CCPS Project Plan
Architecture Summary

3. Introduction

This section abstracts information from the ***CCPS Architecture Specification*** (Reference 2).

3.1 CCPS Environment

The Environment in which CCPS runs is shown in the attached figure ("CCPS Environment"). CCPS basically functions as a terminal emulator for four legacy systems and is implemented on a PC. These systems have existing electronic interfaces, accessed via different communications mechanisms. It is a requirement that no change be made in these systems, so that CCPS must emulate the existing interfaces.

3.2 Basic CCPS Functions

CCPS allows the operator to enter data using a GUI interface, stores that data, and then selects specific information to deliver to the other systems. CCPS automatically connects to each system and emulates a terminal session that would have been used by an operator to enter the same data.

The operator may also enter a "closure" transaction, indicating that the work implied by the entry transaction has been completed. This information is selectively transmitted to the other systems. In addition, CCPS selects certain data and "Faxes" same to a specified number.

3.3 Ancillary CCPS Functions

There is no requirement for CCPS to maintain a historical record of activity, so that there is no long-term journaling.

CCPS has an internal "watchdog" that provides visual indication that the system is actually functioning by maintaining a clock on the screen.

3.4 CCPS Internals

Basically, the CCPS architecture is connectionless client-server, with a data base serving both as a repository for "orders," and as a communication mechanism. Communication between modules occurs via memory-mapped files.

The logical modules are shown in the attached "CCPS Architecture" and their high level roles are described functionally below. (For more information the reader is referred to the detailed CCPS architecture specification, Reference 2.)

3.4.1 Graphical User Interface (GUI)

The GUI module provides the following principal functions:

- Presents a convenient, uniform interface for order entry and closure
- Does syntactic validity checking during order entry and closure
- Displays any errors discovered by the error monitor
- Passes input data to the (order) Initiator
- Passes close request to the Finder, receives assembled data back
- Passes closure data to the Closer

3.4.2 Initiator

- Receives order from GUI
- Does further error checking
- Sets initial values for all order flags
- Writes order into data base

3.4.3 Closer

- Receives (closed) order from GUI
- Does further error checking
- Sets closure values for all order flags
- Writes closed order into data base
- Notifies FAXer that an order has been closed

3.4.4 Finder

- Receives order identifier from GUI
- Assembles all required information for GUI, returns same to GUI

3.4.5 Sender

- Scans data base queues for changes of state (null to open, open to closed, closed to Faxed)
- Extracts appropriate information and delivers to the Talkers
- Sets delivery status in data base

3.4.6 Talker(s)

Each talker is functionally similar, differing primarily according to its target system protocols:

- Receives data from sender
- Delivers to downstream system
- Sets delivery status in data base

3.4.7 X.25

- Is a hardware device
- Provides X.25 protocol
- Receives from a Talker over the IBM PC bus
- Performs modem function

3.4.8 FAXer

- Receives closure data from Closer

- Gather order data from data base
- Prepares and Faxes closure form to original requester
- Uses standard modem

3.4.9 ErrMon

- Continually scans database queues for timely transitions from state to state
- Return errors to GUI
- Continually checks that all other modules are alive
- Maintains visual status indicator for GUI interface

3.5 Performance

A detailed analysis is given in Reference 2. Basically, the transaction rate is so low that performance is not an issue for the hardware. However, throughput might be limited by the operator's input ability. Current estimates indicate that the operator should be able to keep up with busy hour load. If is a problem in the Beta application, a simple solution is the addition of a second PC.

3.6 Hardware

The detailed hardware specification is given in Reference 2. Basically, it is a modern PC with 528 Mb of memory and a 20 Gb hard disk. It is ordered with LAN boards and FAX modem, and the CCPS project adds the custom X.25 board. A variety of vendors provide similar machines.