# Software Engineering

# (2) Domain Analysis Requirements Analysis

Sokendai / National Institute of Informatics

Fuyuki Ishikawa / 石川　冬樹

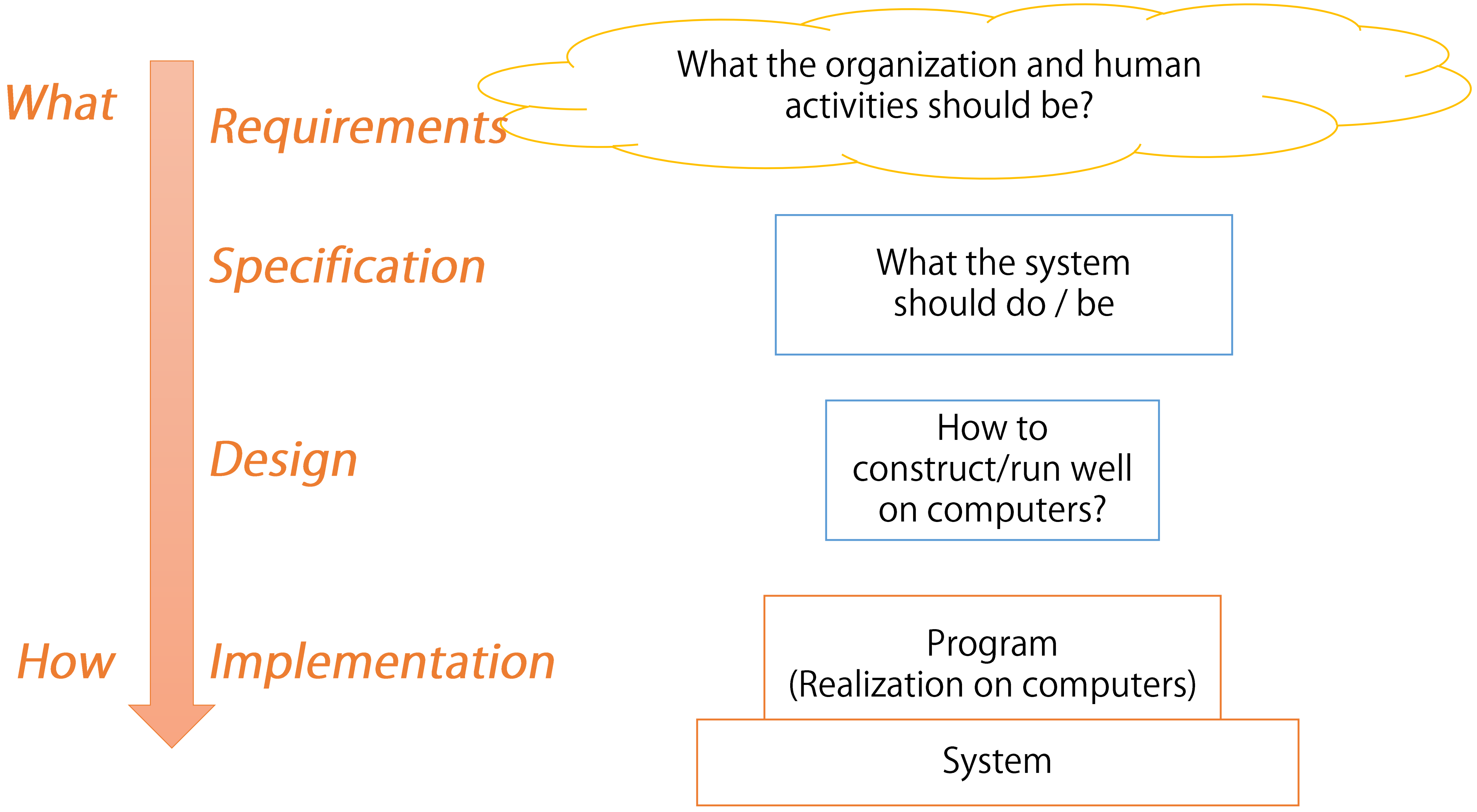
f-ishikawa@nii.ac.jp / @fyufyu

http://research.nii.ac.jp/~f-ishikawa/

大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

## TOC

# [Review] Abstraction Level in Software Development


What
Requirements
What the organization and human activities should be?
Specification
What the system should do / be
Design
How to construct/run well on computers?
How
Implementation
Program (Realization on computers)
System

## Software Requirements

- Software Requirements:
  Properties that should be satisfied to solve problems in the real world
- Requirements Engineering （要求工学）：
  Engineering about elicitation, analysis, specification, validation, and management of software requirement
- Keywords
  - Problem Domain（問題領域）, before Solution（解決領域）
  - Stakeholders

# Significance of Requirements

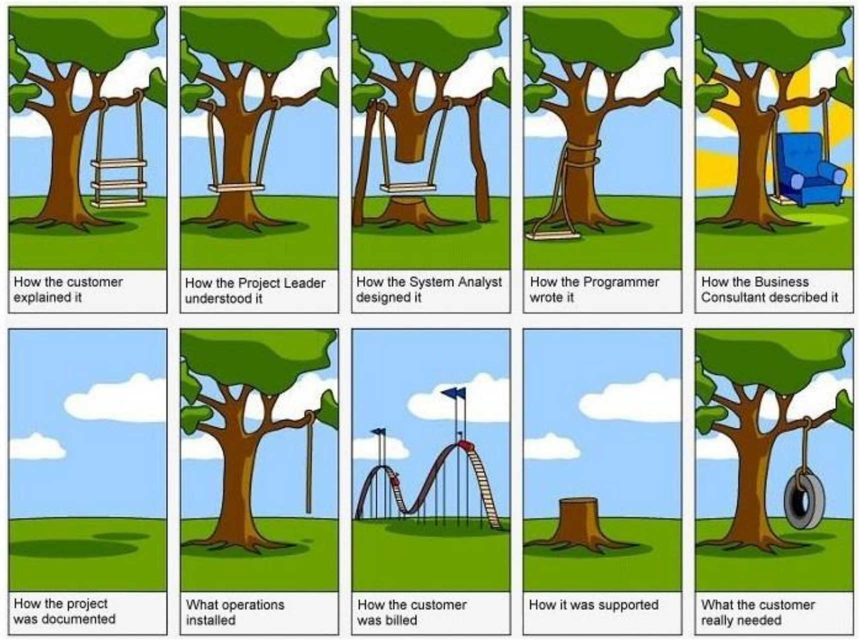
- Consume 30-50% of resource with problematic requirements

[ B.W. Boehm et al., Understanding and controlling software costs, 1988 ]

- Problems in requirements occupy 55% of delay causes

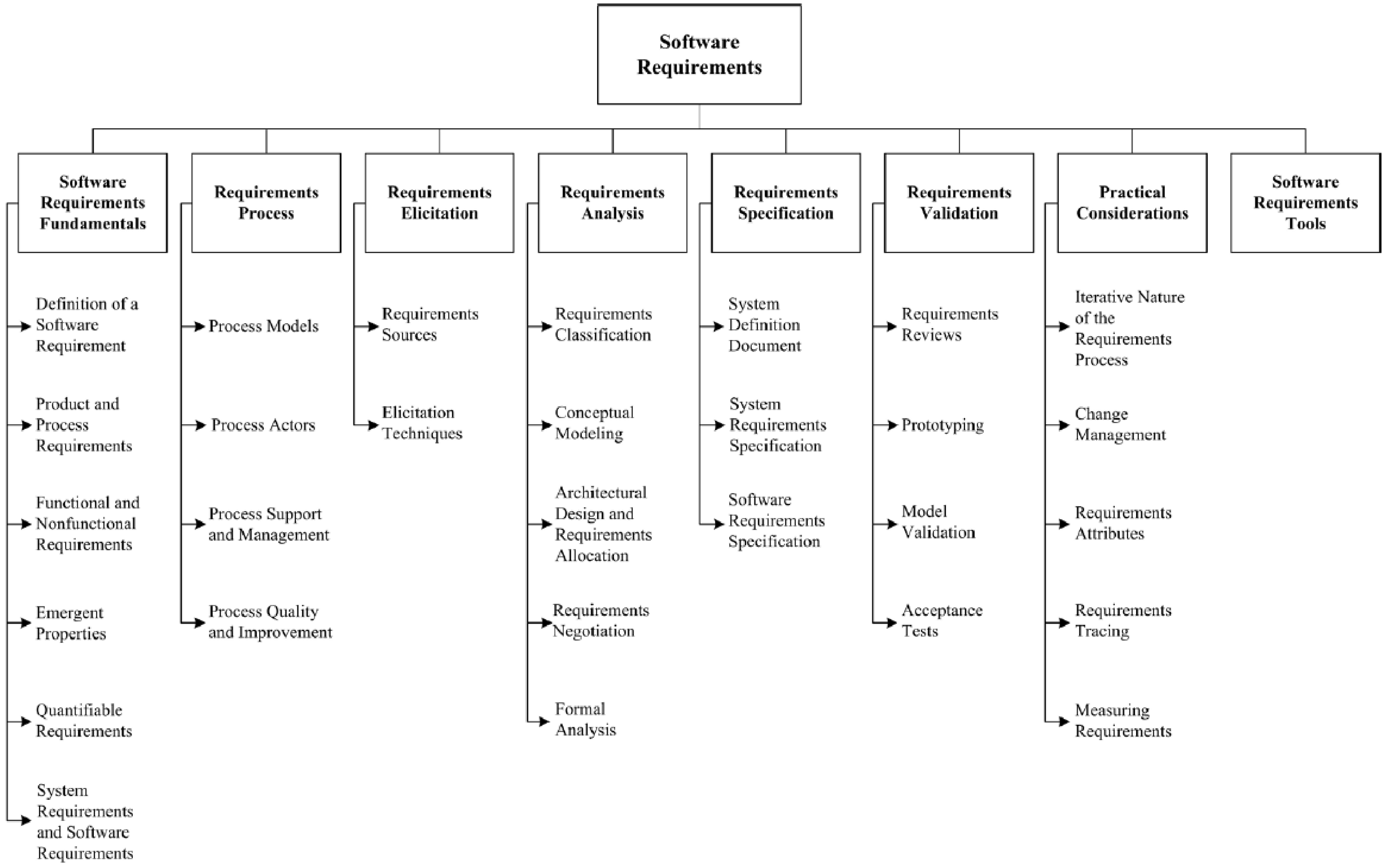
[ JUAS・ユーザー企業ソフトウェアメトリックス調査2016 ]

- Famous irony


How the customer explained it
How the Project Leader understood it
How the System Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

[ https://danielksmith.wordpress.com/2012/09/25/what-the-customer-really-wanted/ ]
(I could not trace into the original source)

# Requirements Knowledge in SWEBOK


Software Requirements
Software Requirements Fundamentals
Definition of a Software Requirement
Product and Process Requirements
Functional and Nonfunctional Requirements
Emergent Properties
Quantifiable Requirements
System Requirements and Software Requirements
Requirements Process
Process Models
Process Actors
Process Support and Management
Process Quality and Improvement
Requirements Elicitation
Requirements Sources
Elicitation Techniques
Requirements Analysis
Requirements Classification
Conceptual Modeling
Architectural Design and Requirements Allocation
Requirements Negotiation
Formal Analysis
Requirements Specification
System Definition Document
System Requirements Specification
Software Requirements Specification
Requirements Validation
Requirements Reviews
Prototyping
Model Validation
Acceptance Tests
Practical Considerations
Iterative Nature of the Requirements Process
Change Management
Requirements Attributes
Requirements Tracing
Measuring Requirements
Software Requirements Tools

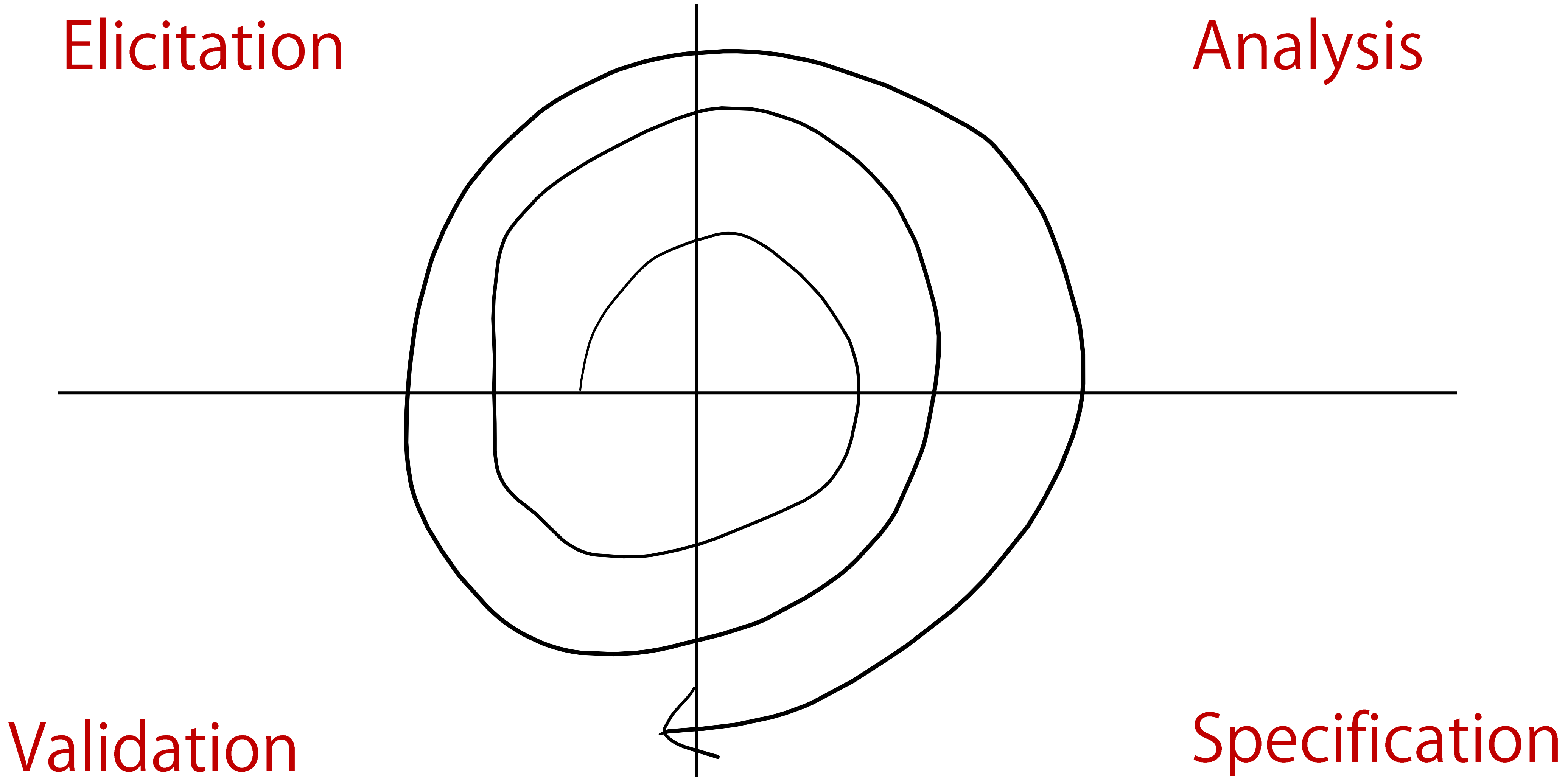
## Activities in Requirements Engineering

- Elicitation: elicit requirements through understanding the problems that should be solved

*Sometimes people prefer "discover" or "develop" to "elicit" or "extract"*

- Analysis: classify requirements, detect and resolve conflicts, clarify the boundary
- Specification: build documents for sharing, validation, and agreement
- Validation: check preciseness, consistency, completeness, …

## Iterative Nature of Requirements Engineering



## IEEE 830

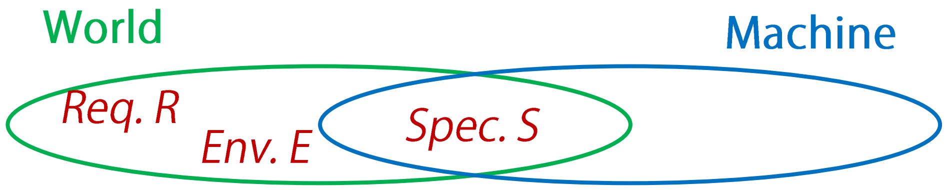
- IEEE 830 (1998): *IEEE Recommended Practice for Software Requirements Specifications*
  - Quality characteristics defined as follows

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

## One View of Requirements and World

- The model by Zave/Jackson

$$\boxed{S, E \models R}$$



*Does the specification S of the machine satisfy the requirements R under the environmental assumptions E?*

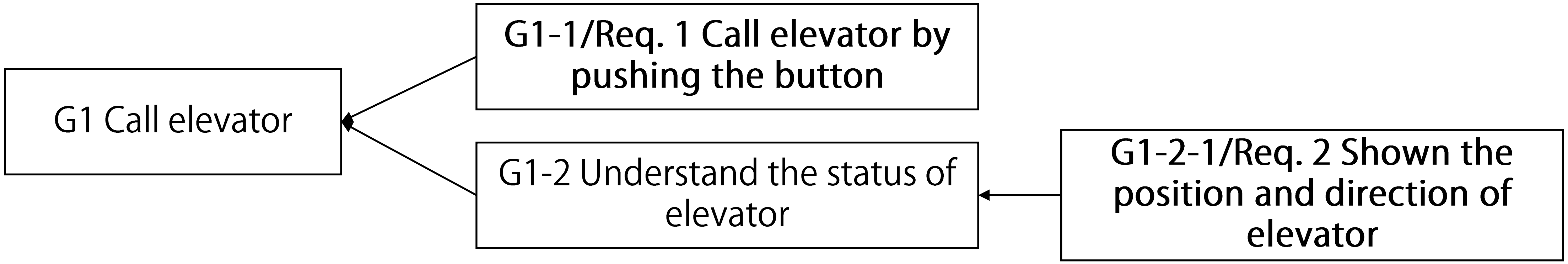
*[ image from http://www.fujitaka.com/ ]*

| | |
|---|---|
| R | • #enter ≦ #pay |
| S | • Unlock when pay occurs<br>• Lock when push occurs |
| E | • push and enter alternates<br>• push cannot occur after lock until unlock occurs |

[ Zave et al., Four Dark Corners of Requirements Engineering, 1997 ]

## Example of Terminology (High Diversity)

- Goal models
  - Goal: what (various) stakeholders want to achieve
  - Subgoal (of a goal): goals necessary to achieve the goal, possibly multiple (all necessary or alternatives)
  - Requirements: goals that have been concretized enough to judge the feasibility and chosen to be realized


G1 Call elevator
G1-1/Req. 1 Call elevator by pushing the button
G1-2 Understand the status of elevator
G1-2-1/Req. 2 Shown the position and direction of elevator

## Example of Terminology (High Diversity)

- 要求仕様
  - 要求が陽に文書化された場合に，こう呼ぶことが多い
- 実際にはいろいろな用語が使われている
  - 標準により異なる
  - 組織内用語が産まれる
  - 「真の要求」：前頁の用法で言うなら「顕在化しづらいゴール」
  - 「要求と要件」：前頁「ゴールと要求」の関係を指すことが多い
  - 単に言葉が適当なだけなことも

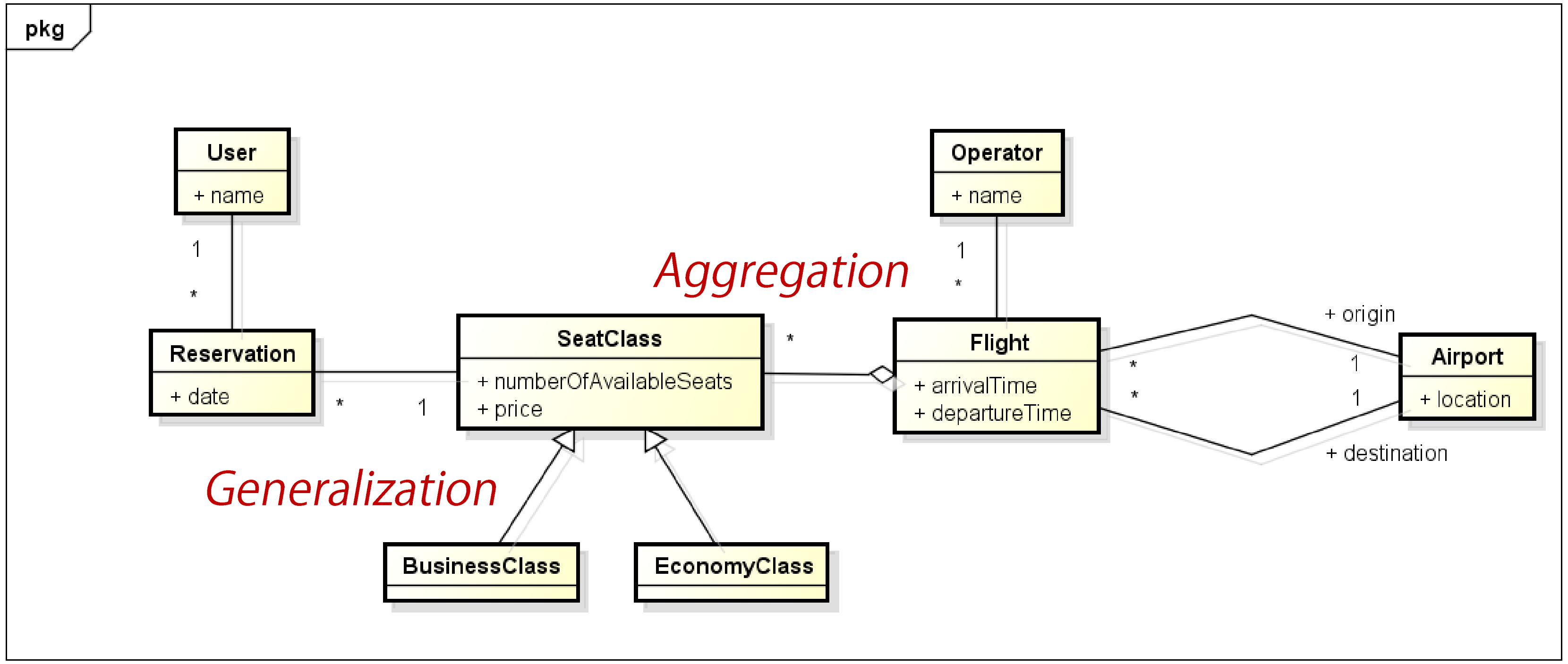
## TOC

## Domain Modeling and Analysis

- Domain Modeling and Analysis (Conceptual Modeling and Analysis)
  - Understand and represent concepts and relationships that constitute the target domain of the real world
  - Describe basically the As-Is without To-Be, or the system to be developed

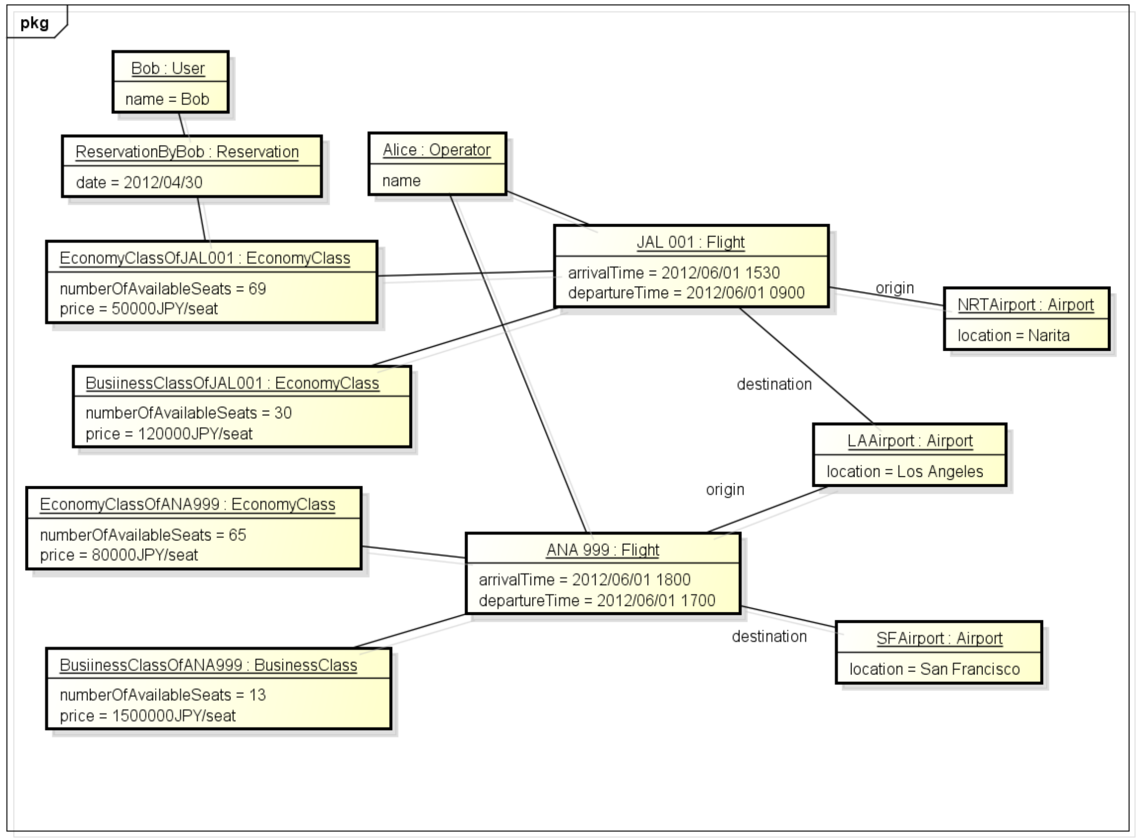
## Conceptual Modeling

- Concepts about flight reservation in the UML class diagram
  - Relations, especially, aggregation (has-a), generalization (is-a), and multiplicity



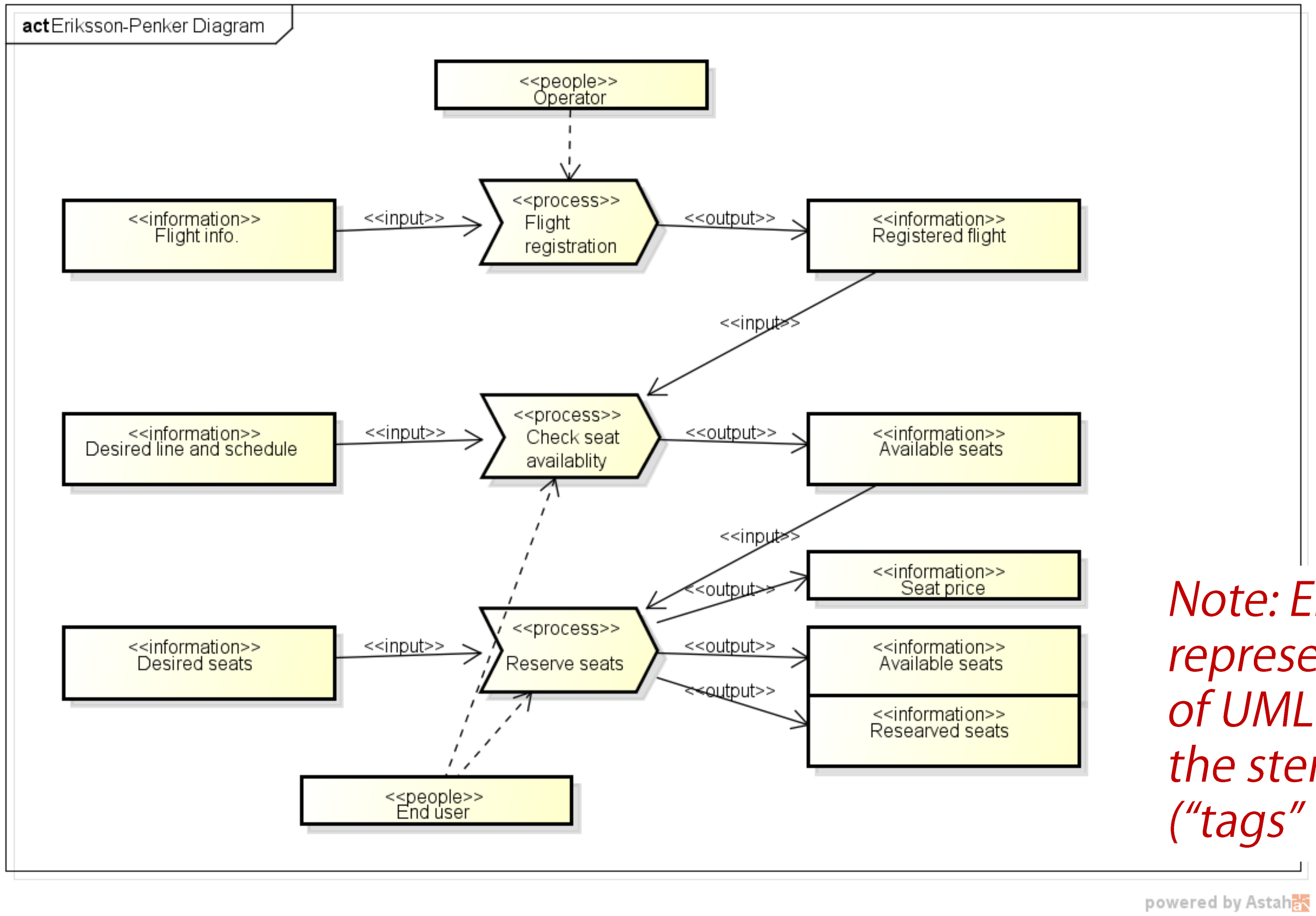
# Conceptual Modeling

- Instance of the previous flight reservation concepts in UML instance diagram


pkg
Bob : User
name = Bob
ReservationByBob : Reservation
date = 2012/04/30
Alice : Operator
name
EconomyClassOfJAL001 : EconomyClass
numberOfAvailableSeats = 69
price = 50000JPY/seat
JAL 001 : Flight
arrivalTime = 2012/06/01 1530
departureTime = 2012/06/01 0900
origin
NRTAirport : Airport
location = Narita
BusiinessClassOfJAL001 : EconomyClass
numberOfAvailableSeats = 30
price = 120000JPY/seat
destination
LAAirport : Airport
location = Los Angeles
origin
EconomyClassOfANA999 : EconomyClass
numberOfAvailableSeats = 65
price = 80000JPY/seat
ANA 999 : Flight
arrivalTime = 2012/06/01 1800
departureTime = 2012/06/01 1700
destination
SFAirport : Airport
location = San Francisco
BusiinessClassOfANA999 : BusinessClass
numberOfAvailableSeats = 13
price = 1500000JPY/seat

## Conceptual Modeling

- Process of flight reservation (Eriksson-Penker Diagram)


act Eriksson-Penker Diagram
<<people>> Operator
<<information>> Flight info.
<<input>>
<<process>> Flight registration
<<output>>
<<information>> Registered flight
<<input>>
<<information>> Desired line and schedule
<<input>>
<<process>> Check seat availablity
<<output>>
<<information>> Available seats
<<input>>
<<output>>
<<information>> Seat price
<<information>> Desired seats
<<input>>
<<process>> Reserve seats
<<output>>
<<information>> Available seats
<<output>>
<<information>> Researved seats
<<people>> End user
powered by Astah

*Note: Eriksson-Penker diagram is represented as an extended version of UML activity diagram with the stereotype mechanism ("tags" with the <<XXX>> notation)*

## TOC

## Requirements Modeling and Analysis

- Solving real-world problems by computer systems
  - Draw the To-Be on the basis of As-Is
- Use Case（ユースケース）：

defines scenarios about how user tasks are completed

(with the system to be developed)

## Requirements Modeling and Analysis

- Suppose the following results from hearing

In a very old days…

  - Cannot make reservations to visit office branches or make calls during the work hours

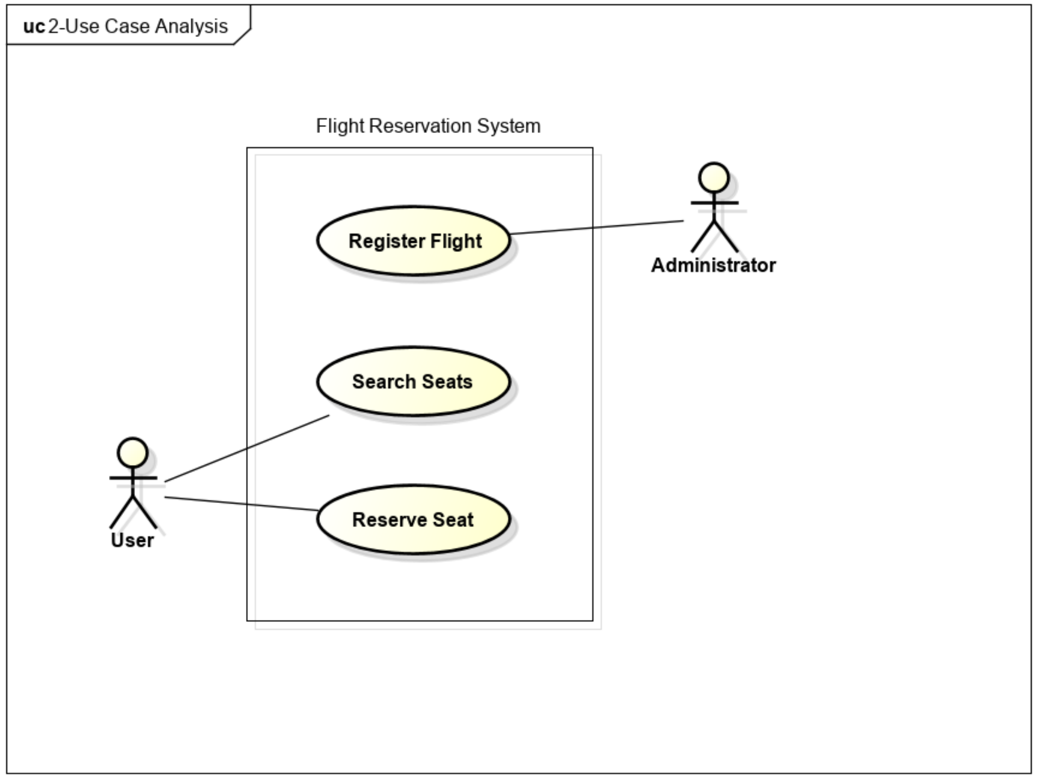
  ➡ Traveling users should be able to make reservations by themselves anytime at any place

  - Too costly to organize and analysis the present reservation status to make decisions on prices

  ➡ The status should be automatically organized so that staffs of the flight company can easily understand it

## Requirements Modeling

- Use case diagram for the flight reservation system
  - Actors
  - Use cases
  - System boundary



# Requirements Modeling

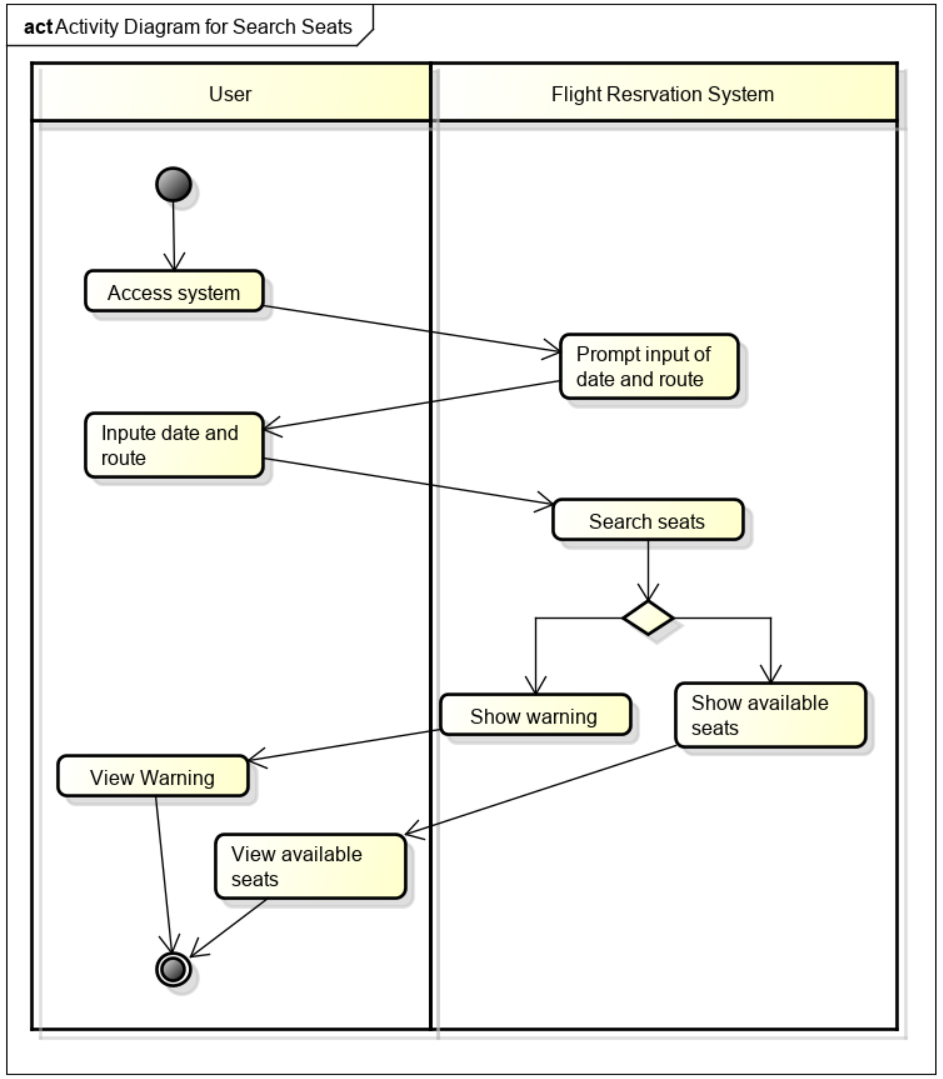
## Use case description

**Use Case**: Register Flight
**Actor**: Administrator
**Purpose**: make a new flight accessible on the system
**Precondition**: none
**Postcondition**: a flight with inputted route and schedule is registered
**Basic Sequence**:
1. The actor indicates an intention to register a new flight
2. The system prompts the actor to input the route (departure, destination) and schedule (departure, arrival), as well as price and the number of available seats for each class
3. The actor inputs the information to the system
4. The system registers the flight information
5. The system shows the registered information to the actor

**Alternative Sequence**: none
**Remarks**: none
**Scenario**: Bob with the administrator role indicates … inputs a flight information from Tokyo Narita to Paris CDG …

## Requirements Modeling

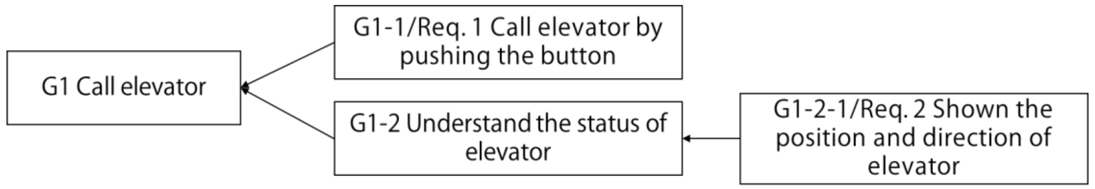
- Activity diagram:
  a sequence
  for each use case
  - Actors and the system
    (in partitions)
  - Ordering and branch



## TOC

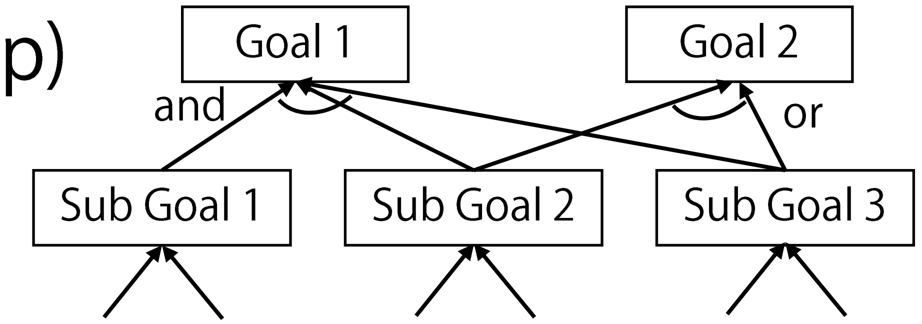
# Goal-Oriented Requirements Analysis

- Goal-Oriented Requirements Analysis
  - The previous UML models assumed use cases are obtained somehow, rather than supporting to derive them
  - The focus should be deriving use cases from high-level goals
  - Goals should be structured in a hierarchical way
    - From abstract business goals
    - Into concrete requirements to be satisfied by the computer system
    - "We want A for B" relationships


G1 Call elevator
G1-1/Req. 1 Call elevator by pushing the button
G1-2 Understand the status of elevator
G1-2-1/Req. 2 Shown the position and direction of elevator

## Goal Modeling in General

- Hierarchical structure of AND/OR decomposition
  - Generally, DAGs (directed acyclic graphs) (multiple sub-goals for one goal, no loop)



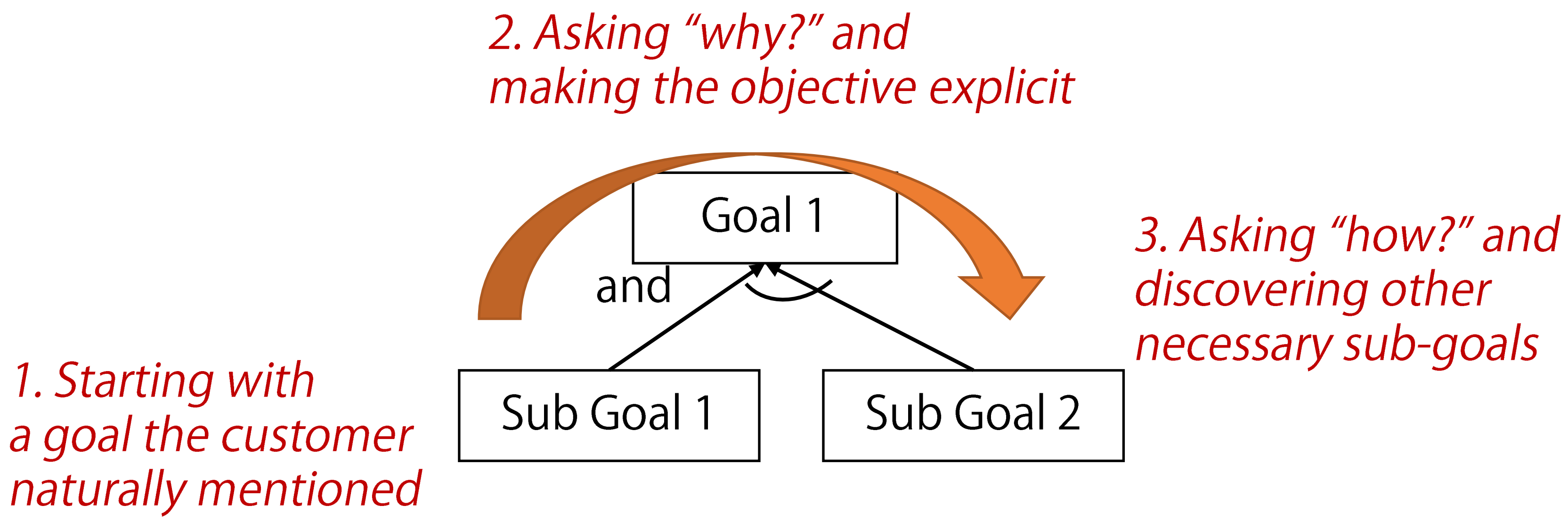
- Distinction of hard goals and soft goals
  - Whether goal satisfaction is judged in a binary way (true/false) or in "how much"
  - Non-functional aspects appear often as soft goals

## Expected Benefits of Goal Models

- Facilitates validation
  - Do we cover all what stakeholders want?
  - Don’t we include something other than what stakeholders want?
- Facilitates comparison of different means
  - By OR decomposition
- Facilitates traceability
  - e.g., understanding change impact

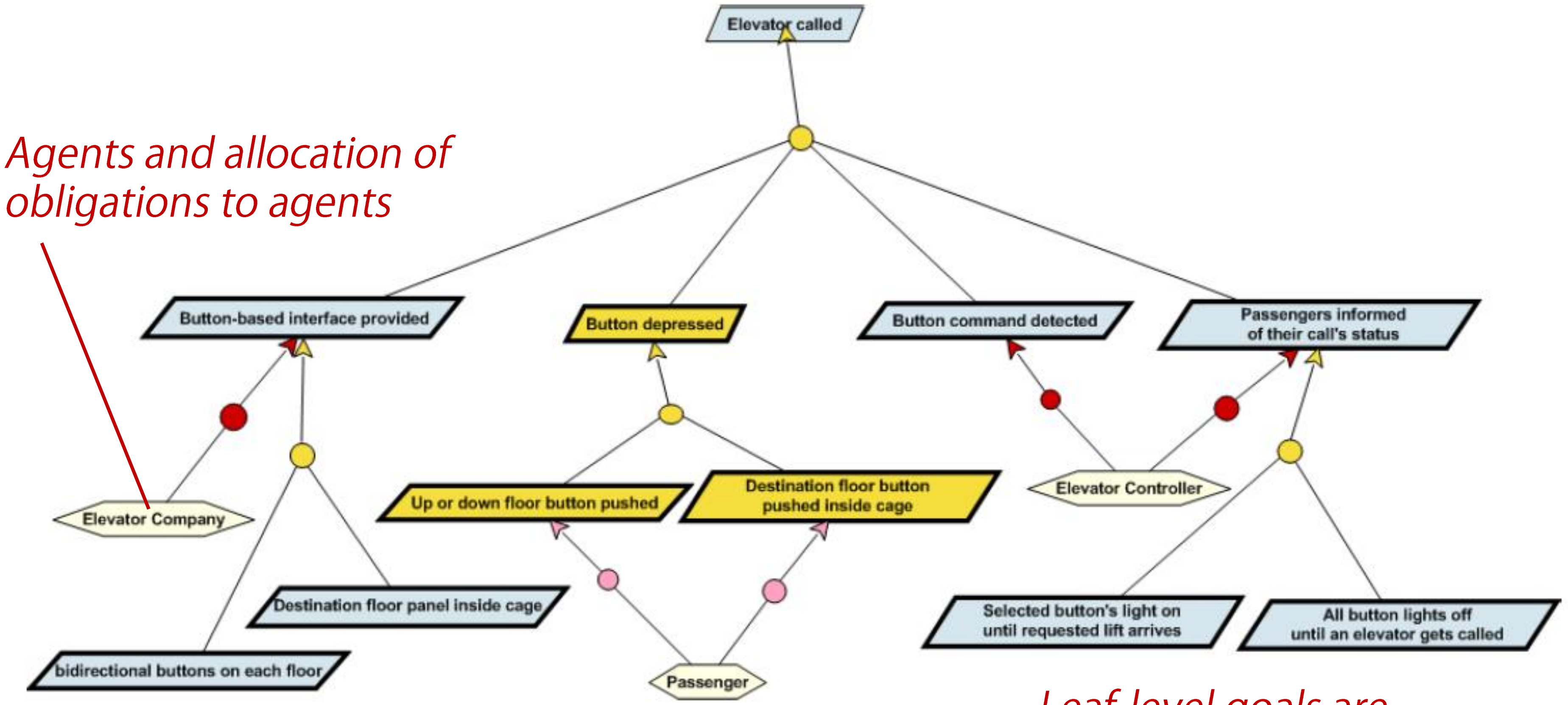
## Order of Goal Modeling

- How to build goal models
  - Top-down (from abstract to concrete): asking "how?"
  - Bottom-up (from concrete to abstract): asking "why?"
  - Typically, iteration of both directions



# Method Example: KAOS

*Agents and allocation of obligations to agents*


Elevator called
Button-based interface provided
Button depressed
Button command detected
Passengers informed of their call's status
Elevator Company
Up or down floor button pushed
Destination floor button pushed inside cage
Elevator Controller
Destination floor panel inside cage
bidirectional buttons on each floor
Passenger
Selected button's light on until requested lift arrives
All button lights off until an elevator gets called

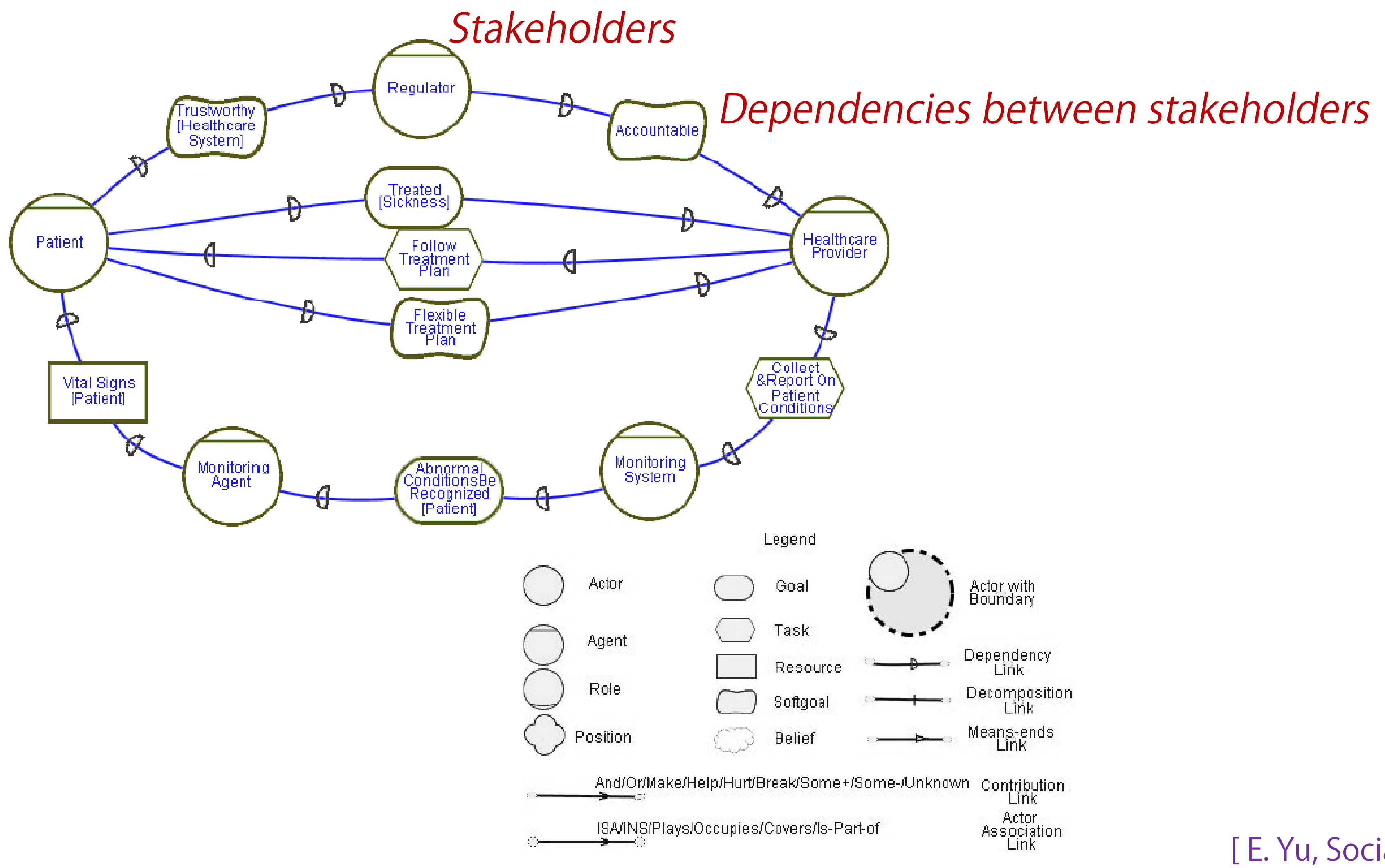
*Yellow goals allocated to users represent assumptions (what the system is not responsible for)*

*Leaf-level goals are requirements (we decided to satisfy them, and they are enough concrete to be implemented)*

[ http://www.objectiver.com/ ]

# Method Example: i*

Strategic Dependency Model for As-Is



[ E. Yu, Social Modeling and i* ]

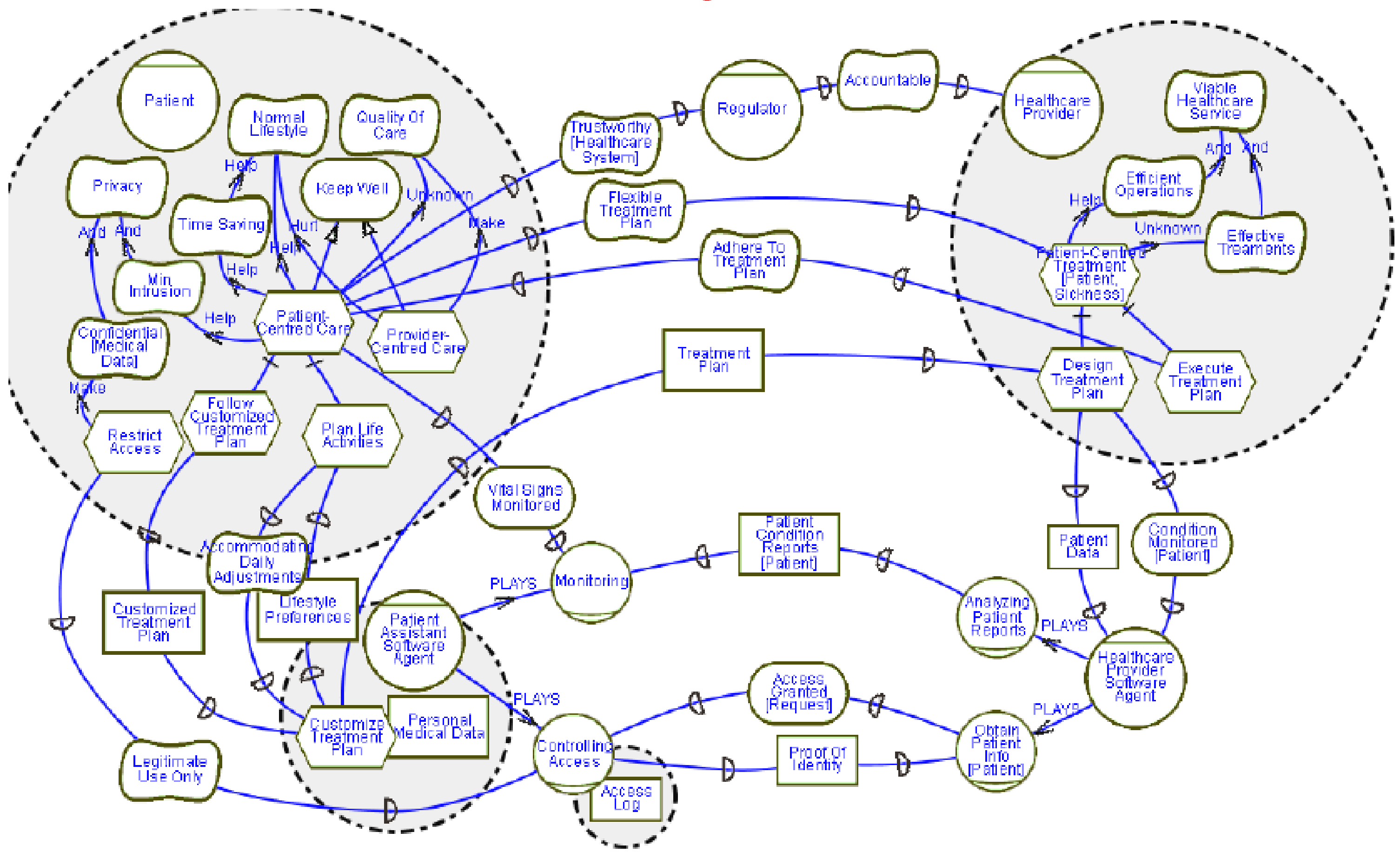
# Method Example: i*

Strategic Rationale Model for To-Be

*Goals for each stakeholder*

*Dependencies between goals of different stakeholders*

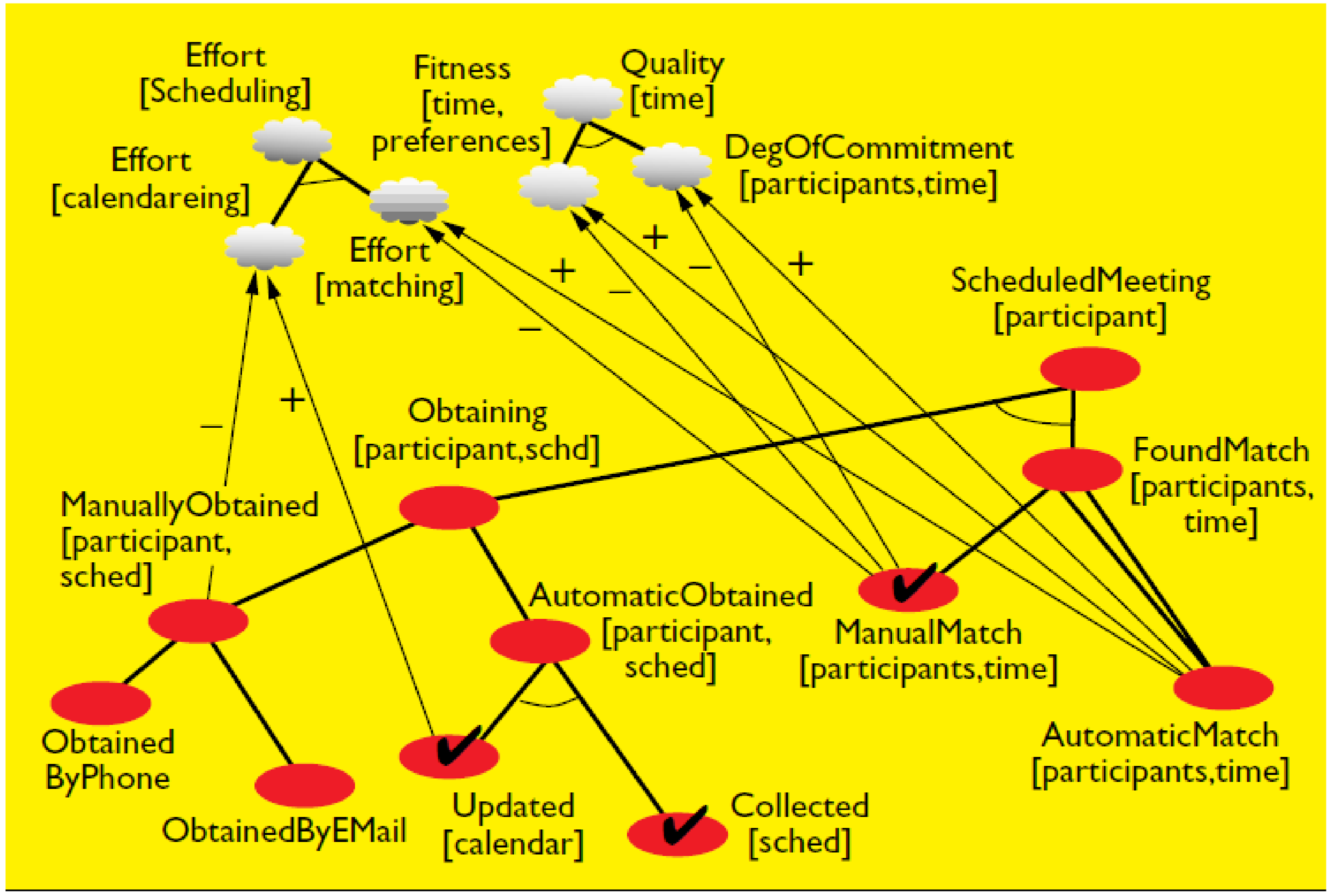
*Levels clarified for how much each sub-goal contributes to the soft super-goal (Make, Help, etc.)*



[ E. Yu, Social Modeling and i* ]

# Method Example: NFR

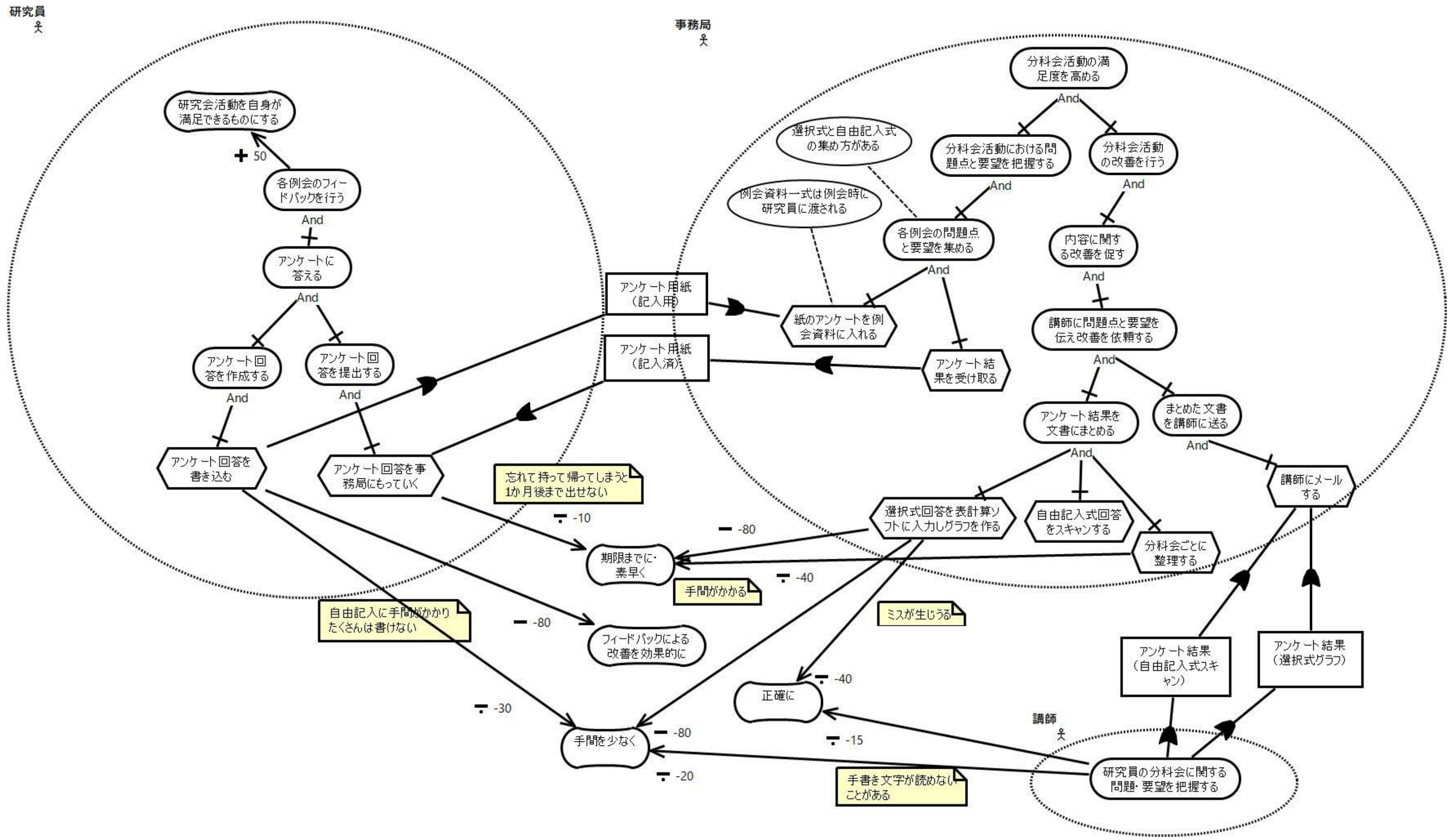
*Non-functional goals (clouds) and contributions of sub-goals (+ / -)*


Effort [Scheduling]
Fitness [time, preferences]
Quality [time]
Effort [calendareing]
DegOfCommitment [participants,time]
Effort [matching]
+
+
–
–
–
+
ScheduledMeeting [participant]
–
+
Obtaining [participant,schd]
FoundMatch [participants, time]
ManuallyObtained [participant, sched]
AutomaticObtained [participant, sched]
ManualMatch [participants,time]
AutomaticMatch [participants,time]
Obtained ByPhone
ObtainedByEMail
Updated [calendar]
Collected [sched]

*Alternative goals (or decomposition) are evaluated 2and selected*

[ J. Mylopoulos et al., From object-oriented to goal-oriented requirements analysis, CACM'99 ]

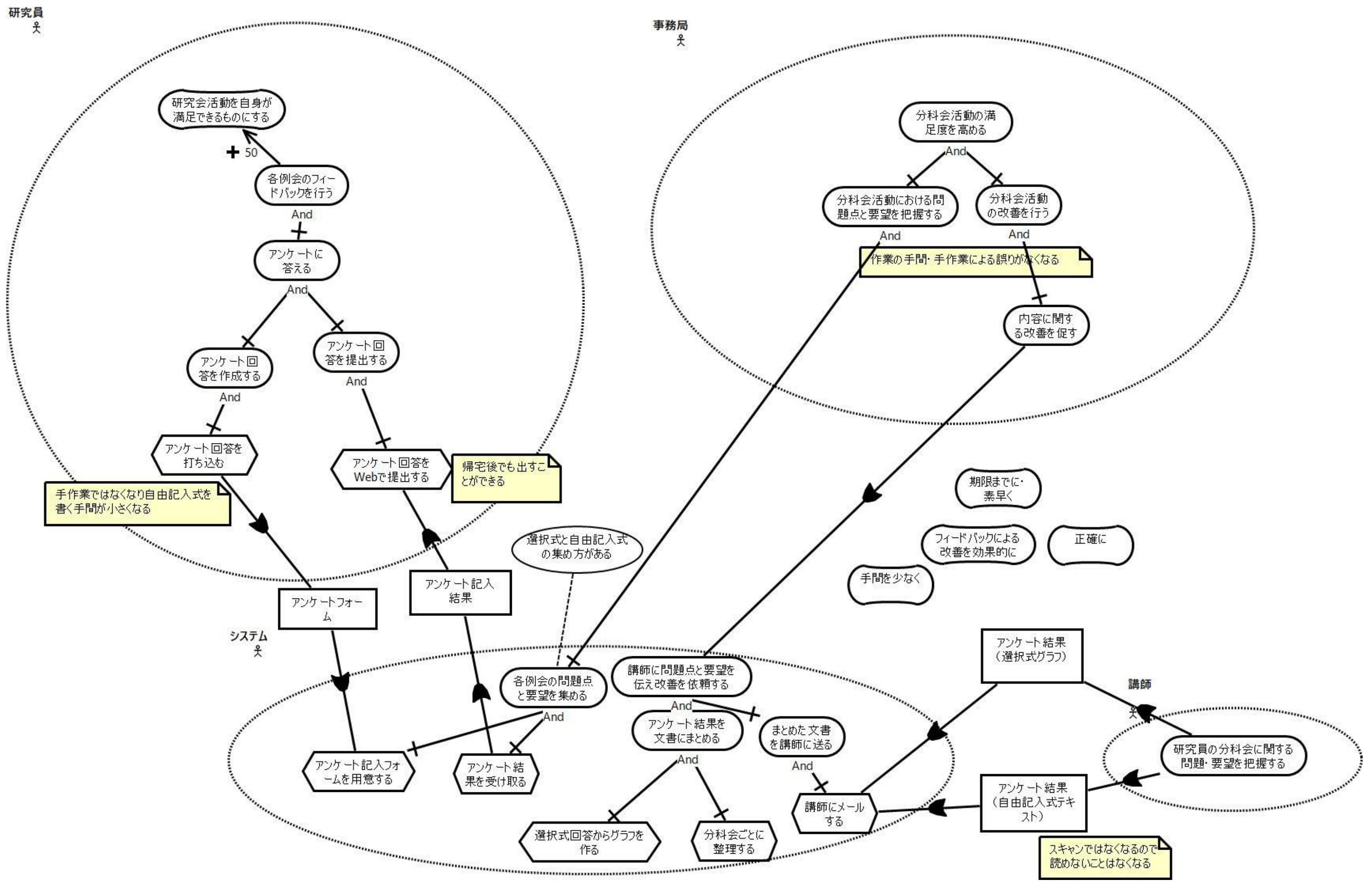
# Example (1): AsIs



研究員
事務局
講師
研究会活動を自身が満足できるものにする
50
各例会のフィードバックを行う
And
アンケートに答える
And
アンケート回答を作成する
And
アンケート回答を提出する
And
アンケート回答を書き込む
アンケート回答を事務局にもっていく
アンケート用紙（記入用）
アンケート用紙（記入済）
忘れて持って帰ってしまうと1か月後まで出せない
-10
-80
期限までに・素早く
-40
手間がかかる
自由記入に手間がかかりたくさんは書けない
-80
フィードバックによる改善を効果的に
-30
手間を少なく
-80
-20
正確に
-40
-15
ミスが生じうる
手書き文字が読めないことがある
分科会活動の満足度を高める
And
選択式と自由記入式の集め方がある
分科会活動における問題点と要望を把握する
And
分科会活動の改善を行う
And
例会資料一式は例会時に研究員に渡される
各例会の問題点と要望を集める
And
内容に関する改善を促す
And
紙のアンケートを例会資料に入れる
アンケート結果を受け取る
講師に問題点と要望を伝え改善を依頼する
And
アンケート結果を文書にまとめる
And
まとめた文書を講師に送る
And
講師にメールする
選択式回答を表計算ソフトに入力しグラフを作る
自由記入式回答をスキャンする
分科会ごとに整理する
アンケート結果（自由記入式スキャン）
アンケート結果（選択式グラフ）
研究員の分科会に関する問題・要望を把握する

*User Requirements Notation (URN) (standardized by combining i* and NFR)*

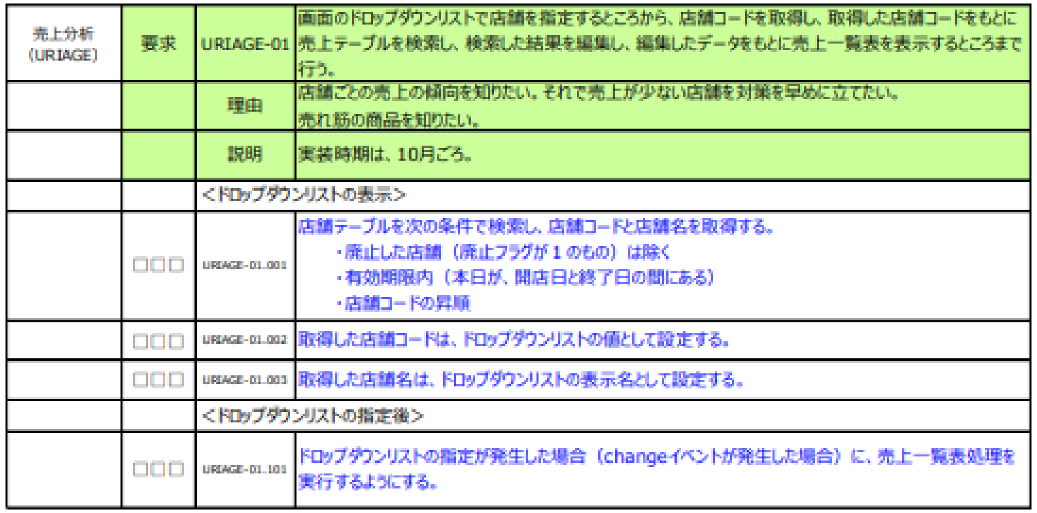
# Example (2): ToBe


研究員
研究会活動を自身が満足できるものにする
+ 50
各例会のフィードバックを行う
And
アンケートに答える
And
アンケート回答を作成する
And
アンケート回答を提出する
And
アンケート回答を打ち込む
手作業ではなくなり自由記入式を書く手間が小さくなる
アンケート回答をWebで提出する
帰宅後でも出すことができる
アンケートフォーム
アンケート記入結果
選択式と自由記入式の集め方がある
事務局
分科会活動の満足度を高める
And
分科会活動における問題点と要望を把握する
And
分科会活動の改善を行う
And
作業の手間・手作業による誤りがなくなる
内容に関する改善を促す
期限までに・素早く
フィードバックによる改善を効果的に
正確に
手間を少なく
システム
各例会の問題点と要望を集める
And
講師に問題点と要望を伝え改善を依頼する
And
アンケート結果を文書にまとめる
And
まとめた文書を講師に送る
And
アンケート記入フォームを用意する
アンケート結果を受け取る
選択式回答からグラフを作る
分科会ごとに整理する
講師にメールする
アンケート結果（選択式グラフ）
講師
研究員の分科会に関する問題・要望を把握する
アンケート結果（自由記入式テキスト）
スキャンではなくなるので読めないことはなくなる

*User Requirements Notation (URN) (standardized by combining  i* and NFR)*

## Actual Usages of Goal Models

- The examples were “initial ones that made the impact”
  - Few people may use them directly nowadays (?)
- The “we want A for B” relationship is very, very essential
  - e.g., it is very common to use spreadsheets to model the few-level goal structures



| | | | |
|---|---|---|---|
| 売上分析 (URIAGE) | 要求 | URIAGE-01 | 画面のドロップダウンリストで店舗を指定するところから、店舗コードを取得し、取得した店舗コードをもとに売上テーブルを検索し、検索した結果を編集し、編集したデータをもとに売上一覧表を表示するところまで行う。 |
| | | 理由 | 店舗ごとの売上の傾向を知りたい。それで売上が少ない店舗を対策を早めに立てたい。<br>売れ筋の商品を知りたい。 |
| | | 説明 | 実装時期は、10月ごろ。 |
| | | <ドロップダウンリストの表示> | |
| | □□□ | URIAGE-01.001 | 店舗テーブルを次の条件で検索し、店舗コードと店舗名を取得する。<br>・廃止した店舗（廃止フラグが１のもの）は除く<br>・有効期限内（本日が、開店日と終了日の間にある）<br>・店舗コードの昇順 |
| | □□□ | URIAGE-01.002 | 取得した店舗コードは、ドロップダウンリストの値として設定する。 |
| | □□□ | URIAGE-01.003 | 取得した店舗名は、ドロップダウンリストの表示名として設定する。 |
| | | <ドロップダウンリストの指定後> | |
| | □□□ | URIAGE-01.101 | ドロップダウンリストの指定が発生した場合（changeイベントが発生した場合）に、売上一覧表処理を実行するようにする。 |

USDM format, cited from [ https://affordd.jp/previous/tech_documents/affordd-t2-usdmtext-basic_1.3.pdf ]

## Other Topics for Requirements Engineering

- Business analysis
  - SWOT, Business Model Campus, …
- More upper-level: creating and organizing ideas
  - 「超上流要求工学」
  - Brain storming, K-J method, …
- Methods for elicitation and communication
  - Interview, ethnography, persona analysis, …
- Methods for validation
  - Prototyping, …

## Note for Following Topic: Uncertainty of Requirements

- It is very difficult to "completely" enumerate what should be done in the target business or human activities
  - Can you be confident for the 800-page specification document?
  - Are they still valid after 1 year of development?
  - (but this may be inevitable in some large systems)

➡ One possibility is to work on a smaller set of requirements, for 2-3 weeks, and have iteration
(to be discussed in agile software development)

## Summary

- Domain Analysis / Requirements Analysis
  - The most significant activities that affect the success/failure of the project
  - Understanding of the problem domain, and draw ToBe from AsIs
  - Systematic methods to tackle the very difficult activities involving various stakeholders, without no “right answer”