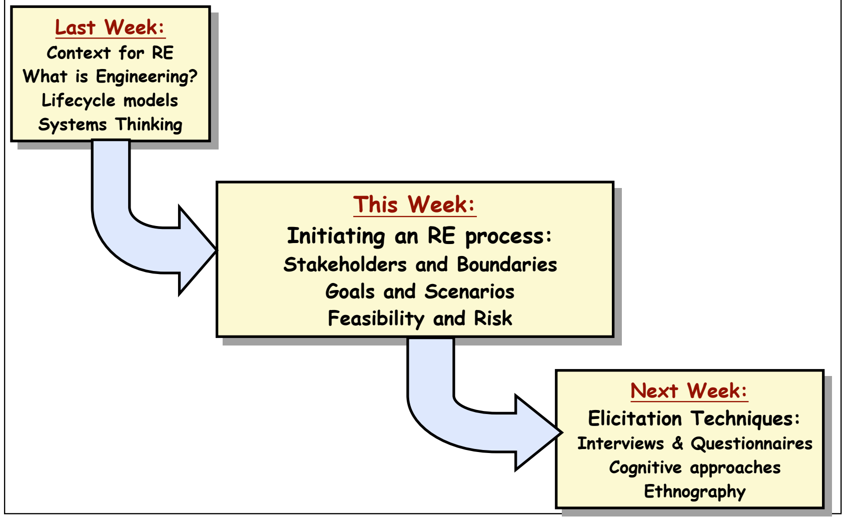
# information acquisition?

THE PROJECT REQUIREMENTS ARE FORMING IN MY MIND.
NOW THEY'RE CHANGING... CHANGING... CHANGING... OKAY. NO, WAIT... CHANGING... DONE.
NATURALLY, I WON'T BE SHARING ANY OF THESE THOUGHTS WITH ENGINEERING.
I BUDGETED FOR SOME GOONS TO BEAT IT OUT OF YOU.
S. Adams E-Mail: SCOTTADAMS@AOL.COM
© 1994 United Feature Syndicate, Inc.
9-22

# Lecture 3: starting points



**Last Week:**
Context for RE
What is Engineering?
Lifecycle models
Systems Thinking

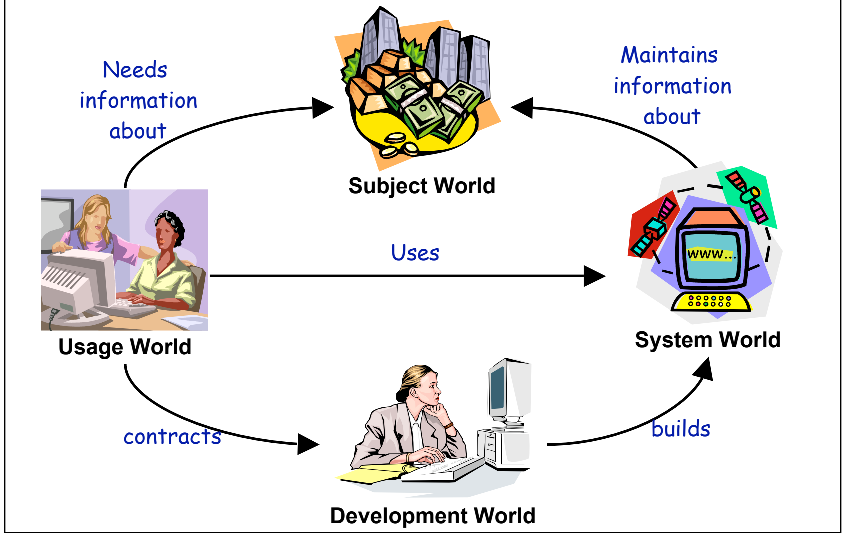
**This Week:**
Initiating an RE process:
Stakeholders and Boundaries
Goals and Scenarios
Feasibility and Risk

**Next Week:**
Elicitation Techniques:
Interviews & Questionnaires
Cognitive approaches
Ethnography

# Starting Points

- Stakeholders
  - Importance of Customer Links
  - Who are the stakeholders?
- Boundaries
  - How do you scope the problem?
- Goals and Scenarios
  - A useful way to organise initial collection of information
- Feasibility
  - How to conduct a feasibility study
  - How to choose which project to persue
- Risk
  - Continuous Risk Management
  - Identifying risks through hazard and fault analysis

# The four worlds



*Source: Adapted from Loucopoulos & Karakostas, 1995, p73*

# Finding out about the four worlds

→ Subject World

- the subject matter of the information system:
  - e.g., customers, accounts, transactions for a bank information system

→ Usage World

- the environment within which the planned system will operate
  - e.g., people, such as managers, clerks, customers; also business processes such as handling a withdrawal, a deposit of foreign currency,...

→ System World

- what the system does within its operational environment;
- what information it contains and what functions it performs;
  - e.g., system records all transactions in a database, reports on transactions for a particular account, gives account balance,...

→ Development World

- the development process, team, schedule, required qualities (security, performance,…) etc.
  - e.g., system to be delivered in 12 months, fully tested to MCDC standard, etc.

# Stakeholders

→ Stakeholder analysis:

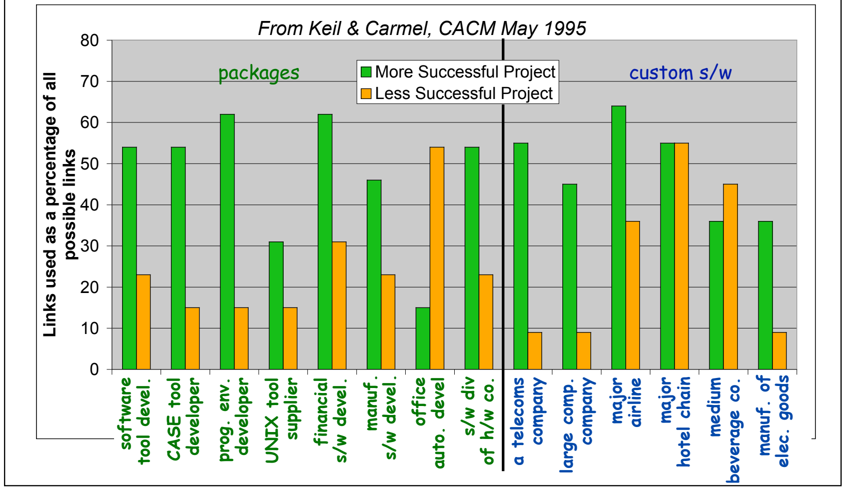
- Identify all the people who must be consulted during information acquisition
- Look for stakeholders associated with each of the four worlds

→ Example stakeholders

- Users
  - concerned with the features and functionality of the new system
- Designers
  - want to build a perfect system, or reuse existing code
- Systems analysts
  - want to "get the requirements right"
- Training and user support staff
  - want to make sure the new system is usable and manageable
- Business analysts
  - want to make sure "we are doing better than the competition"
- Technical authors
  - will prepare user manuals and other documentation for the new system
- The project manager
  - wants to complete the project on time, within budget, with all objectives met.
- "the customer"
  - whoever it is that pays for the new system!

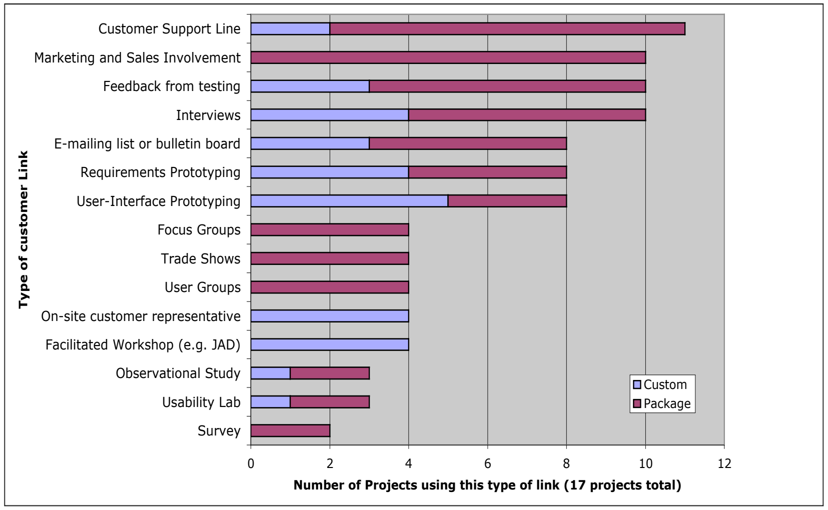
# Links with customers

→ Successful projects tend to have more links with customer(s)



***Source:*** *Adapted from Keil and Carmel, 1995, p37*

# Customer-Developer Links



## Difficulties of Elicitation

- Thin spread of domain knowledge
  - The knowledge might be distributed across many sources
    - It is rarely available in an explicit form (I.e. not written down)
  - There will be conflicts between knowledge from different sources
    - People have conflicting goals
    - People have different understandings of the problem
- Tacit knowledge (The “say-do” problem)
  - People find it hard to describe knowledge they regularly use
    - Descriptions may be inaccurate rationalizations of expert behaviour
- Limited Observability
  - The problem owners might be too busy solving it using the existing system
  - Presence of an observer may change the problem
    - E.g. the Probe Effect and the Hawthorne Effect
- Bias
  - People may not be free to tell you what you need to know
    - Political climate & organisational factors matter
  - People may not want to tell you what you need to know
    - The outcome will affect them, so they may try to influence you (hidden agendas)

## Example

- The problem area:
  - Loan approval department in a large bank
  - The analyst is trying to elicit the rules and procedures for approving a loan
- Why this might be difficult:
  - Implicit knowledge:
    - There is no document in which the rules for approving loans are written down
  - Conflicting information:
    - Different members of the department have different ideas about what the rules are
  - Say-do problem:
    - The loan approval process described to you by the loan approval officers is quite different from your observations of what they actually do
  - Probe effect:
    - The loan approval process used by the officers while you are observing is different from the one they normally use
  - Bias:
    - The loan approval officers fear that your job is to computerize their jobs out of existence, so they are deliberately emphasizing the need for case-by-case discretion (to convince you it has to be done by a human!)

## Psychological Considerations

- Experts are not used to describing what they do.
  - Three stage model of learning:
    1) cognitive - verbal rehearsal of tasks;
    2) associative - reinforcement through repetition, verbal mediation disappears
    3) autonomous - compiled, no conscious awareness of performance.
  - Procedural and declarative are different mechanisms
    - Declarative knowledge becomes procedural with repeated application - experts lose awareness of what they know and cannot introspect reliably
    - Experts have little or no introspective access to higher order cognitive processes
- Representational Problems
  - Experts don’t have the language to describe their knowledge
    - No spoken language offers the necessary precision
    - Knowledge Engineer and Expert must work together to create a suitable language
  - Different knowledge representations are good for different things
    - Epistemological adequacy: does the formalism express expert's knowledge well?
- Brittleness
  - Knowledge is created, not extracted.
    - Knowledge models are abstractions of reality and hence are unavoidably selective
    - Brittleness caused by the simplifying assumptions - instead of adding more knowledge, a better (more comprehensive) model is needed.

## Expert Bias

- What is bias?
  - Bias only exists in relation to some reference point
    - can there ever be “no bias”? (reflects reality or truth)
  - We cannot perceive reality directly:
    - It is interpreted through a filter of mental models
    - …mediated by our senses and neural pathways.
  - All decision making is based partly on personal value systems.
- Types of bias:
  - Motivational bias
    - expert makes accommodations to please the interviewer or some other audience
  - Cognitive bias
    - expert filters information to fit with her own mental model

### Sources of Bias

- Social pressure
  response to verbal and non-verbal cues from an interviewer
- Group think
  response to reactions of other experts
- Impression management
  response to imagined reactions of managers, clients, etc.
- Wishful thinking
  response to hopes or possible gains.
- Misinterpretation
  Analyst selectively interprets to support what she currently believes.
- Misrepresentation
  expert cannot accurately fit a response into the requested response mode
- anchoring
  contradictory data is ignored once an initial solution is available
- inconsistency
  assumptions made earlier are forgotten
- availability
  some data are easier to recall than others
- underestimation of uncertainty
  tendency to underestimate by a factor of 2 or 3.

# Scoping decision I

- Decide the scope of the **problem**:
  - E.g. Bookstore example:
    "Textbooks are often not ordered in time for the start of classes"
  - But that's just a symptom. (So you ask the manager "why?")
    "Because we don't receive the booklists from instructors early enough"
  - Is that just a symptom of some other problem? (…so ask the instructors "why?")
    "Because the instructors aren't allocated to courses early enough"
  - Is that just a symptom of some other problem? (…so ask the UG office "why?")
    "Because we never know who's available to teach until the last minute"
  - Is that just a symptom of some other problem? (…so ask the dept chair "why?")
    "Because there's always uncertainty about who gets hired, sabbaticals, etc."
  - Is that just a symptom of some other problem? (…so ask the dept chair "why?")
    "Because instructors we want to hire don't accept our offers early enough"
  - Is that just a symptom of some other problem? (…so ask the new recruits "why?")
    "Because some other universities seem to wait for ages before making offers"
  - Is that just a symptom of some other problem? (…so ask U of Waterloo, etc, "why?")
    "Because it takes our department a long time to reach consensus on hiring"
  - Is that just a… …oh wait… …maybe we can develop a decision support system for faculty hiring at U of Waterloo, and that will help us get our textbooks for the start of class…

# How to scope the problem

- Difficulty:
  - Every problem can be seen as as symptom of some other (larger) problem
  - You can keep on tracing root causes forever if you're not careful
- Approach: (…ask yourself these questions…)
  - Is there a reasonable expectation that this problem can be solved?
    (…independently of the larger problem?)
  - Is there a reasonable expectation that solving this problem will help?
    (…without also solving the larger problem?)
  - Is this a problem that the stakeholders want solved?
    (do the "local experts" think this problem is the one that matters?)
  - Is this a problem that someone will pay you to solve?
    (Hint: a feasibility study should quantify the return on investment)

# Scoping Decision II

- Decide the scope of the **solution**
  - Say you decided that *delay in processing booklists from instructors* is the right level of problem to tackle.
    - "So, let's computerize the submission of textbook forms from instructors"
  - But while we're at it:
    - "it would help if we also computerized the submission of orders to the publishers"
  - …and of course:
    - "we ought to computerize the management of book inventories too, so we can quickly check stock levels before ordering new books"
  - …and in that case:
    - "we might as well computerize the archives of past years booklists so that we can predict demand better"
  - …and therefore:
    - "it would also make sense to provide a computerized used book exchange, because that has a big effect on demand for new books"
  - …and then of course there's … oh, wait, this is going to cost millions!
    - Bookstore manager: "tell me again how this automated used book exchange will help me order books faster?"

# How to scope the solution

- Difficulty:
  - We could keep on throwing more technology at the problem forever
  - It's hard to decide when to stop adding extra "bells and whistles"
- Approach (…select among alternatives carefully…)
  - Is there a reasonable expectation that this alternative can be implemented?
    (…independently of all the other options?)
  - Is there a reasonable expectation that implementing this alternative will (help to) solve the original problem?
    (…without also having to address other aspects of the problem?)
  - Is this a solution that the stakeholders can live with?
    (do the "local experts" think they would use all these functions?)
  - Is this a solution that someone will pay you to build?
    (Hint: a feasibility study should quantify the return on investment for each alternative)

# Goal-based Approaches

## ➜ Approach

- Focus on why systems are constructed
- Express the ‘why’ as a set of stakeholder goals
- Use goal refinement to arrive at specific requirements
- Goal analysis
  - document, organize and classify goals
- Goal evolution
  - refine, elaborate, and operationalize goals
- Goal hierarchies show **refinements** and **alternatives**

## ➜ Advantages

- Reasonably intuitive
- Explicit declaration of goals provides sound basis for conflict resolution

## ➜ Disadvantages

- Captures a static picture - what if goals change over time?
- Can regress forever up (or down) the goal hierarchy

**Source:** *Adapted from Anton, 1996.*

# Goal Modeling

## ➜ (Hard) Goals:

- Describe functions that must be carried out. E.g.
  - Satisfaction goals
  - Information goals

## ➜ Softgoals:

- Cannot really be fully satisfied. E.g.
  - Accuracy
  - Performance
  - Security
  - …

## ➜ Also classified temporally:

- Achieve/Cease goals
  - Reach some desired state eventually
- Maintain/Avoid goals
  - Keep some property invariant
- Optimize
  - A criterion for selecting behaviours

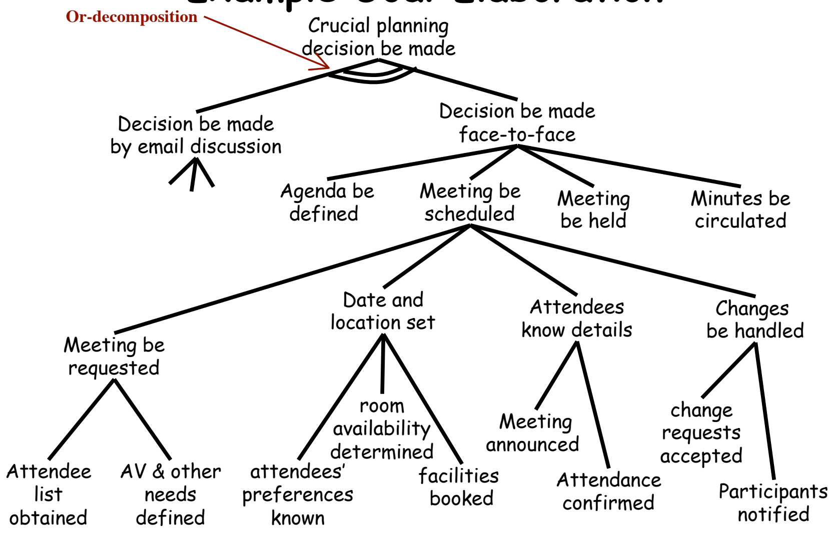
## ➜ Agents:

- Owners of goals
- Choice of when to ascribe goals to agents:
  - Identify agents first, and then their goals
  - Identify goals first, and then allocate them to agents during operationalization

## ➜ Modelling Tips:

- Multiple sources yield better goals
- Associate stakeholders with each goal
  - reveals viewpoints and conflict
- Use scenarios to explore how goals can be met
- Explicit consideration of obstacles helps to elicit exceptions

# Example Goal Elaboration



Or-decomposition

Crucial planning decision be made

Decision be made by email discussion

Decision be made face-to-face

Agenda be defined

Meeting be scheduled

Meeting be held

Minutes be circulated

Meeting be requested

Date and location set

Attendees know details

Changes be handled

Attendee list obtained

AV & other needs defined

attendees’ preferences known

room availability determined

facilities booked

Meeting announced

Attendance confirmed

change requests accepted

Participants notified

# Goal Analysis

## ➜ Goal Elaboration:

- “Why” questions explore higher goals (context)
- “How” questions explore lower goals (operations)
- “How else” questions explore alternatives

## ➜ Relationships between goals:

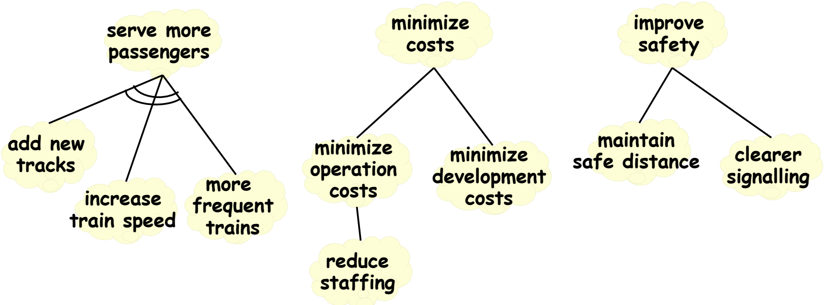
- One goal **helps** achieve another (+)
- One goal **hurts** achievement of another (-)
- One goal **makes** another (++)
  - Achievement of one goal guarantees achievement of another
- One goal **breaks** another (--)
  - Achievement of one goal prevents achievement of another
- Precedence ordering – must achieve goals in a particular order

## ➜ Obstacle Analysis:

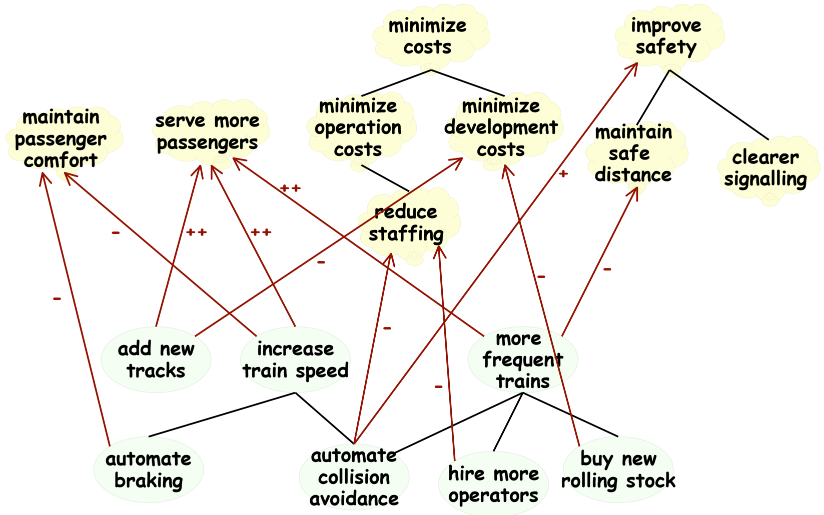
- Can this goal be obstructed, if so how?
- What are the consequences of obstructing it?

# Softgoals

- → Some goals can never be fully satisfied
  - Treat these as softgoals
    - E.g. "system be easy to use"; "access be secure"
    - Also known as 'non-functional requirements'; 'quality requirements'
  - Will look for things that contribute to satisficing the softgoals
  - E.g. for a train system:



# Softgoals as selection criteria



# Scenarios

- → Scenarios
  - Specific sequence of interaction between actor and system
  - Tend to be short (e.g between 3 and 7 steps)
  - May be:
    - positive (i.e. required behavior)
    - negative (i.e an undesirable interaction)
  - May be indicative (describe current system) or optative (how it should be)
- → Advantages
  - Very natural: stakeholders tend to use them spontaneously
    - E.g "suppose I'm admitted to hospital - what happens during my admission?"
    - Typical answer: "You, or the person accompanying you would talk to the person at the admissions desk. You have to show your OHIP card and explain who referred you to the hospital. Then you…" [and so on]
  - Short scenarios very good for quickly illustrating specific interactions
- → Disadvantages
  - Lack of structure: need use cases or task models to provide higher level view

*Source:* *Adapted from Dardenne, 1993.*

# Example Scenario

Title: Successful meeting scheduled using messaging option

Participants: Alice (initiator, not attending); Bob, Carlo, Daphne (attendees)

| Action | Goals satisfied | Obstacles / Problems |
|---|---|---|
| Alice requests meeting, specifying participants, timeframe | Meeting requested;<br>Attendee list obtained | What if selected timeframe is infeasible? |
| AS sends participant requests to Bob, Carlo and Daphne | ? | Did we miss a goal? |
| Bob reads message | Participants informed | Can't detect when messages are read; what happens if Bob reads the message but doesn't reply? |
| Carlo reads message | | |
| Daphne reads message | | |
| Bob replies with preferences | Attendees preferences known | What if the preferences are mutually exclusive?<br>Should we allow some to be higher priority? |
| Carlo replies with preferences | | |
| Daphne replies with preferences | | |
| AS schedules meeting | Room availability determined; room booked | |
| AS notifies Alice, Bob, Carlo, Daphne of time and location | Meeting announced;<br>Attendance Confirmed (?) | How do we know if they've all read the announcement? What if the schedule is no longer convenient for one of them? |

# Use Cases

## → What is a use case?

- Each different way that an actor interacts with a system is a use case
  - "a description of a *sequence of actions* that a system performs that yields an *observable result of value to a particular actor* " [Booch]
  - All the use cases need to be enumerated (or the requirements will not be complete)
- A description of a set of possible scenarios, with a common purpose
- Typically written in natural language
- No internal description of the system; just the interaction.

## → Combining use cases

- extends/uses

## → Advantages & Disadvantages

- detailed characterization of all possible interaction with the system
- helps in drawing system boundary, and scoping the requirements
- Use cases do not capture domain knowledge!!
- Don't confuse use cases with a precise specification!

*Source:* *Adapted from Rumbaugh 1997, p123-124*

# Use Cases - Example

### Use Case Description

**Name:** Place Order

**Precondition:** A valid user has logged into the system.

**Description:**

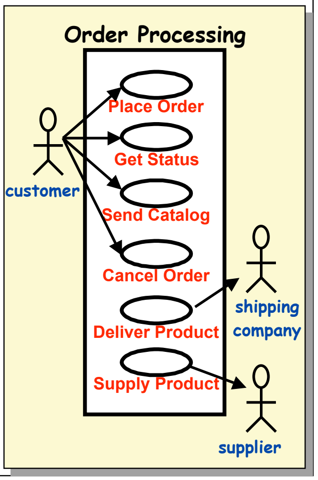
1. The use case starts when the customer selects Place Order.
2. The customer enters his or her name and address.
3. If the customer enters only the zip code, the system will supply the city & state.
4. The customer will enter product codes for the desired products.
5. The system will supply a product description and price for each item.
6. The system will keep a running total of items ordered as they are entered.
7. The customer will enter credit card payment information.
8. The customer will select Submit.
9. The system will verify the information, save the order as pending, and forward payment information to the accounting system.
10. When payment is confirmed, the order is marked Confirmed, an order ID is returned to the customer, and the use case ends.

**Exceptions:**

In step 9, if any information is incorrect, the system will prompt the customer to correct the information.

**Postcondition:** The order has been saved in the system and marked confirmed.

### Use Case Diagram



# Feasibility studies

## → Objectives of a feasibility study:

- To find out if a software development project can be done:
  - ...is it possible?
  - ...is it justified?
- To suggest possible alternative solutions.
- To provide management with enough information to know:
  - Whether the project can be done
  - Whether the final product will benefit its intended users
  - What the alternatives are (so that a selection can be made in subsequent phases)
  - Whether there is a preferred alternative

## → A feasibility study is a management-oriented activity

- After a feasibility study, management makes a "go/no-go" decision.
- Need to examine the problem in the context of broader business strategy

# Content of feasibility study

## → Things to be studied in the feasibility study:

- The present organizational system
  - users, policies, functions, objectives,...
- Problems with the present system
  - inconsistencies, inadequacies in functionality, performance,…
- Objectives and other requirements for the new system
  - Which problems need to be solved?
  - What needs to change?
- Constraints
  - including nonfunctional requirements on the system (preliminary pass)
- Possible alternatives
  - "Sticking with the current system" should always be studied as one alternative
  - Different business processes for solving the problems
  - Different levels/types of computerization for the solutions
- Advantages and disadvantages of the alternatives

## → Things to conclude:

- Feasibility of the project
- The preferred alternative.

# Four Types of feasibility

- ➜ Technical feasibility
  - Is the project possible with current technology?
    - How much technical risk is there?
  - Does the technology exist at all?
    - Is it available locally?
    - Can it be obtained?
    - Will it be compatible with other systems?
- ➜ Economic feasibility
  - Is the project possible, given resource constraints?
  - What benefits will result from the system?
    - Both tangible and intangible benefits
    - Quantify them!
  - What are the development and operational costs?
  - Are the benefits worth the costs?
- ➜ Schedule feasibility
  - Is it possible to build a solution in time to be useful:
    - Any constraints on the schedule?
    - Can these constraints be met?
- ➜ Operational feasibility
  - Urgency of the problem and the acceptability of any solution:
    - If the system is developed, will it be used?
  - Human and social issues…
  - internal issues:
    - Available of human resources?
    - Potential labour objections?
    - Manager resistance?
    - Organizational conflicts and policies?
  - external issues:
    - Social acceptability?
    - legal aspects and government regulations?

# Benefits and Costs

- ➜ Tangible Benefits
  - Readily quantified as $ values
  - Examples:
    - increased sales
    - cost/error reductions
    - increased throughput/efficiency
    - increased margin on sales
    - more effective use of staff time
- ➜ Intangible benefits
  - Difficult to quantify
    - But maybe more important!
    - business analysts help estimate $ values
  - Examples:
    - increased flexibility of operation
    - higher quality products/services
    - better customer relations
    - improved staff morale
- ➜ How will the benefits accrue?
  - When - over what timescale?
  - Where in the organization?

- ➜ Development costs (OTO)
  - Development and purchasing costs:
    - Cost of development team
    - Consultant fees
    - software used (buy or build)?
    - hardware (what to buy, buy/lease)?
    - facilities (site, communications, power,...)
  - Installation and conversion costs:
    - installing the system,
    - training personnel,
    - file conversion,....
- ➜ Operational costs (on-going)
  - System Maintenance:
    - hardware (repairs, lease, supplies,...),
    - software (licenses and contracts),
    - facilities
  - Personnel:
    - For operation (data entry, backups,…)
    - For support (user support, hardware and software maintenance, supplies,…)
    - On-going training costs

# Analyzing Costs vs. Benefits

- ➜ Identify costs and benefits
  - Tangible and intangible, one-time and recurring
  - Assign values to costs and benefits
- ➜ Determine Cash Flow
  - Project costs and benefits over time, e.g. 3-5 years
  - Calculate Net Present Value for all future costs/benefits
    - determines future costs/benefits of the project in terms of today's dollar values
    - A dollar earned today is worth more than a potential dollar earned next year
- ➜ Do cost/benefit analysis
  - Calculate Return on Investment:
    - Allows comparison of lifetime profitability of alternative solutions.

$$ROI = \frac{\text{Lifetime benefits - Lifetime costs}}{\text{Lifetime costs}}$$

  - Calculate Break-Even point:
    - how long will it take (in years) to pay back the accrued costs:

Accrued Cost (initial + incremental) < Accrued Benefit

# Risk and RE

- ➜ Risk Management is a systematic activity
  - Requires both technical and management attention
  - Requires system-level view
  - Should continue throughout a project
- ➜ Techniques exist to identify and assess risks
  - E.g. fault tree analysis
  - E.g. Risk assessment matrix
- ➜ Risk and Requirements Engineering
  - Risk analysis can uncover new requirements
    - Especially for safety-critical or security-critical applications
  - Risk analysis can uncover feasibility concerns
  - Risk analysis will assist in appropriate management action

# Risk - Key concepts

- About Risk
  - Risk is "the possibility of suffering loss"
  - Risk itself is not bad, it is essential to progress
  - The challenge is to manage the amount of risk
- Two Parts:
  - Risk Assessment
  - Risk Control
- Useful concepts:
  - For each risk: Risk Exposure
    - RE = p(unsat. outcome) X loss(unsat. outcome)
  - For each mitigation action: Risk Reduction Leverage
    - RRL = ($RE_{before}$ - $RE_{after}$) / cost of intervention

# Principles of Risk Management

- Global Perspective
  - View software in context of a larger system
  - For any opportunity, identify both:
    - Potential value
    - Potential impact of adverse results
- Forward Looking View
  - Anticipate possible outcomes
  - Identify uncertainty
  - Manage resources accordingly
- Open Communications
  - Free-flowing information at all project levels
  - Value the individual voice
    - Unique knowledge and insights
- Integrated Management
  - Project management is risk management!
- Continuous Process
  - Continually identify and manage risks
  - Maintain constant vigilance
- Shared Product Vision
  - Everybody understands the mission
    - Common purpose
    - Collective responsibility
    - Shared ownership
  - Focus on results
- Teamwork
  - Work cooperatively to achieve the common goal
  - Pool talent, skills and knowledge

***Source:*** *Adapted from SEI Continuous Risk Management Guidebook*

# Risk Assessment

- Quantitative:
  - Measure risk exposure using standard cost & probability measures
  - Note: probabilities are rarely independent
- Qualitative:
  - Develop a risk classification matrix:

| | | Likelihood of Occurrence | | |
|---|---|---|---|---|
| | | **Very likely** | **Possible** | **Unlikely** |
| **Undesirable outcome** | **(5) Loss of Life** | Catastrophic | Catastrophic | Severe |
| | **(4) Loss of Spacecraft** | Catastrophic | Severe | Severe |
| | **(3) Loss of Mission** | Severe | Severe | High |
| | **(2) Degraded Mission** | High | Moderate | Low |
| | **(1) Inconvenience** | Moderate | Low | Low |

# Top 10 Development Risks (+ Countermeasures)

- Personnel Shortfalls
  - use top talent
  - team building
  - training
- Unrealistic schedules/budgets
  - multisource estimation
  - designing to cost
  - requirements scrubbing
- Developing the wrong Software functions
  - better requirements analysis
  - organizational/operational analysis
- Developing the wrong User Interface
  - prototypes, scenarios, task analysis
- Gold Plating
  - requirements scrubbing
  - cost benefit analysis
  - designing to cost
- Continuing stream of reqts changes
  - high change threshold
  - information hiding
  - incremental development
- Shortfalls in externally furnished components
  - early benchmarking
  - inspections, compatibility analysis
- Shortfalls in externally performed tasks
  - pre-award audits
  - competitive designs
- Real-time performance shortfalls
  - targeted analysis
  - simulations, benchmarks, models
- Straining computer science capabilities
  - technical analysis
  - checking scientific literature

***Source:*** *Adapted from Boehm, 1989*