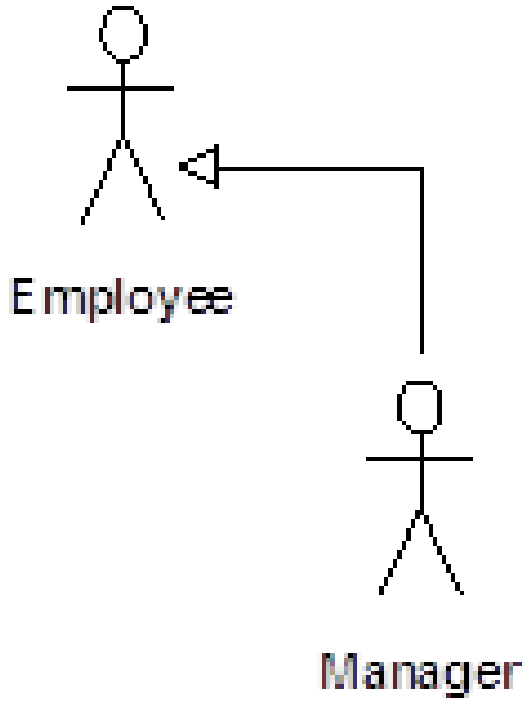
- Write includes into use cases as bolded names in course of events
  - **Do Log On (SF24)** workflow returns here
  - <<includes>> **Do Log On (SF24)**
- Alternate flow (extends) – separate section not in course of events
  - Optional partials
    - **(#3) Print Receipt (SF33)**
    - (#3) <<extends >> **Print Receipt (SF33)**

Structuring for security

- Security is a use case wrapper around other use cases
  - Start session (authenticate)
  - <<include>> Do secure process
  - End session (clean up and deaccess)
- This is allows for <<extends>> Do another secure process as an option
- Don't do
  - <<include>> Start secure session
  - Process tasks…
  - <<include>> End secure session

Another generalization

- Actors can share use case initiation
- Show with generalization arrow



Exercise

- **Structure** – create Do Secure Session and a partial <<includes>> for Withdraw Cash
- **Short** – detail the Transfer Funds use case
  - Use the previous exercise as guide.

Use case numbering / id symbols

- By system component (ATM1, ATM2, B1, B2, INV1)
- A group of use cases
  - *ATM2 – Do A Transaction (generalized partial goals)
  - ATM2.1 – Do Withdrawal (one partial goal
  - ATM2.1.5 Do Withdrawal step 5
  - ATM2.1.5b Do Withdrawal step 5 and 2nd system requirement in step.
- A partial use case with system security
  - SS+PUR2.1 Check shopping cart contents (include this)
  - +*PUR2 Check account property (include one of a group of partials)
  - SS#5+PUR13.1 – Purchase item but check shopping cart contents at step 5. (include which focuses on one step)

Exercise - long

- Create an application and do the full process with one or two detailed use cases. Check with instructor at each process step.
  - Candidate actors
  - Candidate use case names
  - Validate actors and use case names
  - Draw diagram
  - Pick one use case to detail.
- Order A Book (from Amazon)
- Here's the beginning for a general user…
  - System greets user. User searches for book.
  - System gives search results. User selects book.
  - System gives book details. User adds book to cart.
  - System confirms adding book.

ATM solution

- Explanation of solution
- How an ATM works
  - http://www.sciencechannel.com/tv-shows/machines-how-they-work/machines-how-they-work-videos/the-inside-of-an-atm/

UML - Use case diagrams

UML

- The current standard way to describe business processes and software
- Diagrams
  - use case
  - class
  - collaboration
  - interaction sequence
  - activity
  - state chart
  - implementation

Evaluation

- http://mtm.cebglobal.com/centriq1