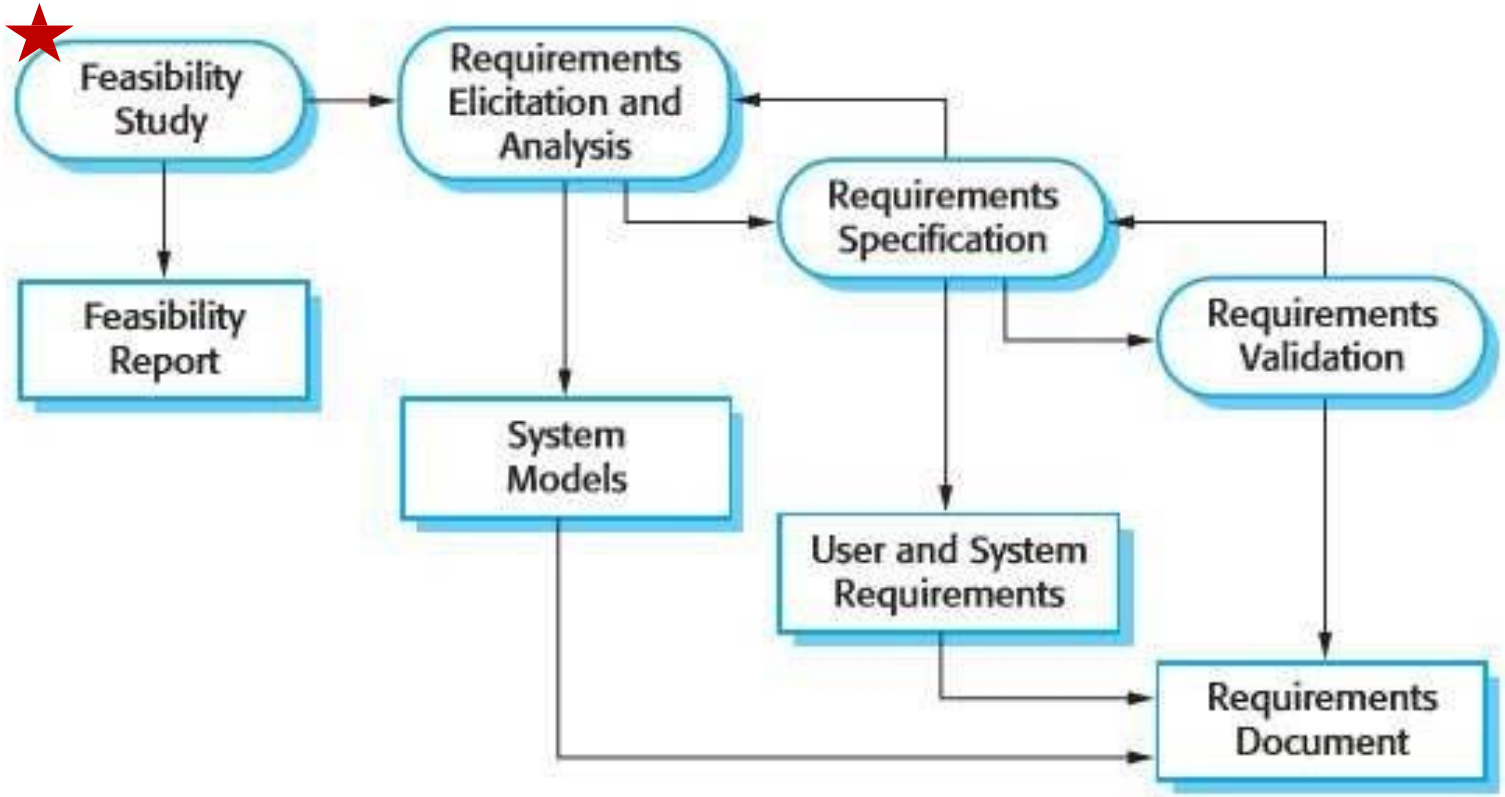
2. Feasibility Study

Requirements Engineering Process



Why a Feasibility Study?

- **Objectives:**
  - **To find out if** a system 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
    - Whether there is a preferred alternative

- **A management-oriented activity:**
  - After a feasibility study, management makes a "***go/stop***" 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 (**existing**) organizational system
    - Stakeholders, users, policies, functions, objectives
  - **Problems** with the present system
    - inconsistencies, inadequacies in functionality, performance
  - **Goals** and **other requirements** for the new system
    - Which problems need to be solved?
    - What would the stakeholders like to achieve?
  - **Constraints**
    - Including nonfunctional requirements on the system
  - Possible **alternatives**
    - "Sticking with the current system" is always an 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 (Go / Stop)
  - The preferred alternative

4 Types of Feasibility Study

Technical Feasibility	• ***"Is the project possible with current technology?"***
	• What technical risk is there?
	• Availability of the technology
	  • Is it available locally?
	  • Can it be obtained?
	  • Will it be compatible with other systems?

Economical Feasibility	• ***"Is the project possible, given resource constraints?"***
	• What are the benefits?
	  • Both tangible and intangible
	  • Quantification requires
	• 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?"*** • What are the consequences of delay? • Any constraints on the schedule? • Can these constraints be met?

Operational Feasibility	• ***"If the system is developed, will it be used?"***
	• Human and social issues:
	  • Potential labor objections?
	  • Manager resistance?
	  • Organizational conflicts and policies?
	  • Social acceptability?
	  • Legal aspects and government regulations?

Comparing Alternatives

- **Feasibility Analysis Matrix**
  - Each cells contains the feasibility assessment notes for each candidate.
    - Can be assigned a rank or score for each criterion
  - A final ranking or score is recorded in the last row.

	Candidate 1 Name	Candidate 2 Name	Candidate 3 Name
Description			
Operational Feasibility			
Technical Feasibility			
Schedule Feasibility			
Economic Feasibility			
Ranking			

Feasibility Analysis Matrix

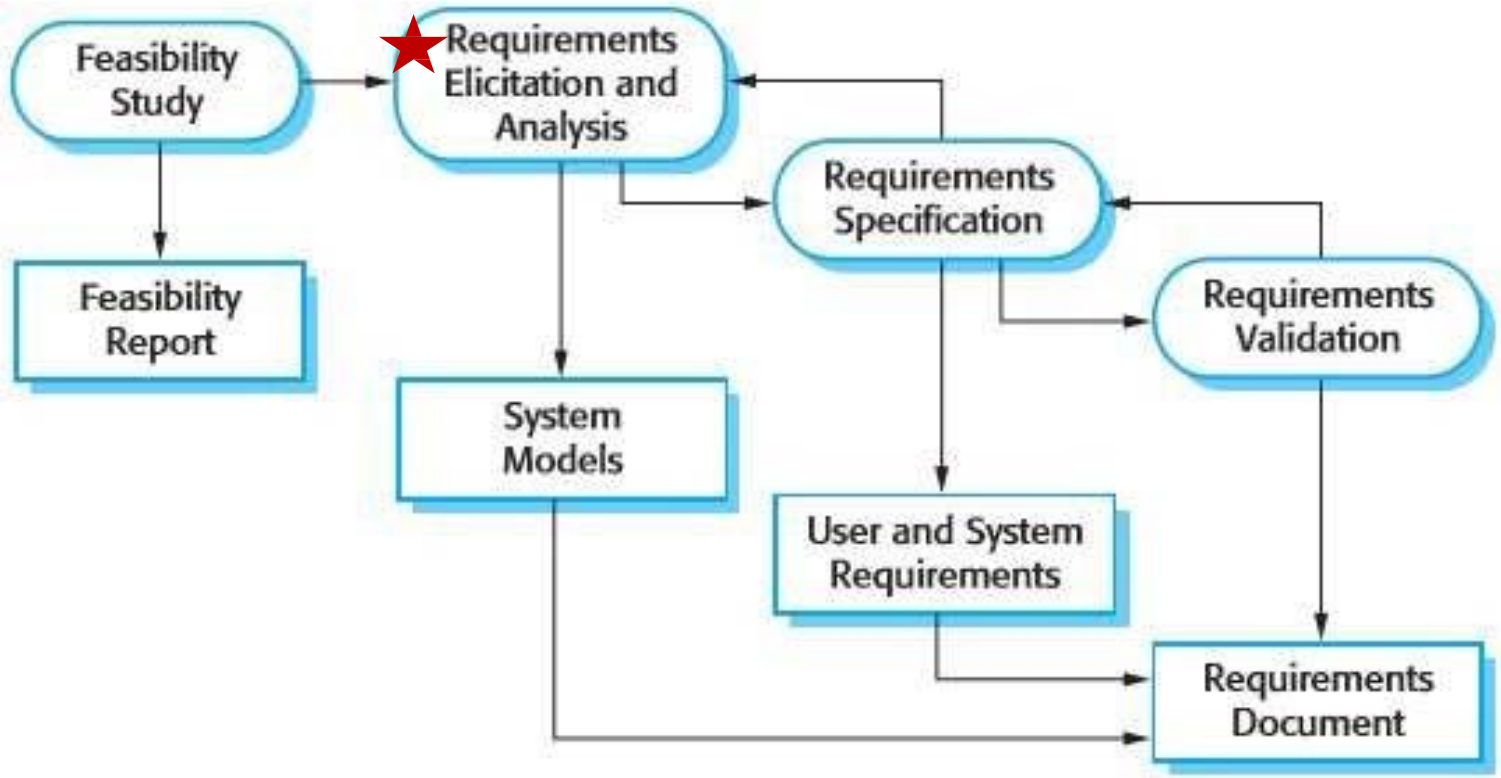
Feasibility Criteria	Wt.	Candidate 1	Candidate 2	Candidate 3	Candidate …
Operational Feasibility **Functionality**. Describes to what degree the alternative would benefit the organization and how well the system would work. **Political**. A description of how well received this solution would be from both user management, user, and organization perspective.	30%	Only supports Member Services requirements and current business processes would have to be modified to take advantage of software functionality **Score: 60**	Fully supports user required functionality. **Score: 100**	Same as candidate 2. **Score: 100**	
Technical Feasibility **Technology**. An assessment of the maturity, availability (or ability to acquire), and desirability of the computer technology needed to support this candidate. **Expertise**. An assessment to the technical expertise needed to develop, operate, and maintain the candidate system.	30%	Current production release of Platinum Plus package is version 1.0 and has only been on the market for 6 weeks. Maturity of product is a risk and company charges an additional monthly fee for technical support. Required to hire or train C++ expertise to perform modifications for integration requirements. **Score: 50**	Although current technical staff has only Powerbuilder experience, the senior analysts who saw the MS Visual Basic demonstration and presentation, has agreed the transition will be simple and finding experienced VB programmers will be easier than finding Powerbuilder programmers and at a much cheaper cost. MS Visual Basic 5.0 is a mature technology based on version number. **Score: 95**	Although current technical staff is comfortable with Powerbuilder, management is concerned with recent acquisition of Powerbuilder by Sybase Inc. MS SQL Server is a current company standard and competes with SYBASE in the Client/Server DBMS market. Because of this we have no guarantee future versions of Powerbuilder will "play well" with our current version SQL Server. **Score: 60**	

Feasibility Analysis Matrix

Feasibility Criteria	Wt.	Candidate 1	Candidate 2	Candidate 3	Candidate …
Operational Feasibility	**30%**	**Score: 60**	**Score: 100**	**Score: 100**	
Technical Feasibility	**30%**	**Score: 50**	**Score: 95**	**Score: 100**	
Economic Feasibility **Cost to develop:** **Payback period (discounted):** **Net present value:** **Detailed calculations:**	**30%**	 Approximately $350,000. Approximately 4.5 years. Approximately $210,000. See Attachment A. **Score: 60**	 Approximately $418,040. Approximately 3.5 years. Approximately $306,748. See Attachment A. **Score: 85**	 Approximately $400,000. Approximately 3.3 years. Approximately $325,500. See Attachment A. **Score: 90**	
Schedule Feasibility An assessment of how long the solution will take to design and implement.	**10%**	Less than 3 months. **Score: 95**	9-12 months **Score: 80**	9 months **Score: 85**	
Ranking	**100%**	**60.5**	**92**	**83.5**	

3. Requirements Elicitation

Requirements Engineering Process



Requirements Elicitation

- There should be a "***problem***" that needs solving.
  - Dissatisfaction with the current state of affairs
  - New business opportunity
  - Potential saving of cost, time, resource usage, etc.

- **Collect enough information to** Identify the "***problem***" and "***opportunity***"
  - Which problem needs to be solved? (identify problem Boundaries)
  - Where is the problem? (understand the Context/Problem Domain)
  - Whose problem is it? (identify Stakeholders)
  - Why does it need solving? (identify the stakeholders' Goals)
  - How might a software system help? (collect some Scenarios)
  - When does it need solving? (identify Development Constraints)
  - What might prevent us solving it? (identify Feasibility and Risk)

Problems of Requirements Elicitation

- **Vague problem stated by the customer (stakeholders)**
  - **Stakeholders** don't know what they really want.
  - Stakeholders express requirements in their own terms.
  - Different stakeholders may have conflicting requirements.
  - New stakeholders may emerge and the business environment changes.
- **Organizational** and **political factors** influence the system requirements.
- The requirements **keep changing** during the analysis process.

Stakeholders

- **Stakeholder analysis**
  - **Identify all the people** who must be consulted during information acquisition

- **Typical stakeholders**

User	Concerned with the features and functionality of the new system
Designer	Want to build a perfect system, or reuse existing code
System Analyst	Want to “get the requirements right”
Training and User Support	Want to make sure the new system is usable and manageable
Business Analyst	Want to make sure “we are doing better than the competition”
Technical Author	Will prepare user manuals and other documentation for the new system
Project Manager	Wants to complete the project on time, within budget, with all objectives met.
Customer	Wants to get best value for money invested

Finding Stakeholders through Goal Analysis

- **Goal Analysis**
  - Focus on *why* a system is required
    - Express the 'why' as a set of stakeholder goals
  - Goal refinement to arrive at specific requirements
    - Document, organize and classify goals
  - Goal evolution
    - Refine, elaborate, and operationalize goals
  - Goal hierarchies show refinements and alternatives
  - **Goal model** visualizes goal analysis

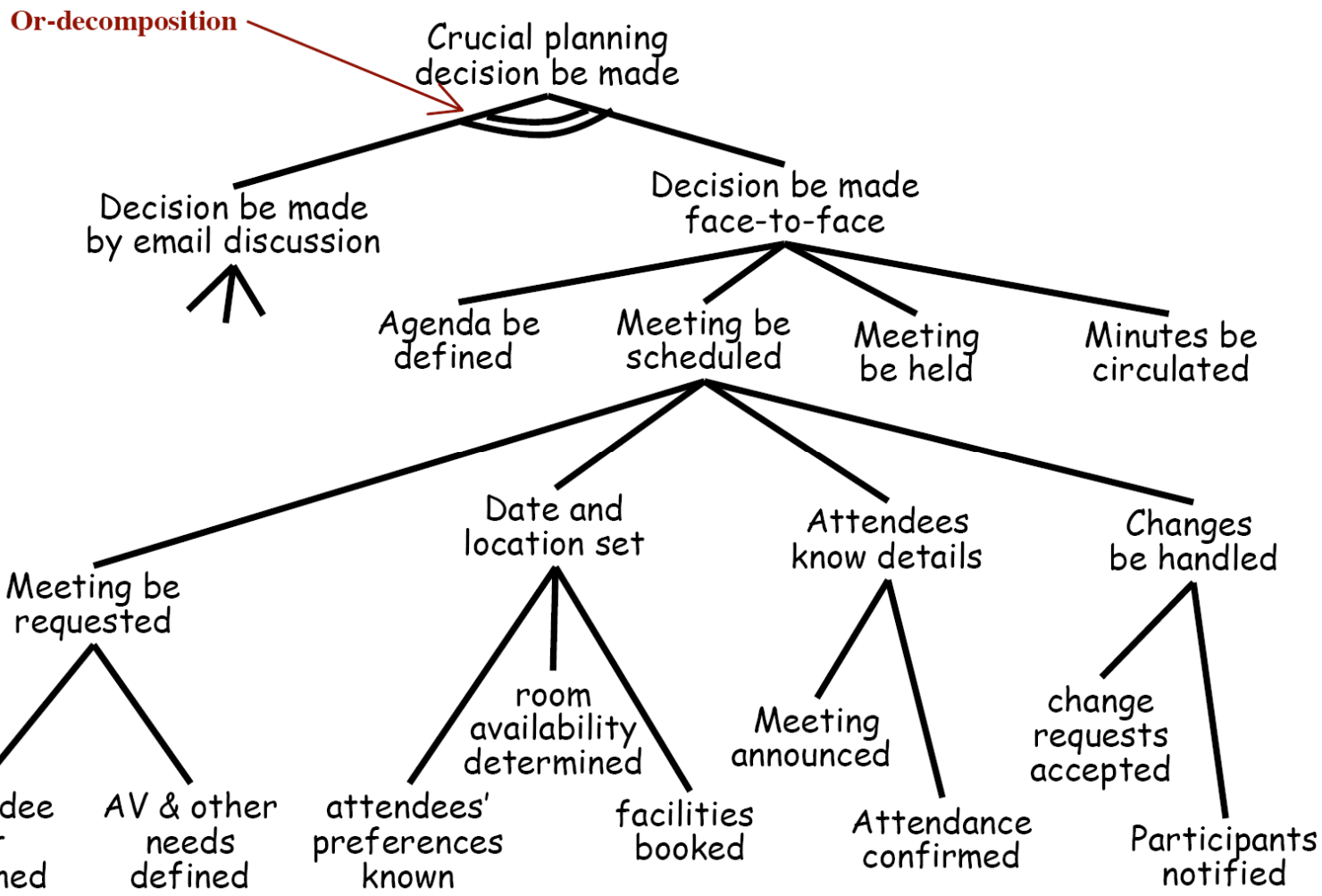
- Pros
  - Reasonably intuitive
    - Explicit declaration of goals provides sound basis for conflict resolution

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

Goal Modeling

- **(Hard) Goal**
  - Describe functions that must be carried out.
- **Softgoal**
  - Cannot really be fully satisfied.
    - Accuracy, Performance, Security, etc.
- Modelling Tips:
  - Multiple sources yield better goals
  - Associate stakeholders with each goal
    - Reveals various viewpoints and conflict
  - Use scenarios to explore how goals can be met
    - Explicit consideration of obstacles helps to elicit exceptions

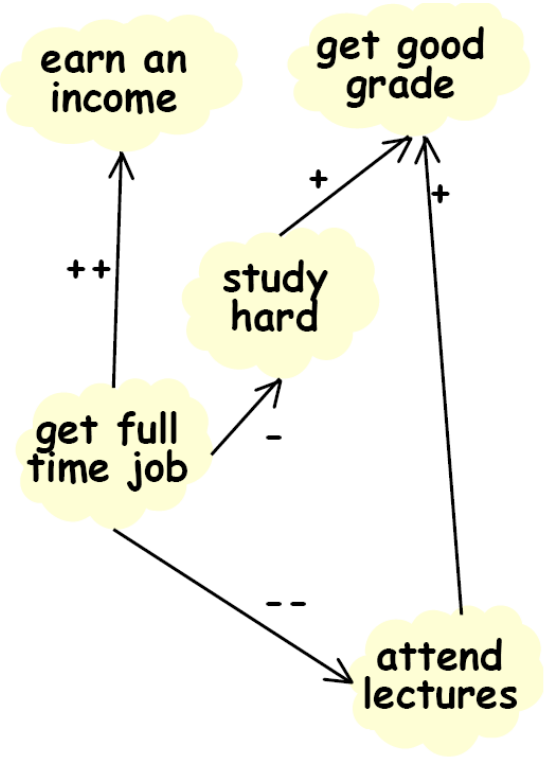
Example Goal Elaboration



Goal Analysis

- **Goal Elaboration**
  - "**Why**" questions explore higher goals (context)
  - "**How**" questions explore lower goals (operations)
  - "**How else**" questions explore alternatives

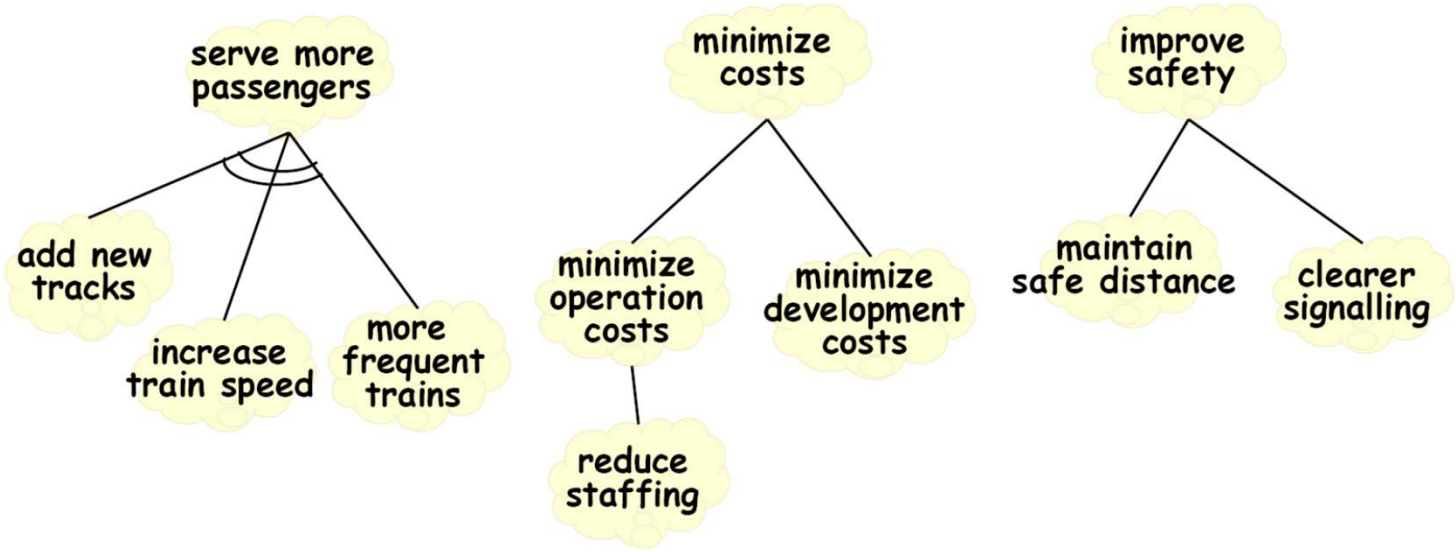
- Relationships between goals
  - One goal helps achieve another (+)
  - One goal hurts achievement of another (-)
  - One goal makes another (++)
    - Achievement of goal A guarantees achievement of goal B
  - One goal breaks another (--)
    - Achievement of goal A prevents achievement of goal B



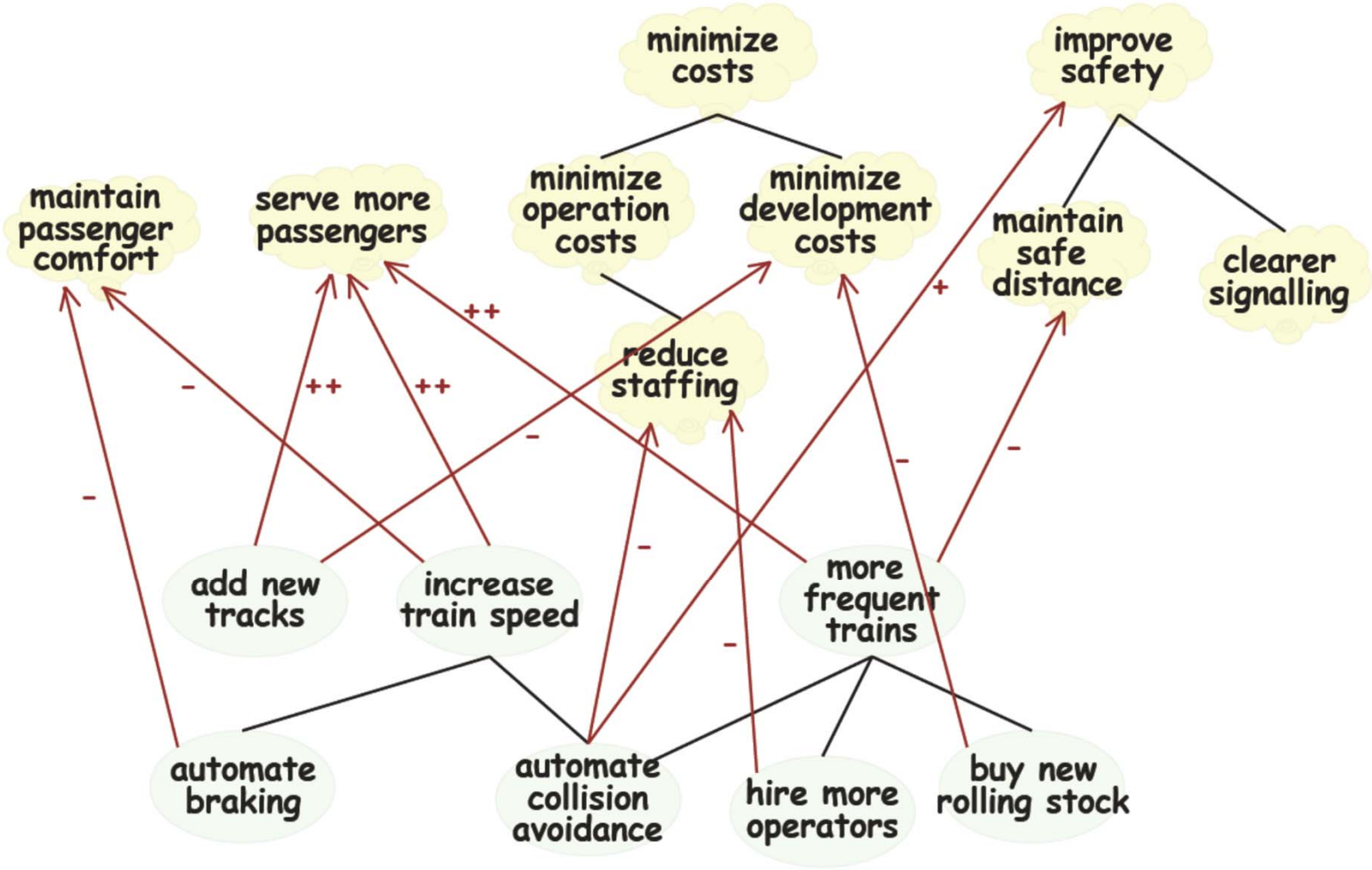
Softgoals

- Softgoals: Goals can never be fully satisfied.
  - E.g., "***system should be easy to use***" , "***access should be secure***"
  - Also known as **NFR**(Non-Functional Requirements) or **Quality attributes/requirements**
  - We have to look for things that contribute to satisfying softgoals.

  - Example: for a train system, we have several softgoals.



Softgoals as Selection Criteria



The Requirements Elicitation and Analysis Activities

1. Requirements Discovery

- Interacting with stakeholders to discover their requirements
- Domain requirements are also discovered at this stage.

2. Requirements Classification and Organization

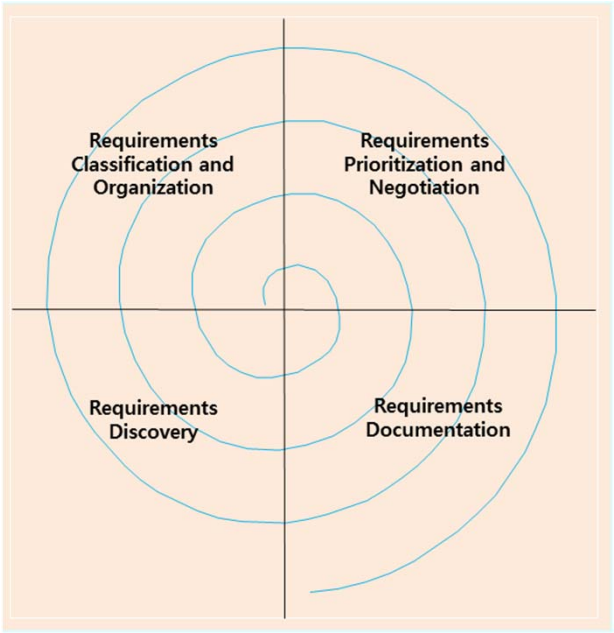
- Groups related requirements and organizes them into coherent clusters

3. Prioritization and Negotiation

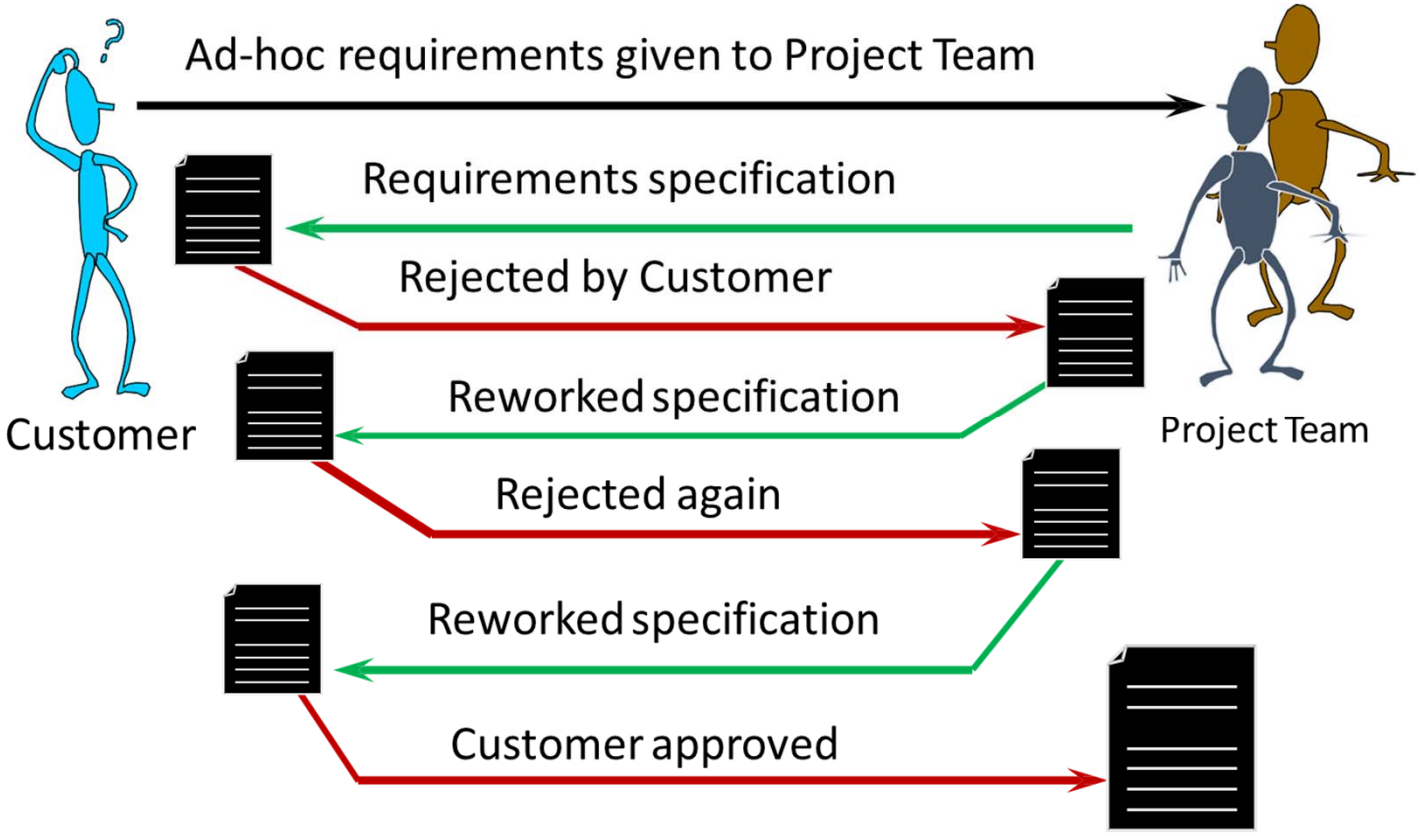
- Prioritizing requirements and resolving requirements conflicts

4. Requirements Documentation

- Document requirements
- Input it into the next round of the spiral



A Typical Situation Encountered



Things to Remember When Eliciting Requirements

1. Don’t Lose Sight of the Goal
2. Think Who’s Smart
3. A Single Stakeholder Can’t Speak for All
4. Use Appropriate Elicitation Methods
5. Accept Requirements Changes
6. Manage Elicited Requirements

1. Don’t Lose Sight of the Goal

- Establish the system’s vision and scope to reduce the risk of building the wrong system
- Try to obtain early commitment from stakeholders

2. Think Who's Smart

- Don't try to convince stakeholders that YOU are smart.
- Instead take everybody to show you think the STAKEHOLDER is smart

- Contrast these 2 cases:

1. My Elevators Are Too Slow!

2-1. I See. Tell Me Why You Feel They Are Too Slow.

2-2. I Don't Think So. I Think You Have an Elevator Throughput Problem, not a Speed Problem.

3. A Single Stakeholder Can't Speak for All

Stakeholder	Role
User	• Users of the system and the results of the system • ALWAYS included • Often many classes - make sure all are represented
Customer	• People with decision making authority • ALWAYS included; no project otherwise! • Often many classes - make sure all are represented • Closely aligned with marketing function
Marketing	• ESSENTIAL; The experts in the "market" • Too easy for development to dismiss them • In a commercial setting, they know the pulse of customers
Subject Matter Experts (SME)	• Helpful to learn foundation requirements • Helpful to alleviate disagreements among stakeholders
Developer	• Helpful to learn system implications • Helpful to learn evolution / maintenance requirements
Development Managers	• Knows the development capability and resources
Tester	• Useful a bit later in project • Knows which requirements are testable
Loser Users	• People who loses power as a result of the project • Useful if a system has "loser users"
Technical Writers Trainers / Customer Support	• Can also help • Experts in making the system easy to use/teach/explain

4. Use Appropriate Elicitation Methods

- **Methods** for requirements elicitation:
  - **Interviews**
  - **Role Playing**
  - **Brainstorming**
  - **Requirements Workshop**
  - **Prototyping**
  - **Survey/Questionnaire**
  - **Use Case**

- **A single method may not be sufficient**.
  - Consider requirements' size, complexity, etc., and select several ones.

5. Accept Requirements Changes

- **Requirements change is inevitable.**
  - Clients have right to change requirements.
  - The more features the product has, the more customers want.

- Don’t ever ask:
  - “Okay, is that your final requirement ?”

- Change is not a threat, it’s an opportunity.

6. Manage Elicited Requirements

- Record the **rationale** of each requirement
  - Reason why requirement is necessary
  - Assumptions on the requirement

- Managing the rationale with **annotated requirements lists**
  - Do not simply rewrite the requirement
  - Make it unique for each requirement
  - Keep it simple

- Example:
  - Requirements : "*The truck shall have a height of no more than 14 feet*."
  - Rationale : "*99% of all U.S. Interstate highway overpasses have a 14-foot or greater clearance.*"

Techniques for Eliciting Stakeholder Needs

- **Requirements Elicitation Methods**
  1. Requirements workshop
  2. Brainstorming
  3. Storyboards
  4. Interviews
  5. Questionnaires
  6. Role playing
  7. Prototypes
  8. Review customer requirement specification

1. Requirements Workshop *

- **Gather all stakeholders together** for an intensive and focused period
  - Create consensus on the scope, risks and key features of the software system
  - Results immediately available
  - Outputs:
    - Problem statement , Key features , Initial business object model, Use-case diagram , Prioritized risk list, etc.

- Provide a **framework** for applying other elicitation techniques such as
  - Brainstorming, use-case workshops, storyboarding, etc.

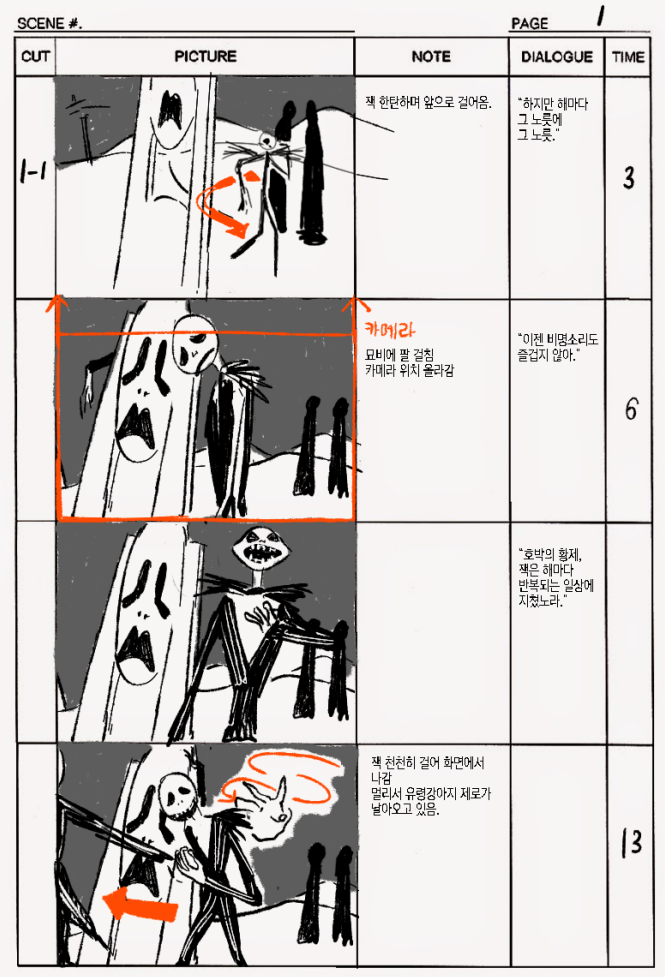
2. Brainstorming *

- **Rules for Brainstorming**
  - Clearly state the objective of the session
  - Generate as many ideas as possible
  - Let your imagination soar
  - Do not allow criticism or debate
  - Mutate and combine ideas
- **Generate as many ideas as possible**
  - Even the impractical, absurd ideas should not be neglected
  - Merge the various ideas to create new ideas
- **Express freely**
  - Do not explain or specify the ideas
  - Do not evaluate or argue about the ideas
  - Do not put names on the ideas
  - Encourage the unexpected and imaginative
- **Put up ideas openly**
  - Ideas should be put up on a whiteboard where all can see
  - Participants themselves may put up ideas on the board
  - Put tabs of Post-Its on the center table

3. Storyboards

- Visually tell and show:
  - Who/what the players are (actors)
  - What happens to them
  - When it happens

- Benefits
  - Help gather and refine customer requirements
  - Encourage creative and innovative solutions
  - Encourage team review
  - Prevent features that no one wants
  - Ensure that features are implemented in an accessible and intuitive way
  - Ease the interviewing process
  - Help to avoid blank-page syndrome



4. Interviews

- Provide **a simple and direct technique** to gain understanding of problems and solutions

- Types of interviews
  - **Open interview**
    - No pre-set agenda
    - Irrelevant data can be gathered
    - Needs time and training
  - **Closed interview**
    - Fairly open-questions agenda
    - Needs extended preparation
    - Prevents biases

- Interview tips
  - Avoid asking people to describe things they don't usually describe
    - Example: Describe how to tie your shoes
  - Avoid "Why…?" questions
  - Ask open-ended (context-free) questions
    - High-level abstract questions

5. Questionnaires

- Give access to a wide audience
  - Apply to broad markets where questions are well-defined
- **Statistical analysis** is applicable.
  - Powerful, but not a substitute for an interview
- Assumptions:
  - Relevant questions can be decided in advance
  - Questions phrased, so reader hears as intended

고 객 명		소 속		조사일자	20 년 월 일
■ 설문조사내용					
구 분	평 가 내 용		평 가		점 수
품질	1. 고객께서 알고 있는 회사명의 이미지는 어떠하십니까?		A : 좋 다 B : 보 통 C : 나쁘다		
	2. 고객께서는 회사명으로 전화를 하셨을 때 친절하게 응대를 하였습니까?		A : 좋 다 B : 보 통 C : 나쁘다		
	3. 고객께서는 회사명의 제품에 만족 하십니까?		A : 만 족 B : 보 통 C : 불만족		
	4. 당사의 제품에 이상 발생 시 고객께서 요구했을 때 신속한 대응이 있었습니까?		A : 그렇다 B : 보 통 C : 아니다		
	5. 품질관련해서 동종업계화의 수준에 있어 어느 정도의 수준이라고 생각하십니까?		A : 상 위 B : 중 위 C : 하 위		
	평 가 점 수 A : 20 B : 15 C : 10			소 계	
생산기술	1. 회사명을 과거에 방문하신 적이 있으면 어떤 느낌을 받았습니까?		A : 좋 다 B : 보 통 C : 나쁘다		
	2. 회사명의 생산기술에 있어 동종 업계와의 기술수준은 어느 정도라고 생각하십니까?		A : 상 위 B : 중 위 C : 하 위		
	3. 제품에 대한 상호간 정보교환은 잘 이루어지고 있습니까?		A : 그렇다 B : 보 통 C : 아니다		
	4. 신규 제품 개발 시 경쟁업체와 비교했을 때 개발일정 및 진행이 잘 되고 있습니까?		A : 그렇다 B : 보 통 C : 아니다		
	평 가 점 수 A : 25 B : 20 C : 15			소 계	
영업	1. 당사의 납입 준수율은 어느 정도라고 생각하십니까?		A : 80%이상 B : 50%이상 C : 50%미만		
	2. 당사의 납품수량은 정확하다고 생각하십니까?		A : 정 확 B : 보 통 C : 미 흡		
	3. 회사명의 서비스(친절함) 수준은 어느 정도라고 생각하십니까?		A : 좋 다 B : 보 통 C : 나쁘다		
	4. 포장 및 제품에 청결함과 더불어 사용하는데 불편함이 있었습니까?		A : 그렇다 B : 보 통 C : 아니다		
	평 가 점 수 A : 25 B : 20 C : 15			소 계	
고객의견				평 점	

6. Role Playing *

- Perform requirements elicitation from the viewpoint of the roles
  - Learns and performs user's job
  - Performs a scripted walkthrough

- Advantages
  - Gain real insights into the problem domain
  - Understand problems that users may face

7. Prototypes

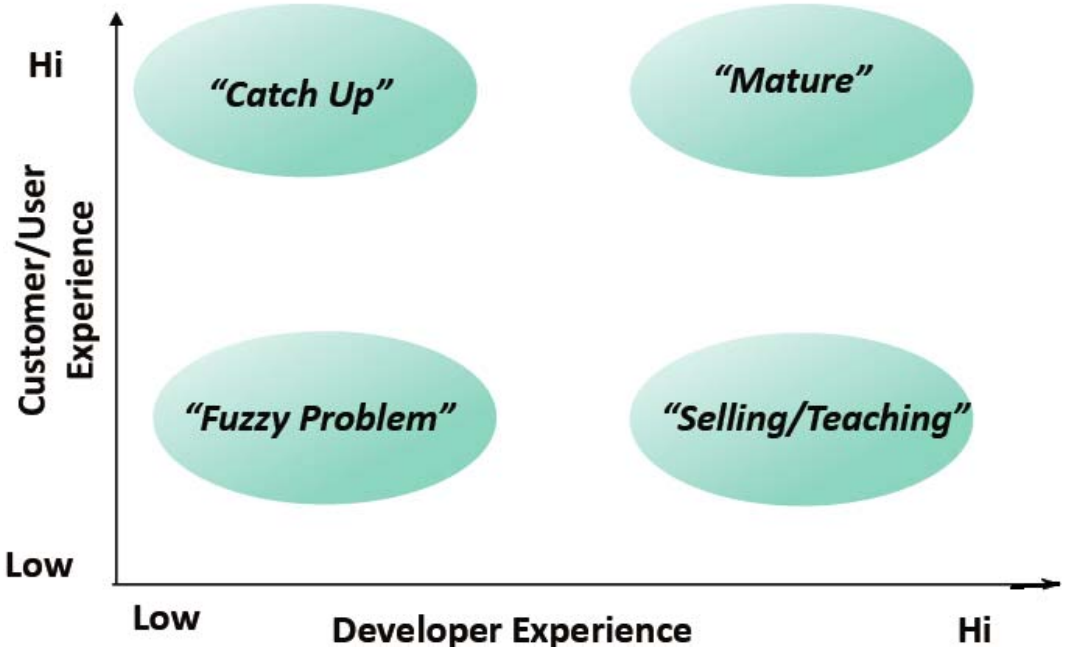
- Demonstrate some or all of the externally observable behaviors of a system through building prototypes quickly

- Used to:
  - Demonstrate understanding of the problem domain
  - Gain feedback on proposed solution
  - Validate known requirements
  - Discover unknown requirements
  - Create simulations
  - Elicit and understand requirements
  - Prove and understand technology
  - Reduce risk
  - Enhance shared understanding
  - Improve
    - Cost and schedule estimates
    - Feature definition

8. Review Customer Requirement Specifications

- **Customer Requirements Review**
  - Recognize and label
    - Application behaviors
    - Behavioral attributes
    - Issues and assumptions
  - **Ask customers directly**

Which Techniques to Use?

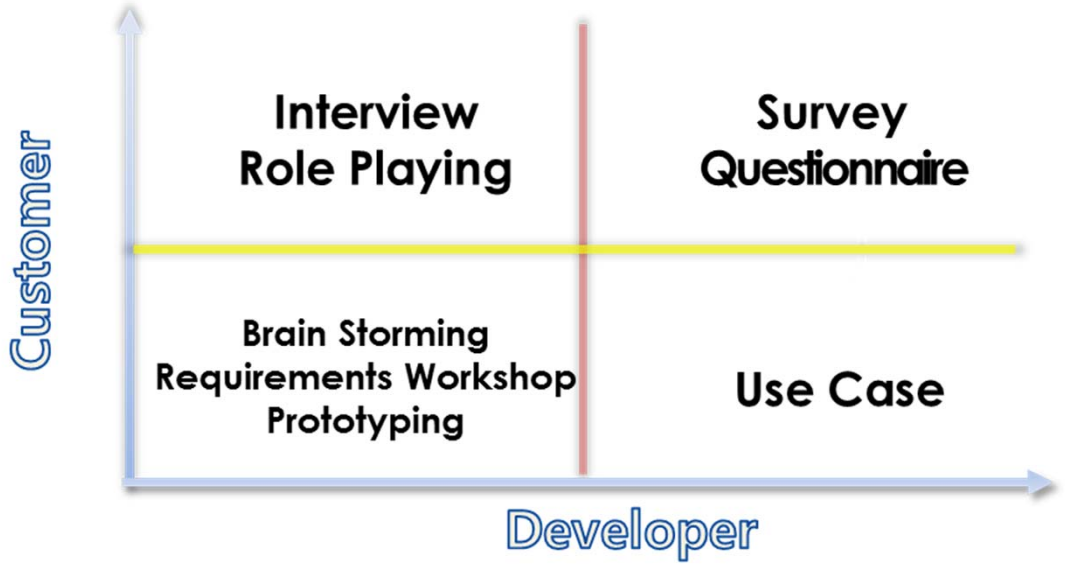


- **Catch Up**
  - Role Playing
  - Interview
  - Requirements Review
- **Fuzzy Problem**
  - Requirements Workshops
  - Brainstorming
  - Storyboards
- **Selling / Teaching**
  - Use Case
  - Business Modeling
- **Mature**
  - Questionnaires
  - Prototyping

Which Techniques to Use?

- **No single technique is sufficient** for realistic projects.

- Appropriate method should be chosen based on:
  - The size and complexity of requirements
  - Problem domain
  - The number of involved stakeholders



What to Do with Elicited Requirements?

- **Maintain requirements in lists**
  - Maintaining **a list of requirements** can support all activities of requirements.

- Enables you to answer questions such as:
  - How many requirements do you have?
  - How many high priority requirements do you have?
  - What percentage of the candidate requirements have you chose to satisfy in your next release?
  - What percentage of the requirements deemed high priority by customer X are you satisfying with?

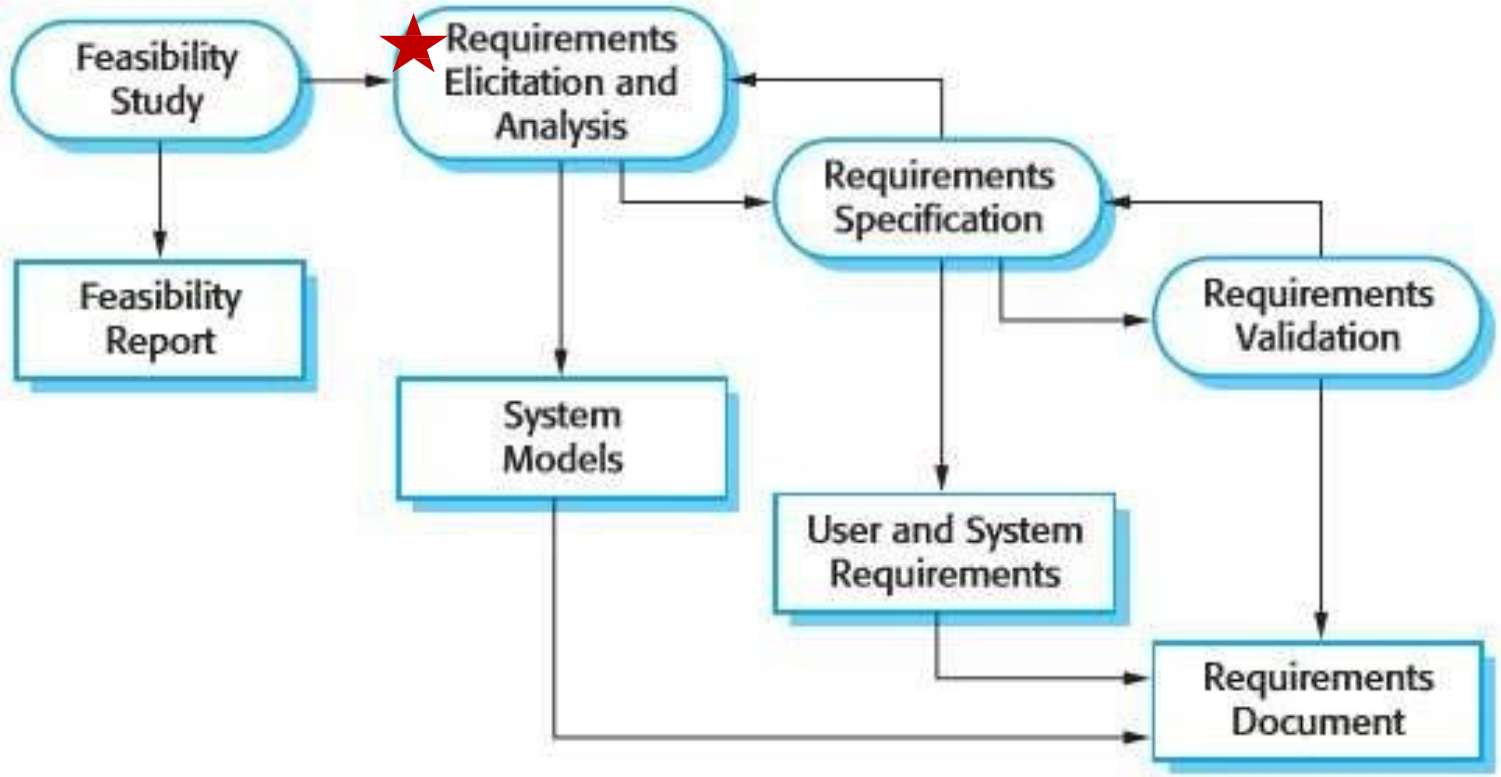
Example of Elicitation Results

- **A list of annotated requirements**

ID	Requirement Text
961	No formal training shall be required to operate the RLM.
955	Any new releases or versions of the software shall be sold as new products. Users must p...
954	User software will not be modified or upgraded.
512	The RLM shall return to the refuel location or dump area to within 10 cm of the user-define...
432	Pressing the screen in an area without a command shall make no sound nor shall it be int...
415	The screen shall be capable of displaying alphanumeric data in blocked, uppercase char...
500	The RLM shall accept lawn and obstacle programming from the user. During programming, th...
321	The RLM shall initiate communications with the GPS through external interface EL-GPS
300	The RLM shall interface with two different external systems, The GPS and the Electronically S...
310	External interfaces include the receipt of location data from GPS and detection of obstacles...
511	The RLM shall not overcut or undercut the border and user defined obstacles by more tha...
510	The RLM shall cut the lawn only within the area defined by the user during the programming.
550	Border programming shall be required to be completed by the user prior to accepting the oth...
411	The Screen shall be 16.25 mm (high) by 105 mm (wide) and capable of displaying two row...
446	Serious errors (for example, blade fouling, Requirement 179) shall not have a button on th...
553	Programming border data shall be terminated by a user request, or when the RLM returns t...
554	After the termination, the RLM shall be ready to receive another command.
418	The screen shall be used to display information from the RLM to the user and accept dire...
561	User shall guide the RLM to the obstacle and indicated that the boundary of obstacle will ...
562	RLM shall record sufficient data (e.g. from GPS) to meet the accuracy requirements stated...
552	RLM shall record sufficient data (e.g. from GPS) to meet accuracy requirements stated in ...
551	User shall guide the RLM to the border of the lawn and indicate that the boundary will be ...

4. Requirements Negotiation

Requirements Engineering Process



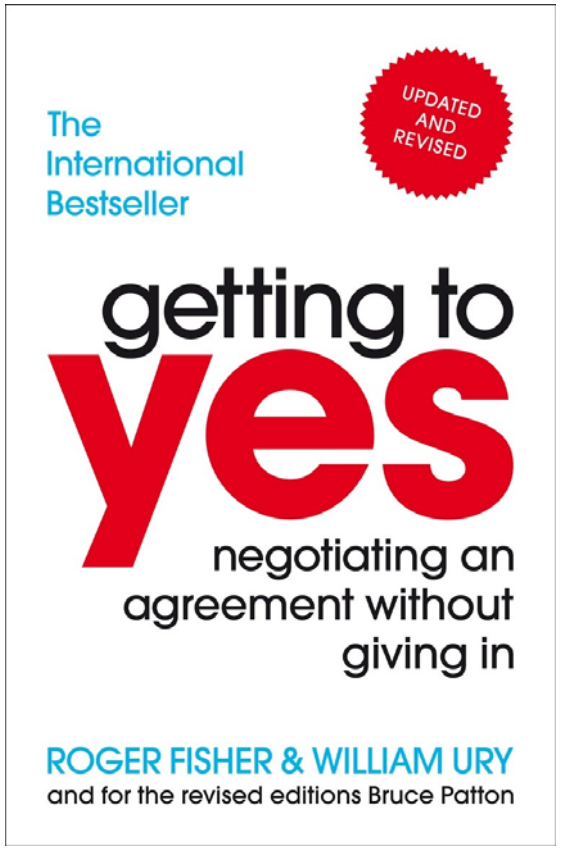
Needs for Requirements Negotiation

- **Requirements are negotiated** to achieve mutually satisfactory agreements.
  - Users, customers, managers, domain experts, and developers share different skills, backgrounds and expectations.
  - Requirements emerge from a process of co-operative learning in which they are explored, prioritized, negotiated, evaluated, and documented.

Negotiation Principles

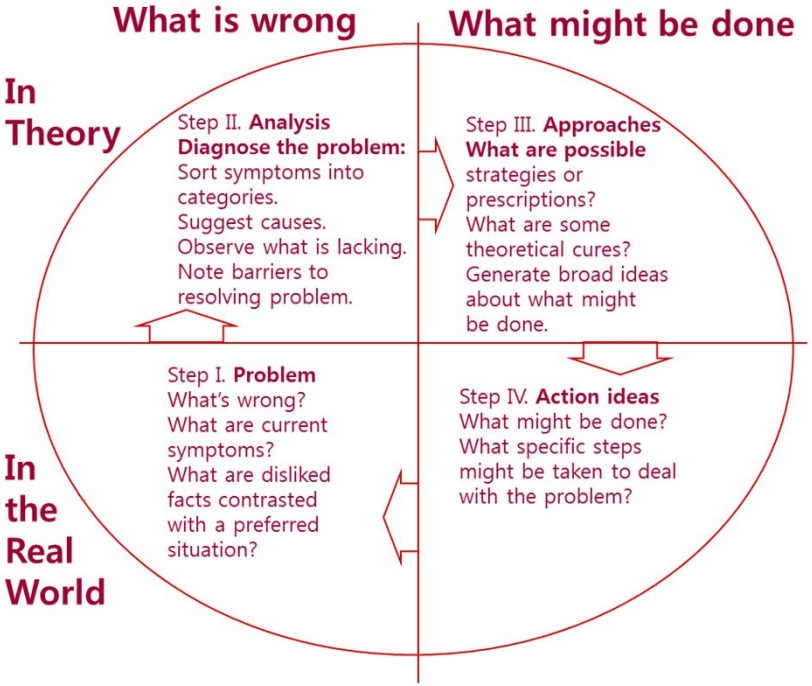
- [Fisher & Ury, "Getting to Yes," 1981]
  - **"*Don't bargain over positions*"**

- Use **4-step solution** approach
  - Separate the people from the problem
  - Focus on interests, not positions
  - Invent options for mutual gain
  - Insist on using objective criteria



WinWin

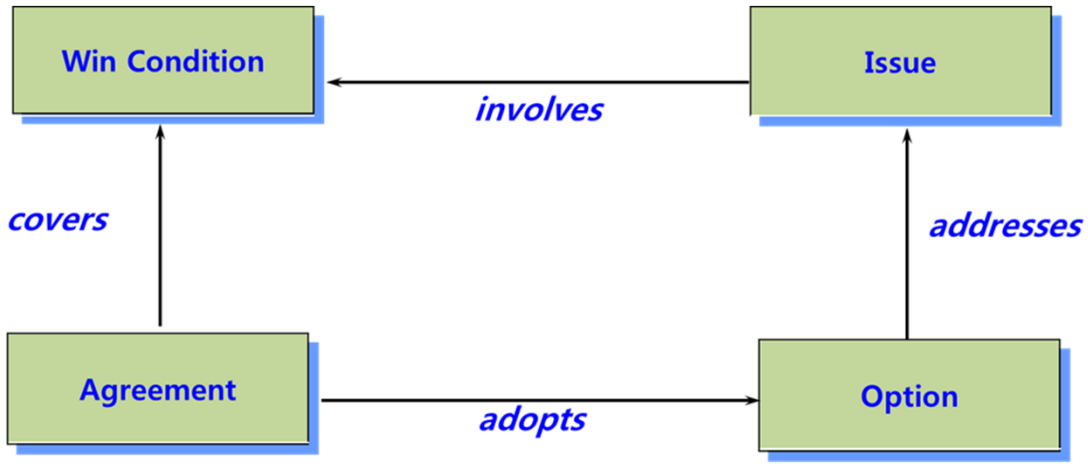
- **The WinWin approach**
  - A set of principles, practices and tools
  - Enabling a set of interdependent stakeholders to work out a mutually satisfactory (win-win) set of shared commitments
  - **Win-lose generally becomes Lose-lose.**
    - Actually, **nobody wins** in these situations.



Key Concepts in WinWin

- **Win Condition**: objective which makes a stakeholder feel like a winner
- **Issue**: conflict or constraint on a win condition
- **Option**: a way of overcoming an issue
- **Agreement**: mutual commitment to an option or win condition

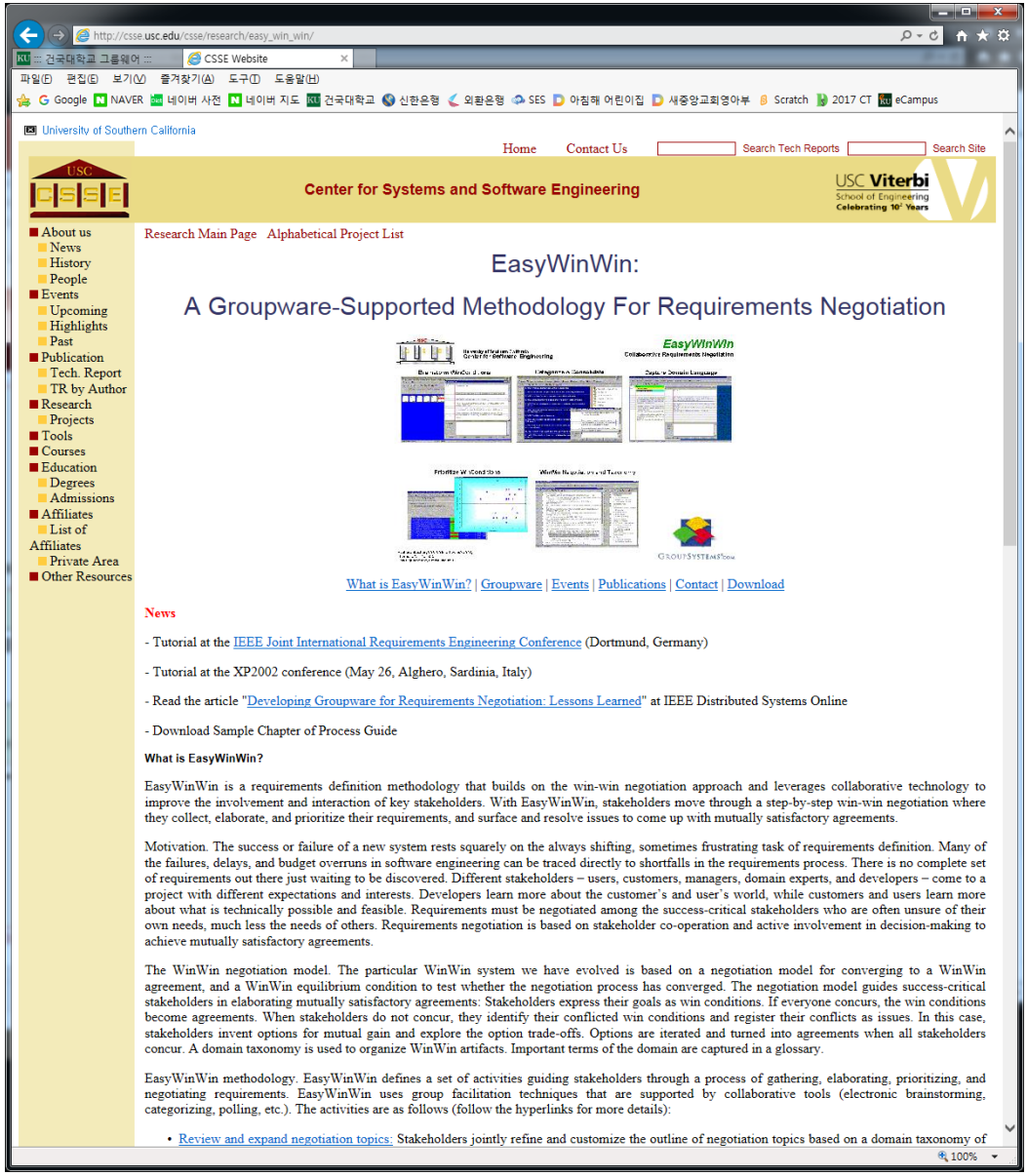
- **WinWin Equilibrium State**
  - All Win Conditions are covered by Agreements
  - No outstanding Issues



Steps of WinWin

1. Identify success-critical stakeholders
2. Identify stakeholders' win conditions
3. Identify issues conflicting win conditions
4. Negotiate top-level win-win agreements
   - Invent options for mutual gain
   - Explore option tradeoffs
   - Manage expectations
5. Embody win-win agreements into specs and plans
6. Elaborate steps 1-5 until product is fully developed
   - Confront, resolve new win-lose, lose-lose risk items

The WinWin in Requirements Engineering



http://csse.usc.edu/csse/research/easy_win_win/

Using the WinWin Spiral Model: A Case Study

Fifteen teams used the WinWin spiral model to prototype, plan, specify, and build multimedia applications for USC's Integrated Library System. The authors report lessons learned from this case study and how they extended the model's utility and cost-effectiveness in a second round of projects.

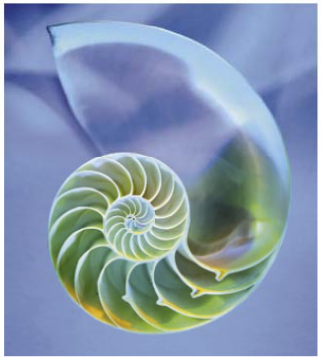
Barry Boehm
Alexander Egyed
Julie Kwan
Dan Port
Archita Shah
University of Southern California

Ray Madachy
Litton Data Systems and University of Southern California

At the 1996 and 1997 International Conferences on Software Engineering, three of the six keynote addresses identified negotiation techniques as the most critical success factor in improving the outcome of software projects. At the USC Center for Software Engineering, we have been developing a negotiation-based approach to software system requirements engineering, architecture, development, and management. Our approach has three primary elements:

- *Theory W*, a management theory and approach, which says that making winners of the system's key stakeholders is a necessary and sufficient condition for project success.[1]
- *The WinWin spiral model*, which extends the spiral software development model by adding Theory W activities to the front of each cycle. The sidebar "Elements of the WinWin Spiral Model" describes these extensions and their goals in more detail.
- *WinWin*, a groupware tool that makes it easier for distributed stakeholders to negotiate mutually satisfactory (win-win) system specifications.[2]

In this article, we describe an experimental validation of this approach, focusing on the application of the WinWin spiral model. The case study involved extending USC's Integrated Library System to access multimedia archives, including films, maps, and videos. The Integrated Library System is a Unix-based, text-oriented, client-server COTS system designed to manage the acquisition, cataloging, public access, and circulation of library material. The study's specific goal was to evaluate the feasibility of using the WinWin spiral model to build applications written by USC graduate student teams. The students developed the applications in concert with USC library clients, who had identified many USC multimedia archives that seemed worthy of transformation into digitized, user-interactive archive management services.



The study showed that the WinWin spiral model is a good match for multimedia applications and is likely to be useful for other applications with similar characteristics—rapidly moving technology, many candidate approaches, little user or developer experience with similar systems, and the need for rapid completion. The study results show that the model has three main strengths.

- *Flexibility*. The model let the teams adapt to accompanying risks and uncertainties, such as a rapid project schedule and changing team composition.
- *Discipline*. The modeling framework was sufficiently formal to maintain focus on achieving three main, or "anchor-point," milestones: the life-cycle objectives, the life-cycle architecture, and the initial operational capability. (Table A in the sidebar describes these milestones.)
- *Trust enhancement*. The model provided a means for growing trust among the project stakeholders, enabling them to evolve from adversarial, contract-oriented system development approaches

0018-9162/98/$10.00 © 1998 IEEE

Exercise 2 : Requirements Elicitation

- Let's do **Requirements Workshop**
  - Supposed to develop **a new advanced OOO digital watch** **next season**
  - **Brainstorming** for eliciting requirements with markers and Post-Its®
    - Eliciting **30** ideas
  - **Role playing** of 13 different roles :
    - User (전자시계 동호회 대표), 백화점 판매팀장, CEO, Marketing Manager, 상품기획팀장, HW/UX Designer, Project Team Leader, SW Engineer, HW Engineer, Security Manager, UI Designer, Product Quality Manager, SW Tester
  - **Requirements Negotiation (WinWin)**
    - List-up your features and requirements
    - Categorize and analyze them with respect to roles and values
    - Perform negotiations to select/revise requirements

- **What features/requirements** should we consider?
  - Basic HW features :
    - 4 Buttons , 1 Buzzer , 1 LCD , GPS, LTE, Wi-Fi, 1 SW downloadable
  - Addition/Extensions are available.
    - Bluetooth, 5G, Camera, Sensors, etc.
    - Decide the selling price as well as the HW/SW features
  - How could you resolve different values of different roles?

- **WinWin** analysis for each requirements discovered (total 30 requirements)

Stakeholders	Candidate Requirements								
	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
CEO	O								
Marketing Manager	O								
Project Team Leader	X								
SW Engineer	X								
HW Engineer	△								
Designer	△								
Others, if any									

for each requirement,

Stakeholders	Candidate Requirement [1]				
	[1]	Issues	Options	Agreements	Total Agreement
CEO	O				Revised Requirement [1]
Marketing Manager	O				
Project Team Leader	X				
SW Engineer	X				
HW Engineer	△				
Designer	△				
Others, if any					