131250129 梁思宇

ADD方法是一个递归的分解过程，在每个阶段都选择策略和构架模式来满足一组质量属性场景。我们先进行了需求分析，然后运用ADD方法定义软件架构。ADD的结果是构架的模块分解视图和其它视图的最初几个层次，因此我们最终得到的成果是粗粒度的，然而ADD方法对实现期望的质量属性来说还是非常重要的。

在使用ADD方法的过程中，首先将功能需求和约束作为输入，并把质量需求表示为一组特定于系统的场景。开始ADD方法之后，首先选择要分解的系统，接着对分解模块进行求精，然后对需要进一步分解的每个模块重复以上两个步骤。在实验中我们的工作分为了三个迭代，在第一个迭代中，我们将模块分解成子模块，在第二、三迭代中，根据优先级选择了重要的ASR，为ASR建立候选策略表，并从中选择出最合理的策略。选择策略的过程中，既要考虑构架驱动因素本身，又要考虑实施这个策略对其他质量属性的影响，权衡选择。三个迭代完成后，进行验证，看是否通过分解满足了所有质量属性场景。如果质量属性场景不满意当前的分解，就还需要再分解，直到所有的质量属性场景都得到满足。

在这次的小组作业中，我作为成员参加了每一次的小组讨论，与其他成员一起进行需求分析，探讨ASR的选择，加深对ADD方法的理解，彼此协商交流意见。在最终的报告中，我负责的是视图归档部分，我选择了合适的UML图做了有关模块视图的分析并将内容组织为文档。