

Object-Oriented Software Engineering
Conquering Complex and Changing Systems

REQ/QOC Tutorial

Allen Dutoit & Barbara Paech



Preliminaries

For everybody:

Today:

- ♦ **REQ/QOC tutorial**

Next week

- ♦ **Thursday: Configuration Management lecture**
- ♦ **Friday: TogetherJ tutorial**

For STARS students:

Today:

- ♦ **Post your GUI Mockup in PowerPoint or HTML on the Bboard**
- ♦ **Post the name of the presenter for Monday's review**

Monday

- ♦ **GUI Mockup review (5-10 minutes for each team).**

Tutorial outline

Requirements engineering

- ♦ **Summary from last 2 lectures**

Requirements engineering process for STARS

- ♦ **Elicitation & review with REQ/QOC**
- ♦ **Analysis with TogetherJ**

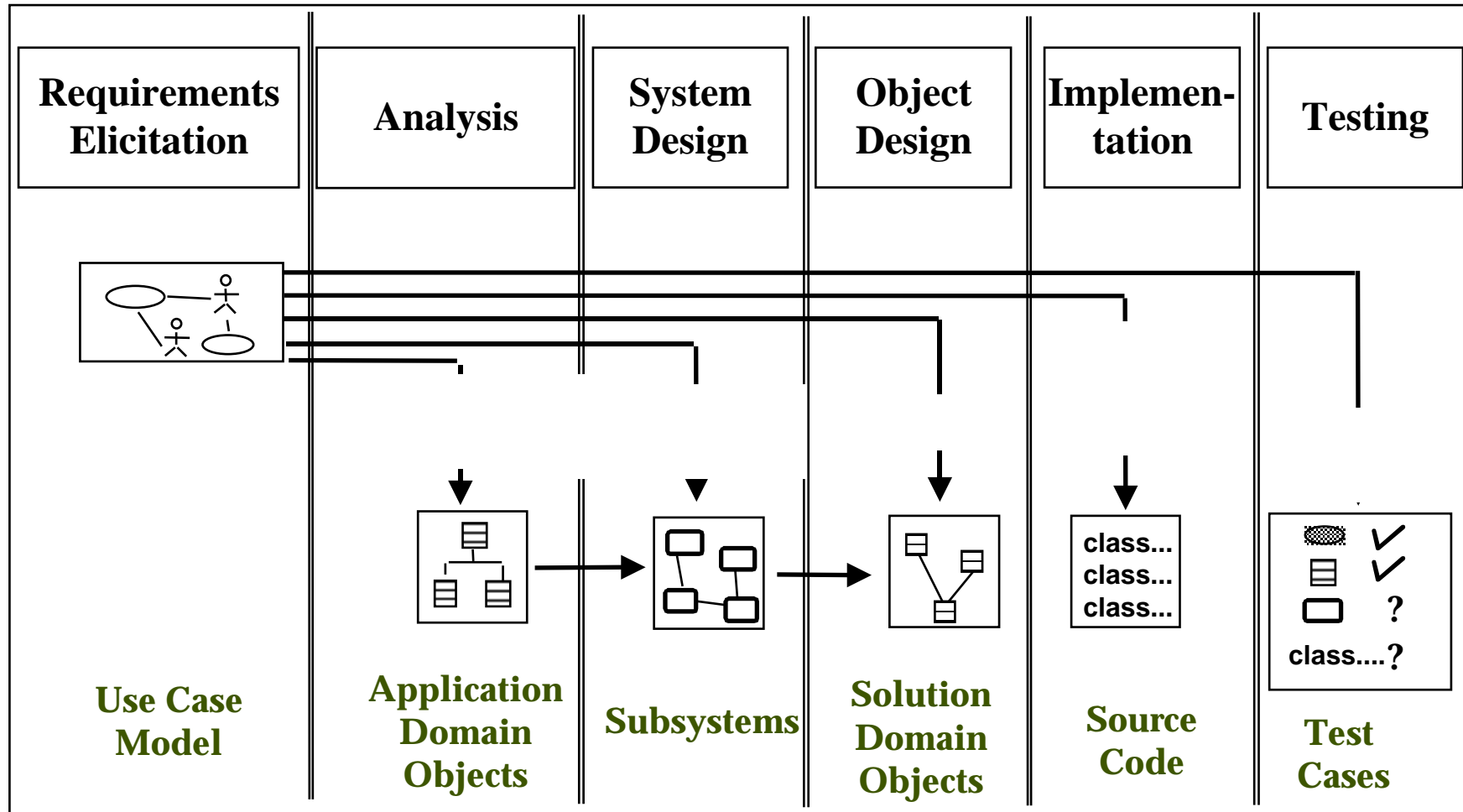
SuperMarket example

REQ/QOC

- ♦ **Requirements elicitation**
- ♦ **Review & collaboration**

Summary & next steps

Requirements Engineering Summary

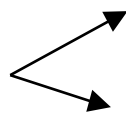


Requirements specification

A requirements specification includes 3 descriptions:

Requirements Elicitation:

*System specification
(use case model)*



Requirements: What do users do?

Interactions: How do users use the system to accomplish their work?

Analysis:

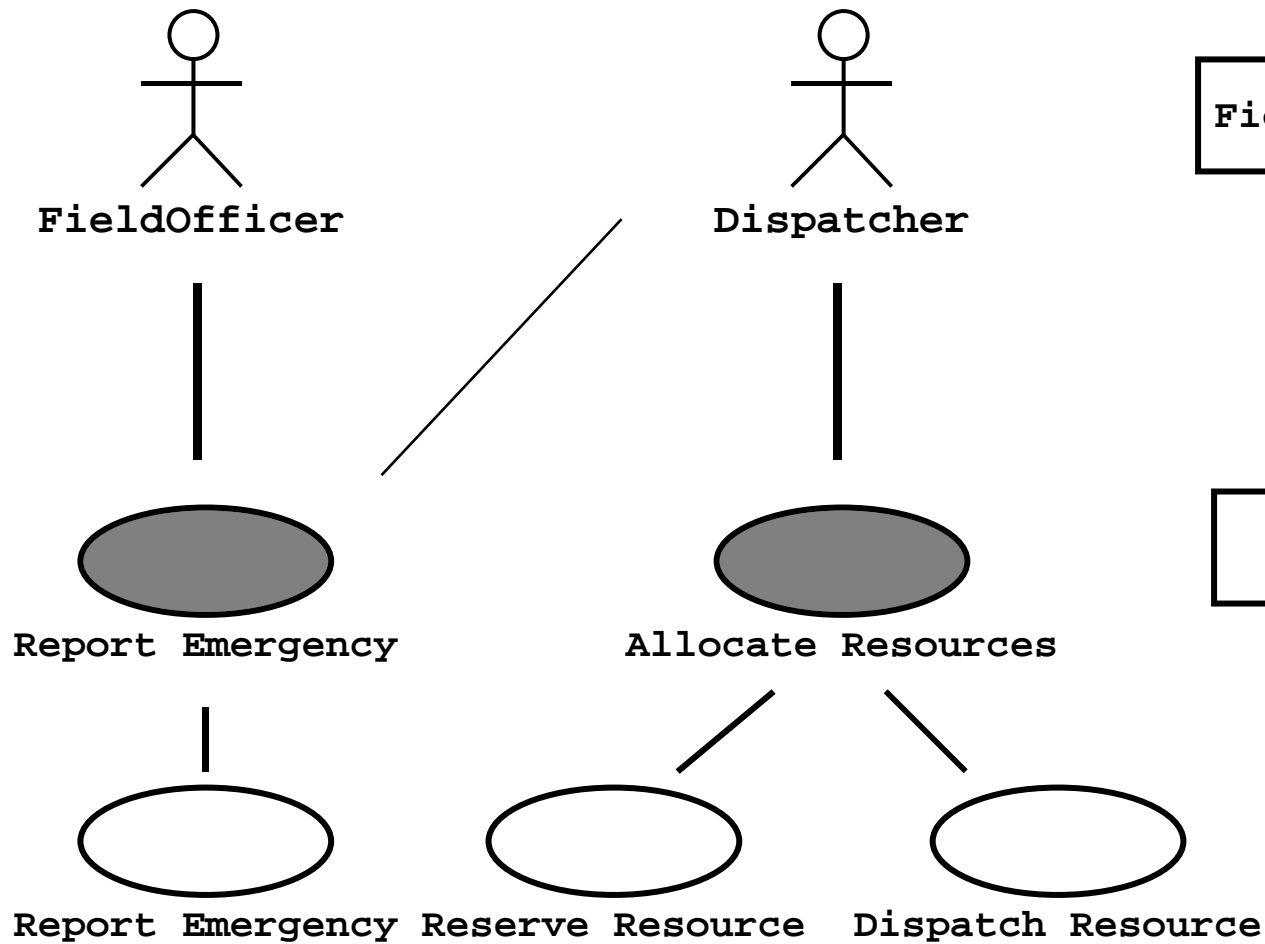
*Requirements analysis
model (object model)*



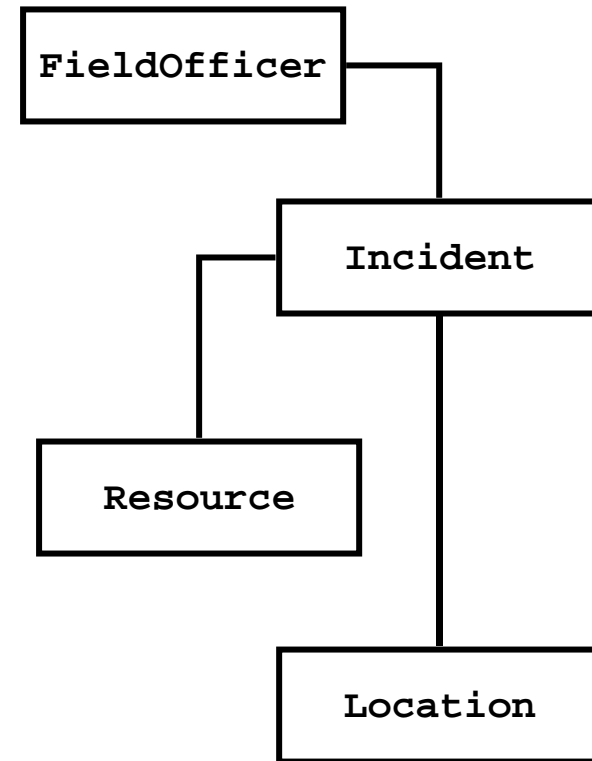
Specification: What does the system do?

Requirements specification (continued)

System specification



Requirements analysis model



Requirements elicitation activities

Define the boundary of the system:

- ◆ **Identify and describe actors**

Define the needs of the user

- ◆ **Describe one or more user tasks per actor**

Describe the interactions between the actors and the system

- ◆ **Describe one or more use cases per user task**
- ◆ **Exceptions & nonfunctional constraints**

Describe the functionality of the system

- ◆ **Identify all services needed to realize the use cases**
- ◆ **Each use case uses one or more services**
- ◆ **Each service can be used by one or more use cases**

Review the system specification with the client

Process for STARS (1)

REQ/QOC for the requirements specification

- ♦ **Web-based tool**
- ♦ **Actors, User Tasks, Use Cases, & Services**
- ♦ **Constraints & Glossary**

REQ/QOC for review and negotiation

- ♦ **Questions, Options, Criteria, Assessments**
- ♦ **Discussion**
- ♦ **What's new, what's revised, conflict detection**

TogetherJ for Analysis (class diagram)

Process for STARS (2)

Instructors/coaches

7.11 RAD v.0 from coaches

- ◆ **Actors & user tasks**

21.11 Questions from coaches

27.11 More questions ...

Teams

Before 17.11

- ◆ **Questions to coaches**

17.11 RAD v.1 due

- ◆ **Add'l user tasks**
- ◆ **Use cases, Services, Constraints**
- ◆ **Glossary**

20.11 Requirements review

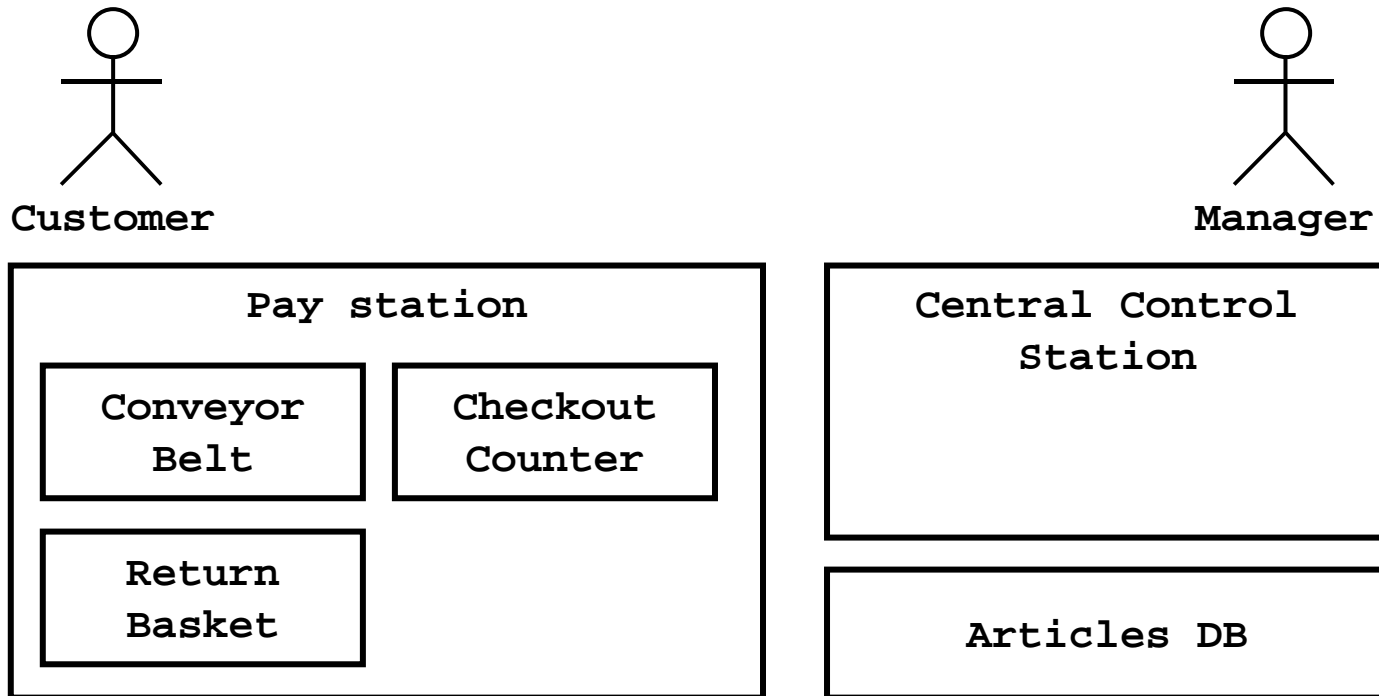
24.11 RAD v.2 due

- ◆ **Options, assessments**
- ◆ **Decisions**
- ◆ **Revised requirements elements**
- ◆ ***Analysis (TogetherJ)***

SuperMarket: Problem statement

Food vendor needs new pay system:

- ◆ **Articles can be purchased by customers**
- ◆ **Manager can monitor inventory**
- ◆ **System periodically checks for outdated articles**



REQ/QOC: Overview

and of a cancelation device with which [Customers](#) can cancel the whole transaction.

After identifying the [Articles](#) on the [Conveyor Belt](#) the system checks the data validity and informs the [Customers](#) and the [Manager](#) in case of outdated [Articles](#). Otherwise the [Articles](#) ... cases the [System](#) checks whether a given nu... and informs the [Manager](#), if this is

2. Actors [\[New Actor\]](#)

- [Customer](#)
- [Manager](#)
- [System](#)

3. Needs [\[New User Task\]](#)[\[Help\]](#)

- [Check for Outdated Articles](#) *dutoit, 11/2/00 8:22 PM* [\[Edit\]](#)[\[Delete\]](#)
- [Monitor Inventory](#) *dutoit, 11/2/00 2:56 PM* [\[Edit\]](#)[\[Delete\]](#)
- [Purchase Articles](#) *dutoit, 10/31/00 10:06 AM* [\[Edit\]](#)[\[Delete\]](#)

4. Interactions [\[New Use Case\]](#)[\[Help\]](#)

- [Check One Article](#) *dutoit, 11/2/00 3:13 PM* [\[Edit\]](#)[\[Delete\]](#)
- [Identify Article](#) *dutoit, 10/31/00 12:00 PM* [\[Edit\]](#)[\[Delete\]](#)
- [Pay For Articles](#) *dutoit, 11/2/00 8:14 PM* [\[Edit\]](#)[\[Delete\]](#)

SuperMarket Questions By Date

Subject	Status	Author	Date
[Edit] Assistance service	Resolved	dutoit	11/2/00 8:24 PM

REQ/QOC: Overview (2)

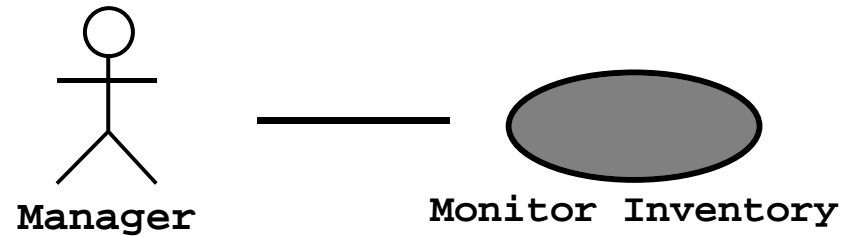
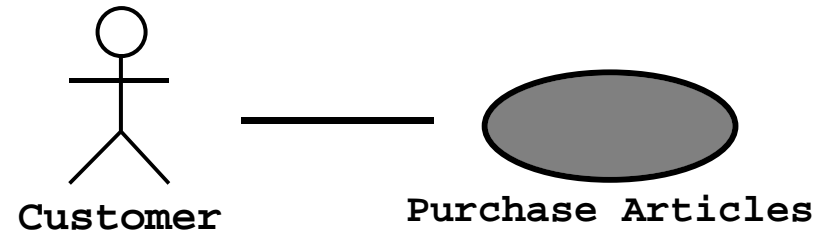
The screenshot shows a web application interface for 'System: SuperMarket-Tutorial'. The interface includes a sidebar with 'History', 'Search', 'Scrapbook', and 'Page Holder' buttons. The main content area has a title bar and navigation links: [Refresh][Printable Version] | [Question] | [Help]. Below the title bar, there are six numbered sections, each with a title, a link, and a status message:

- 1. Introduction** [Edit] *No introduction specified.*
- 2. Actors** [New Actor][Help] *No actors specified.*
- 3. Needs** [New User Task][Help] *No user tasks specified.*
- 4. Interactions** [New Use Case][Help] *No use cases specified.*
- 5. Services** [New Service][Help] *No services specified.*
- 6. Nonfunctional Constraints** [New Constraint][Help] *No constraints specified.*

Five callout boxes with red text point to specific sections:

- 'System Boundaries' points to the 'Introduction' section.
- 'Requirements' points to the 'Actors' section.
- 'Use cases' points to the 'Interactions' section.
- 'Specification' points to the 'Services' section.
- 'Nonfunctional reqs & constraints' points to the 'Nonfunctional Constraints' section.

SuperMarket: Identifying Actors & User Tasks



REQ: Creating actors

System: SuperMarket-Tutorial

[\[Refresh\]](#)[\[Printable Version\]](#) | [\[Questions\]](#)

1. Introduction [\[Edit\]](#)
No introduction specified.

2. Actors [\[New Actor\]](#)[\[Help\]](#)
No actors specified.

3. Needs [\[New User Task\]](#)[\[Help\]](#)
No user tasks specified.

4. Interactions [\[New Use Case\]](#)

New Actor:

Use the form below to specify a new actor. The name of the actor is a required field and must be unique.

Name:

Description:

Create Actor

REQ: Creating user tasks

New User Task:

Use the form below to specify a new user task. The name of the user task is a required field and must be unique. The initiating actor and the flow of events fields should be specified in a complete requirements specification but can be specified at a later time.

History Search Scrapbook Page Holder

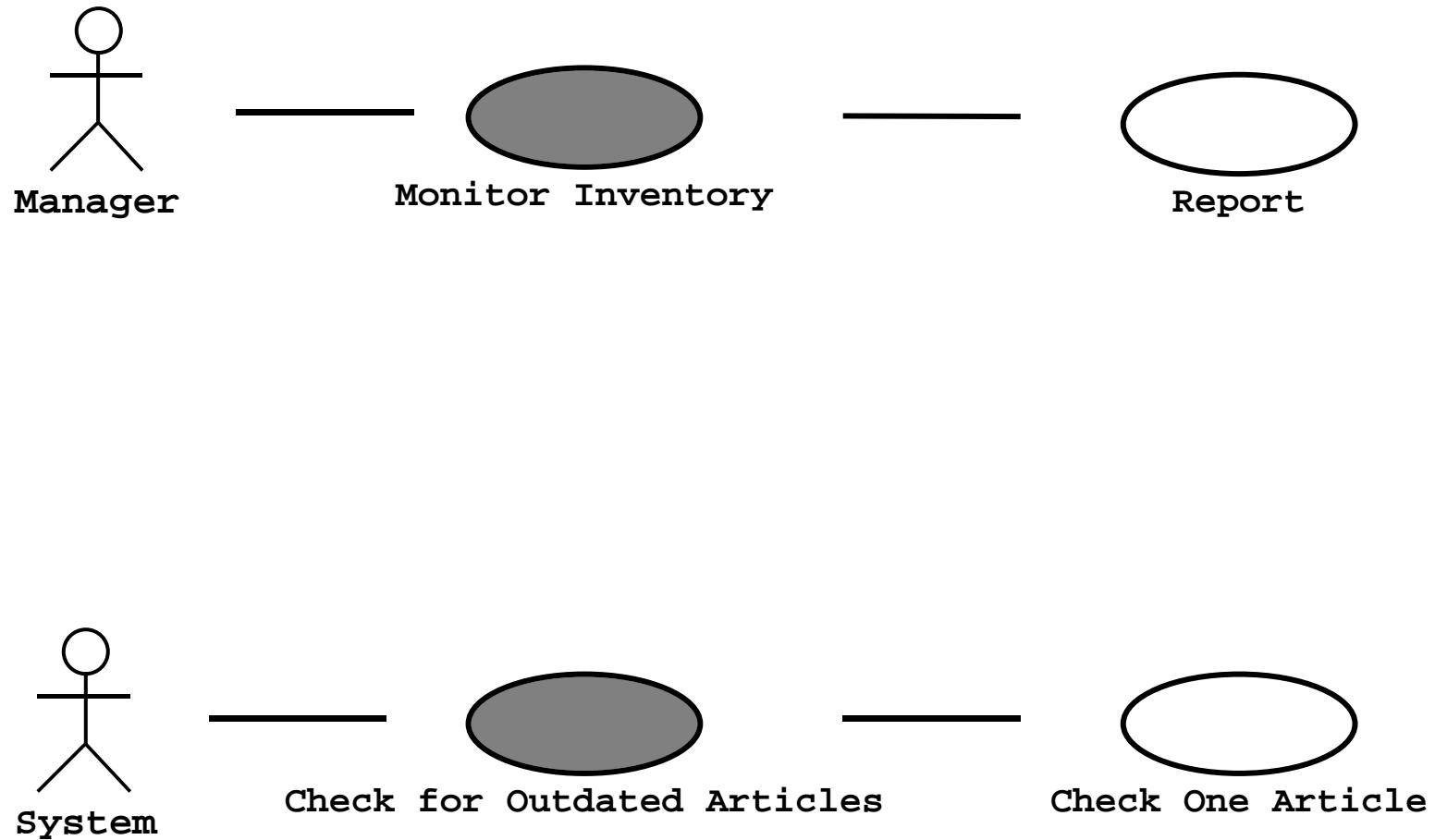
System: Super

[\[Refresh\]](#) [\[Printable Version\]](#) | [\[Question\]](#)

- 1. Introduction** [\[Edit\]](#)
No introduction specified.
- 2. Actors** [\[New Actor\]](#) [\[Help\]](#)
No actors specified.
- 3. Needs** [\[New User Task\]](#) [\[Help\]](#)
No user tasks specified.
- 4. Interactions** [\[New Use Case\]](#) [\[E\]](#)

Name	<input type="text" value="Purchase Articles"/>
Initiating Actor:	<input type="text" value="Customer"/>
Participating Actors:	<input type="text" value="Customer"/>
Flow of Events:	<input type="text" value="The Customer pay for their articles at the pay station, stack their articles on the conveyor belt, pay the resulting bill, and leave."/>
Frequency:	<input type="text"/>
Preconditions:	<input type="text" value="The Customer is in the store"/>

SuperMarket: Identifying Use Cases



REQ: Creating Use

History Search Scrapbook Page Holder

System: SuperM

[\[Refresh\]](#)[\[Printable Version\]](#) | [\[Question\]](#) | [\[H](#)

- 1. Introduction** [\[Edit\]](#)
No introduction specified.
- 2. Actors** [\[New Actor\]](#)[\[Help\]](#)
No actors specified.
- 3. Needs** [\[New User Task\]](#)[\[Help\]](#)
No user tasks specified.
- 4. Interactions** [\[New Use Case\]](#)[\[Help\]](#)

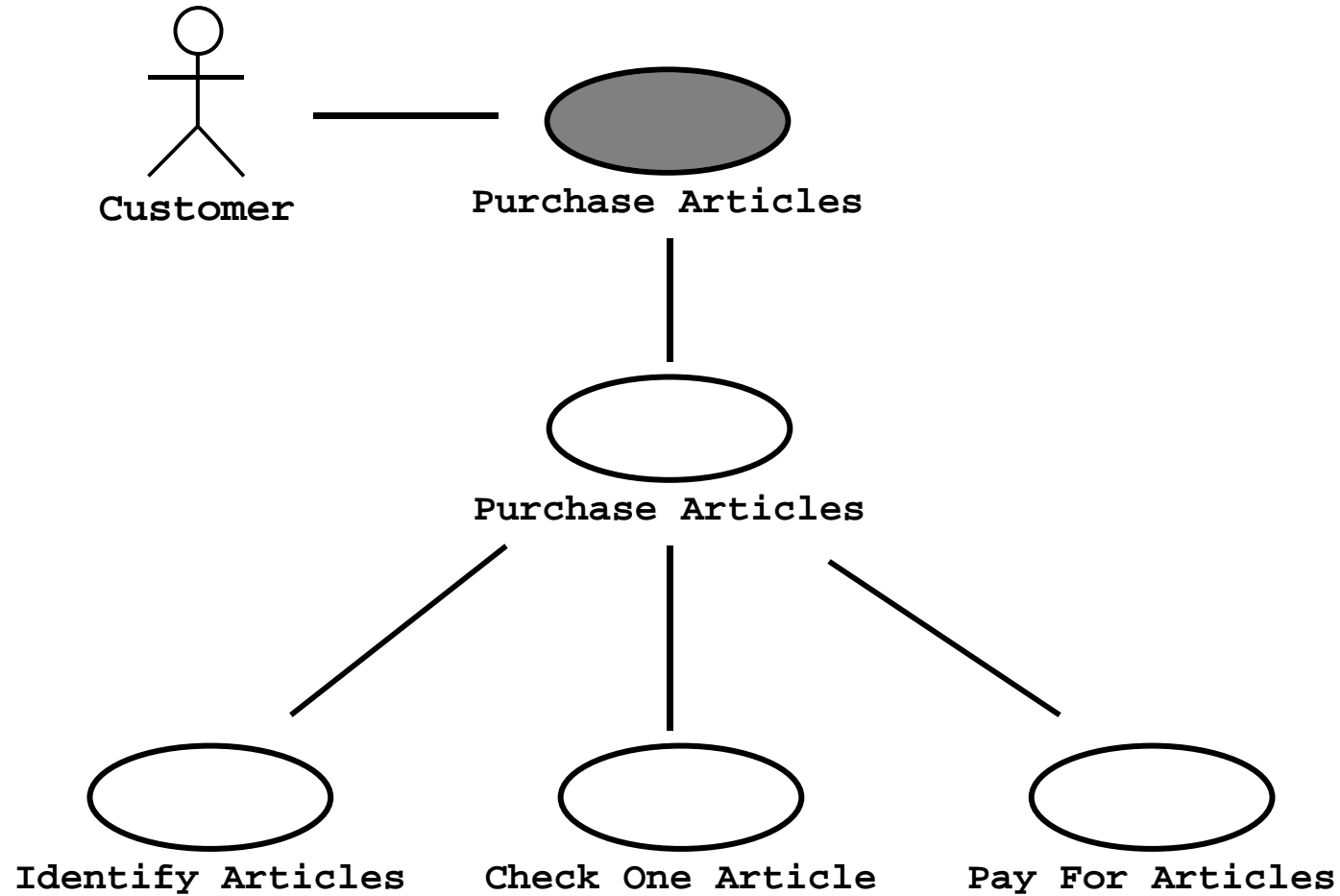
New Use Case

Use the form below to specify a new use case. The name of the is a required field and must be unique. The realized user task a flow of events fields should be specified in a complete requiren specification but can be specified at a later time.

Name:	Check One Article
Flow of events	1. System checks Article 1.1 If Article out of date: 1.1.1 System displays error message 1.1.2 System informs Manager of necessary action 1.1.3 Actor puts Article into Return Basket 1.2 If amount of Article not sufficient: 1.2.1 System informs Manager of necessary action
Preconditions:	Purchase transaction exists, Article data known

History Search Scrapbook Page Holder

SuperMarket: Refining Use Cases



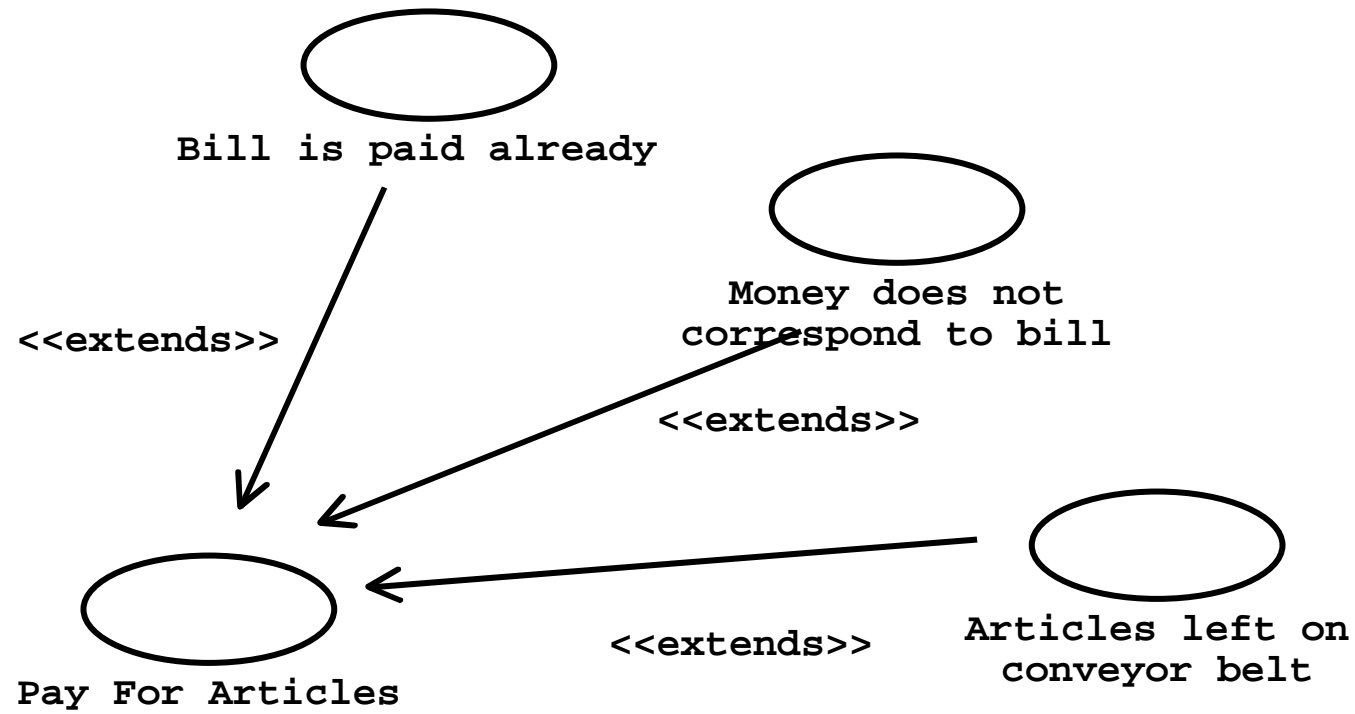
REQ: Relationships among use cases

New Use Case

Use the form below to specify a new use case. The name of the use case is a required field and must be unique. The realized user task and the flow of events fields should be specified in a complete requirements specification but can be specified at a later time.

Name:	Purchase Articles
Flow of events	<ol style="list-style-type: none">1. Customer puts article on the Conveyor Belt.2. System creates a Purchase transaction3. System identifies article (-> Use Case Identify Article)4. System checks article and informs the Manager of necessary actions (-> Use Case Check One Article)5. System registers article on the bill6. Actor put further articles on the Conveyor Belt (-> see 1-4)7. Actor requests bill8. System outputs bill9. Actor pays (-> Use Case Pay for Articles)10. System releases Articles and updates Article data in the data base11. System archives Purchase transaction12. Customer receives Articles and leaves the store At any point exception[cancelation] possible
Preconditions:	Conveyor belt is idle, Customer has collected Articles in the store and cannot leave the

SuperMarket: Identifying Exceptions



REQ: Describing exceptions

New Use Case

Use the form below to specify a new use case. The name of the use case is a required field and must be unique. The realized user task and the flow of events fields should be specified in a complete requirements specification but can be specified at a later time.

Name: Pay for Articles

Flow of events

1. Customer calls Payment function [bill is paid already] , [Articles left on the Conveyor Belt]
2. System displays a choice between cash or credit card
3. If Customer chooses cash:
4. Customer inserts money
5. System registers money [Money does not

Exceptions:

[bill is paid already]

1. System displays error message

[articles left on the Conveyor Belt]

1. System displays error message and choice between end of Purchase or continuation
2. If the actor chooses continuation:
3. Use case Pay for Articles is completed (see Use case Purchase Articles)
4. If the actor chooses end:

Create Use Case

SuperMarket: Specifying constraints

User Task: Check for Outdated Articles

[\[Refresh\]](#)[\[Edit\]](#)[\[Delete\]](#) **[\[New Constraint\]](#)** [New Use Case](#) | [\[Question\]](#) | [\[Help\]](#)

Name: Check for Outdated Articles

Initiating Actor: [System](#)

Participating Actors: [Manager](#)

Flow of Events: The [System](#) periodically checks for outdated [Articles](#).

Nonfunctional Constraints: [Min. Outdated Articles Customer Hands](#)

Use Cases: [Check One Article](#)

Open: [Clarification of Min. outdated articles](#)

Questions: [constraint](#)

[Back to the SuperMarket page](#)

Created on 10/31/00 10:06 AM by [dutoit](#).
Last revised on 11/2/00 8:22 PM by [dutoit](#).

[Link:](#) <http://atbruegge13:8080/servlet/REQ?action=NewConstraintForm&Syste>

QOC Asking questions (1)

System: SuperMarket

[Home](#) | [Printable Version](#) | [\[Question\]](#) | [Help](#)

Introduction [\[Edit\]](#)

part of its overall infrastructure a food vendor
its branches. This system shall support the follo

- purchasing of articles by the [Customers](#) and
- monitoring of the stored articles and remov

system consists of:

- a [Database of Articles](#) on sale in a branch
- a [Central Control Station](#) for the branch ma
data about deliveries and an overview of th
status includes the number of articles of eac
number of [Pay Stations](#) consisting of a C

New Question

*Enter your question below by typing a name and a brief descrip
name is used to identify the question and should be brief. The de
is used for discussions and should be complete.*

Name:

Description:

Criteria relevant to this question:

Responsive Assistance	▲
Min. Outdated Articles Customer Hands	☰
Usability of Reports	▬
Extension of report service with search service	▼

QOC: Asking questions (2)

Describe Possible Options

Specify below possible options for answering this question. Even if you do now know the right answer, providing options below can start a more focused discussion.

Question: Assistance Service
How should customers request assistance in case of failure of the pay station or confusion of the customer?

Option1:

Option2:

Option3:

QOC: Resolving questions

Question: Clarification: functions vs. services

[\[Refresh\]](#) [\[Edit\]](#) [\[Select Criteria\]](#) [\[Select Refs\]](#) [\[Reopen\]](#)

Question: Are functions, system services? (*dutoit, 11/2/00 9:39 PM*)

References: [NFC: Extension of report service with search service](#)

Decision: Replace all occurrences of "function" with "service". (*paech, 11/2/00 9:48 PM*)

Options [\[Propose Option\]](#)

- [\[Edit\]](#) Replace all occurrences of "function" with "service". (*dutoit, 11/2/00 9:40 PM*)
- [\[Edit\]](#) Replace all occurrences of "service" with "function". (*dutoit, 11/2/00 9:40 PM*)
- [\[Edit\]](#) Do nothing (services and functions are different) (*dutoit, 11/2/00 9:40 PM*)

Discussion [\[Post Comment\]](#)

		Author	Date
[Reply] new!	Sorry, I got the terminology mixed up. I will change "function" to "service".	<i>paech</i>	<i>11/2/00 9:47 PM</i>

REQ: Exporting the requirements specification

System: SuperMarket

[Refresh](#) | [Printable Version](#) | [Question](#) | [Help](#)

1. Introduction [Edit](#)

As part of its overall infrastructure a food vendor wants to install a new pay system in all its branches. This system shall support the following tasks of the branches:

- purchasing of articles by the [Customers](#) and
- monitoring of the stored articles and removal of outdated articles.

The system consists of:

- a [Database of Articles](#) on sale in a branch
- a [Central Control Station](#) for the branch management which allows the data about deliveries and an overview of the current status of all article status includes the number of articles of each kind and the date of valid
- a number of [Pay Stations](#) consisting of a [Conveyor Belt](#) where the [Cus](#) place their articles, of a scanner which identifies the articles, of a check counter which allows for Pay for [Articles](#) with cash or credit card, of a [Basket](#) where [Customers](#) can place not paid [Articles](#) , and of a cancel device with which [Customers](#) can cancel the whole transaction.

Summary

Requirements elicitation focuses on defining the system from the user's point of view.

RE is a collaborative activity.

REQ/QOC is a tool for supporting RE and collaboration:

- ◆ **Definition of requirements specification**
- ◆ **questions about the requirements elements**
- ◆ **Discussion, negotiation, and resolution of questions**
- ◆ **Finding out what others have done**

More about QOC in 2 weeks.

STARS deadlines

- ◆ **RAD v.1 17.11**
- ◆ **RAD v.2 24.11**

Credits

Definition & Realization of REQ/QOC:

Daniela Ahlisch, Kagan Aksit, Allen Dutoit, Barbara Paech

Testing and Feedback

Andreas Braun, Bernd Brügge, Günter Teubner