Imagine we have interviewed some stakeholders...

**Catering Manager**
- The food loaded is dictated by the number of passengers travelling in a particular class.
- A predicted number of passengers on a flight must be available 24 hours prior to departure.
- Passengers requiring special meals must indicate their request 24 hours prior to departure.

**Airline Sales manager**
- A ticket may only be issued when a fare is paid.
- For some fares, a reservation can be held and not confirmed.
- All tickets must carry appropriate endorsements relating to the terms and conditions of issue of tickets.

**Chief Security Officer**
- The number of bags in the aircraft’s hold should tally against the list of passengers on board.
- Passenger lists should not be made available to the public.
- Passengers should check-in only once.

**Travel Agent**
- An agent is responsible for holding and canceling reservations.
- Tickets offered by an agency have different fares negotiated with the airline sales department.

How do we get from here to an agreed specification?
RE involves a lot of modelling

- A model is more than just a description
  - it has its own phenomena, and its own relationships among those phenomena.
  - The model is only useful if the model’s phenomena correspond in a systematic way to the phenomena of the domain being modelled.

- Example:

  ![Diagram](image)

For every B, at least one P exists such that R(P, B)

Remember: “It’s only a model”

- There will always be:
  - phenomena in the model that are not present in the application domain
  - phenomena in the application domain that are not in the model

- A model is never perfect
  - “If the map and the terrain disagree, believe the terrain”
  - Perfecting the model is not always a good use of your time…
Modeling...

- **Modeling can guide elicitation:**
  - Does the modeling process help you figure out what questions to ask?
  - Does the modeling process help to surface hidden requirements?
    - i.e. does it help you ask the right questions?

- **Modeling can provide a measure of progress:**
  - Does completeness of the model imply completeness of the elicitation?
    - i.e. if we've filled in all the pieces of the model, are we done?

- **Modeling can help to uncover problems**
  - Does inconsistency in the model reveal interesting things?
    - e.g. inconsistency could correspond to conflicting or infeasible requirements
    - e.g. inconsistency could mean confusion over terminology, scope, etc
    - e.g. inconsistency could reveal disagreements between stakeholders

- **Modeling can help us check our understanding**
  - Can we test that the model has the properties we expect?
  - Can we reason over the model to understand its consequences?
  - Can we animate the model to help us visualize/validate the requirements?

---

Type of Model

Can choose a variety of conceptual schema:

- **natural language**
  - extremely expressive and flexible
  - very poor at capturing the semantics of the model
  - better used for elicitation, and to annotate models for communication

- **semi-formal notation**
  - captures structure and some semantics
  - can perform (some) reasoning, consistency checking, animation, etc.
    - E.g.s: diagrams, tables, structured English, etc.

- **formal notation**
  - very precise semantics, extensive reasoning possible
  - long way removed from the application domain
    - note: requirements formalisms are geared towards cognitive considerations, hence differ from most computer science formalisms

---

Source: Adapted from Loucopoulos & Karakostas, 1995, p72-73
Desiderata for Conceptual Schema

- Implementation Independence
  - does not model data representation, internal organization, etc.

- Abstraction
  - extracts essential aspects
    - e.g. things not subject to frequent change

- Formality
  - unambiguous syntax
  - rich semantic theory

- Constructability
  - can construct pieces of the model to handle complexity and size
  - construction should facilitate communication

- Ease of analysis
  - ability to analyze for ambiguity, incompleteness, inconsistency

- Traceability
  - ability to cross-reference elements
  - ability to link to design, implementation, etc.

- Executability
  - can animate the model, to compare it to reality

- Minimality
  - No redundancy of concepts in the modeling scheme
    - i.e. no extraneous choices of how to represent something

Meta-Modeling

- Can compare modeling schema using meta-models:
  - What phenomena does each scheme capture?
  - What guidance is there for how to elaborate the models?
  - What analysis can be performed on the models?

- Example meta-model:
  - Propositions about the application domain
  - Activities: modify, trigger
  - Facts: acknowledge
  - Events: State changes in the application domain
  - Actions inducing change of facts in the application domain
Modeling Principle 1: Partitioning

- Partitioning
  - captures aggregation/part-of relationship

- Example:
  - goal is to develop a spacecraft
  - partition the problem into parts:
    - guidance and navigation;
    - data handling;
    - command and control;
    - environmental control;
    - instrumentation;
    - etc
  - Note: this is not a design, it is a problem decomposition
    - actual design might have any number of components, with no relation to these sub-problems
  - However, the choice of problem decomposition will probably be reflected in the design

Modeling Principle 2: Abstraction

- Abstraction
  - A way of finding similarities between concepts by ignoring some details
  - Focuses on the general/specific relationship between phenomena
    - Classification groups entities with a similar role as members of a single class
    - Generalization expresses similarities between different classes in an "is_a" association

- Example:
  - requirement is to handle faults on the spacecraft
  - might group different faults into fault classes

  based on location of fault:
  - instrumentation fault,
  - communication fault,
  - processor fault,
  - etc

  OR

  based on symptoms of fault:
  - no response from device;
  - incorrect response;
  - self-test failure;
  - etc...
Modeling Principle 3: Projection

- Projection:
  - Separates aspects of the model into multiple viewpoints
    - Similar to projections used by architects for buildings

- Example:
  - Need to model the communication between spacecraft and ground system
  - Model separately:
    - Sequencing of messages;
    - Format of data packets;
    - Error correction behavior;
    - Etc.

- Note:
  - Projection and Partitioning are similar:
    - Partitioning defines a 'part of' relationship
    - Projection defines a 'view of' relationship
  - Partitioning assumes the parts are relatively independent

Survey of Modeling Techniques

- Modeling Enterprises
  - Goals & objectives
  - Organizational structure
  - Activities, processes, and products
  - Agents and work roles

- Modeling Functional Requirements
  - Information Structure
  - Behavioral views
  - Timing/Sequencing requirements

- Modeling Non-functional Requirements
  - Product requirements
  - Process requirements
  - External requirements

Information modeling:
- ERD
- Organization modeling:
  - i*, SSM, ISAC
- Goal modeling:
  - KAOS, CREWS

Structured Analysis:
- SADT, SSADM, JSD
Object Oriented Analysis:
- OOA, OOSE, OMT, UML
Formal Methods:
- SCR, RSML, Z, Larch, VDM

Quality tradeoffs:
- QFD, win-win
Specific NFRs:
- Timed Petri nets (performance)
- Task models (usability)
- Probabilistic MTTF (reliability)
Approaches to Enterprise Modeling

- **1970's**
  - Soft Systems Approaches:
    - Involve the entire organisation
    - Be sensitive to political and social context for organisational change
  - Examples: SSM, ISAC

- **1980's**
  - Knowledge-based Approaches:
    - Use knowledge representation schemes to build executable domain models
    - Capture static and dynamic aspects of the domain
  - Examples: RML, Requirements Apprentice, Nature

- **1990's**
  - Teleological Approaches:
    - Requirements are really just goals, so model goal hierarchies
    - Focus on the ‘why’ question, rather than ‘what’/ ‘how’
    - …and use scenarios as concrete examples of how goals are (can be) satisfied
  - Examples: KAOS, i*, CREWS,…

- **2000's ...?**

---

Entity Relationship Diagrams

- **ER diagrams**
  - Widely used for information modeling
  - Simple, easy to use
    - Note: this is a notation, not a method!

- **Used in many contexts:**
  - Domain concepts
    - Objects referred to in goal models, scenarios, etc.
  - Data to be represented in the system
    - For information systems
  - Relational Database design
  - Meta-modeling

**Key**

- entity
- attribute
- relationship
- cardinality of relationship
- identifier
- composite identifier

**Diagram:**

- **Actor**
  - `name`
  - `age`
  - `nationality`

- **Film**
  - `producer`
  - `director`
  - `title`
  - `year`

- **Cast**
  - `producer` (1,n)
  - `director` (1,n)

- **Cardinality**
  - `(a,b)`
  - `(c,d)`
ISAC

Information Systems Work & Analysis of Changes (ISAC)
- Developed in the 1970's in Sweden
- Emphasizes cooperation between users, developers and sponsors
  - Developers' role is to facilitate the process
- Good for information systems; not applicable to control systems.

ISAC Process
1. Change Analysis
   - What does the organization want?
   - How flexible is the organization with respect to changes?
2. Activity Study
   - Which activities should we regroup into information systems?
   - Which priorities do the information systems have?
3. Information Analysis
   - Which inputs and outputs do each information system have?
   - What are the quantitative requirements on each information system?
4. Implementation
   - Which technology (info carriers; h/w; s/w) do we use for the information systems?
   - Which activities of each information system are manual, which automatic?

ISAC Change Analysis
1. List problems
   - dissatisfactions with current system
     - list all problems...
     - then remove any that are trivial or intractable
2. List interest groups
   - these are “problem owners”
   - draw matrix of problems against owners
     - This exercise is done with the problem owner’s involvement
3. Analyze problems
   - Use cause-effect analysis
     - Eliminate solution-oriented problems, to get to underlying causes
   - performed by domain specialists
   - quantify the problems
4. Make Current Activity Model
   - Notation: A-schemas (similar to dataflow diagrams)
5. Analyze Goals
   - Declarative statement of goals
     - i.e. desired result, not how to get there
     - Result should be a tree of goals
6. Define Change Needs
   - Goals should explain why the problems exist; problems frustrate goals
   - Cluster problems into related groups
   - Each group is a change need
7. Generate Change Alternatives
8. Model desired situations
   - make packages of change alternatives
9. Evaluate Alternatives
10. Choose an alternative
Soft System Methodology (SSM)

- **Background**
  - Developed by Checkland in late 1970's
  - Reality is socially constructed, and therefore requirements are not objective
  - Rationale:
    - Problem situations are fuzzy (not structured) and solutions not readily apparent.
    - Impact of a computerization may be negative (e.g. intro of new system reduced productivity as it removed employee motivation)
    - Full exploitation of computerization may need radical restructuring of work processes.

- **Approach**
  - Analyze problem situation using different viewpoints
    - Determining the requirements is a discussion, bargaining and construction process.
  - Out of this process emerges not just a specification, but also:
    - plans for a modified organization structure
    - task structures
    - objectives
    - understanding of the environment

SSM Approach

1. **Existing situation**  
   (unstructured problem)

2. **Analyze the problem situation**
   - Draw a rich picture
   - look for problem themes (describe them in natural language)

3. **Define relevant systems and root definitions (CATWOE)**
   - a root definition is a concise description of a human activity system

4. **Build a conceptual model**
   - of the activity system needed to achieve the transformation
   - process oriented model, with activities & flow of resources

5. **Compare conceptual model with step (2)**
   - Ordered questioning - questions based on the model
   - Event reconstruction - take past events and compare them to the model
   - General comparison - look for features of the model that are different from current situation
   - Model overlay - point by point comparison of the two models

6. **Debate feasible and desirable changes**
   - Three types of change: structural, procedural, attitudinal

7. **Implement changes**
SSM modeling

Root definition:
"A hospital-owned system, which provides records of spending on drugs so that control action by administrators and doctors to meet defined budgets can be taken jointly"

Customers: Administrators, Doctors
Actors: not stated
Transformation: Need to know spending on drugs → Need met by recording info.
Weltanschauung: Monitoring spending on drugs is possible and is an adequate basis for joint control action
Owner: Hospital
Environment: Hospital mechanisms, roles of administrators and doctors, defined budgets

Approach:
- SD model shows dependencies between actors:
  - goal/softgoal dependency - an actor depends on another actor to attain a goal
  - resource dependency - an actor needs a resource from another actor
  - task dependency - an actor needs another actor to carry out a task
- SR model shows interactions between goals within each actor
  - Shows task decompositions
  - Shows means-ends links between tasks and goals
E.g. Strategic Dependency Model

LEGEND

Dependent Dependee
- Resource Dependency
- Task Dependency
- Goal Dependency
- Softgoal Dependency
  ○ Open (noncommittal)  × Critical

This diagram ©2001, Eric Yu

E.g. Strategic Rationale Model

"Functional" Alternatives

This diagram ©2001, Eric Yu
KAOS

Background
- Developed in the early 90's
  - first major teleological requirements modeling language
  - full tool support available
  - has been applied to a number of industrial case studies
- Two parts:
  - Informal goal structuring model
  - Formal definitions for each entity in temporal logic

Approach
- Method focuses on goal elaboration:
  - define initial set of high level goals & objects they refer to
  - define initial set of agents and actions they are capable of
- Then iteratively:
  - refine goals using AND/OR decomposition
  - identify obstacles to goals, and goal conflicts
  - operationalize goals into constraints that can be assigned to individual agents
  - refine & formalize definitions of objects & actions

KAOS meta-model