在功能的定义中说过，组件间存在相互作用是存在功能的前提条件，因此若不存在与被剪裁组件执行相似功能的组件时，退而求其次，可以寻找那些与原功能对象有相互作用的组件来代替。比如外出时所有类似书包、塑料袋这样执行装载功能的组件都没有了，那可以看看要携带的衣物行李与什么有接触。我们的衣物平时是和其他衣物一起放在衣橱里的，那么有没有可能用其他衣物来装载要携带的衣物行李呢？比如把旧衣物当成包袱。

功能再分配条件4：如果一个组件具备执行原功能所需的资源，那么它有可能替代被剪裁的组件，如图5.2.9所示。

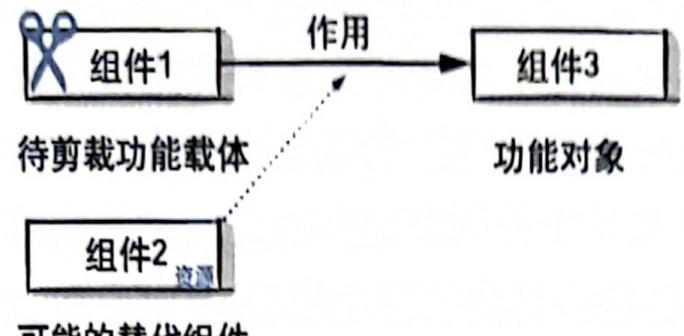


图5.2.9 功能再分配条件4

如果满足前三个功能再分配条件的组件都不存在，那么最后可以考虑那些虽然与原来的功能对象没有相互作用（自然也不存在功能），但是具备要执行功能所需的各种资源的组件。比如这次外出时连多余的旧衣物都没有了，那么就要思考执行“装载”功能需要哪些资源呢？比如需要有一个易于封闭起来的空间（空间资源）来完成“装”的工作，还需要有一定强度的材料（物质资源）来完成“载”的工作，因此可能就会找到家中的床单、方巾等作为替代组件。虽然这些物品平时与行李并没有什么相互作用，也不用来执行装载的功能，但由于它们具备装载所需的物质和空间资源，因此特殊情况下完全可以用来替代行李箱应急。

通过介绍，相信读者也能体会到，上述4个功能再分配条件是存在递进关系的。当我们要执行剪裁寻找替代组件时，可以依次尝试。

**5.2.4 如何建立剪裁模型**

剪裁模型是剪裁工具最终的输出。从应用的角度讲，剪裁是在原有功能模型的基础上删除部分组件，并将其有用功能分配给其他组件的过程。因此，所谓剪裁模型并不是一个新的概念，而是指剪裁之后技术系统的功能模型。

需要注意的是，当我们实施剪裁时选择不同的剪裁对象或者选用不同的剪裁规则，会产生不同的剪裁模型。因此在实际工程项目中，往往需要做出一个剪裁的计划，称之为剪裁方案，可以用表5.2.1作为剪裁方案模板。

表5.2.1 剪裁方案列表模板

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 剪裁对象 | 功能 | 剪裁规则 | 替代组件 | 剪裁问题 |
| 组件1 | 功能A | 剪裁规则1 |  |  |
| 组件2 | 功能B | 剪裁规则2 | 组件4（功能B的对象） | 如何使组件4执行功能B |
| 组件3 | 功能C | 剪裁规则3 | 组件5 | 如何使组件5执行功能C |
| 功能D | 剪裁规则3 | 组件6 | 如何使组件6执行功能D |

按照此模板，依次填写选择的待剪裁组件及其执行的所有有用功能，选择运用的剪裁规则，除剪裁规则1之外需要按照功能再分配条件寻找代替执行功能的新载体，以及剪裁后出现的剪裁问题。对于存在多个有用功能的组件（如上表中组件3），需要按照功能分别选择剪裁规则并寻找替代组件。每完成一个组件的剪裁方案，就可以画出相应的剪裁模型。

### 案例背景

假设我们在计算机科学与技术专业的学习中，遇到服务器系统中的冗余模块导致资源浪费和系统复杂度增加的问题。我们尝试用剪裁的方式，将冗余模块去除。其有用功能包括提升系统可靠性和数据恢复，选择运用剪裁规则2，由其他组件实现这些功能。产生的剪裁问题就是“如何在不增加其他模块的情况下提升系统可靠性”以及“如何实现数据恢复”，依次填入剪裁方案列表。

### 表格模板

| **剪裁对象** | **功能** | **剪裁规则** | **替代组件** | **剪裁问题** |
| --- | --- | --- | --- | --- |
| 冗余模块 | 提升系统可靠性 | 剪裁规则2 | 主服务器 | 如何在不增加其他模块的情况下提升系统可靠性 |
|  | 数据恢复 | 剪裁规则2 | 主服务器 | 如何实现数据恢复 |

### 剪裁方案

**1. 冗余模块：提升系统可靠性**

* **功能**：提升系统可靠性
* **剪裁规则**：剪裁规则2
* **替代组件**：主服务器
* **剪裁问题**：如何在不增加其他模块的情况下提升系统可靠性

**2. 冗余模块：数据恢复**

* **功能**：数据恢复
* **剪裁规则**：剪裁规则2
* **替代组件**：主服务器
* **剪裁问题**：如何实现数据恢复

### 剪裁后的功能模型

**图5.2.10 剪裁冗余模块后服务器系统的剪裁模型**

此模型中，并没有通过例如优化冗余模块算法、增加硬件资源等方式来解决“冗余模块导致资源浪费和系统复杂度增加”的初始问题，而是以剪裁的方式将冗余模块从服务器系统中去除，自然也就消除了初始问题。同时，剪裁掉冗余模块后，还可能带来一些其他利好，比如系统成本降低、复杂度减少等等。当然，通过剪裁并没有形成真正的解决方案，而是把初始问题转换成了前述的剪裁问题，后续可以作为关键问题，利用其他技术手段和工具加以解决，例如通过分布式存储架构或者云服务实现数据恢复和系统可靠性的提升。

### 替代解决方案

**1. 提升系统可靠性**

可以使用分布式系统架构，利用多节点的协作来保证系统的高可靠性，即使某个节点出现故障，其他节点也能继续工作。

**2. 数据恢复**

利用云备份服务或者基于区块链技术的去中心化存储来实现数据恢复，不依赖于单一的冗余模块。

### 总结

通过这种方式，我们不仅解决了初始问题，还能为系统带来更多的优化空间和灵活性，有助于更高效地利用资源和提升系统性能。

**5.3 剪裁案例**

在开始剪裁之前，首先需要建立功能模型。与3.4.3小节那个简单的功能模型略有不同，为了更好地展示从温和到激进的剪裁过程，此处建立一个更加细化的功能模型，如图5.3.1所示。接下来根据不同的项目目标，可以尝试几套不同的剪裁方案。

**5.3.1 降低成本为导向的剪裁方案**

1．选择剪裁对象

按照剪裁流程，首先需要选择剪裁对象。如果项目目标是降低成本，那么在剪裁开始时，可以依据功能-成本分析，优先选择那些技术系统中的低价值组件。

造成组件价值低的原因可能是其功能性较弱，也可能是其成本较高，或者两者兼有。如果采用温和的剪裁思路，可以从功能性较弱的组件入手，因为功能弱意味着容易被取代，剪裁难度就会比较低；如果采用激进的剪裁思路，则可以从成本偏高的组件人手，只是当这些组件具备较强功能性的时候，剪裁难度也会