需求规格说明文档

软件工程 2017HYSE04 小组

2020 年 3 月 19 日

目录

1 引言

1.1 编写目的

本手册针对 Pythy 语言的用户以及开发者编写，用于明确软件开发的综合需求，并对未来软件开发做出一定的规范。

1.2 背景

本软件是一个编程语言 Pythy 的解释器及集成开发环境，能够实现对 Pythy 语言源文件的编辑、执行与调试功能，并将在指定期限内完成所需的所有功能。本软件由软件工程 2017HYSE04 小组共同开发，人员包括武汉大学弘毅学堂的罗溥晗、莫会民、胡宬、章博文。指导老师为武汉大学计算机学院伍春香教授。验收小组为软件工程 2017HYSE03 小组。

1.3 名词解释

名词	解释
表达式	由运算符，变量，常量组成的符合数学规则的运算式。
语句	能够被执行的最小单位，如赋值，控制，循环，在 Pythy 中通常一行便是一条语句。
编译	对用户输入的程序进行处理，转化成能够被 Pythy 执行器识别的中间表示。
抽象语法树	Pythy 语言的中间表示，用于存储表达式或者语句中包含的所有有效信息。
表达式语法树列表	当输入的程序被编译后，其中的每一个语句的抽象语法树会按顺序组成一个列表。
执行器	给定表达式语法树列表，执行器负责按照正确的顺序执行其中的语句。
源文件	可以被识别并执行的后缀为.py 文件
集成开发环境	一个图形化界面，集成了 Pythy 源文件操作以及编译执行的功能。

表 1.1 名词及其解释

2 任务概述

2.1 目标

Pythy 语言是 Python 语言的子集，能够实现基本的表达式运算、分支、循环等语法，并且在此基础上实现特殊的输入输出语句。语言的具体内容在 Pythy 语言的概述文档[1]中已经陈述。

本软件将实现 Pythy 语言的解释器，同时实现一个配套的集成开发环境，为 Pythy 的源文件提供文件操作、编译以及调试的功能。同时还会提供配套的文档，本说明也是文档的一部分。

2.2 用户的特点

Pythy 语言针对的用户是有一定编程基础或编程基础一般的学生或办公人员，他们可以使用 Pythy 语言编写轻量级的脚本，以方便其工作以及学习。同时 Pythy 语言后续也希望能实现一些在运算能力上的拓展，从而可以被一些用户用作教学方面的用处，比如数值计算，矩阵运算等。

Pythy 语言的维护人员主要是本小组的成员，以及希望能够为此编程语言增添功能的计算机专业的学生。

2.3 假定和约束

本软件的开发工作期限为武汉大学 2019-2020 学年下学期 1-8 周教学周，经费为 0.

2.4 需求实现表格

需求	优先级（数字越小优先级越高）
编译部分	0
执行部分	1
集成开发环境	2

表 2.1 需求实现表格

3 需求规定

3.1 对功能的规定

在这一节中，将以面向对象的分析方法对软件的功能进行需求上的挖掘,基本规范参照《软件工程 面向对象与传统方法》[2]

3.1.1 功能模型

本节以用例图的形式展示了用户与产品之间的交互行为。对用户而言，解释器 IDE 上主要有两个功能可以使用。操作文件：用户可以新建一个文件或者打开一个已存在的文件，对其进行修改编辑并保存，也可以关闭文件。调试运行：用户选择单步执行或连续执行，解释器将执行结果和运行环境展示给用户。

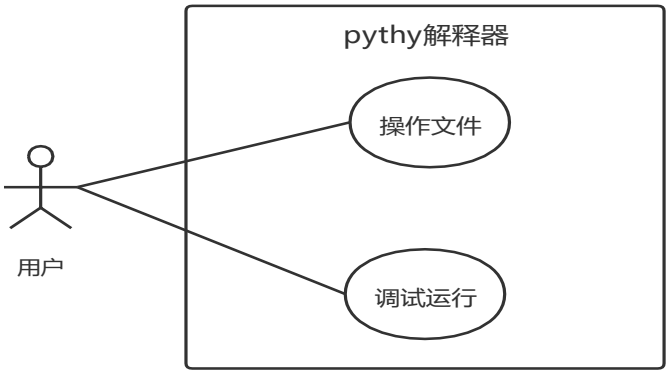

图 3.1 功能建模图

以下给出一个用户新建一个文件，编写代码，正确编译并连续执行的场景描述：

1.用户点击“新建”按钮，选择“新建文件”。 2.解释器 IDE 展示一个空白文件在左侧编辑器。 3.用户用 pythy 语言编写代码。 4.用户点击“保存”，编辑器将文件保存在磁盘中。 5.用户点击“执行”，调试器给编译器发送一条消息并传递执行文件名，编译器开始编译。 6.编译器完成编译，结果正确，给控制器发送一条消息，将控制权和编译结果移交给控制器。 7.控制器根据编译结果依次选择下一条执行的语句，给语句 AST 执行接口发送一条消息执行语句，语句 AST 执行接口给运行环境类发送一条消息修改其中内容，给调试器发送一条消息显示执行结果，同时可能调用表达式 AST 执行接口进行求值。 8.执行结束，用户点击“关闭”，程序退出。

图 3.2 成功执行的场景

3.1.2 对象模型

根据面向对象方法，我们根据功能对解释器的实体进行挖掘，得出了如图 3.3 的类图，其中编辑类、调试类、编译类、执行类是四个基本的类，而在之后的动态建模的过程中，我们也只会考虑这四个类的抽象层面上的动态交互。而其他几个类由更为具体的功能划分出。

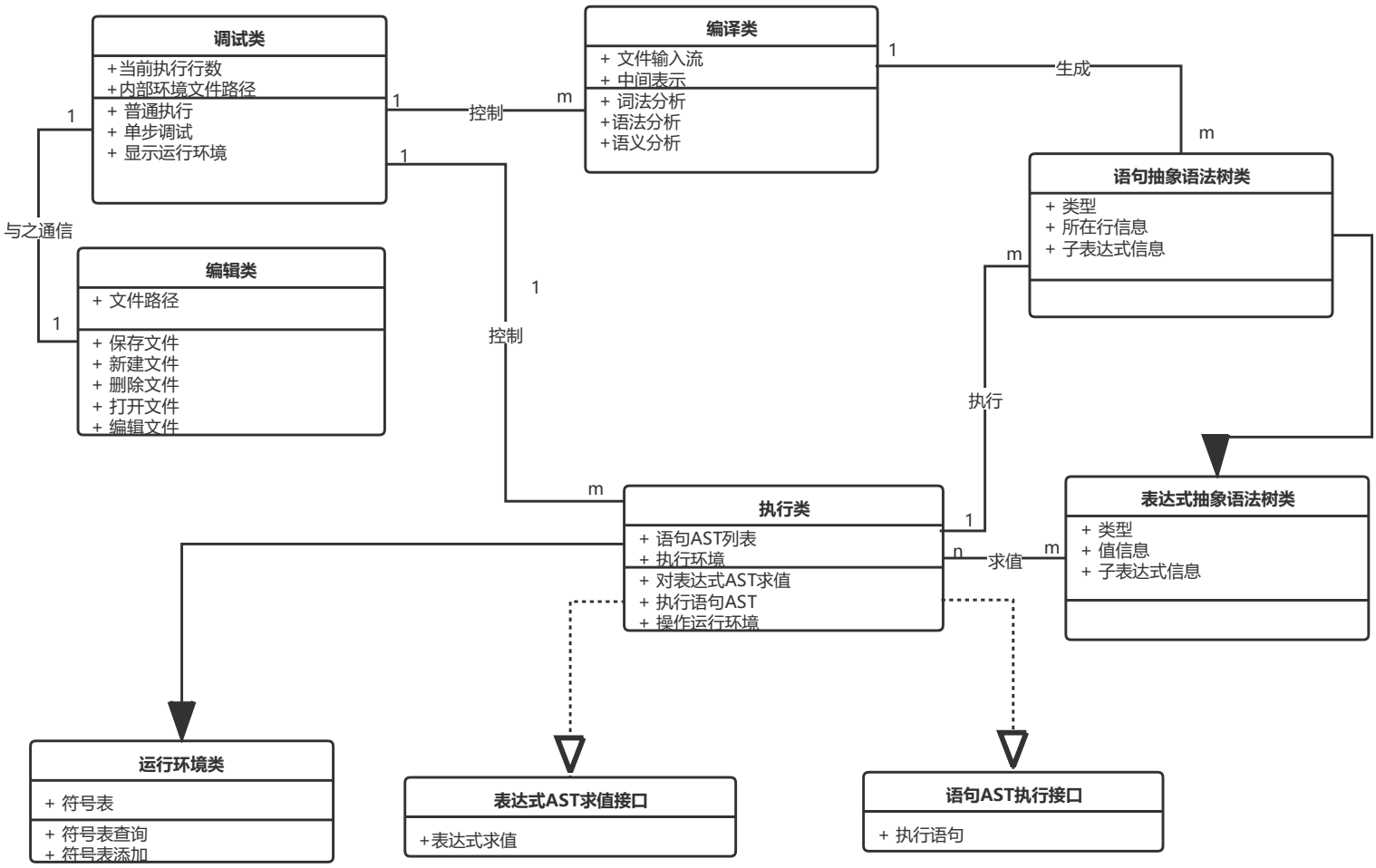

图 3.3 软件的类图建模

为了解释上文类图中，各个类的基本职责，以及各个类之间的协作关系，我们对以上几个关键的类做出了 CRC 卡片，呈现在下图 3.4.

图 3.4 主要类的 CRC 卡片

3.1.3 执行顺序图

基于前两节的用例模型以及对象模型，这里提出解释器的基本执行顺序图，用于揭示本解释器在用户角度的基本功能。

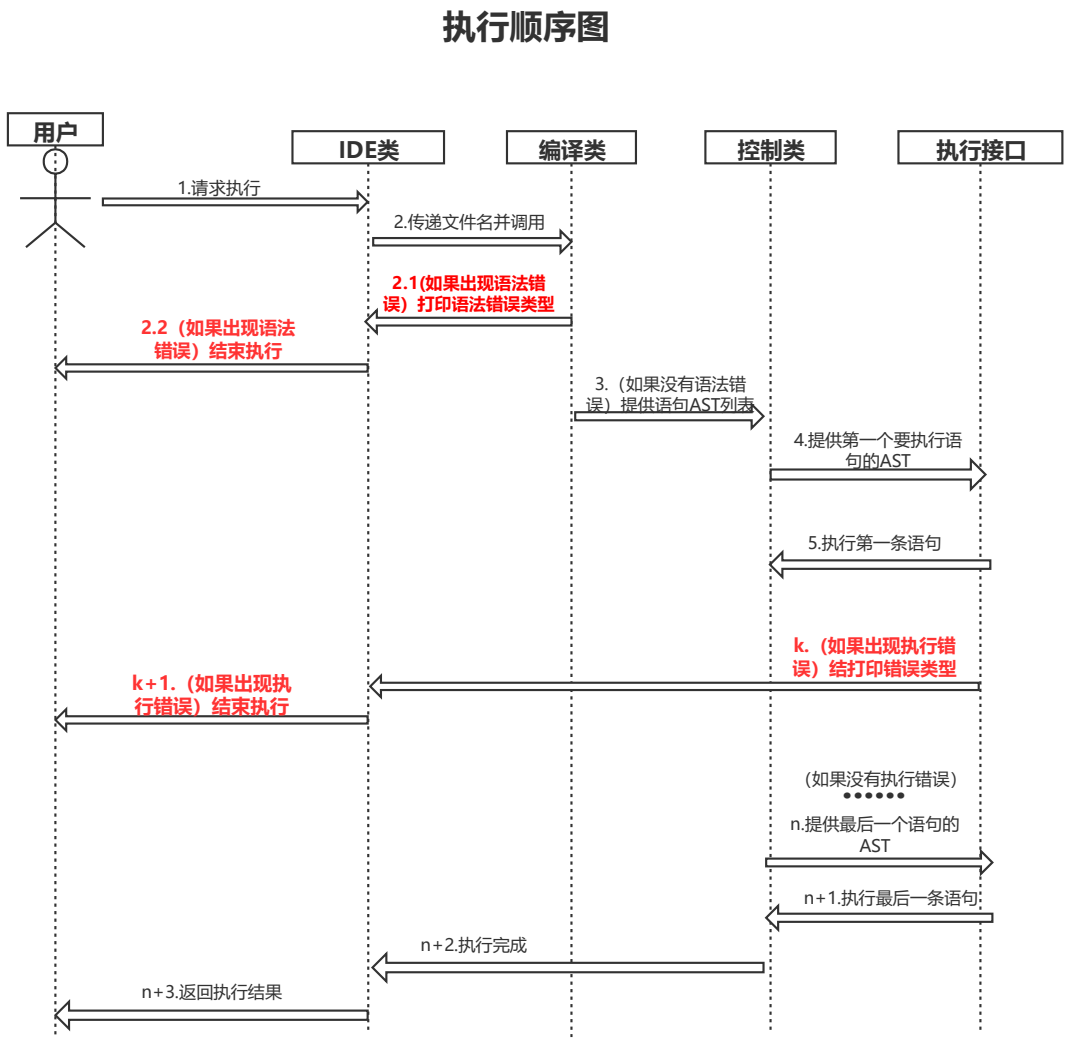

图 3.5 执行顺序图

执行文件部分是本软件的核心部分。

当用户提交执行请求，IDE 类会将当前要执行的文件名传递给编译类，编译类接收到文件后，对其进行编译，得到由本文件中所生成的所有 AST 构成的一个 AST 数组。

编译类将该 AST 数组传递给控制类，控制类根据控制规则，决定将下一条要执行语句的 AST 传给执行接口，待执行接口执行完毕后，再将下一条要执行语句的 AST 传给执行接口。重复传递直至程序执行完毕。在执行过程中，如果用户要求单步调试，则每当执行完一条语句，就展示一次当前符号表里的内容。否则直接展示文件执行的最终结果。

3.1.4 动态模型

本节揭示了动态模型，用来揭示各个模块之间的动态依赖关系。

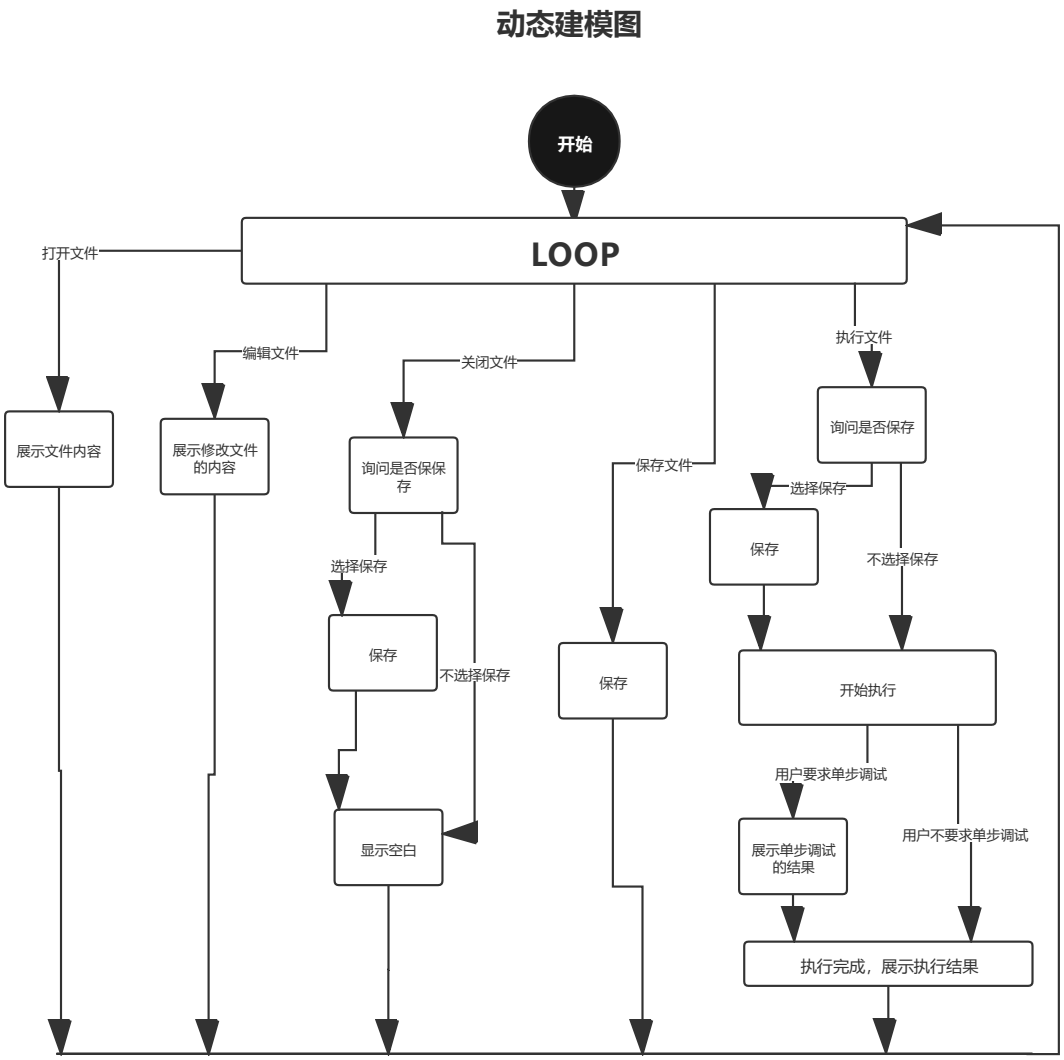

图 3.6 动态建模图

针对文件操作，会产生四种状态：1.当用户要求打开文件，本软件展示文件的内容，并回到 LOOP 起点。2.当用户编辑文件，本软件会展示用户编辑文件后的结果，并回到 LOOP 起点。3.当用户关闭文件，本软件会显示空白，并回到 LOOP 起点。4.保存状态不仅应用于用户点击，还出现在文件关闭、执行之前，以保证 IDE 操作的安全性。

针对执行操作，有两种可能的情况：1.如果用户不要求展示中间结果，则直接展示执行完文件的结果，并回到 LOOP 起点。2.如果用户要求单步调试展示中间结果，则用户每点击一次“下一步”，软件的相应界面上就展示当前符号表的内容，直至文件执行完成，并回到 LOOP 起点。

3.2 对性能的规定

3.2.1 精度

集成开发环境在输出时要具有准确性，能够正确并清晰地展示运行信息、错误信息、环境信息。

3.2.2 时间特性要求

1.解释器对源文件的执行速度能够接近 Python 语言。
2.集成开发环境中文件操作的响应时间不超过 3s.
3.编辑器中的代码高亮等特性（如果实现）的响应时间在用户可接受的范围内。

3.2.3 灵活性

将会以Java打包文件的形式发布，能够运行在具有JVM环境的所有设备上。

3.3 输入输出要求

项目	要求	具体说明
输入	Pythy 源文件，由 utf-8 编码的字符串	由用户通过 IDE 编辑或导入
输出 1	控制台的输出，以字符串的形式	用户在调试过程中，随着执行的进行而变化
输出 2	内部环境的输出，以图片的形式	展示符号表，由 graphviz 产生的图片

表 3.1 输入输出要求表格

4 运行环境规定

4.1 设备

本软件运行在所有支持 Java 虚拟机（JVM）且具有文件系统的设备上

4.2 接口

本软件需要 Java 运行时支持，同时需要在运行过程中调用系统的命令行。

4.3 控制

本软件的 IDE 是一个图形化界面窗口，需要使用鼠标来选择文件的操作或编译或执行，同时需要获取键盘输入来编辑源文件。

参考文献

[1] 2017HYSE04 小组 《软件工程选题报告》，2020.
[2] Stephen R.Schach 《软件工程 面向对象和传统的方法 原书第 8 版 中文版》[M]. 第 8 版. 北京: 机械工业出版社, 2011.