SESSION REPORT

SHARE

61	M572	Experience with PCS/ADS as an Application Development Tool	20
SHARE NO.	SESSION NO.	SESSION TITLE	ATTENDANCE

Application Development Management	Elliot Chikofsky	UIH
PROJECT	SESSION CHAIRMAN	INST. CODE

University of Michigan, Industrial and Operations Eng., 231 W. Engineering Bldg.,
SESSION CHAIRMAN'S COMPANY, ADDRESS, and PHONE NUMBER
Ann Arbor, Michigan 48109 (313) 763-2238

Experience with PCS/ADS as an
Application Development Tool

John W. Gerber

Information Systems Department
The University of Iowa Hospitals and Clinics
Room C500, GH
Iowa City, Iowa 52242

Installation Code: UIH

Application Development Management Project

M572

ABSTRACT

The desire for heightened productivity in the application development process has created an increased interest in application generators. Patient Care System/Application Development System (PCS/ADS) is a general purpose, CICS-based application generator that is used primarily in the health care field. It has been in use at The University of Iowa Hospitals since 1978. In conjunction with other application development tools, it is responsible for a substantial reduction in the amount of time and effort required to develop new applications.

University of Iowa Hospitals Background

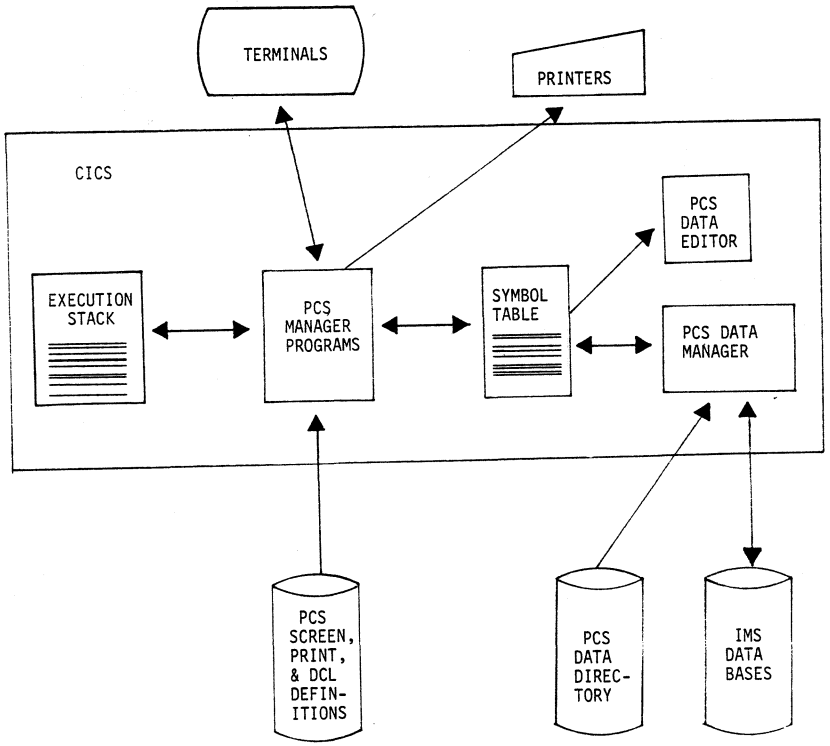
The University of Iowa Hospitals, with 1,100 beds, 1,000 physicians and dentists, 1,400 nursing personnel, and a support staff of 3,500, is the largest university-owned, teaching hospital in the nation. As Iowa's tertiary-level health care center, it served 40,000 inpatients and 335,000 outpatients last year. The Information Systems Department was formed in 1970, and the first application, Clinic Scheduling and Patient Registration, went into production in 1973. At that time, the Hospitals' terminal network consisted of ten terminals and one printer, connected to an IBM 360 Model 40 computer. Today, the Hospital Information System has grown to include 13 major applications running on dual IBM 3081 computers, with a local network of 450 CRT terminals and 150 printers.

When application development began in 1970, it was based on the use of CICS macro-level Assembler language and a structure of indexed and direct files. By 1976, with new applications growing in complexity, increasing requirements for modifications to be made to existing applications, and continuing demand for new information to be added to the files, new development was becoming increasingly more difficult and time consuming. A search was begun for tools that would help speed up the development process and ease future maintenance and modifications. One of the tools acquired in 1978 was PCS/ADS.

PCS/ADS Description

Patient Care System/Application Development System is a very flexible application generator that runs as a CICS transaction. While it was developed at Duke University Hospital and marketed by IBM primarily to health care institutions, it is actually a general-purpose development tool. The keys to its operation are the "Symbol Table", used to pass data, and the "Execution Stack", used to control execution of PCS commands and application programs. Each PCS transaction has its own stack and symbol table. See Figure 1 for a diagram of the PCS Execution Environment. The first detail to notice in the diagram is that all data passage is through the symbol table. Data retrieved from the data base by the PCS Data Manager is placed in the symbol table for use by the application. Data to be passed between parts of an application, such as two different screens, is placed into and retrieved from the symbol table. Data in the symbol table is accessed only by name. Applications are not aware of the structure of the data base or of the format of the data in the symbol table. This provides a high degree of independence between applications and data and helps to minimize the impact that a change to part of an application has on the rest of the application.

The Execution Stack is a last-in/first-out stack that controls the sequence of execution of the parts of an application. Applications can place the names of screens to be displayed or modules to be executed on the stack. PCS will then display the screen or execute the module whose name is currently on top of the stack.



PCS Execution Environment

Figure 1

The PCS Data Editor provides about 30 standard edit functions for use by applications. Data to be edited must be in the symbol table.

The PCS Data Manager provides access to VSAM files or IMS data bases without requiring the application developer to be aware of the data structure in the file or data base. Through the PCS Data Directory, it provides the ability to map data from records or segments to the data lists required by the application. As with the symbol table, the Data Manager helps to isolate applications from physical data structures. Such isolation is extremely desirable when applications or data structures must be changed.

A short example of a PCS application will now be presented to illustrate some typical PCS features. This application consists of five screens that collect a user's sign-on information, present three levels of function menu-selection screens, and finally display the user's profile information. The extensive use of light-pen selectable menu lists is very typical of PCS applications.

Figure 2 shows the sign-on screen "DOCSSION" as it appears on the terminal. Figure 3 shows the screen definition for the DOCSSION screen. There are two parts to the screen definition. The top part defines the appearance of the screen, including the locations and types of input and output fields, and any literal text contained in it. The bottom part of the screen definition is the processing section. This screen has three output fields that are scanned horizontally and three input fields that are scanned vertically. The output fields display the variables T-SYSTEM, which is equal to "IHISNET2", SYSDTTM, which is "07/28/83 15:18", and ERRMSG, which is blank, from the symbol table. The first two input fields move data to the symbol table variables SIGNONID and PASSWORD. The third input field, SIGNOFF, is light-pen detectable and causes the user to be signed off of PCS.

Figure 4 shows the next screen that is displayed after the user ID and password have been successfully entered. It is a PCS menu screen that is part of our own security system and is used only for testing. It allows the application developer to emulate any user environment and perform any application function. In the production environment, the security system uses PCS menu selection screens to restrict each terminal operator to a set of functions that he requires to do his job. In addition, his visibility is limited to patients within his area of the Hospital. In this example, the terminal operator will use the light-pen to select a menu of "Technical Support" functions.

Figure 5 shows the screen definition for the screen that appears in Figure 4. It illustrates some of the processing power available in screen definitions. Notice that for the selected "Technical Support" option, the values of several variables in the symbol table are checked before the name of the next screen to be displayed is placed on top of the stack. If the conditions are not satisfied, this same screen (MSTPCØØØ) is displayed again with an error message. Further up in the definition, there is also a good example of multiple operations being placed on the execution stack. If the terminal user selects "Department", three program names and a screen name will be pushed into the stack, with the last one, screen name MSTPCØØØ, being the first one in and, therefore, the last to be used. The result will be that programs PCSC316, PCSC317, and PCSC318 will be executed in that order and then screen MSTPCØØØ will be displayed.

```
SD43@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
 IHIS - IOWA HOSPITAL INFORMATION SYSTEM - SIGN ON
```



```
PLEASE TYPE YOUR SECURITY ID AND PASSWORD, THEN PRESS THE ENTER KEY.
==========================================================================

          ID:                                        PCS screen as
                                                     displayed!
          PASSWORD:



IHISNET2 -------------------------------- 08/16/83   13:18

DOCSSION                                                               SIGN-OFF
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 2. Screen DOCSSION as displayed

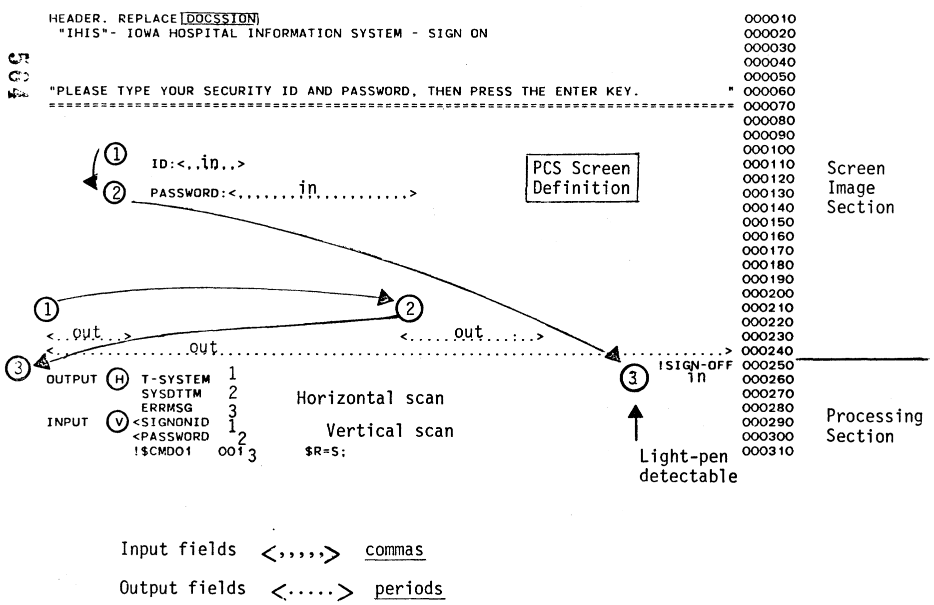


Figure 3. Screen definition for DOCSSION

```
SD43@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
MASTER MENU SELECTION

SELECT DESIRED ACTION(S).
==========================================================================

          DISPLAY USER PROFILE

          MODIFY USER PROFILE
          -------------------
               DEPARTMENT:   ....................
               DIVISION:     ....................
               POPULATION:   ....................
               CRT LOCN:     ....................

          DISPLAY/EXECUTE MASTER MENU SCREEN SELECTION
          --------------------------------------------
               DEPARTMENTAL                  COMPUTER OPERATIONS
               DEMONSTRATION                 TECHNICAL SUPPORT
               SECURITY                                    Light-pen select

IHISNET2 ------------------------------ 08/16/83   13:24   JOHN SMITH ---------

MST000                                                                 SIGN-OFF
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 4. Screen MSTPCØØØ

```
HEADER. REPLACE MSTPC000                                                    000010
 MASTER MENU SELECTION                                                      000020
                                                                            000030
 SELECT DESIRED ACTION(S).                                                  000040
============================================================================000050
                                                                            000060
        !DISPLAY USER PROFILE                                               000070
                                                                            000080
         MODIFY USER PROFILE                                                000090
         -------------------                                                000100
            !DEPARTMENT: <.....................>                           000110
            !DIVISION:   <.....................>                           000120
            !POPULATION: <.....................>                           000130
            !CRT LOCN:   <.....................>                           000140
                                                                            000150
         DISPLAY/EXECUTE MASTER MENU SCREEN SELECTION                       000160
         -------------------------------------------                        000170
            !DEPARTMENTAL                 !COMPUTER OPERATIONS             000180
            !DEMONSTRATION                !TECHNICAL SUPPORT               000181  12/09/82
            !SECURITY                                                       000182  12/09/82
                                                  Light-pen select          000210
                                                                            000220
<.........>                               <.............:..><.....................> 000230
<.....................................................................................> 000240
                                                                !SIGN-OFF   000250
OUTPUT  H  DEP-NAME                                                         000760
           DIV-NAME                                                         000770
           POP-NAME                                                         000780
           UNT-NAME                                                         000790
           T-SYSTEM                                                         000800
           SYSDTTM                                                          000810
           SYSOPNAM                                                         000820
           ERRMSG                                                           000830
*                                                                           000840
* DISPLAY USER PROFILE INFORMATION                                          000850
INPUT   H !$CMD01   001       $PROG=PCSC321,$S=MSTPC000;                    000860
* CHANGE DEPARTMENT                                                        000870
          !$CMD01   001       $PROG=PCSC316,$PROG=PCSC317,$PROG=PCSC318,    000871  10/26/82
                              $S=MSTPC000;                                  000872  10/26/82
* CHANGE DIVISION                                                          000900
          !$CMD01   001       $IF=(((T-SYSTEM='IHISNETA'¦ T-SYSTEM='IHISNETB')&  000901  10/14/82
                              T-DEPT='INF')                                 000902  10/14/82
                                THEN 'ERRMSG=*** ERROR - USER CAN NOT PERFORM TH 000903  10/14/82
                              IS FUNCTION ***,$S=MSTPC000'                  000904  10/14/82
                                ELSE '$PROG=PCSC317,$PROG=PCSC318,$S=MSTPC000'); 000905  10/14/82
                                                                            000920
* CHANGE POPULATION                                                        000930
          !$CMD01   001       $PROG=PCSC318,$S=MSTPC000;                    000940
* CHANGE UNIT                                                              000950
          !$CMD01   001       $PROG=PCSC319,$S=MSTPC000;                    000960
* DEPARTMENTAL                                                             000961  10/14/82
          !$CMD01   001       $IF=((T-SYSTEM='IHISNETA'¦ T-SYSTEM='IHISNETB')  000962  10/14/82
                                THEN '$S=PCSPC357,$PROG=PCSC320'            000963  10/14/82
                                ELSE '$PROG=PCSC320');                      000980
* COMPUTER OPERATIONS                                                      000981  10/26/82
          !$CMD01   001       $IF=(((T-SYSTEM='IHISNET2'¦ T-SYSTEM='IHISNET1') ¦ 000982  10/26/82
                              (T-DEPT='INF' &  T-DIVSN='TOP'))              000983  10/26/82
                                THEN '$S=MSTPC2WO'                          000984  10/26/82
                                ELSE 'ERRMSG=*** ERROR - USER NOT AUTHORIZED FOR 000985  10/26/82
                              THIS FUNCTION ***,$S=MSTPC000');              001000
* DEMONSTRATION                                                            001001  10/14/82
          !$CMD01   001       $IF=((T-SYSTEM='IHISNETA'¦ T-SYSTEM='IHISNETB')  001002  10/14/82
                                THEN 'ERRMSG=*** ERROR - DEMONSTRATION NOT VALID 001003  10/14/82
                              ON THIS NETWORK ***,$S=MSTPC000'              001004  10/14/82
                                ELSE '$S=PCSPC354');                        001005  12/09/82
* SYSTEMS SUPPORT                                                          001021  10/26/82
          !$CMD01   001       $IF=(((T-SYSTEM='IHISNET2'¦ T-SYSTEM='IHISNET1') ¦ 001022  10/26/82
                              (T-DEPT='INF' &  T-DIVSN='TOP'))              001023  12/09/82
                                THEN '$S=MSTPC2LO'                          001024  10/26/82
                                ELSE 'ERRMSG=*** ERROR - USER NOT AUTHORIZED FOR 001025  10/26/82
                              THIS FUNCTION ***,$S=MSTPC000');              001026  10/14/82
* SECURITY                                                                 001027  10/26/82
          !$CMD01   001       $PROG=PCSC306,$S=MSTPC000;                    001060
* SIGN OFF                                                                 001070
          !$CMD01   001       $R=S;
```

Figure 5. Screen definition for MSTPC000

Figure 6 is an example of the primary screen for the high-level testing facility that is part of PCS. When the test option is turned on, this screen will be displayed after each application screen display, program execution, or Data Collection List (DCL) execution. It shows the command that was executed last (in this case, a PCS-generated command resulting from the conditional logic on the previous screen), the condition code from that command, and the current contents of the execution stack, which in this case contains only one screen name. From this screen, the developer has the option of inspecting and modifying the symbol table or modifying the stack.

Figure 7 shows the stack modification screen. It allows the developer to change the execution flow of the transaction from what was originally coded. Figure 8 is an illustration of the symbol table modification screen. Since all data in PCS is passed through the symbol table, this is a very useful testing and debugging tool. Using these test facilities, a developer can enter and inspect test data and can execute new screens, programs, and DCL's before the transaction that they are part of has been completely developed.

Figure 9 shows the Technical Support menu screen that is displayed when we return to the application. Its display is a result of the selection made on the screen of Figure 4. The light-pen will be used to select the "Security" option on this screen.

Illustrated in Figure 10 is the Security Function Menu Screen displayed as a result of the Security selection on the previous screen. Its screen definition is shown in Figure 11. Notice that in the coding for the selection that will be made on this screen, "Review Your User Profile", there are two $P commands that will be placed on the stack. They are the Data Collection Lists (DCL) PCSPD3Ø3 and PCSPD3Ø9. They are illustrated in Figures 12 and 13.

The PCS Data Collection List is a very powerful processing facility. Its capabilities fall between those of a PCS screen definition and a conventional program. The two DCL's shown here are very simple examples. The first one, in Figure 12, clears all of the listed variables out of the symbol table. This is a default for any variable named in a DCL. Since no other value is assigned to any of these variables in this DCL, they remain cleared. The second DCL, in Figure 13, is shorter but illustrates an interesting feature of DCL's. The first statement sets the variable "F1" in the symbol table to the value "ENTR/UPOPERID/SIGNONID". The second statement invokes program "$FN" to assign a value to variable "F1CC" in the symbol table. In order for the DCL to complete, $FN must assign a value to F1CC. This is a general feature of DCL's. The DCL will continue to be re-executed until all data specified for collection by the DCL has been collected.

Figure 14 shows the final screen of the transaction, a display of the terminal user's security profile.

In all of the screen definitions and DCL's that made up this example, no data base accesses were shown. However, it is very simple to issue a command from either a screen definition or a DCL that will move data between the symbol table and a data base. An example of such a command is "$DM=(GET=TESDATA)". In this case, there would be an entry for "TESDATA" in the PCS Data Directory that maps fields in the data base, possibly in multiple segments, onto variables in the PCS symbol table.

```
SD43@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
------------------------- DEBUG PRIMARY OPTION MENU ------------------------------
SELECT OPTION ===> 3                                          ---------------------
                                                              | SYSTEM STACK
  1   STACK      - MANIPULATE THE SYSTEM STACK.               | CMD    OPERAND
  2   SYMBOL     - MANIPULATE THE SYMBOL TABLE.               |
  3   EXIT       - EXECUTE NEXT STACK ITEM.                   |
  4   END DEBUG  - END DEBUG MODE (OR PF3)                    |
                                                              |
  USE OPTION  3  OR PROBE  CONTINUE                           |
  TO EXECUTE THE NEXT STACK ITEM.                             |
                                                              |
-------------------                                           |
  MOST RECENTLY   |                                           |
EXECUTED COMMAND  |                                           |
  $EX %SET0001    |                                           |
------------------|                                           |
  RETURN CODE     |                                           |
 AFTER EXECUTION  |                                           |
      0000        |                                           | $S     MSTPC2L0  BASE
-------------------                                           ---------------------

IHISNET2 ------------------------------ 08/16/83  13:24  JOHN SMITH --------

DEBUG001  ..................................................           CONTINUE
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 6. PCS/ADS Debug screen

```
SD43@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
------------------------------ DEBUG STACK MENU -----------------------------
SELECT OPTION ===> 1                                          ---------------------
                                                              | SYSTEM STACK
  1  MODIFY      - MODIFY STACK AS SHOWN.                     | CMD    OPERAND
  2  RETURN      - RETURN TO PRIMARY OPTION MENU.             |
                                                              |
  TO ADD STACK ITEMS, ENTER THEM ANYWHERE IN THE              |
  STACK ( BLANK STACK ITEMS WILL BE IGNORED ).                |
  TO DELETE STACK ITEMS, BLANK ( NULLS WILL NOT               |
  WORK ) THE CMD FIELD.                                       |
                                                              |
                                                              |
                                                              |
                                                              |
                                                              | $S     MSTPC2L0  BASE
                                                              |
----------------------------------------                      |
                                                              |
DEBUG003                                                      |
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 7. Stack modification screen

```
SD430@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
 -------------------------- DEBUG SYMBOL TABLE MENU ------------------------------
SELECT OPTION ===> 4
  1  ENTER       - ENTER NAMED SYMBOL.
  2  ENTERX      - ENTER NAMED SYMBOL ( VALUE IN HEX ).
  3  DELETE      - DELETE NAMED SYMBOL.
  4  SCAN        - RETRIEVE SYMBOLS BEGINNING WITH NAMED SYMBOL.
  5  RETURN      - RETURN TO PRIMARY OPTION MENU.
  SYMBOL NAME  ===> PATNUM3
  SYMBOL VALUE ===>

 SYMBOL    LEN  VALUE
CASEKEY    005  4040404040
CASEUPDT   001  40
DOCSCMD    004  $EX
DOCSOPER   008  %SET0001
DOCSSCR    006  000000224C70
ERRMSG     001  40
PASSWORD   020  010101A6A6A6B2B2B2B2B2B2B2B2B2B2B2B2B2
PATNUM1    002  4040
PATNUM2    005  4040404040
PATNUM3    001  40
--------------------------------------------

DEBUG002
?@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 8. Symbol table and modification screen

```
SD430@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
  TECHNICAL SUPPORT MASTER MENU

  PROBE DESIRED ACTION(S).
==================================================================================

  BROADCAST MESSAGE ........... BRDC       APPLY BATCH TO ONLINE ......... EPBU
  DISPLAY A PCS SCREEN ........ CALL       EXTRA PARTITION FILE DEQUEUE .. EPDQ
  CICS MASTER TERMINAL FUNC.... CEMT       DEBUG FILE OPTIONS ............ FILE
  REPRINT MESSAGES                         HIERAM CALL ................... HIER
  DEBUG ....................... CORE       INITIATE LAB PROCESSING ....... LBUP
  CICS FE TEST ................ CSFE       OMEGAMON ...................... OMON
  SYSTEM KEYPOINTING .......... CSKP       STATUS OF BATCH UPDATES ....... QBAT
  SYSTEM STATUS INQUIRY ....... CSMT       QUERY STATUS OF LAB QUEUES .... QLAB
  CICS SIGN-OFF ............... CSSF       DISPLAY STATUS OF TERMINALS ... STAT
  CICS SIGN-ON ................ CSSN       OLD DATABASE ACCESS METHODS ... TDAM
  CICS SEND WTO MESSAGE........ CWTO       UPDATE NEXT AVAILABLE PT. NUMBERS
  DISPLAY DCT ................. DCT        DATABASE UTILITY TRANSACTION .. UTIL
  DATA DICTIONARY SHUT DOWN ... DDSD       CICS STATISTICS FUNCTIONS ..... CSTT
  DATA DICTIONARY START UP .... DDSU       PCS-ADS STATISTICS DISPLAY .... SDSP
  DATA DICTIONARY TRANSACTION . DICT

IHISNET2 -------------------------------- 08/16/83  13:25  JOHN SMITH ---------

MST2L0    IHIS OVERVIEW                                  [SECURITY]      SIGN-OFF
?@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 9. Technical Support function menu screen

```
SD43@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
USER PROFILE SECURITY - FUNCTION MENU


SELECT A FUNCTION FROM THE LIST BELOW.
================================================================================

                         CHANGE YOUR PASSWORD

                        |REVIEW YOUR USER PROFILE|







IHISNET2 ------------------------------ 08/16/83  13:25  JOHN SMITH ----------

PCS304                                                  MASTER     SIGN-OFF
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 10. Security function menu screen

```
HEADER. REPLACE PCSPC304
 USER PROFILE SECURITY - FUNCTION MENU



"SELECT A FUNCTION FROM THE LIST BELOW.                                       "
==============================================================================


                   !CHANGE YOUR PASSWORD

                  |!REVIEW YOUR USER PROFILE|










<........>                                   <.............:..><...................>          12/06/82
<............................................................................>
                                                       !MASTER     !SIGN-OFF
OUTPUT  H  T-SYSTEM                                                                             12/06/82
           SYSDTTM                                                                              12/06/82
           SYSOPNAM
           ERRMSG
INPUT   H !$CMDO1    004 001    T-DBFUNC=PSWD,$P=PCSPD303,$P=PCSPD307,$S=PCSPC313,
                                T-STATUS=20;
                                T-DBFUNC=GU,$P=PCSPD303,$P=PCSPD309,
                               |$PROG=PCSC305,$S=PCSPC324,T-STATUS=10;|
                                $R=M;
                                $R=S;
           $AUTO                $S=PCSPC304;
```

Figure 11. Screen definition for screen PCSPC304

```
DCL. REPLACE PCSPD303
     50      CPDEPTA
     50      CPSNAME
     50      FS
     50      FO
     50      NEWSPNM
     50      NEWSOFNM
     50      OLDSOFNM
     50      OLDSPNM
     50      SCOFCNEW
     50      SCOFCOLD
     50      SUPVNEW
     50      SUPVOLD
     50      T-BRTHDT
     50      T-COUNT
     50      T-DOCNUM
     50      T-DEPT1
     50      T-DEPT2
     50      T-DEPT3
     50      T-NEWPSW
     50      T-OLDPSW
     50      T-OPERID
     50      T-OPSWRD
     50      T-PASSWD
     50      T-REASON
     50      T-SECNME
     50      T-SPVNME
     50      T-SRNNMA
     50      T-SRNNMB
     50      T-TITLE
     50      T-UPPSWD
     50      T-USERID
     50      UPCSSNID
     50      UPCSSNL
     50      UPCSSNPS
     50      UPBRTHMO
     50      UPBRTHDA
     50      UPBRTHYR
     50      UPCURSPV
     50      UPDEPTA
     50      UPDEPTB
     50      UPDEPTC
     50      UPEMNRUM
     50      UPLUPDTN
     50      UPLUPDTT
     50      UPOPERID
     50      UPPSWDTE
     50      UPRUPTQ
     50      UPSECOFC
     50      UPSECLV
     50      UPSNAME
     50      UPSRNID
     50      UPSRNIDB
     50      UPTITLE
     50      UPTRNDTE
     50      UPTRNFLG
     50      UPTRNSCN
     50      USERNAME
```

Figure 12. Data Collection List PCSPD3Ø3

```
DCL. REPLACE PCSPD309
       10    F1       ENTR¦UPOPERID¦SIGNONID:
       10    F1CC     $PROG=$FN:
```

Figure 13. Data Collection List PCSPD309

```
SD43@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@TOP
USER PROFILE SECURITY - DISPLAY

=====================================================================================

                  USER: JOHN SMITH --------------
            BIRTH DATE: 11  11  1911
       EMPLOYEE NUMBER: 999999999

        SECURITY LEVEL: SECURITY FUNCTION I
            DEPARTMENT: INFORMATION SYSTEMS
              DIVISION: TECHNICAL OPERATIONS
            POPULATION: NOT APPLICABLE


            SUPERVISOR: JOHN GERBER -------------
      SECURITY OFFICER: JOHN GERBER -------------


 PASSWORD LAST CHANGED: 08/16/83
  PROFILE LAST UPDATED: 08/16/83   13:16   BY   DALE WILHELM -------

IHISNET2 ------------------------------- 08/16/83   13:26   JOHN SMITH ---------

PCS324                                          SECURITY      MASTER      SIGN-OFF
?@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@BOT
```

Figure 14. User profile display screen

In addition to the Screen Manager and the DCL Processor, there is one additional, widely-used PCS manager that did not appear in the example; that is the Print Manager. The PCS Print Manager uses Print Definitions that are very similar to Screen Definitions in appearance, capability, and use. The difference between them is that the Print Definitions have no input fields. Output can be routed to individual printers or to groups of printers, and routing can be either fixed or dynamic.

The purpose of this brief description has been to give you a feel for how PCS works, and to give an idea of its power, flexibility, and ease of use. With the information that I've presented, you could understand most straightforward PCS transactions, and you could probably code a simple transaction. Compare that with the level of knowledge and effort required to code a command-level CICS transaction.

The Application Development Environment

The application development environment is not just the sum of the tools and methodologies that make it up. A poorly matched set will yield disappointing results. A well chosen set of complementary tools and methodologies will show a synergistic effect, yielding exceptional results. For this reason, it is difficult to evaluate the effectiveness of a part of the development environment, such as PCS/ADS, in isolation.

The development environment at The University of Iowa Hospitals is shown in Figure 15. Combined with a development methodology that emphasizes prototyping and iterative development of large applications, the tools shown have proven to be very satisfactory. However, since all of them, except the Data Directory and Data Dictionary, which are more recent additions, came into use in the same time frame, it is difficult to attribute particular increases in productivity to individual tools.

Since PCS/ADS is so well suited to prototyping and iterative development, I would like to briefly comment about those two techniques. Prototyping involves building a mock-up system for demonstration to the user prior to development of the complete functional system. This allows the user to actually operate a terminal to see what his final system will look like and what it will be like to use. This is much more effective in eliciting comments, criticisms, and suggestions than thirty pages of paper screen simulations. It helps to avoid the "But that isn't what I wanted" types of problems that can occur at the end of a project.

Iterative development is a natural follow-on to prototyping. It involves gradually replacing the simulated data and flow controls of the prototype with real data and application logic. PCS/ADS is ideally suited to these techniques. It is nearly as easy to code PCS Screen and Print Definitions as it is to just lay out an image of the screen on a piece of paper. Fields that will eventually be used to display variables have constants coded into them instead, and what will eventually be processing logic in the screen definition is simply coded as a call to the next screen. These prototype screen definitions are simple enough

The University of Iowa Hospitals and Clinics

Application Development Environment

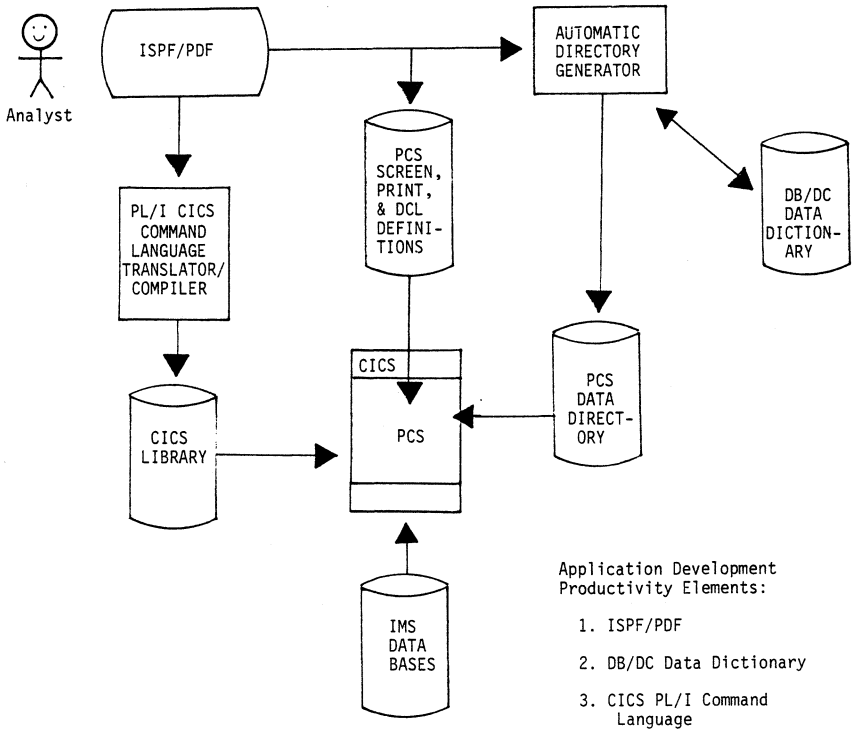


Figure 15. University of Iowa Hospitals Development Environment

so that they can be created or modified by user personnel. Once the prototype screen definitions have been finalized, programs, DCL's, and data base accesses can be added to complete the application. If the user is kept involved during this process, misunderstandings and changes in requirements can be caught before they become major problems.

While I have seen some claims that PCS/ADS applications can be developed by user personnel without the assistance of professional programmer/analysts, and I understand that this is being done at some institutions, it has not been our policy to do so at The University of Iowa Hospitals. There are two primary reasons for this decision. First, our applications are developed around an integrated Hospital Data Base. Its evolution and use must be centrally coordinated and controlled. Second, while it may not be prohibitively difficult to train user personnel to use PCS/ADS effectively, we have not felt that it would be practical to train them to develop, document, and maintain complex interrelated application systems.

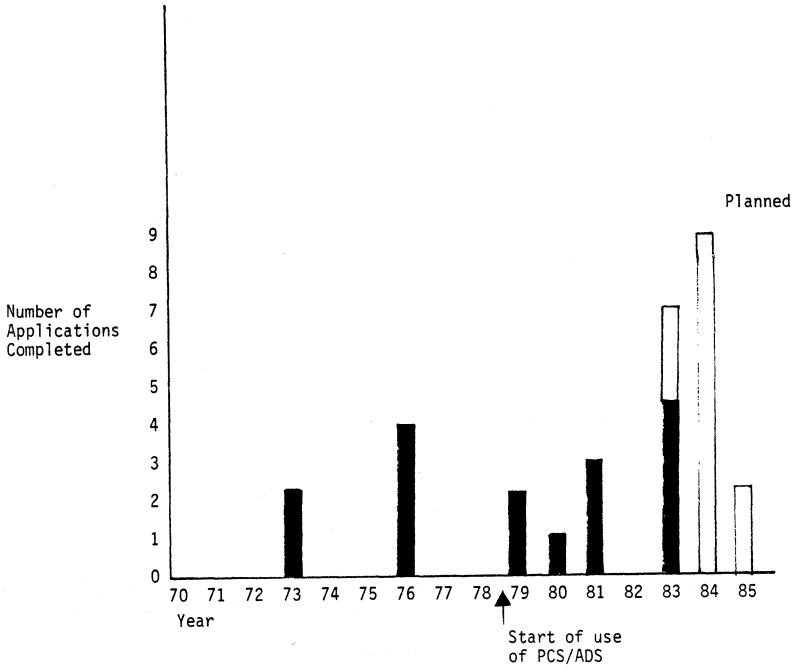
The PCS application developers at The University of Iowa Hospitals have primary skills of systems analysis and design, although they are also able to write any CICS PL/I Command language programs that are required. I feel that this emphasis on analysis, design, and familiarity with the application area has been most beneficial. Serious or fatal flaws in applications systems are usually in the design of the system, rather than in its programming. I think that this emphasis on design, which is a direct result of the power and ease of use of PCS, has enabled us to produce more usable and reliable systems than would have been possible otherwise.

Figure 16 provides a rough measure of the effectiveness of the applications development environment, including PCS/ADS, currently in place at The University of Iowa Hospitals. It shows the number of major applications completed each year since the Department's inception. Those completed to date are comprised of 350 PCS functions (equivalent to transaction types), with 2,000 different screen definitions. While the increased rate after 1978 is not entirely due to the use of PCS/ADS, its use was certainly a factor.

Disadvantages of PCS/ADS

So far, I've discussed the advantages of using PCS/ADS. There are also some disadvantages. The first is the additional processing overhead that is incurred, compared to CICS macro-level, assembler-coded transactions. Since we also began using the IMS Data Base Manager and CICS command-level PL/I coding in the same time frame that we began using PCS, it is difficult to quantify the amount of additional overhead that can be attributed to PCS. We have not considered it to be prohibitive.

Two more problems are a direct result of the ease with which new application systems can be created with PCS. Part of the ease of use of PCS is due to the high level at which it allows the application developer to work. He is isolated from machine procedures and physical data structures. However, this feature also makes it easy to develop transactions that suffer from poor performance



APPLICATION DEVELOPMENT RATE AT

UNIVERSITY OF IOWA HOSPITALS

Figure 16

and use excessive system resources. For this reason, and because of the rapid rate at which new applications can be developed, it is particularly important in a PCS environment to do an effective job of predicting and tracking system resource usage.

There are three more "technical" problems that currently affect the use of PCS/ADS. First, all PCS transactions run under a single CICS transaction name. To CICS, PCS is a single transaction. This prevents assigning differing priorities to different PCS-based applications. Second, the PCS execution modules are written in CICS macro-level Assembler language. Some new CICS facilities can only be used with command level transactions. Finally, the PCS DCL processor runs as a conversational transaction, causing data areas for DCL-based transactions to tie up virtual storage for long periods of time. We have not considered any of these problems to be prohibitive to our use of PCS. Indeed, some of them are by-products of its overriding advantages. In addition, we are optimistic that the technical problems cited above will be resolved.

Summary

PCS/ADS is a general purpose, CICS-based application generator that has been in use at The University of Iowa Hospitals since 1978. It's primary advantages are:

1. The development cycle is speeded up due to the minimal requirements for conventional programming.
2. PCS/ADS facilitates system prototyping and an iterative development process.
3. The reduction of conventional programming requirements allows the systems developer to concentrate on understanding user requirements and on system analysis and design.
4. The systems developer has the flexibility to use screen descriptions, Data Collection Lists, or conventional programs to implement PCS/ADS transactions.
5. Users are kept involved by the iterative development process and may also create or modify screen and print formats.
6. PCS/ADS includes a good, high-level test facility.

We have found these advantages to far outweigh its disadvantages. In conjunction with the other application-development tools and the development methodologies in use here, PCS/ADS has proven to be an effective and valuable tool.

SHARE

SHARE SESSION REPORT

61	M583	Measuring Application Development & Maintenance	450
SHARE NO.	SESSION NO.	SESSION TITLE	ATTENDANCE

ADM	Steve Theby	MA
PROJECT	SESSION CHAIRMAN	INST. CODE

McDonnell Douglas Automation PO Box 516 St. Louis, MO 63166 314-233-3994

SESSION CHAIRMAN'S COMPANY, ADDRESS, AND PHONE NUMBER

ABSTRACT

Consistently defined and applied application development and maintenance measurements are essential to a program to improve the application development and maintenance activity in an organization. Those measures are required:

1. To identify and promote practices which help.
2. To identify and avoid practices which hurt.
3. To support rational estimating processes.
4. To portray productivity improvement trends.

These basic objectives of productivity measurement will be used to define a measure called Function Points. Experience with this measure will be described.