解释器模式常用于对简单表达式语言的编译或分析实例中，为了掌握好它的结构与实现，必须先了解\*\*\*编译原理\*\*\*中的“文法、句子、语法树”等相关概念。

#### 1) 文法

文法是用于描述语言的语法结构的形式规则。没有规矩不成方圆，不管它是机器语言还是自然语言，都有它自己的文法规则。例如，中文中的“句子”的文法如下。

〈句子〉::=〈主语〉〈谓语〉〈宾语〉

〈主语〉::=〈代词〉|〈名词〉

〈谓语〉::=〈动词〉

〈宾语〉::=〈代词〉|〈名词〉

〈代词〉你|我|他

〈名词〉7大学生I筱霞I英语

〈动词〉::=是|学习

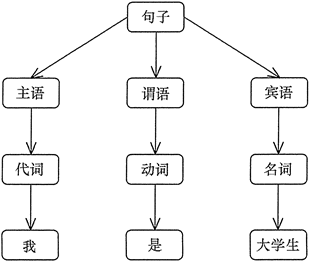
注：这里的符号“::=”表示“定义为”的意思，用“〈”和“〉”括住的是非终结符，没有括住的是终结符。

#### 2) 句子

句子是语言的基本单位，是语言集中的一个元素，它由终结符构成，能由“文法”推导出。例如，上述文法可以推出“我是大学生”，所以它是句子。

#### 3) 语法树

语法树是句子结构的一种树型表示，它代表了句子的推导结果，它有利于理解句子语法结构的层次。图 1 所示是“我是大学生”的语法树。



#### 1. 模式的结构

解释器模式包含以下主要角色。

1. 抽象表达式（Abstract Expression）角色：定义解释器的接口，约定解释器的解释操作，主要包含解释方法

值 interpret( )。

1. 终结符表达式（Terminal Expression）角色：是抽象表达式的子类，用来实现文法中与终结符相关的操作，文法中的每一个终结符都有一个具体终结表达式与之相对应。
2. 非终结符表达式（Nonterminal Expression）角色：也是抽象表达式的子类，用来实现文法中与非终结符相关的操作，文法中的每条规则都对应于一个非终结符表达式。
3. 环境（Context）角色：通常包含各个解释器需要的数据或是公共的功能，一般用来传递被所有解释器共享的数据，后面的解释器可以从这里获取这些值。
4. 客户端（Client）：主要任务是将需要分析的句子或表达式转换成使用解释器对象描述的抽象语法树，然后调用解释器的解释方法，当然也可以通过环境角色间接访问解释器的解释方法。

