博客作业

• 1. 论述测试与正确性论证的效果差异，比较其优缺点

• 2. 调研OCL语言，并比较其与课程所介绍的JSF规格之间的相似和不同 之处

• 3. 根据第十四次作业的单电梯系统，针对调度器、电梯、请求队列和 请求，至少整理出 一幅UML类图、一幅顺序图和一幅状态图，并使用 图(graph)来表示出模型

• 4. 整理总结一个学期所学所练

4.1阐述四个单元模块知识点之间的关系

4.2梳理自己所设计实现的程序，分析自己在设计、测试和质量上的进步

4.3阐述自己对工程化开发的理解

4.4对课程的任何期望或建议

1. 论述测试与正确性论证的效果差异，比较其优缺点

测试是面对程序，构造样例进行验证，让程序运行，观察程序的运行结果和预期结果比较，来判断在这种情况下程序的正确性。有点像形式验证。

优点：相对容易，比较直观

缺点：不能保证绝对正确

正确性是根据规格推断程序的逻辑，检测是否完成任务。类似于逻辑验证。是进行测试的另一种方式。

优点：能够较好地保证验证效果。

缺点：繁琐复杂，不直观，成本高

1. 调研OCL语言，并比较其与课程所介绍的JSF规格之间的相似和不同之处

OCL(Object Constraint Language)语言是面向对象设计中对图形符号的补充，能够完善建模元素的相关细节。其与JSF规格有相似之处，也有不同。

相同之处：

都是一种形式化的语言

具有无二义性

每个变量都有类型

不会改变系统的运行状态

都采用前置条件和后置条件对方法的运行加以约束

不同之处：

变量的类型：

OCL有自己的一套变量（如Integer, Real, String, Boolean, Set, Bag, Sequence）

JSF不仅支持Java语言定义的所有类型，还支持用户自定义的类型（比如有一个Request类， 则可以在JSF中用Request修饰变量）

变量的运算：

OCL对变量规定了可以进行的运算方式（如 Boolean类型的and, or）

JSF支持Java语言定义的所有运算方式

逻辑分支：

OCL支持if, else一类的逻辑分支。

JSF不支持if, else, while等逻辑分支。

上下文

OCL支持用context表达上下文

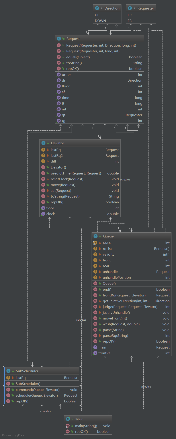
JSF不支持上下文表述。

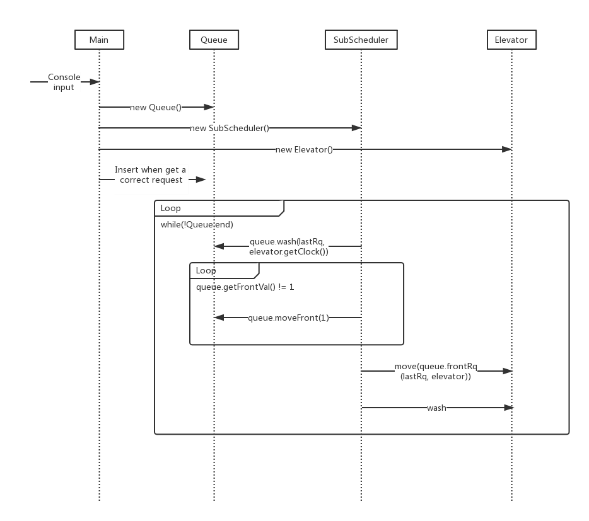
符号系统

OCL用pre, post表达前置条件和后置条件

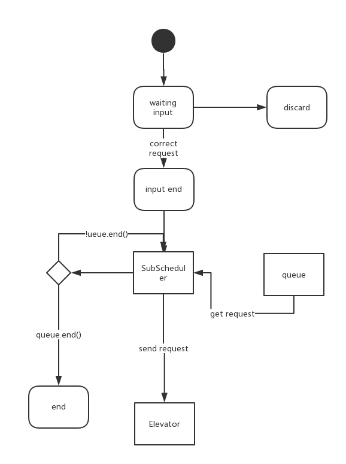
JSF用REQUIRES, EFFECTS表达前置条件和后置条件

1. 根据第十四次作业的单电梯系统，针对调度器、电梯、请求队列和请求，至少整理出一幅UML类图、一幅顺序图和一幅状态图，并使用图(graph)来表示出模型。

类图：

时序图：

状态图：



1. 整理总结一个学期所学所练

* 阐述四个单元模块知识点之间的关系

第一单元概述面向对象，区别面向对象和面向过程。训练从类的角度看问题和事物。第二单元主要是多线程编程，学习并发时的同步与互斥。第三单元主要是JSF规格的撰写。学习如何通过规格约束代码，养成好的程序设计习惯。第四单元主要是测试与正确性论证，同样十分消耗精力。四个单元呈现出递进的层次关系，从单线程到多线程，从编写程序到测试程序到论证程序，从局部设计到整体规划，循序渐进。同时也是难度的增加，工程性的提高。

* 梳理自己所设计实现的程序，分析自己在设计、测试和质量上的进步

单个类的方法减少了，方法的行数也减少了。在设计上学会了提高多用性，提高代码的通用性。学会了形式验证的覆盖性测试和逻辑验证的正确性验证。同时大量的编程也提高了我的编程和debug能力。

* 阐述自己对工程化开发的理解

工程化即系统化、模块化、规范化的一个过程。指将具有一定规模数量的单个系统或功能部件，按照一定的规范，组合成一个模块鲜明、系统性强的整体。工程化往往包含大量学科和学科分支的知识，是一个复杂的系统工程过程。我们这门课的作业的编写可能涉及到了许多工程化开发的思想，而实际的代码规模还谈不上工程化的水平。不过除了代码量之外，工程化开发重要的还是工程代码的可拓展性和可维护性，像是作业中的出租车系列，电梯系列，虽说都是基于前一次作业上稍加修改，但是在编写过程中能够很明显地感受到不同的设计方法，带来的拓展代价的差异巨大。

* 对课程的任何期望或建议。

规范互测。

继续推行客服制度。

支持从一而终不换题目。