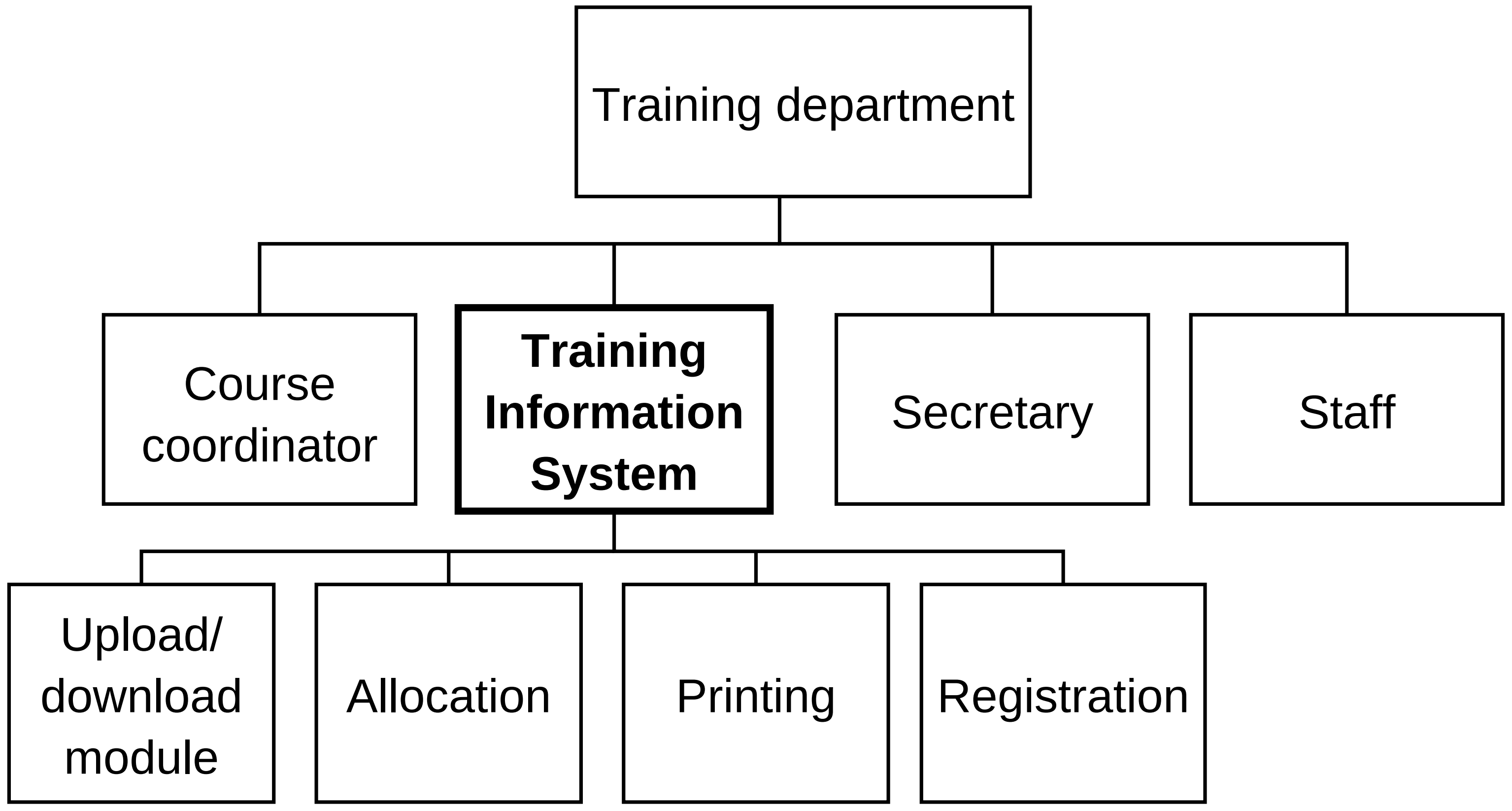
Chapter 4. Software Specifications

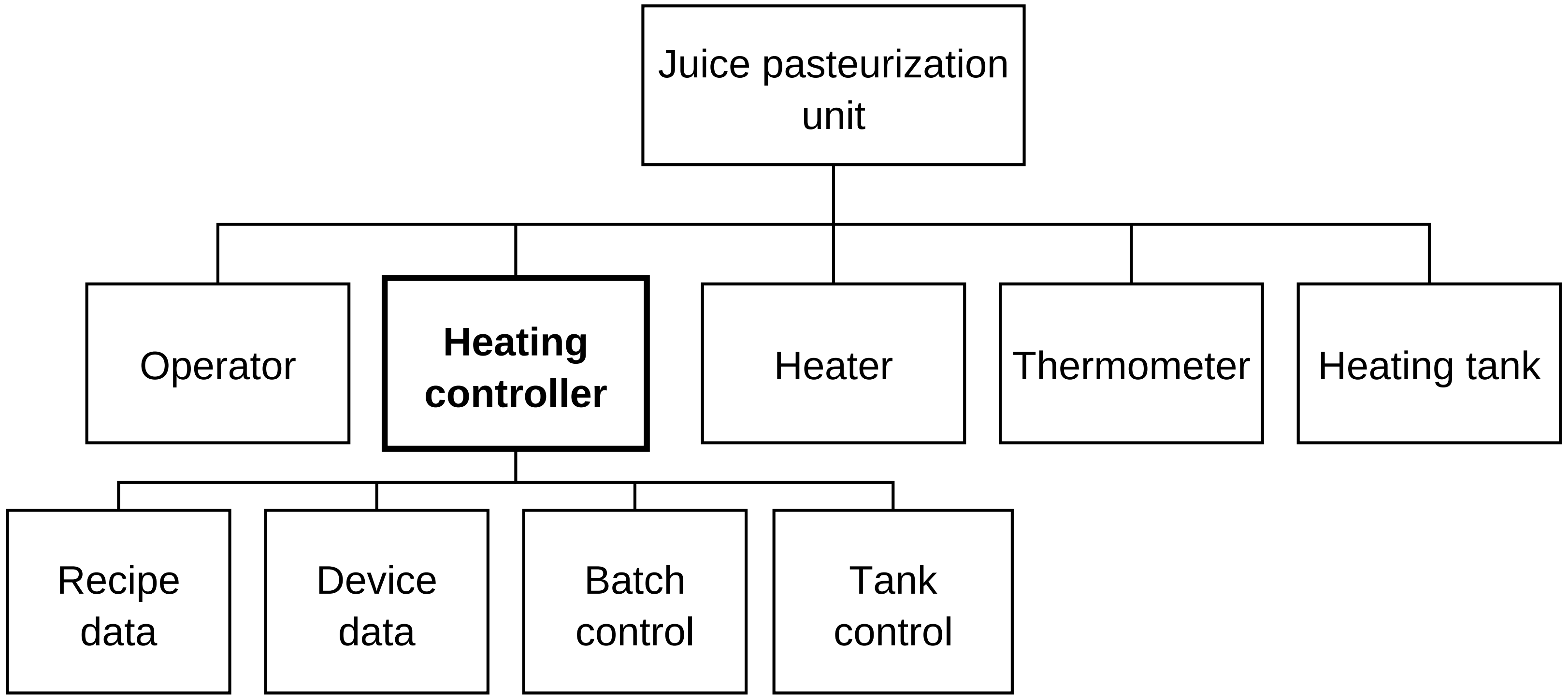
- We call any creative decision about a system a design decision.
- To design a system is to make a plan how it will be built.
- A specification is a description of design decisions.
- A reactive system specification must describe
  - the place of the SuD in the system hierarchy,
  - it must describe functions, behavior and communication of the SuD, and
  - it must describe its composition.
- The specification must be used to motivate the design in a systems engineering argument.

Systems engineering argument example 1



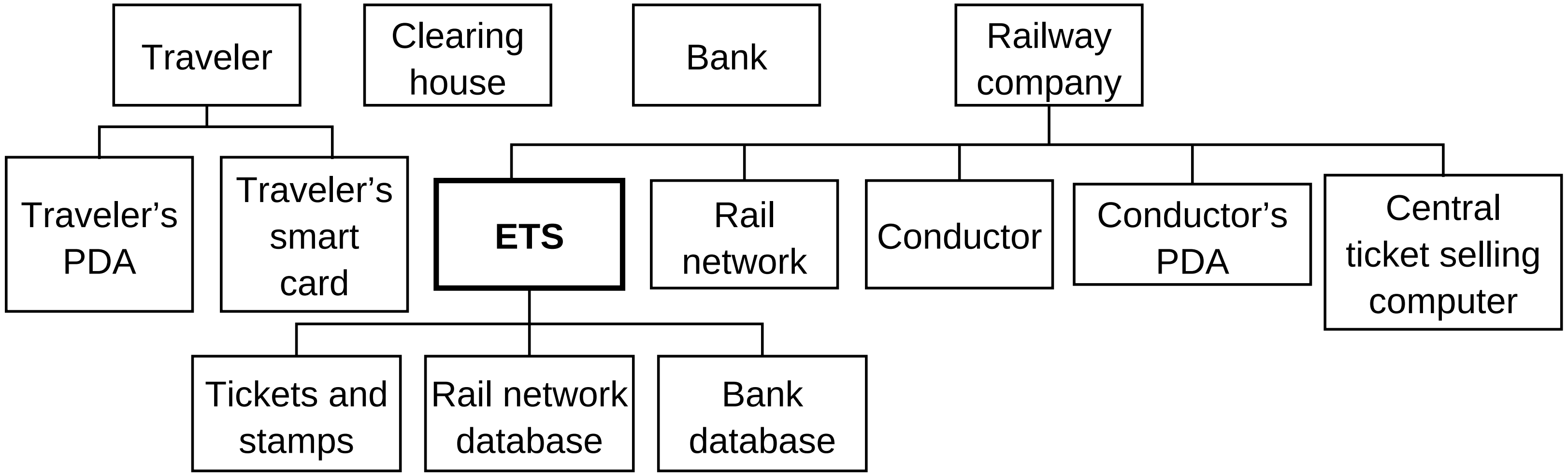
- If TIS allows registration of unexpected participants
- and the department keeps extra staff,
- then department is able to handle newcomers efficiently.

Systems engineering argument example 2



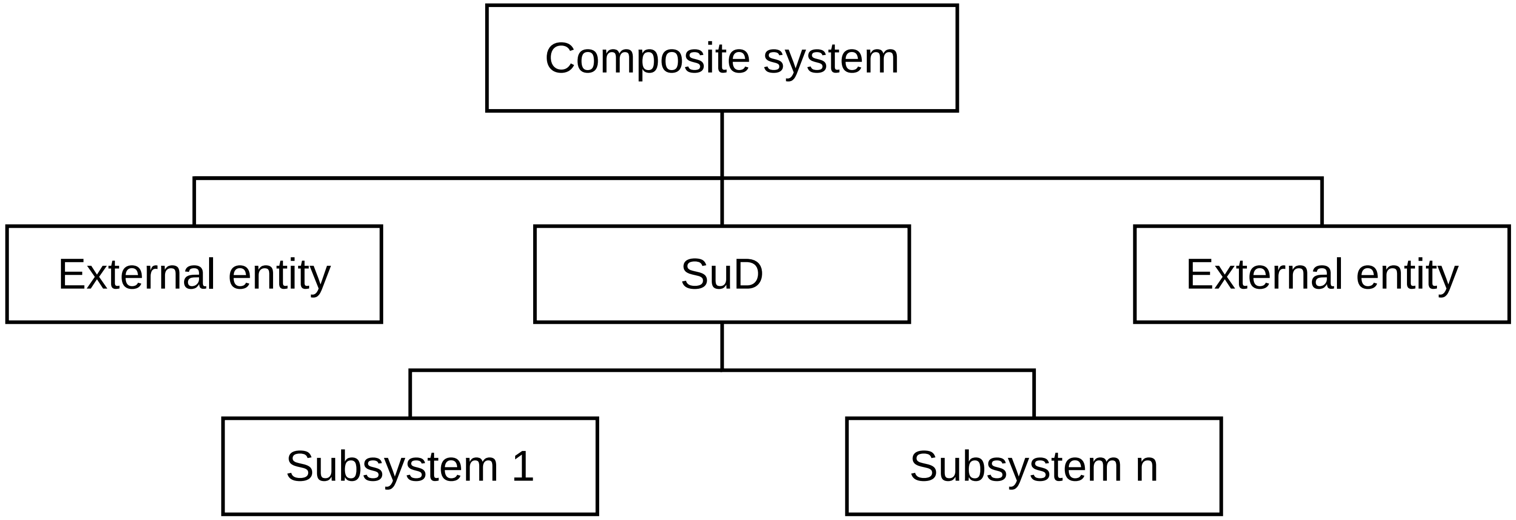
- If controller controls heater according to batch recipe
- and thermometer is functioning,
- then pasteurization unit heats batch according to recipe.

Systems engineering argument example 3



- If ETS allows travelers to buy tickets through their PDA
- and traveler's PDA interfaces with ETS,
- then railway company reduces operating costs.

System engineering argument



- If SuD satisfies specification S
- and environment satisfies assumptions A
- then composite system has emergent properties E.

Emergent properties arise by interaction of component systems.
They should satisfy *goals* of composite system.

The role of assumptions

- If assumptions are not satisfied by environment, the composite system goal may not be reached.
- System cannot guarantee the assumptions.

Examples:

- Heat will rise when “switch on” sent to heater.

  Assume laws of thermodynamics and assume that devices work.

- Ticket is stamped for segment of the current route.

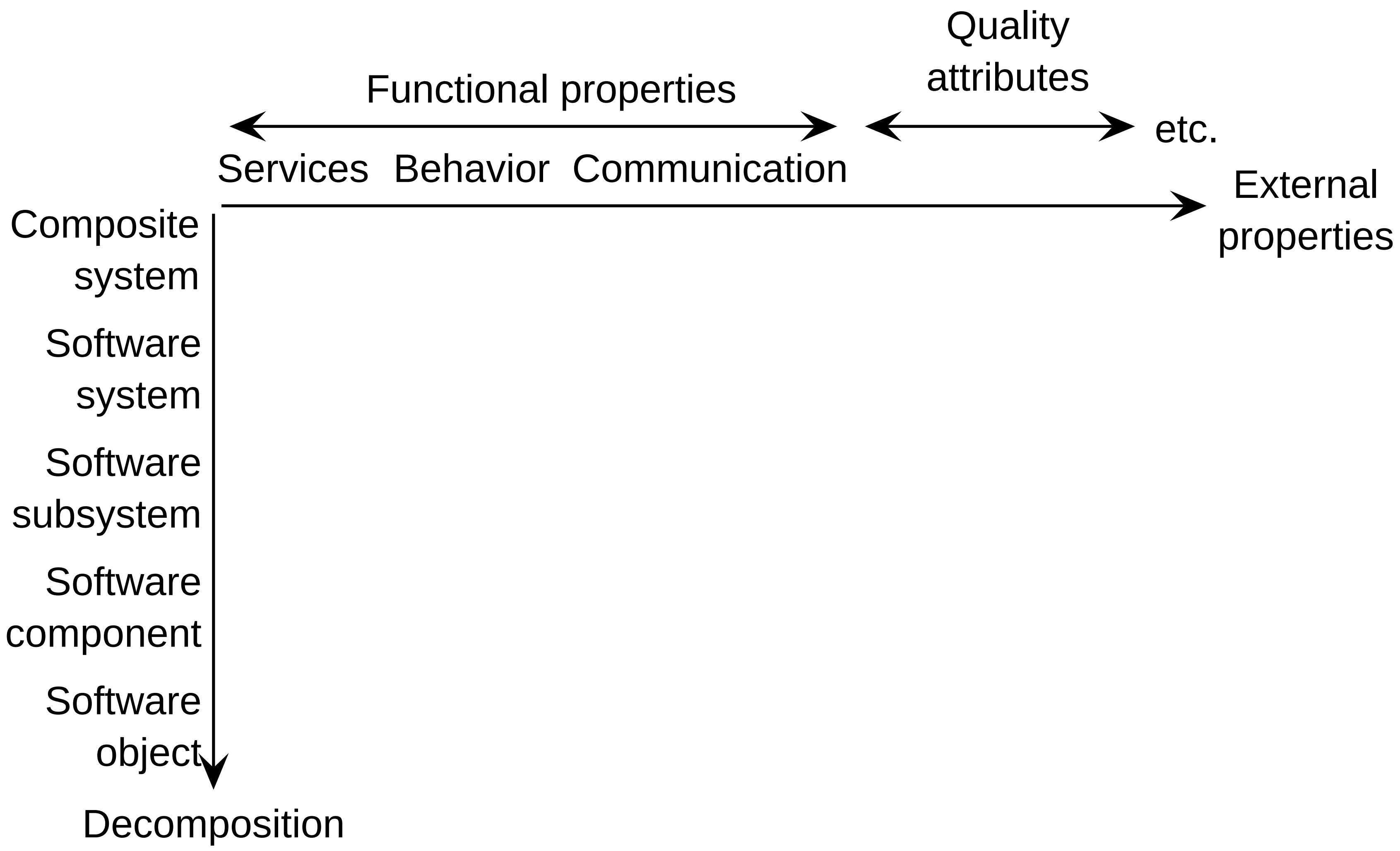
  Assume that the conductor, traveler and smart card are physically located in the segment for which ticket is stamped.

Kinds of assumptions

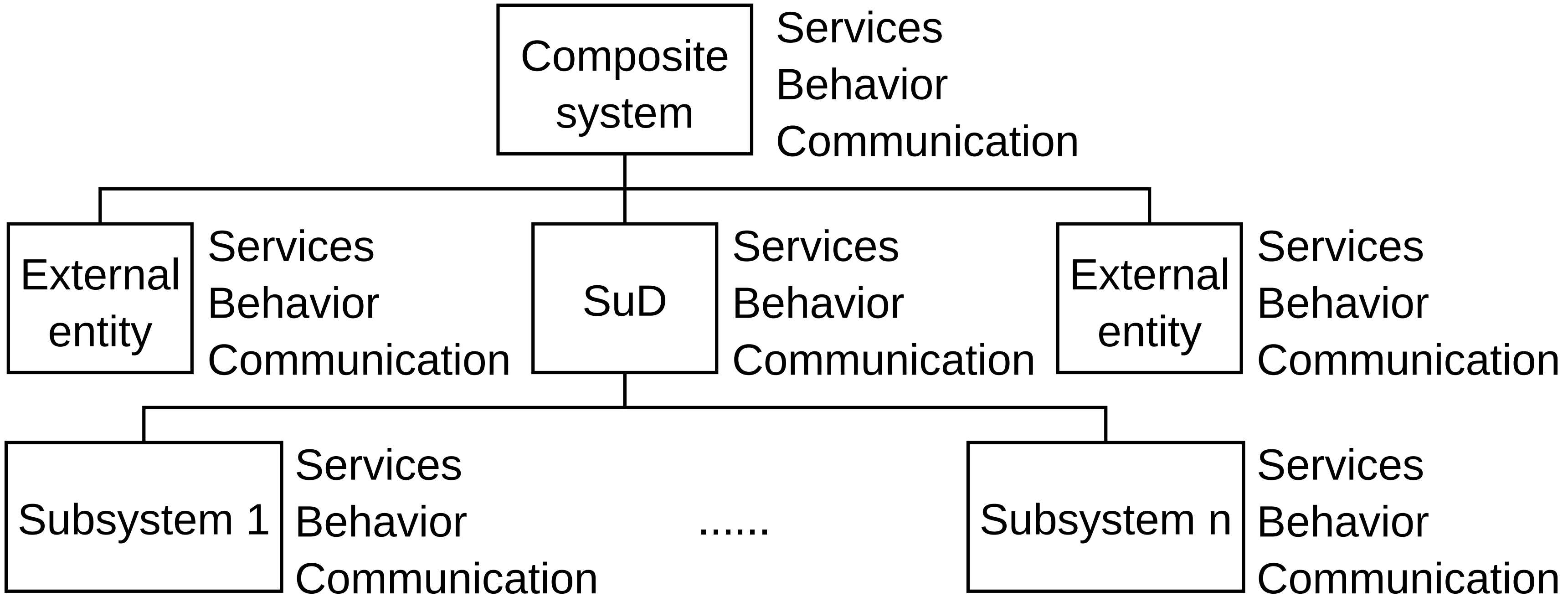
- laws of nature
- specifications of devices
- rules for people (laws, procedures)
- definitions of conceptual structures (e.g. meaning of stamps and tickets)

Only laws of nature are infallible ... we assume. Many assumptions are about the subject domain and about the connection domain.

Kinds of properties



Properties occur at every level



Functional properties appear at every level.

- Service = Interaction that delivers desired effect.
- Behavior = Ordering of interactions over time.
- Communication = Symbol flow between different entities.

Terminology

With respect to the SuD, we talk about:

- Requirement = desired property.
- Constraint = imposed property.
- Aspect = group of properties.

Operational specification

Specification of set of reproducible operations to find out whether a property is present.

Important class of operational specs has the form

- If stimulus s occurs
- and system is in state C
- then produce response r.

Also called Event-Condition-Action (ECA) rule. Appears in state transition table (see chapter 12).

Main points

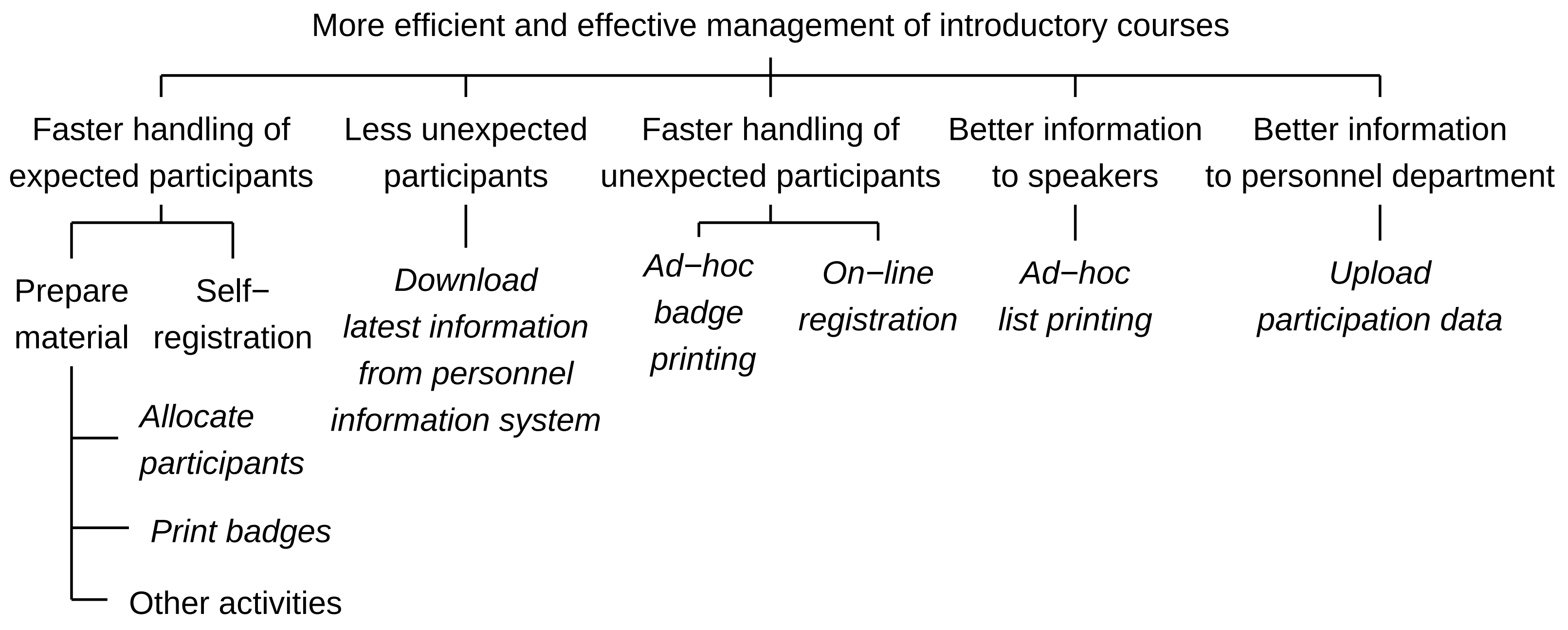
- Systems have functional properties and quality properties.
- Functional properties are services, behavior, communication.
- These reappear at every level in aggregation hierarchy.
- (There are really three aggregation hierarchies.)
- SuD needs external entities to jointly produce desired emergent properties of composite system. System engineering argument.

Chapters 5–7. Mission Statement, Function Refinement Tree, Service Description

- We describe the utility of the system for its environment by describing its functions for the environment.
- But a function is not a component; it is not a program; it is not a part of a program. It is the added value, or utility, of the SuD for its environment.
  - “The function of this coffee machine is to brew coffee.”
  - “The added value of this coffee machine is to brew coffee.”
  - “The utility of this coffee machine is to brew coffee.”

  In this course, these three sentences mean the same thing.

Example 1: Goal tree of the training department

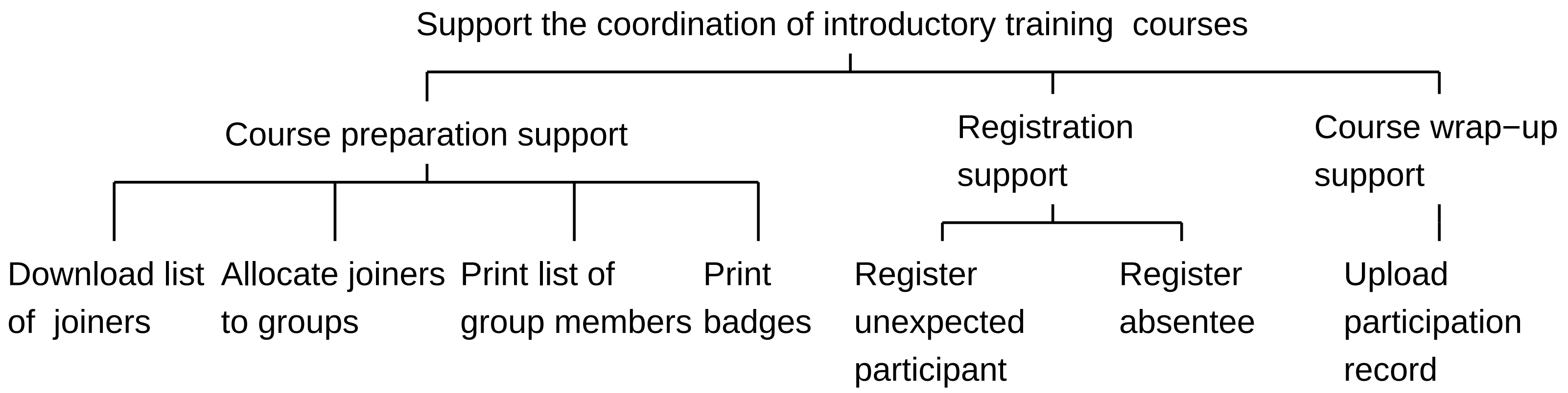

More efficient and effective management of introductory courses
Faster handling of expected participants
Less unexpected participants
Faster handling of unexpected participants
Better information to speakers
Better information to personnel department
Prepare material
Self−registration
Download latest information from personnel information system
Ad−hoc badge printing
On−line registration
Ad−hoc list printing
Upload participation data
Allocate participants
Print badges
Other activities

TIS should contribute to goals in italics.

Example 1: Mission of TIS

- **Name:** Training Information System
- **Acronym:** TIS
- **Purpose:** To support the management of monthly introductory training courses.
- **Responsibilities:**
  - To support course preparation, including allocation of participants to groups and printing badges
  - To support course handling (unexpected, absentees)
  - To support course wrap-up
- **Exclusions:**
  - Data of unexpected participants is not checked.
  - No support for allocation of speakers to groups.
  - No support for allocation of groups to rooms.

Example 1: Function refinement tree of TIS


Support the coordination of introductory training courses
Course preparation support
Registration support
Course wrap−up support
Download list of joiners
Allocate joiners to groups
Print list of group members
Print badges
Register unexpected participant
Register absentee
Upload participation record

Example 1: Description of a TIS service

Download joiners

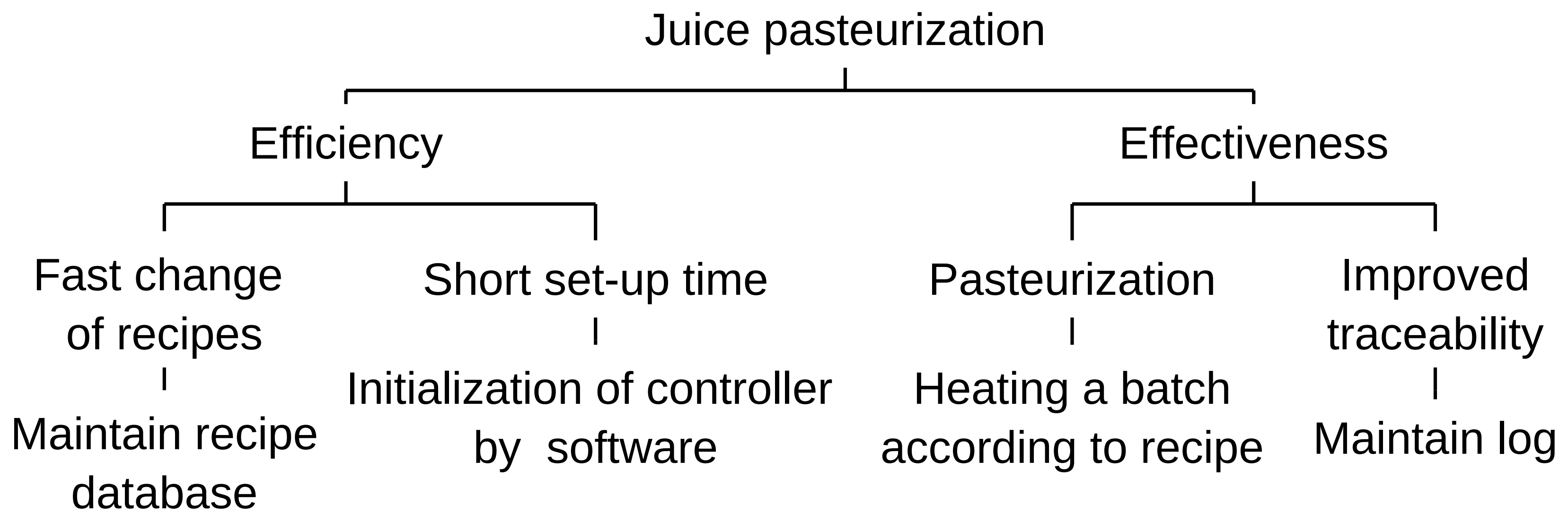
- **Triggering event:** Coordinator requests to download list of joiners from the personnel information system.
- **Delivered service:** Download the list of people from the Personnel Information System who have joined the company since the previous training.
- **Assumptions:** The data in the Personnel Information System reflects the situation accurately with a time lag of not more than one working day.

Example 1: Description of another TIS service

Upload participant record

- **Triggering event:** Coordinator requests to upload list of joiners to personnel information system.
- **Delivered service:** Upload the list of people who participated in the training to the Personnel Information System.
- **Assumptions:** The Personnel Information Systems is able to deal with data about unexpected participants, including any remaining errors in that data.

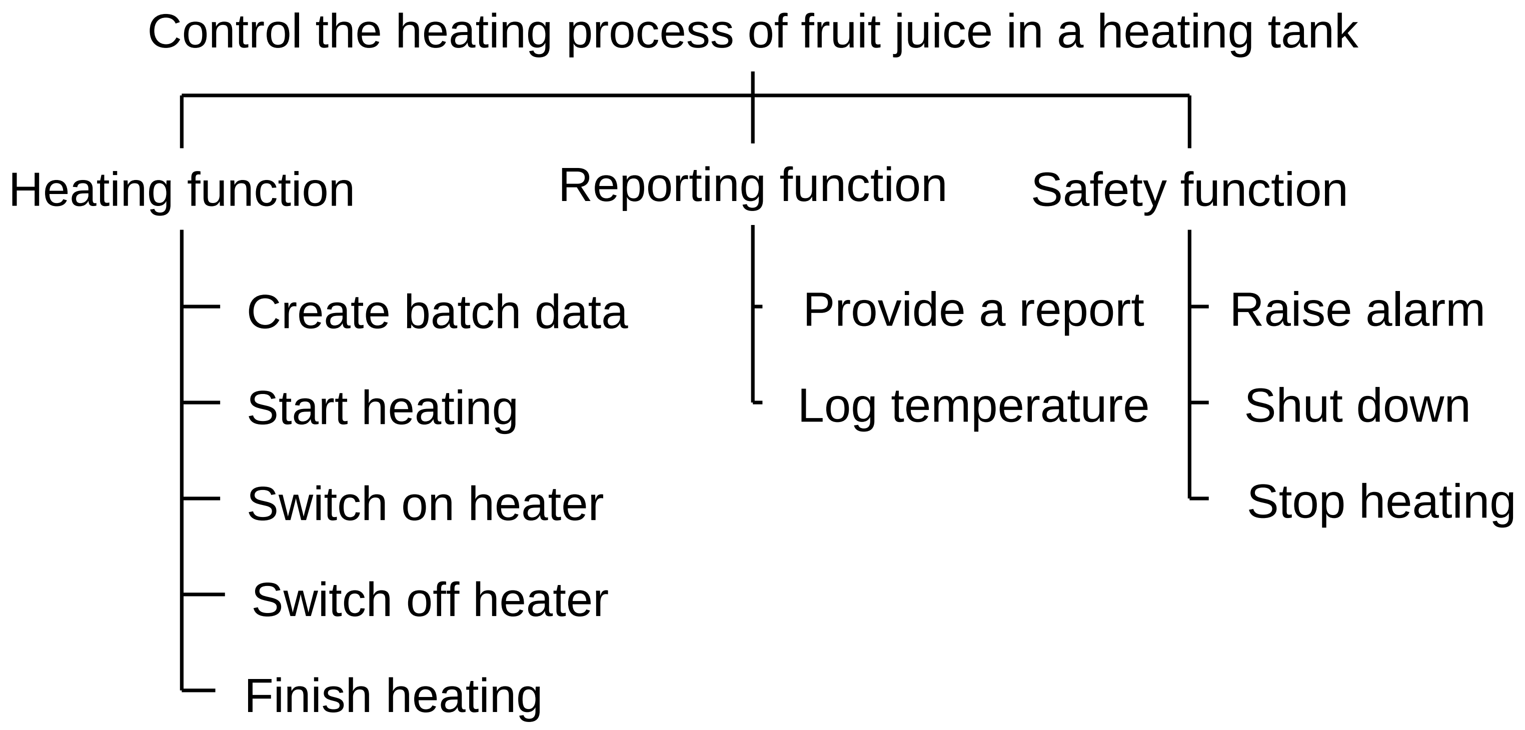
Example 2: Goal tree of juice pasteurization plant



Example 2: Mission of heating controller

- **Name:** Juice heating controller.
- **Purpose:** To control the heating process of fruit juices in a heating tank.
- **Responsibilities:**
  - To initialize itself with batch data and heat the batch according to recipe.
  - To report on the heating process.
  - To maintain safe conditions in a tank.
- **Exclusions:**
  - Filling the storage tanks with juice.
  - Transferring pasteurized juice to the canning line.

Example 2: Function refinement of heating controller



Example 2: A service description

- **Name:** P1. Heat batch according to recipe.
- **Triggering event:** Operator command “start heating batch b according to recipe”.
- **Delivered service:** Upon reception of this command, the controller ensures that a heating process takes place in the heating tanks in which b is stored, according to the recipe of b.
- **Assumptions:** There is a batch in the heating tank.

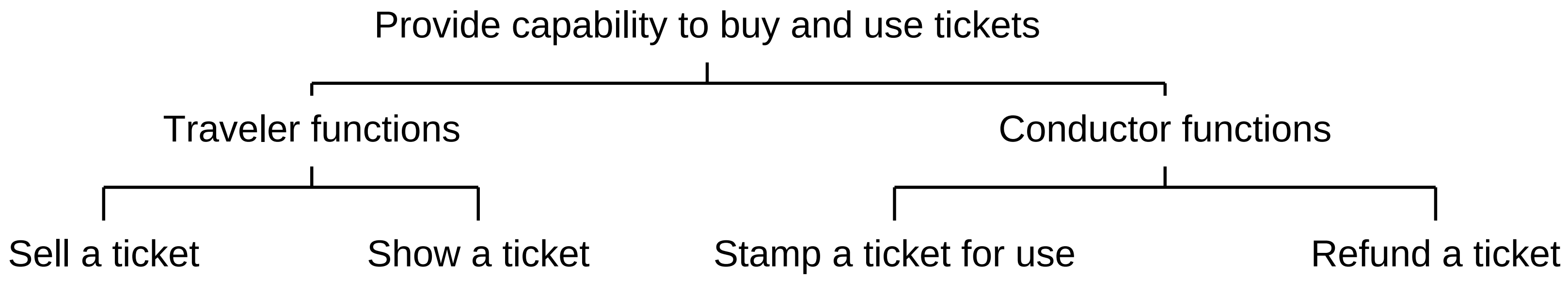
Example 2: Another service description

- **Name:** Log temperature.
- **Triggering event:** When an execution of P1 starts, and then every 10 seconds during this execution of P1.
- **Delivered service:** The controller records the measured temperature in each tank in which b is stored.

Example 3: Mission statement of ETS

- **Name of the system:** Electronic Ticket System (ETS).
- **Purpose:** Provide capability to buy and use tickets of a railway company using a PDA and a smart card.
- **Composition:** Software distributed over smart cart, PDA's, central computer.
- **Responsibilities of the system:**
  - To support ticket buying
  - To support ticket usage
  - To support ticket refunding
- **Exclusions:**
  - The system does not perform travel planning
  - Only tickets for one person and one trip (single or return).

Example 3: Function refinement of ETS

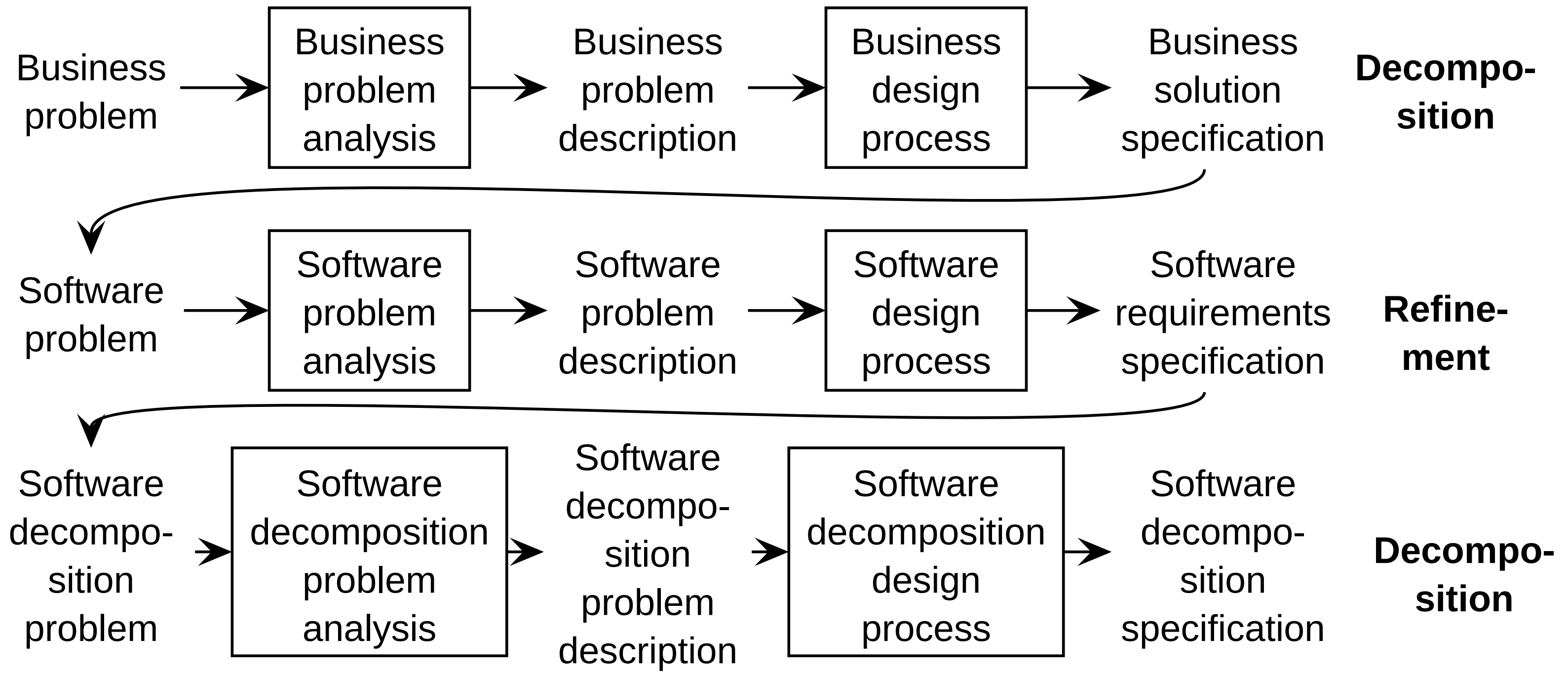


Two service descriptions of ETS

- **Name:** Sell a ticket.
- **Triggering event:** Traveler requests to buy a ticket.
- **Delivered service:** Allow a traveler to buy a ticket at any time and place chosen by the traveler.

- **Name:** Show a ticket.
- **Triggering event:** Traveler requests to view a ticket.
- **Delivered service:** Display ticket attributes to the user.

Levels of design


Business problem
Business problem analysis
Business problem description
Business design process
Business solution specification
Decompo-sition
Software problem
Software problem analysis
Software problem description
Software design process
Software requirements specification
Refine-ment
Software decompo-sition problem
Software decomposition problem analysis
Software decompo-sition problem description
Software decomposition design process
Software decompo-sition specification
Decompo-sition

- A business solutution specification describe a decomposition of the business that solves a business problem.
- A software specification refines the software part of a business solution.

Goal analysis

- First separate the design levels as on previous slides.
- Next identify business goals.
  - In a goal tree, achievement of children goals is sufficient to achieve parent goal.
  - Leaves of a business goal tree usually are desirable business activities.
- From the goals of the business solution, derive from the software goals.
- This gives us statement of purpose and major responsibilities of the software.

Mission statement

Highest level software specification. Talks about software solutions instead of business solutions.

- $\surd$ To find software mission, analyze business goals.
- $\surd$ To find mission, find desired emergent properties E of composite system.
- $\surd$ Justify mission and responsibilities by system engineering argument: Mission statement + environment assumptions entail business goals.

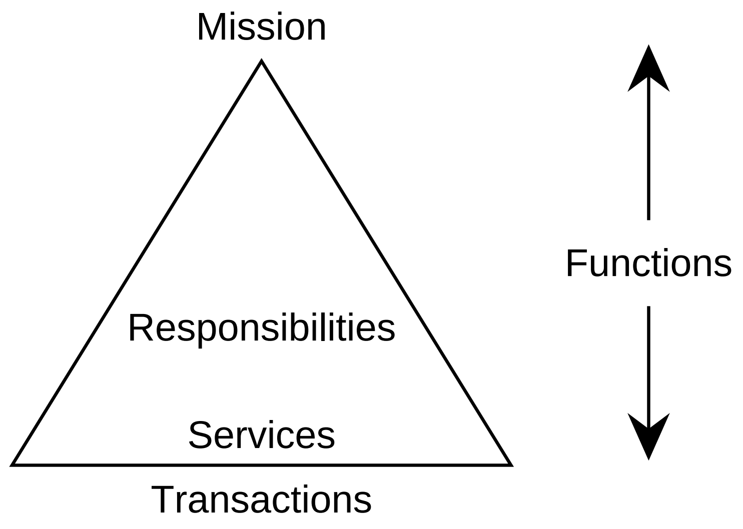
Mission statement is updated as our understanding improves during the project.

Function refinement tree.

Makes responsibilities more specific.

- Not a system structure: Just an indented shopping list.
- Organization of tree is subjective: Determined by discussion with customer.
- The tree bounds functionality of the system.
- It is the most implementation-independent description of the system.
- It justifies the presence of services.
- It prevents the presence of unnecessary services.

Terminology



Function = ability to create desired effect in the environment.

- **Responsibility** = Contribution to environment goal.
- **Service** = Useful interaction triggered by event.
- **Transaction** = Atomic interaction.

Service descriptions

- Each service is identified by a
  (1) triggering event and a
  (2) delivered value (benefit).
- Each service may make assumptions about the environment.
- Do not give details about system behavior nor about communication channels with the system.
- Just describe the valuable effect that the system should have on the environment.
- A service may have a simple or a complex behavior; describe this later, using techniques from chapters 11 and 12.

Service descriptions are not system components

- For programmers, it is hard to see a piece of text *not* as a software component.
- A service description is not a software component;
- it is a description of something useful done by the software.

Main points

- Mission relates system functions to business goals.
- Function refinement tree relates services to mission.
- Services have a discrete beginning and deliver a value.
- Services may be non-atomic, consisting of many transactions.