区块链应用程序员
职业能力水平评价标准
（试行稿）

1　项目概况

1.1 项目名称

区块链应用程序员

1.2 项目定义

从事区块链项目的系统设计（系统结构图等）、编码（常用的编码工具如 Linux 下的 vi 等）、测试（常用的编译系统如 Linux 下的 g++等）和维护(常用的代码维护工具如 Git 等)的人员。

1.3 能力等级

本项目共设三个技能等级，分别为：初级、中级、高级。

1.4 能力特征

具备一定的组织、理解、判断能力，具备较强的学习能力、分析解决问题的能力、沟通能力和准确理解业务场景的能力等。理解区块链技术的特点、作用和应用场景。掌握区块链开发与维护的基本知识等。

1.5 职业能力水平评价要求

1.5.1 申报条件

具备以下条件之一者，可申报初级：

（1）累计从事相关职业工作1年（含）以上。

（2）相关专业在校学生。

具备以下条件之一者，可申报中级：

（1）取得本项目或相关职业初级评价证书（含职业资格证书、职业技能等级证书等）后，累计从事相关职业工作2年（含）以上。

（2）累计从事相关职业工作4年（含）以上。

（3）取得相关专业毕业证书。

具备以下条件之一者，可申报高级：

（1）取得本项目或相关职业中级评价证书（含职业资格证书、职业技能等级证书等）后，累计从事相关职业工作3年（含）以上。

（2）累计从事相关职业工作6年（含）以上。

（3）具有高等职业学校、高级技工学校、技师学院相关专业毕业证书，并取得本项目或相关职业中级评价证书（含职业资格证书、职业技能等级证书等）。

（4）具有大专及以上相关专业毕业证书，并取得本项目或相关职业中级评价证书（含职业资格证书、职业技能等级证书等）后，累计从事相关职业工作1年（含）以上。

1.5.2 申报条件注释

（1）满足本项目高级别申报条件可申报本项目低级别；

（2）相关职业：信息通信网络运行管理员、网络与信息安全管理员、信息通信信息化系统管理员、信息安全测试员、计算机程序设计员、计算机软件测试员、计算机软件工程技术人员、计算机网络工程技术人员、云计算工程技术人员、区块链工程技术人员等；

（3）相关专业：

技工院校电工电子类、信息类、财经商贸类等专业；

中等职业学校专业目录中加工制造类、信息技术类、财经商贸类等专业；

高等职业学校专业目录中装备制造大类、电子信息大类、财经商贸大类等专业；

普通高等学校本科专业目录中自动化类、电子信息类、计算机类、金融学类、经济与贸易类等专业。

1.5.3 评价方式

职业能力水平评价考试包括理论知识、技能操作两个科目，较高等级必要时可增加综合评审。

理论知识考试以笔试为主，可以机考，条件成熟时试点开展网络考试，主要考核从业人员从事本职业应掌握的基本要求和相关知识要求。技能操作考核主要采用现场操作、模拟操作、面试答辩等方式进行，主要考核从业人员从事本职业应具备的技能水平。综合评审通常采取审阅申报材料、技术答辩等方式进行全面

评议和审查。理论知识考试和技能操作考核均采用百分制，成绩达到 60 分以上者为合格。

1.5.4 监考人员、考评人员与考生配比

理论知识考试和技能操作考核中的监考人员与考生配比不低于 1：15，且每个考场不少于 2 名监考人员。技能操作考核中考评人员与考生配比为 1:4～1:6，且为 3 人以上单数。

1.5.5 评价时间

理论知识考试时间不少于90分钟；技能操作考核时间不少于90分钟。

1.5.6 评价场所设备

理论知识考试在标准教室内进行；技能操作考核根据各模块的工作要求，在配备有计算机的场所进行。

2 基本要求

2.1 职业道德

（1）忠于职守，精益求精；

（2）实事求是，诚实守信；

（3）依法行事，严守秘密；

（4）公正透明，服务社会。

2.2 基础知识

2.2.1 计算机网络

（1）计算机网络的定义

（2）计算机网络的分类

（3）计算机网络的性能

（4）网络协议的层次与划分

（5）TCP/IP 协议的基本架构

2.2.2 操作系统

（1）操作系统的概念

（2）操作系统的架构

（3）Linux 操作系统的运行原理

（4）Windows 操作系统的运行原理

2.2.3 分布式系统

（1）分布式系统的定义

（2）分布式系统的 CAP 理论

（3）分布式系统的设计要点

（4）分布式系统的通信模式

（5）分布式系统的一致性

（6）容错性和安全性

2.2.4 密码学

（1）加密技术的分类

（2）对称加密的概念

（3）非对称加密的概念

（4）椭圆曲线的定义

（5）椭圆曲线的特点

（6）SHA 家族算法

（7）数字签名的定义

（8）数字签名的流程

（9）公钥与私钥的概念

2.2.5 区块链基础知识

（1）比特币发展史简介

（2）比特币基础概念、运行原理

（3）以太坊发展史简介

（4）以太坊基础概念、运行原理

（5）共识机制的定义、分类

（6）常用共识机制的特点

（7）比特币及以太坊使用的加密算法

3 工作要求

本标准对初级、中级、高级各级别的技能要求依次递进，高级别涵盖低级别的要求。

3.1 初级

职业功能	工作内容	技能要求	相关知识要求
1.系统分析	1.1 需求分析	1.1.1 能够分析项目的应用场景、性能需求和实现目标 1.1.2 能够参与撰写需求分析文档	1.1.1 软件需求分析的步骤 1.1.1 软件需求分析的要点
	1.2 软件结构分析	1.2.1 了解子系统，确定子系统间的接口和通信方式 1.2.2 了解一种或多种公有链（比特币、以太坊等）及联盟链系统（Hyperledger 等）中各子模块（如共识模块、点对点通信模块、加密算法模块等）之间的功能	1.2.1 一种或多种常见软件体系结构（B/S、C/S 等）及其特点和划分 1.2.2 一种或多种公有链（比特币、以太坊等）及联盟链系统（Hyperledger 等）中各子模块（如共识模块、点对点通信模块、加密算法模块等）之间的接口及通信方式
	1.3 数据结构分析	1.3.1 能够根据需求了解全局变量、局部变量 1.3.2 能够了解常见的数据结构（树，图，链表，堆栈等）	1.3.1 变量的多种作用范围（全局变量、局部变量） 1.3.2 常见的数据结构（树，图，链表，堆栈等）
	1.4 整体系统架构分析	1.4.1 了解系统的整体功能及各项子功能 1.4.2 了解系统整体的输入和输出	1.4.1 系统整体功能和子功能的划分 1.4.2 系统输入和输出的判断方法
2.编码	2.1 选择操作系统	2.1.1 能够操作一种或多种操作系统（如 Linux、Windows 等） 2.1.2 能够在操作系统上安装常用的编程软件（如 git、g++）	2.1.1 常用的操作系统（如 Linux、Windows 等）的操作指南、技巧 2.1.2 常用的编程应用软件（如 git、g++）的安装、配置
	2.2 选择编码语言	2.2.1 能够使用一种或多种常用编程语言编程（如 C++、Java、Python、Go、Rust 等） 2.2.2 能够编译运行常见编程语言（如 C++、Java、Python、Go、Rust 等）编写的代码	2.2.1 常用的编程语言（如 C++、Java、Python、Go、Rust 等）的基础知识 2.2.2 常用的编程语言（如 C++、Java、Python、Go、Rust 等）在编译运行中的问题排查和解决方案

	2.3 选择开发环境	2.3.1 能够安装配置一种或多种常用的代码管理系统（如 Github、Gitlab、SourceForge、SourceSafe 等） 2.3.2 能够使用一种或多种常用的代码管理系统（如 Github、Gitlab、SourceForge、SourceSafe 等）	2.3.1 常用的代码管理系统（如 Github、Gitlab、SourceForge、SourceSafe 等)的安装及配置流程 2.3.2 常用的代码管理系统（如 Github、Gitlab、SourceForge、SourceSafe 等）的使用方法
	2.4 选择编码风格	2.4.1 熟悉常见编程语言（如 C++、Java 等）及操作系统（Unix/Linux）的编码风格 2.4.2 能够阅读基本的英语技术文档	2.4.1 常用的 Unix/Linux 编码风格 2.4.2 C++、Java 等编码风格 2.4.3 基本的英语单词、语法及计算机专业的基本词汇
3.测试	3.1 编写测试文档	3.1.1 能够根据系统设计文档理解系统测试的目标 3.1.2 能够解析测试文档的目标、构成和要素	3.1.1 软件工程对测试流程的定义 3.1.2 测试文档的目标、构成和要素
	3.2 功能测试	3.2.1 能够完成功能测试的一般流程和步骤 3.2.2 能够判断功能测试的有效性	3.2.1 软件工程对功能测试一般流程和步骤的定义 3.2.2 软件工程对功能测试完成标准的定义
	3.3 单元测试	3.3.1 能够完成单元测试的一般流程和步骤 3.3.2 能够判断单元测试的有效性	3.3.1 软件工程对单元测试一般流程和步骤的定义 3.3.2 软件工程对单元测试完成标准的定义
	3.4 集成测试	3.4.1 能够完成集成测试的一般流程和步骤 3.4.2 能够判断集成测试的有效性	3.4.1 软件工程对集成测试一般流程和步骤的定义 3.4.2 软件工程对集成测试完成标准的定义
	3.5 系统测试	3.5.1 能够完成系统测试的一般流程和步骤 3.5.2 能够判断系统测试的有效性	3.5.1 软件工程对系统测试一般流程和步骤的定义 3.5.2 软件工程对系统测试完成标准的定义
4.维护	4.1 软件系统维护	4.1.1 能够根据系统维护流程进行日常的软件维护操作（如系统恢复、数据备份等） 4.1.2 能够进行常用的操作系统（如 Windows、Linux 等）维	4.1.1 软件工程对系统维护的定义、流程和方法 4.1.2 常用的操作系统（如 Linux、Windows）的安装、更新和修复的步骤及方法

		护（如安装、更新、修复等） 4.1.3 能够了解常用的网络协议	4.1.3 常用的网络协议（如TCP/IP、P2P 网络等）
	4.2 硬件系统维护	4.2.1 能够根据系统维护流程进行日常的硬件维护操作（如机械故障的排除，电器故障的排除等） 4.2.2 能够进行常用的服务器设备的维护（如安装、升级、维修等） 4.2.3 了解常用网络设备的运行原理	4.2.1 项目硬件设备对硬件维护的流程和方法 4.2.2 常用的服务器设备的维护（如安装、升级、维修等）方法 4.2.3 常用网络设备的运行原理
	4.3 系统代码维护	4.3.1 能够根据系统维护流程进行日常的代码操作（如下载代码、修改代码、提交代码等） 4.3.2 能够进行日常的代码维护（如编译、运行等）	4.3.1 软件工程对系统代码维护的定义、流程和方法 4.3.2 日常代码维护（如编译、运行等）的定义、流程和方法

3.2 中级

职业功能	工作内容	技能要求	相关知识要求
1.系统分析	1.1 需求分析	1.1.1 能够发掘项目的潜在应用场景、性能需求和目标 1.1.2 能够画业务逻辑图	1.1.1 软件需求文档编写的一般规范、要素和写作方式 1.1.2 常用的业务逻辑图工具（如 Visio、STARUML 等）的使用方法
	1.2 软件结构分析	1.2.1 能够划分子系统的模块元素 1.2.2 能够根据软件需求划分每个模块的功能 1.2.3 能够画软件的流程图	1.2.1 软件工程中的模块可重用技术 1.2.2 常用编程语言（如 C++、Java 等）中的模块设计方法 1.2.3 常用的软件流程图工具（如 Visio、STARUML 等）的使用方法
	1.3 数据结构分析	1.3.1 能够编写常见的算法（搜索、排序、遍历等） 1.3.2 能够定义常见公有链（如比特币、以太坊）及联盟链（如 Hyperledger）中的区块链数据结构（如 Merkle 树、UTXO 等）	1.3.1 常见的算法（搜索、排序、遍历等） 1.3.2 常见的哈希算法（如 SHA256 等） 1.3.2 公有链（如比特币、以太坊）及联盟链（如 Hyperledger）

			中的区块链数据结构（如 Merkle 树、UTXO 等）
	1.4 设计整体系统架构	1.4.1 能够根据子功能分析组成系统的子系统 1.4.2 能够分析各子系统之间的控制、通信和数据关联等	1.4.1 软件工程关于子功能的定义和分析 1.4.2 常见的软件设计方法（如面向过程、面向对象、面向服务等） 1.4.3 了解常见的软件过程模型（如瀑布模型、原型模型、增量模型、敏捷模型、喷泉模型等）
2.编码	2.1 选择操作系统	2.1.1 能够掌握常见操作系统（如 Linux、Windows）在性能、安全性及使用上的特点 2.1.2 能够判别常见操作系统（如 Linux、Windows）适用的区块链	2.1.1 常用的操作系统（如 Linux、Windows 等）在性能、安全性及使用上的特点 2.1.2 常见操作系统（如 Linux、Windows）适用的区块链系统
	2.2 选择编码语言	2.2.1 能够优化常用编程语言（如 C++、Java、Python、Go、Rust 等）的代码 2.2.2 能够使用常用的智能合约编程语言（如 Solidity 等）编程	2.2.1 常用编程语言（如 C++、Java、Python、Go、Rust 等）的特点 2.2.2 常用的智能合约编程语言（如 Solidity 等）的基础知识
	2.3 选择开发环境	2.3.1 能够根据项目选择合适的代码管理系统 2.3.2 能够安装常见的编译环境（如 g++、gcc 等）	2.3.1 常用的代码管理系统（如 Github、Gitlab、SourceForge、SourceSafe 等）在安全性、操作性及使用场景上的优劣 2.3.2 常用编译环境（如 g++、gcc 等）的安装及配置方法
	2.4 选择编码风格	2.4.1 熟悉常见智能合约语言（如 Solidity）在变量命名、缩进格式、接口定义等方面的编码风格 2.4.2 能够书写基本的英文注释	2.4.1 常见智能合约语言（如 Solidity）在变量命名、缩进格式、接口定义等方面的规定 2.4.2 接口定义等方面的编码风格 2.4.3 计算机专业英文注释的写作规范
	2.5 编码实现	2.5.1 能够阅读智能合约源代码 2.5.2 能够编写简单的智能合	2.5.1 常用智能合约语言（如 Solidity 等）的语法基础知识 2.5.2 智能函数及模块的基本

		约函数及模块	知识
3.测试	3.1 编写测试文档	3.1.1 能够分析系统测试的方法和策略 3.1.2 能够编写测试用例	3.1.1 系统测试的方法和策略及要素 3.1.2 测试用例设计的要素和流程
	3.2 功能测试	3.2.1 能够解析一种或多种常用编程语言（如 C++、Java、Python、Go、Rust 等）编写的变量定义、语句和控制结构 3.2.2 能够解析一种或多种常用智能合约编程语言（如 Solidity 等）编写的变量定义、语句和控制结构等	3.2.1 一种或多种常用编程语言（如 C++、Java、Python、Go、Rust 等）编写的变量定义、语句和控制结构等 3.2.2 一种或多种常用智能合约编程语言（如 Solidity 等）编写的变量定义、语句和控制结构等
	3.3 单元测试	3.3.1 能够解析常用编程语言（如 C++、Java、Python、Go、Rust 等）编写的函数、接口等 3.3.2 能够解析常用智能合约编程语言（如 Solidity 等）编写的函数、接口等	3.3.1 常用的编程语言(如 C++、Java、Python、Go、Rust 等）编写的函数、接口等 3.3.2 常用的智能合约编程语言（如 Solidity 等）编写的函数、接口等
	3.4 集成测试	3.4.1 能够阅读常用编程语言（如 C++、Java、Python、Go、Rust 等）编写的数据结构、系统输入输出等 3.4.2 能够阅读常用智能合约编程语言（如 Solidity 等）编写的数据结构、系统输入输出等	3.4.1 常用的编程语言(如 C++、Java、Python、Go、Rust 等）编写的数据结构、系统输入输出等 3.4.2 常用的智能合约编程语言（如 Solidity 等）编写的数据结构、系统输入输出等
	3.5 系统测试	3.5.1 能够阅读常用编程语言（如 C++、Java、Python、Go、Rust 等)编写模块或子系统之间的调用关系及接口之间的输入输出 3.5.2 能够阅读常用智能合约编程语言（如 Solidity 等）编写模块或子系统之间的调用关系及接口之间的输入输出	3.5.1 常用的编程语言(如 C++、Java、Python、Go、Rust 等）编写模块或子系统之间的调用关系及接口之间的输入输出 3.5.2 常用的智能合约编程语言（如 Solidity 等）编写模块或子系统之间的调用关系及接口之间的输入输出
4.维护	4.1 软件系统维护	4.1.1 能够对常用数据库（如 Oracle, MySQL 等）进行日常的维护操作（如数据备份、管理、	4.1.1 常用数据库（如 Oracle, MySQL 等)日常维护操作方法(如数据备份、管理、更新等）

		更新等） 4.1.2 能够根据需求对软件系统的变更执行指定的操作 4.1.3 能够进行常用的网络维护	4.1.2 软件工程对软件系统变更流程的定义、要求等 4.1.3 常用的网络维护命令（如ping、ipconfig 等）
	4.2 硬件系统维护	4.2.1 能够对数据库系统硬件进行日常的维护操作（如安装、升级、维修等） 4.2.2 能够根据需求对硬件系统的变更执行指定的操作 4.2.3 能够对网络系统硬件设备进行维护操作	4.2.1 数据库系统硬件日常的维护操作（如安装、升级、维修等）方法 4.2.2 硬件手册对硬件维护定义的操作、流程和方法 4.2.3 网络系统硬件设备维护的操作、流程和方法
	4.3 系统代码维护	4.3.1 能够对数据库系统相关代码进行日常的运行维护操作（如监控、性能测试） 4.3.2 能够根据需求对系统代码的变更执行指定的操作	4.3.1 数据库系统进行日常运行维护操作（如监控、性能测试）的定义、流程和方法 4.3.2 软件工程对系统代码的变更流程的定义、要求等
5.培训、指导与管理	5.1 培训	5.1.1 能对本职业中级及以下人员进行理论、技能培训 5.1.2 能编写区块链系统分析、编码、测试和维护类培训讲义	5.1.1 培训教学方法 5.1.2 培训讲义编写方法

3.3 高级

职业功能	工作内容	技能要求	相关知识要求
1.系统分析	1.1 需求分析	1.1.1 分析需求的可行性 1.1.2 能够独立撰写需求分析文档 1.1.3 会使用建模工具	1.1.1 项目可行性的判断方法、要点和流程 1.1.2 软件需求文档编写的一般规范，要素和写作方式 1.1.3 常用的建模工具的使用方法（如UML等）
	1.2 软件结构分析	1.2.1 能够定义模块接口和模块接口的数据结构 1.2.2 能够评估软件结构质量，进行结构优化	1.2.1 软件工程对模块接口及数据结构的定义和设计建议 1.2.2 软件工程中模块的常见连接方式（如非直接耦合、数据耦

			合、印记耦合、控制耦合、外部耦合、公共耦合、内容耦合）等
	1.3 数据结构分析	1.3.1 能够定义输入、输出文件的结构 1.3.2 能够定义区块链系统数据库中的表结构和视图结构	1.3.1 输入输出文件的特点、使用场景和使用方法 1.3.2 常见关系数据库（MySQL、Oracle、SQL Server等）中的表结构和视图结构
	1.4 整体系统架构分析	1.4.1 能够分析区块链系统的性能 1.4.2 能够根据项目需求定义设计模式 1.4.3 能够分析公有链、联盟链的整体架构	1.4.1 系统整体性能的概念、要点和分析方法 1.4.2 常见的设计模式（如单例模式、策略模式、代理模式、观察者模式、装饰模式、适配器模式、命令模式、组合模式、简单工厂模式） 1.4.3 分布式系统架构及特点 1.4.3 常见的公有链（比特币、以太坊等）及联盟链系统（Hyperledger等）的整体架构 1.4.3 常见的共识机制（如PoW、PoS、DPoS、PBFT）
2.编码	2.1 选择操作系统	2.1.1 能够根据区块链项目需求选择合适的操作系统 2.1.2 能够在常用操作系统下（如Linux、Windows等）运行常见区块链（如比特币、以太坊等）项目的节点	2.1.1 常用的操作系统（如Linux、Windows等）在常见区块链系统（如比特币、以太坊等）中的运行性能及优劣势 2.1.2 常见区块链（如比特币、以太坊等）在常用操作系统（如Linux、Windows等）下的安装及运行方法

	2.2 选择编码语言	2.2.1 能够比较区分常见区块链公链（如比特币、以太坊）用不同语言实现的客户端 2.2.2 能够优化常用智能合约语言（如Solidity）编写的代码	2.2.1 常见公链（如比特币、以太坊）不同语言客户端程序的基本知识 2.2.2 常用的智能合约编程语言（如Solidity等）的特性
	2.3 选择开发环境	2.3.1 能够优化常见的编译环境（如g++、gcc等） 2.3.2 能够根据不同区块链项目的需求选择编译环境 2.3.3 能够使用常用的智能合约编译工具（如IDE Remix等）	2.3.1 常见编译环境（如g++、gcc等）的使用方法 2.3.2 常见编译环境（如g++、gcc等）的适用场景、安全性及操作性特点 2.3.3 常用的智能合约编译工具（如IDE Remix等）的使用方法
	2.4 选择编码风格	2.4.1 能够制定适合区块链项目的编码风格 2.4.2 能够用区块链相关专业术语书写英文说明文档	2.4.1 Unix/Linux编码风格、常见编程语言（如C++、C、Java）编码风格的优劣和适用场合 2.4.2 计算机专业英文文档书写规范那、区块链相关专业术语
		2.5.1 能够阅读一种或多种常见公链（如比特币、以太坊等）的源代码 2.5.2 能够编写完整的智能合约	2.5.1 C++、go语言的语法知识 2.5.2 常用智能合约语言的编写方法
3.测试	3.1 编写测试文档	3.1.1 能够编写完整的测试文档（包括功能测试、单元测试、集成测试和系统测试等） 3.1.2 能够根据相关标准制订测试文档的书写规范	3.1.1 完整测试文档（如功能测试、单元测试、集成测试、系统测试等）的要点和构成 3.1.2 软件文档的国家标准（如GB8567等）

	3.2 功能测试	3.2.1 能够根据给定的功能测试文档执行测试任务 3.2.2 能够编写功能测试报告	3.2.1 功能测试的目的、方法和流程 3.2.2 功能测试报告的撰写方法
	3.3 单元测试	3.3.1 能够根据给定的单元测试文档执行测试任务 3.3.2 能够编写单元测试报告	3.3.1 单元测试的目的、方法和流程 3.3.2 单元测试报告的撰写方法
	3.4 集成测试	3.4.1 根据给定的集成测试文档执行测试任务 3.4.2 能够编写集成测试报告	3.4.1 集成测试的目的、方法和流程 3.4.2 集成测试报告的撰写方法
	3.5 系统测试	3.5.1 根据给定的系统测试文档执行测试任务 3.5.2 能够编写系统测试报告	3.5.1 系统测试的目的、方法和流程 3.5.2 系统测试报告的撰写方法
4.维护	4.1 软件系统维护	4.1.1 能够撰写软件系统变更登记表 4.1.2 能够撰写软件系统的日常维护日志 4.1.3 能够撰写网络系统的日常维护日志	4.1.1 软件系统变更登记表的撰写方法 4.1.2 软件系统的日常维护日志的撰写方法 4.1.3 网络系统日常维护日志的撰写方法
	4.2 硬件系统维护	4.2.1 能够撰写硬件系统变更登记表 4.2.2 能够撰写硬件系统的日常维护日志 4.2.3 能够撰写网络硬件系统的日常维护日志	4.2.1 硬件系统变更登记表的撰写方法 4.2.2 硬件系统的日常维护日志的撰写方法 4.2.3 网络硬件系统日常维护日志的撰写方法
	4.3 系统代码维护	4.3.1 能够撰写系统代码变更登记表 4.3.2 能够撰写系统代码的日常维护日志	4.3.1 系统代码变更登记表的撰写方法 4.3.2 系统代码的日常维护日志的撰写方法

5.培训、指导与管理	5.1 培训	5.1.1 能对本职业高级及以下人员进行理论、技能培训 5.1.2 能制订区块链系统分析、编码、测试和维护类培训方案	5.1.1 培训大纲编写方法 5.1.2 培训方案制订方法

4　权重表

4.1 理论知识权重表

项目		技能等级		
		初级	中级	高级
		（%）	（%）	（%）
基本要求	职业道德	5	5	5
	基础知识	15	10	5
相关知识	系统分析	10	15	25
	编码	10	25	25
	测试	30	20	10
	维护	30	15	10
培训指导	培训	—	5	10
	指导	—	5	10
合计		100	100	100

4.2 技能要求权重表

项目		技能等级		
		初级	中级	高级
		（%）	（%）	（%）
技能要求	系统分析	15	25	40
	编码	25	30	30
	测试	30	20	5
	维护	30	15	5
培训指导	培训	—	5	10
	指导	—	5	10
合计		100	100	100