**测试覆盖表**

# 一．对node本身测试用例覆盖表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能性测试 | 测试用例名称 | 测试用例标识 | 对应需求文档的需求 | 测试覆盖情况 |
| 创建模块 | NO.201 | 需求文档V1.1.0的2.1.2节 | 完全覆盖（即所有分支都测到） |
| 加载模块 | NO.202 | 需求文档v1.1.0的2.2.2节 | 完全覆盖 |
| 创建包 | NO.203 | 需求文档v1.1.0的2.3.2节 | 完全覆盖 |
| 加载包 | NO.204 | 需求文档v1.1.0的2.4.2节 | 完全覆盖 |
| 管理包 | NO.2051-NO.2056 | 需求文档v1.1.0的2.5.2节 | NO.2056发布包因需要注册npm账号没有进行测试 |
| 构建TCP服务器 | NO.206 | 需求文档v1.1.0的2.6.2节 | 分支中RFS5 显示内容不是服务器端内容 无法测试，只有网络错误才可能出现 |
| 构建UDP服务器 | NO.207 | 需求文档v1.1.0的2.7.2节 | 分支中RFS5 显示内容不是服务器端内容 无法测试，只有网络错误才可能出现 |
| 构建HTTP服务器 | NO.208 | 需求文档v1.1.0的2.8.2节 | 分支中RFS5 显示内容不是服务器端内容 无法测试，只有网络错误才可能出现 |
| 构建WebSocket服务器 | NO.209 | 需求文档v1.1.0的2.9.2节 | 分支中RFS7 显示内容不是服务器端内容 无法测试，只有网络错误才可能出现 |
| 删除文件测试 | NO.210 | 需求文档v1.1.0的2.10.2节 | RFS5 删除文件文件依然存在 异常没有办法测试 |
| 读取文件测试 | NO.211 | 需求文档v1.1.0的2.11.2节 | 完全覆盖 |
| 文件重命名测试 | NO.212 | 需求文档v1.1.0的2.12.2节 | 完全覆盖 |
| 文件写入测试 | NO.213 | 需求文档v1.1.0的2.13.2节 | 完全覆盖 |
| 创建目录测试 | NO.214 | 需求文档v1.1.0的2.14.2节 | 完全覆盖 |
| 删除目录测试 | No.215 | 需求文档v1.1.0的2.15.2节 | 完全覆盖 |
| 非功能性测试 | 兼容性测试 | NO.301 | 需求文档v1.1.0的3.1.2节 | 在linux和windows上均测试了所有功能模块 |
| 容错性测试 | NO.303 | 需求文档v1.1.0的3.3.2节 | 分支全部覆盖，但是测试数据无法覆盖到所有错误，只是挑选了代码错误和删除不存在文件这两个。 |
| 可扩展性测试 | NO.304 | 需求文档v1.1.0的3.4.2节 | 在创建模块加载模块的过程已经完成此测试，另外对Node改进也是可扩展性测试一部分。 但未尝试编写C++模块来进行扩展，只是扩展JS模块。 |

# 二．测试用例覆盖分析

## 2.1 功能测试分析

在功能测试方面，测试用例覆盖到了所有的需求用例，但单个测试用例有些异常分支没有测试到，比如发布包的功能因为需要npm账号，考虑到编写一个空包对npm仓库的污染问题未测试，再比如网络通信的测试用例要浏览器显示内容与服务器端内容不符这种异常确实存在，但一般出现在网络传输出错的情况，由于node并未实际部署到服务器一直部署在本机，所以不存在网络传输丢失数据的问题，也没有覆盖到。具体的异常分支覆盖情况请参考第一章测试用例覆盖表。

就我们测试的情况来看，node对我们所有的测试在正常情况下都能正确执行，对异常情况也都有对应的错误提示。

由于网络通信部分测试的局限性，我们未发现node自带的网络通信服务器对于网络传输情况不佳情况的问题，不知道其是否能够像tomcat这种独立的服务器一样给出类似404这种错误代码以及错误页。

## 2.2 非功能性分析

在非功能性测试方面，需求中包含兼容性，高效性，容错性，可扩展性四个部分，我们集中精力测试了兼容性，容错性和可扩展性三个非功能性需求。

### 2.2.1 兼容性

兼容性测试方面，因为node兼容windows和linux系统，windows只选用了最新win10进行测试，而linux版本众多，也只用了ubuntu进行测试。尽管将所有测试用例在windows和linux上均进行了测试，并且测试都已通过，但是还是无法保证node对所有系统的兼容。尤其是linux系统众多，redhat,fedora,centOS这些都无法保证兼容。

### 2.2.2 容错性

容错性测试方面，分支全部覆盖，但是测试数据无法覆盖到所有错误，只是挑选了代码错误和删除不存在文件这两个。没有发现node对于其他方面容错的问题。

### 2.2.3 可扩展性

在创建模块加载模块的过程已经完成此测试，另外对Node改进也是可扩展性测试一部分。 但未尝试编写C++模块来进行扩展，只是扩展JS模块。没有发现node对于C++模块扩展的问题。

# 三．改进部分测试

核心模块用十二个单元测试覆盖到了所有的语句，在正确使用的情况下可以正确的完成异步过载保护的任务。测试用例也包括了关闭限流功能，函数未添加回调函数的异常情况，在这些异常情况下均做出了正确的处理。在队列已满的情况下成功触发了Full事件。在设置了超时时间的情况下，有异步调用超时的话也正确给出了超时的报错，并且继续执行了后续的调用。

DNS查询系统的测试，测试的数据包括了能想到的所有情况，在输入正确域名的情况下均可以返回正确的IP，输入错误的情况下也都给出了错误的提示。但是测试发现了在自身网络不支持IPV6协议的情况下，无法查询到域名的IPV6地址。

# 四．测试结论

## 4.1 node质量

通过测试，node的确是一个很好的工具，在功能和非功能性的需求都完成的比较好。

## 4.2 缺陷和限制

在测试过程中未发现缺陷。

## 4.3 结论存在的风险

本次测试总体完成了对node和对node扩展部分的测试，但由于一些测试设备和环境的局限性，对node通过兼容性测试的结论有一定的风险，因为linux版本众多，我们只是保证了ubuntu的一个版本号的兼容，无法确定node对于linux其他诸如redhat,centos这些版本的兼容。