文章编号：1005－8893（2001）02－0033－03

外部调用接口在 CICS/400 中的应用*

郑笑峰[1]，马正华[2]
（1. 中国农业银行常州分行 信息电脑中心，江苏 常州 213001；2. 江苏石油化工学院 计算机科学与工程系，江苏 常州 213016）

摘要：主要阐明了客户信息控制系统和外部调用接口的基本概念及其应用环境，介绍了在外部调用接口中几个重要参数的使用方法，另外具体探讨了 CICS/400 客户端编程的方法，并以一应用实例论述了外部调用接口在 CICS/400 客户端编程的应用。
关键词：客户信息控制系统；客户信息控制系统/400；客户端编程；外部调用接口
中图分类号：TP 311　　文献标识码：A

1　CICS & ECI 概述

CICS（Customer Information Control System）客户信息控制系统是 IBM 公司的软件产品的成员。基于不同的操作系统平台，有 CICS/6000，CICS/400，CICS/ESA，CICS/VSE，CICS/OS2 等多种产品。

CICS 的应用是为商业应用提供一个交易环境。所谓交易环境是指应用程序对数据的修改或是全部成功或是全部失败。CICS 提供这样一个环境，其中运行的交易或程序失败时，它对文件记录或数据项的修改都被自动回滚。应用程序员不必编著写失败处理程序；CICS 系统明确地记录了应用程序对数据的修改，并当交易或应用程序失败后，自动地把对数据的修改回滚。CICS 环境的最大好处是它所提供的完整性知识不需要在应用程序中具有明显的处理逻辑[1]。

非CICS 应用程序，即不包含 EXEC CICS 命令的应用程序，可以使用 ECI 调用 CICS/400 应用程序。这个客户程序调用 CICS 程序的方法类似于 EXEC CICS LINK 命令，而被调用的 CICS 程序称为服务器程序。客户程序调用服务器程序时需要一个服务器程序名和可选的通信存储区，通过通信存储区，客户程序将数据传递给服务器程序，还可以通过这个通信存储区接收服务器程序修改过的数据。另外 ECI 调用时，还可以传递安全性参数、交易名和其它可选控参数。

1.1　逻辑工作单元（LUW）

当 CICS 应用服务器程序执行完成后，这个程序对可恢复资源的修改可保持在未提交状态。这个程序的 LUW 由调用它的 ECI 参数来指定。若 ECI 调用是扩展方式，则 ECI 调用所确定的 LUW 并没有被提交，而后的 ECI 调用可以运行另外服务器程序，提交这个 LUW 或回滚这个 LUW。若 LUW 是扩展的，一个记号或 LUW 标识符在第一次 ECI 调用时将被返回，客户程序可以从 eci_luw_token 参数中得到，而后的 ECI 调用必须使用这个值以确定相同的 LUW。对于一个新的 LUW，eci_luw_token 参数值必须设置成 0。

1.2　传递参数

ECI 客户程序向 CICS 服务器程序传递参数时需要两个参数：存储区指针 eci_commarea 和 eci_commarea_length。这个存储区的内容被复制到 CICS 域中，并经过 CICS 服务器程序修改，返回给 ECI 客户程序。在数据传输过程中（ECI 到 CICS

* 收稿日期：2001－04－09
作者简介：郑笑峰（1976－），男，江苏常州人，学士，高级程序员，主要从事金融应用软件的开发。

或CICS 到ECI)，数据可以被压缩以减少网络负担和数据流。压缩方法是不发送通信区中数据串中两边的二进制 0。由 ECI 向 CICS 服务器程序发送数据的存储区长度要足够长，以保证 CICS 服务器程序能够返回 ECI 所要的所有结果数据。

1.3 同步和异步调用

有两种类型的 ECI 调用：同步调用和异步调用。所谓同步调用是指 ECI 调用需要等待 CICS 服务器程序在 CICS 域中执行完成后，程序控制权才交给前端客户程序。在 ECI 控制结构中，元素 eci_call_type 指定为 ECI_SYNC 表示 ECI 调用是同步调用；所谓异步调用是指前端客户程序不需要等待CICS 服务器程序执行完成，而直接返回程序控制权。当 CICS 服务器程序执行结束后，使用 ECI REPLY 命令来查询其执行结果。在 ECI 控制结构中，元素 eci_call_type 指定为 ECI_ASYNC 表示调用为异步调用。我们在应用的实际环境中由于需要立即看到调用结果，所以通常使用异步调用方式。

1.4 在 ECI 控制结构中指定交易名

当客户程序使用 ECI 调用服务器程序时，可以指定交易名 eci_transid。若在 ECI 控制结构中指定了 eci_tpn 属性，则 eci_transid 属性就被忽略。这个参数的使用方法取决于发送请求的客户特征：

(1) 对于开放系统上的 CICS 客户，这个字段是指服务于客户请求的交易名。

(2) 在其它环境中，被调用的程序在镜像交易 CPMI 下运行，但被连接到 eci_transid 交易上。被调用的程序可以通过指定的交易名查询交易标识符，有些服务器使用交易标识符来确定被调用程序的安全性和性能属性。如果 ECI 是扩展方式的，则eci_transid 仅仅是指第一个 LUW 交易名。

2 ECI 编程和 ECI 调用实例：

CICS/400 基于客户端的 ECI 程序可以使用 C 语言编写，并且在源文件中要声明一个 C 的头文件cics_eci. h。在 ECI 库中有两个 ECI 函数调用：CICS_ExternalCall 和 CICS_ListSystems：

CICS_ExternalCall 调用 CICS 服务器程序在 CICS 应用服务器中运行，另外还可以：提交一个扩展方式的 LUW；回滚一个扩展方式的 LUW；扩展一个扩展方式的 LUW；检查异步 ECI 调用的状态。函数 CICS-ExternalCall（）具有下列语法：

Short CICS_ExternalCall (ECI_PARMS *)

其中 ECI_PARMS 是C 结构，控制着 CICS 外部调用以及其属性。若其中元素没有显式设置，它的值应该设置为二进制“0”。最好的方法是先将整个结构以二进制“0”填充，然后再给那些需要使用的元素赋值。把结构清空的方法是：

ECI_PARMS ECI_control;

Memset (&ECI_control, 0, sizeof (ECI_control));

以下列举一 ECI 调用实例：

```
int MakeEciCallToHost()
{
  ECI_PARMS eci_parms;
  int ReturnCode;
  char EciComm[2048];
  int TIA_LEN;
  char err_code[5];

  memset(EciComm,'\0',sizeof(EciComm));
  memset((char*)&eci_parms,'\0',sizeof(ECI_PARMS));
  TIA_LEN= sizeof(TIA);/* TIA 为从终端输入传给AS/400 的数据*/
  sprintf(EciComm,FMTIsc,ECITIA);
  EciComm[TIA_LEN+1]='\0';
  ascii_to_ebcdic_string(EciComm);/*由于AS/400 使用 EBCD 码,所以将字符串转换成由 ASCII 码转换成 EBCD 码*/ strcpy(eci_parms.eci_system_name,CICSRGN);
  /*服务器端 CICS 区域名*/
  strcpy(eci_parms.eci_program_name,"SCOS051");
  /* ECI 调用的服务器端程序名*/
  strcpy(eci_parms.eci_userid,CICSUSR);
  /* CICS 程序执行的用户号*/
  strcpy(eci_parms.eci_password,CICSPWD);
  /* CICS 程序执行的用户口令*/
  strcpy(eci_parms.eci_transid,"SCCT");
  /* CICS 程序调用的服务器端交易号*/
  eci_parms.eci_call_type= ECI_SYNC;
/*采用同步调用方式*/
eci_parms.eci_commarea=EciComm;
```

```
/*指定 CICS 调用通信区*/
 eci_parms. eci_commarea_length = sizeof (EciComm);
/*指定 CICS 调用通信区长度*/
eci_parms.eci_version=ECI_VERSION_1A;
/*指定 ECI 调用版本号*/
 eci_parms. eci_extend_mode = ECI_NO_EXTEND;
/*指定 CICS 控制域中 LUW 采用的组织方式*/
eci_parms.eci_luw_token=ECI_LUW_NEW;
eci_parms.eci_timeout=0;
ReturnCode=CICS_ExternalCall( &eci_parms);
/*调用服务器端程序*/
if(ReturnCode!=ECI_NO_ERROR)
      if ( ReturnCode = = ECI_ER-
R_SYSTEM_ERROR)
   {
   RPT_ERR(prgname, errname, "ecicall error!");
   return(CNST_FAIL);
   }
ebcdic_to_ascii_string(EciComm);
tscanf(EciComm, FMTOsc, ECITOA);
if(TOA.MSG_TYPE=='E'){
      strncpy(err_code, TOA.MSG_DATA, 4);
      RPT_ERR(prgname, errname, err_code);
      return(CNST_FAIL);
   }
return(CNST_SUCC);
}/* end of ECICALL */
```

在上例 MakeEciCallToHost 函数中，字符数组 EciComm 作为通信存储区，TIA_LEN 作为通信存储区的长度。

3 结 论

通过使用 ECI CALL 调用能方便地实现客户/服务器编程，避免了使用 TCP/IP 套接字编程带来的不便。以上介绍的虽然是基于 CICS/400 环境中客户端 ECI 编程方法，但也同样适用于 CICS/6000，CICS/ESA，CICS/VSE 等环境。

参考文献：

[1] Neil Kolban. CICS/6000 应用开发指南 [M]. 刘宝华，译. 北京：电子工业出版社，1999. 181－186.

The Application of ECI in CICS/400

ZHENG Xiao－feng[1]，MA Zheng－hua[2]

(1. Information & Computer Center，Agricultural Bank of China，Changzhou Branch，Changzhou 213001，China；2. Department of Computer Science and Engineering，Jiangsu Institute of Petrochemical Technology，Changzhou 213016，China)

Abstract：This paper discusses the basic concept and the application environment of CICS and ECI，and it also introduces the methods of several important parameters in external call interfaces. Moreover，the paper clarifies the method of client programming for CICS/400，and it describes the use of ECI in client/server applications with an example.

Key words：CICS；CICS/400；client programming；ECI