# 软件代码度量工具系统需求规格说明书

**目录**

一、项目介绍

1.1 需求规格说明的目的

1.2 软件产品的作用范围

二、一般性描述

2.1 运行环境

2.1.1 系统软件运行环境

2.1.2 系统硬件运行环境

2.2 产品功能

2.3 限制与约束

2.3.1软件功能约束：

2.3.2软件系统开发约束

三、具体需求

3.1 功能或行为需求

3.2 性能需求

3.3 项目架构

3.3.1、B/S架构

3.2.2、MVC设计模型

**正文**

**1. 项目介绍**

**1.1 需求规格说明的目的**

软件代码度量工具需求规格说明书，是软件开发者及分析人员根据系统用户提出的需求对本项目做出详细的需求分析，进一步定制软件开发的细节问题，为了使用户、软件开发者及分析和测试人员对该软件的初始规定有一个共同的理解，它说明了本软件的各项功能需求、性能需求和数据需求，明确标识各项功能的具体含义，为开发人员、维护人员及用户之间提供共同的协议以保证开发任务顺利并行地开展，同时进行建模。规格文档根据功能需求、性能需求和数据提出的要求进行说明，明确背景及用户群范围，为用户解决问题，实现所需目标创造条件。

**1.2 软件产品的作用范围**

随着软件质量需求的不断提出，软件度量作为其中的重要技术支持越发受到人们的关注，大量的人力物力在研究和应用方面的投入，使得软件度量技术获得了长足的进展，并成为软件工程研究中热点方向之一。

软件代码度量，即是对一些影响软件测试的属性进行定量的测量。通过在测试过程中收集到的相关数据，可以规划項目的可视化程度，帮助测试员理解当前的任务，为软件项目的管理提供可靠数据。在这其中，面向对象的软件度量是传统的软件度量在面向对象软件中的应用，由于面向对象引入了重载、继承、多态等新机制，在软件分析、设计、实现方面带来新特征，从而产生了许多针对面向对象的软件度量模型，这些模型从不同角度来分析对象间的耦合、类/(子)系统的内聚、实体复杂度和实体间关系复杂度以及重用性等。

为了更好的衡量代码的质量属性，我们希望开发相关的代码度量工具，能实现对这些统计量的分析输出，使得开发人员可以利用这部分数据去评估和修正自己的代码。

**2. 一般性描述**

**2.1 产品与其环境之间的关系**

**2.1.1 系统软件运行环境**

（1）系统建模采用支持UML的Microsoft Visio2010开发环境。

（2）系统实现语言采用JavaWeb语言编程。

（3）数据管理采用MySQL Community Server数据库管理系统。

（4）系统操作平台采用微软操作系统Windows 7系统。

（5）客户机运行环境：Windows 7 或更高版本。

**2.1.2 系统硬件运行环境**

服务器运行环境：

内存1GB以上，处理器AMD4800+以上，高速硬盘主400GB，高速缓存液晶显示器。

用户端运行环境：

采用内存1GB以上，硬盘160GBPersonal Computer。网络：遵守802.11标准建立的局域网

**2.2. 产品功能**

（1）用户登录

（2）提交源代码

（3）代码度量分析

（4）结果返回

**2.3 限制与约束**

**2.3.1软件功能约束：**

（1）语言识别限制：开发初期只能识别C++语言，后期可能迭代出识别面向对象的语 言，如果时间等各方面成本足够，可以接着实现面向过程等语言。

（2）度量属性限制：本软件目前只能做到识别代码的一些基本属性，例如行数、函数 数等简单属性，还做不到能识别出代码的语法或词法错误。

（3）运行环境限制：本项目基于B/S架构，需要用户在网页端自行提交代码。

**2.3.2软件系统开发约束**

（1）软件系统要求用户操作简便，熟悉操作界面，可扩充，易于维护。

（2）开发期限：24周以内。

**3. 具体需求**

**3.1 功能或行为需求**

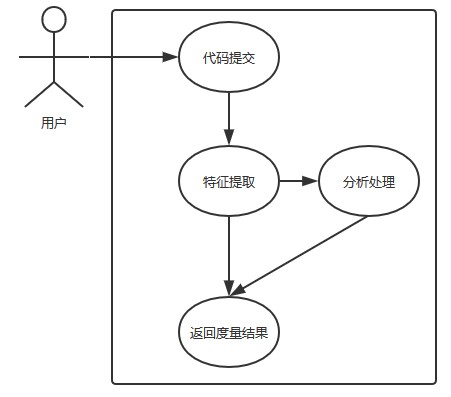
（1）数据收集和处理：在进行度量分析前通过B/S架构获取用户提交的代码信息；

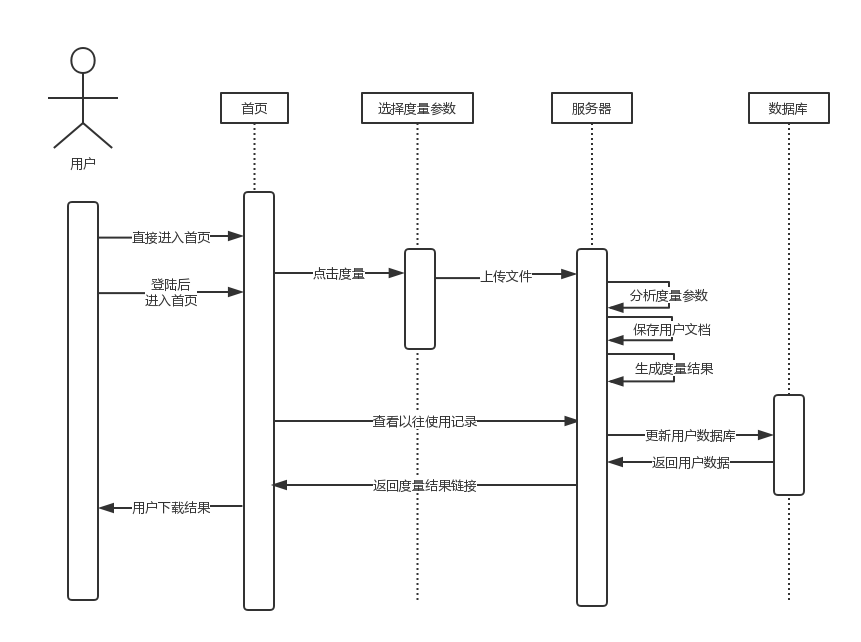
（2）代码度量：数据数据收集之后，针对不同语言，进行词法分析和语法分析；再进行对象特征 的抽取，比如行数、注释量、方法数、圈复杂度总数、最大继承深度等参数；

（3）分析评价：对已有的指标结果经过分析处理，给出综合软件属性评价；

（4）结果返回：将分析结果返回网页呈现给用户，等待下一次代码度量。

1. 系统用例图：



（6）系统顺序图

**3.2性能需求**

软件代码度量工具可以返回给出代码的一些度量指标，下面列出本项目应完成的项目代码度量目标属性。

1、方法总数（Ms）

对于一个类来说，它的集合中包括数据成员和数据函数，但是成员函数对于个体复杂性的影响并非是线性的，而且类的复杂性往往更明显的受到最大复杂度方法的影响，所以方法对于类的复杂度计算有着重要贡献。

2、最大继承深度（ID）

对于允许重继承的C++语言来说，继承树不是一颗简单的树，而是一个无环路图，我们把类的继承关系抽象成一棵树，以此计算出树的深度来表示代码的最大继承深度。

3、子类数目

由于多重继承的存在，子类数目的计算变得更加复杂，因为一个类可能作为一些类的第一父类，同时又是另一些类的第二父类，所以子类数目指标应该是两类子类数目之和。

4、圈复杂度

圈复杂度也称为条件复杂度或循环复杂度，用来表示程序的复杂度，其符号为 VG 或是 M。圈复杂度是对源代码中线性独立路径数的定量测量。例如：如果程序中不包含控制、判断、条件语句，那么复杂度就是 1 ，因为整个程序只有一条执行路径；如果程序包含一条IF语句，那么就会有两条路径来执行完整个程序（IF为 TRUE，IF 为 FALSE），所以这时候的复杂度就是 2；两个嵌套的 IF 语句，或者包含两个判断条件的一个 IF 语句，复杂度就是 4。

**3.3项目架构选取**

**3.3.1、B/S架构**

B/S架构即浏览器和服务器结构，它是基于C/S结构理论上的一种改进的结构。B/S三层架构就是将组成应用程序的功能以及资源划分为表现层、业务逻辑层、数据访问层。表现层负责与用户进行交互，Web服务器用HTTP 协议把所需要的文件资料传给用户，客户端接收文件信息并在浏览器上输出。业务逻辑层接收到浏览器端的请求，将处理好的请求转化为数据库请求传达给数据库服务器，再将这个交互结果传送给浏览器端，数据访问层直接操作数据库，对数据进行增、删、改、查等操作。B/S架构能够有效的在网络环境下组织和管理数据。

**3.2.2、MVC设计模型**

MVC模型是软件工程中的一种软件架构模式，它具有模型、视图和控制器三个核心模块，分别承担不同的任务。MVC模式的目的是实现一种动态程序设计，以便于修改和简化程序，重复利用程序的某一部分。其中M是Model（模型）是应用程序中用于处理应用程序数据逻辑的部分，通常模型对象负责在数据库中存取数据；V是View（视图）是应用程序中处理数据显示的部分，通常视图是依据模型数据创建的；C是Controller（控制器）是应用程序中处理用户交互的部分，通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据。

MVC 分层有助于管理复杂的应用程序，因为可以在一个时间内专门关注一个方面同时也让应用程序的测试更加容易。MVC 分层还简化了分组开发。不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。