# 一、结构设计 （何）

# 二、模块功能设计

## 建模模块 （刘）

## 验证模块 （王）

## 代码生成模块 （李）

# 三、信息处理设计（输入、输出）（何）

# 四、关键数据结构设计

## Model类

核心的底层数据结构，用于存储由xml文件生成的模型组件（包括类图、状态及、序列图等）。

### 1.1 属性：

list<Process\*> processes;

list<Property\*> properties;

list<InitialKnowledge\*> initialKnowledges;

SequenceDiagram\* sequenceDiagram;

其中processes属性用于存储Model模型中的所有进程；properties用于存储需要验证的安全属性；initialKnowledges用于存储各个process中的初始知识，sequenceDiagram用于存储序列图。

### 1.2 方法：

## Process类

### 2.1 属性：

list<Attribute\*> attributes;

list<Method\*> methods;

list<Signal\*> signals;

FiniteStateMachine\* fsm;

Model\* model;

其中attributes是用于存储Process中定义的所有属性，我们支持原生的int、bool数据类型，用户还可以根据需求自己定义数据类型；methods是用于存储Process中定义的所有方法；signals是用于存储Process中定义的所有信道，信道是各进程之间用于通信的通道；fsm是Process对应的有限状态系统，用来定义Process的行为；model是Process所属的模型。

### 2.2 方法：

## FiniteStateMachine类

### 3.1 属性：

list<Vertex\*> vertices;

list<Edge\*> edges;

Vertex\* startVertex;

Process\* process;

其中vertices是用于存储FiniteStateMachine中的状态集合；edges用于存储FiniteStateMachine中所有的边集合，每条边上我们运行定义一个Guard语句以及若干个Action语句，其中Guard语句用于根据条件判断是否执行后面的语句，Action语句用于对属性的操作；startVertex是用于存储FiniteStateMachine中的初始状态；process是其所属的进程。

### 3.1 方法：

## Parse类（刘）

## Translator类（王）

## CodeGenerator类（李）

# 五、接口设计

## 内部接口设计

### 1.1 建模模块与模型接口 （刘）

### 1.2 验证模块与模型接口 （王）

### 1.3 代码生成模块与模型接口 （李）

## 外部接口设计 （何）

# 六、执行序列设计 （何）