

MACIASZEK, L.A. (2005):
Requirements Analysis and System Design, 2nd ed.
Addison Wesley, Harlow England, 504p.
ISBN 0 321 20464 6

Chapter 2
Requirements Determination

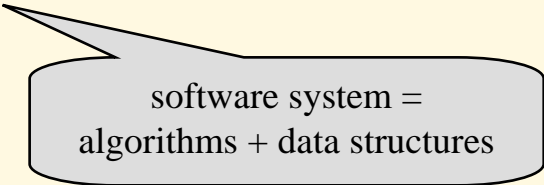
© Pearson Education Limited 2005

Topics

- *Functional and nonfunctional requirements*
- *Requirements elicitation*
 - *traditional methods and modern methods*
- *Requirements negotiation and validation*
- *Requirements management*
- *Requirements business model*
 - *system scope, business use case model, business glossary, business class model*
- *Requirements document*

Principles of requirements determination

- *Requirements determination:*
 - *The least technical phase of system development*
 - *Demands social, communication and managerial skills*
 - *Delivers a (mostly narrative) definition of user requirements*
- *Requirements define*
 - *System services → functional requirements*
 - *scope of the system*
 - *necessary business functions*
 - *required data structures*
 - *System constraints → nonfunctional requirements*
 - *regarding 'look and feel', performance, security, etc.*
 - *also known as supplementary requirements*



software system =
algorithms + data structures

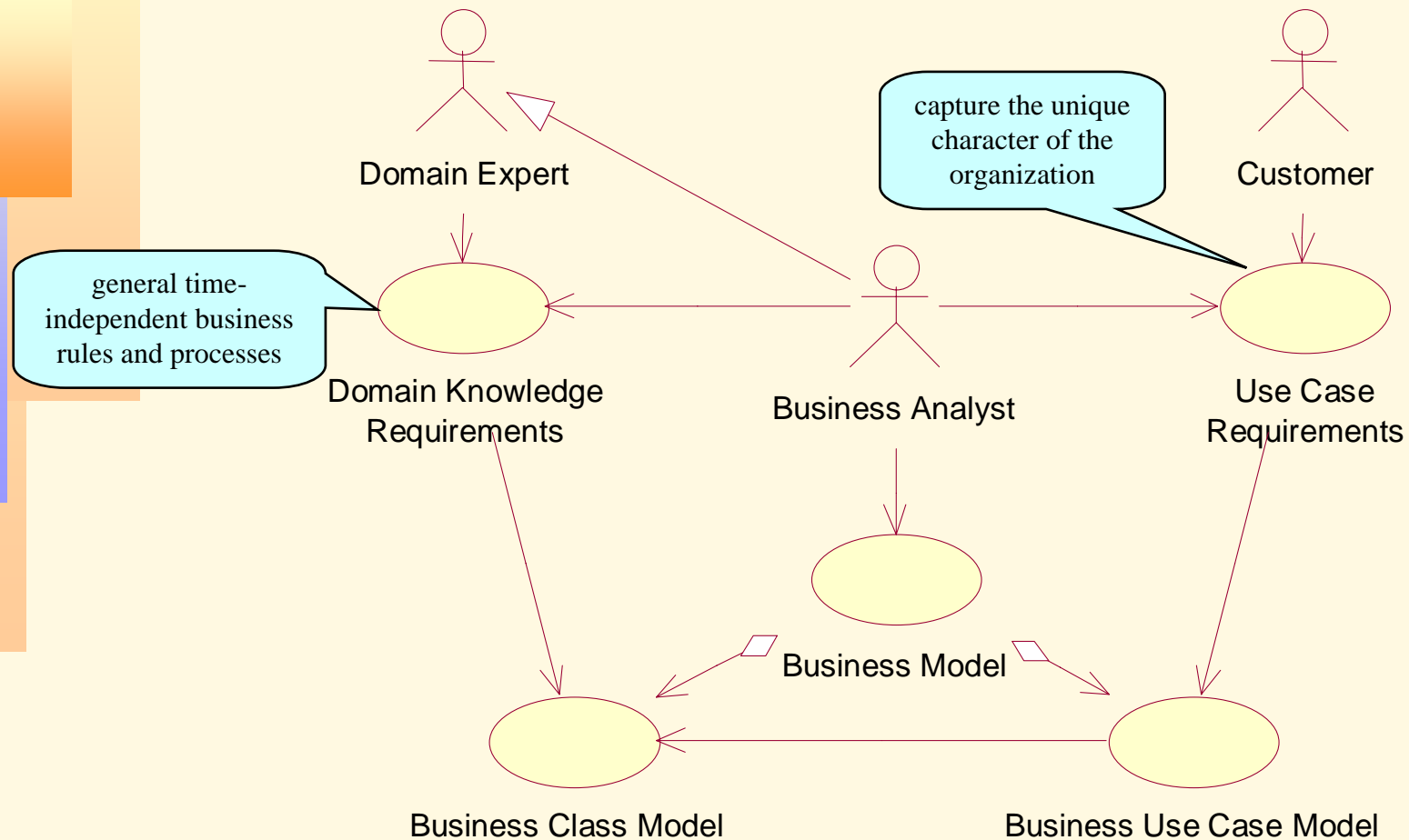
Functional and nonfunctional requirements

- *Requirements elicitation*
 - *concerned mostly with **functional reqs***
 - *...but **nonfunctional requirements** cannot be an afterthought*
- *Reviews and re-negotiations*
- *Requirements document*
- *A **moving target** even after the req doc is signed off*
 - ***requirements management** is concerned, inter alia, with managing change and estimating the impact of change on the project*

Nonfunctional requirements

- **Usability** – ease of use
 - *documentation and help, training, user interface, error handling*
- **Reusability** – ease of component reuse
- **Reliability**
 - *mean time between failures, graceful recovery, uninterrupted availability, accuracy of results*
 - *reliable = dependable (we can depend on it)*
- **Performance**
 - *response time, transaction throughput, resource consumption, concurrency level*
- **Efficiency** – in terms of time and cost
- **Supportability** = *understandability + maintainability + scalability*
- **Other constraints**
 - *policy decisions, legal issues, portability and interoperability demands, timeliness of product delivery*

Requirements elicitation



Class – an abstraction that describes a set of objects with common attributes, operations, relationships, and constraints

Use case – a major functional unit

Traditional methods of requirements elicitation

- *Simple and cost effective*
- *When clear objectives and low project risks*
- *Include:*
 - *Interviewing customers and domain experts*
 - *Questionnaires*
 - *Observation*
 - *Study of documents and software systems*

Interviewing customers and domain experts

- ***With customers*** – mostly use case reqs
- ***With domain experts*** – frequently a straight knowledge transfer
- ***Structured (formal) interview***
 - *Open-ended questions (unanticipated responses)*
 - *Close-ended questions (a list of possible responses known)*
- ***Unstructured (informal) interview***
- ***Questions to be avoided***
 - ***Opinionated*** questions (*do we have to do things the way we do them?*)
 - ***Biased*** questions (*you are not going to do this, are you?*)
 - ***Imposing*** questions (*you do things this way, don't you?*)
- ***Summary report*** of the interview should be sent to the interviewee within a day or two, along with a request for comments

Interview – kinds of questions

- *about specific details*
 - *five Ws: what, who, when, where, and why.*
- *about vision for the future*
- *about alternative ideas*
 - *these may be questions to an interviewee and suggestions from the interviewer asking for opinions*
- *about minimally acceptable solution to the problem*
 - *good usable systems are simple systems*
- *about other sources of information*
 - *can discover important documents and other knowledge sources unknown before to the interviewer*
- *soliciting diagrams*
 - *drawn by an interviewee to explain business processes may prove invaluable for understanding the requirements*

Questionnaires

- ***In addition to interviews***
- ***A passive technique***
 - *advantage – time to consider the answers*
 - *disadvantage – no possibility to clarify questions and answers*
 - *who are the people which did not respond? how would they respond?*
- ***Close-ended questions***
 - ***Multiple-choice questions***
 - *additional comments may be allowed*
 - ***Rating questions (e.g. strongly agree, agree, ...)***
 - *when seeking opinions*
 - ***Ranking questions***
 - *ranked with numbers, percent values, etc.*

Observation

- ***In addition** to interviews (and possibly questionnaires)*
- *When the user cannot convey sufficient information and/or has only fragmented knowledge*
- **Three forms**
 - **Passive**
 - *no interruption or direct involvement*
 - *video camera may be used*
 - **Active**
 - **Explanatory**
 - *explaining what is done when observed*
- *Should be carried for a **prolonged time**, at different times and at different workloads*
- *People tend to **behave differently** when watched*
 - *'**work to rule**' is a form of industrial action*
 - *'**knowledge work**' is not susceptible to observation*

Study of documents and software systems

- ***Always*** used, but may target portions of the system
- ***Use case*** requirements
 - *Organizational documents*
 - *including procedures, policies, descriptions, plans, charts, internal and external correspondence*
 - *System forms and reports (if prior computer system exists)*
 - *record of change requests (defects and enhancements)*
- ***Domain knowledge*** requirements
 - *domain journals and reference books*
 - *ERPSs*
 - *using Internet searches*

Modern methods of requirements elicitation

- *Offer better insights at higher cost and effort*
- *When high project risks (unclear objectives, undocumented procedures, unstable requirements, eroded user expertise, inexperienced developers, insufficient user commitment, etc.)*
- *Include:*
 - *Prototyping*
 - *Brainstorming*
 - *Joint Application Development (JAD)*
 - *Rapid Application Development (RAD)*

Prototyping

- **'Quick and dirty'** solution to obtain feedback
- Necessary in **complex** and **innovative** projects
- Two kinds:
 - **Throw-away** prototype
 - targets requirement determination
 - **Evolutionary** prototype
 - targets the speed of product delivery

Brainstorming

- *to form new ideas or to find a solution to specific problem **by putting aside** judgment, criticism, social inhibitions and rules*
- *to reach consensus among stakeholders*
- *'cool' analysis and decision making done afterwards*
- *The process:*
 - *prior to meeting, the **facilitator** provides **probortunity statement** (the problem/opportunity area)*
 - *generic trigger questions to challenge the participants (e.g. what features should the system support? what are the main 'business objects'?)*
 - *12 to 20 participants **around a table**, feeling equal with the facilitator "one of the crowd"*
 - *answers to trigger questions **recorded and passed around** to stimulate more answers/ideas*
 - *answers/ideas get **anonymous***
 - *answers/ideas are **discussed***
 - ***voting** to prioritize answers/ideas*

JAD

- **Brainstorming-like technique capitalizing on *group dynamics***
 - *Groups increase productivity, learn faster, make more educated judgments, eliminate more errors, take riskier decisions (this may be a negative though!), focus participants' attention to most important issues, integrate people, etc*
- **Frequently part of *facilities management* - when the operation/implementation of an organization's information systems are contracted out to a third party**
- **The *membership***
 - *Leader (communication skills, not a stakeholder, knowledge of business domain)*
 - *Scribe (touch typing, software development knowledge, CASE tool skills)*
 - *Customers*
 - *Users*
 - *Managers*
 - *Developers*

RAD

■ Five techniques

- **Evolutionary prototyping**
- **CASE tools**
 - with code generation and round-trip engineering
- **Specialists with Advanced Tools (SWAT)**
 - colocated with the users
- **Interactive JAD**
 - scribe replaced by a SWAT team with collaborative CASE tools
- **Timeboxing**
 - no 'scope creep', timeboxed project

■ Problems

- suitable for **smaller projects**; too risky for larger developments
 - inconsistent GUI
 - specialized solutions with little reuse potential
 - deficient docs
 - unsupportable solution likely

Requirements negotiation and validation

- *Needed because reqs*
 - **overlap** and **conflict** → requirements dependency matrix (next slide)
 - may be **ambiguous** or **unrealistic**
 - may remain **undiscovered**
 - may be **out of scope** (as captured by a reference model such as a context diagram (explained later))
 - sometimes out of the “project” scope, but in the scope of the “information system” (req too difficult to implement and should be done manually, may be of low priority, may be implemented in hardware)
- *frequently done in parallel with requirements elicitation*
- *inseparable from the production of requirements document*
 - *negotiation starts from the draft req doc*
 - *validation reviews and ‘rubber stamps’ the doc*

Requirements dependency matrix

Requirement	R1	R2	R3	R4
R1				
R2	Conflict			
R3				
R4		Overlap	Overlap	

Requirements risks and priorities

- ***Risk** is a threat to the project plan*
- *Risks determine project's **feasibility***
- ***Risk analysis** identifies requirements that are likely to cause development difficulties*
- ***Prioritization** is needed to allow easy rescoping of the project when faced with delays*
- ***Risk categories***
 - *Technical*
 - *Performance*
 - *Security*
 - *Database integrity*
 - *Development process*
 - *Political*
 - *Legal*
 - *Volatility*

Requirements management

- *Requirements identification and classification*
- *Requirements hierarchies*
- *Change management*
- *Requirements traceability*

Requirements identification and classification

■ **Natural language statements**

- *'The system shall schedule the next phone call to a customer upon telemarketer's request.'*

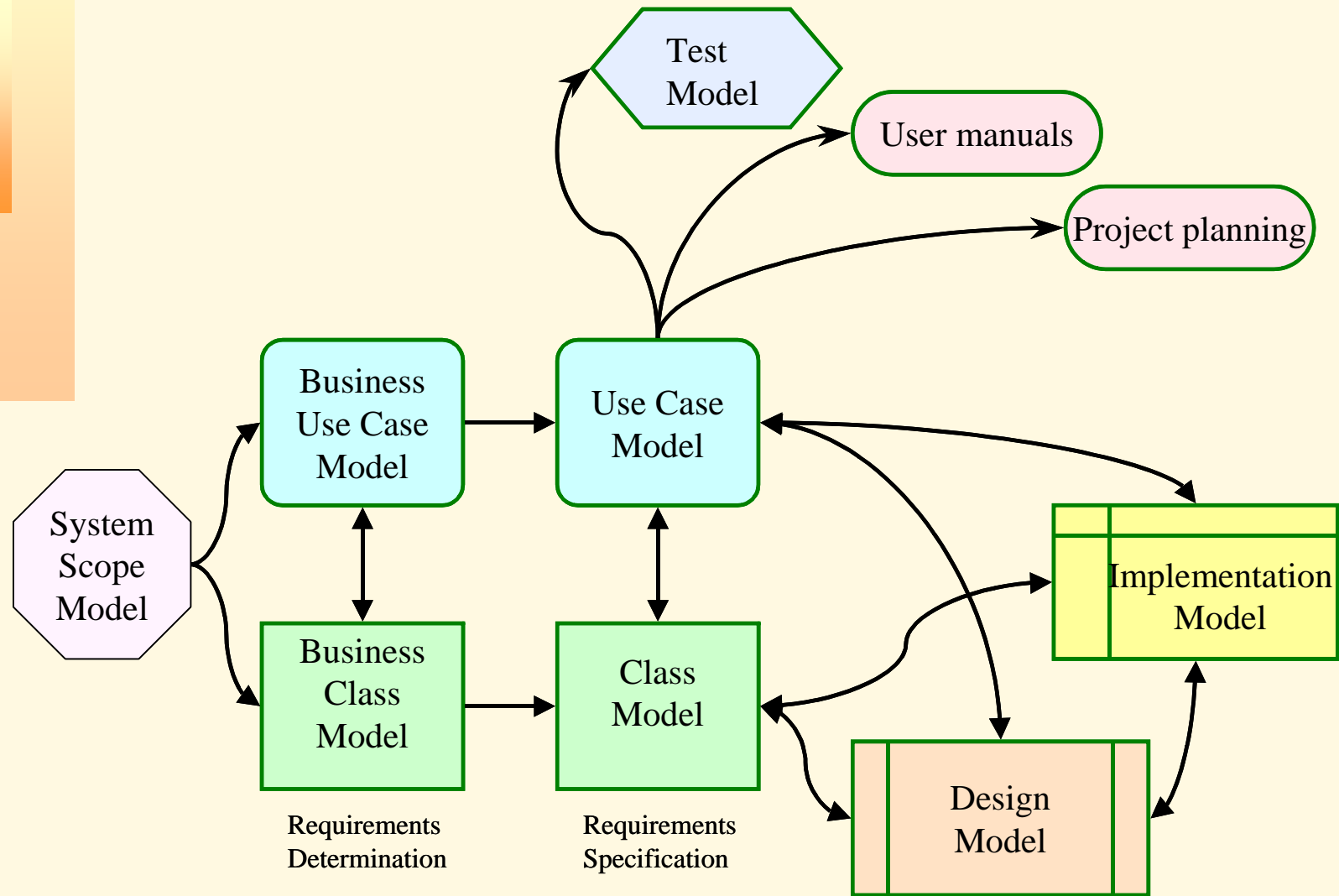
■ **Identification and classification scheme**

- **Unique identifier** (automatically generated)
 - *database generated, if possible*
 - *including version number, if possible*
- **Sequential number with document hierarchy**
 - *the seventh requirement in the third section of the second chapter would be numbered 2.3.7*
- **Sequential number with requirement's category**
 - *where the categories of requirements can be: function requirement, data requirement, performance requirement, security requirement, etc*

Requirements hierarchies

- *Parent-child relationships*
- *Reflect varying abstraction levels*
- 1. *"The system shall schedule the next phone call to a customer upon telemarketer's request."*
 - 1.1 *"The system shall activate Next Call push button upon entry to Telemarketing Control form or when the previous call has terminated."*
 - 1.2 *"The system shall remove the call from the top of the queue of scheduled calls and make it the current call."*
 - 1.3 *etc.*

Requirements business model



Telemarketing – problem statement

A charitable society sells lottery tickets to raise funds. The fundraising is done in **campaigns** to support currently important charitable causes. The society keeps a list of past contributors (**supporters**). For each new campaign, a subset of these supporters is pre-selected for telemarketing and/or direct mail contact.

The society uses some innovative schemes to gain new supporters. The schemes include special **bonus campaigns** to reward supporters for bulk buying, for attracting new contributors, etc. The society does not randomly target potential supporters by using telephone directories or similar means.

To support its work, the society decided to contract out the development of a new telemarketing application. The new system is required to support up to fifty telemarketers working simultaneously. The system must be able to **schedule the phone calls** according to pre-specified priorities and other known constraints.

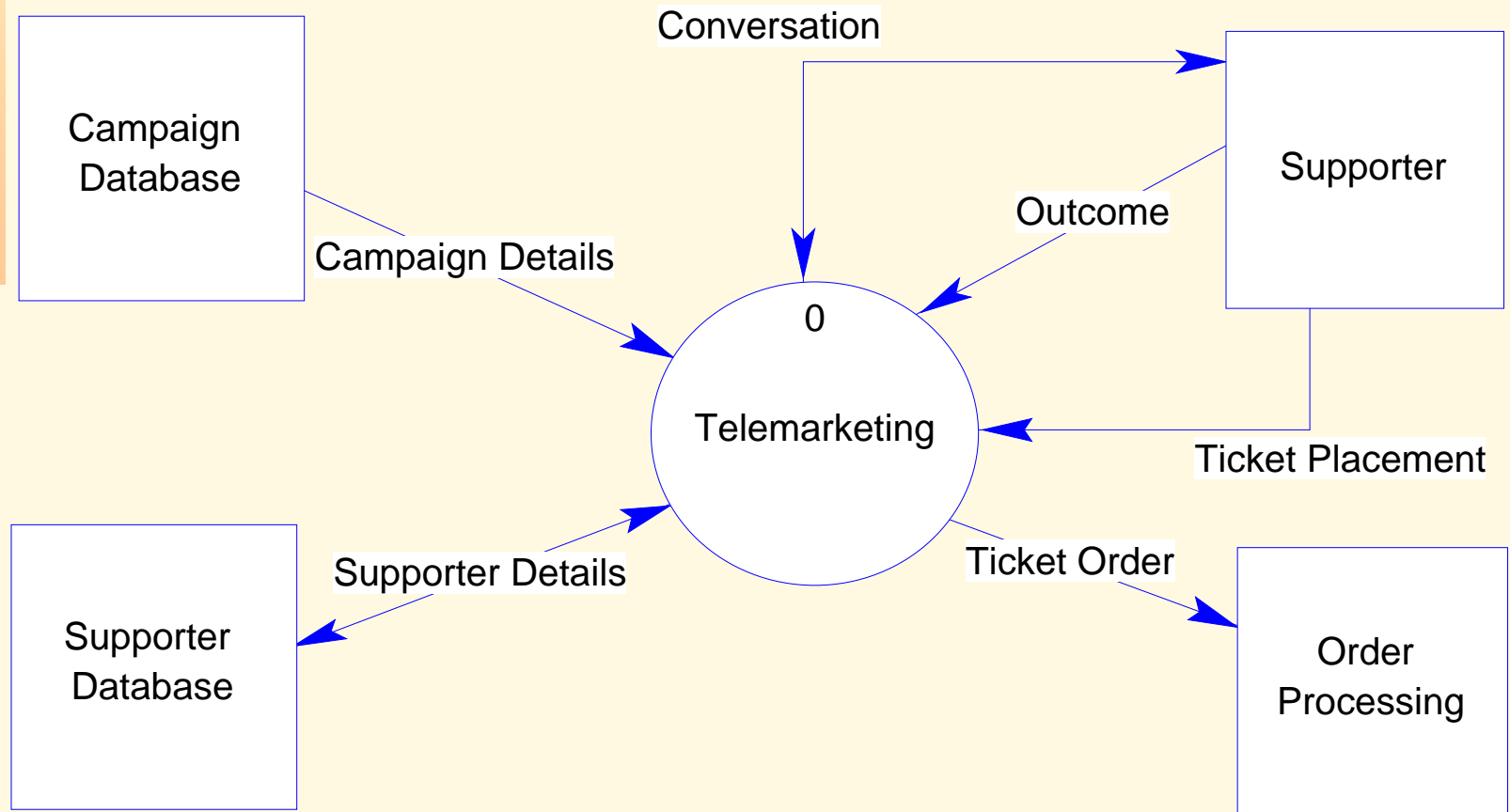
The system is required to **dial up** the scheduled phone calls. Unsuccessful connections must be **re-scheduled** and tried again later. Telephone **callbacks** to supporters must also be arranged. The **conversation outcomes**, including ticket orders and any changes to supporter records, ought to be maintained.

Telemarketing example

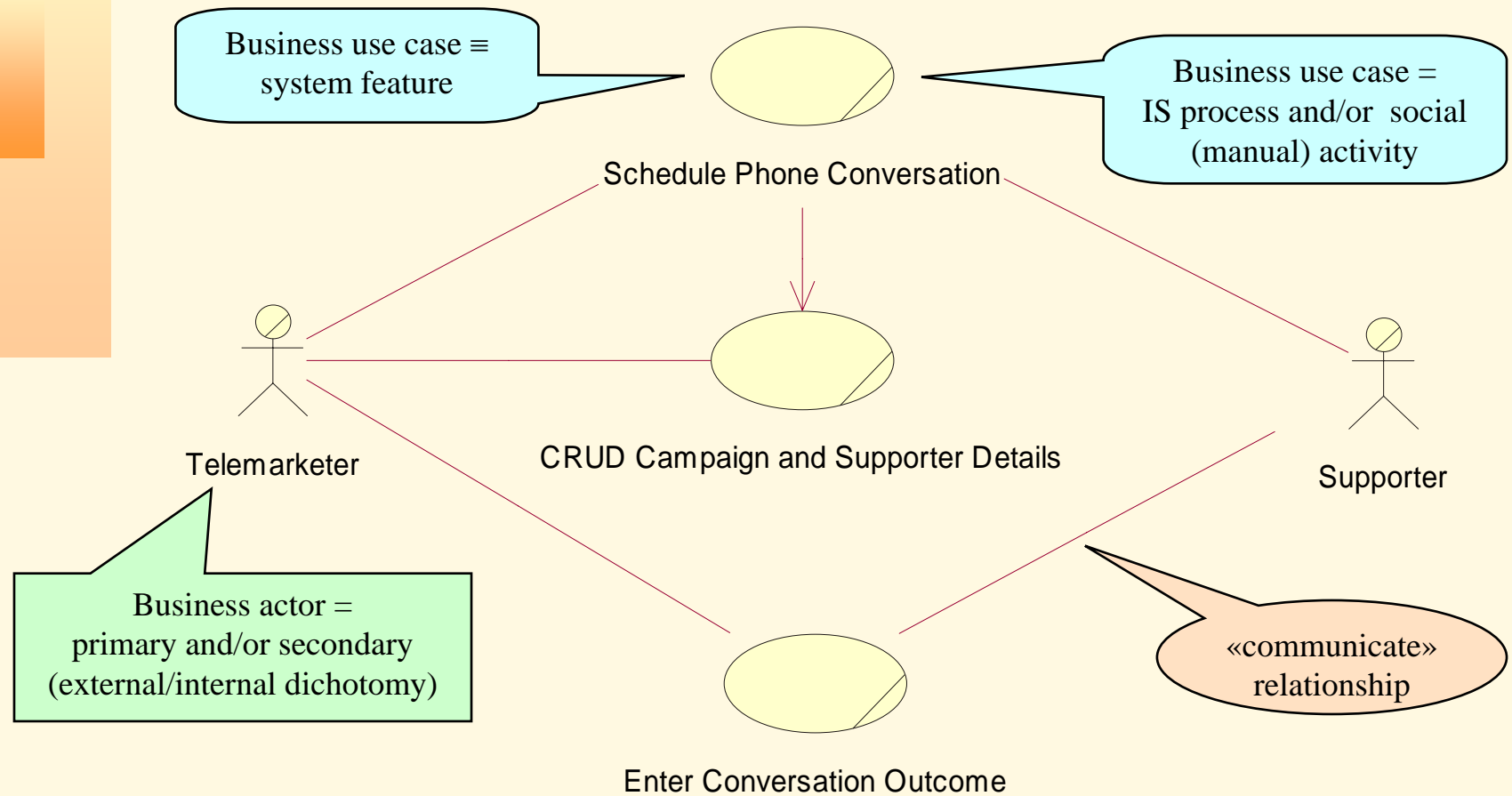
■ *Telemarketing*

- *The campaigns are planned on recommendation from the society trustees*
- *The campaigns have to be approved by the local government*
- *The design and planning of campaigns is supported by a separate Campaign Database application system*
- *There is also a separate Supporter Database that stores and maintains information about all past and present supporters – used to select supporters to be contacted in a particular campaign*
- *Orders from supporters for lottery tickets are recorded during telemarketing for perusal by the Order Processing system*
- *Order Processing System maintains status of orders in the Supporter Database*

System scope model – context diagram



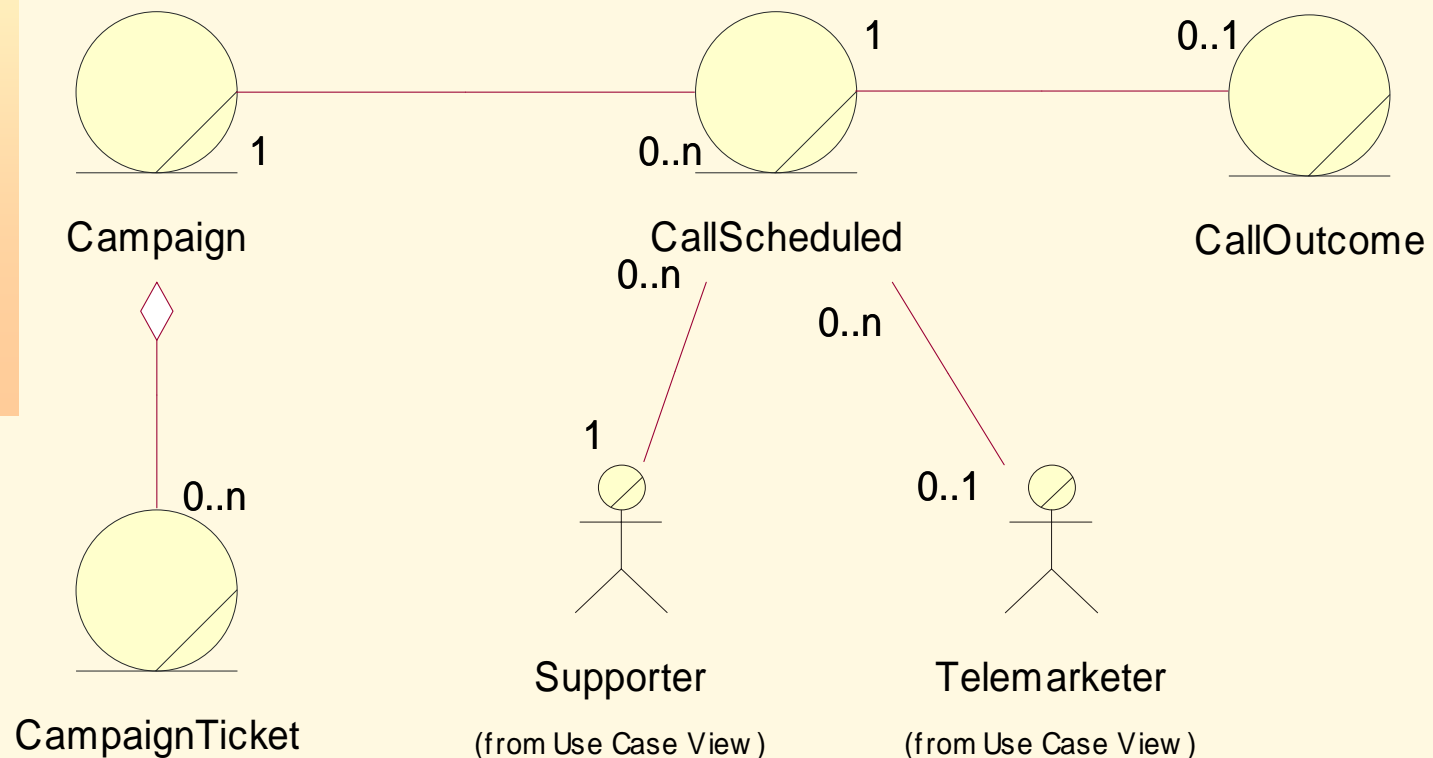
Business use case model



Business glossary

Term	definition
bonus campaign	A special series of activities, conducted within a <i>campaign</i> , to additionally entice <i>supporters</i> to buy the campaign <i>tickets</i> . Typical examples are giving free tickets for bulk or early buying or for attracting new supporters. A particular kind of bonus campaign can be used in many campaigns.
campaign	A government approved and carefully planned series of activities which are intended to achieve a <i>lottery</i> objective.
draw	An act of randomly choosing a particular <i>lottery ticket</i> as a winning ticket.
lottery	A funds raising game of chance, organized by the charity in order to make money, in which people (<i>supporters</i>) buy numbered <i>tickets</i> to have a chance of winning a <i>prize</i> if their number is chosen in a <i>draw</i> .
placement	Acquisition of one or more <i>lottery tickets</i> by a <i>supporter</i> during <i>telemarketing</i> . The placement is paid by a supporter with a credit card.

Business class model



1. The emphasis in the system is on call scheduling. The call scheduling itself is a procedural computation, i.e. the solution to it is algorithmic and computational. Nevertheless, the scheduled call queues and the outcomes of calls must be stored in some data structure.
2. Information about actors is stored in classes.

Requirements document

Requirements Document

Table of Contents

- 1. Project Preliminaries**
 - 1.1 Purpose and Scope of the Product
 - 1.2 Business Context
 - 1.3 Stakeholders
 - 1.4 Ideas for Solutions
 - 1.5 Document Overview
 - 2. System Services**
 - 2.1 The Scope of the System
 - 2.2 Function Requirements
 - 2.3 Data Requirements
 - 3. System Constraints**
 - 3.1 Interface Requirements
 - 3.2 Performance Requirements
 - 3.3 Security Requirements
 - 3.4 Operational Requirements
 - 3.5 Political and Legal Requirements
 - 3.6 Other Constraints
 - 4. Project Matters**
 - 4.1 Open Issues
 - 4.2 Preliminary Schedule
 - 4.3 Preliminary Budget
- Appendices**
- Glossary
 - Business Documents and Forms
 - References

Project preliminaries

- *Targets managers and decision makers*
- *Begins with purpose and scope of the project*
- *Makes a business case for the system*
- *Identifies stakeholders*
- *Offers initial ideas for the solution*
 - *including off-the-shelf solutions*
 - *off-the-shelf does not dispense with the need for RASD*
- *Includes an overview of the rest of the document*

System services

- *Dedicated to the definition of **system services** - what the system must accomplish*
- *Likely to account for more than half of the entire document*
- *Contains high-level requirements business models*
 - ***Context diagram** (the system scope)*
 - ***Business use case diagram** (function requirements)*
 - ***Business class diagram** (data requirements)*
 - *main attributes, but no operations*
 - ***Business glossary** moved to the Appendix*

System constraints

- *Dedicated to the definition of system constraints - how the system is constrained when accomplishing services with regard to*
 - *Interface requirements*
 - *'look and feel' only*
 - *Performance requirements*
 - *response times, but also reliability, availability, throughput indicators*
 - *Security requirements*
 - *access privileges*
 - *Operational requirements*
 - *hardware/software environment*
 - *Political and legal requirements*
 - *Other constraints*
 - *usability*
 - *maintainability*

Project matters

- *Open issues*
 - *Future requirements*
 - *Current requirements to be implemented in the future – **enhancements***
 - *Potential problems when the system is deployed*
- *Preliminary schedule*
 - *Human and other resources*
 - *Planning charts (PERT, Gantt)*
- *Preliminary budget*
 - *Project cost – range rather than figure*
 - *In some cases, better estimation possible (e.g. function point analysis)*

Appendices

- *Glossary*
 - *Terms*
 - *Acronyms*
 - *Abbreviations*
- *Documents and forms*
 - *Examples of completed (filled in) forms*
- *References*
 - *To books and other published sources*
 - *Meetings' minutes, memoranda, internal documents*

Summary

- **Requirements determination** is about discovering requirements and documenting them
- Two lines of discovery – the discovery from the **domain knowledge** and from the **use cases**
- **Methods of requirements elicitation** include interviewing customers and domain experts, questionnaires, observation, study of documents and software systems, prototyping, JAD and RAD
- **Requirements negotiation and validation** to resolve overlaps and conflicts
- Requirements have to be **managed**
- **Requirements business model** uses diagrams – Context Diagram, Business Use Case Diagram, and Business Class Diagram
- The resulting document is called the **Requirements Document**