需求分析

1601214481 酋妍

前情提要

以供电企业管理系统为例

准备工作

- 涉众分析

获取需求

- 定义边界
- 发现主角
- 业务建模
- 领域建模
- 非功能性需求

需求分析

系统分析

需求分析

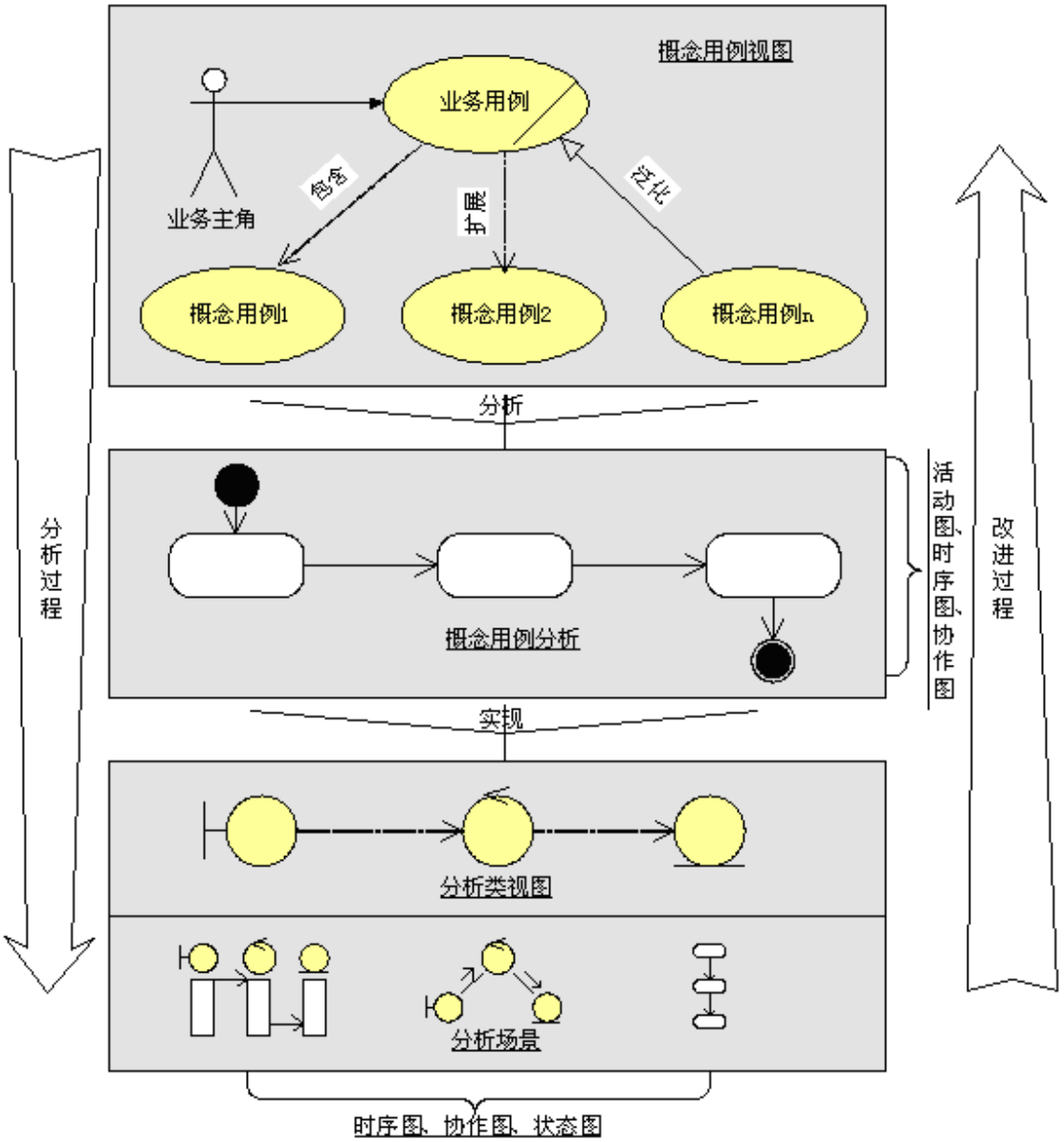
- 关键概念分析
  - 获取概念用况
  - 分析概念用况
  - 建立概念模型
- 业务架构
  - 获得小的构件
  - 构件之间的依赖关系
- 系统原型

关键概念分析

建立概念模型

- 概念模型始于业务用况，从业务模型中抽象出一些概念用况，针对概念用况进行分析，得到一些分析类和分析场景。

- 概念模型是针对需求中的关键业务，所以
  1. 需求人员已经把握了需求
  2. 概念模型需要精准非全面
- 需求与系统的桥梁
- 以下以供电企业管理系统为例

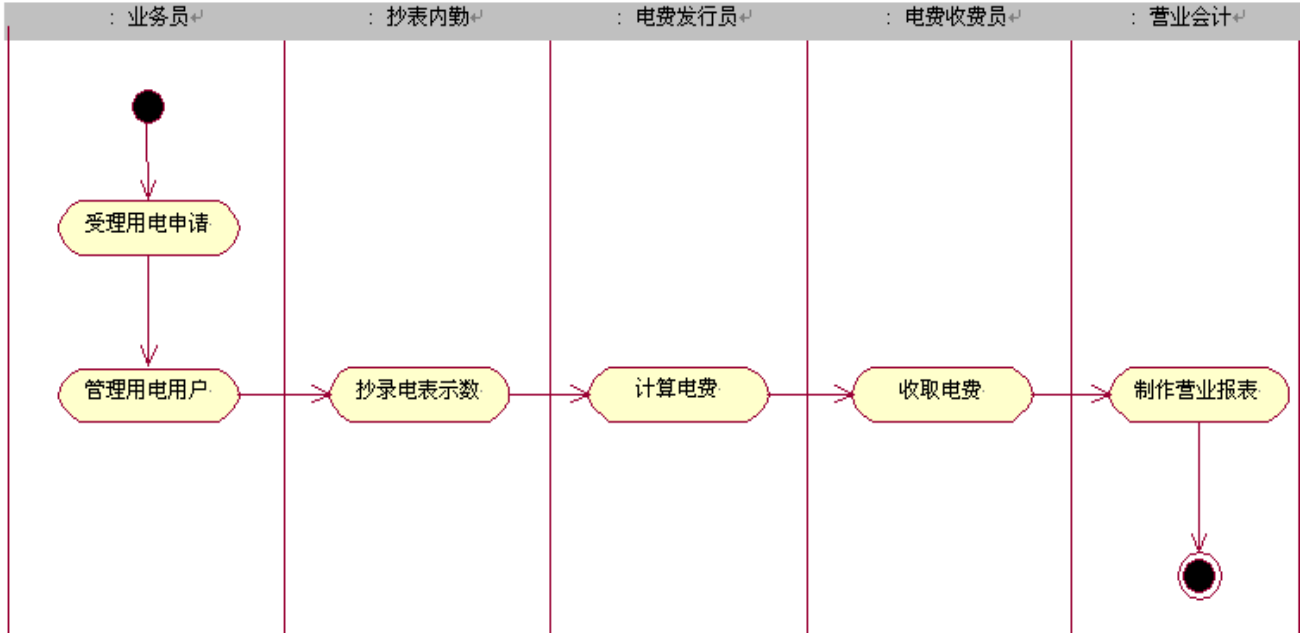


从业务用况到分析类是分析过程，在分析过程中可以向上追溯，则是改进过程。

获取概念用况

- 供电企业业务的主线=核心业务

- 被挑选出来作为关键概念分析的业务用况
- 获得概念用况的目的是为了完成业务主线，因此不需要展现所有的业务用况细节，只需概括展现能够完成业务主线的部分。

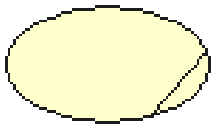


很多业务用况都会跟主线有关，但如果有关都挑出来就会全部被挑出来，所以只需要视情况挑选1-2有代表性的业务用况就可以。

eg.永久用电和临时用电→永久，因为最常用且典型。

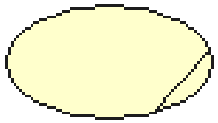
eg.抄录电表示数→有关的业务用况有:导出上月示数/分配抄表任务/录入抄表示数。录如抄表最具有典型性，因为不管抄表有多少业务用况，录入才是支持电费计算的最终手段

关键业务用例



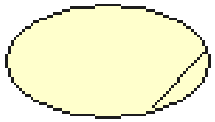
bu_申请永久用电

(from 用电客户服务)



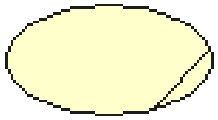
录入本月抄表示数

(from 营业财务)



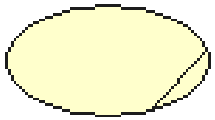
制作营业报表

(from 营业财务)



计算电费

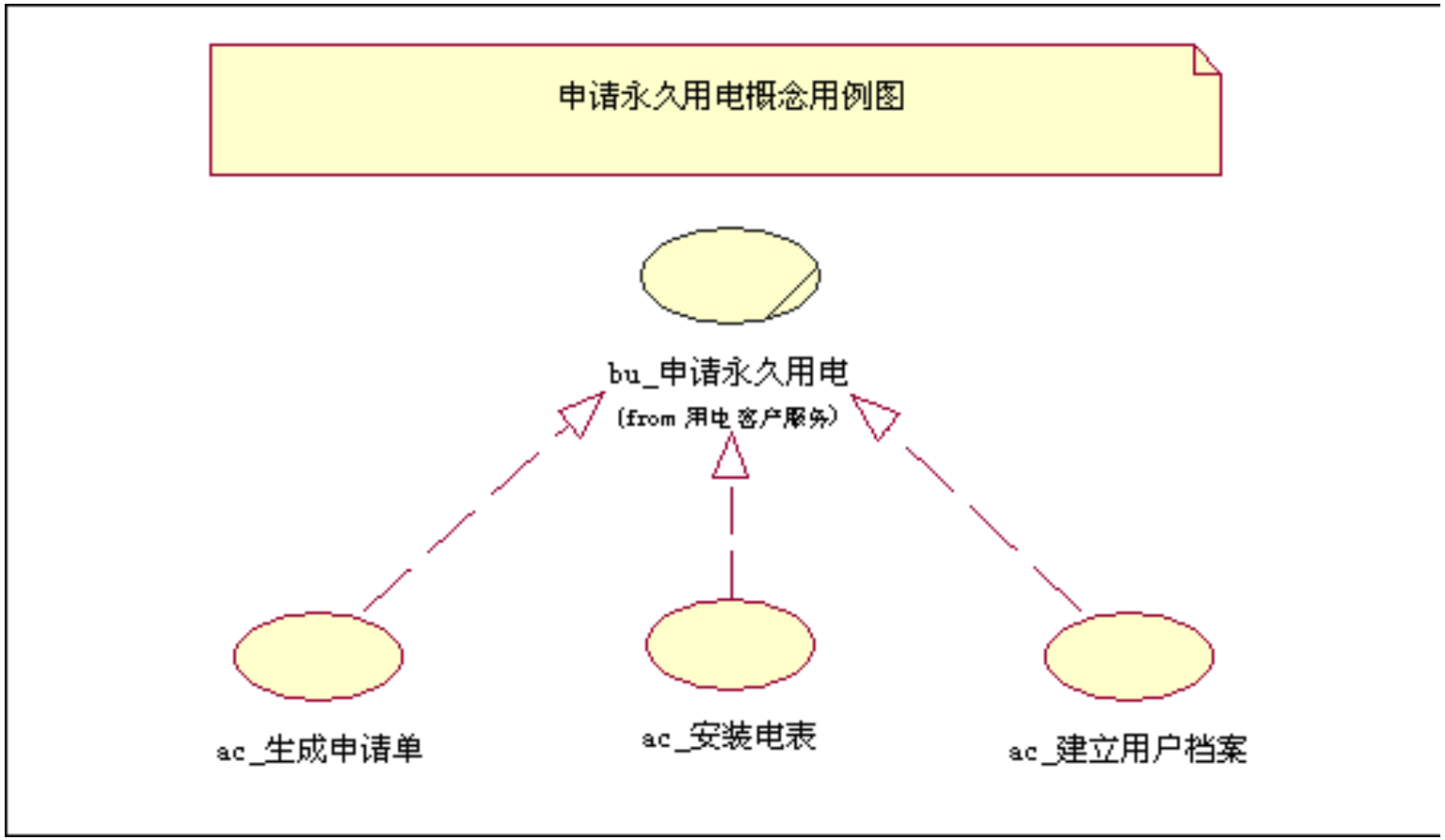
(from 营业财务)



收取电费

(from 营业财务)

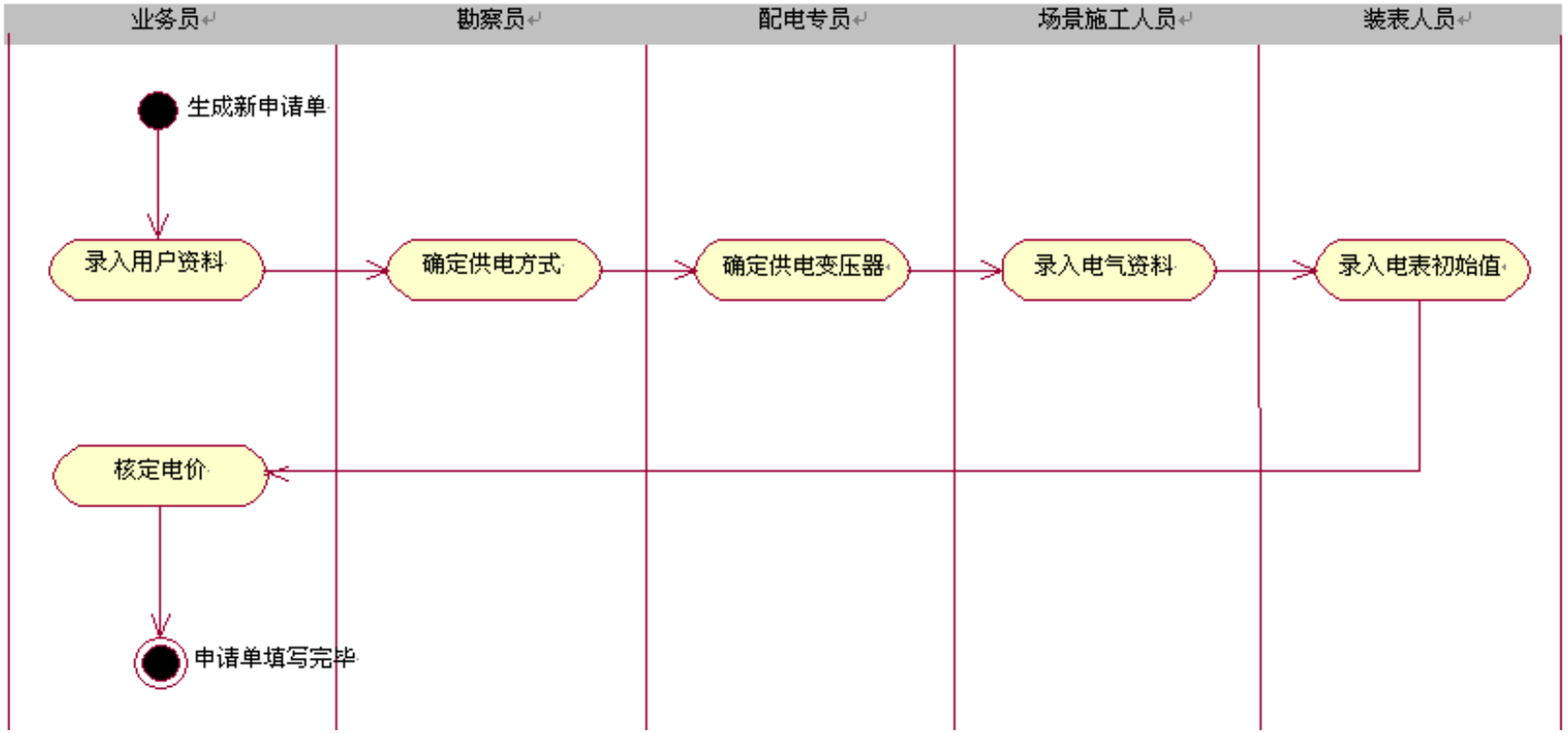
- 只有这三项与业务主线关系最紧密，只要三项齐备业务主线就能够执行。
- 其他业务用况以同样的方式绘製概念用况图。



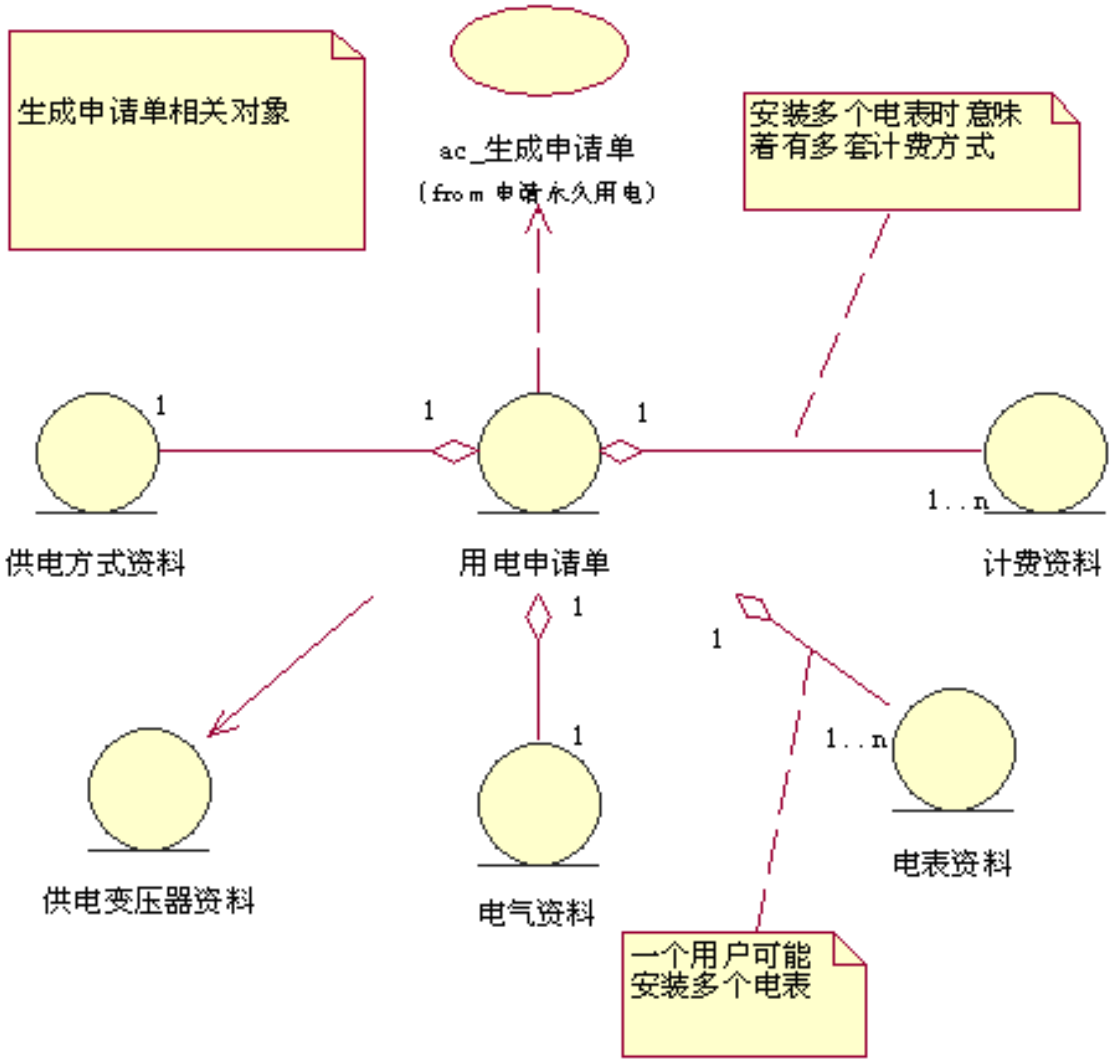
分析概念用况

- 分析概念用况，分析过程与获取需求时业务用况建模的过程一样。
  - 业务用况建模的过程：
  - 业务用况场景图→业务用况规约→业务对象模型实现视图
- 已经不再关注在业务上，而用系统/抽象角度来分析概念用况。

- (针对ac_生成申请单概念用况绘製的场景图)
- 整个概念场景围绕申请单如何从产生到完成的活动建立，活动构成了申请单的生成过程，从此场景可以得到概念用况的**关键对象**。



依刚刚获得的概念对象得到的对象图(概念用况场景示例)

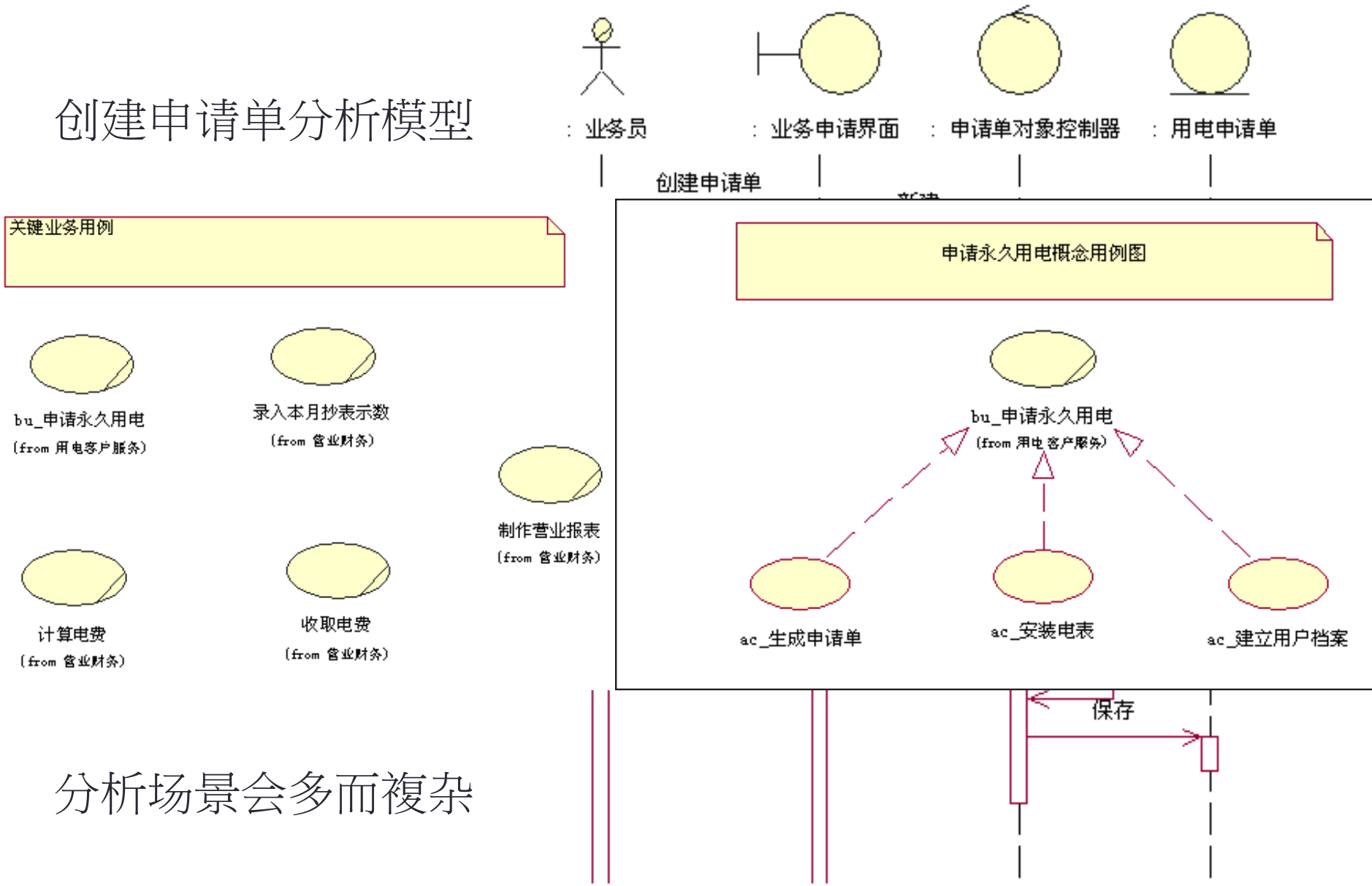


建立概念模型

- 关键对象就是建立概念模型的基础，用这些关键对象去实现业务主线。
- 关键概念分析：

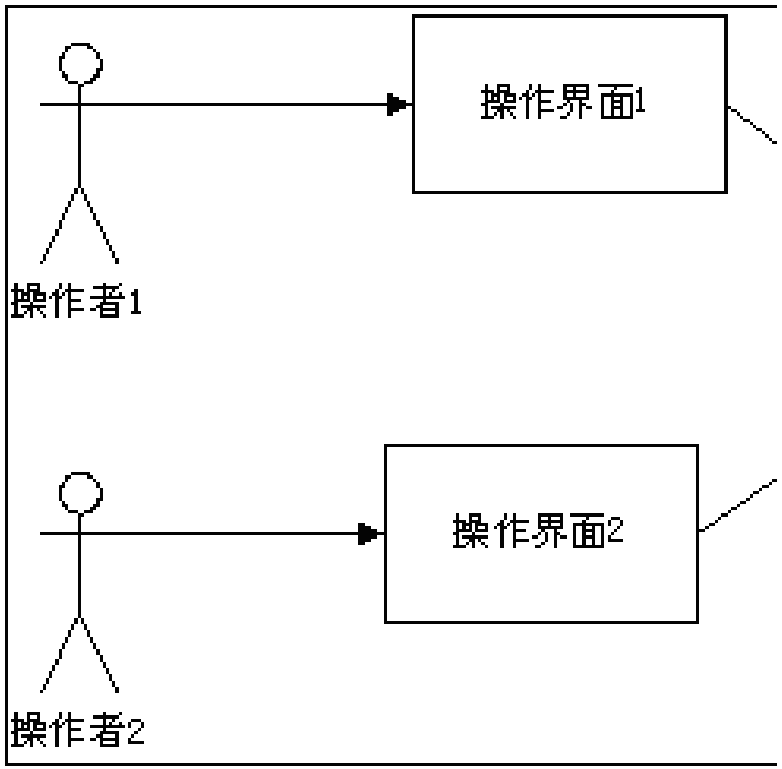
  核心业务→分析模型s→软件架构→实现概念用况场景
- 采用分析模型的方式实现核心业务
- 分析模型采用MVC模式，将用况场景中描述的业务分为边界，控制和实体，提供了系统实现需求的理解和输入。
- 场景图只是概念性的描述，实现必须引入软件架构。

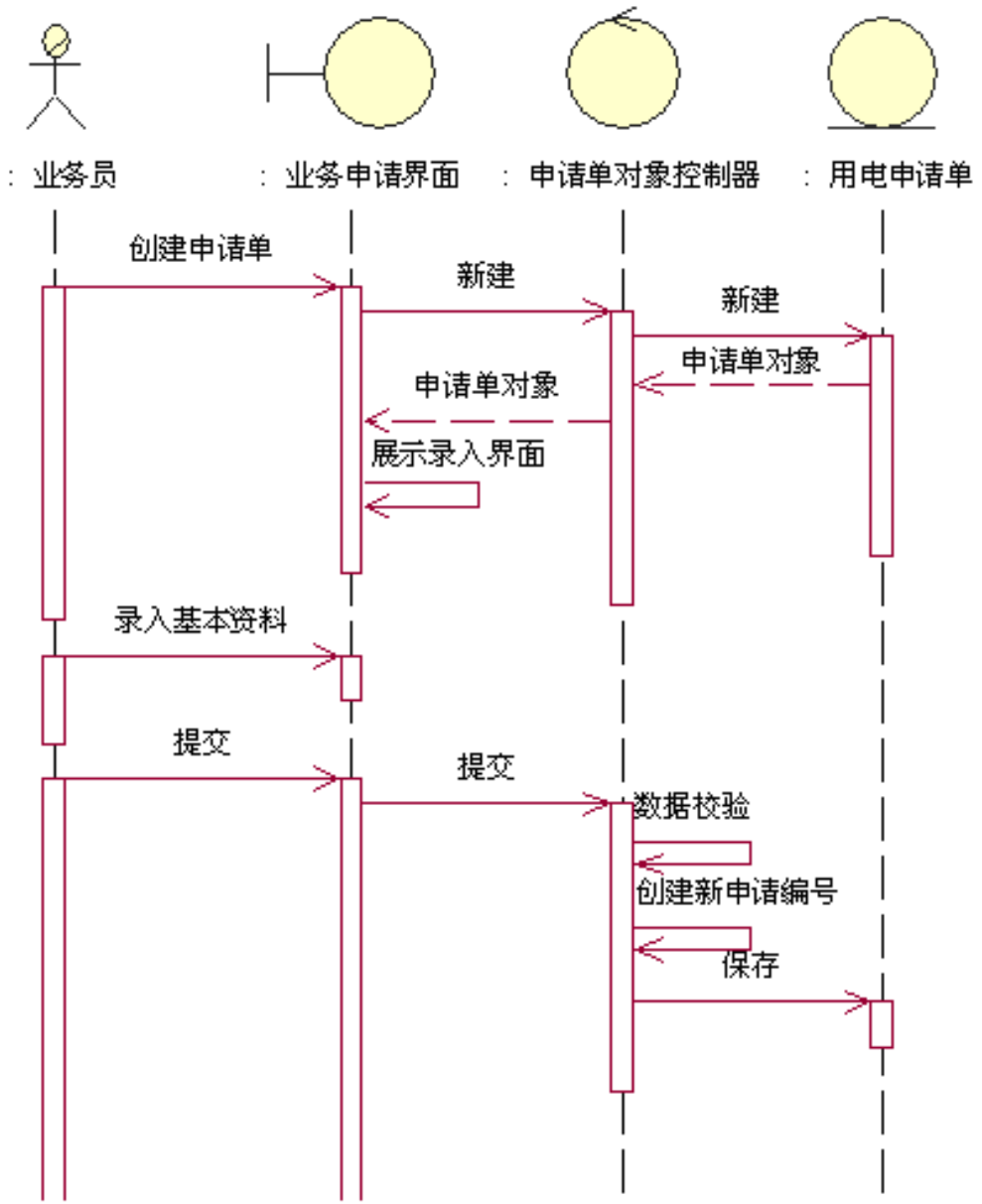
创建申请单分析模型



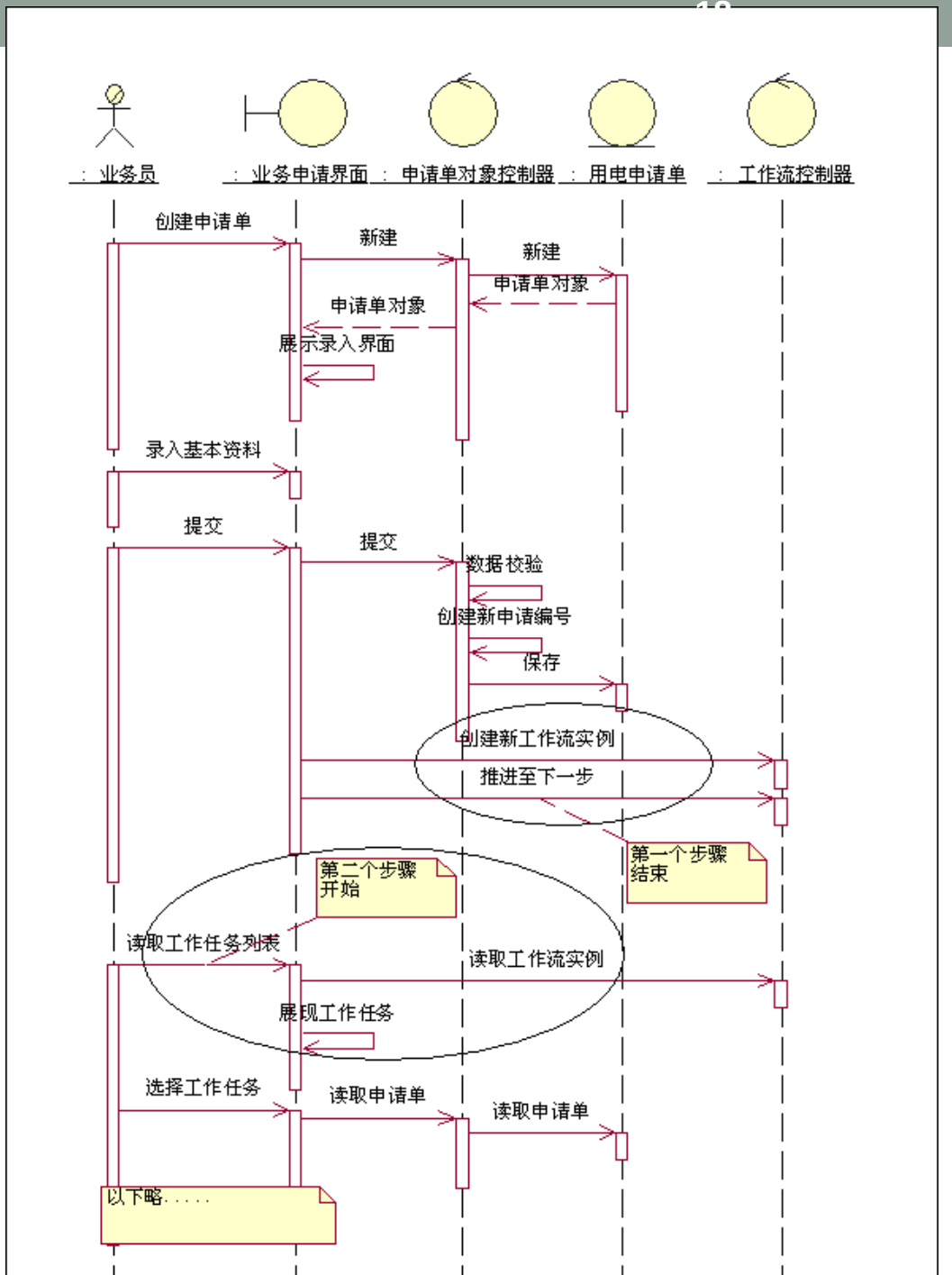
分析场景会多而複杂

C/S架构


操作界面1
操作者1
操作界面2
操作者2


: 业务员
: 业务申请界面
: 申请单对象控制器
: 用电申请单
创建申请单
新建
新建
申请单对象
申请单对象
展示录入界面
录入基本资料
提交
提交
数据校验
创建新申请编号
保存

工作流


：业务员
：业务申请界面
：申请单对象控制器
：用电申请单
：工作流控制器
创建申请单
新建
新建
申请单对象
申请单对象
展示录入界面
录入基本资料
提交
提交
数据校验
创建新申请编号
保存
创建新工作流实例
推进至下一步
第一个步骤结束
第二个步骤开始
读取工作任务列表
读取工作流实例
展现工作任务
选择工作任务
读取申请单
读取申请单
以下略.....

讨论1 概念模型和领域模型

- 看似相似：从业务用况场景出发找到实体类，用实体类实现业务场景获得业务在系统中的理解。

- 实则不同：领域模型不一定针对业务而是针对问题/概念模型只针对业务。

- 目的: 因为业务比较复杂，透过领域建模以对应複杂的业务/在业务向系统理解转化前，在项目早期发现问题。

- 分辨方法: 发现某个问题值得研究/需要验证需求、进行早期的可行性试验

讨论2 软件架构的引入

- 软件架构不只是一组技术框架。
- 从业务的角度还是从技术(流行/先进/技术)的角度选择软件架构?
- 架构的意义在于是否能够很好的解决业务问题。

业务架构

业务架构

- 透过业务架构的活动，面相对象设计的方法→业务构件
- 需求分析与关键概念分析都是**自底向上**的分析，但业务架构是**自顶向下**。
- 将需求分成清晰规模小的构件，设计开发时可以一个一个构件进行。
- 业务架构= 业务用况模型+领域模型+概念模型

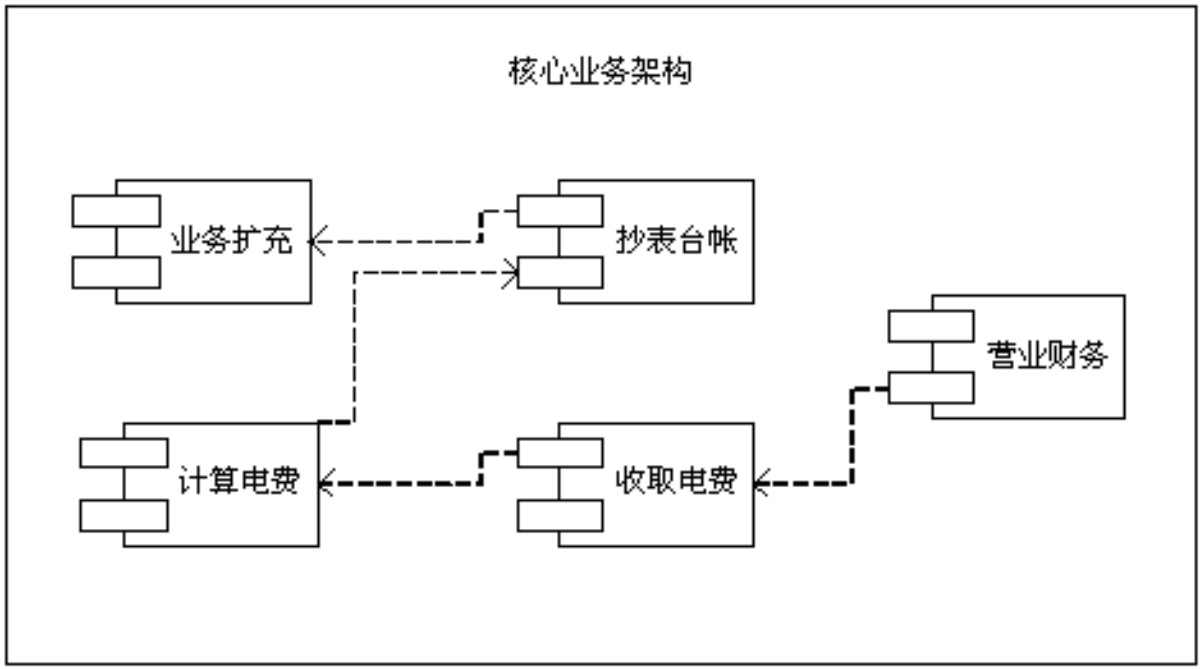
▲供电管理系统核心业务：建立并管理用电用户、计算电费、收取电费、形成供电收入

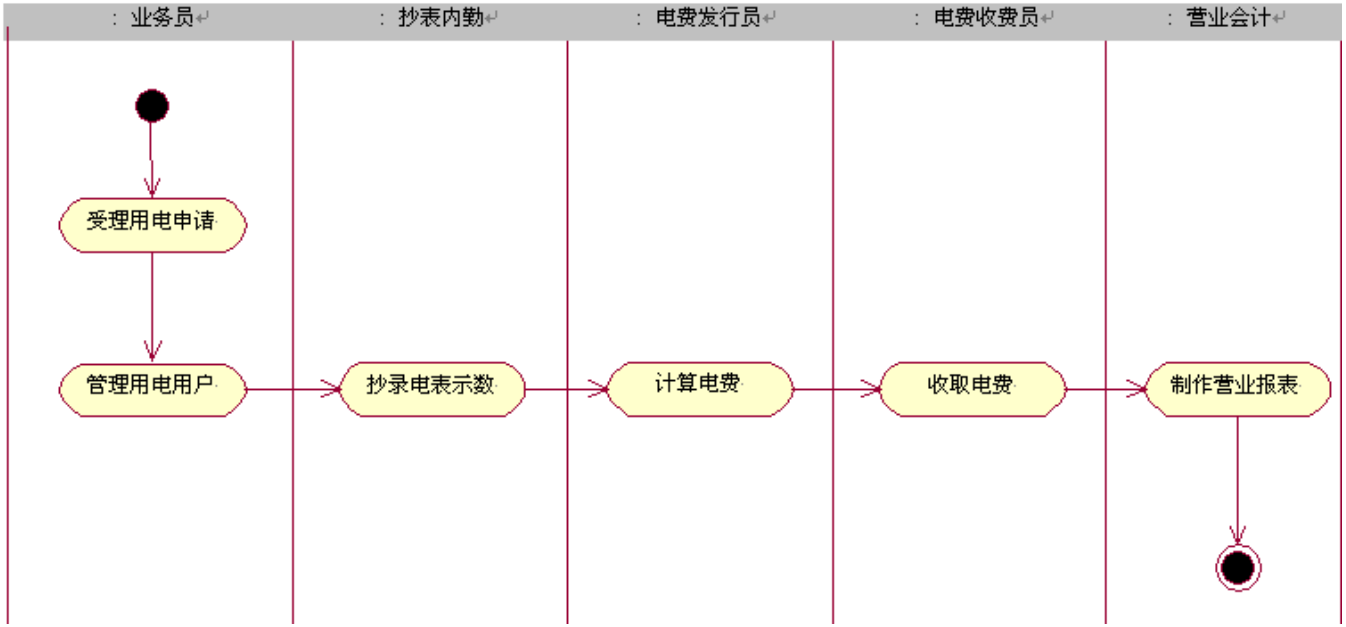
▲对业务名词进行抽象化的改动

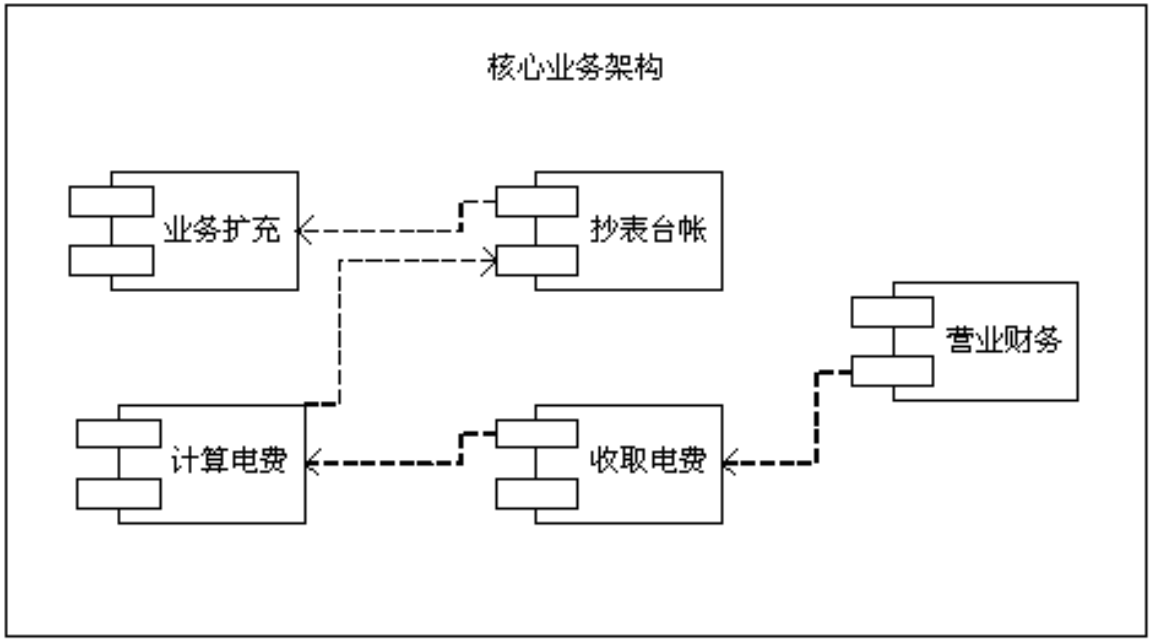
eg.业务扩充，抄表台帐

▲从面向对象的角度来说，形成继续抽象的基础：

获得小的构件，得到构件之间的依赖关系




：业务员
：抄表内勤
：电费发行员
：电费收费员
：营业会计
受理用电申请
管理用电用户
抄录电表示数
计算电费
收取电费
制作营业报表

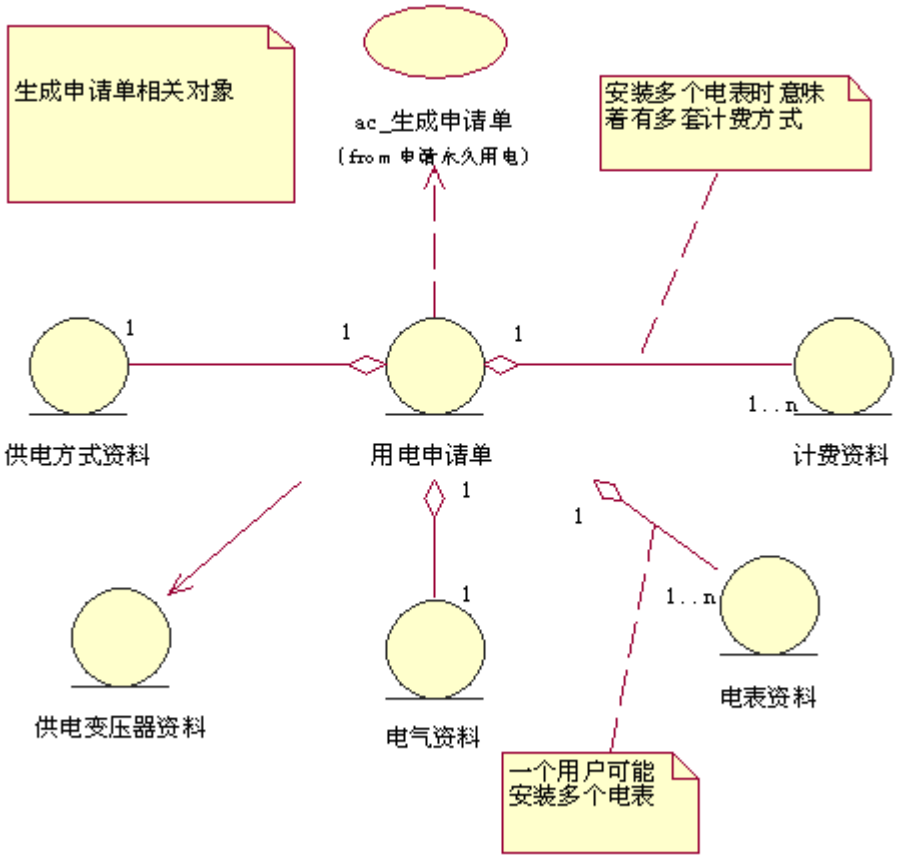

核心业务架构
业务扩充
抄表台帐
营业财务
计算电费
收取电费

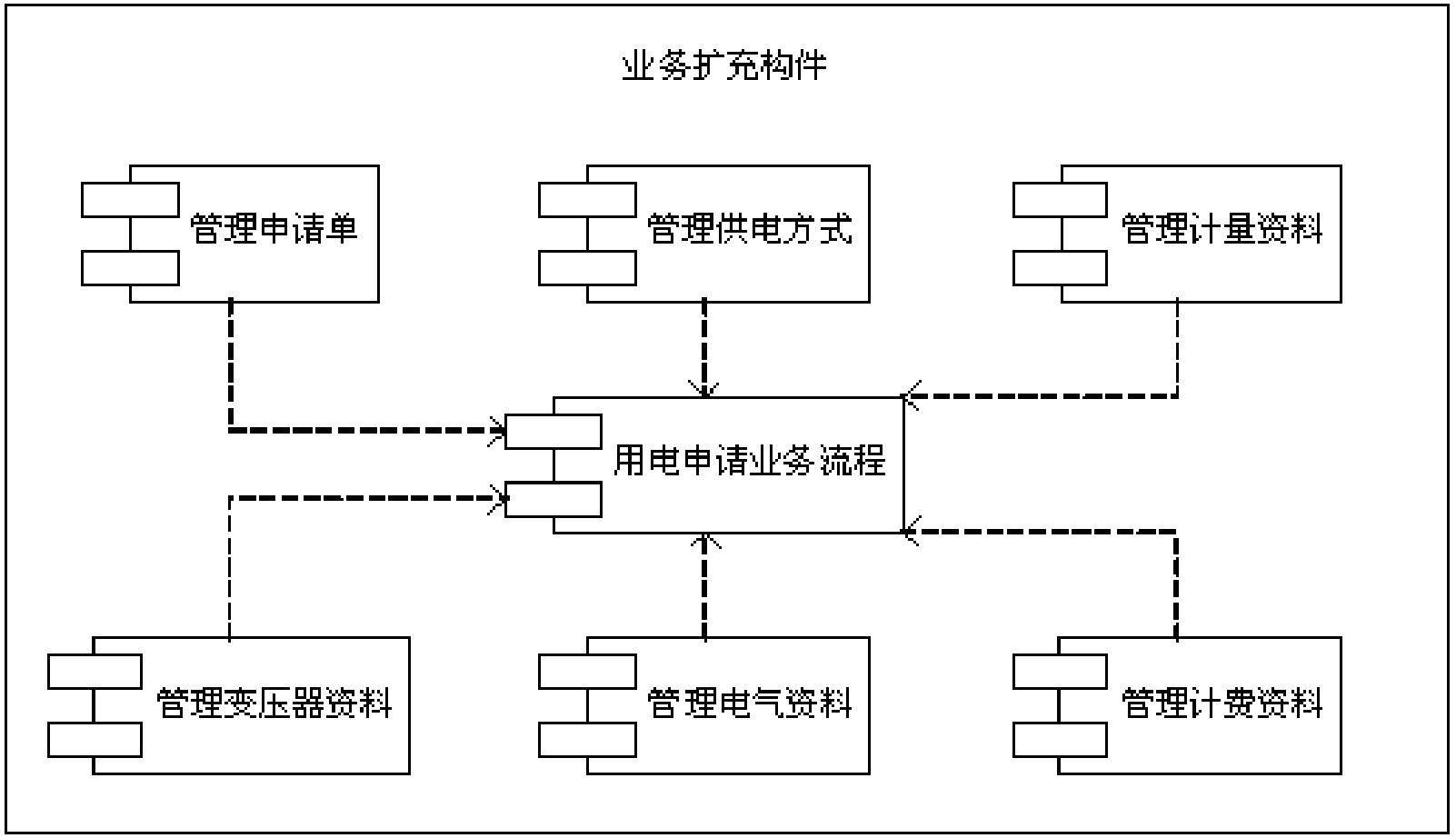
获得小的构件

Q1每个大的构件是由那些小的构成?(领域模型10.6的对象就是业务扩充的基础)

A1每个业务构件都代表一个业务功能单元，每个业务构件都是专业化的，从复杂的业务中拨离出结构化开发的元件。

- 仍然有问题！
- 业务架构可能错误、
  可能不满足实际业务需求。




业务扩充构件
管理申请单
管理供电方式
管理计量资料
用电申请业务流程
管理变压器资料
管理电气资料
管理计费资料

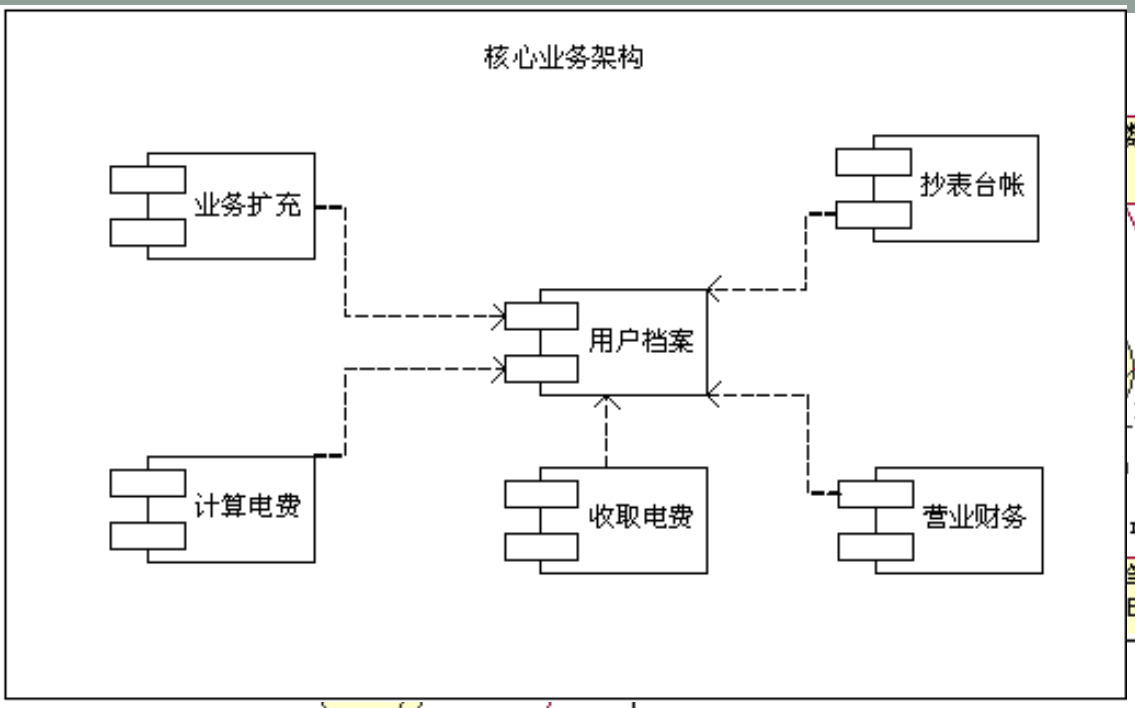
构件之间的依赖关系

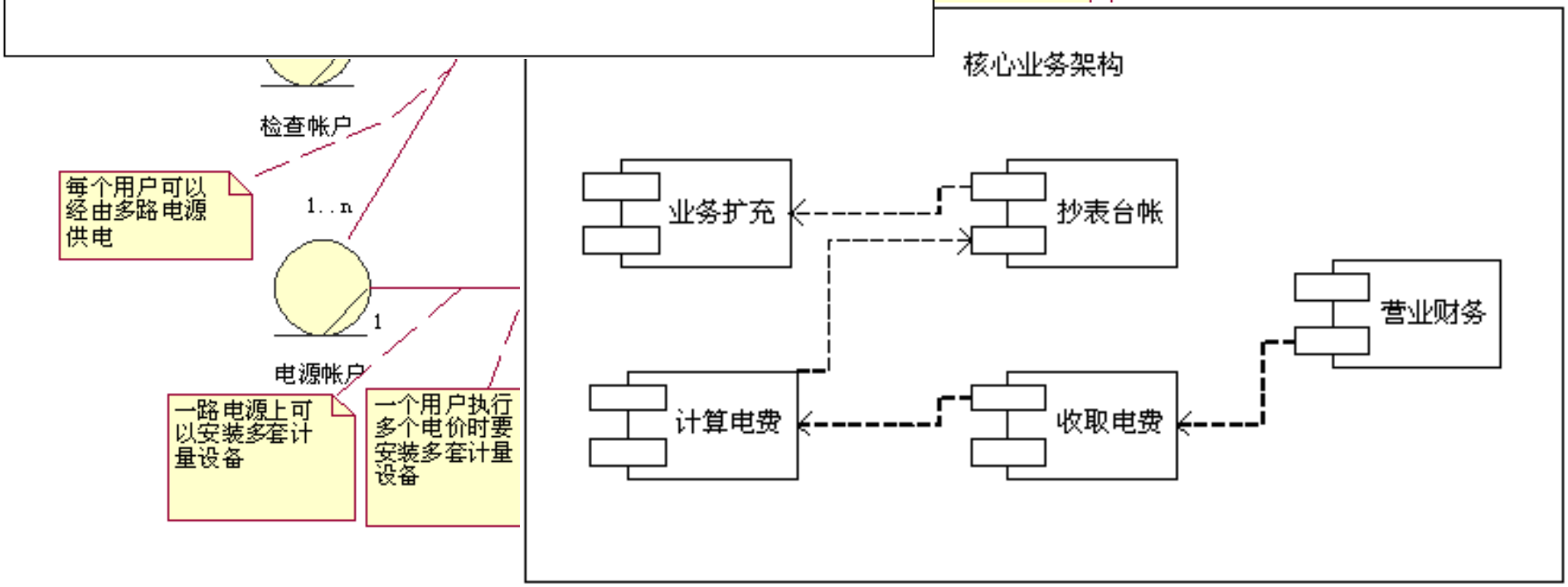
Q2构件之间的依赖关系透过什么维系?

A2 用户档案

粗粒度：由领域建模我们知道**用户档案**是维系这些构件的关键因素，结合**核心业务架构**得到新核心业务架构: 通过读写用户档案，独立的业务构件通过用户档案构件维繫依赖关系。

细粒度：依据领域模型往下分析业务构件间的依赖关系或业务关系，若问题还是太複杂可以再次建立新的领域模型/概念模型。


核心业务架构
业务扩充
抄表台帐
用户档案
计算电费
收取电费
营业财务

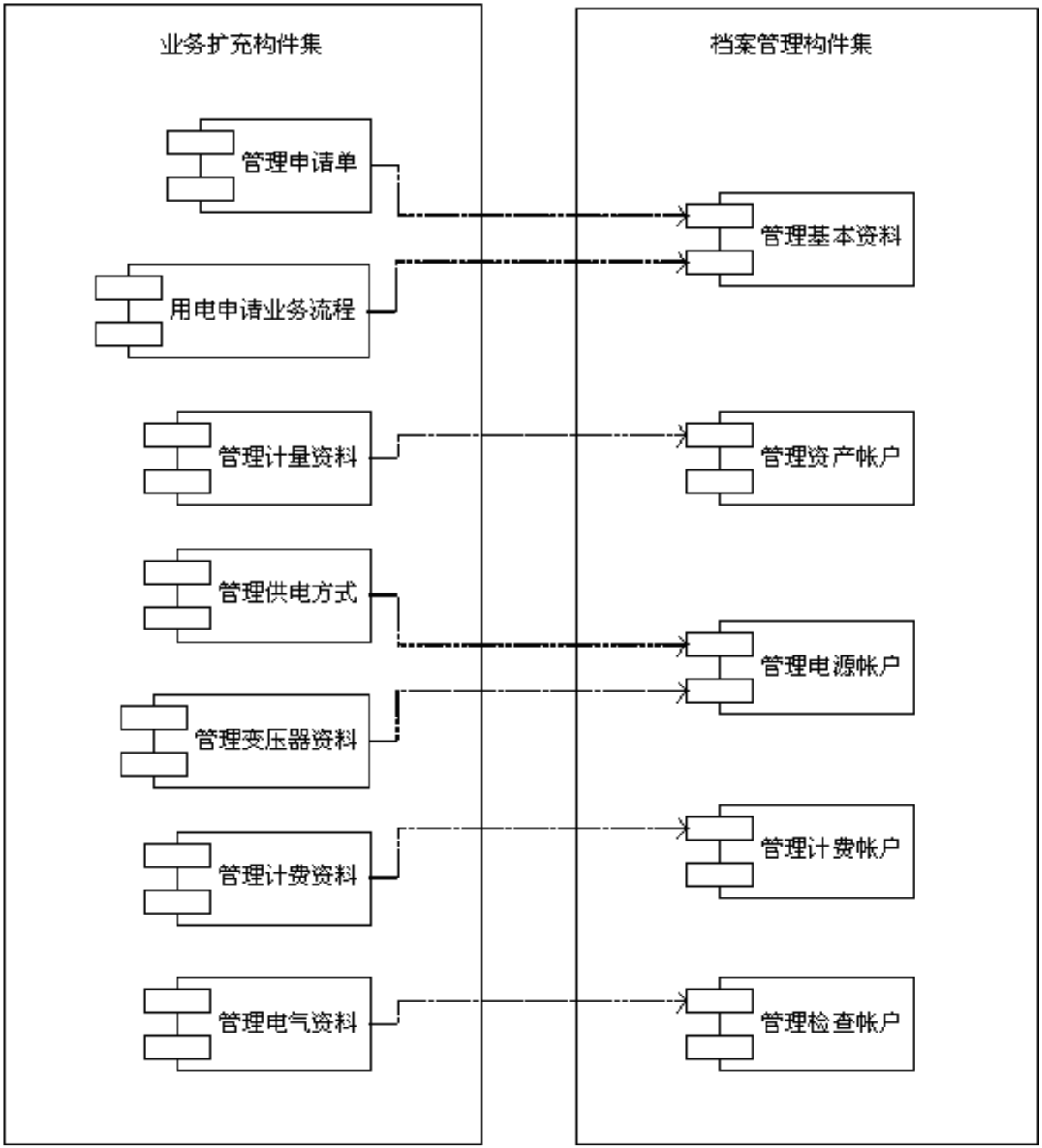

数计
1
1
电费台帐
n
1
笔电费可以
由多次收费
检查帐户
每个用户可以
经由多路电源
供电
1..n
1
电源帐户
一路电源上可
以安装多套计
量设备
一个用户执行
多个电价时要
安装多套计量
设备
核心业务架构
业务扩充
抄表台帐
营业财务
计算电费
收取电费

构建依赖关系的例子

- 管理申请与用户档案的业务架构建模

- 最终实现的时候这些关系可以是数据库表的外键关系，也可以是实体对象的依赖关系，取决于系统实际选择的实现方式。

- 得出其他构件集间的关系后，整个供电企业管理系统中核心业务的业务架构就建立完毕了。

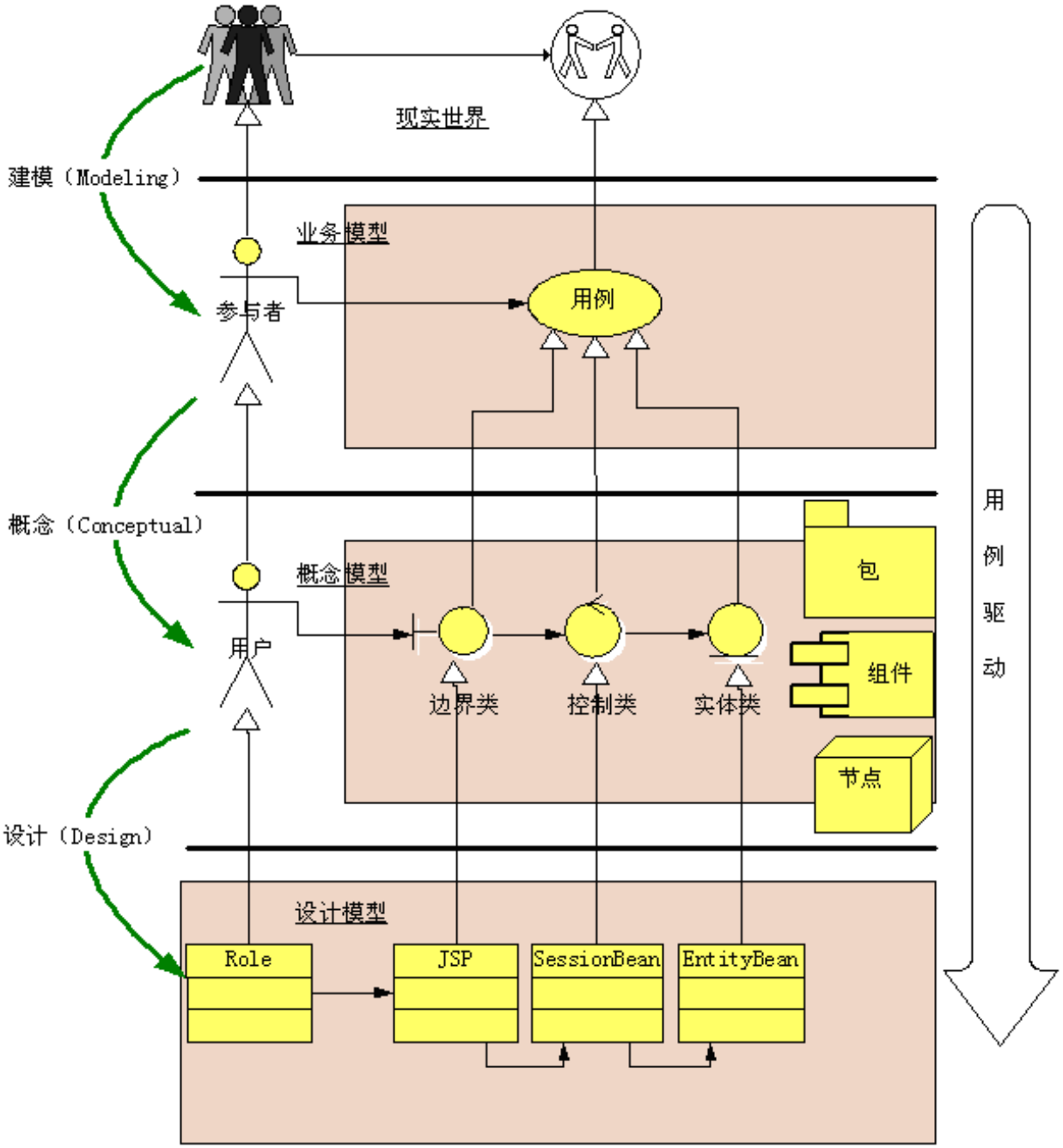

业务扩充构件集
档案管理构件集
管理申请单
用电申请业务流程
管理计量资料
管理供电方式
管理变压器资料
管理计费资料
管理电气资料
管理基本资料
管理资产帐户
管理电源帐户
管理计费帐户
管理检查帐户

讨论1 结构化设计方法和业务架构方法

业务架构	不同的角度	面相过程中结构化的设计方法
抽象抽取	动作	分解
严密推倒过程,模型或场景 透过模型和场景获得业务构件	方法	子系统以部门或业务模块分类，功能模块把子系统功能简单分化
功能点集合 (eg.申请单的整个生命週期管理围绕申请单一个实体类,并提供系列接口和服务给外部使用)	结果	子系统,功能模块

讨论2业务构件和业务实体

- 业务构件围绕业务实体，因为大部分管理类的软件核心就是业务实体。但实际上为了让系统具备更好的扩展性和可维护性，构件可以只封装逻辑。
- eg.申请单管理不包含某些规则限制，所以构件一个管理申请单业务规则的构件，只封装业务规则。


现实世界
建模（Modeling）
业务模型
参与者
用例
概念（Conceptual）
概念模型
用户
包
组件
节点
边界类
控制类
实体类
用例驱动
设计（Design）
设计模型
Role
JSP
SessionBean
EntityBean

问题

在本节中，业务架构的建立是从之前的建模工作当中推导出来的，请读者思考以下问题：

甲、业务用力模型的意义和价值是什麼? 业务用力模型将如何影响到领域模型和概念模型?

乙、领域模型的意义和价值是什麼? 领域模型如何帮助你理解需求进而影响到设计?

丙、概念模型的意义和价值是什麼? 概念模型如何帮助你建立业务架构，加深业务理解?

丁、你能描述出上述模型是如何结合再一起为业务架构和设计工作提供帮助的吗?

系统原型

系统原型

- 解决问题：

  对业务的理解正确吗?

  业务架构合理吗?

  软件架构适用吗?

- 保证不在晚期发现问题的方法：
  - 防范：更详细的需求分析/合理软件过程/评审制度/质量保证机制
  - 快速处理：敏捷方法
  - 系统原型

系统原型的分类

- 按目的分类：抛弃型、渐进型
- 验证性原型：验证目前的工作是否正确
- 探索型原型：能进行多少步骤
- 辅助型原型：直观表达产品概念给客户

系统原型的优点

- 帮助发现项目中的问题
- 客户可以对系统有直观的体验，增进改善沟通，提供反馈，更深刻的理解需求，对产品产生亲切感
- 避免顾客对系统的失望，增进对系统的忍耐度

Thank You

- 業務用力>概念用況>分析概念用況>找關鍵對象 (>協作圖) >業務主線>概念模型

實現核心業務

分析模型

步驟銜接

分析模型

- 用力場景的業務
- 邊界類—消息→控制類—數據→實體類

步驟銜接