# Lecture 9

Use Case Modelling

Inception Phase Requirements Discipline

# Agenda

- Goals and Scope of a Use Case
- Finding Primary Actors, Goals, and Use Cases

# How should use cases be discovered?

- Which of these is a valid use case?
- Negotiate a Supplier Contract
- Handle Returns
- Log In

# EBP

- For requirements analysis for a computer application, focus on use cases at the level **of elementary business processes (EBPs).**
  - A task performed by one person in one place at one time, in response to a business event, which adds measurable business value and leaves the data in a consistent state. e.g., Approve Credit or Price Order [original source lost].

# *Use Cases and Goals*

- And it leads to a recommended procedure:
- 1. Find the user goals.
- 2. Define a use case for each.

# Finding Primary Actors, Goals, and Use Cases

- Choose the system boundary. Is it just a software application, the hardware and application as a unit, that plus a person using it, or an entire organiza tion?
- Identify the primary actors.
- For each, identify their user goals. Raise them to the highest user goal level that satisfies the EBP guideline.
- Define use cases that satisfy user goals; name them according to their goal.

# *Step 1: Choosing the System Boundary*

- For this case study, the POS system itself is the system under design;
- everything outside of it is outside the system boundary, including the cashier, payment authorization service, and so on.

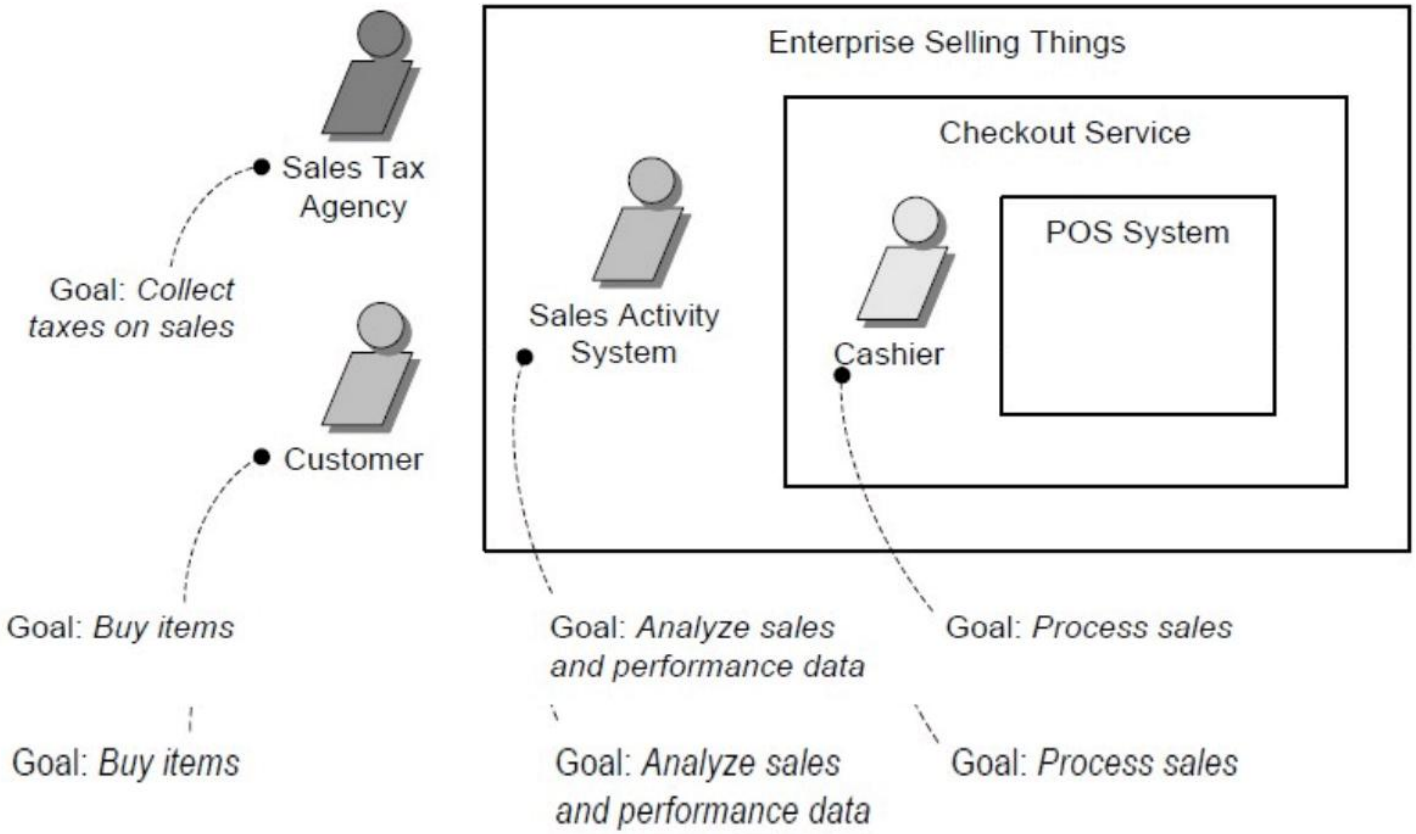
# *Steps 2 and 3: Finding Primary Actors and Goals*

In addition to obvious primary actors and user goals, the following questions help identify others that may be missed:

| | |
|---|---|
| Who starts and stops the system? | Who does system administration? |
| Who does user and security management? | Is "time" an actor because the system does something in response to a time event? |
| Is there a monitoring process that restarts the system if it fails? | Who evaluates system activity or performance? |
| How are software updates handled? Push or pull update? | Who evaluates logs? Are they remotely retrieved? |

| Actor | Goal | Actor | Goal |
|---|---|---|---|
| Cashier | process sales<br>process rentals<br>handle returns<br>cash in<br>cash out<br>… | System Administrator | add users<br>modify users<br>delete users<br>manage security<br>manage system tables<br>… |
| Manager | start up<br>shut down<br>… | Sales Activity System | analyze sales and performance data |
| … | … | … | … |

# Primary actors and goals at different system boundaries

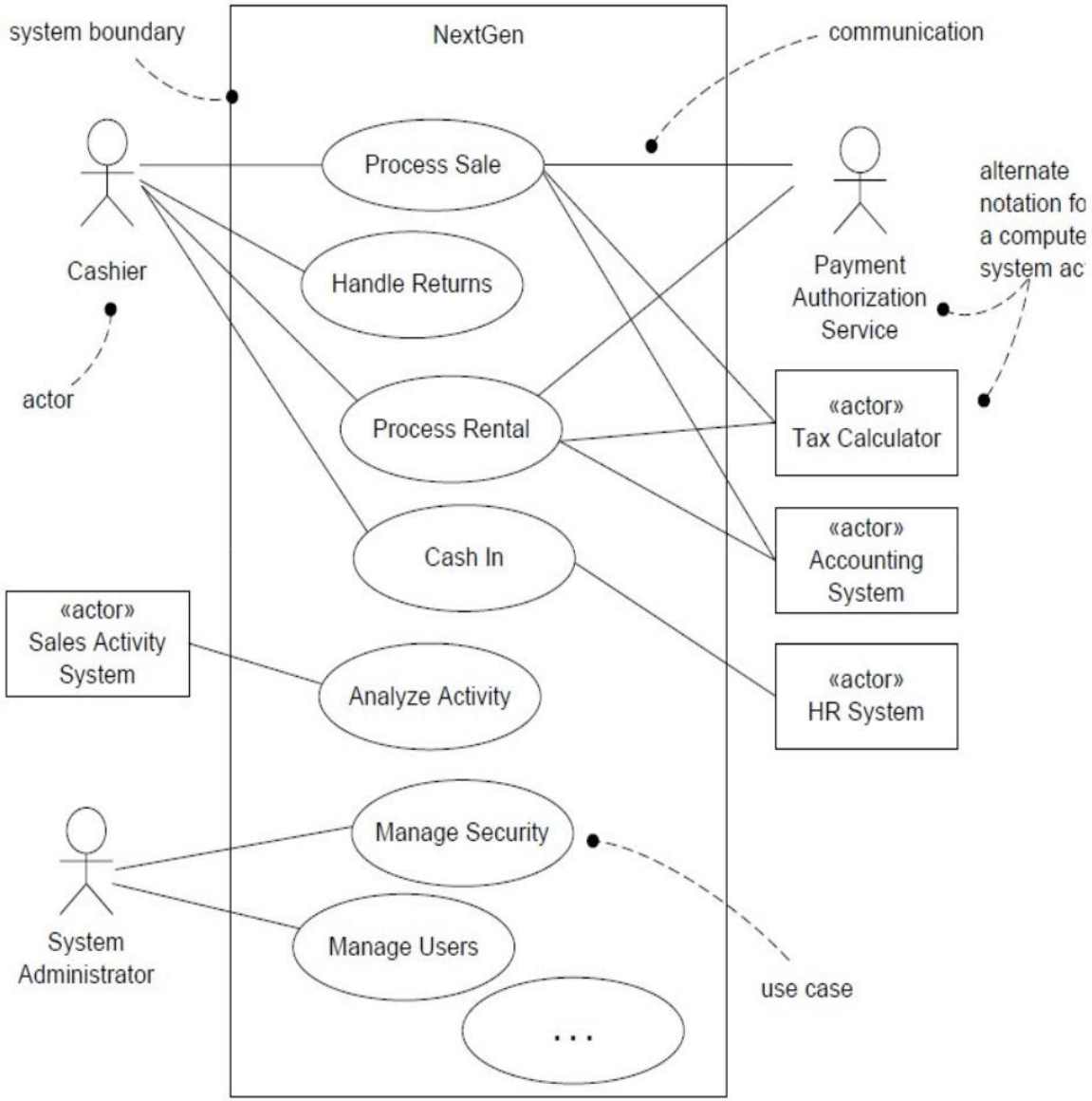


# Actors and Goals via Event Analysis

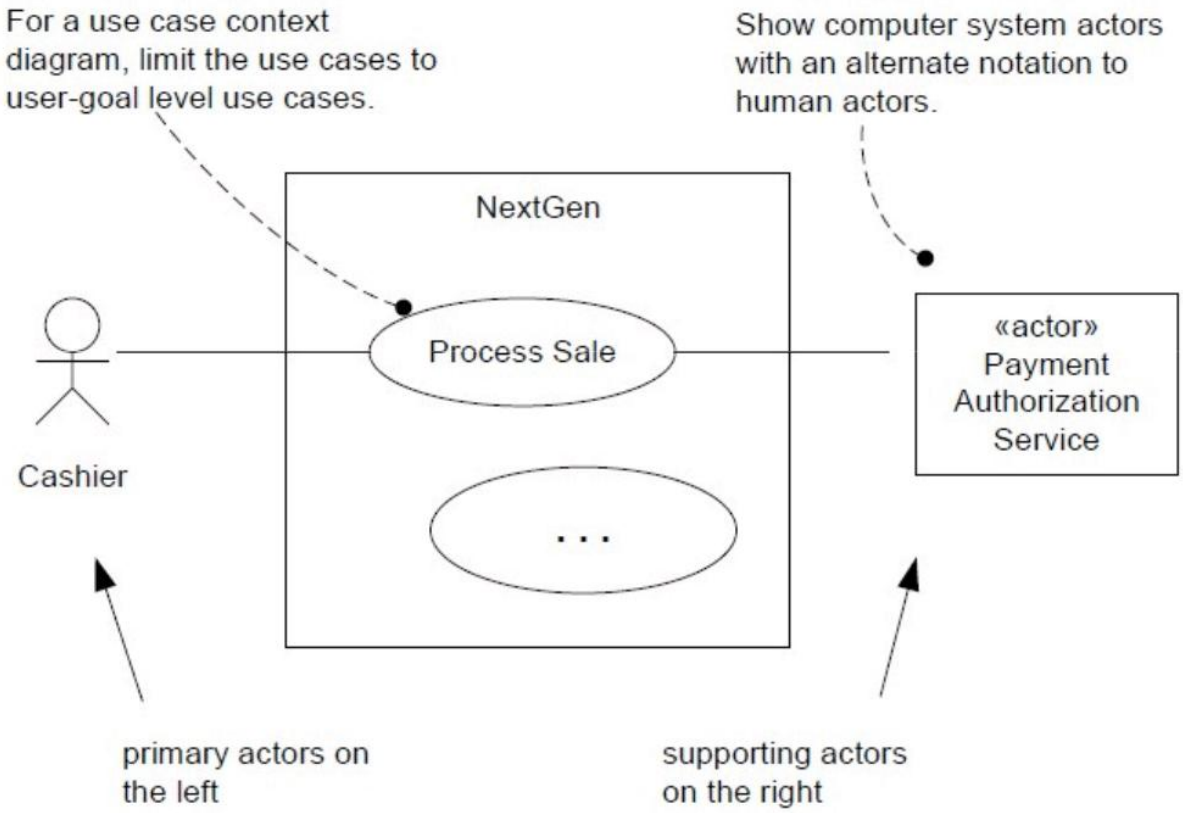
| External Event | From Actor | Goal |
|---|---|---|
| enter sale line item | Cashier | process a sale |
| enter payment | Cashier or Customer | process a sale |
| ... | | |

# Actors

- **Primary actor**—has user goals fulfilled through using services of the SuD. For example, the cashier.
  - Why identify? To find user goals, which drive the use cases.
- **Supporting actor**—provides a service (for example, information) to the SuD. The automated payment authorization service is an example. Often a computer system, but could be an organization or person.
  - Why identify? To clarify external interfaces and protocols.
- **Offstage actor**—has an interest in the behavior of the use case, but is not primary or supporting; for example, a government tax agency.
  - Why identify? To ensure that *all* necessary interests are identified and satisfied. Offstage actor interests are sometimes subtle or easy to miss unless these actors are explicitly named.


system boundary
NextGen
communication
Cashier
Process Sale
Handle Returns
Process Rental
Cash In
Analyze Activity
Manage Security
Manage Users
...
actor
«actor»
Sales Activity
System
System
Administrator
Payment
Authorization
Service
«actor»
Tax Calculator
«actor»
Accounting
System
«actor»
HR System
alternate
notation fo
a compute
system ac
use case

# *Diagramming Suggestions*



# *High-Level System Feature Lists Are Acceptable*

**Summary of System Features**

- sales capture
- payment authorization (credit, debit, check)
- system administration for users, security, code and constants tables, and so on
- automatic offline sales processing when external components fail
- real-time transactions, based on industry standards, with third-party systems, including inventory, accounting, human resources, tax calculators, and payment authorization services
- definition and execution of customized "pluggable" business rules at fixed, common points in the processing scenarios
- …

# Use Cases Are Not Object-Oriented

- use cases are a broadly applicable requirements analysis tool that can be applied to non-object-oriented projects, which increases their usefulness as a requirements method.
- However, as will be explored, use cases are a pivotal input into classic OOA/D activities

| Discipline | Artifact<br>Iteration-> | Incep.<br>Il | Elab.<br>El. .En | Const.<br>CL..Cn | Trans.<br>T1..T2 |
|---|---|---|---|---|---|
| Business Modeling | Domain Model | | s | | |
| Requirements | *Use-Case Model* | s | r | | |
| | Vision | s | r | | |
| | Supplementary Specification | s | r | | |
| | Glossary | s | r | | |
| Design | Design Model | | s | r | |
| | SW Architecture Document | | s | | |
| | Data Model | | s | r | |
| Implementation | Implementation Model | | s | r | r |
| Project Management | SW Development Plan | s | r | r | r |
| Testing | Test Model | | s | r | |
| Environment | Development Case | s | r | | |

# References

- Chapter 6 Applying UML and Patterns by Craig Larman