**3.确定测试**

3.1 测试方法

黑盒测试

3.2 测试计划

为了使软件能满足用户需求，我们还将从如下角度进行测试：

1）系统对用户需求响应的及时性测试

2）系统对不同部位的友好性测试

3.3 测试过程（）

测试人员扮演角色：两个人分别担任测温者和被测温者；

 流程：

A.运行程序。

B.点击预加载，将脸对准摄像头，进行匹配，匹配成功后会显示测温人员的姓名和学号。

C.点击测温按钮，被测温人员将手心对准传感器，并将体温数据呈现在电脑屏幕。

D.数据异常则点击重新测试，数据正常则点击保存保存数据。

E.测温人员点击查看区域体温，可以查看某个区域的平均提问曲线。

F.测温人员点击查看某个人员体温，输入学号可以查询某个学生的体温信息和历史体温

3.4 配置复审

对软件配置是否齐全进行再次确认：

包括：不同平台的安装包、用户手册、各接口间工作是否能正确进行

4 系统测试

**4.1 恢复测试**

恢复测试（Recovery testing）是测试一个系统从灾难或出错中能否很好地恢复的过程，如遇

到系统崩溃、硬件损坏或其他灾难性出错。简单的说，恢复测试是一种对抗性的测试过程。

对于我们的订餐系统，系统可恢复性显得十分重要，可恢复性对系统的稳定性、可靠性影响

很大。但可恢复性测试很容易被忽视，因为可恢复测试相对来说是比较难的，一般情况下是

很难设想得出来让系统出错和发生灾难性的错误，这需要足够的时间和精力，需要得到我们

更多的关注

**4.2 测试内容**

硬件及有关设备故障。在本系统中，我们需要考虑如设备掉电后（如客户端和服务器端

断电）的可恢复程度的问题，即系统是否具有诊断、故障报告及指示处理方法的能力，

是否具备冗余及自动切换能力，故障诊断方法是否合理和及时。

软件系统故障。我们需要测试系统的程序及数据是否有足够可靠的备份措施，在系统遭

破坏后是否具有重新恢复正常工作的能力，对系统故障是否自动检测和诊断的功能。

防止“毁灭性”灾难的发生而导致的不可挽回的结果。

数据故障。是测试数据处理周期未完成时的恢复程度。

 通信故障和错误。测试有没有纠正通信传输错误的措施，有没有恢复到与其他系统通信

发生故障前原状的措施，还有对通信故障所采取的措施是否满足运行要求等。

我们在进行恢复测试时，较好的达到了测试手册、联机帮助、测试分析报告和应用程序测试

需求的完整性和一致性。

**4.3 安全测试**

**4.3.1 测试方式**

安全测试期间，检测各种可能导致程序崩溃的bug。例如：

 死锁

**4.4 强度测试**

我们为了确定系统在最差工作环境的工作能力,也可能是用于验证在标准工作压力下的各种

资源的最下限指标。（非标准工作环境下，人为降低工作环境资源，如网络带宽、系统内存、

数据锁等，以测试系统在资源不足的情况下的工作状态,通过强度测试,可以确定本系统正常

工作的最差环境.）简言之，就是我们的订单系统在同一时间内能够容纳多少用户同时在线。

**4.4.1 测试指标**

 并发量(吞吐量)

 延迟(最大\最小\平均)

 顺序指标

**4.5 性能测试**

性能测试（Performance Testing）通过自动化的测试工具模拟多种正常、峰值以及异常负载

条件来对系统的各项性能指标进行测试。在软件开发过程中，需要对对产品的性能进行测试，

检验是否达标、是否能够保持。

在大访问量中我们尤其需要使用工具，基于本系统的特点，我们用到了如下的并发性能测试

工具：

 python.

**4.5.2 软件性能测试的指标**

并发用户：给定时间内，某个时刻与服务器同时进行会话操作的用户数。本系统需要很强的

实时性与高并发量，所以需要支持同一时间多用户访问。

响应时间：客户端发出请求到得到服务器返回结果的整个过程所经历的时间。本系统在软件

规划阶段的目标是响应时间在 1~2 秒内。

**5. 排错**

在我们网上订单系统的软件测试中，排错(即调试)与成功的测试形影相随。测试成功的标志

是发现了错误。我们依靠如下的排错技术，根据错误迹象来确定错误的原因和准确位置，并

加以改正。

**5.1 排错过程**

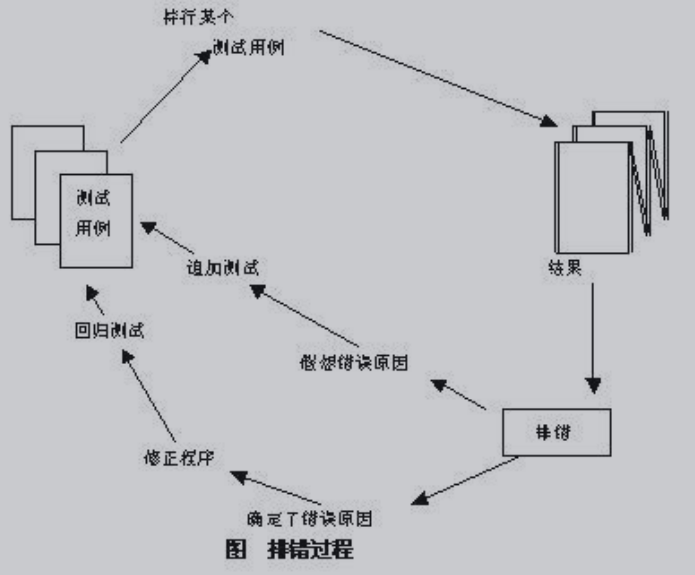
如下图所示，我们的排错过程开始于一个测试用例的执行，若测试结果与期望结果有出入，

即出现了错误征兆，排错过程首先要找出错误原因，然后对错误进行修正。因此排错过程有

两种可能，一是找到了错误原因并纠正了错误，另一种可能是错误原因不明，排错人员只得

做某种推测，然后再设计测试用例证实这种推测，若一次推测失败，再做第二次推测，直到

发现并纠正了错误。



**5.2 排错策略**

在排错过程中，我们先后用到了以下三种排错策略：

1.原始类

这种排错方法是我们最常用但也是最低效的方法，只有在万般无奈的情况下才使用它，主要

思想是“通过计算机找错”。例如输出存储器、寄存器的内容，在程序安排若干输出语句等。

由于我们的数据`较大，在对每一个模块进行排错的时候，使用原始类的方法耗费了我们大量的时间和精力

2.回溯法

回溯法能成功地用于程序的排错。方法是从出现错误征兆处开始，人工地沿控制流程往回追

踪，直至发现出错的根源。不幸的是，我们的测温程序较为庞大，可能的回溯路

线显著增加，以致我们人工进行完全回溯到望而不可及。所以我们在采用回溯法排错的过程

中能够采用了一些自动排错的工具来辅助我们排错，以提高效率.

3.排除法

排除法基于归纳和演绎原理，采用“分治”的概念，首先将与错误出现有关有所有数据，假想

一个错误原因，用这些数据证明或反驳它;或者一次列出所有可能的原因，通过测试一一排

除。只要某次测试结果说明某种假设已呈现倪端，则立即精化数据，乘胜追击。这种方法是

我们在开发过程中使用较多的一种方法，通过我们已有的开发经验以及对错误的敏感度来假

设错误出现的原因，能够命中并解决一部分错误，达到排错的目的.

在使用上述每一类方法进行排错时，我们都辅以了排错工具。比如，我们用到了的 Eclipse

的 Debugger 和 SQL Server 中的 Debugger。总的来说，这些方法都是基于 CASE 的软件测试

和排错，我们采用上述的策略并辅以自动测试的工具，来对代码进行测试和排错.