**漠影人系统**

**软件测试报告**

**版本1.0**

**编写者：张栩**

**编写日期：2018年3月22日**

**版本历史：**

**表1 版本历史表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 日期 | 版本 | 文件名 | 文档格式 | 状态 |
| 2017.3.21 | V1.0 | 软件测试报告 | Word | 创建 |

**更改列表：**

**表2 更改列表**

|  |  |
| --- | --- |
| NO. | 描述 |
|  |  |

目录

[1 引言 2](#_Toc510711528)

[1.1 编写目的 2](#_Toc510711529)

[1.2 背景 3](#_Toc510711530)

[1.3 定义 3](#_Toc510711531)

[1.4 参考资料 3](#_Toc510711532)

[2 测试对象和概要 3](#_Toc510711533)

[2.1 测试方法和工具 3](#_Toc510711534)

[2.2 2测试环境与配置 4](#_Toc510711535)

[2.2.1 功能测试 4](#_Toc510711536)

[2.2.2 性能测试 4](#_Toc510711537)

[2.3 测试用例设计 4](#_Toc510711538)

[2.3.1 单元测试 4](#_Toc510711539)

[2.3.2 功能测试 10](#_Toc510711540)

[2.3.3 性能测试 12](#_Toc510711541)

[2.4 2.4 整体策略 18](#_Toc510711542)

[2.5 测试范围 19](#_Toc510711543)

[3 测试内容和执行情况 19](#_Toc510711544)

[4 覆盖分析 19](#_Toc510711545)

[5 缺陷的统计与分析 20](#_Toc510711546)

[6 测试结论与建议 20](#_Toc510711547)

[6.1 测试结论 20](#_Toc510711548)

[6.2 建议 20](#_Toc510711549)

# 引言

## 编写目的

本测试报告为漠影人系统项目的测试报告，目的在于总结测试阶段的测试以及分析测试结果，描述系统是否符合需求，为纠正软件缺陷提供依据，预期参考人员包括用户、测试人员等。

## 背景

本次测试的软件为漠影人系统，测试环境与实际环境可能会存在一些差异。

## 定义

无。

## 1.4 参考资料

[1]张栩.《漠影人系统规格说明书》,2017年10月.

[2]张栩.《漠影人系统软件体系结构设计规格说明书》 ，2018年3月.

[3]张栩.《漠影人系统软件详细设计规格说明书》 ，2018年3月.

# 测试对象和概要

## 测试方法和工具

表2-1 测试方法和工具表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试内容 | 测试方法 | 测试工具 | 备注 |
| 单元测试 | 白盒、手工 | Junit |  |
| 功能测试 | 黑盒、手工 | 无 |  |
| 性能测试 | 黑盒、手工、回归、工具自动 | JMeter |  |

## 2测试环境与配置

### 功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 数据库服务器配置 | | | |
| 机器名（IP） | CPU | 内存 | 软件环境 |
| 144.123.7.62 | 英特尔i7 | 8G | 操作系统：windows10  应用软件：mysql |
| 应用服务器配置 | | | |
| 机器名（IP） | CPU | 内存 | 软件环境 |
| 144.123.7.62 | 英特尔i7 | 8G | 操作系统:windows10  应用软件:tomcat |
| 客户端配置 | | | |
| 机器名（IP） | CPU | 内存 | 软件环境 |
| 144.123.7.62 | 英特尔i7 | 8G | 操作系统:windows10  应用软件:FireFox |
| 说明 | 访问地址:144.123.7.63:8080//movieshare | | |

### 性能测试

测试时间：2018.3.22

测试工具：JMeter，Badboy

测试内容：利用BadBoy录制脚本，将录制的脚本利用JMeter进行压力测试。

## 测试用例设计

### 单元测试

1. 影片查看基本路径测试

(1)影片查看service层代码：

public Movie checkMovie(Integer movieID) {

2 if(movieID<=0||movieID==null){

return null;

}else{

5 Movie movie = movieDao.getMovieByID(movieID);

6 if (movie == null) {

return null;

7 }else{

8 List<Mark> marks = markDao.getMarkByMovie(movieID);

movie.setMarkList(marks);

10 List<Comment> comments = commentDao.getCommentByMovie(movieID);

movie.setCommentList(comments)

12 List<FilmmakerMovie> fms =

filmmakerMovieDao.getFilmmakerByMovie(movieID);

movie.setFilmmakerList(fms);

15 List<TheType> types = typeDao.getTypeByMovie(movieID);

movie.setTypeList(types)

return movie;

}

}

20}



图2.3.1.1.1程序流程图

(2)根据流程给出控制流图



.图2.3.1.1.2程序控制流图

(3)计算程序的环形复杂度，决定独立路径条数

利用判定节点发

V(G)=P+1;

得V(G)=3；

所以环形复杂度为3.

(4)导出基本路径集，确定程序的独立路径

根据上面的计算方法，可得到3个独立的路径

路径1:2-20

路径2:2-5-6-20

路径3:2-5-6-8-10-12-15-20

(5)根据上一步的独立路径设计测试用例

表2.3.1.1基本路径测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 输入 | 预期输出 | 路径 |
| movieID | movie |
| Test1 | -1 | null | 1 |
| Test2 | null | null | 1 |
| Test3 | 1 | 影片信息 | 2 |
| Test4 | 100000001 | null | 3 |

1. 影人添加基本路径测试

（1）影人添加service层代码

1. public FilmmakerType addFilmmaker(Filmmaker filmmaker){ Connection conn = null;
2. FilmmakerType filmmakerType = null;
3. try{
4. conn = JdbcUtils.*getConnection*();
5. conn.setAutoCommit(false);
6. Integer getFilmmaker = filmmakerDao.getFilmmakerByName(

filmmaker.getFilmmakerName());

1. if(getFilmmaker==null)
2. filmmakerDao.saveFilmmaker(conn,filmmaker);
3. conn.commit();

Integer fId = filmmakerDao.getFilmmakerByName(

filmmaker.getFilmmakerName());

1. List<Job> jobs = filmmaker.getJobList();
2. for(Job job : jobs){
3. filmmakerJobDao.saveFilmmakerJob(

conn,fId,job.getJobID());

1. }
2. conn.commit();
3. filmmakerType = FilmmakerType.*AddSuccess*;
4. }else{
5. filmmakerType = FilmmakerType.*FilmmakerExist*;
6. }
7. }catch(Exception e){
8. e.printStackTrace();
9. conn.*connRollBack* ();
10. filmmakerType = FilmmakerType.*AddFailed*;
11. }finally{
12. JdbcUtils.*closeConnection*(conn);
13. }
14. return filmmakerType;
15. }



图2.3.1.2程序流程图

(2)根据流程图给出控制流图



图2.3.1.1.2程序控控制流图

(3)计算程序的环形复杂度，决定独立路径条数

利用判定节点发

V(G)=P+1;

得V(G)=3；

所以环形复杂度为3。

(4)导出基本路径集，确定程序的独立路径

根据上面的计算方法，可得到3个独立的路径

路径1:4-20-21-27

路径2:4-6-7-17-27

路径3: 4-6-7-9-11-15-27

(5)根据上一步的独立路径设计测试用例

表2.3.1.2基本路径测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 输入 | 预期输出 | 路径 |
| new Filmmaker() | FilmmakerType |
| Test1 | new Filmmaker(“周星驰”,”男”,…, new Job(1)) | AddSuccess | 3 |
| Test2 | new Filmmaker(“周星驰”,”男”,…,new Job(1)) | FilmmakerExist | 2 |
| Test3 | new Filmmaker(”周润发”,”男”,…,new Job(3)) | AddFailed | 1 |

### 功能测试

等价类测试用例的设计方法

1.定义

等价类划分法是一种典型的黑盒测试方法，它完全不考虑程序的内部结构，只根据程序规格说明书对输入范围进行划分，把所有可能的输入数据,即程序的输入域划分成若干个不相干的子集,称为等价类，然后从每一个等价类中选取少数具有代表性的数据作为测试用例，进行测试。

2.划分等价类

等价类是指某个输入域的子集合。在该子集合中,各个输入数据对于揭露程序中的错误都是等效的，并合理地假定：测试某等价类的代表值就等于对这一类其它值的测试，因此,可以把全部输入数据合理划分为若干等价类,在每一个等价类中取一个数据作为测试的输入条件就可以用少量代表性的测试数据取得较好的测试结果。等价类划分可有两种不同的情况：有效等价类和无效等价类。

1)有效等价类

是指对于程序的规格说明来说是合理的、有意义的输入数据构成的集合。利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能。

2)无效等价类

与有效等价类的定义恰巧相反。无效等价类指对程序的规格说明是不合理的或无意义的输入数据所构成的集合。对于具体的问题，无效等价类至少应有一个，也可能有多个。

设计测试用例时,要同时考虑这两种等价类。因为软件不仅要能接收合理的数据,也要能经受意外的考验，这样的测试才能确保软件具有更高的可靠性。

3.等价类划分测试用例设计步骤

1. 确定被划分的对象。
2. 为每个被划分的对象进行有效等价类及无效等价类划分。
3. 为每个等价类规定一个唯一的编号。
4. 设计一个新的测试用例，尽可能多地覆盖尚未被覆盖的有效等价类，重复这一步，直到测试用例覆盖了所有的有效等价类。
5. 设计一个新的测试用例，使其覆盖并且只覆盖一个还没有被覆盖的无效等价类，重复这一步，直至测试用例覆盖了所有的无效等价类。

测试用例设计

1. 影片查看问题描述

用户通过漠影人系统查询影片信息，要求用户输入的影片ID存在且为大于0的整型，如何用户输入的影片ID不存在或不符和要求，则跳转至错误信息页面。

表2.3.2.1影片查看问题的等价类

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入影片ID（大于0的整数） | 有效等价类 | 编号 | 无效等价类 | 编号 |
| 输入存在且大于0的整型类型的数。 | 1 | 输入不存在的影片ID | 2 |
| 输入小于0的整型类型的数 | 3 |
| 输入非整形类型 | 4 |

有效等价类测试用例设计：尽可能多地覆盖尚未被覆盖的有效等价类。

表2.3.2.2影片查看问题的有效等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 输入数据 | 期望输出 | 覆盖等价类 |
| Test1 | 1 | 影片查看页面 | 1 |

无效等价类测试用例设计：覆盖并且只覆盖一个还没有被覆盖的无效等价类。

每次只覆盖一个无效等价类，是因为一个测试用例若覆盖了多个无效等价类，那么某些无效等价类可能永远不会被检测到，因为第一个无效等价类的测试用例可能会屏蔽或终止其他无效等价类的测试执行。

表2.3.2.3影片查看问题的无效等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 输入数据 | 期望输出 | 覆盖等价类 |
| Test2 | 100000001 | 错误信息页面 | 2 |
| Test3 | -1 | 错误信息页面 | 3 |
| Test4 | ”功夫” | 错误信息页面 | 4 |

1. 影人添加问题描述

用户添加影人信息时，要求输入影人名，影人名为汉字与·组成长度为2-20字符的字符串,如果输入长度小于2或大于20，则提示”影人名请在2-20字符之间”，如果输入不为汉字与·，则提示”影人名格式不正确”。

表2.3.2.4影人添加问题等价类

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入影人名(汉字与·组成，长度为2-20字符) | 有效等价类 | 编号 | 无效等价类 | 编号 |
| 汉字与·组成，长度为2-20的字符。 | 1 | 长度小于2的汉字。 | 2 |
| 长度大于20由汉字与·组成的字符。 | 3 |
| 非汉字与·组成的2-20字符。 | 4 |

有效等价类设计：

表2.3.2.5影人添加有效等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 输入数据 | 期望输出 | 覆盖等价类 |
| Test1 | 小罗伯特·唐尼 | 影人名正确。 | 1 |

无效等价类设计：

表2.3.2.6影人添加无效等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 输入数据 | 期望输出 | 覆盖等价类 |
| Test2 | 张 | 影人名长度不符合规范。 | 2 |
| Test2 | 这是姓·这是中间名·为了凑够二十字为了凑够二十字。 | 影人名长度不符合规范。 | 3 |
| Test4 | 1+2·3=7 | 影人格式不符规范。 | 4 |

### 性能测试

1.名词定义

配置元件（config elements ）

　　元件会影响其作用范围内的所有元件。

前置处理程序（Per-processors）

　　元件在其作用范围内的每一个sampler元件之前执行。

定时器（timers ）

　　元件对其作用范围内的每一个sampler 有效

后置处理程序（Post-processors）

元件在其作用范围内的每一个sampler元件之后执行。

断言（Assertions）

元件对其作用范围内的每一个sampler 元件执行后的结果执行校验。监听器（Listeners）

元件收集其作用范围的每一个sampler元件的信息并呈现。

测试计划：用来描述一个性能测试，所有内容都是基于这个计划的。

线程组：这个就是我们通常添加运行的线程。

Setup thread group：可用于执行预测试操作。这些线程的行为完全像一个正常的线程组元件。不同的是，这些类型的线程执行测试前进行定期线程组的执行。类似Loadrunner中的init。

Teardown thread group：可用于执行测试后动作。这些线程的行为完全像一个正常的线程组元件。不同的是，这些类型的线程执行测试结束后执行定期的线程组。类似Loadrunner中的end。

Ramp-Up Period：单位是秒，默认时间是1秒。它指定了启动所有线程所花费的时间，比如，当前的设定表示“在5秒内启动5个线程，每个线程的间隔时间为1秒”。如果你需要Jmeter立即启动所有线程，将此设定为0即可

循环次数：表示每个线程执行多少次请求。

1. 对被测试页面进行压力测试

2.1影片查看压力测试

下图为200个线程，每一秒启动10个，循环10次的测试

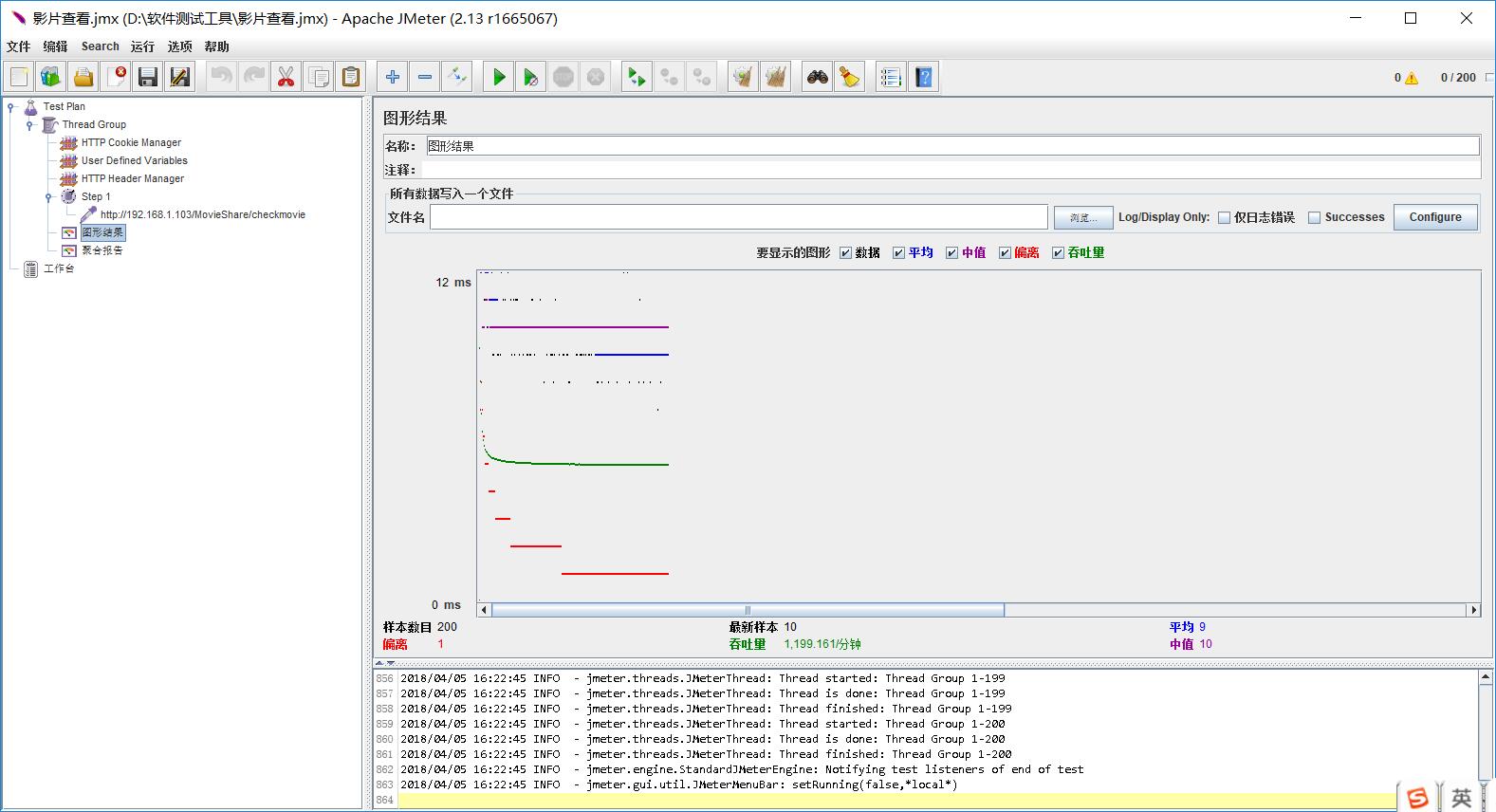


图2.3.3.1影片查看200线程图形结果

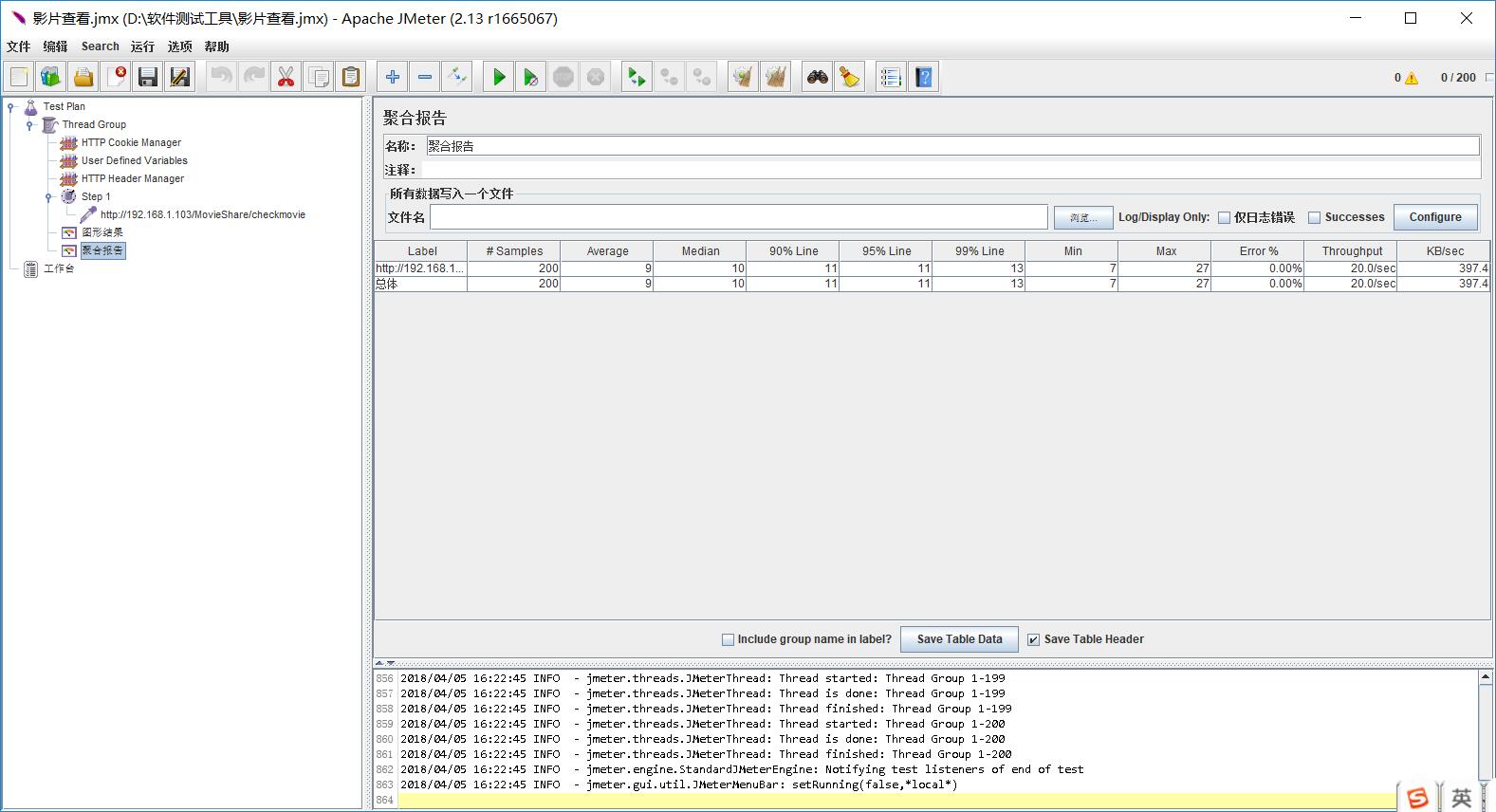


图2.3.3.2影片查看200线程聚合报告

下图为200个线程，每一秒启动10个，循环10次的测试

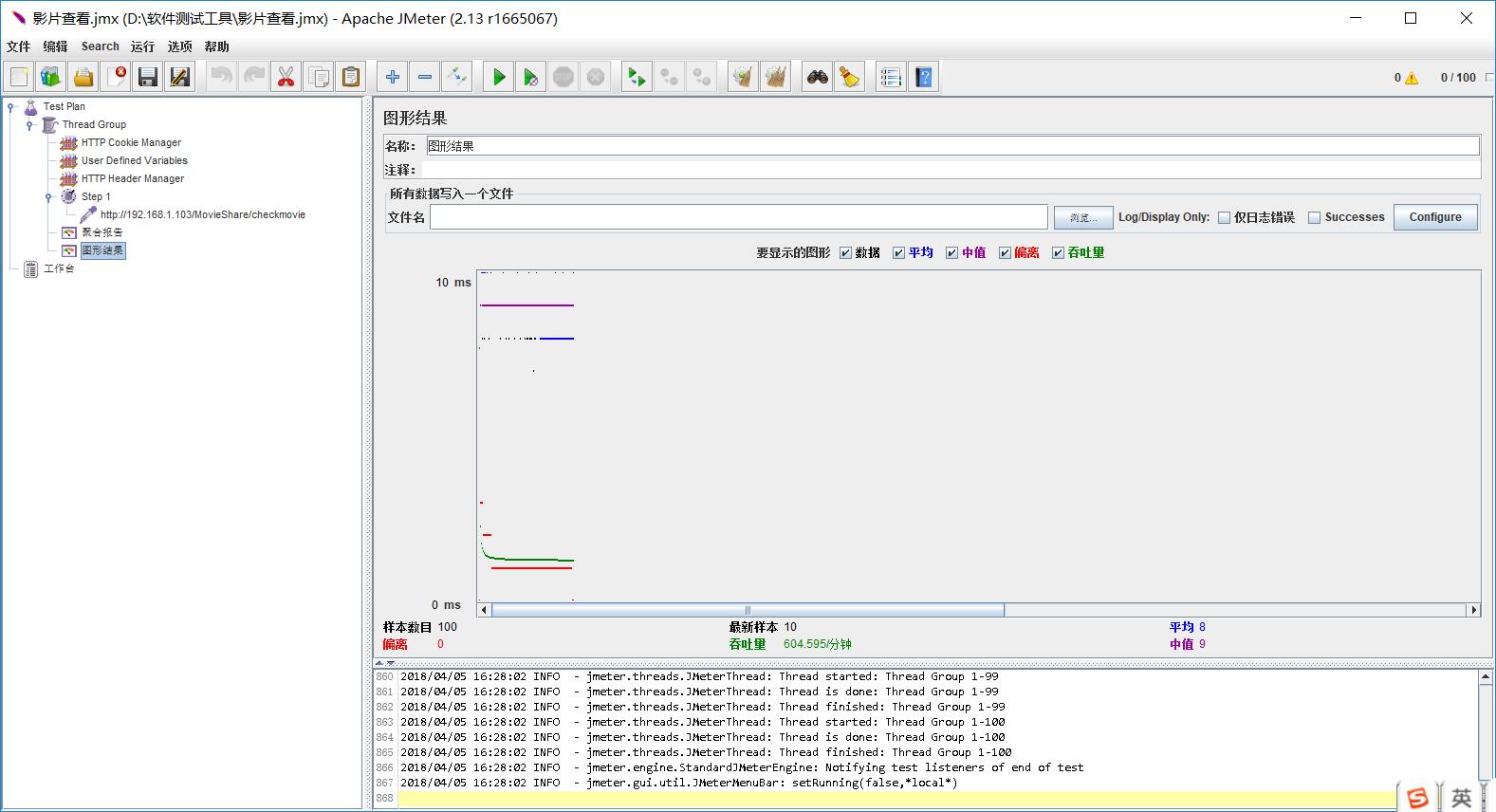


图2.3.3.3影片查看100线成图形结果

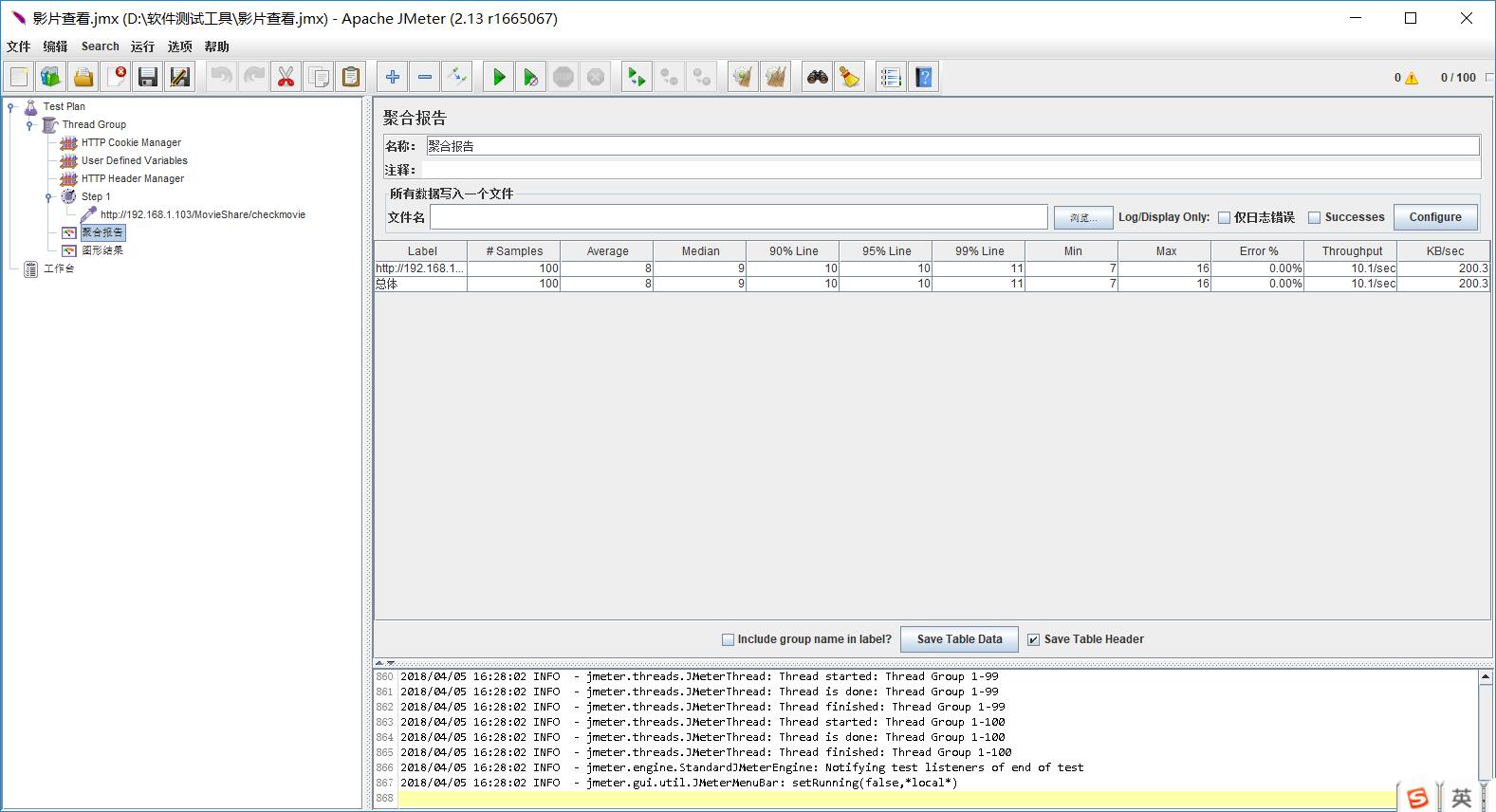


图2.3.3.4影片查看100线程聚合报告

测试结果分析

(1)图形结果

图表底部参数的含义如下：

样本数目是总共发送到服务器的请求数。

最新样本是代表时间的数字,是服务器响应最后一个请求的时间。

吞吐量是服务器每分钟处理的请求数。

平均值是总运行时间除以发送到服务器的请求数。

中间值是代表时间的数字，有一半的服务器响应时间低于该值而另一半高于该值。

偏离表示服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布。

说明：两个登录测试结果对比，图形结果相差不大。

(2)聚合报告

#Samples：表示这次测试中一共发出了多少个请求。

Average：平均响应时间。

Median：中位数，也就是 50％ 用户的响应时间。

90% Line：90％ 用户的响应时间。

99% Line：99％ 用户的响应时间。

Min：最小响应时间。

Max：最大响应时间。

Error%：本次测试中出现错误的请求的数量/请求的总数。

Throughput：吞吐量。

KB/Sec：每秒从服务器端接收到的数据量。

说明：相比之下不同线程数下，Average，Median，90% Line，95% Line，99% Line，Min，Max无变化，线程数变大，吞吐量会变大。

2.2影评回复

下图为200个线程，每一秒启动10个，循环10次的测试

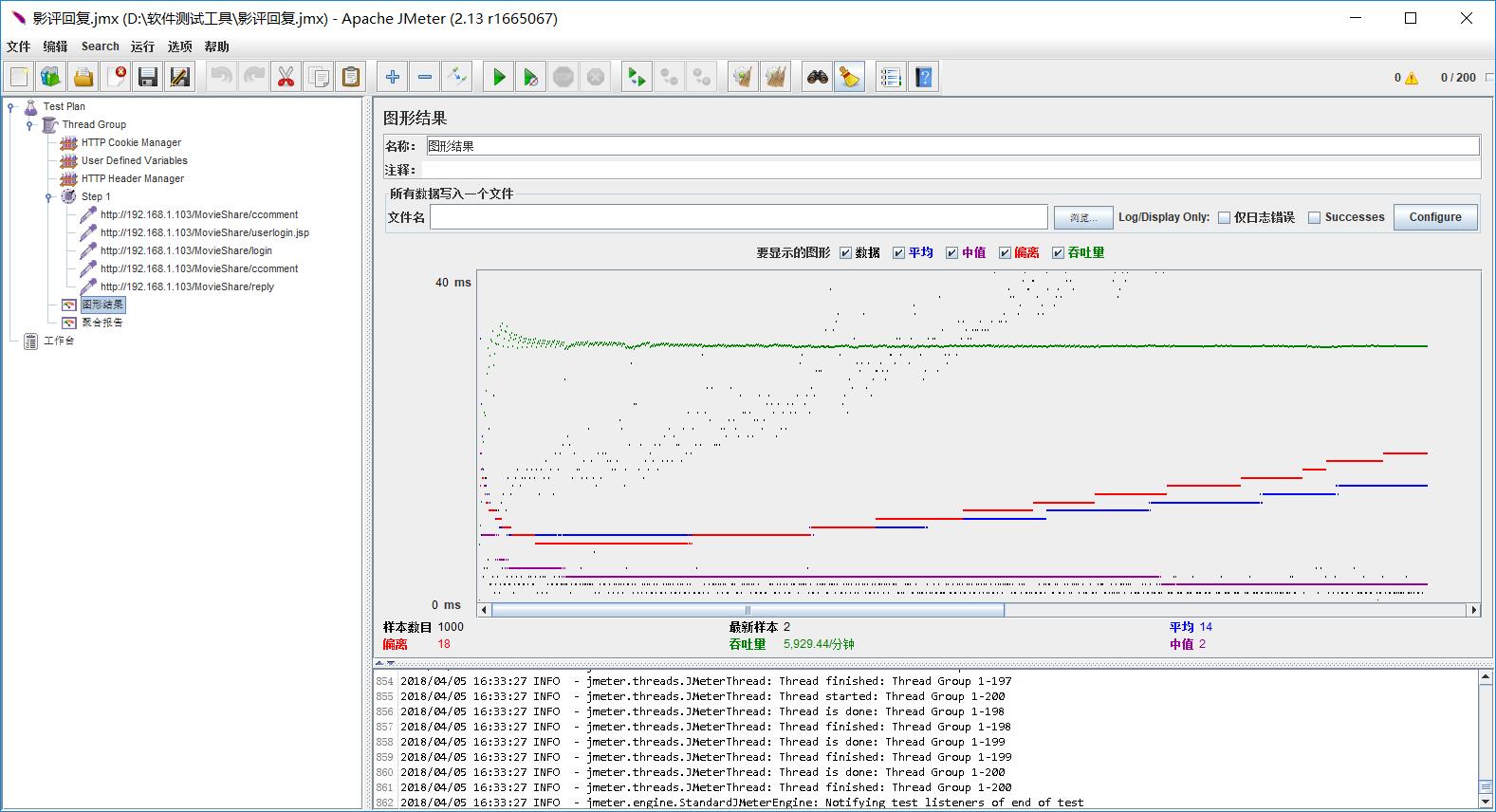


图2.3.3.5影评回复200线程图形结果

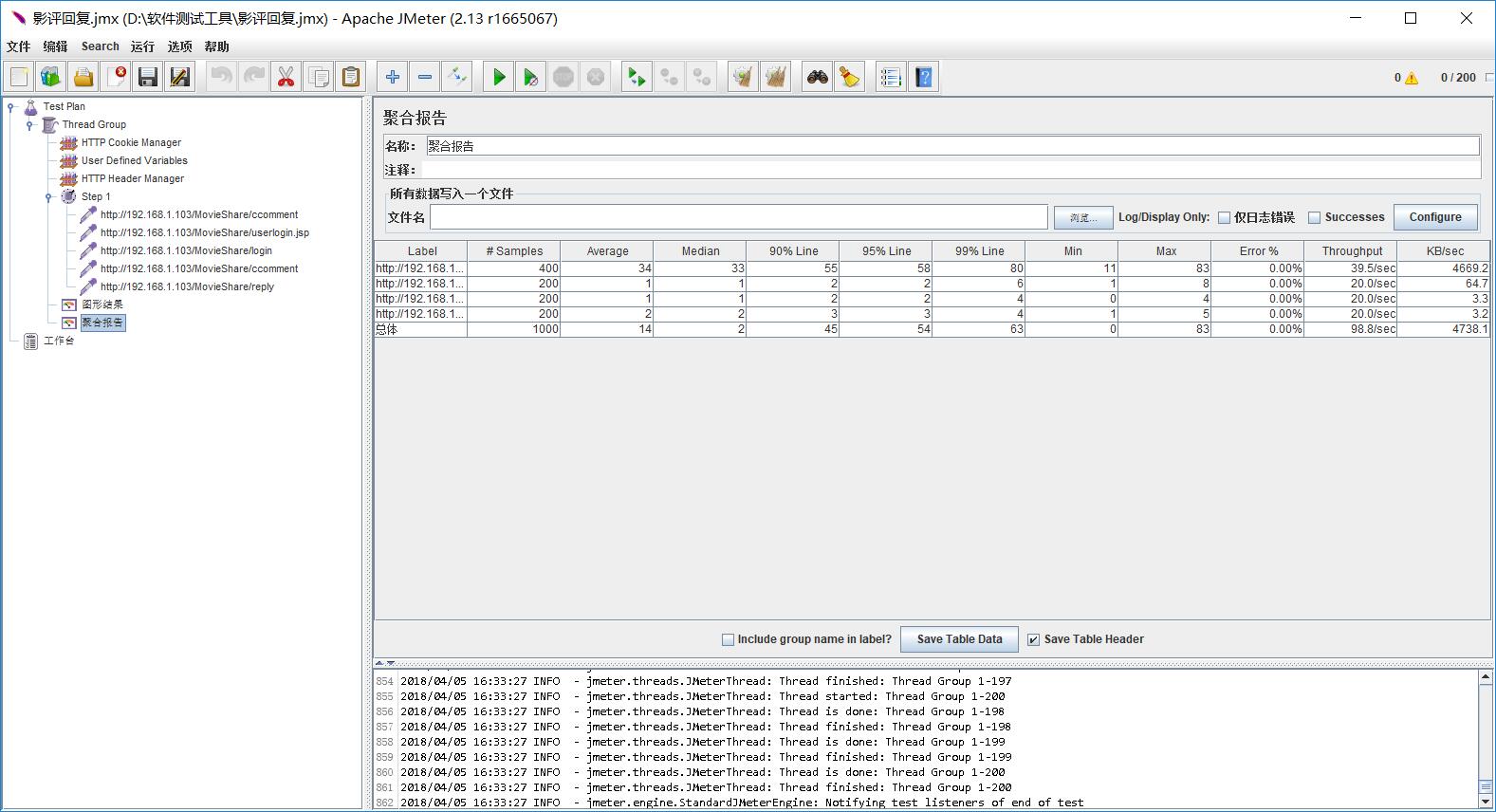


图2.3.3.6影评回复200线程聚合报告

下图为100个线程，每一秒启动10个，循环10次的测试

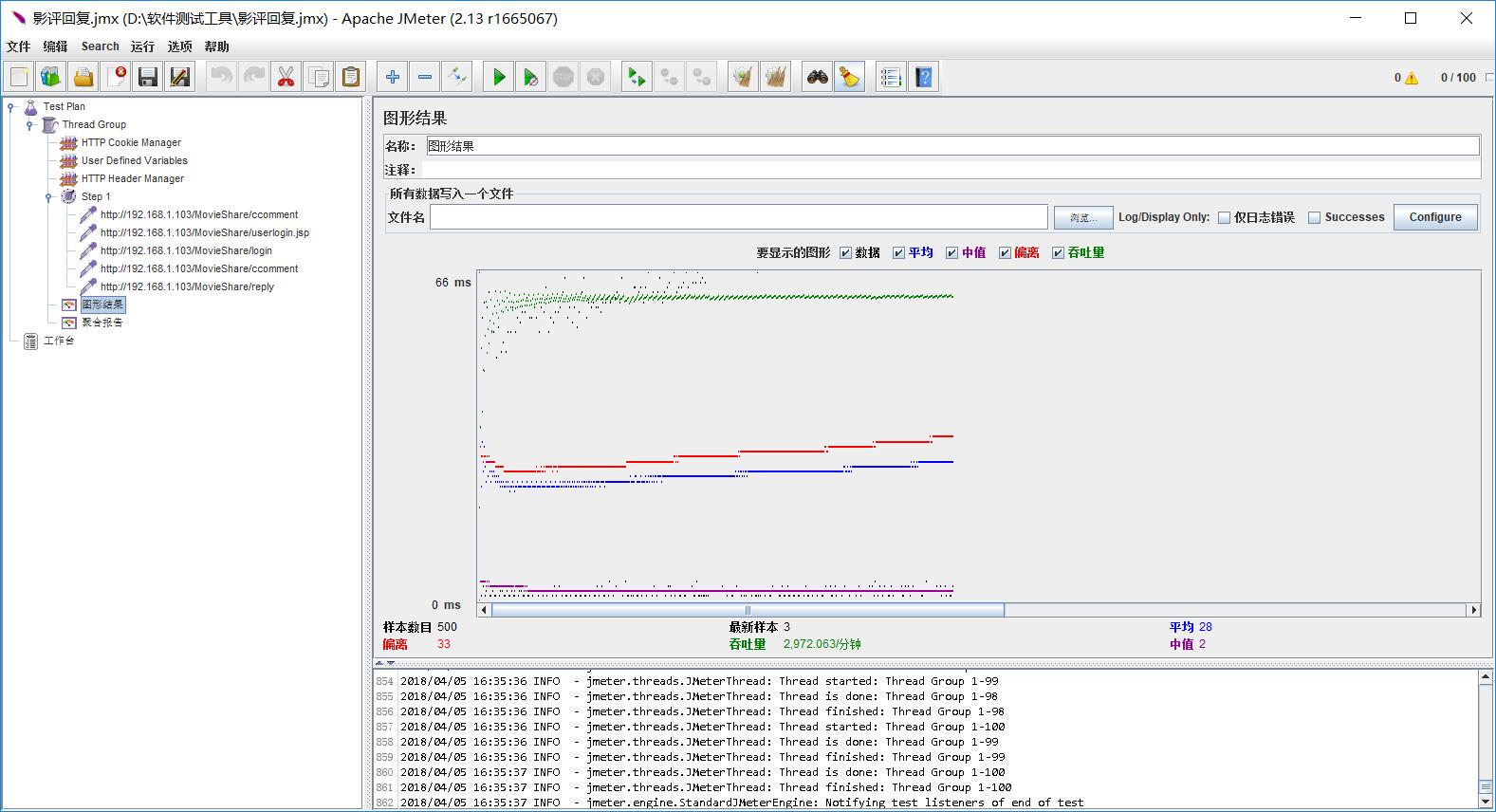


图2.3.3.7影评回复100线程图形结果

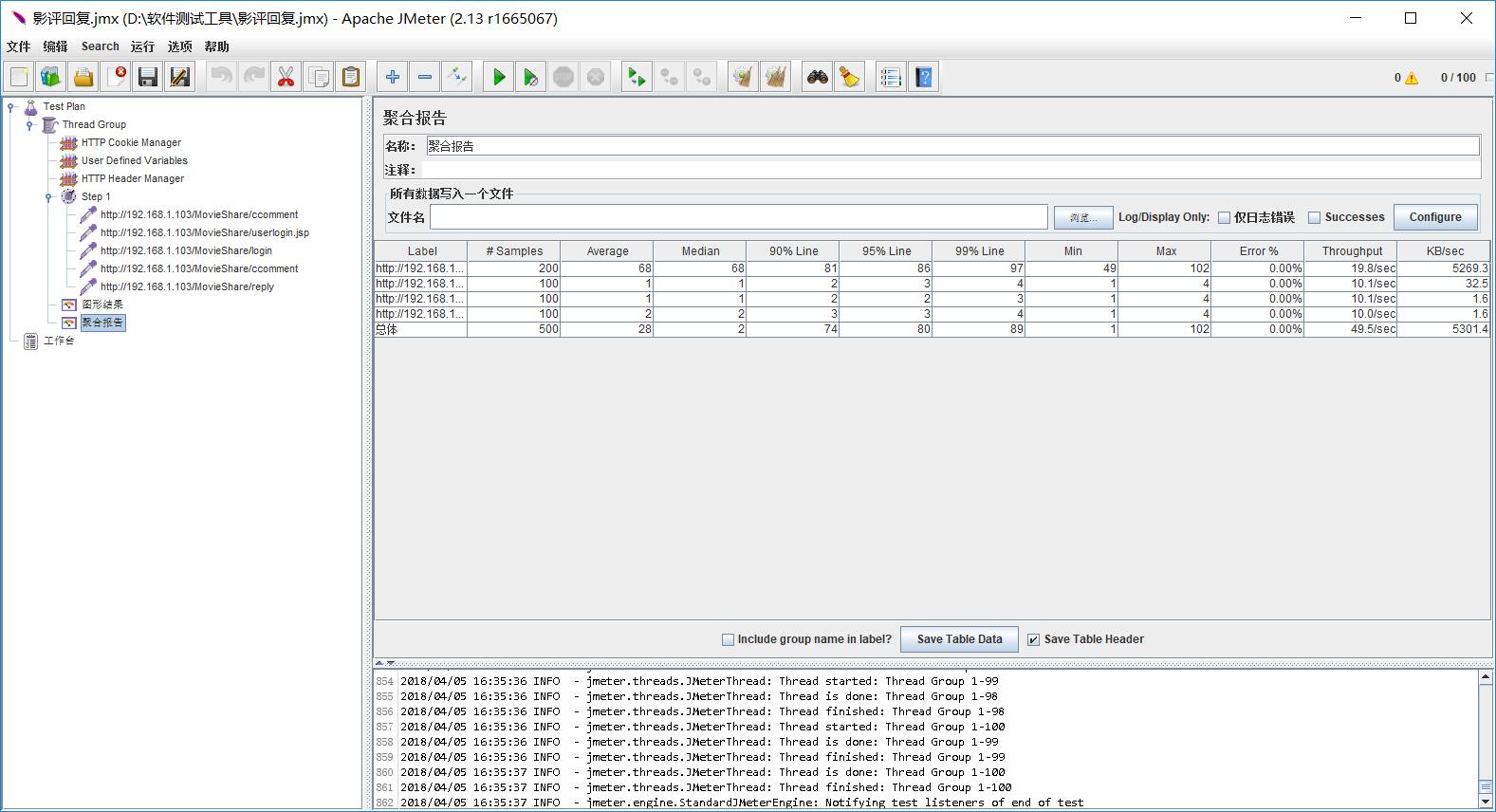


图2.3.3.8影评回复100线程聚合报告

说明：相比之下不同线程数下，Average，Median，90% Line，95% Line，99% Line，Min，线程数变大时，Max和吞吐量变大。

## 整体策略

（1）本项目的特点：

1.参与的测试人员都是第一次接触漠影人系统。

2.系统已经做过一些测试，并且已经在运行。

3.相对于项目要做的事情来说，时间进度非常紧。

4.本次项目测试支队系统进行一轮测试。

根据以上特点，制定本项目的测试过程策略如下：

1.以80/20原理为指导。

尽量做到在有限的时间里发现尽可能多的缺陷（尤其是严重缺陷）

2.测试计划与需求制定、用例设计同步进行

3.必须制定测试需求。

通过确定要测试的内容和各自的优先级、重要性，使测试设计工作更有目的性，在需求的指导下设计出更多更有效的用例。

4.逐步完善测试用例库。

测试用例库的建设是一个不断完善的过程，我们要在有限的时间里，先设计出一整套的测试用例，重要的部分用例需要设计得完善一些，一般部分的则指出测试的要点，在以后的测试工作中再不断去完善测试用例库。

5.测试过程要受到控制。

根据事先定义的测试执行顺序进行测试，并填写测试记录表，保证测试过程是受控的。

6.确定重点。

测试重点放在各子系统的功能实现上。

7.不测试其实现技术。

本次测试不对漠影人系统中的功能实现的核心技术（环境仿真等）进行测试验证。

（2）测试技术

本项目采用白盒测试和黑盒测试技术。

（3）依据标准

本次测试中测试文档的编写、测试用例的编写、具体的执行测试以及测试中各项资源的分配和估算，都是依照漠影人系统软件提供的使用手册为标准。软件的执行以系统逻辑设计构架为依据。

（4）测试过程



**图26 测试过程图**

## 测试范围

本次测试的范围为漠影人系统软件的各项功能。

# 测试内容和执行情况

比较成功。

# 覆盖分析

大约6%。

# 缺陷的统计与分析

无。

# 测试结论与建议

## 测试结论

“漠影人系统”进行功能、可靠性、安全性、可使用性、兼容性、安装和手册功能七个方面进行了全面、严格、规范的测试。测试结果表明：“漠影人系统”完全达到业务需求文档中的要求，并具有以下特点：

1.系统架构先进、简单。该系统采用先进的B/S开发架构，后台支持各种大小数据库，系统结构清晰明确。

2.功能全面。用户端包括登陆，注册，修改账号资料，修改密码，上传照片，收藏照片，评论照片，转发照片，编辑照片，删除照片，社会化分享网页，发表文章，编辑文章，删除文章，评论文章，发起话题，参与话题，取消收藏，取消关注。

3.系统安全性较好。系统使用账号密码登录，上传照片等功能只有登录后才可以使用，保证用户隐私。

4.系统可靠性高。对用户输入不符合要求的数据，给出了简洁、准确的提示信息，必要时给出了帮助。

5.测试结论：通过。

## 建议

无。