1. What are advantages of the goal modelling?

- Goal modelling is reasonably intuitive
- Goal modelling captures a static picture, but does not consider the change of goals over time
- Explicit declaration of goals provides sound basis for conflict resolution
- Goal modelling can regress forever up (or down) the goal hierarchy
- None from above

1. 目标建模的优点是什么?

o 目标建模相当直观 o 目标建模捕获静态图片，但不考虑目标随时间的变化 o 明确的目标声明为解决冲突提供了坚实的基础 o 目标建模可以永远向上（或向下）目标层次结构 o上面没有

Goals

- **Approach**
  - Focus on ***why*** a system is required
  - Use goal refinement to arrive at specific requirements
  - Goal analysis
    - document, organize and classify goals
  - Goal hierarchies show **refinements** and **alternatives**
- **Advantages**
  - Reasonably intuitive
  - Explicit declaration of goals provides sound basis for conflict resolution
- **Disadvantages**
  - Captures a static picture - what if goals change over time?
  - Can regress forever up (or down) the goal hierarchy

- **Goals:**
  - Describe functions that must be carried out
- **Actors:**
  - Owners of goals
- **Tips:**
  - Multiple sources - better goals
  - Associate stakeholders with each goal
  - Use scenarios to explore how goals can be met

目标

- 方法
  - 关注为什么需要一个系统
  - 使用目标细化来达到特定要求

  - 目标分析
    - 记录、组织和分类目标
  - 目标层次结构显示了细化和备择方案
- 优点
  - 相当直观
  - 明确的目标声明为解决冲突提供了坚实的基础

- 缺点
  - 捕捉静态图片 - 如果目标随着时间的推移而改变怎么办?
  - 可以在目标层次上永远向上（或向下）回归

- 目标:
  – 描述功能必须进行
- 演员:
  – 目标的所有者
- 尖端:
  – 多个来源 – 更好目标
  – 相关利益相关者每个目标
  – 使用场景进行探索如何实现目标

1. What are advantages of the goal modelling?

- ○ **Goal modelling is reasonably intuitive**
- ○ Goal modelling captures a static picture, but does not consider the change of goals over time
- ○ **Explicit declaration of goals provides sound basis for conflict resolution**
- ○ Goal modelling can regress forever up (or down) the goal hierarchy
- ○ None from above

1. 目标建模的优点是什么?

o 目标建模相当直观
o 目标建模捕获静态图片，但不考虑目标随时间的变化 o 明确的目标声明为冲突提供了良好的基础

解决

o 目标建模可以在目标层次结构中永远向上（或向下）回归 o 从上面没有

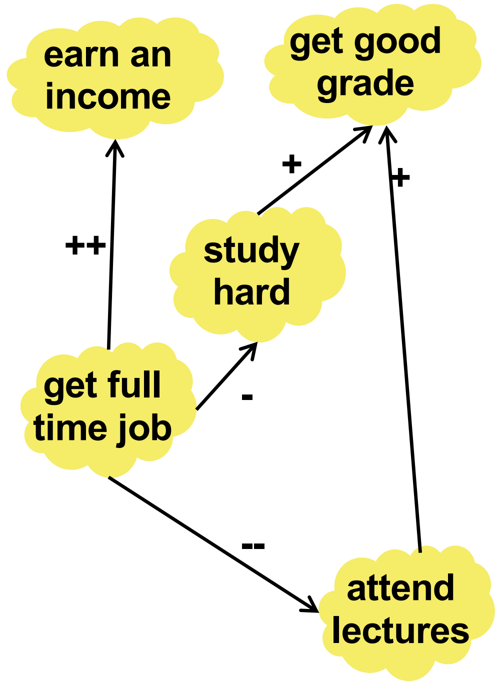
2. How goals are elaborated?

- ○ Using “What” questions to define structure
- ○ Using “Why” questions to explore context
- ○ Using “How” questions to explore operations
- ○ Using “How else” questions to explore alternatives
- ○ None from above

2. 目标是如何制定的？ o 使用“ 什么” 问题来定义结构 o 使用“ 为什么” 问题来探索背景 o 使用“ 如何” 问题来探索操作 o 使用“ 否则如何” 问题来探索替代方案 o 以上都没有

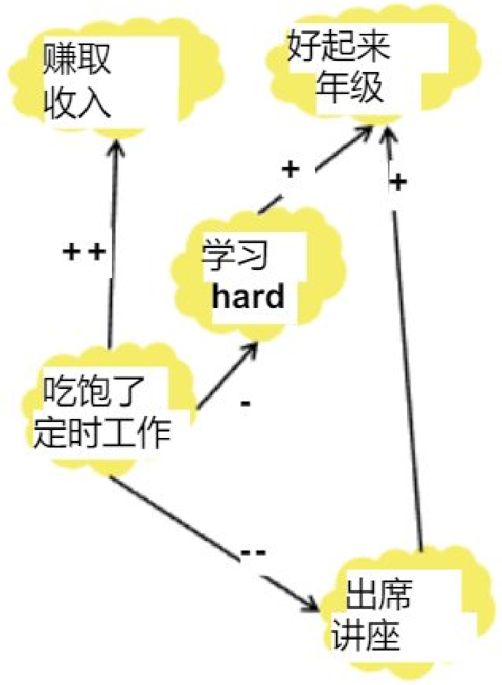
Goal analysis

- **Relationships between goals:**
  - One goal **helps** achieve another (+)
  - One goal **hurts** achievement of another (-)
  - One goal **makes** another (++)
    - Achievement of goal A guarantees achievement of goal B
  - One goal **breaks** another (--)
    - Achievement of goal A prevents achievement of goal B
- **Goal Elaboration:**
  - "**Why**" questions explore higher goals (context)
  - "**How**" questions explore lower goals (operations)
  - "**How else**" questions explore alternatives



目标分析

- 目标之间的关系：
  - 一个目标有助于实现另一个目标 (+)
  - 一个目标会损害另一个目标的实现 (-)
  - 一个目标成就另一个目标 (+ +)
    - 目标A的实现保证了目标B的实现
  - 一球打破另一球 (- -)
    - 目标 A 的实现阻碍了目标 B 的实现
- 目标阐述：
  - “ 为什么” 问题探索更高的目标（背景）
  - “ 如何” 问题探索较低的目标（运营）
  - “ 不然怎么办” 问题探索替代方案



2. How goals are elaborated?

- ○ Using “What” questions to define structure
- ○ **Using “Why” questions to explore context**
- ○ **Using “How” questions to explore operations**
- ○ **Using “How else” questions to explore alternatives**
- ○ None from above

2. 目标是如何制定的？ o 使用“ 什么” 问题来定义结构

o 使用“ 为什么” 问题来探索背景 o 使用“ 如何” 问题来探索
操作 o 使用“ 否则如何” 问题来探索替代方案 o 以上都没有

3. How are KAOS goals, which are effectively assigned to software agent, called?

- o Expectation
- o Requirement
- o Software agent goals
- o Domain properties
- o None from above

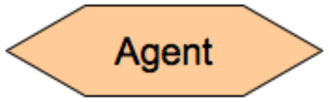
3. 有效分配给软件代理的 KAOS 目标是如何调用的?

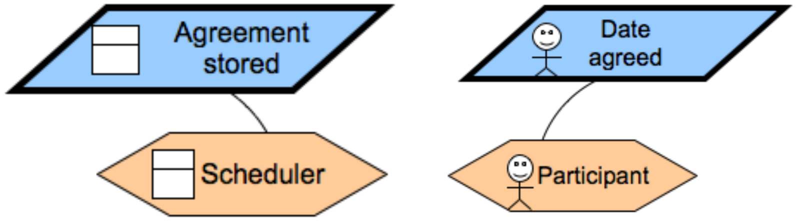
Ø 期望
Ø 要求
o 软件代理目标
o 域属性 o 上面没有

KAOS

Constructs of Goal and Agent models

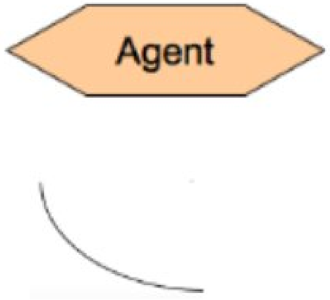
- **Agent**
  - Active object which plays a specific role towards goal achievement by monitoring or controlling specific object behavior
- **Assignment**
  - A possible assignment of a goal to an agent
  - **Responsibility** – an actual assignment of a goal to an agent
- A goal effectively assigned to
  - A **software agent** is called **requirement**
  - An **environment agent** is called **expectation**



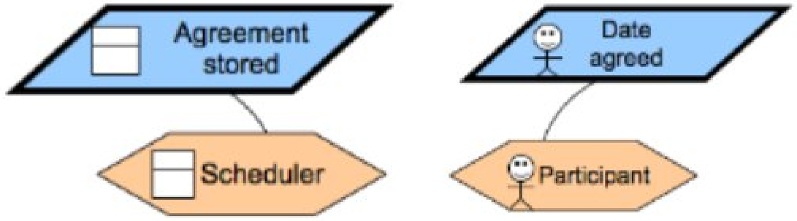


KAOS
目标和代理模型的构造

- 代理人
  - 主动对象通过监视或控制特定对象行为来实现目标的特定作用



- 任务
  - 可能将目标分配给代理
  - 责任——将目标实际分配给代理人
- 有效分配的目标
  - 软件代理称为需求
  - 环境代理称为期望



3. How are KAOS goals, which are effectively assigned to software agent, called?

- Expectation
- **Requirement**
- Software agent goals
- Domain properties
- None from above

3. 有效分配给软件代理的 KAOS 目标是如何调用的?

Ø 期望
Ø 要求
o 软件代理目标
o 域属性 o 上面没有

4. What are exploratory scenarios?

- o Scenarios for understanding the process operations, involved agents, triggering events, and other
- o Scenarios for exploring and evaluating possible, alternative solutions to support the selection of one alternative solution
- o Scenarios for explaining a goal, an alternative solution or a sequence of interactions
- o Scenarios for misusing the functions of the considering system
- o None form above

4. 什么是探索性场景?

o 用于理解流程操作、涉及的代理、触发事件和其他的场景 o 用于探索和评估可能的替代解决方案以支持选择一个替代解决方案的场景 o 用于解释目标、替代解决方案或一系列交互的场景 o滥用所考虑系统的功能的场景 o 以上没有任何形式

Scenario Types

- Current state and desired state
- Positive and negative
- Misuse scenarios
- **Descriptive, exploratory and explanatory**
  - Understanding the process operations, involved agents, triggering events, and other
    - **Illustrate meaning of goals and requirements**
  - Explore and evaluate possible, alternative solutions in order to support the selection of one alternative solution
    - **Support decision making**
  - Aims explaining a goal, an alternative solution or a sequence of interactions
    - **Explain complex facts**

场景类型

- 替代解决方案或一系列交互 o 滥用所考虑系统功能的场景 o 以上没有任何形式
当前状态和期望状态
- 正面及负面
- 误用场景
- 描述性、探索性和解释性
  - 了解流程操作、涉及的代理、触发事件等
    - 说明目标和要求的含义
  - 探索并评估可能的替代解决方案，以支持选择一种替代解决方案
    - 支持决策
  - 旨在解释目标、替代解决方案或一系列交互
    - 解释复杂的事实

4. What are exploratory scenarios?

- Scenarios for understanding the process operations, involved agents, triggering events, and other
- **Scenarios for exploring and evaluating possible, alternative solutions to support the selection of one alternative solution**
- Scenarios for explaining a goal, an alternative solution or a sequence of interactions
- Scenarios for misusing the functions of the considering system
- None form above

4. 什么是探索性场景?

o 用于理解流程操作、涉及的代理、触发事件和其他的场景 o 用于探索和评估可能的替代解决方案的场景

支持选择一种替代解决方案
o 解释目标、替代解决方案或一系列交互的场景 o 滥用所考虑系统功能的场景 o 以上没有任何形式

5. How use case must be documented?

- You should prepare a “flow of events”
- You should always break down the function to misuses and malicious use cases
- You should document from an actor’s point of view
- You should describe what the system must provide to the actor when the use case is executed
- None from above

5. 必须如何记录用例?

o 您应该准备“ 事件流程”
o 您应该始终将功能分解为误用和恶意用例 o 您应该从参与者的角度进行记录 o 您应该描述执行用例时系统必须向参与者提供什么 o 上面没有

Documenting Use Cases

- **For each use case:**
  - prepare a “flow of events”
  - document from an actor’s point of view
  - describe what the system must provide to the actor when the use case is executed
- **Typical contents**
  - How the use case starts and ends
  - Normal flow of events
  - Alternate flow of events
  - Exceptional flow of events
- **Documentation style**
  - Textual use case description
  - Sequence diagrams

记录用例

- 对于每个用例：
  - 准备“ 事件流程”
  - 从演员的角度来看的文件
  - 描述执行用例时系统必须向参与者提供什么

- 典型内容
  - 用例如何开始和结束
  - 事件的正常流程
  - 交替的事件流程
  - 异常的事件流程
- 文档风格
  - 文本用例描述
  - 序列图

5. How use case must be documented?

- **You should prepare a “flow of events”**
- You should always break down the function to misuses and malicious use cases
- **You should document from an actor’s point of view**
- **You should describe what the system must provide to the actor when the use case is executed**
- None from above

5. 必须如何记录用例?

o 您应该准备“ 事件流程”
o 您应该始终将功能分解为误用和恶意用例 **o** 您应该从参与者的角度进行记录 **o** 您应该描述执行用例时系统必须向参与者提供什么 **o** 上面没有

6. Which statements are correct?

- ○ Goals initiate and influence the definition of scenarios
- ○ Goals classify scenarios
- ○ Scenarios explain if and why a new software intensive system is required
- ○ Scenarios illustrate satisfaction of goals
- ○ Scenarios lead to the identification of new goals

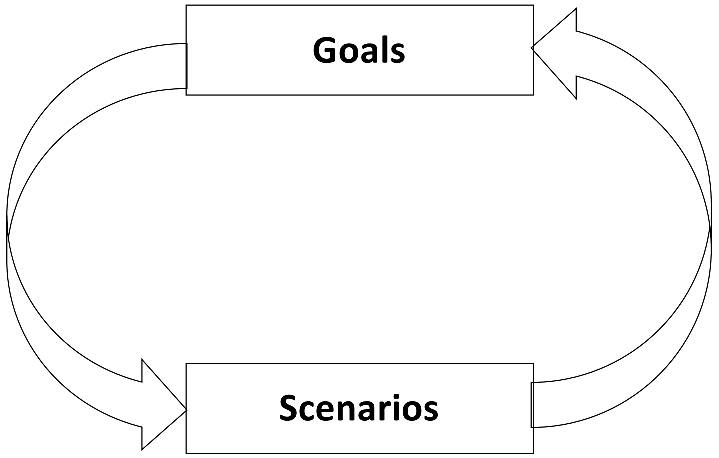
6. 哪些说法是正确的?

o 目标启动并影响场景的定义 o 目标对场景进行分类 o 场景解释是否需要以及为何需要新的软件密集型系统 o 场景说明目标的满足情况 o 场景导致新目标的识别

Goal-Scenario coupling

… initiate and influence the definition of …

… classify …

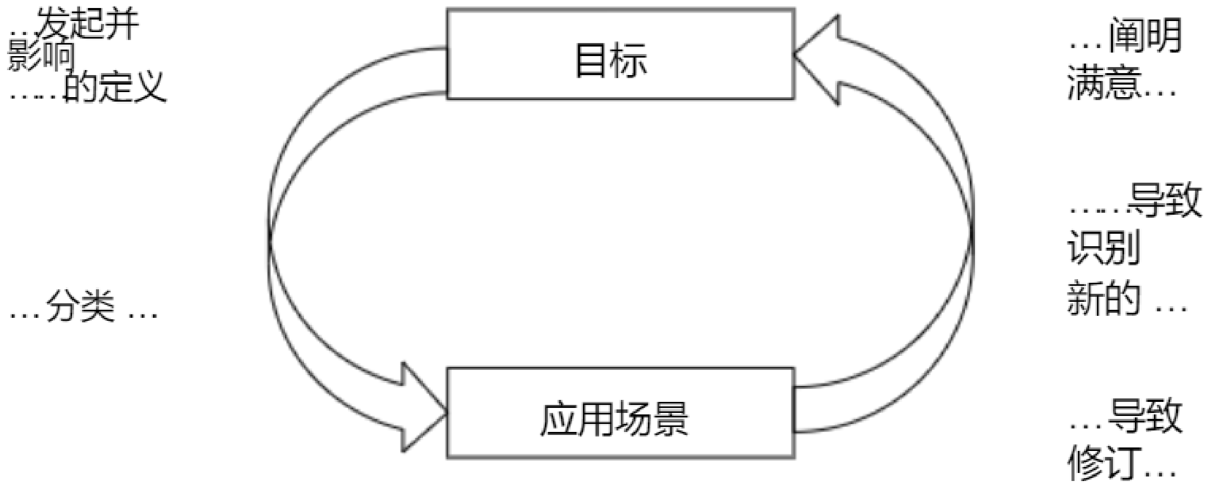


… illustrate satisfaction…

… lead to the identification of new …

… lead to revision of …

目标-场景耦合



6. Which statements are correct?

- ○ **Goals initiate and influence the definition of scenarios**
- ○ **Goals classify scenarios**
- ○ Scenarios explain if and why a new software intensive system is required
- ○ **Scenarios illustrate satisfaction of goals**
- ○ **Scenarios lead to the identification of new goals**

6. 哪些说法是正确的?

o 目标发起并影响场景的定义 **o** 目标对场景进行分类

o 场景解释是否以及为何需要新的软件密集型系统

o 情景说明目标的满足情况 **o** 情景导致新目标的确定

Task 1

Diagram 1: What are the social relationships between the stakeholders? To support your answer, create a **strategic dependency model** (using the *i** modelling language), where the social viewpoint of the given case is illustrated.

Diagram 2: Create **strategic rationale model** to illustrate what Owner and Renter should do to fulfil the social relationships.

任务1

图1：利益相关者之间的社会关系是什么？为了支持您的答案，请创建一个战略依赖模型（使用 i* 建模语言），其中说明了给定案例的社会观点。

图 2：创建战略原理模型来说明业主和承租人应该做什么来履行社会关系。

Task 1

Diagram 1: What are the social relationships between the stakeholders? To support your answer, create a **strategic dependency model** (using the *i** modelling language), where the social viewpoint of the given case is illustrated.

NOT social-technical!!!

Diagram 2: Create **strategic rationale model** to illustrate what Owner and Renter should do to fulfil the social relationships.

任务1

图1：利益相关者之间的社会关系是什么？为了支持您的答案，请创建一个战略依赖模型（使用 i* 建模语言），其中说明了给定案例的社会观点。

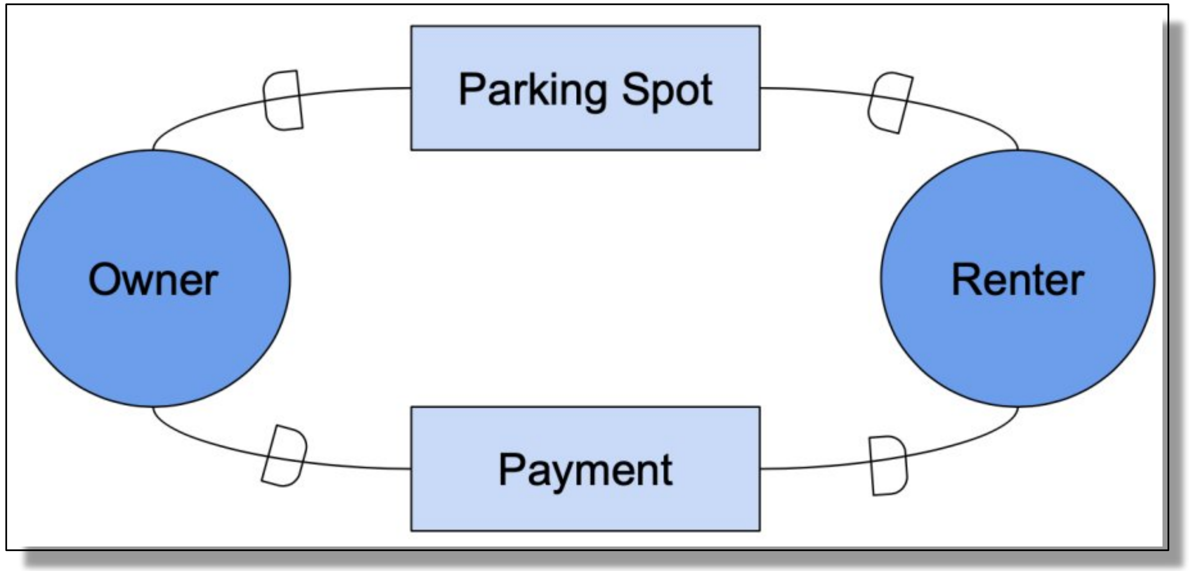
图 2：创建战略原理模型来说明业主和承租人应该做什么来履行社会关系。

不是社会技术!

Stakeholders and their dependencies

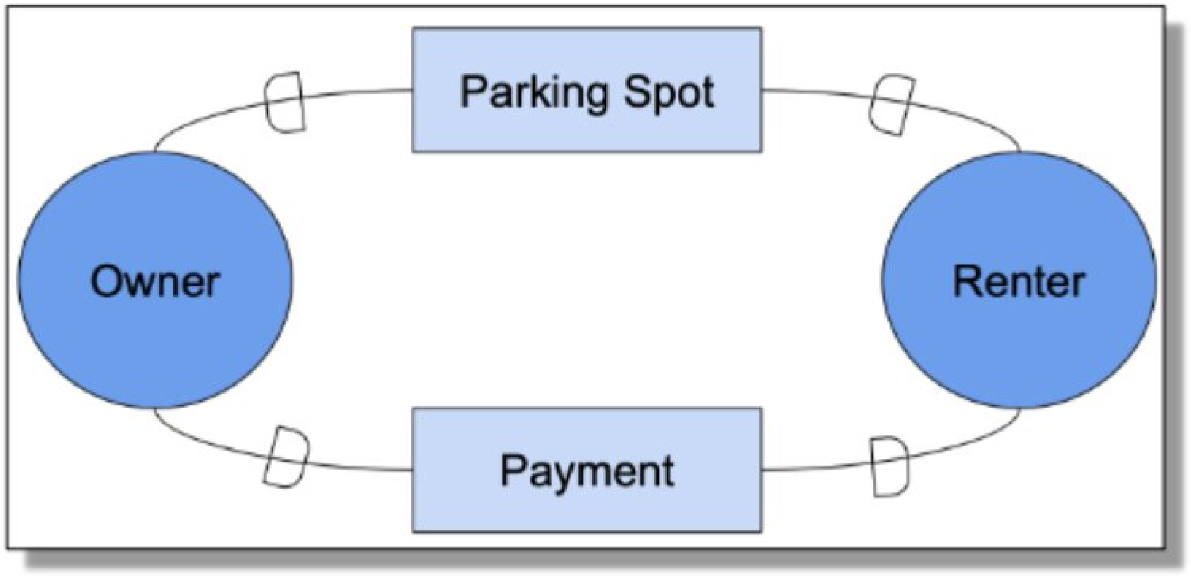
- **Owner** – owner of the parking spot

- **Renter** – user of the parking spot



利益相关者及其依赖关系

- 所有者 - 停车位的所有者
- 承租人 – 停车位的使用者



Task 1

Diagram 1: What are the social relationships between the stakeholders? To support your answer, create a **strategic dependency model** (using the *i** modelling language), where the social viewpoint of the given case is illustrated.

Diagram 2: Create **strategic rationale model** to illustrate what Owner and Renter should do to fulfil the social relationships.

任务1

图1：利益相关者之间的社会关系是什么？为了支持您的答案，请创建一个战略依赖模型（使用 i* 建模语言），其中说明了给定案例的社会观点。

图 2：创建战略原理模型来说明业主和承租人应该做什么来履行社会关系。

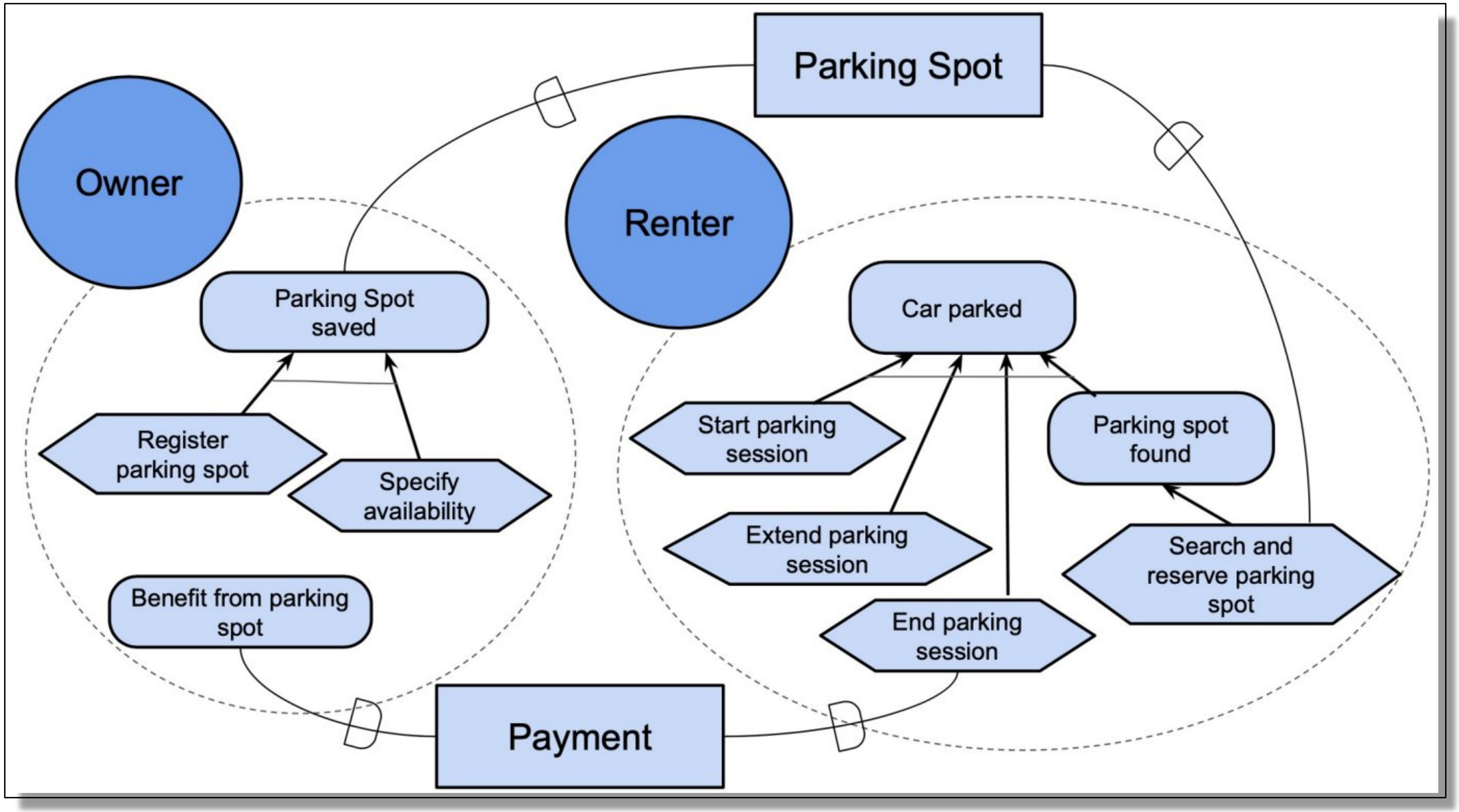
Stakeholders and their dependencies

- **Owner** – owner of the parking spot
  - Register parking spot and specify its available
  - Received payment
- **Renter** – user of the parking spot
  - Use the parking spot;
  - Search and reserve parking spot
  - Start/end parking session
  - Extend parking session

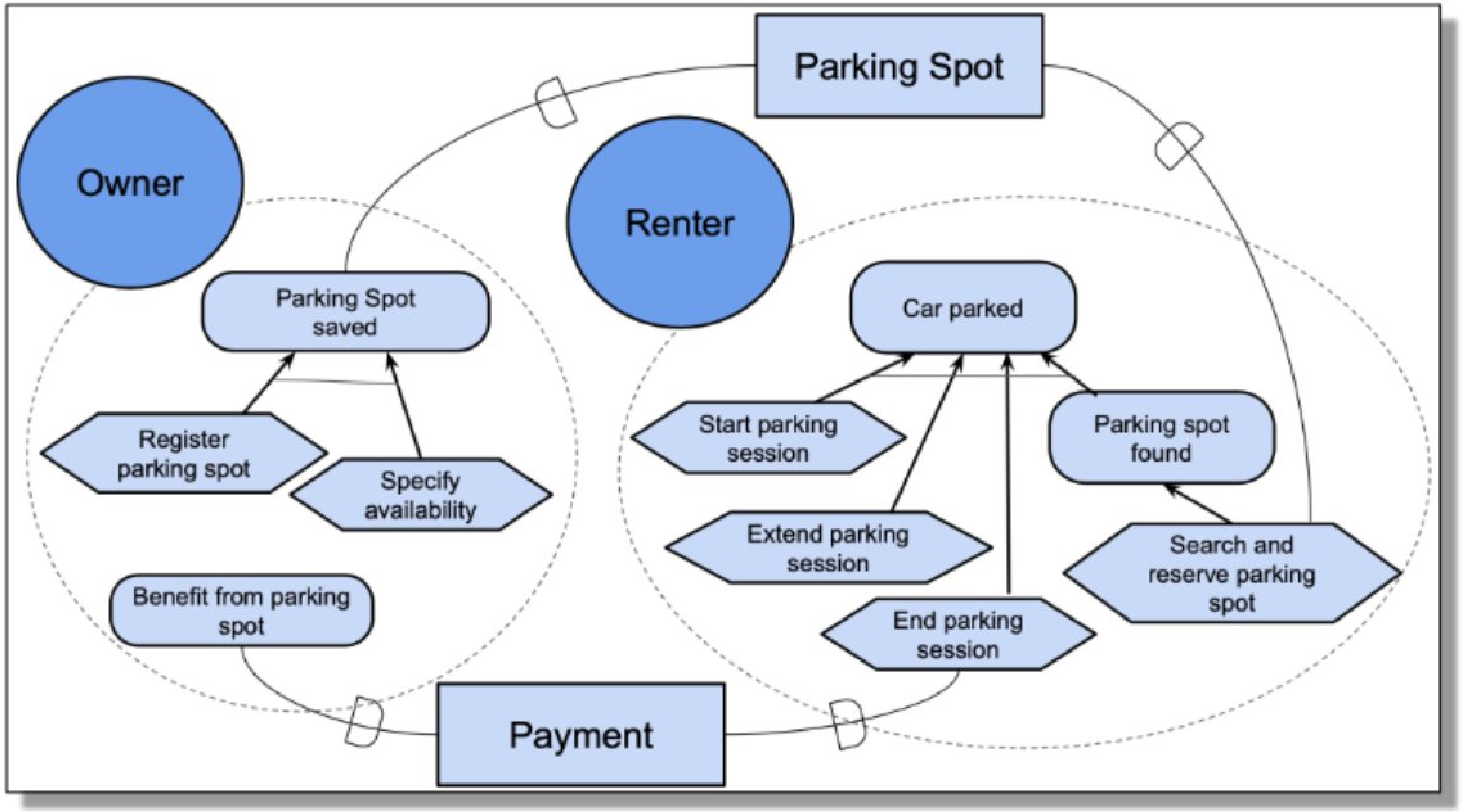
利益相关者及其依赖关系

- 所有者 - 停车位的所有者
  - 注册停车位并指定其可用
  - 已收到付款
- 承租人 – 停车位的使用者
  - 使用停车位;
  - 搜索并预订停车位
  - 开始/结束停车时段
  - 延长停车时间

Stakeholders and their dependencies



利益相关者及其依赖关系

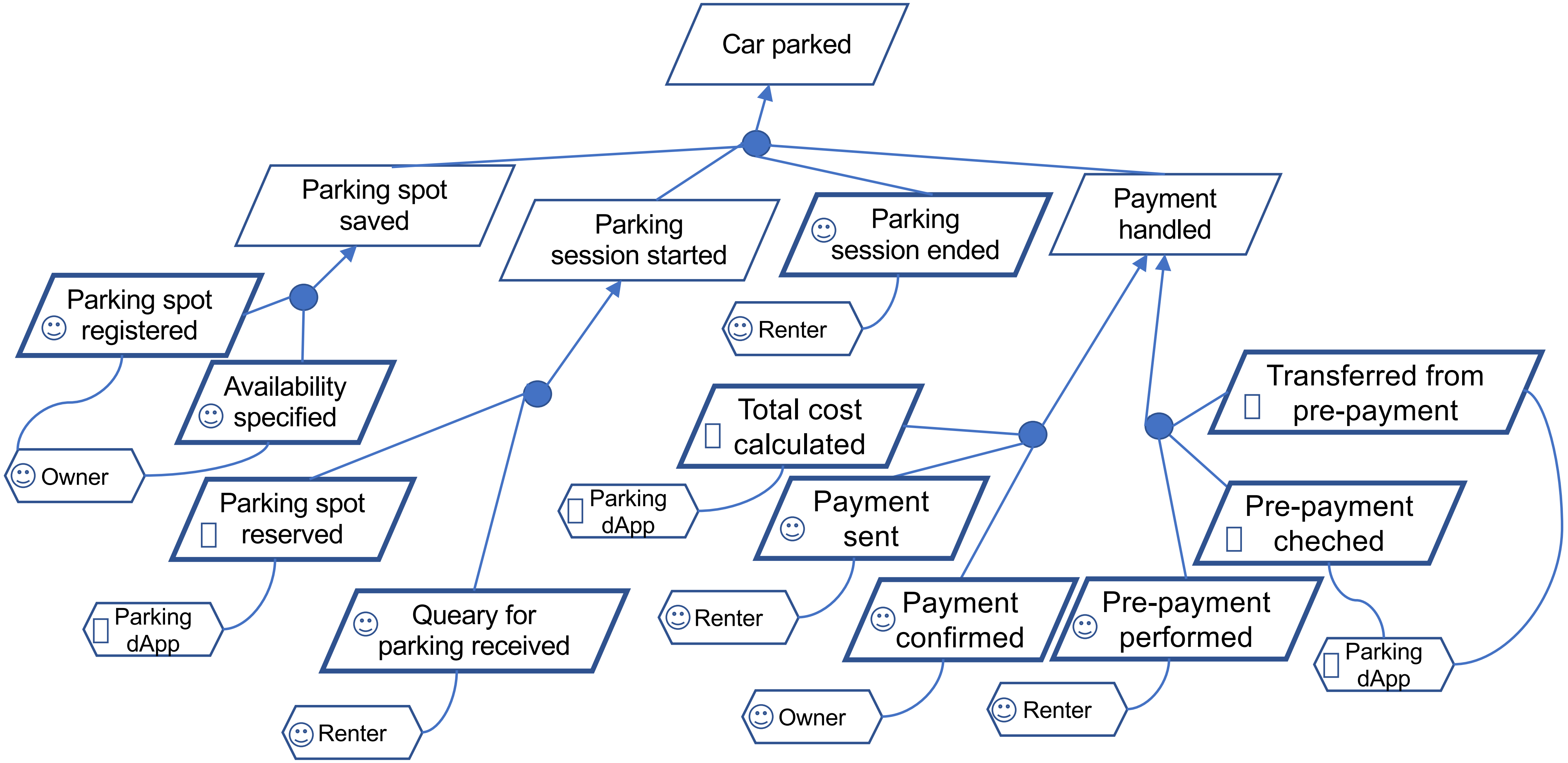


Task 2

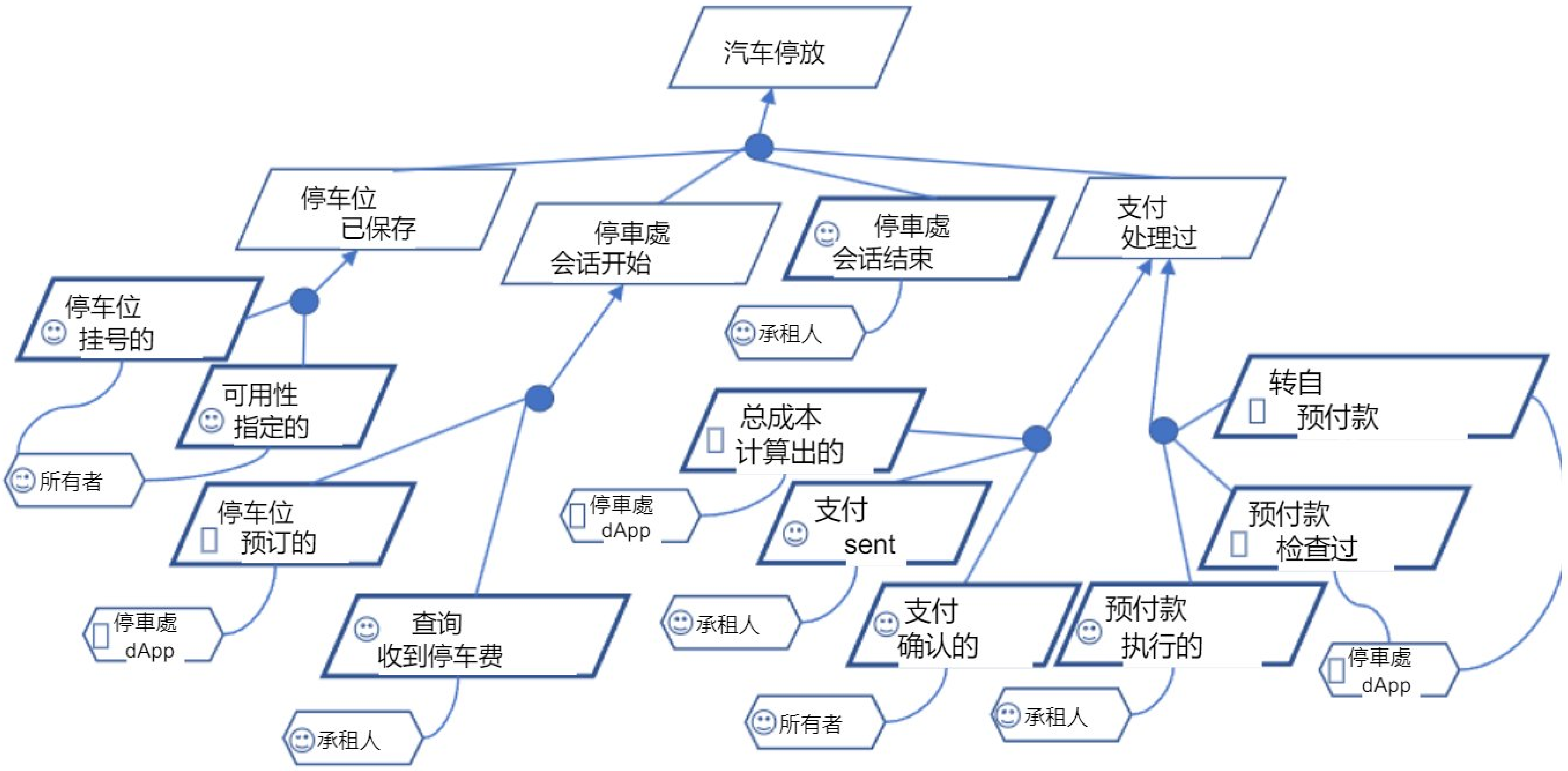
Use **KAOS modelling languages** and refine one goal to the goal hierarchy (containing at least 3 hierarchy levels and including at least 2 alternative refinements). Your model should separate between requirements and expectations.

任务2

使用 KAOS 建模语言，将一个目标细化为目标层次结构（包含至少 3 个层次结构级别，并包括至少 2 个替代细化）。您的模型应该区分需求和期望。



Car parked
Parking spot saved
Parking session started
Parking session ended
Payment handled
Parking spot registered
Availability specified
Owner
Parking spot reserved
Parking dApp
Queary for parking received
Renter
Renter
Total cost calculated
Parking dApp
Payment sent
Renter
Payment confirmed
Owner
Transferred from pre-payment
Pre-payment cheched
Pre-payment performed
Renter
Parking dApp


汽车停放
停车位 已保存
停車處 会话开始
停車處 会话结束
支付 处理过
停车位 挂号的
可用性 指定的
所有者
停车位 预订的
停車處 dApp
查询 收到停车费
承租人
承租人
总成本 计算出的
停車處 dApp
支付 sent
承租人
转自 预付款
预付款 检查过
支付 确认的
预付款 执行的
所有者
承租人
停車處 dApp

Task 3:

Create a **use case diagram** to illustrate functions of the Parking dApp. Diagram must include at least 5 use cases. One of these use cases should be named “Handle payment”.

任务3：

创建用例图来说明 Parking dApp 的功能。图表必须包含至少 5 个用例。这些用例之一应命名为“ 处理付款”。

Stakeholders and their dependencies

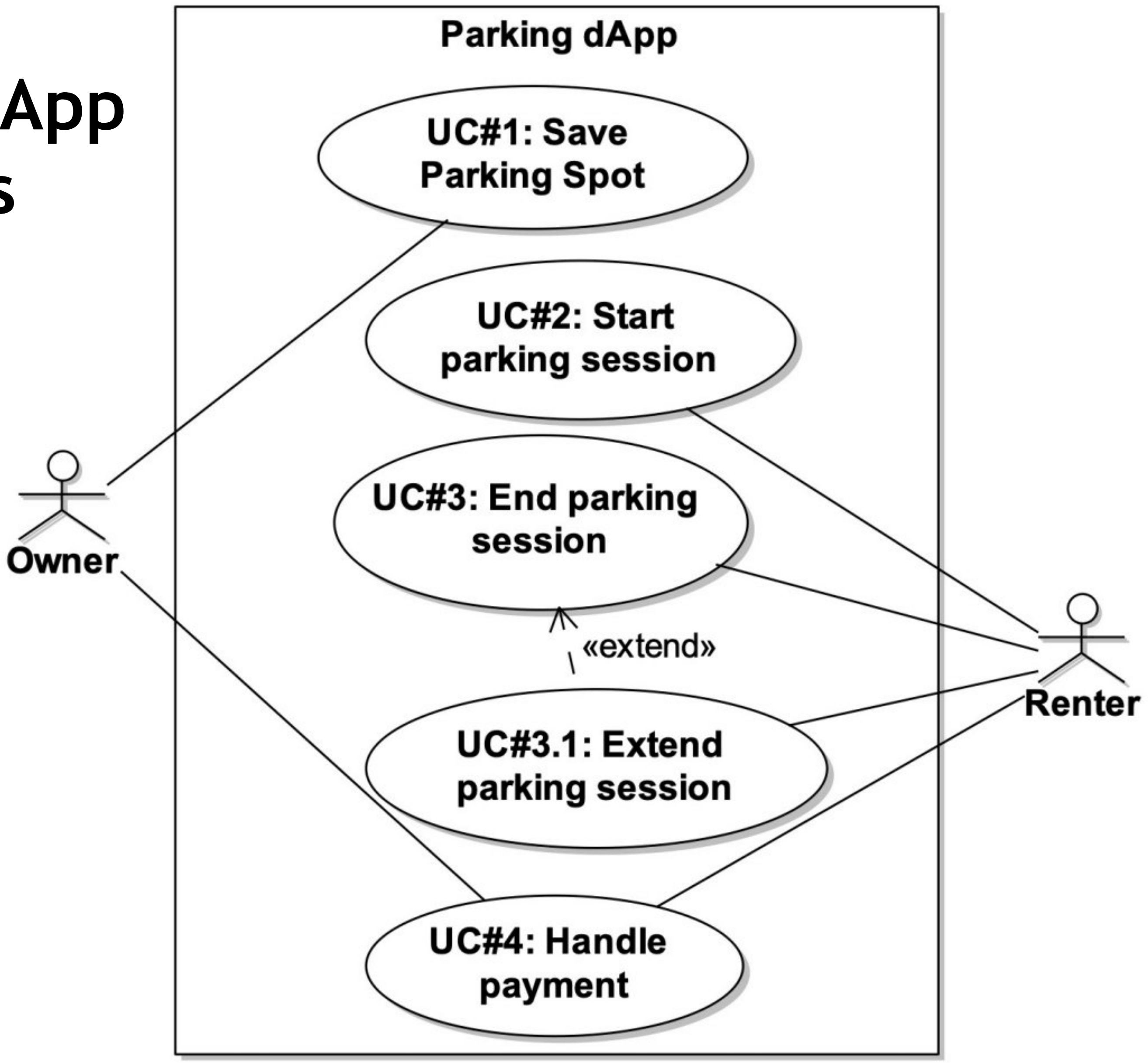
- **Owner** – owner of the parking spot
  - Register parking spot and specify its available
  - Received payment

- **Renter** – user of the parking spot
  - Use the parking spot;
  - Search and reserve parking spot
  - Start/end parking session
  - Extend parking session

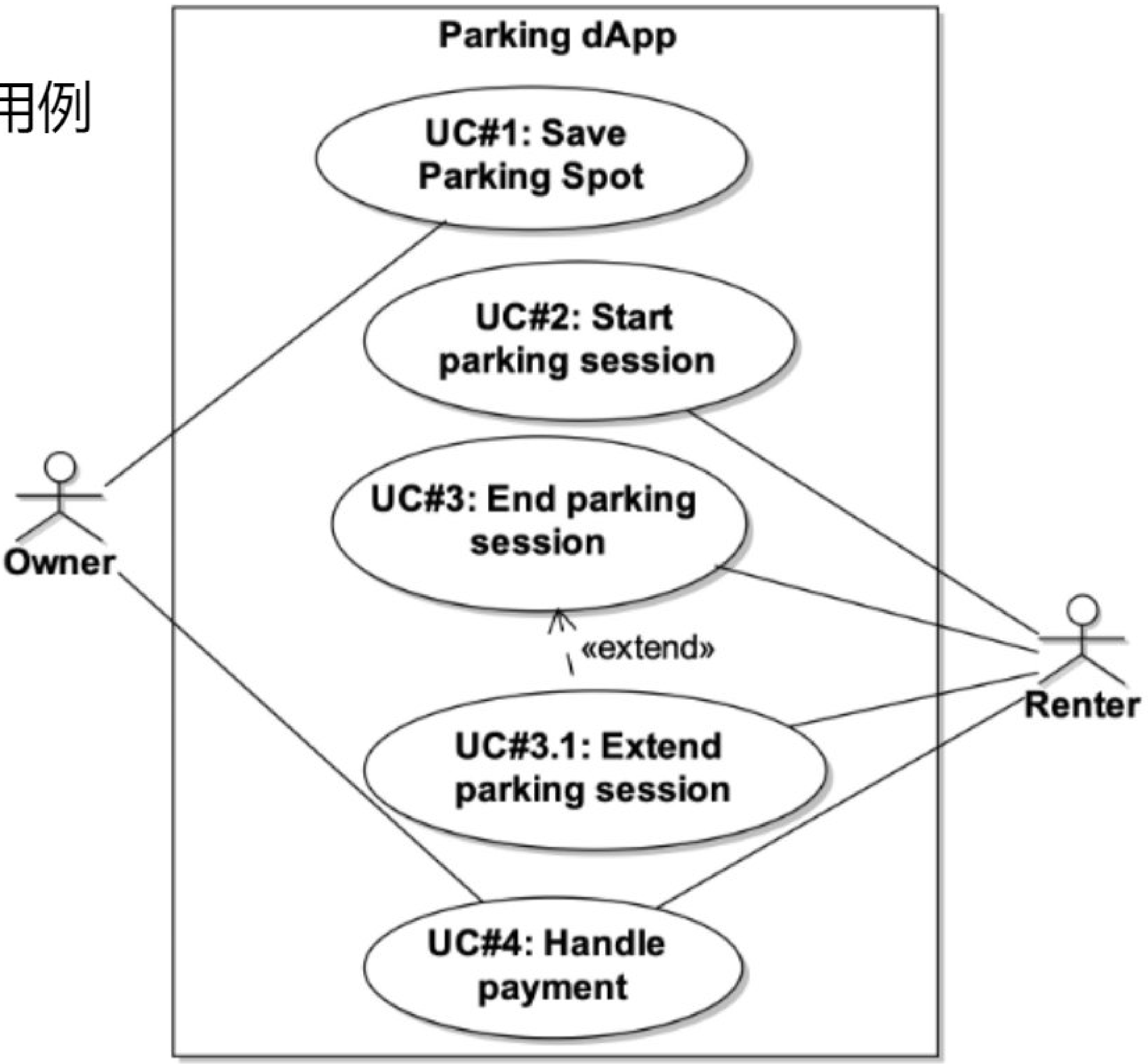
利益相关者及其依赖关系

- 所有者 - 停车位的所有者
  - 注册停车位并指定其可用
  - 已收到付款
- 承租人 - 停车位的使用者
  - 使用停车位;
  - 搜索并预订停车位
  - 开始/结束停车时段
  - 延长停车时间

Parking dApp Use Cases


Parking dApp
UC#1: Save Parking Spot
UC#2: Start parking session
UC#3: End parking session
«extend»
UC#3.1: Extend parking session
UC#4: Handle payment
Owner
Renter

停车 dApp 用例


Parking dApp
UC#1: Save Parking Spot
UC#2: Start parking session
UC#3: End parking session
«extend»
UC#3.1: Extend parking session
UC#4: Handle payment
Owner
Renter

Task 4

Refine “Handle payment” use case in the use case template.

任务4

在用例模板中细化“ 处理付款” 用例。

UC4: Payment handled

Use case ID: name:	UC#4: Payment handled
Date created:	01.01.2019
Actors:	Owner, Renter
Description:	Once the renter has finished using the parking spot he pays to the owner.
Trigger:	Renter has finished the parking and want to leave, but first he has to pay for parking.
Precondition:	Parking session has ended.
Postcondition:	Payment completed.
Main flow:	1. Parking dApp calculates the total cost. 2. Renter pays for parking (spot). 3. Owner confirms payment.
Alternative flow:	none
Priority:	Must
Assumptions:	Owner has banking account.

UC4：付款已处理

用例 ID：名称：UC #4：	已处理付款
创建日期:	01.01.2019
参与者：业主、承租人 描述：一旦承租人使用完停车位，他就向业主付款。	
触发点：	承租人已完成停车并想要离开，但首先他必须支付停车费。
前提条件：	停车时段已结束。
后置条件：	付款完成。
主要流程：	1. 停车dApp 计算总费用。 2. 承租人支付停车费（停车位）。 3. 业主确认付款。
替代流程：	无
优先级：	必须
假设：	所有者有银行账户。