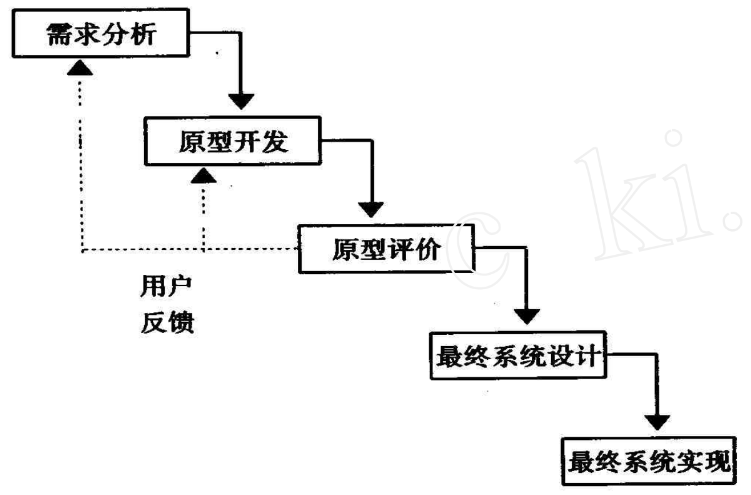
## 抛弃型原型和演化型原型

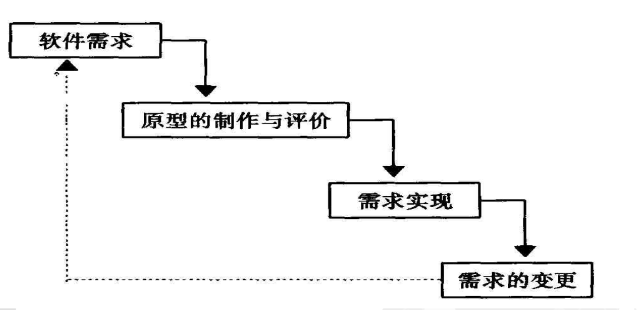
如果想解释一些问题，解决不确定性以及改进需求，就可以**快速，低成本**的创建一个**抛弃型原型**。创建抛弃性原型时，会忽略成品软件构建技术。相比健壮性，可靠性，性能和长期可维护性，可抛弃性原型更注重快速实现以及快速修改。

当团队觉得需求不确定，有歧义，不完整或者含糊时，就可以选择建立抛弃型原型。



相比较于抛弃型原型，演化型原型更加稳固。用前期的迭代来调整未来开发周期的方向，并通过一系列迭代来完成产品的构建。这就是演化型原型的实质。

构建**演化型原型**时，一开始就要考虑**健壮性**，写产品级质量的代码。需要重视软件架构和稳健的设计原则。每一轮迭代都会在实现部分需求的时候获取下一轮迭代需要修改的内容。整个产品就是由一系列对原型进行周期化演化积累而获得的。



|  |  |  |
| --- | --- | --- |
|  | 抛弃型 | 演化型 |
| 演示型模型 | 澄清与提炼用户需求和功能  识别被遗漏的功能  研究UI方法 | 实现核心用户需求  基于优先级实现额外的用户需求  实现和优化网站、  使系统与快速变化的业务需要相适应 |
| 概念证明 | 演示技术的可行性  评估性能  获得更多知识以提升估算能力 | 实现和拓展核心多层级功能以及通信层  实现和优化核心算法  性能测试和调优 |