



信息与软件工程学院

综合设计 II 中期报告

课程名称：_____ 综合设计 II _____

课题名称：银行信用卡业务后台子系统的设计与实现

指导教师：_____ 许毅 _____

所在系别：_____ 软件工程（大型主机方向） _____

执行学期：_____ 大三下 _____

学生信息：

序号	学号	姓名
1（组长）	2014220402030	杨林彬
2	2014220402033	罗阳星
3	2014220401008	郑伟
4		
5		
6		

目 录

第一章 综合设计的进展情况	1
1.1 针对工程问题的方案设计	1
1.2 针对工程问题的推理分析	2
1.3 针对工程问题的具体实现	4
1.4 知识技能学习情况	5
第二章 存在问题与解决方案	6
2.1 存在的主要问题	6
2.2 解决方案	6
第三章 前期任务完成度与后续实施计划	10
参考文献	14

第一章 综合设计的进展情况

1.1 针对工程问题的方案设计

我们选择的课题名称为银行信用卡业务后台子系统的设计与实现。经过仔细的调研，我们对于银行信用卡的基本业务流程有了一定的了解。



图-1 日常生活中的信用卡

^[1]银行信用卡（Credit Card）是银行向个人和单位发行的，可以凭此向特约单位购物、消费和向银行存取现金，其形式是一张正面印有发卡银行名称、有效期、号码、持卡人姓名等内容，由芯片、磁条、签名条组成的磁卡。持卡人持信用卡消费时无需支付现金，待账单日时再还款。

信用卡业务中的关键词：

1. 账单日：银行每月会固定一天对持卡人的信用卡账户中当期发生的各项交易、费用等进行汇总结算，并结记利息，计算持卡人当期总欠款金额和最小还款额，打印出对账单邮寄给客户以便持卡人进行核对，并提示持卡人还款的最后日期和还款金额。
2. 到期还款日：发卡行要求持卡人归还当期信用卡应付款项的最后日期，

一般为对账单日起的第 20 天。

3. 全额还款：持卡人再信用卡到期还款日前偿还对账单上所列示的全部应付款项。采取全额还款的持卡人其当期对账单所列示的信用卡消费可以享受免息还款的待遇。
4. 最低还款：持卡人在到期还款日前偿还当期对账单上的全部应付款项有困难时，可以按发卡行规定的最低还款额进行还款，如持卡人选择最低还款，就不能享受免息还款的待遇。
5. 免息还款期：持卡人利用信用卡进行交易消费，从银行记账日起至到期还款日的期间，在此期间，持卡人只要全额还清当期对账单上所列示的全部应付款项，便不用向银行支付消费交易所产生的贷款利息。

结合信用卡业务和实际开发情况，我们对于本次项目在功能上提出了以下的目标：

1. 系统能够模拟用户交易的数据
2. 数据文件将会囊括消费、取现、还款三大客户行为
3. 程序处理数据文件，生成账单，即统计文件。

我们的计划是使用本学期新学习的一门脚本语言 REXX，来模拟用户的交易数据，把模拟出来的数据通过 FTP 上传到主机上，然后用 COBOL 语言实现业务的后台逻辑。

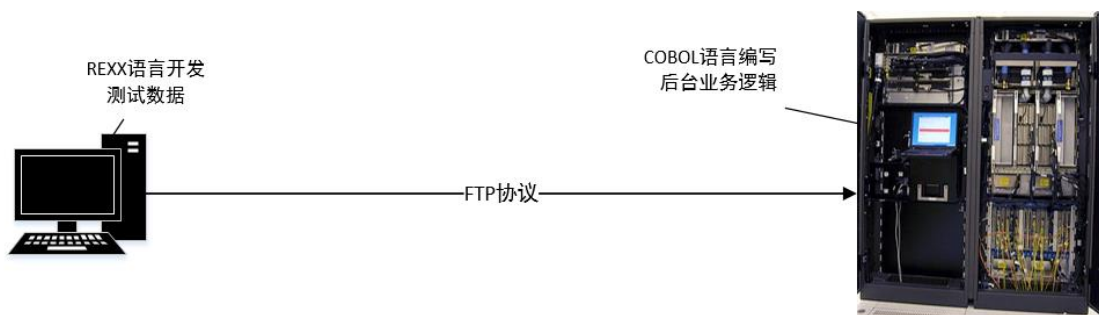


图-2 初步方案

1.2 针对工程问题的推理分析

这个项目从表面上来看是不复杂的，主要就是使用 COBOL 语言实现信用卡的后台业务逻辑。但是软件作为一种工具，它存在的价值不是在于其本身开发的

复杂程度，而是要看该软件是否很好地对现实业务进行了模拟。所以前期我们地主要精力没有放在实际开发上面而是着重于银行卡后台业务的调研工作。我们以下面这个例子来解释以下银行卡的具体业务过程：

【例】Alice 的账单日为每月 18 日，到期划款日为每月的 7 日。4 月 18 日银行为 Alice 打印的本期账单包括了他从 3 月 19 日到 4 月 18 日之间的所有交易业务。本账单周期 Alice 仅有一笔消费：在 4 月 15 日消费人名币 1000 元。在 4 月 18 日的账单上显示 Alice 要还金额为 1000 元，如果 Alice 在 5 月 7 日前偿还 100 元，计算 5 月 18 日对账日显示的利息？

解：具体的计算如下

$$1000 \text{ 元} \times 0.05\% \times 22 \text{ 天}(4 \text{ 月 } 15 \text{ 日} - 5 \text{ 月 } 7 \text{ 日}) + (1000 \text{ 元} - 100 \text{ 元}) \times 0.05\% \times 12 \text{ 天}(5 \text{ 月 } 7 \text{ 日} - 5 \text{ 月 } 18 \text{ 日}) = 16.40 \text{ 元}$$

这个例子只是针对消费的过程，但是根据我们设计指导书上面的要求是要模拟用户取现的，因此对于取现我们查阅了资料，发现取现的过程不同于消费，从取现日期开始一个周期内，每天都按照 0.05% 的利率计算利息。所以我们在实际开发的过程中应该要把利息分成两类：

1. 消费利息
2. 取现利息

在预估了系统的功能实现方案后，我们对于系统的结构进行了一个大致的规划，制作了下面的这个系统总用例图：

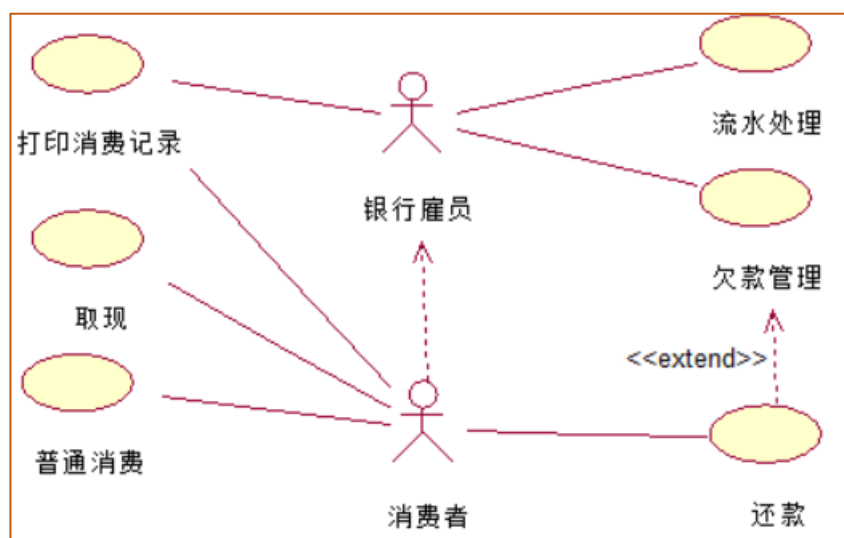


图-3 系统总用例图

我们在前期和老师做了多次交流，我们这个项目主要是使用 COBOL 编写批处理脚本程序，要把精力集中在对于信用卡业务的研究和逻辑实现上，更加深入的研究与开发会在综合设计III中来展开。

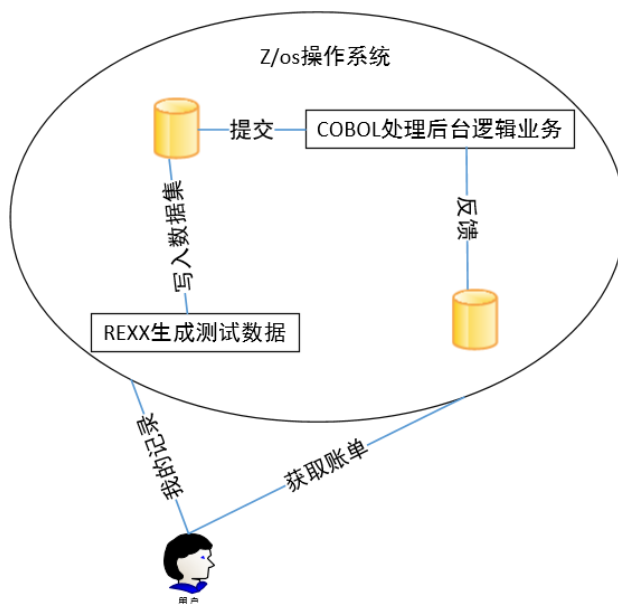


图-4 系统大致框架

1.3 针对工程问题的具体实现

正如前面文档中提到的，我们的测试数据是使用 REXX 开发的。信用卡后台业务逻辑打算使用 COBOL 语言来开发。我们之所以选择 REXX 来作为测试数据的开发工具主要有以下几点考虑：

1. REXX 是一种解释型语言，风格比较简约，上手容易
 2. REXX 语言风格特殊，写出来的代码非常易懂，便于大家传阅交流
- 交易数据设计的格式为：

表格 1-1

交易号	持卡人	交易数额	交易日期	交易类型	银行卡额度
日期+持卡人代号+随机数	持卡人代号：XR+序列号	为一个正整数	格式为：YYYYMMDD	0- 消费 1- 取现 2- 还款	额度值：5000

使用 REXX 模拟的情况：

20161001XR0002	XR000	104	20161001	1	5000
20161002XR0001	XR000	172	20161002	1	5000
20161002XR0002	XR000	185	20161002	1	5000
20161002XR0004	XR000	148	20161002	1	5000
20161003XR0006	XR000	179	20161003	1	5000
20161004XR0005	XR000	159	20161004	0	5000
20161004XR0006	XR000	124	20161004	1	5000
20161005XR0005	XR000	152	20161005	0	5000
20161006XR0002	XR000	128	20161006	1	5000
20161007XR0007	XR000	155	20161007	0	5000
20161007XR0008	XR000	198	20161007	0	5000
20161008XR0004	XR000	200	20161008	1	5000
20161008XR0007	XR000	170	20161008	1	5000
20161011XR0002	XR000	178	20161011	1	5000
20161011XR0003	XR000	135	20161011	0	5000
20161011XR0005	XR000	124	20161011	0	5000

图-5 REXX 模拟用户交易数据

1.4 知识技能学习情况

前期还没有涉及到 COBOL 代码的开发，主要是对于信用卡业务的调研以及 REXX 语言的学习。

对于 REXX，我们主要还是在 windows 平台上开发。后期随着我们学习的深入，会考虑在主机上的 Z/OS 操作系统上进行开发，这样就省去了 FTP 传输文件的步骤，我们的开发工作也会高效得多。主机开发和 windows 平台下的开发还是有很大不同的，举个例子，写入文件（数据集），在 windows 平台上使用的是 call lineout 命令而在主机上则是用 EXECIO 命令。所以我们还需要对 REXX 深入的学习，目前生成的数据以后更具实际开发的需要可能还会做出调整。

第二章 存在问题与解决方案

2.1 存在的主要问题

在开发过程中主要遇到的问题：

一、缺少开发环境：

该银行信用卡后台业务子系统需要在 Z/OS 系统中开发，由于大型主机机房开发时间有限，所以需要自行搭建并配置 Z/OS 虚拟机环境以便进行开发。

二、盘卷已满导致无法启动 TSO：

在编写程序时由于 COBOL 程序无限循环导致盘卷已满，此时无法再次登入系统进行操作。

2.2 解决方案

一、开发环境搭建^[2]：

IBM 主机系统可以借助仿真软件 Hercules 在 PC 上面运行。在安装主机系统之前，必须遵循 IBM 对于主机软件的有关规定。

1、准备并安装相关软件和文件：

主机系统卷文件，PCOM 3270 仿真终端，Hercules，Hercules CTCI-W32 等 TCP/IP 支持软件。

为了方便管理，约定以下几个路径，存放 z/OS 模拟环境所需的各项文件和程序。

表 2-1 虚拟机文件路径

D:/ADCDV1R6	所有与模拟环境相关的东西都放在这个目录下
D:/ADCDV1R6/ZOSV1R6	存放卷文件，即 CCKD 文件
D:/ADCDV1R6/Config	存放启动配置文件
D:/ADCDV1R6/HercGUI-1.11.1	存放 Hercules GUI 和 FishLib
D:/ADCDV1R6/hercules-3.07	存放 Hercules 的可执行文件和 CTCI-W32

第二章 存在问题与解决方案

D:/ADCDV1R6/Log

存放系统运行日志

2、配置 Hercules 启动配置文件 zos1.9.cnf:

在输入配置文件时，需要注意以下几个字段，

- LOADPARM 是 z/OS 在 IPL 时需要用到的参数，参数选择不同，z/OS 中启动后，运行的组件也不同。

- MAINSIZE 表示 Hercules 占用的物理内存大小，单位是 MB，如果内存够大，最好设为 1024，在这里暂且设为 832，建议不要小于 512，否则 MIPS 会很低，系统运行会非常的慢。

- NUMCPU 表示虚拟的主机有多少个 CPU。

- Display Terminals 表示 3270 终端的数量，演示所用的 3270 终端号是从 0700 开始的，以十六进制记录，如 0700-0710 则表示 16 个端口，其中 0700 是控制台专用端口，0701 至 0710 为用户连接端口。

- DASD Device，要把所有下载的 CCKD 文件都列进去，这一段的每一行分为三个部分，第一个是设备编号，如 0A80，第二个是设备类型，如 3390，第三个是卷文件的路径和文件名，这里可以写绝对路径也可以是相对路径。

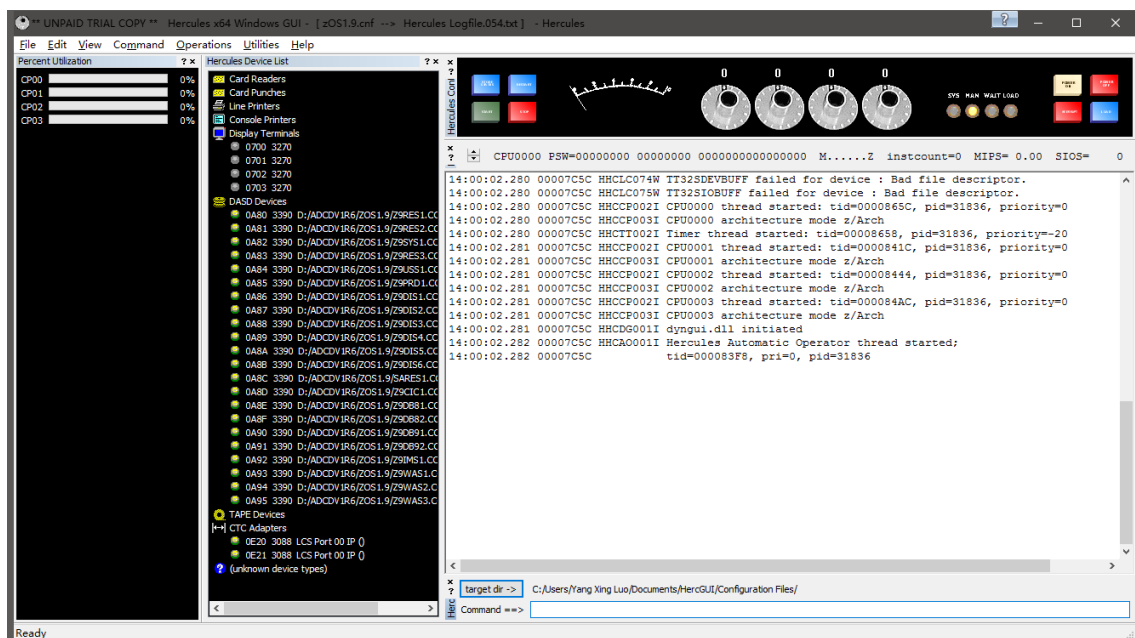


图-66 Hercules 仿真软件

3、启动虚拟机系统：

打开 **Hercules** 仿真软件，选择配置文件加载盘卷文件，再打开一个 **PCOM** 仿真终端窗口，再在 **Hercules** 启动虚拟机，此时在第一个 **PCOM** 窗口进行初始化操作，当操作完成后就可以打开一个新的 **PCOM** 窗口，就可以登录 **Z/OS** 系统了。

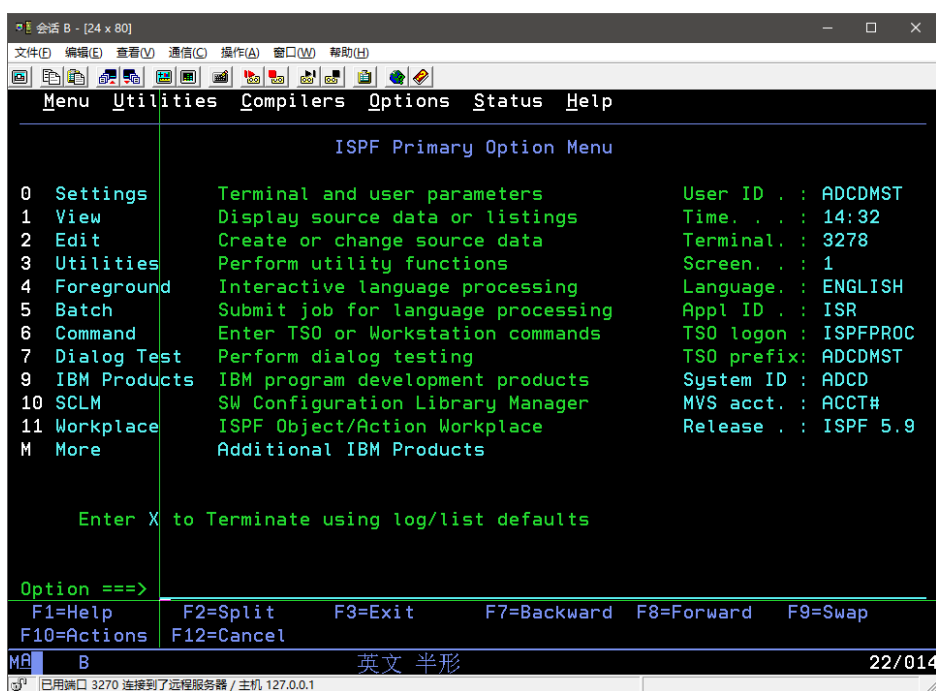
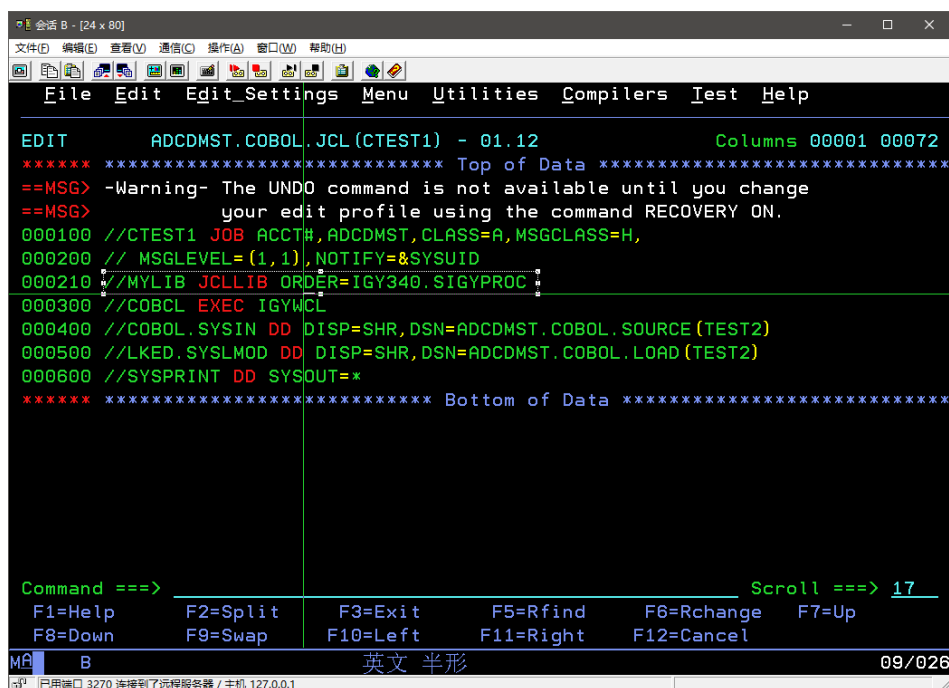


图-7 启动后的虚拟机

4、配置 COBOL 编译器：

虚拟机启动后，还需要配置 **COBOL** 编译器，在编写编译 **COBOL** 的 **JCL** 时，需要在调用编译器的语句之前指明编译器的位置



```
会话 B - [24 x 80]
文件(F) 编辑(E) 查看(V) 通信(C) 操作(A) 窗口(W) 帮助(H)
File Edit Edit_Settings Menu Utilities Compilers Test Help

EDIT      ADCDMST.COBOL.JCL (CTEST1) - 01.12          Columns 00001 00072
***** Top of Data *****
==MSG> -Warning- The UNDO command is not available until you change
==MSG>         your edit profile using the command RECOVERY ON.
000100 //CTEST1 JOB ACCT#,ADCDMST,CLASS=A,MSGCLASS=H,
000200 // MSGLEVEL=(1,1),NOTIFY=&SYSUID
000210 //MYLIB JCLLIB ORDER=IGY340.SIGYPROC
000300 //COBCL EXEC IGYWCL
000400 //COBOL.SYSIN DD DISP=SHR,DSN=ADCDMST.COBOL.SOURCE(TEST2)
000500 //LKED.SYSLMOD DD DISP=SHR,DSN=ADCDMST.COBOL.LOAD(TEST2)
000600 //SYSPRINT DD SYSOUT=*
***** Bottom of Data *****

Command ==>
F1=Help    F2=Split   F3=Exit    F5=Rfind   F6=Rchange F7=Up
F8=Down    F9=Swap     F10=Left   F11=Right  F12=Cancel

MA B      英文 半形      09/026
已用端口 3270 连接到了远程服务器 / 主机 127.0.0.1
```

图-8 配置编译器的位置

二、冷启动虚拟机^[3]:

Hercules 的 LOADPARM 参数指明了虚拟机的启动方式,其形式为 0A82XXM1,其中 XX 可以有很多形式,32 即代表冷启动,重新设置参数后,启动大机,就可以再次登入系统。

冷启动时需要根据提示回复信息: R 0, COUPLE = **。

第三章 前期任务完成度与后续实施计划

3.1 前期任务完成度

前期的工作主要是围绕着信用卡的业务调研和开发环境的搭建和配置上面开展的。就像前面的报告提到的那样，本次项目从实际开发来说难度是不大的，可是，一个软件产品的好坏并不是从开发技术复杂度来判断的，而是在于该软件是否能够对现实世界的实体进行很好的建模模拟。我们前期的调研分析实际上就是在为后期 COBOL 后台业务逻辑编码做准备。

经过前期的分析设计与初步的实现，学习了 REXX 语言以及 COBOL 语句在项目关键部分的使用，现在我们已经大致设计出了交易数据的格式，搭建并配置好了 Z/OS 操作系统的虚拟机环境，后期会持续跟进，尽快写出 COBOL 的 demo 出来，因为考虑到最终我们还会根据实际对程序进行不断的调优。

前期团队的具体分工如下表 3-1 所示

表 3-1 前期团队分配工作情况表

团队成员	前期主要任务	角色及责任
杨林彬	分配工作任务，组织需求与设计的讨论，设计信用卡消费逻辑，使用 COBOL 进行逻辑的预实现	组织、设计、编码
罗阳星	参与信用卡消费逻辑的设计，使用 COBOL 语句完成数据集的读写操作预实现	编码、调试
郑伟	参与信用卡消费逻辑的设计，使用 REXX 语句完成测试数据的生成与调试	编码、调试

完成的测试消费记录数据的生成代码（部分）如下图 3-1（a）、3-1（b）、3-1（c）所示：

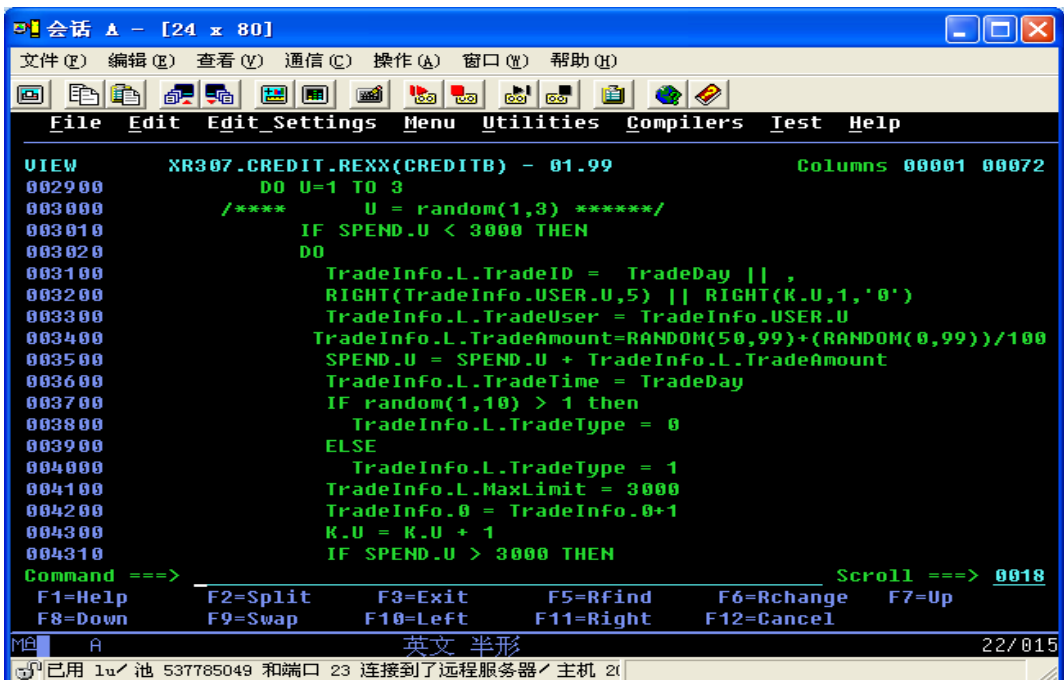


```

VIEW          XR307.CREDIT.REXX(CREDITB) - 01.99          Columns 00001 00072
==MSG>          your edit profile using the command RECOVERY ON.
000100 /***** REXX *****/
000200 TradeInfo.0=1
000300 SPEND.0 = 3
000400 SPEND.1 = 0
000500 SPEND.2 = 0
000600 SPEND.3 = 0
000601 WINTEREST.0 = 3
000602 WINTEREST.1 = 0
000603 WINTEREST.2 = 0
000604 WINTEREST.3 = 0
000610 YEAR = 2017
000620 MONTH = 01
000630 DAY = 01
000700 TradeInfo.USER.0 = 3
000800 TradeInfo.USER.1 = 'XR001'
000900 TradeInfo.USER.2 = 'XR002'
001000 TradeInfo.USER.3 = 'XR003'
Command ==>
F1=Help      F2=Split    F3=Exit    F5=Rfind    F6=Rchange  F7=Up
F8=Down      F9=Swap     F10=Left F11=Right  F12=Cancel

```

图 3-1 (a) 生成测试数据代码

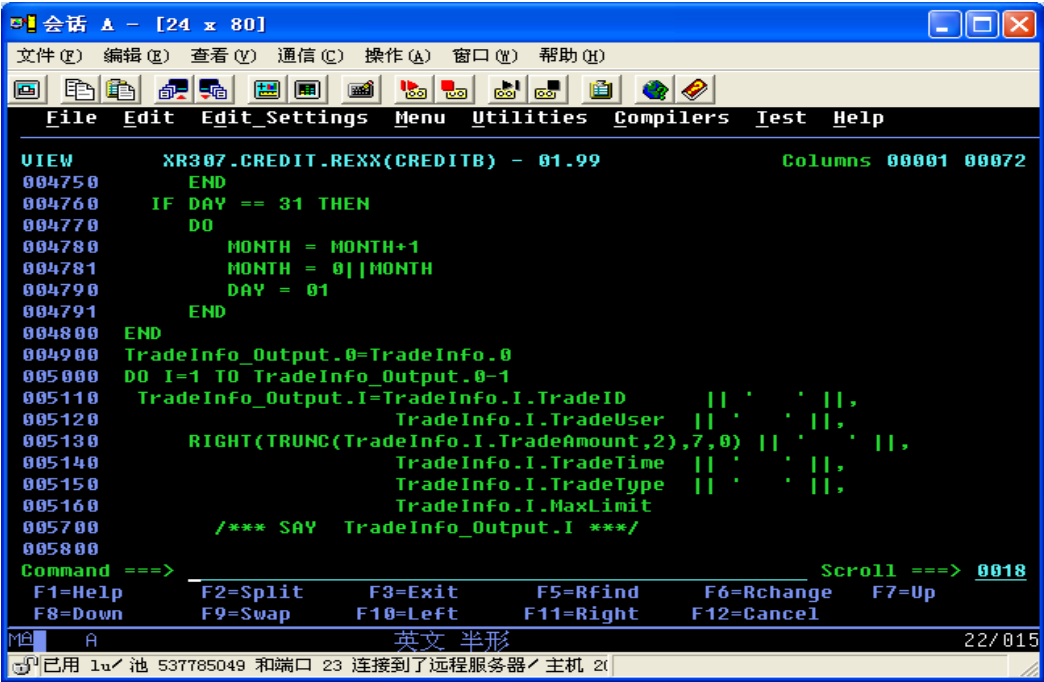


```

VIEW          XR307.CREDIT.REXX(CREDITB) - 01.99          Columns 00001 00072
002900 DO U=1 TO 3
003000 /**** U = random(1,3) *****/
003010 IF SPEND.U < 3000 THEN
003020 DO
003100 TradeInfo.L.TradeID = TradeDay || ,
003200 RIGHT(TradeInfo.USER.U,5) || RIGHT(K.U,1,'0')
003300 TradeInfo.L.TradeUser = TradeInfo.USER.U
003400 TradeInfo.L.TradeAmount=RANDOM(50,99)+(RANDOM(0,99))/100
003500 SPEND.U = SPEND.U + TradeInfo.L.TradeAmount
003600 TradeInfo.L.TradeTime = TradeDay
003700 IF random(1,10) > 1 then
003800 TradeInfo.L.TradeType = 0
003900 ELSE
004000 TradeInfo.L.TradeType = 1
004100 TradeInfo.L.MaxLimit = 3000
004200 TradeInfo.0 = TradeInfo.0+1
004300 K.U = K.U + 1
004310 IF SPEND.U > 3000 THEN
Command ==>
F1=Help      F2=Split    F3=Exit    F5=Rfind    F6=Rchange  F7=Up
F8=Down      F9=Swap     F10=Left F11=Right  F12=Cancel

```

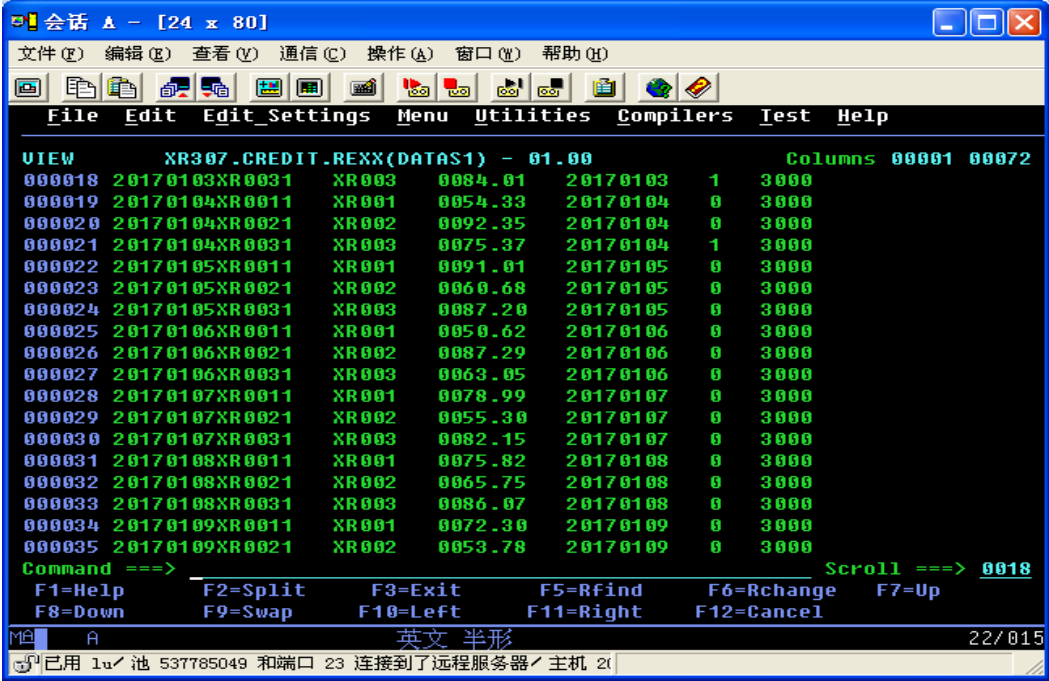
图 3-1 (b) 生成测试数据代码



```
VIEW      XR307.CREDIT.REXX(CREDITB) - 01.99          Columns 00001 00072
004750      END
004760      IF DAY == 31 THEN
004770          DO
004780              MONTH = MONTH+1
004781              MONTH = 01 MONTH
004790              DAY = 01
004791          END
004800      END
004900      TradeInfo_Output.0=TradeInfo.0
005000      DO I=1 TO TradeInfo_Output.0-1
005110          TradeInfo_Output.I=TradeInfo.I.TradeID          || ' ' ||,
005120              TradeInfo.I.TradeUser          || ' ' ||,
005130              RIGHT(TRUNC(TradeInfo.I.TradeAmount,2),7,0) || ' ' ||,
005140              TradeInfo.I.TradeTime          || ' ' ||,
005150              TradeInfo.I.TradeType          || ' ' ||,
005160              TradeInfo.I.MaxLimit
005700          /** SAY TradeInfo_Output.I ***/
005800
Command ==> Scroll ==> 0018
F1=Help      F2=Split      F3=Exit      F5=RFind      F6=Rchange      F7=Up
F8=Down      F9=Swap      F10=Left      F11=Right      F12=Cancel
MA A 英文 半形 22/015
已用 1u 池 537785049 和端口 23 连接到了远程服务器/ 主机 20
```

图 3-1 (c) 生成测试数据代码

生成的数据格式也是符合预期设计的要求，具体如下图 3-2 所示：



```
VIEW      XR307.CREDIT.REXX(DATAS1) - 01.00          Columns 00001 00072
000018  20170103XR0031  XR003  0084.01  20170103  1  3000
000019  20170104XR0011  XR001  0054.33  20170104  0  3000
000020  20170104XR0021  XR002  0092.35  20170104  0  3000
000021  20170104XR0031  XR003  0075.37  20170104  1  3000
000022  20170105XR0011  XR001  0091.01  20170105  0  3000
000023  20170105XR0021  XR002  0060.68  20170105  0  3000
000024  20170105XR0031  XR003  0087.20  20170105  0  3000
000025  20170106XR0011  XR001  0050.62  20170106  0  3000
000026  20170106XR0021  XR002  0087.29  20170106  0  3000
000027  20170106XR0031  XR003  0063.05  20170106  0  3000
000028  20170107XR0011  XR001  0078.99  20170107  0  3000
000029  20170107XR0021  XR002  0055.30  20170107  0  3000
000030  20170107XR0031  XR003  0082.15  20170107  0  3000
000031  20170108XR0011  XR001  0075.82  20170108  0  3000
000032  20170108XR0021  XR002  0065.75  20170108  0  3000
000033  20170108XR0031  XR003  0086.07  20170108  0  3000
000034  20170109XR0011  XR001  0072.30  20170109  0  3000
000035  20170109XR0021  XR002  0053.78  20170109  0  3000
Command ==> Scroll ==> 0018
F1=Help      F2=Split      F3=Exit      F5=RFind      F6=Rchange      F7=Up
F8=Down      F9=Swap      F10=Left      F11=Right      F12=Cancel
MA A 英文 半形 22/015
已用 1u 池 537785049 和端口 23 连接到了远程服务器/ 主机 20
```

图 3-2 生成的初步交易数据格式

3.2 后续实施计划

根据前期任务完成度的情况，项目主要后续工程环节主要包括以下阶段：

项目详细实现阶段：使用 COBOL 语言对消费记录进行完整的读取，并根据所设计的利息计算逻辑进行各用户的利息计算，最后生成账单。

项目测试阶段：使用多种生成的数据对系统进行测试，观察输出结果，完成测试报告。

项目结题阶段：根据前期完成任务以及后期计划实施的情况完成项目的最终报告。

接下来的时间，我们的具体开发计划如下：

1. 利用空闲时间，小组成员一起研究如何生成更为有效和规则的数据格式，并在大机上编译通过运行。
2. 在 12 月中旬之前设计出 COBOL 的 demo 程序，结合测试数据实现基本的功能需求。
3. 月底到答辩前的工作主要围绕着软件产品性能的调优和结题报告书来展开，会给出系统的详细需求分析书和开发手册。

参考文献

- [1]信用卡基本常识[EB/OL].(2016-11-25). <http://jingyan.baidu.com/article/4ae03de31.html>
- [2]修正 zos 安装配置启动错误[EB/OL].(2016-11-25). <https://zhidao.baidu.com/question/118208.html>
- [3]大型机在 Windows 下环境的搭建[EB/OL].(2016-11-25). <http://blog.csdn.net/tuliangde/article/details>