Class 2 CS540

Gregg Vesonder

Stevens Institute of Technology

Roadmap- Class 2

- Clarifications from last class
- Journal entry
- Brooks - Mythical Man Month
- Software Project Planning
- Requirements
- Reading: chapters 3 (tosection 3.9) in BY, preface and chapter 1 in Brooks
- Reading next week: sections 3.9-3.10 and chapters 5&6 in BY, chapter 2 in Brooks

Class 1 Clarifications

- Next 2 slides!

An Example

- 1983: FAA AAS (Advanced Automation System) - overhaul of computers, radar units, software and communications network
- 1988: contract awarded to IBM with budget of $4.8B completed by 1994
- December 1990: project was 19 months behind schedule
- Late 1992: project was 33 months behind schedule, cost now $5.1B, project scaled back
- December 1993: cost of smaller project $5.9B
- April 1994: independent commission project has "high risk of failure"
- June 1994 project shifted to Lockheed Martin, further slimmed, now $6B and estimated for completion in 2000
- September 1996 Lockheed announced a new project STARS (Standard Terminal Automation Replacement System -estimates of AAS costs ranged from $7.6-23B.

Software Crisis Persists

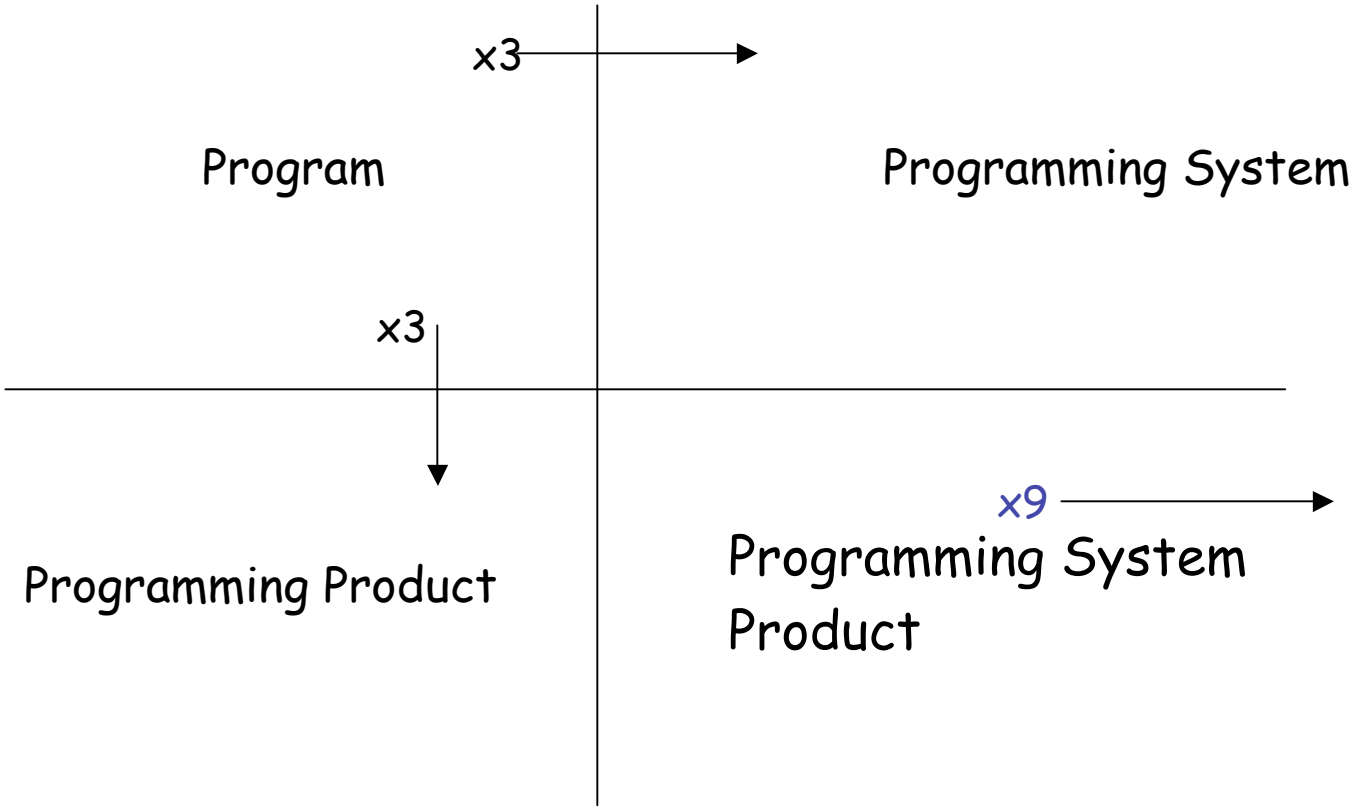
http://www.standishgroup.com/sample_research/chaos_1994_1.php

- (in mid '90s) $250B over 175,000 projects
  - Average cost of projects: large company $2.3M, medium $1.3M small $.4M
- Almost a third will be cancelled before they are completed
- Over half will cost 189% of their current estimates
- Only 16.2% will be finished on time and on budget
- IBM FAA Air Traffic Control project, Mars probe …
- More disasters in van Vliet, add your own
- From paint to building -- long life span
- the University of Oxford has this remark in their Engineering Science description, "The qualities of a good engineer include not only a high degree of technical competence but also imagination, strength of purpose, commonsense - and a social conscience."

Log Book

- Emergent properties
- Your entry

Brooks: System Production



Why?

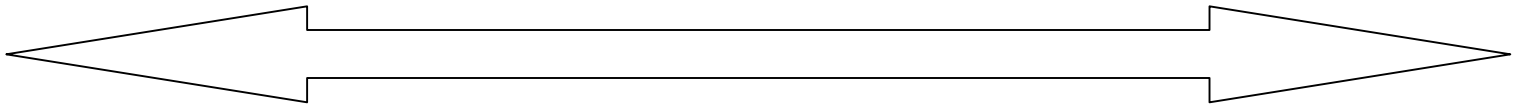
- Making things
- Making things useful to others
- Joys of grappling with complexity
- Always learning
- Mind stuff
- (some downside: perfection, control, debugging)
- Applies to most systems, not just large systems, especially today since we are building "large" systems with small teams

Project Planning and Control

- Control is THE KEY for the lead manager
- Requires clearly defined Roles and Responsibilities
- 3 types of variables:
  - Irregular, cannot be varied, e.g., staffing,
  - Goal variables, project based not problem based, e.g., quality, time, costs
  - Control variables, what you can control
- Must be able to measure the progress of the product and the quality and requirements covered in the product

Project Space - van Vliet

	Realization	Allocation	Design	Exploration
Product Certainty	HIGH	HIGH	HIGH	LOW
Process Certainty	HIGH	HIGH	LOW	LOW
Resource Certainty	HIGH	LOW	LOW	LOW



Waterfall

Proto

Risk Management

- General signals: new domain, new developers, new management, dictated schedules …
- Risk Management process
  - Identify risk factors
  - Determine risk exposure
  - Develop strategies to handle risks (avoidance, transfer, acceptance)
  - Handle it

Top 10 risk factors (Boehm)

- Personnel shortfall
- Unrealistic schedule or budget
- Wrong functionality
- Wrong user interface
- Gold plating (fun and games)
- Requirements volatility
- Bad external components
- Bad external tasks (subcontracts)
- Real time shortfalls
- Capability shortfall (bleeding edge)

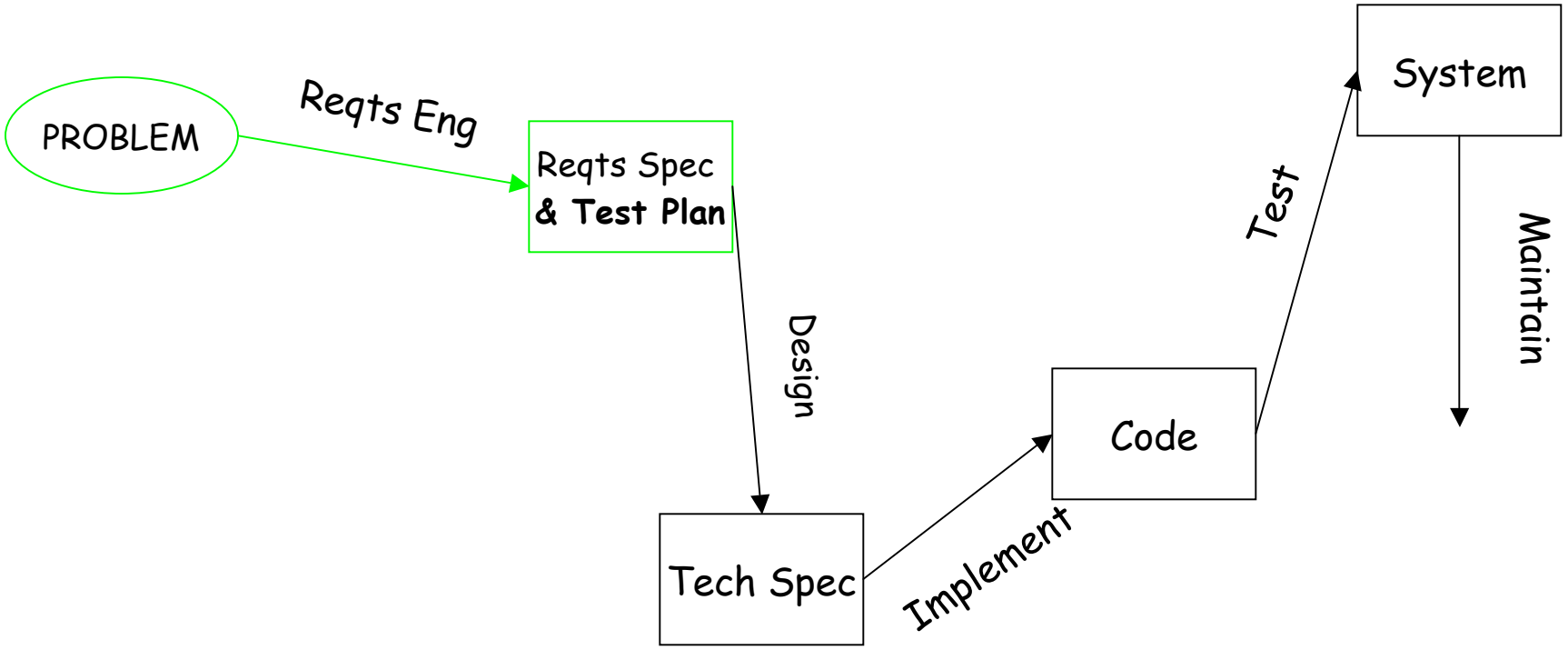
Techniques for Project Planning

- Some sort of work breakdown structure, tasks into subtasks with constraints
- Beware of over and under analysis
- Beware of diffuse responsibility
- Gantt chart - Microsoft project (do not represent dependencies between activities)
- Identify critical path activities (should know w/o automation)
- Sensitivity analysis - what if questions
- Also informal methods -- milestones

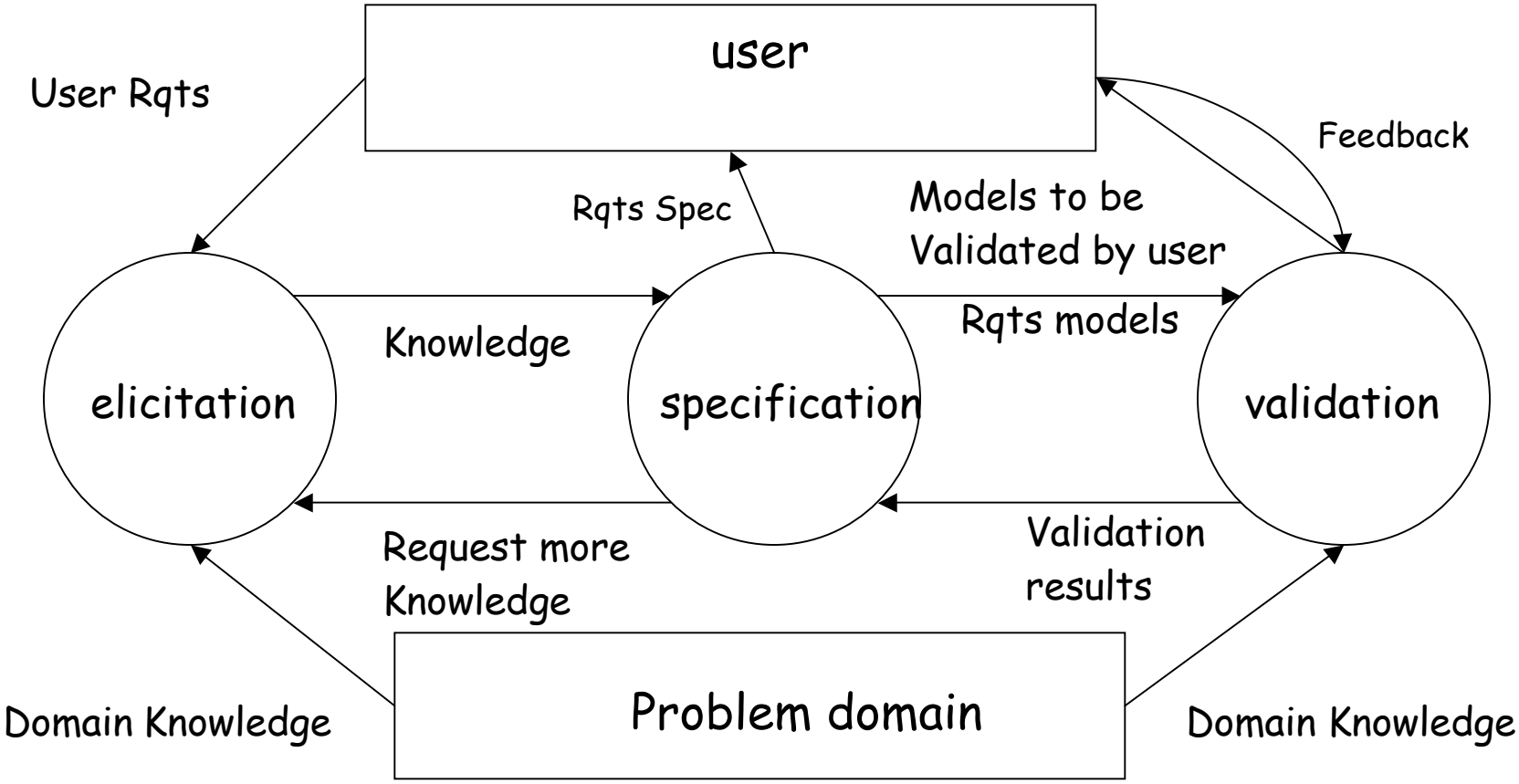
Project Planning Heuristics

- Never plan for heroics
- Be mindful of the 4th Quarter
- Development Plan with milestones
- Know your team
- Informal checks every day, heartbeats once a week
- Worry beads

Simplified Model



Requirements Engineering (van Vliet)



More on Requirements

- Requirements are the what and why but …
- "… it is an illusion to expect that perfect requirements can be formulated ahead of time" - (Endres & Rombach, p. 15)
- Outsourced products require careful requirements - key in today's world
- All stakeholders must participate

Importance of Requirements

- Glass: "Requirements deficiencies are the prime source of project failures." (Endres & Rombach)
- Some studies:
  - Denver airport baggage handling
  - Welfare administration system for Florida
  - FAA air traffic control system
- Too many, unstable due to late changes, ambiguous, incomplete
- Boehm: "Errors are most frequent during the requirements and design activities and are more expensive the later they are removed." (E&R) - cost of errors is high the longer they stay in the project.

Some Success Points

- Should expend 15-30% of effort on requirements
- Requirements should be assigned priorities
- Traceability should be enforced throughout
- Validation! (and verification before)

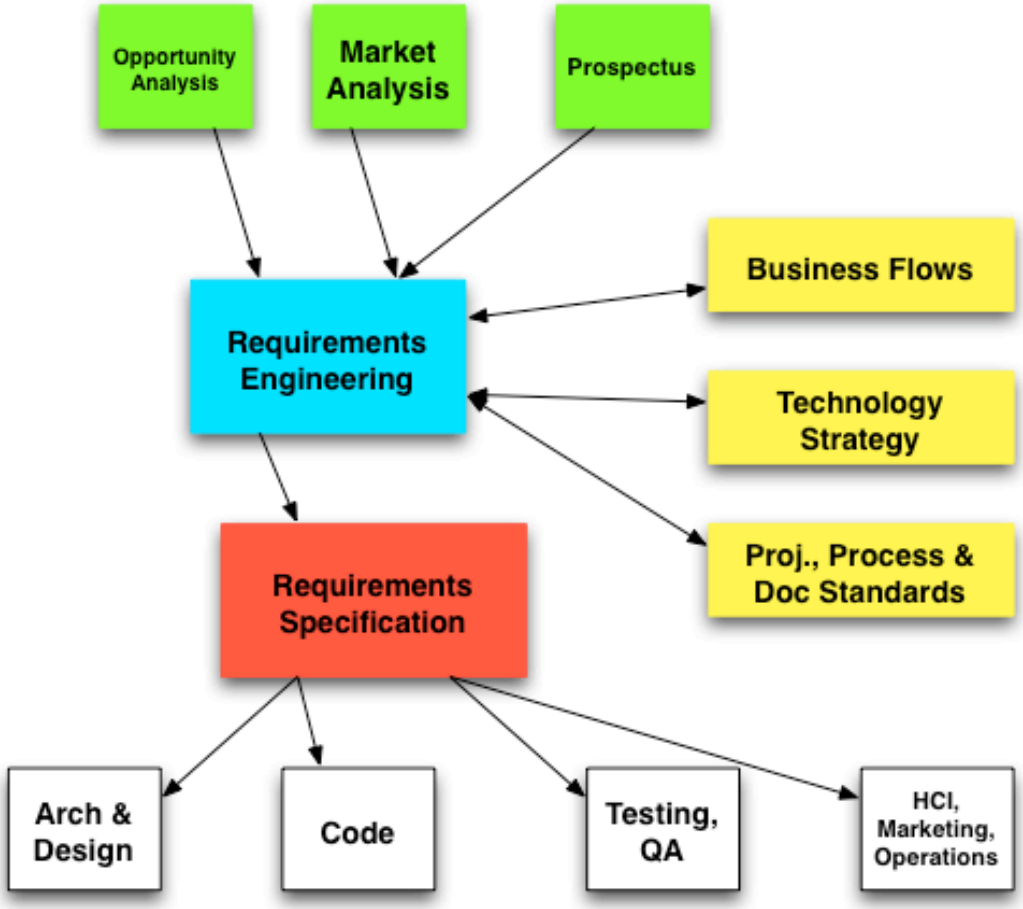
What Can Go Wrong

- Miss or misunderstand a majority of the essential requirements - prototyping and other elicitation techniques are helpful
- Endless requirements - requirements tries to have a cutoff point
- Sales team (or management) interferes and confuses what is desired with what is required
- Spend little time on user interface requirements - what does the user see in the end, eh?

The Requirements Process

- Elicit feature or functional requirements from the user (Boehm -as much as 40% of final features not in requirements specification)
- Understand constraints and non-functional requirements - many are 'ilities
- Analyze the requirements (use cases) to make sure they flow and make sense
- Develop prototype, model or user documentation
- Produce and control a requirements specification

Requirements Engineering



Requirements Elicitation

- Implicit conceptual model of users becomes explicit
- Universe of Discourse (UoD) part of reality we are interested
- Requires us to become quick learners but
- Much of knowledge is
  - Knowledge taken for granted
  - Tacit-knowledge skillfully applied in the background, not verbalized
  - Involves habits, customs, inconsistencies
  - Influenced by frequency and recency
  - What's needed may be different from what exists
- Don't automate an undisciplined work flow - PMO (Present Method of Operation) versus FMO (Future…) - Brooks

Requirements Elicitation Techniques

- Asking: interview, questionnaire, structured interview, Delphi (group based)
- Task analysis: hierarchical decomposition
- Scenario based analysis: instances of tasks, use-case (not only for OO)
- Ethnography: studying folks in natural setting

Requirements Elicitation Techniques (cont'd)

- Form analysis: existing forms (may carry over in DTFs)
- Natural language descriptions: training, manuals, …
- Derivation from existing system
- Domain analysis: study existing systems w/in domain, reuseable components

Requirements Elicitation Techniques (cont'd)

- Business Process Redesign - radically redesign the processes, information processing systems should enable
  - At the very least rethink the existing process (PMO, FMO)
- Prototyping, Modeling
- Understand System of Systems - Description of the environment
- Depending on system - MOV, Measurable Operational Value
- Usually it is a mixture of these

The most important outcome of product definition … a team of stakeholders with enough trust and shared vision …"
BY, p.85

NOT EASY!

COMMUNICATION is the key

More on Prototyping

- Boehm: "Prototyping (significantly) reduces requirements and design errors, especially for user interfaces." (p 19, Endres & Rombach)
- Types of prototypes:
  - Demonstration prototype - clarifies user requirements, impresses user (GUI)
  - Decision prototype - helps select design alternatives, answers particular questions (design/construction phase)
  - Educational prototype - used to understand technology and its characteristics (performance). Differs from Pilot projects that test a new technology that may or may not be included in the product

Still more on prototyping

- All three are throwaways - recall from last week that after proto design can begin (and you may use decision protos to help)
- In ENGINEERING most prototypes use different materials that clearly separate it from final proto, e.g. wind tunnel models … only in software are folks tempted

Expert System Process

- AKA Knowledge Engineering
  - Knowledge Engineer becomes familiar with domain, architecture and operation
  - KE meets with experts to understand operations and issues
  - Team uses knowledge to create first (and subsequent) passes at rules
  - Experts critique results, provide new knowledge and iterate on previous step until a satisfactory (or best possible) conclusion is achieved

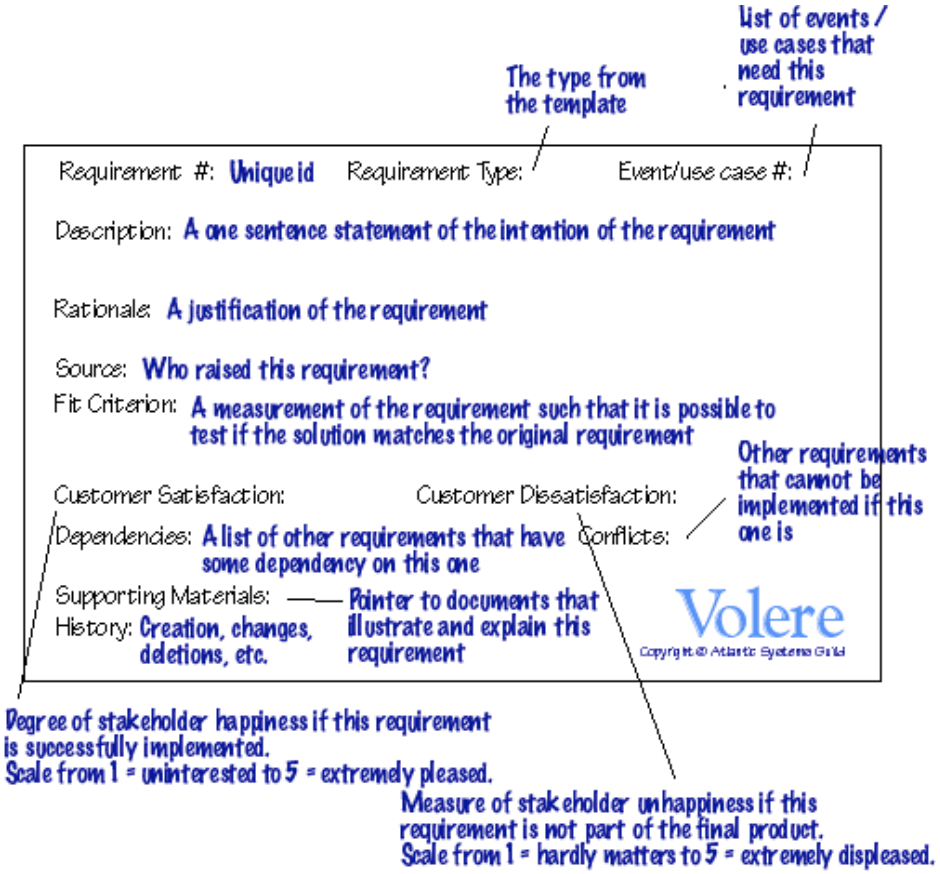
Use Cases - Ivar Jacobson

- Capturing requirements from the user's point of view (and then we adapt) in manageable chunks (OO motivated)
- Actors are entities that reside outside the system and should not be other use cases
- Use case example - "withdraw a book" following normal path and alternate paths
- Need a comprehensive set

The Requirements Specification

- Key output of the process
- Must be baselined -- it will change and this must be measured
- Requirements spec should be readable, understandable, correct (validated against other docs), complete, internally consistent, ranked for importance or stability, verifiable, modifiable and traceable
- Used even in prototyping to describe outcome
- Serves at least two groups, the user and the designer

Recording Requirements



http://www.systemsguild.com/GuildSite/Robs/Template.html

Volere Requirements Specification Contents

- Product Constraints- restrictions and limitations that apply to project and product
- Functional Requirements- the functionality of the product
- Non-Functional Requirements - the product's qualities
- Product Issues-apply to the project that builds the product

Volere Product Constraints

- Purpose of the product
  - User problem
  - Goals of product
- Client, customer and other stakeholders
  - Client = pays and owns
  - Customer buys
  - Other stakeholders, e.g., management, business SMEs
- Users of the product
  - Users
  - Priorities assigned to users (key, secondary, unimportant) see also what volere calls stakeholders

Product Constraints (cont'd)

- Requirements constraints
  - Solution constraints -Windows XP
  - Implementation environment
  - Partner applications
  - COTS-commercial off the shelf software (early)
  - Anticipated workplace environment
  - Known deadlines -how long
  - budget
- Naming conventions and definitions
- Relevant facts
- Assumptions

Volere Functional Requirements

- The scope of the product
  - -context of work, domain knowledge
  - Work partitioning, event list with inputs and outputs
  - Product boundary(users: active, autonomous, cooperative)
- Functional and data requirements
  - Functional - fit criterion (as many place as possible - did I do what I said I'd do)
  - Data (model)

(simple, eh?)

Volere Non-Functional Requirements

- Look and Feel Requirements
- Usability Requirements
  - Ease of use
  - Ease of learning
- Performance Requirements
  - Speed
  - Safety critical (robotics)
  - Precision
  - Reliability and availability
  - Capacity throughput, volume, cpu/memory load

Non Functional (cont'd)

- Operational Requirements
  - Expected physical environment
  - Expected technological environment
  - Partner applications
- Maintainability and Portability Requirements
  - How easy? {how much effort or budget}
  - Special conditions for maintenance
  - portability
- Security Requirements
  - Confidentiality - restricted access
  - File integrity/ system integrity (sync'd)

Non Functional (cont'd)

- Cultural and Political Requirements
- Legal Requirements
  - Law pertaining to product
  - Standard compliance

Volere Project Issues

- Open issues
- Off-the-shelf Solutions (early)
  - Existing product
  - Ready made components
  - Copy or model
- New Problems
  - Problems the new system can cause
  - Affect on current systems
  - Users adversely affected by system
  - Limitations in the anticipated environment that inhibit new system
  - Other problems?

Project Issues (cont'd)

- Tasks
  - Steps to deliver product
  - Development phases {Development Plan}
- Cutover
  - Special requirements to get data and procedures
  - Data modification/translation necessary
- Risks
  - What are they
  - How do you address them

Project Issues (cont'd)

- Costs
- User Documentation
- Waiting Room
- {Worry Beads} - daily, weekly, monthly

Meyer's Approach

- Natural language has problems: noise, silence, over-specification, contradictions, ambiguity, forward references and wishful thinking
- Use formal techniques then translate to natural language
- Some formal techniques: entity-relationship modeling, Finite state machines, Structured Analysis and Design Technique (SADT)

Modeling Approaches

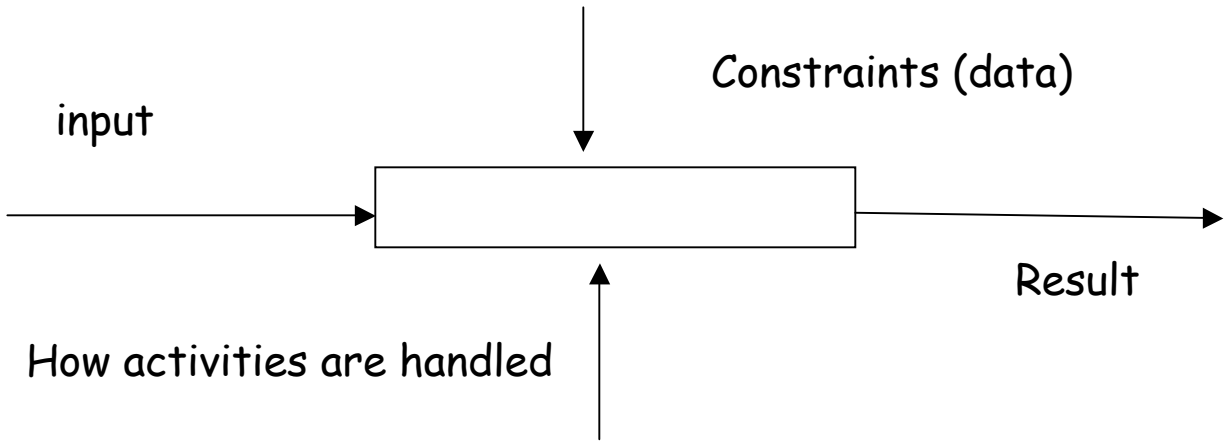
- None are truly comprehensive, usually done to model aspects of the problem.
- Very time intensive
- Some require buying into a full methodology
- Difficult to reflect change and therefore keep synchronized
- Entity-Relationship (Chen), FSMs and SADT (Ross), Data Flow modeling
- Surveying blackboards and white boards usually the pictures are much less formal
- Davis: "The value of a model depends on the view taken, but none is best for all purposes" (E&R)

Entity-Relationship (Chen)

- Modeling data, used in MS Access
- Entity is a thing represented as a rectangle (e.g., employee)
- An entity has attributes represented as ovals, e.g., address, SSN
- Entities are linked through relationships represented as diamonds
- Restrictions are placed on the cardinality of relationships, 1-1, 1-n, n-m

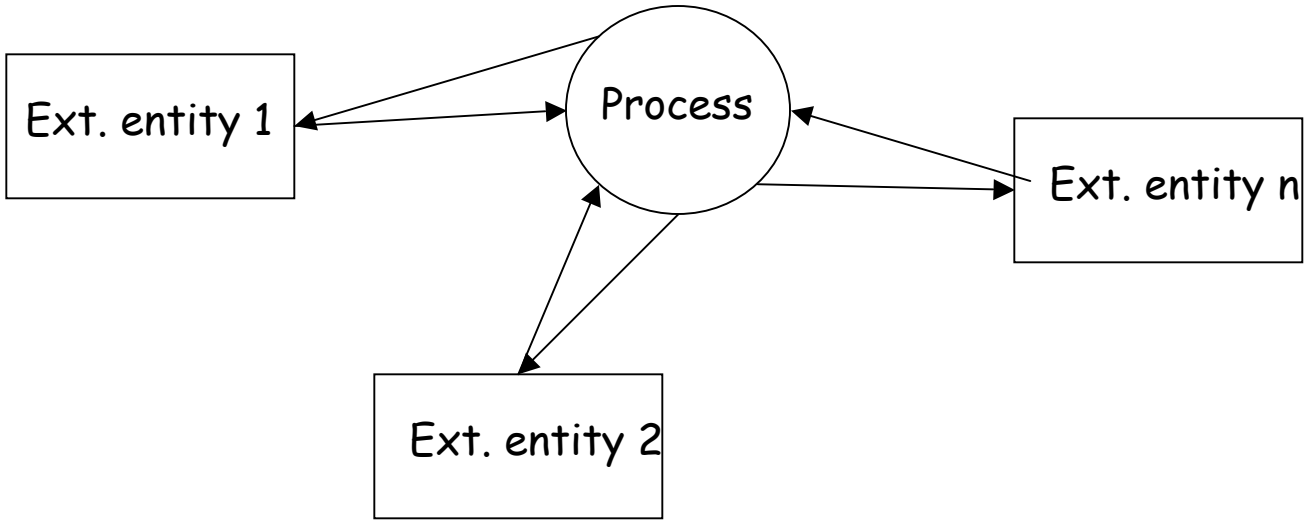
SADT (Ross)

- Really a way of life- covers interviewing through system description
- System viewed from different perspectives - activity viewpoint closest to other techniques



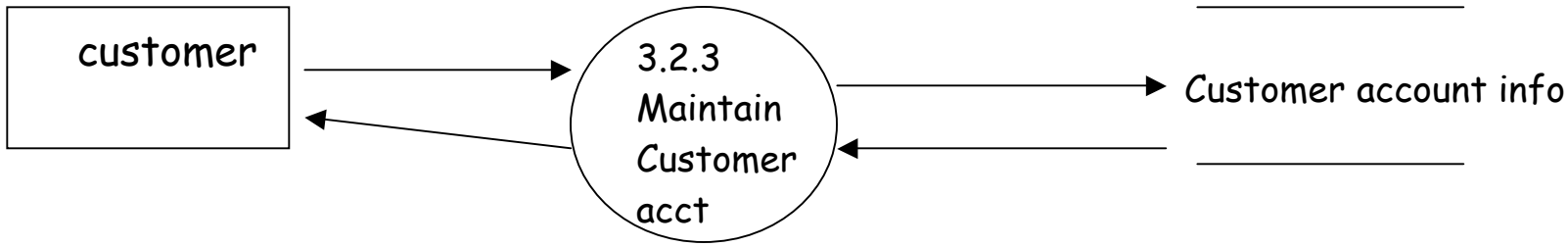
Data Flow Modeling

- Lots of variants. Begins with a Context Diagram that represents function of whole system in relationship to external entities



Data Flow Diagrams continued

- Hierarchical with rectangles representing external entities, circles processes and bars data sources.



Finite State Machines

- System is modeled in terms of states and transitions, states are circles, transitions are arcs
- Modeled in state transition diagrams
- Sometimes used in a hierarchy, a state can expand to a succession of states, e.g., employee paid-> paycheck calculated->taxes calculated
- Again modeling part of the picture

Modeling Framework

- Different communities require different perspectives:

Business	Description of business aspects- business rules	Customer, management, marketing
Information	Information needed for tasks	Same as business
Functionality	External behavior of system	End users, db admin, general OA&M …
Implementation	Internal functioning of system	Developers,maintenance

Requirements Process

1. ICED-T analysis: Intuitive, Consistent, Efficient, Durable, Thoughtful
2. Simplified Quality Functional Deployment
3. Compute effort (Function points one way, next class)
4. Estimate staff and development time
5. Revise requirements to meet above (if exceeds cost, time)
6. Redo 1-4
7. Replan with GANTT chart (optional)
8. Review MOV to see if it is worth it

sQFD

- (see Bernstein)
- Essentially rate degree of technical difficulty and degree of customer importance - more on this and WinWin later in the course.
- Even simpler than Bernstein's method which is simpler than "House of Cards"
- Wideband Delphi - more later

ICED-T

Intuitive	Does the use of this product make sense?	Quantify, prototype
Consistent	Does the product operate in a uniform manner?	Prototype, user manual
Efficient	Is the product quick and agile to use?	Models, analysis, prototype (computing and network
Durable	Does the product respond reliably without breaking?	Reliability measures, MTTF, how much damage
Thoughtful	Does the product anticipate user's needs?	Pts to future, possibility to delight, use cases

Other Requirements Validation

- At this stage everything is informal and (usually) incomplete
- Prototyping/iteration
- Sentence by sentence review of document helps as does baselining
- Leads to test plan

Test Plan

- (more later during testing lecture)
- Includes scope, approach, resources necessary and schedule
- Lists and describes items and features to be tested
- Testing tasks to be performed
- Roles and Responsibilities - who does what

Thought Problems

- You are building a customer care application for a personal computer company with 400 customer care positions in 6 states. The customers can call from around the globe -- what techniques should be used for requirements elicitation and why?
- You are a manager responsible for 2 projects and you have been handed requirements documents for each project, one document is 10 pages long and the other is 200 pages long, what is you approach for V&V?

So Far

- Software Process Models
- Software Project Planning (woosh!)
- Requirements
- Next Time:
  - Estimation
  - Risk Analysis

References

- Brooks
- Van Vliet, H. Software Engineering: Principles and Practice, Second Edition, Wiley, 2000, ISBN: 0-471-97508-7
- Robertson, S. and Robertson, J., Mastering the requirements process, 1999, Addison-Wesley.
- Endres, A. and Rombach, D. A handbook of software and systems engineering. 2003, Addison-Wesley.