

N 60-CR  
233

**GENERIC POCC ARCHITECTURE:  
Revised Recommended Refinements and  
Object-Oriented Interfaces**

Contract Number NAS5-30680  
Task 12

N95-10825

Unclas

63/60 0023653

Prepared For:

NASA/Goddard Space Flight Center  
Data Systems Technology Division  
Greenbelt, MD 20771

June 30, 1989

CTA INCORPORATED  
6116 Executive Boulevard, Suite 800  
Rockville, MD 20853  
(301)816-1200

(NASA-CR-196882) GENERIC POCC  
ARCHITECTURE: REVISED RECOMMENDED  
REFINEMENTS AND OBJECT-ORIENTED  
INTERFACES (Computer Technology  
Associates) 52 p

# 1 INTRODUCTION

This document is a sequel to the report entitled *Generic POCC Architectures*, dated April 5, 1989, prepared by CTA under Contract NAS5-31500, Task 28-11600. That document presented a generic architecture based upon current technology, and a series of three refinements based on object-oriented analysis principles and expectations for POCC evolution. The current document revisits the object-oriented analysis of POCCs. We have reassessed the functional groupings that best adhere to object-oriented principles, and have revised the recommended architecture accordingly.

We present an updated view of the recommended generic POCC architecture using the same graphical models as the previous document:

- Entity-Relationship Diagrams
- Dataflow Diagrams
- Composition Graphs

In addition, we present another view in the form of *Entity-Interface Diagrams* (EIDs). EIDs may be viewed as a precursor to Object Diagrams, which are the basic construct of the General Object-Oriented Design (GOOD) methodology. The Entity-Interface Diagrams, together with their textual annotations (described below), constitute our specification of object-oriented interfaces in the generic architecture.

An arrow in an Entity-Interface Diagram indicates that the source entity *depends on* the destination entity. A textual annotation to the Entity-Interface Diagrams documents the services performed by each entity (the *performs* clause, which is analogous to the *provides* clause in Object Diagrams). The textual annotation also lists the services of other entities on which a given entity depends (the *depends on* clause, which is analogous to the Object Diagram *uses* clause). The terminology of EIDs is different from that of Object Diagrams because the respective concepts are only analogous, not identical. For example, during the transition from specification to design, the direction of dependency between objects may be reversed. In addition, services performed by an entity in an EID may not be provided by the corresponding Object Diagram object; instead, these services may appear as subfunctions of provided services. Other transformations may also occur in the transition from specification to design; thus, it seemed prudent to keep the terminology of EIDs distinct from that used in the design phase.

## 2 REFINEMENTS OF THE GENERIC ARCHITECTURE

The recommended generic architecture in the current document differs from the “state-of-practice” (SOP) generic architecture, which was presented in our previous report. In the following list, we present the refinements by means of which the SOP architecture has been transformed into the recommended architecture. Some of the refinements suggested in our previous report have been overridden or reinterpreted—these issues are discussed following the list.

*Distribute initialization to all subsystems.* Add initialization function to all subsystems, and create an initialization-directive relationship between the Operator subsystem and the other subsystems.

*Make all subsystems responsible for responding to their own directives.* Add a directive processing function to each subsystem, and add a directive routing function to the Operator subsystem.

*Centralize access to the ODB.* Move all ODB access routines to a single subsystem.

*Support interactive maintenance of the ODB.* Add a maintenance function to the ODB subsystem.

*Distribute ODB across subsystems.* Delete the ODB object from the architecture, and add a new subentity to each subsystem. This subentity will contain the portion of the ODB that is written by the subsystem.

*Combine the Operator interface functions into a subsystem.* Create a new subentity in the Operator subsystem which will contain all operator I/O devices. Place the Operator Interface and Display subsystems in the same subsystem.

*Combine control of the External Simulator and AP into a single Operator position.* Add new functions to the Operator subsystem to process External Simulator commands and monitor its operation, and add a control relationship between the Operator subsystem and the External Simulator.

*Provide a Knowledge-based Operator interface.* Add functions to the Operator interface assess S/C and AP status and give Operator advice on these and on the commands he inputs.

*Centralize network interface functions.* Create a single Network subsystem, and move any network interface routines from the other subsystems to the Network subsystem.

*Combine external device drivers into a common subsystem.* Create a single subsystem that contains all external device drivers.

*Place OBC Load and Command verification in a single subsystem.* Remove Command verification from command and place it in Telemetry, or remove OBC verification from Telemetry and place it in command. We have opted for the latter choice.

*Separate CMS interface from Command.* Remove the functions associated with CMS interface from Command and place them in a separate subsystem, or create a separate

subentity in Command that contains the CMS interface routines. The command functions will have to use the new entity to interact with CMS.

*Separate Offline Processing from History.* Remove the functions associated with offline processing from the History subsystem, and place them in a separate subsystem, or create a separate subentity in History that contains all of the offline processing routines. The new entity will have to use the history management routines to access the history data.

*Control Telemetry replay through history.* Add a replay function to the History subsystem.

*Send Telemetry replay to Network.* Add a data transfer relation between History and the Network.

*Place all interaction with the NCC in a single subsystem.* Create a new entity containing all functions associated with AP-NCC interaction, or create a subentity containing these functions and place them in another subsystem.

*Provide internal simulation.* Create a new subsystem containing the functions associated with internal simulation, or place an entity containing these functions in another subsystem.

*Provide direct links between the AP and the External Simulator.* Include control and monitoring relationships between the Operator subsystem and the External Simulator, and include data transfer relationships between the External Simulator and the Network subsystem.

*Distribute some command functions to the customers.* Add functions for receiving and transmitting data between customers and the AP, for verifying the commands received, for scheduling the commands received, and for detecting and resolving command conflict.

*Distribute payload control and data processing to the customers.* Add functions for locating customer requested data and routing this data to the customer, and add functions to monitor and control the use of shared resources.

*Combine the Operator Interface and Display functions.* Combine the Operator Interface and Display entities in the Operator subsystem.

*Place the NCC Interface entity in the Network subsystem.* Move the NCC entity to the Network subsystem, and add the appropriate relationships to the Network subsystem.

*Centralize the device control.* Provide a service subsystem that provides interface routines for controlling external devices.

## **Overriding or Reinterpretation of Previous Recommendations**

*Centralizing ODB access vs. distributing the ODB.* We originally intended to centralize both the ODB access routines *and* the ODB data. Each subsystem, however, would be able to see only that portion of the ODB data that it needed. Our current recommendation is to provide common access functions where appropriate, e.g., through the use of a common Database Management System (DBMS), but *not* to centralize the database itself. Instead, the ODB will be distributed over the subsystems.

We base this opinion on the principle of information hiding. Isolating a data item within the subsystem that operates on it will result in a system that is easier to maintain in the future. Each subsystem's data management techniques can then be selected to match its computational needs. In those systems where DBMS technologies make sense, all of these systems can share the same DBMS, while maintaining different databases. The only requirement for a centralized facility would be an offline data maintenance facility.

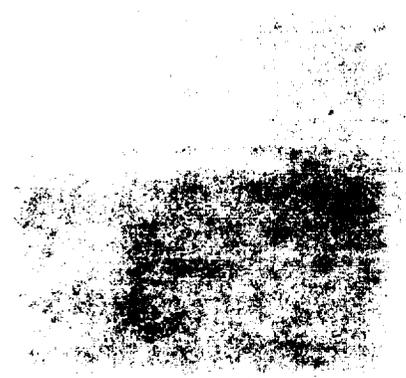
*Centralizing non-network external interfaces vs. leaving them in the subsystems that use them.* In the SOP architecture, we showed all external systems being accessed via the Network subsystem. This is still implicitly true in the recommended architecture, but it is not shown explicitly. Instead, we decided that it is more important to document the application-level interface between external systems and the specific AP subsystems with which they interact. If an external interface is common to more than one AP subsystem, this interface may be encapsulated in a common service-level object that is used by the subsystems in question. As shown in the level-0 Entity-Relationship Diagram, however, we do not see any such cases currently.

*Placing the NCC entity in the Network subsystem vs. combining the NCC interface routines into a single subsystem.* In the SOP architecture we combined all NCC-AP interface routines and data into an NCC entity, which we placed in the Operator subsystem. This placement was based on the fact that the entity's primary purpose is to manage the interaction between the operator and the NCC. The NCC entity provides a means for the operator to query the NCC about network status, and a means for the NCC to determine AP status.

We have since revised our opinion on placement of the NCC entity. In the current recommended architecture, the NCC entity appears within the Network subsystem rather than the Operator subsystem. This change is based on the principle of data encapsulation, in which all functions pertaining to a common set of data elements/types are grouped in a single entity, together with the data. In this case, the common data types are those of the Network. The NCC will probably require knowledge of these data types.

**POCC AP - GENERIC ARCHITECTURE**

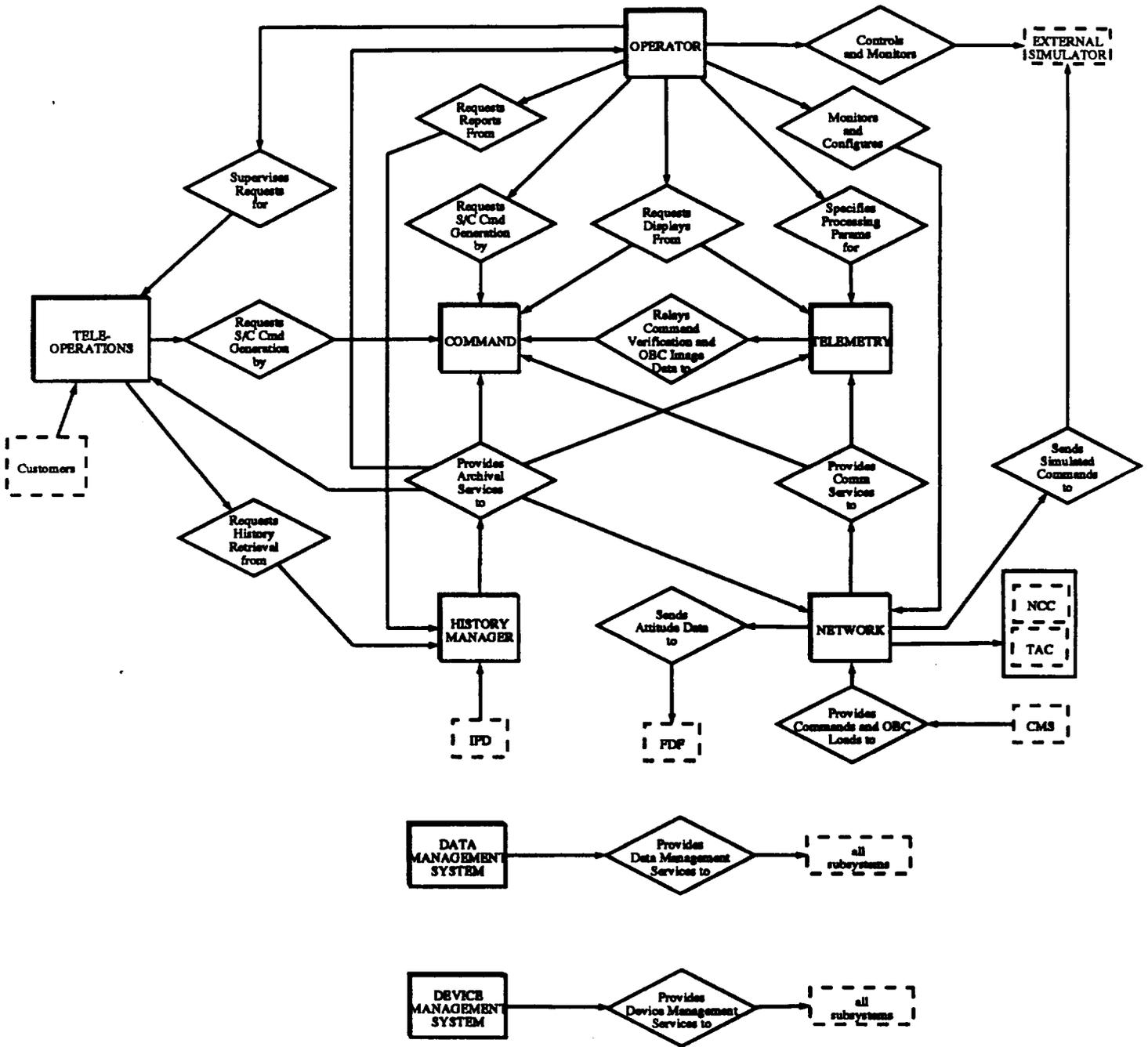
**Entity-Relationship Diagrams**



**ORIGINAL PAGE IS  
OF POOR QUALITY**

POCC AP-GENERIC ARCHITECTURE

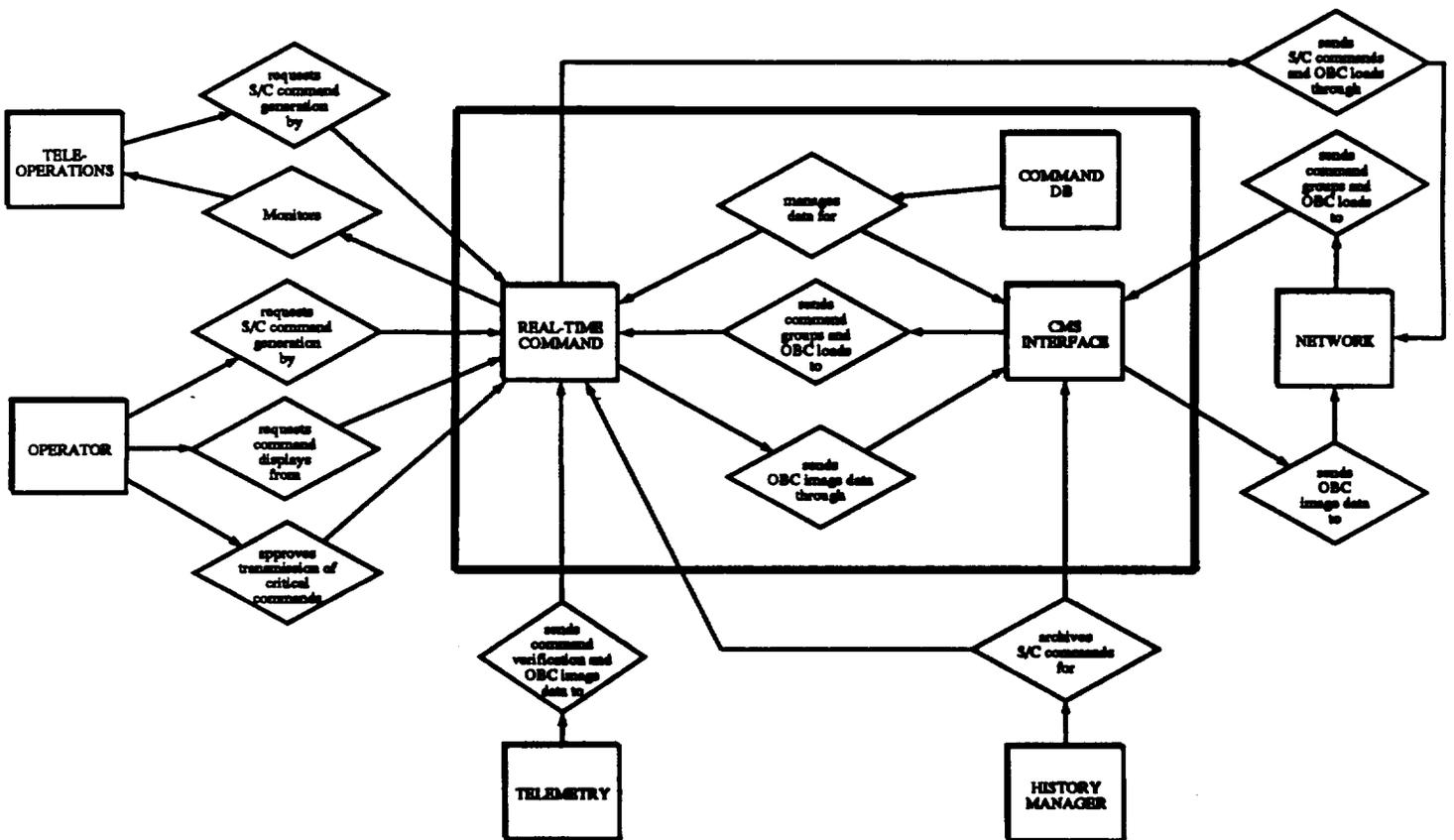
Level 0 Entity Relationship



POCC AP-GENERIC ARCHITECTURE

COMMAND

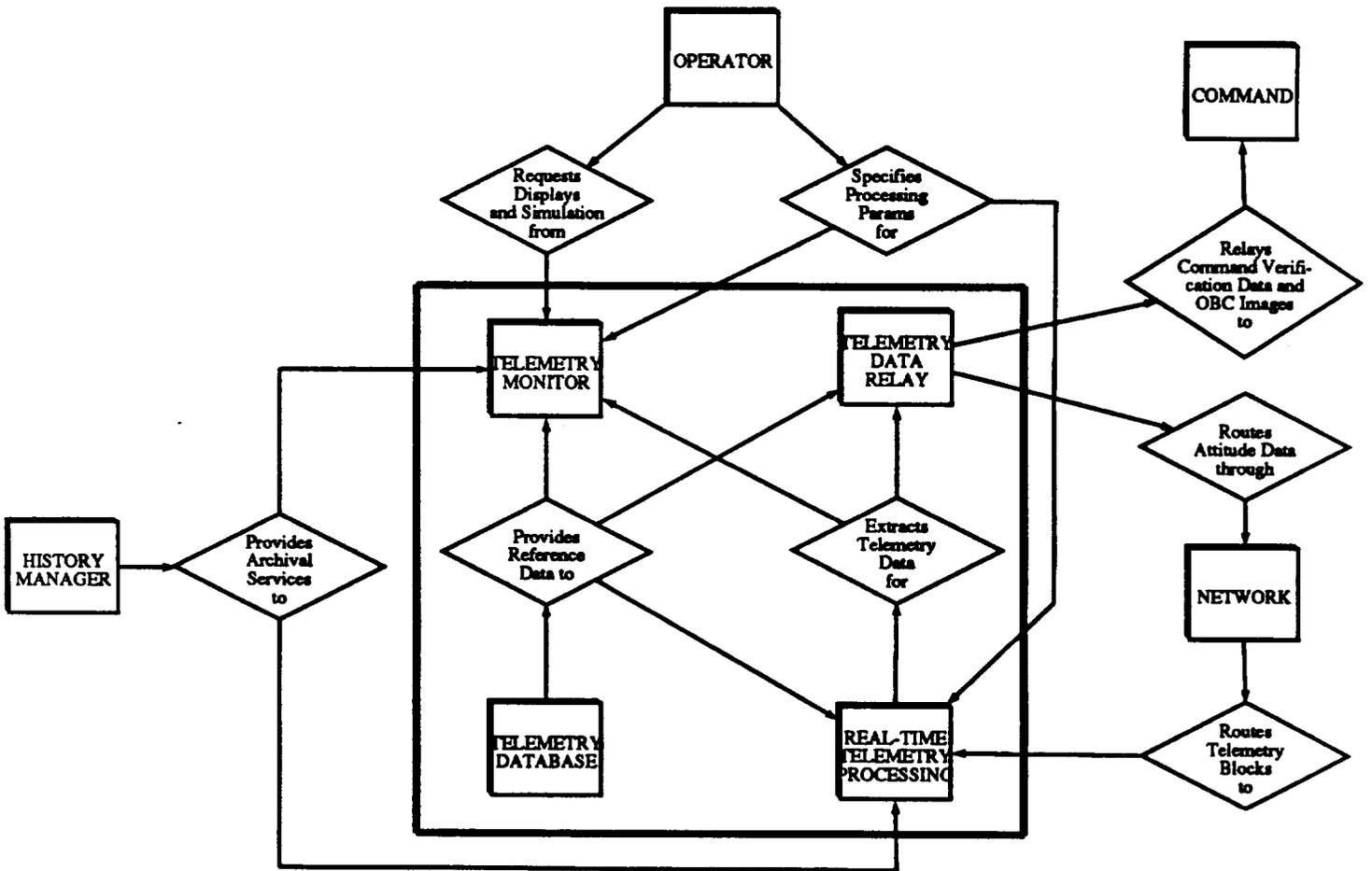
Level 1 Entity Relationship



POCC AP--GENERIC ARCHITECTURE

TELEMETRY

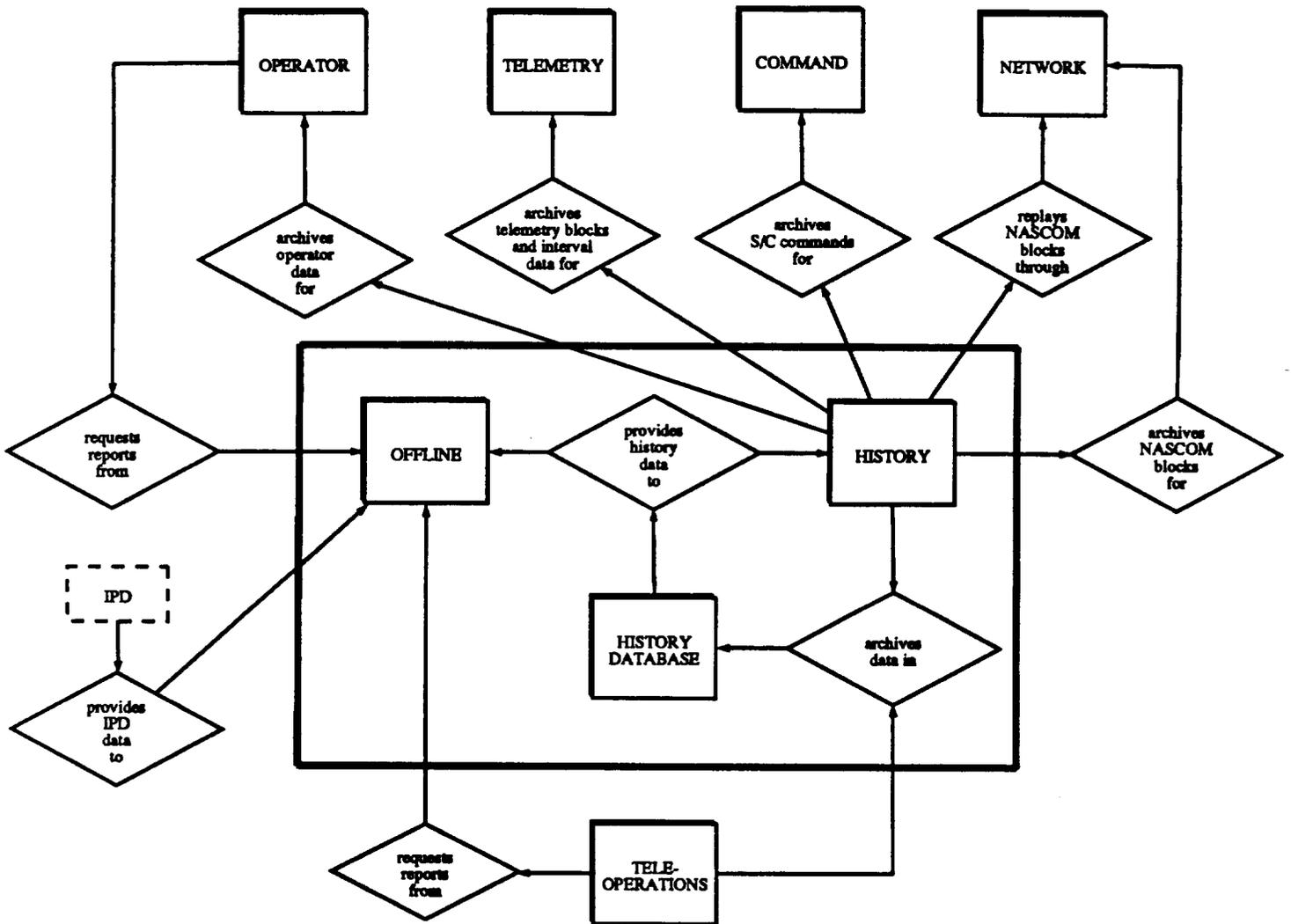
Level 1 Entity Relationship



POCC AP-GENERIC ARCHITECTURE

HISTORY MANAGER

Level 1 Entity Relationship

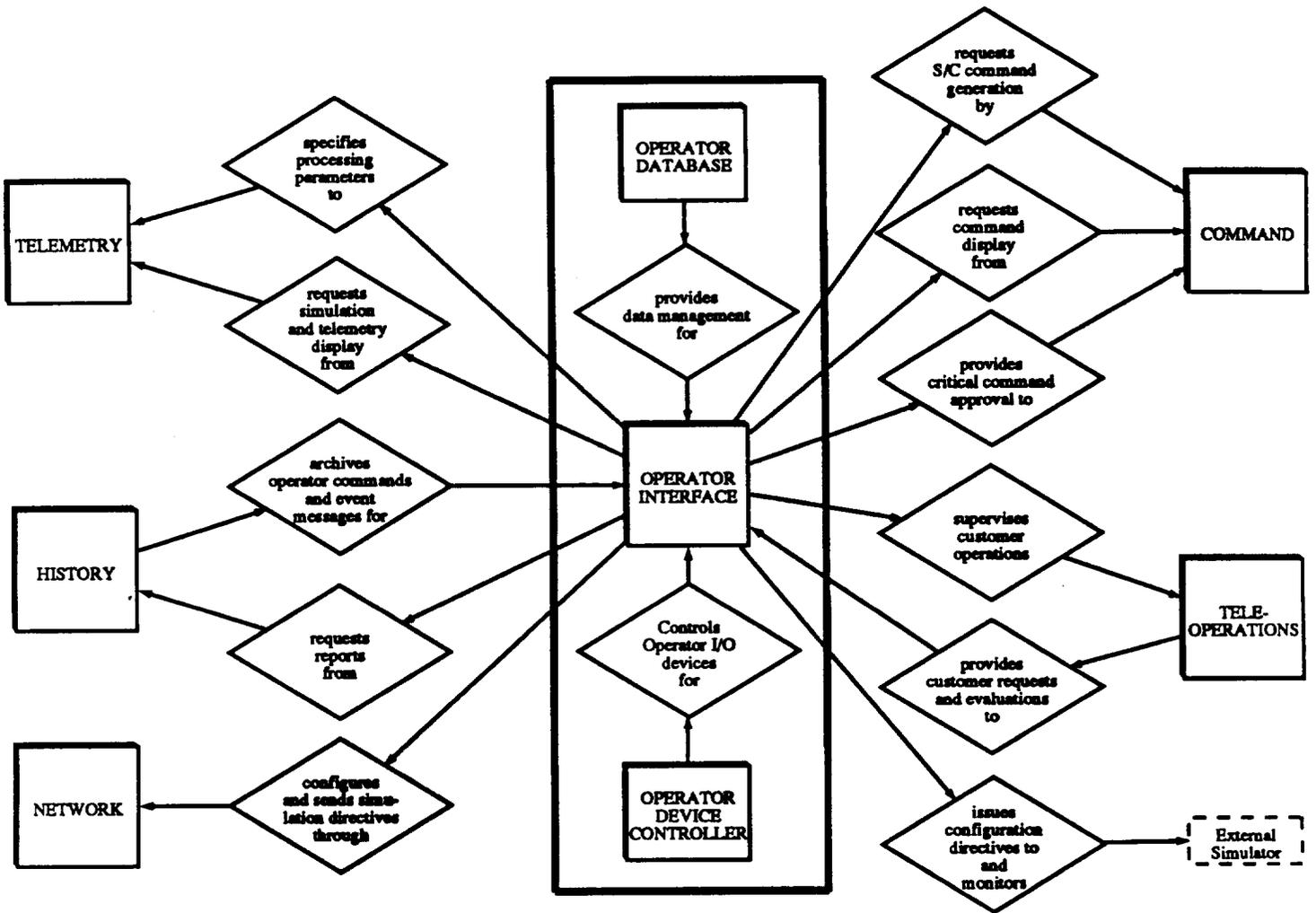




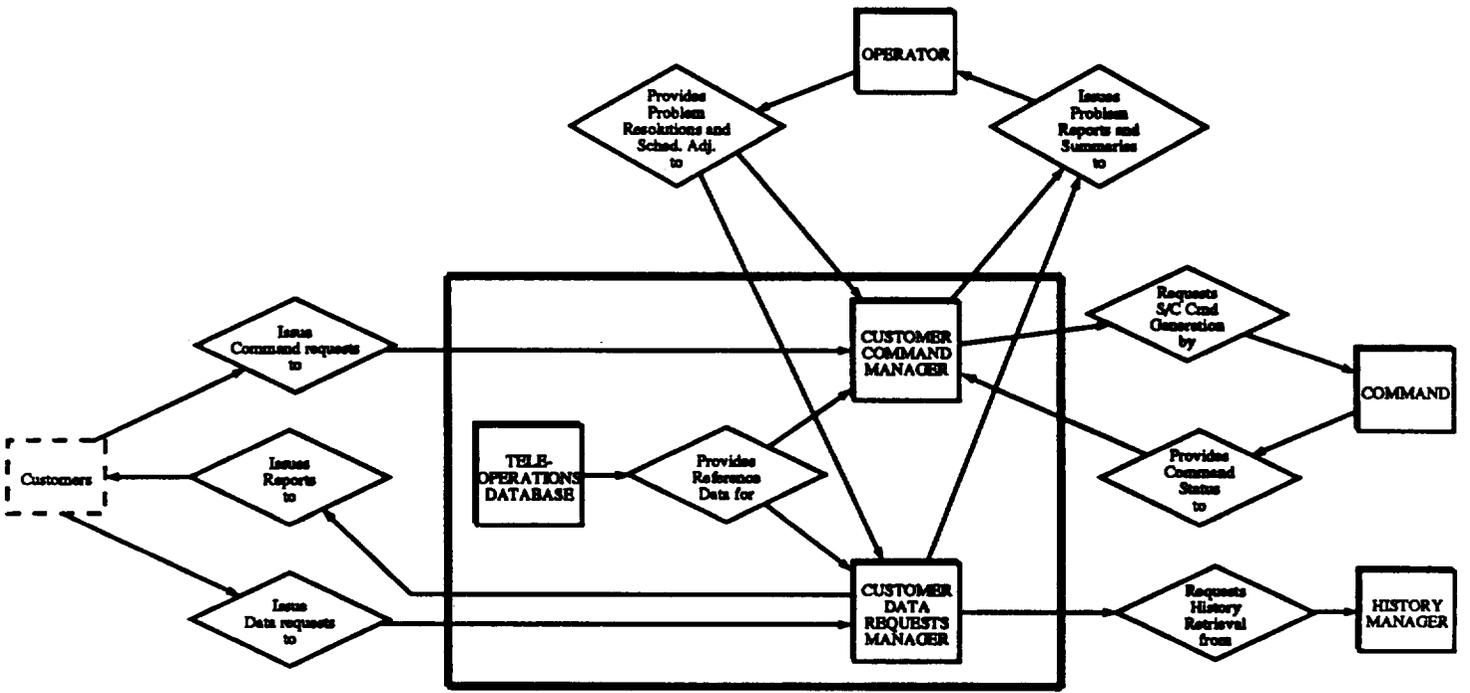
POCC AP-GENERIC ARCHITECTURE

OPERATOR

Level 1 Entity Relationship



POOC AP-GENERIC ARCHITECTURE  
 TELEOPERATIONS  
 Level 1 Entity Relationship



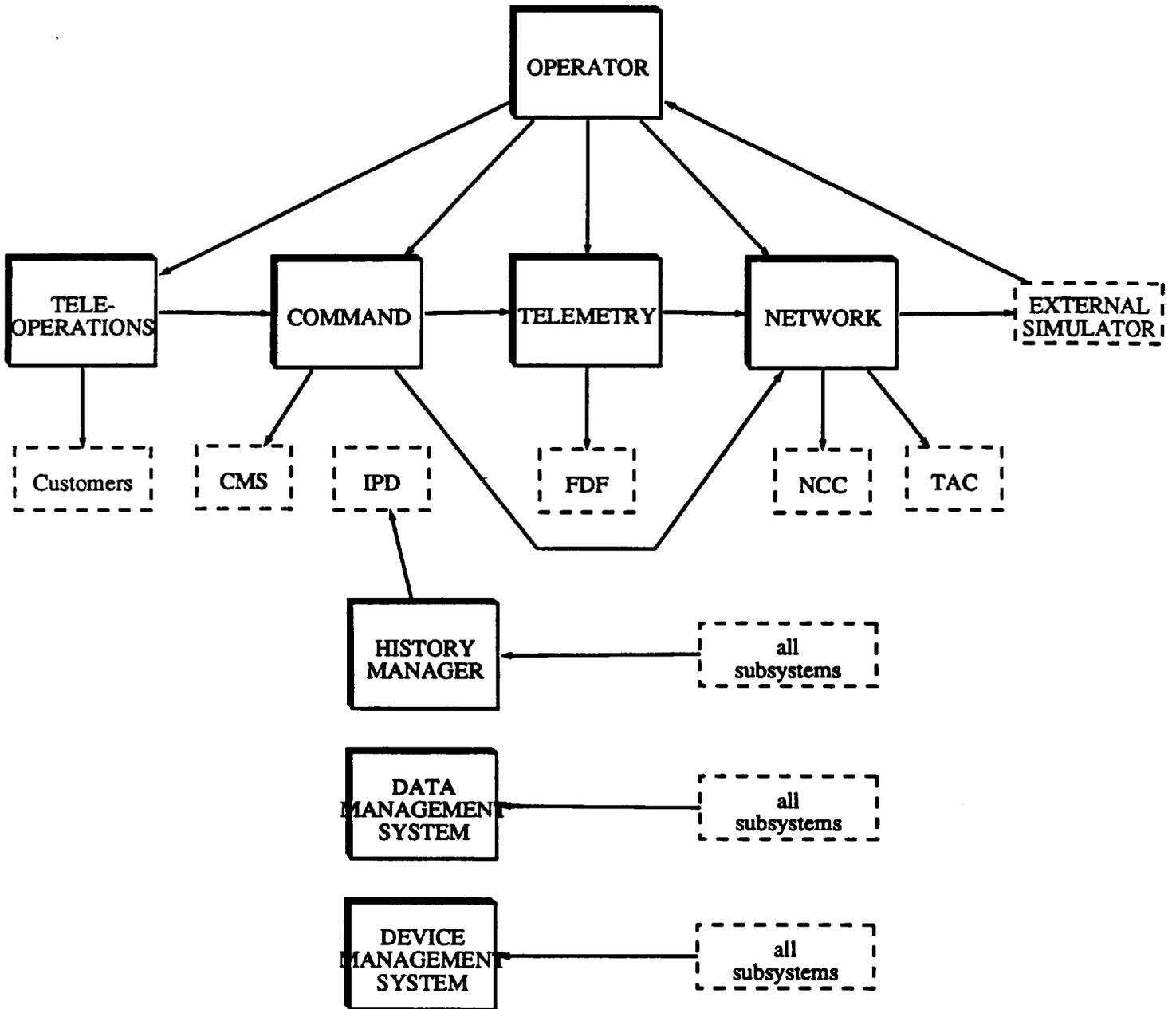
**POCC AP - GENERIC ARCHITECTURE**

**Entity-Interface Diagrams**

ORIGINAL PAGE IS  
OF POOR QUALITY

POCC AP--GENERIC ARCHITECTURE LEVEL 0

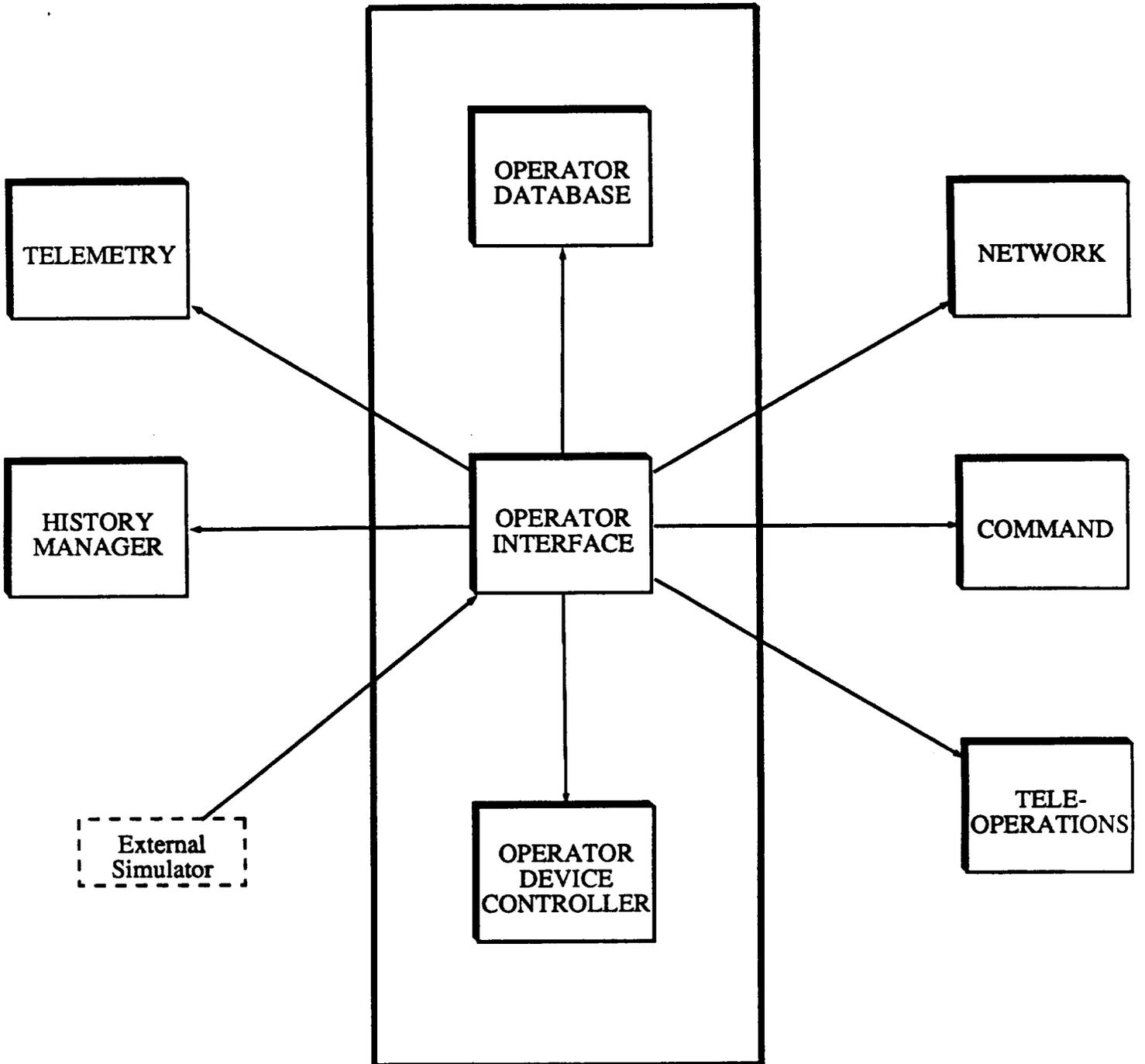
Entity-Interface Diagram



POCC AP-GENERIC ARCHITECTURE LEVEL 1

Entity-Interface Diagram

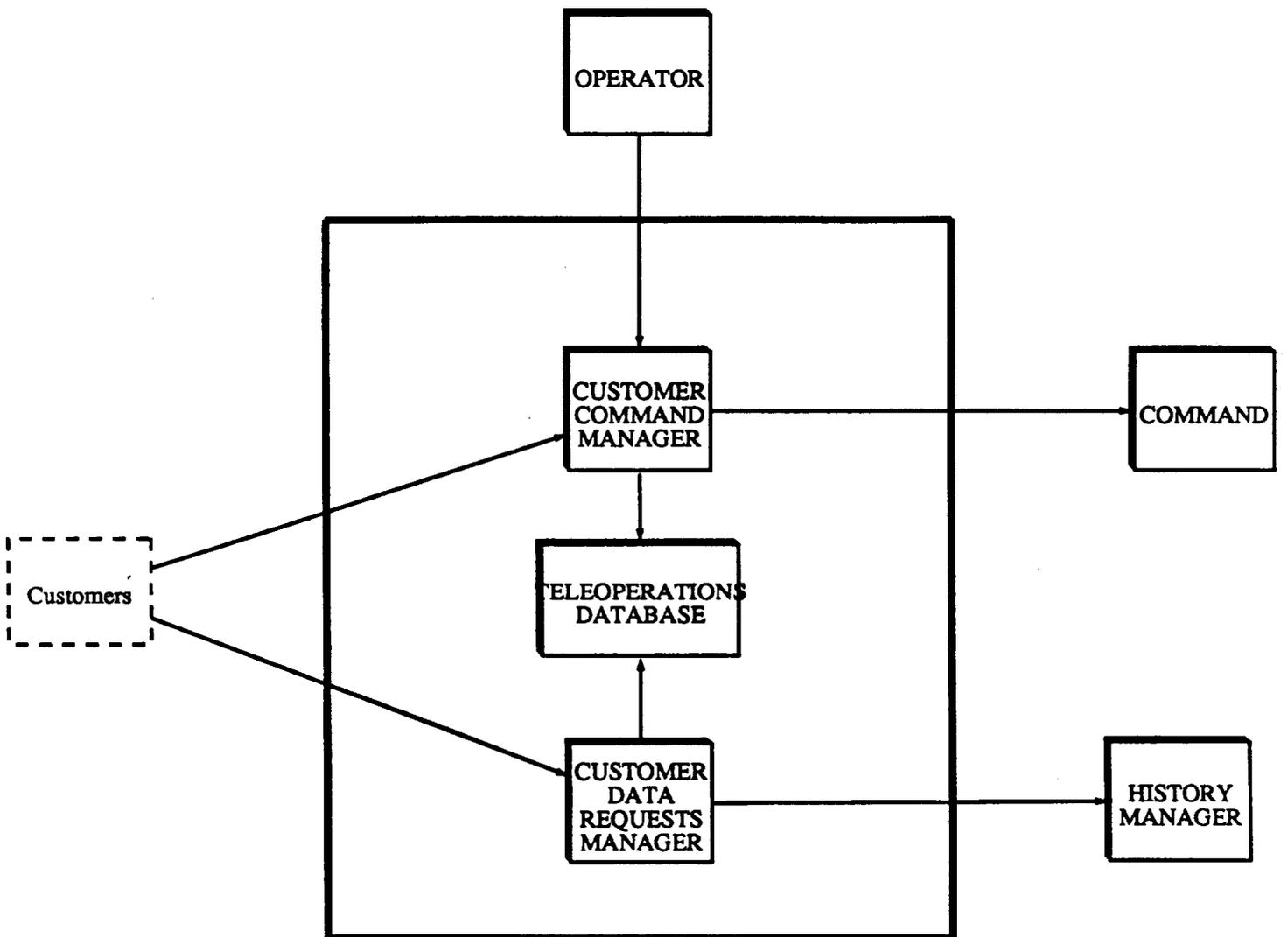
OPERATOR



POCC AP--GENERIC ARCHITECTURE LEVEL 1

Entity-Interface Diagram

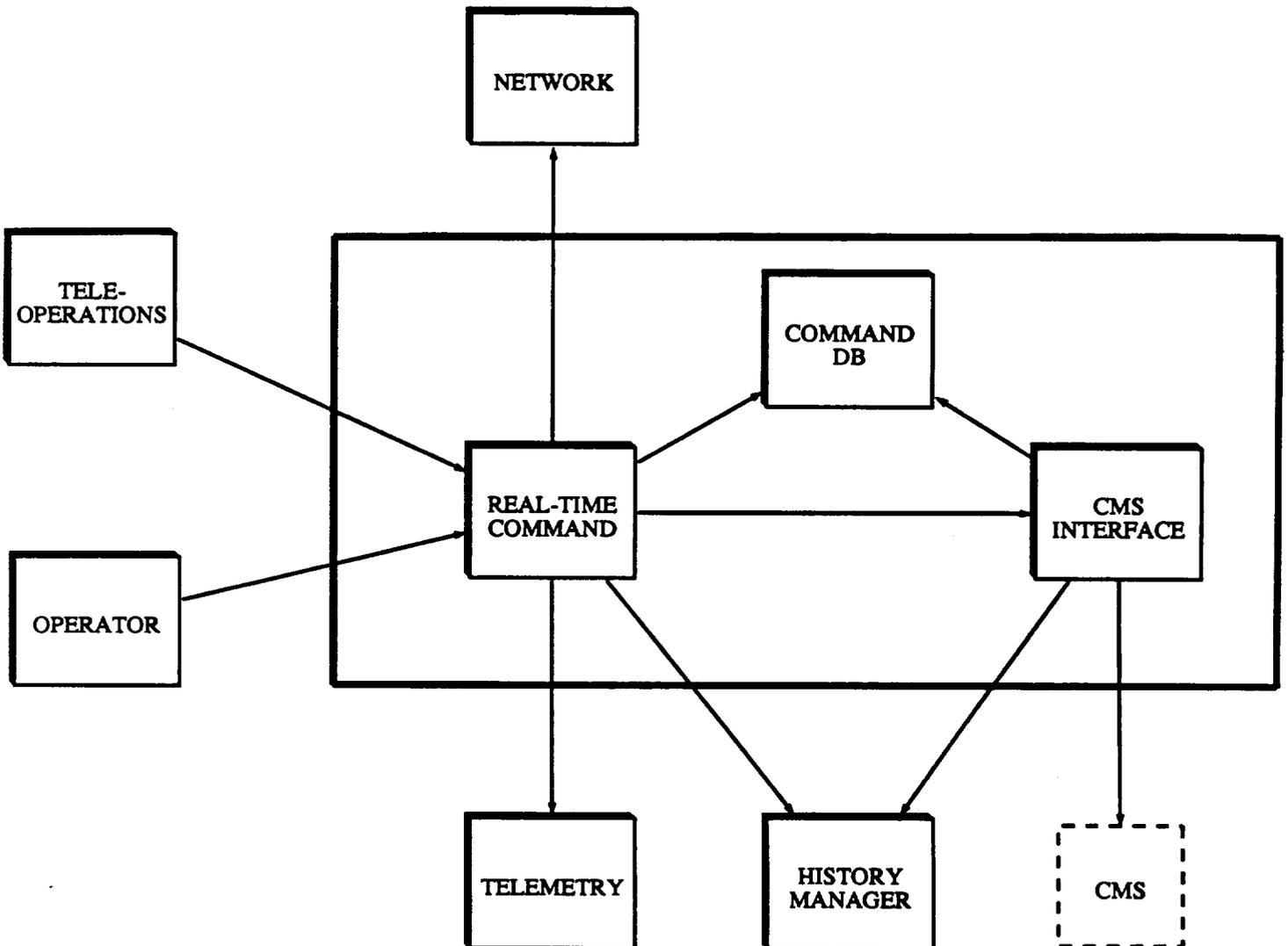
TELEOPERATIONS



POCC AP-GENERIC ARCHITECTURE LEVEL 1

Entity-Interface Diagram

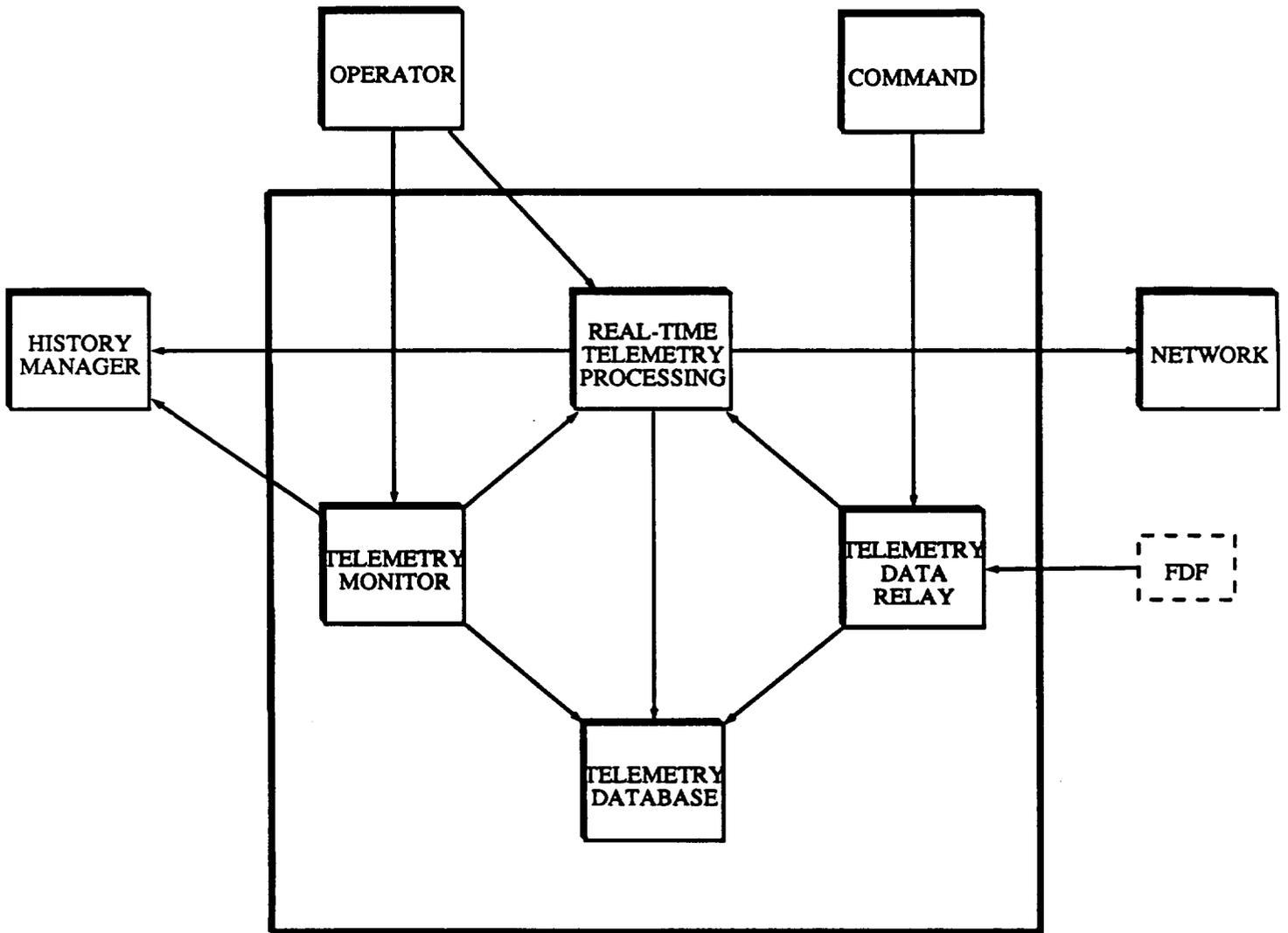
COMMAND



POCC AP--GENERIC ARCHITECTURE LEVEL 1

Entity-Interface Diagram

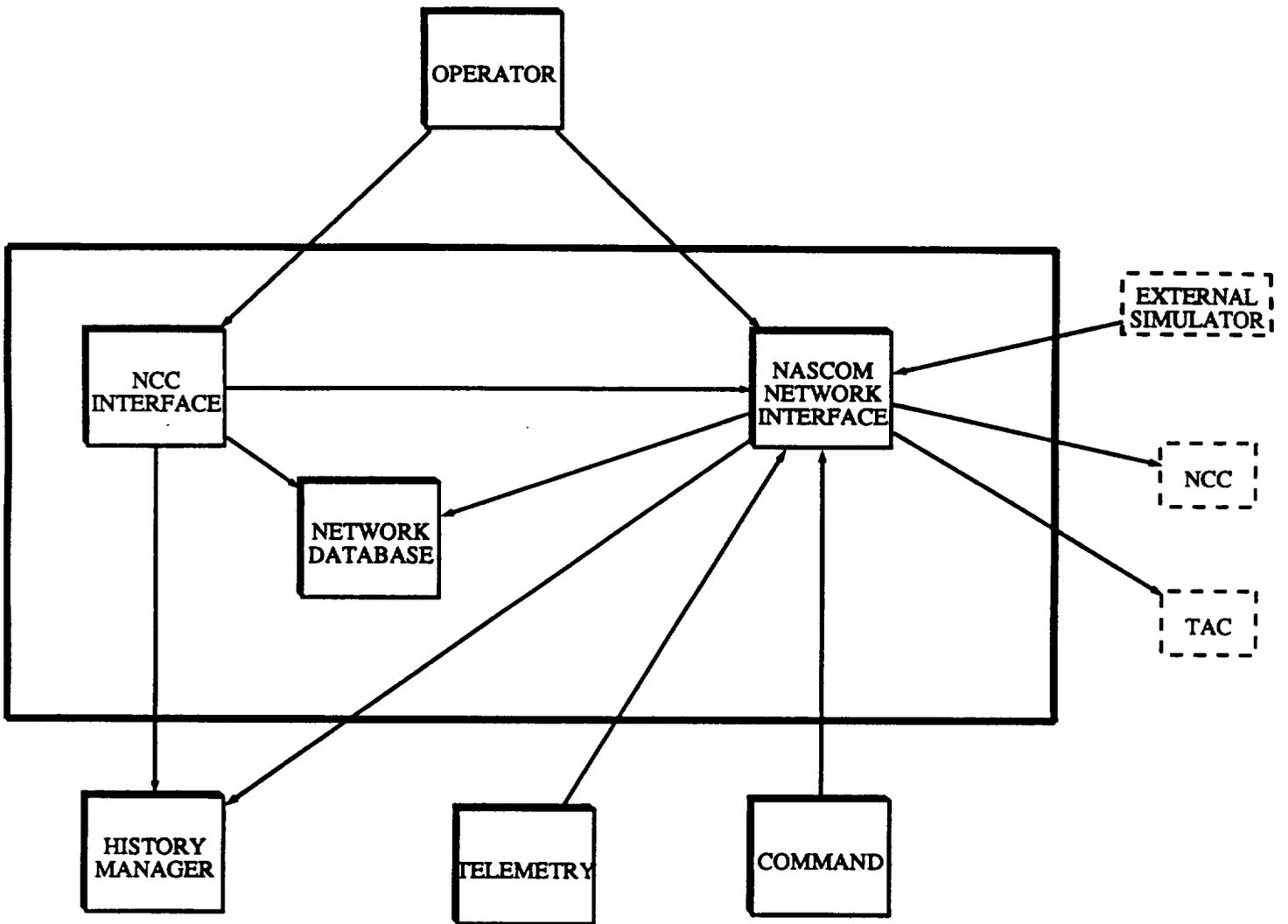
TELEMETRY



POCC AP--GENERIC ARCHITECTURE LEVEL 1

Entity-Interface Diagram

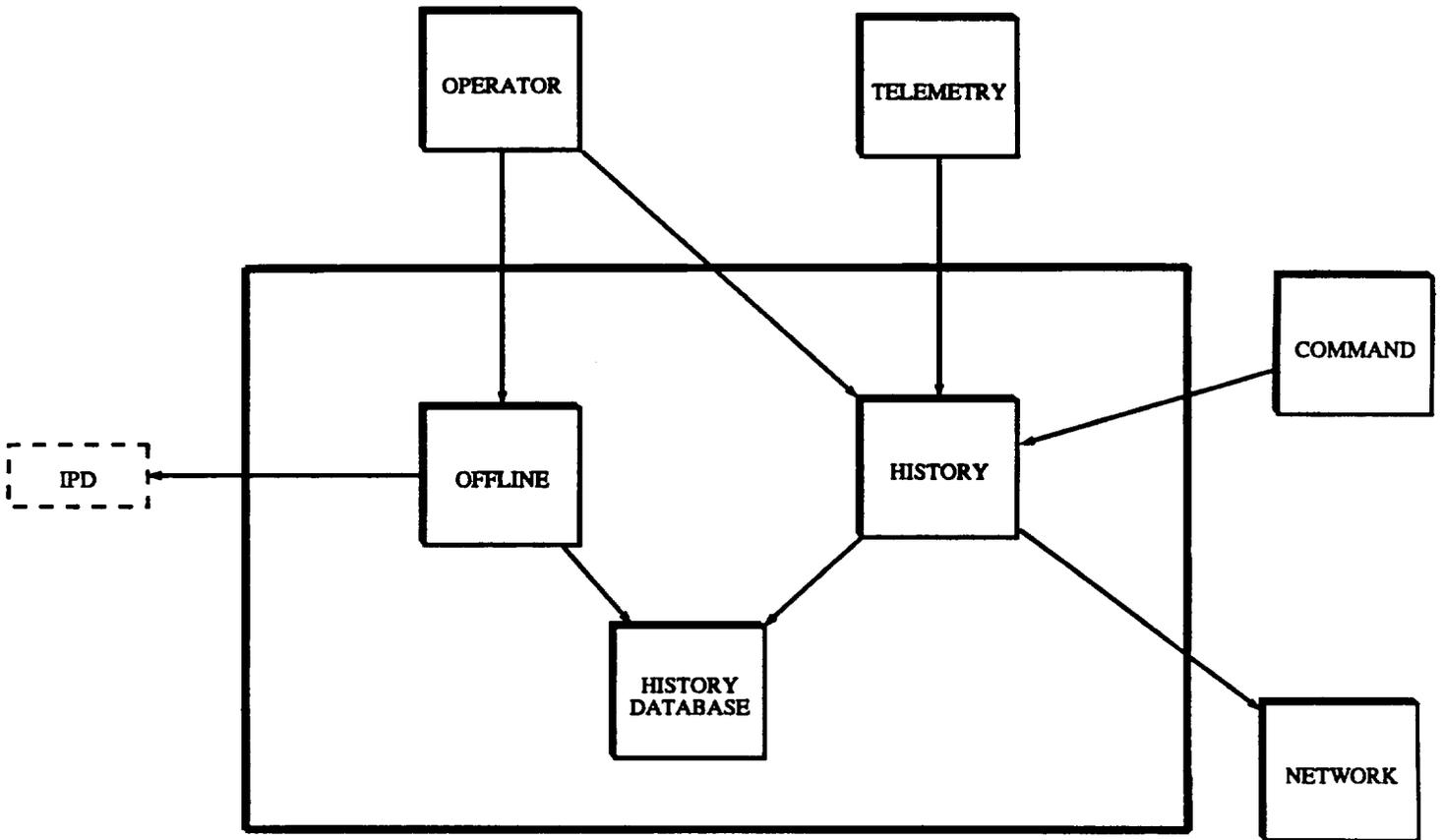
NETWORK



POCC AP-GENERIC ARCHITECTURE LEVEL 1

Entity\_Interface Diagram

HISTORY MANAGER



May 12, 1989

## GENERIC ARCHITECTURE 1--SUBSYSTEM FUNCTIONS

### OPERATOR

#### Performs:

1. Parse and edit operator input
2. Verify User Access
3. Copy Operator Input to History
4. Generate Operator Directives
5. Process Internal Directives
6. Route Directives to Subsystems and External Simulator
7. Manage Event Messages
8. Copy Event Messages to History
9. Provide Expert Guidance
10. Monitor and Control Teleoperations
11. Create NCC Requests and Responses
12. Format Display Pages (Telemetry, SNAP, Special, etc.)
13. Display Event Messages, Requested Reports and System Status
14. Initialize Interface
15. Accept/Reject Critical Commands
16. Resolve Customer Scheduling Conflicts
17. Control and Monitor External Simulator

#### Depends On:

##### All Subsystems

- Initialize Subsystem
- Interpret Operator Directives

##### Command

- Generate OBC Loads and Commands
- Notify Operator of Critical Commands
- Process Operator Command Acceptance/Rejection
- Notify User of Command Acceptance/Rejection
- Notify User of Command Status

##### Telemetry

- Collect Interval Data
- Provide Quick Look Experiments Analysis
- Check Analog Limits
- Monitor Spacecraft Configuration
- Provide Simulation
- Issue Status Reports to Operator

##### Network

- Route outgoing blocks to TAC or External Simulator
- Generate NCC status requests/responses
- Inform Operator of NCC requests
- Process Operator status requests

##### Teleoperations

- Report command and request problems to operator and customers
- Report command and request schedules to operator

##### History Manager

- Perform Long-term trend analysis
- Summarize Out of Limit Data
- Process Customer and Operator Data Requests

- Record History data for all subsystems

## **COMMAND**

### **Performs:**

1. Initialize Subsystem
2. Interpret Operator Directives
3. Receive OBC Loads and Commands from CMS
5. Send OBC image data to Network
6. Receive Command Groups and OBC Loads from Network
7. Receive Images and Verification Data from Telemetry
8. Generate OBC Loads and Commands
9. Notify Operator of Critical Commands
10. Process Operator Command Acceptance/Rejection
11. Notify User of Command Acceptance/Rejection
12. Verify OBC Loads and Commands
13. Notify User of Command Status
14. Transmit OBC Loads and Commands to S/C via Network
15. Copy Commands to History
16. Receive S/C Command Generation Requests from Teleoperations

### **Depends On:**

#### Telemetry

- Process telemetry blocks and route telemetry information to appropriate subsystems

#### Network

- Route outgoing blocks to TAC, NCC or External Simulator
- Routes Commands and OBC Loads to Command
- Receives OBC Images from Command

#### History Manager

- Record History Data from all Subsystems

## **TELEMETRY**

### **Performs:**

1. Interpret and Process Operator Directives
2. Initialize Subsystem
3. Process telemetry blocks and route telemetry information to appropriate subsystems
4. Collect Interval Data
5. Provide Quick Look Experiments Analysis
6. Provide Special Algorithms (Equation Processors)
7. Check Analog Limits
8. Monitor Spacecraft Configuration
9. Provide Simulation
10. Issue Status Reports to Operator
11. Copy Telemetry Blocks and Interval Data to History
12. Send Attitude Data to FDF

### **Depends On:**

#### Network

- Route Incoming Blocks to Telemetry

#### History Manager

- Record History Data from all Subsystems

## **HISTORY MANAGER**

### **Performs:**

1. Initialize Subsystem
2. Interpret Operator Directives
3. Perform S/C clock error calculation
4. Perform Long-term Trend Analysis
5. Summarize Out of Limit Data
6. Convert IPD History Tape to POCC AP Format
7. Download Interval Archival Data to Display (IDT)
8. Record History Data from all subsystems
9. Generate Printer Listings for Event Messages, Commands, NASCOM Blocks, Tape Directory
10. Replay NASCOM Blocks to Telemetry via Network Interface
11. Process Customer and Operator Data Requests

### **Depends On:**

- Network
  - Routes incoming Blocks to Telemetry
- IPD
  - IPD Data

## **NETWORK**

### **Performs:**

1. Initialize Subsystem
2. Interpret and Process Operator Directives (Configuration directives and Status requests)
3. Format outgoing blocks
4. Route outgoing blocks to TAC, NCC or External Simulator
5. Strip incoming block headers
6. Route incoming blocks to Telemetry
7. Copy NASCOM Blocks to History
8. Parse Incoming NCC Blocks
9. Acknowledge NCC Block receipt
10. Generate NCC status requests/responses
11. Monitor NCC responses to status requests
12. Inform Operator of NCC requests
13. Process Operator status requests
14. Copy NCC Messages to History
15. Receives Commands and OBC Loads from CMS
16. Routes Commands and OBC Loads to Command
17. Receives OBC Images from Command
18. Routes OBC Images to CMS
19. Receives Attitude data from Telemetry
20. Routes Attitude data to FDF

### **Depends On:**

- TAC
  - Data transmission and receipt

NCC

- Data monitoring and status reports

CMS

- OBC and Command Data

## **TELEOPERATIONS**

### **Performs:**

1. Parse and edit customer commands and data requests
2. Verify customer access
3. Detect command and request conflicts
4. Schedule customer commands and requests
5. Report command and request problems to operator and customers
6. Assist in problem resolution
7. Report command and request schedules to operator
8. Report command acceptance/rejection to customers
9. Send commands to Command subsystem
10. Send data requests to History subsystem

### **Depends On:**

Command

- Generate OBC Loads and Commands
- Notify User of Command Acceptance/Rejection
- Notify User of Command Status

History

- Process Customer and Operator Data Requests
- Record History Data from all subsystems

Operator

- Resolve Customer Scheduling Conflicts

## **DATA MANAGEMENT SYSTEM**

### **Performs:**

1. Provide DBMS services to subsystems
2. Provide Offline AP Data Maintenance Services

## **DEVICE MANAGEMENT SYSTEM**

### **Performs:**

1. Resolve logical device calls into real device calls
2. Resolve real device results into logical device results

## **EXTERNAL SIMULATOR**

### **Performs:**

1. Initialize Subsystem
2. Interpret NASCOM Blocks
3. Interpret Directives
4. Execute S/C Commands
5. Generate Telemetry

6. Generate NCC Messages and Acknowledgements
7. Send Telemetry and NCC Messages/Acknowledgements to Network Interface

**Depends On:**

**Network**

- Route incoming blocks to Telemetry
- Route outgoing blocks to TAC or External Simulator

**Operator**

- Control and Monitor External Simulator

## **GENERIC ARCHITECTURE 1 - SUBENTITY FUNCTIONS**

### **OPERATOR**

#### **Operator Interface**

**Performs:**

1. Parse and edit operator input
2. Verify User Access
3. Copy Operator Input to History
4. Generate Operator Directives
5. Process Internal Directives
6. Route Directives to Subsystems and External Simulator
7. Manage Event Messages
8. Copy Event Messages to History
9. Provide Expert Guidance
10. Monitor and Control Teleoperations
11. Create NCC Requests and Responses
12. Format Display Pages (Telemetry, SNAP, Special, etc.)
13. Display Event Messages, Requested Reports and System Status
14. Accept/Reject Critical Commands
15. Resolve Customer Scheduling Conflicts

**Depends On:**

**All Subsystems**

- Initialize Subsystem
- Interpret Operator Directives

**Command**

- Generate OBC Loads and Commands
- Notify Operator of Critical Commands
- Process Operator Command Acceptance/Rejection
- Notify User of Command Acceptance/Rejection
- Notify User of Command Status

**Telemetry**

- Collect Interval Data
- Provide Quick Look Experiments Analysis
- Check Analog Limits
- Monitor Spacecraft Configuration
- Provide Simulation
- Issue Status Reports to Operator

**Network**

- Route outgoing blocks to TAC or External Simulator

- Generate NCC status requests/responses
- Inform Operator of NCC requests
- Process Operator status requests

#### Teleoperations

- Report command and request problems to operator and customers
- Report command and request schedules to operator

#### History Manager

- Perform Long-term trend analysis
- Summarize Out of Limit Data
- Process Customer and Operator Data Requests
- Record History data for all subsystems

#### Operator Device Controller

- Perform AP\_to\_Operator Output Requests
- Send Resolved Input to Operator Interface

#### Operator Database

- Manages User Access Rights
- Defines Format and Availability of Operator Directives
- Maintain Operator Parameter Selections

### Operator Device Controller

#### Performs:

1. Resolve Logical AP\_to\_Operator Output Requests
2. Perform AP\_to\_Operator Output Requests
3. Retrieve Operator Input
4. Resolve Logical Operator\_to\_AP Input
5. Send Resolved Input to Operator Interface

### Operator Database

#### Performs:

1. Manages User Access Rights
2. Defines Format and Availability of Operator Directives
3. Maintain Operator Parameter Selections

## COMMAND

### Real-time Command

#### Performs:

1. Initializes Command Processing
2. Processes Command Directives
3. Receive Images and Verification Data from Telemetry
4. Generate OBC Loads and Commands
5. Notify Operator of Critical Commands
6. Process Operator Command Acceptance/Rejection
7. Notify User of Command Acceptance/Rejection
8. Verify OBC Loads and Commands
9. Notify User of Command Status
10. Transmit OBC Loads and Commands to S/C via Network
11. Send OBC Images to CMS Interface
12. Copy Commands to History
13. Receive S/C Command Generation Requests from Teleoperations

**Depends On:****Telemetry**

- Process telemetry blocks and route telemetry information to appropriate subsystems

**Network**

- Route outgoing blocks to TAC or External Simulator

**History Manager**

- Record History Data from all Subsystems

**Command Database**

- Manages OBC Image Data
- Manages User/Command Reference Table
- Provides Reference Data

**CMS Interface**

- Send OBC Loads and Commands to Command
- Receive OBC image data from Command

**Command Database****Performs:**

1. Manages OBC Image Data
2. Manages User/Command Reference Table
3. Provides Reference Data

**CMS Interface****Performs:**

1. Receive OBC Loads and Commands from Network
2. Send OBC Loads and Commands to Command
3. Receive OBC image data from Command
4. Send OBC image data to Network

**Depends On:****CMS**

- OBC and Command Data

**History Manager**

- Records data for all subsystems

**Command Database**

- Provides Reference Data

**Network**

- Routes Commands and OBC Loads to Command
- Receives OBC Images from Command

**TELEMETRY****Telemetry Monitor****Performs:**

1. Initialize Simulation and Data Collection
2. Process Operator Parameter Settings
3. Collect Interval Data
4. Perform Short-term Interval Analysis
5. Monitor Spacecraft Configuration

6. Issue Status Reports to Operator
7. Provide Simulation
8. Copy Interval Data to History

**Depends On:**

History Manager

- Record History Data from all Subsystems

Real-time Telemetry Processing

- Extracts Telemetry Information from Telemetry Blocks

Telemetry Database

- Maintains Operator Parameter settings
- Maintains Interval Data
- Provides reference data

**Real-time Telemetry Processing**

**Performs:**

1. Initialize Data Tables
2. Processes Operator Parameter Settings
3. Extracts Telemetry Information from Telemetry Blocks
4. Extracts Command Verification Data and Routes to Telemetry Data Relay
5. Copies Telemetry Blocks to History Manager
6. Provide Quick Look Experiments Analysis
7. Provide Special Algorithms (Equation Processors)
8. Check Analog Limits
9. Issue Status Reports to Operator
10. Send Attitude Data to FDF

**Depends On:**

Network

- Route Incoming Blocks to Telemetry

History Manager

- Record History Data from all Subsystems

Telemetry Database

- Maintains Operator Parameter settings
- Maintains Experiment data
- Provides reference data

**Telemetry Data Relay**

**Performs:**

1. Route Attitude Data to FDF
2. Sends Command Verification OBC Image Data to Command

**Depends On:**

Real-time Telemetry Processing

- Extracts Command Verification Data and Routes to Telemetry Data Relay

Telemetry Database

- Provides reference data

**Telemetry Database**

**Performs:**

1. Maintains Operator Parameter settings
2. Maintains Interval Data
3. Maintains Experiment data
4. Provides reference data

## **HISTORY MANAGER**

### **History Database**

#### **Performs:**

1. Maintains History Data
2. Retrieves desired History Data

### **History**

#### **Performs:**

1. Record History Data from all subsystems
2. Generate Printer Listings for Event Messages, Commands, NASCOM Blocks, Tape Directory
3. Provide Telemetry and Interval Data for Offline Processing
4. Replay NASCOM Blocks to Telemetry via Network Interface
5. Perform Customer and Operator Data Requests

#### **Depends On:**

##### **Network**

- Routes incoming Blocks to Telemetry

##### **History Database**

- Maintains History Data
- Retrieves desired History Data

### **Offline**

#### **Performs:**

1. Perform S/C clock error calculation
2. Perform Long-term Trend Analysis
3. Summarize Out of Limit Data
4. Convert IPD History Tape to POCC AP Format
5. Provide Interval Archival Data to Operator
6. Generates Operator requested reports

#### **Depends On:**

##### **History Database**

- Maintains History Data
- Retrieves desired History Data

##### **IPD**

- IPD Data

## **NETWORK**

### **NCC Interface**

#### **Performs:**

1. Process Operator status requests
2. Parse Incoming NCC Blocks
3. Acknowledge NCC Block receipt
4. Generate NCC status requests/responses
5. Monitor NCC responses to status requests
6. Inform Operator of NCC requests/responses
7. Copy NCC Messages to History
8. Initialize NCC Interface

**Depends On:**

- NASCOM Network Interface
  - Route incoming NCC blocks to NCC Interface
- Network Database
  - Provides Network reference data
  - Maintains AP-NCC requests and status
- History Manager
  - Record History Data from all subsystems

**NASCOM Network Interface**

**Performs:**

1. Process configuration directives
2. Format outgoing blocks
3. Route outgoing blocks to TAC, NCC or External Simulator
4. Strip incoming block headers
5. Route incoming Telemetry blocks to Telemetry
6. Copy NASCOM Blocks to History
7. Initialize Network Interface
8. Route incoming NCC blocks to NCC Interface

**Depends On:**

- TAC
  - Data transmission and receipt
- NCC
  - Data monitoring and status reports
- Network Database
  - Provides Network reference data
- History Manager
  - Record History Data from all subsystems

**MODLAN Network Interface**

**Performs:**

1. Receives Commands and OBC Loads from CMS
2. Routes Commands and OBC Loads to Command
3. Receives OBC Images from Command
4. Routes OBC Images to CMS
5. Receives Attitude data from Telemetry
6. Routes Attitude data to FDF

**Depends On:**

- Network Database

- Provides Network reference data
- History Manager**
- Record History Data from all subsystems

### **Network Database**

1. Provides Network reference data
2. Maintains AP-NCC requests and status

## **TELEOPERATIONS**

### **Customer Command Manager**

#### **Performs:**

1. Parse and edit customer commands
2. Verify customer access
3. Detect command conflicts
4. Schedule customer commands
5. Report command problems to operator and customers
6. Assist in problem resolution
7. Report command schedules to operator
8. Report command acceptance/rejection to customers
9. Send commands to Command subsystem

#### **Depends On:**

##### **Command**

- Generate OBC Loads and Commands
- Notify User of Command Acceptance/Rejection
- Notify User of Command Status

##### **Teleoperations Database**

- Maintains access rights
- Maintains command schedules
- Maintains command scheduling and conflict constraints

### **Customer Requests Data Manager**

#### **Performs:**

1. Parse and edit customer data requests
2. Verify customer access
3. Detect request conflicts
4. Schedule customer requests
5. Report request problems to operator and customers
6. Assist in problem resolution
7. Report request schedules to operator
10. Send data requests to History subsystem

#### **Depends On:**

##### **History Manager**

- Process Customer and Operator Data Requests
- Record History Data from all subsystems

##### **Teleoperations Database**

- Maintains access rights
- Maintains conflict constraints

## **Teleoperations Database**

### **Performs:**

- 1. Maintains access rights**
- 2. Maintains request and command schedules**
- 3. Maintains command scheduling and conflict constraints**

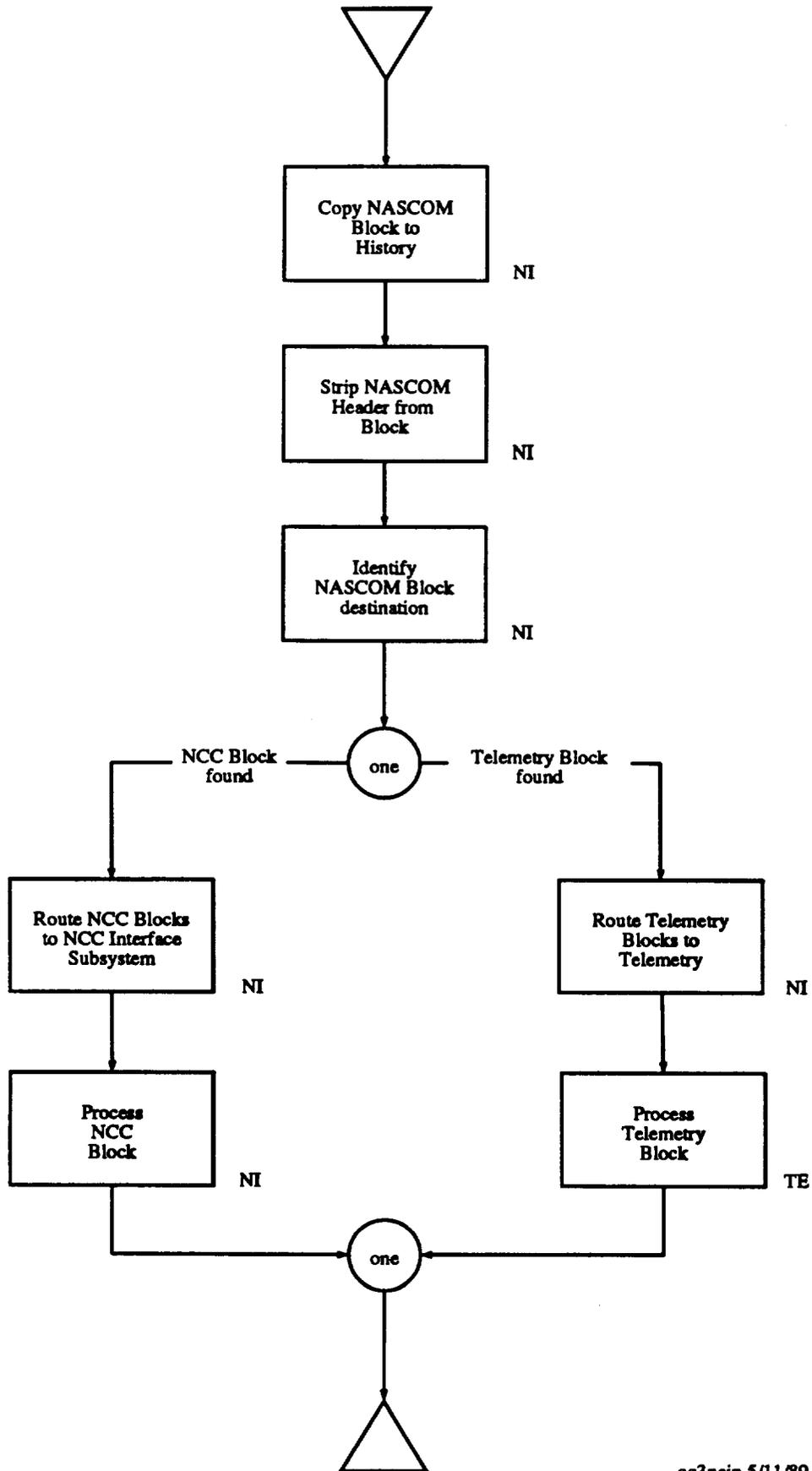
## POCC AP - GENERIC ARCHITECTURE

### Composition Graphs

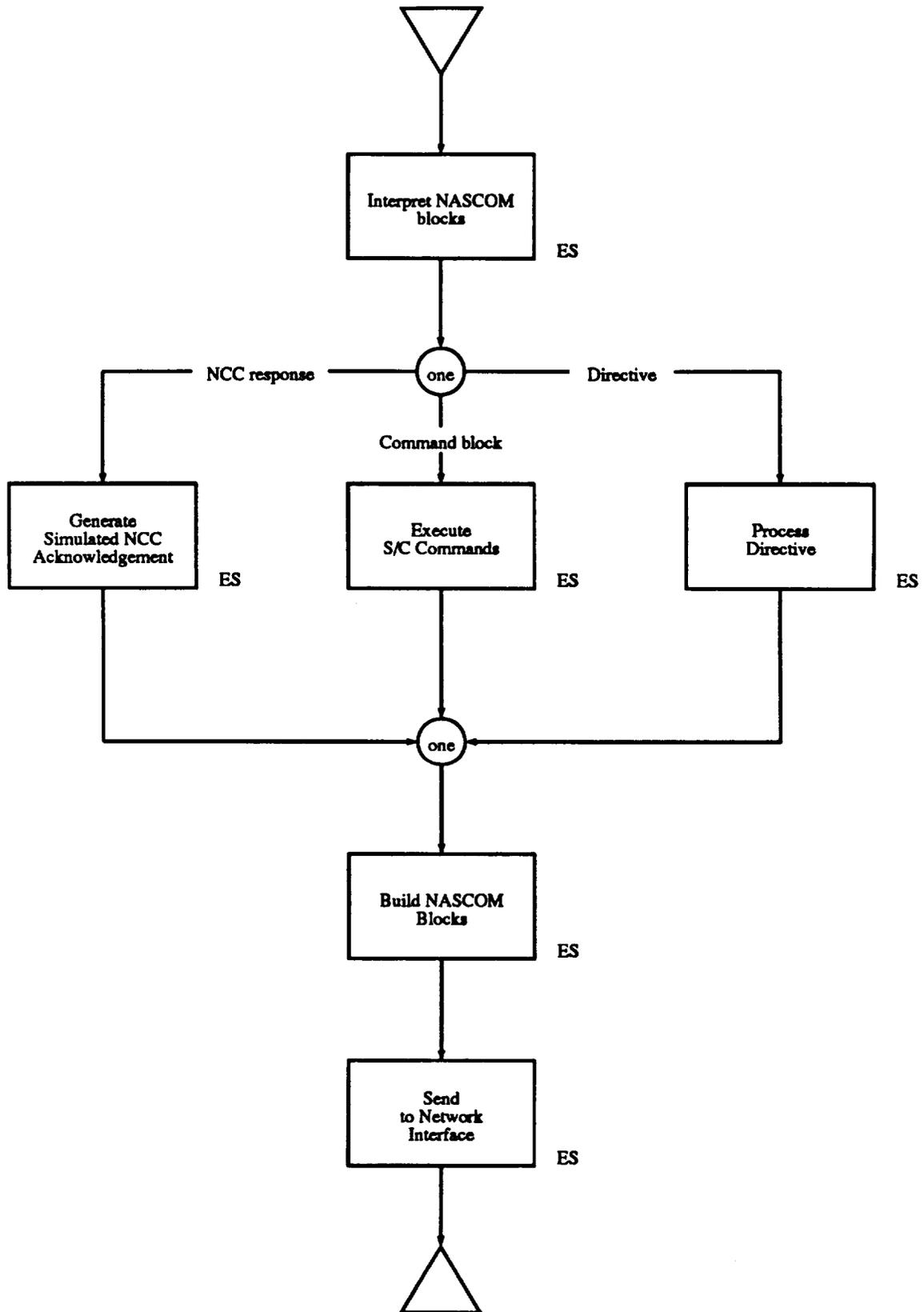
#### Legend

CO	=	Command
OP	=	Operator
HI	=	History Manager
TE	=	Telemetry
TO	=	Teleoperations
NE	=	Network
ES	=	External Simulator
DM	=	Data Management
DV	=	Device Management

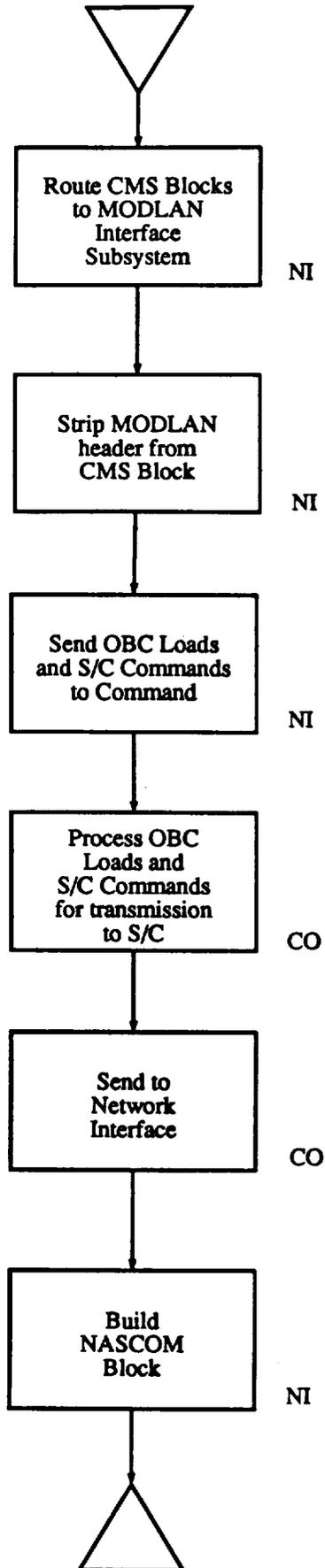
Process Incoming NASCOM Block



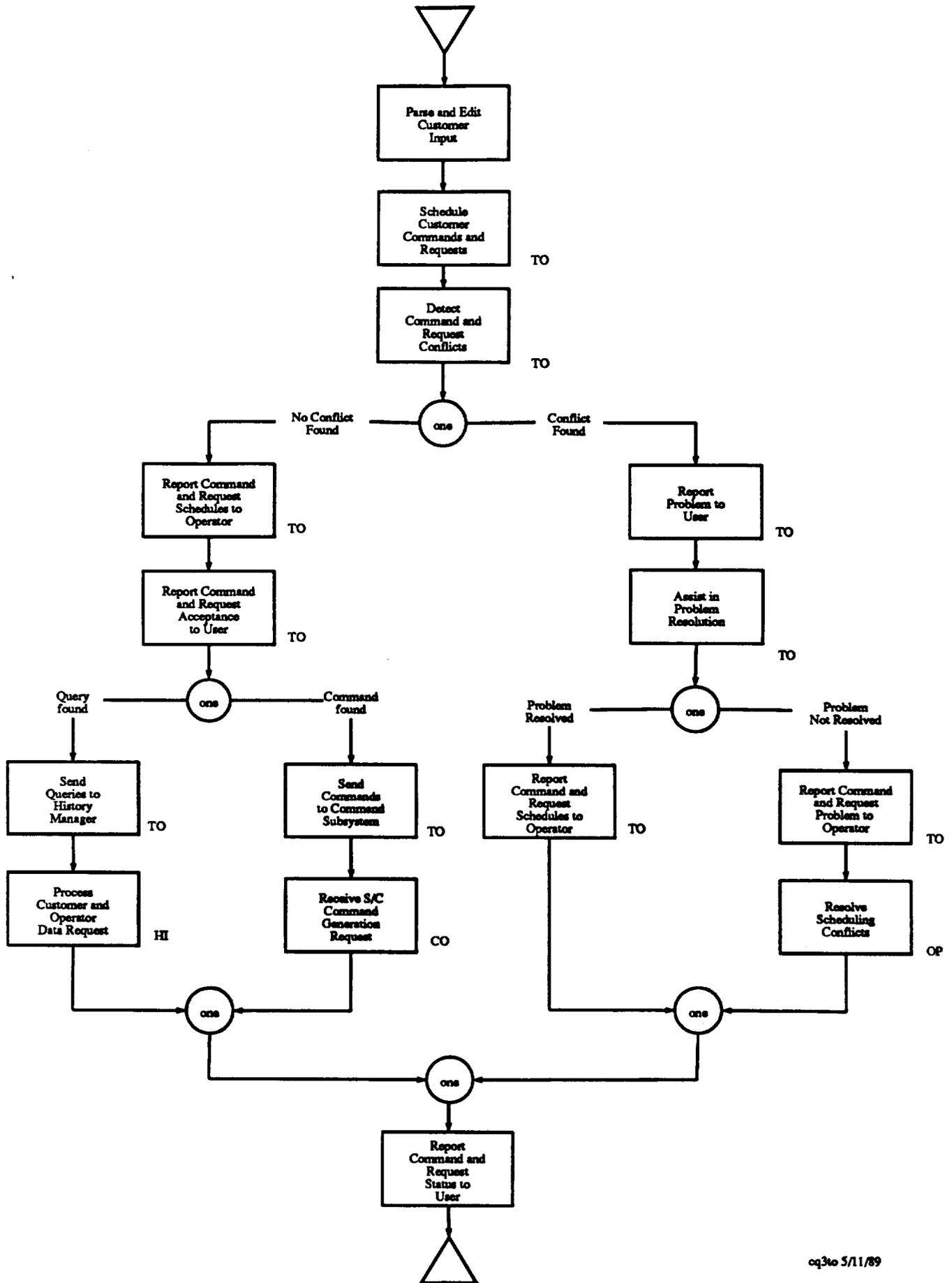
External Simulator Driven by NASCOM Blocks



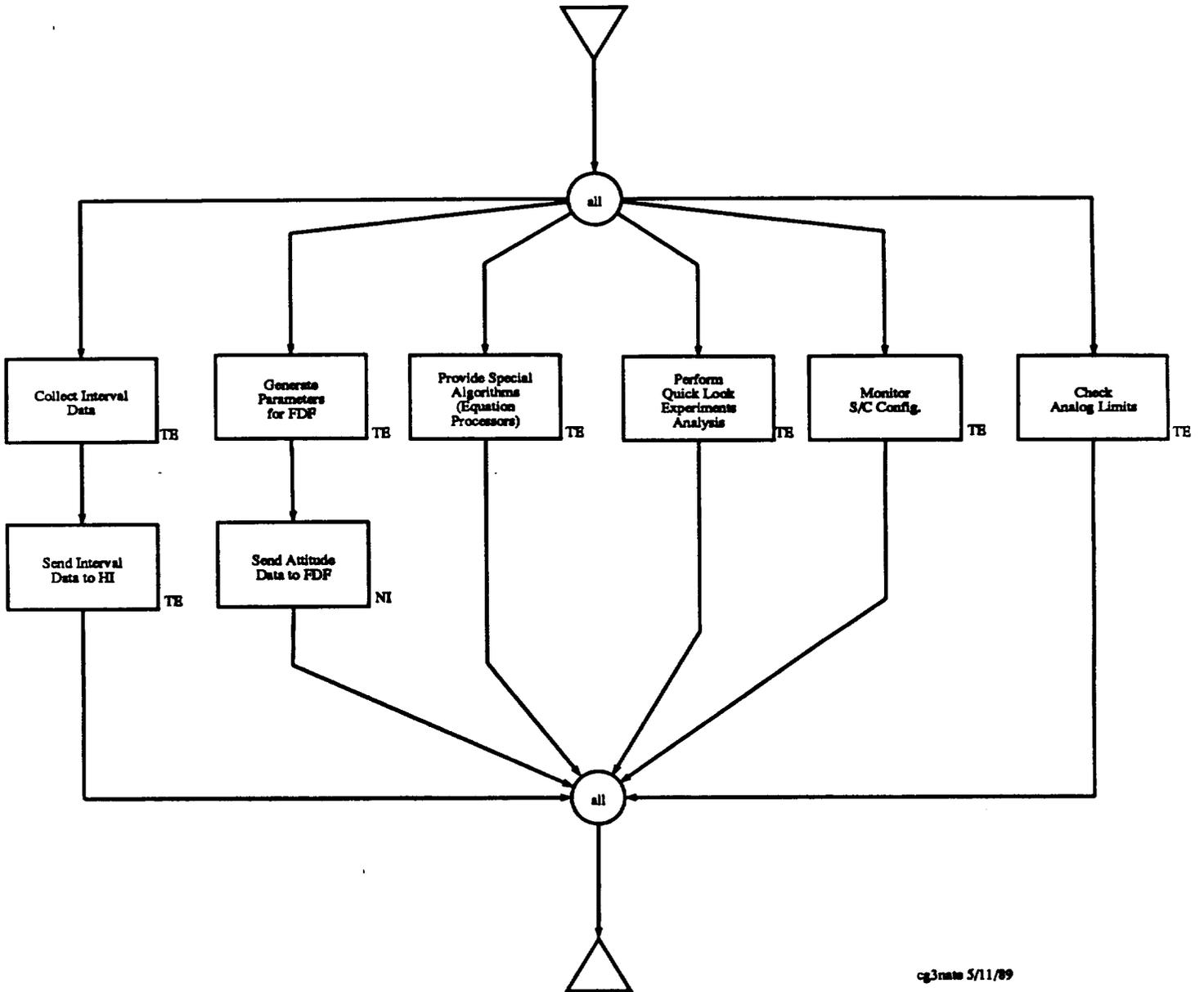
Process CMS Blocks (OBC Loads and S/C Commands)



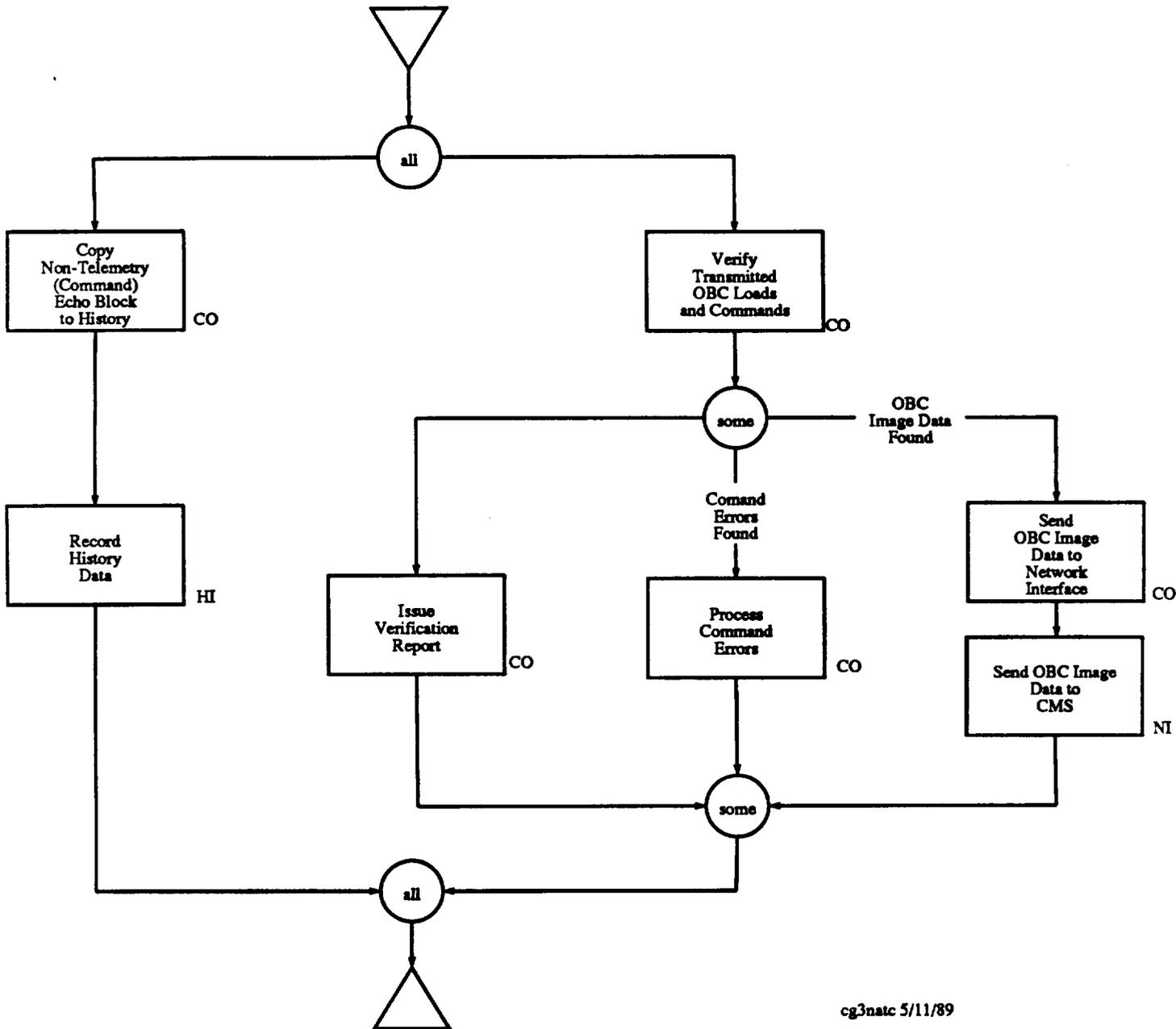
Process Teleoperations Queries and Command Requests



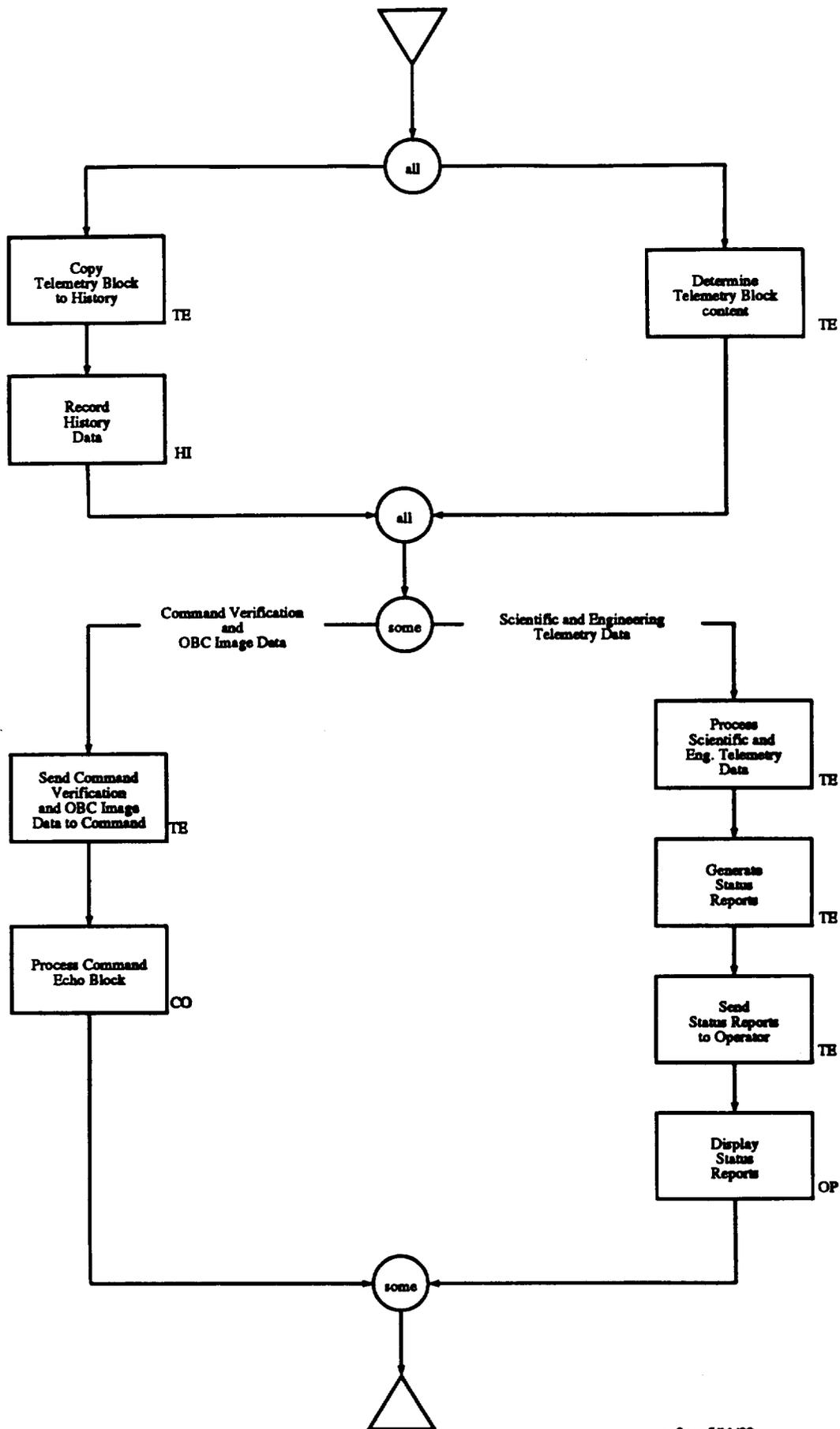
Process Scientific and Engineering Telemetry Data



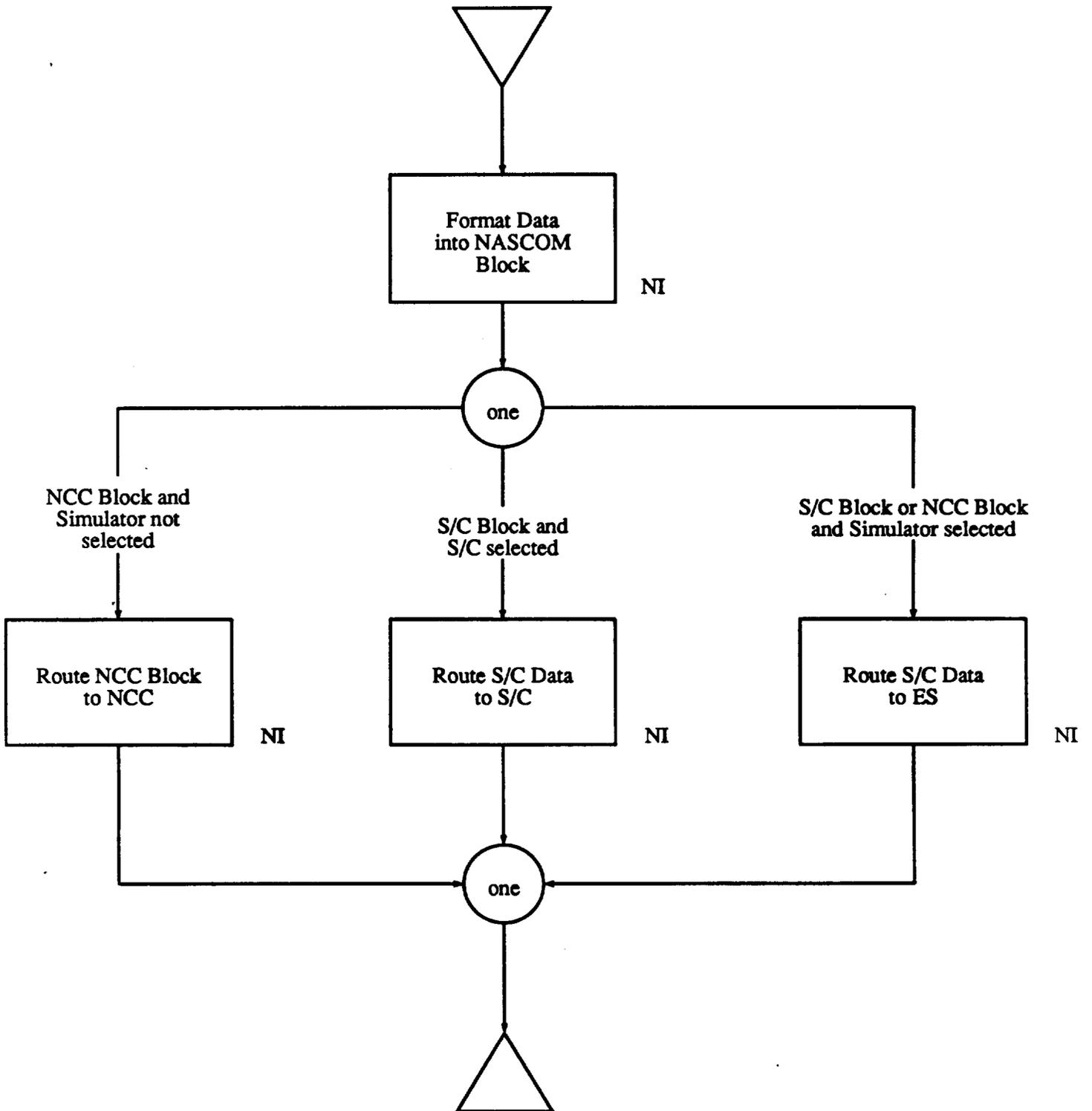
Process Command Echo Block



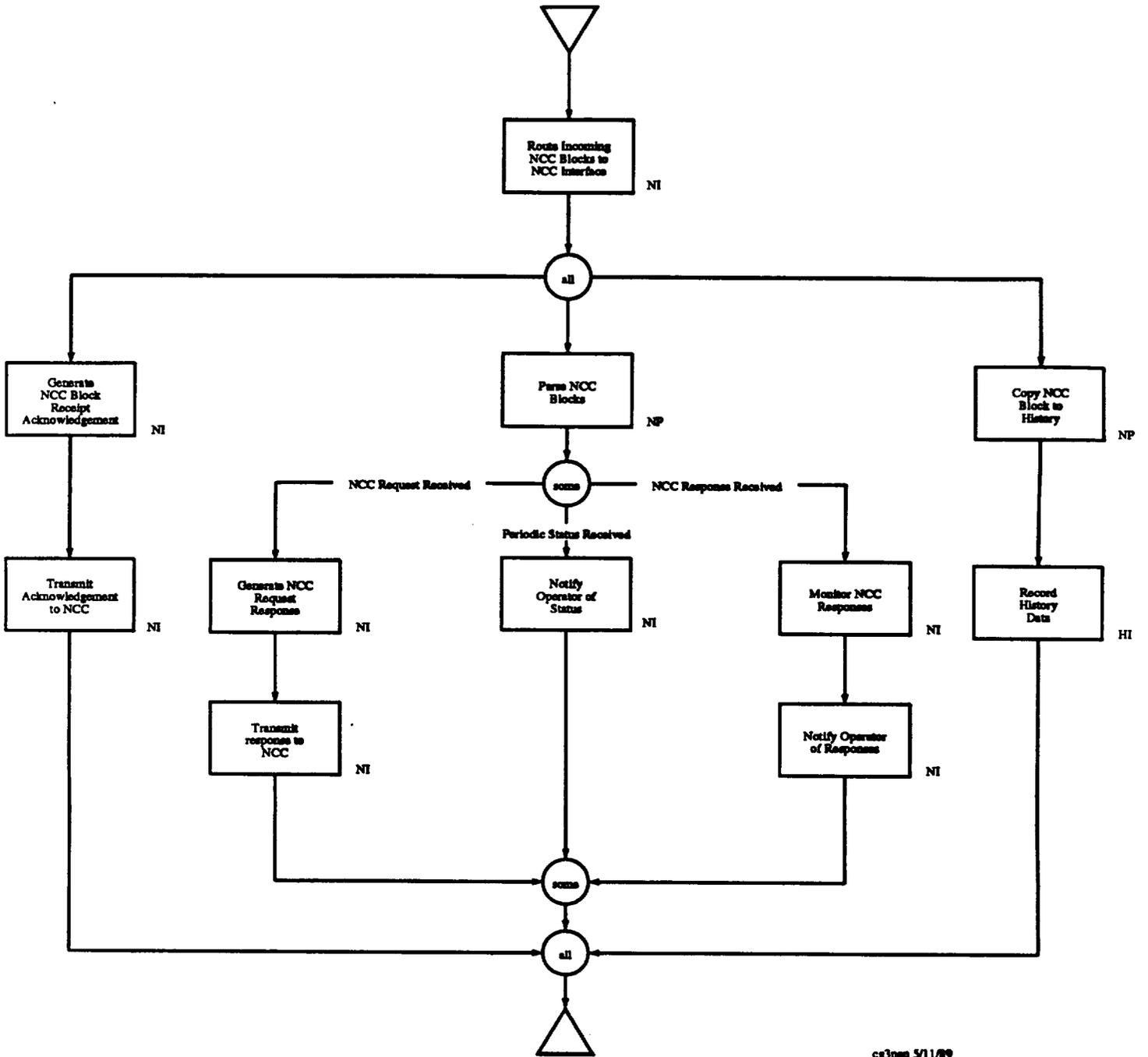
Process Telemetry Block



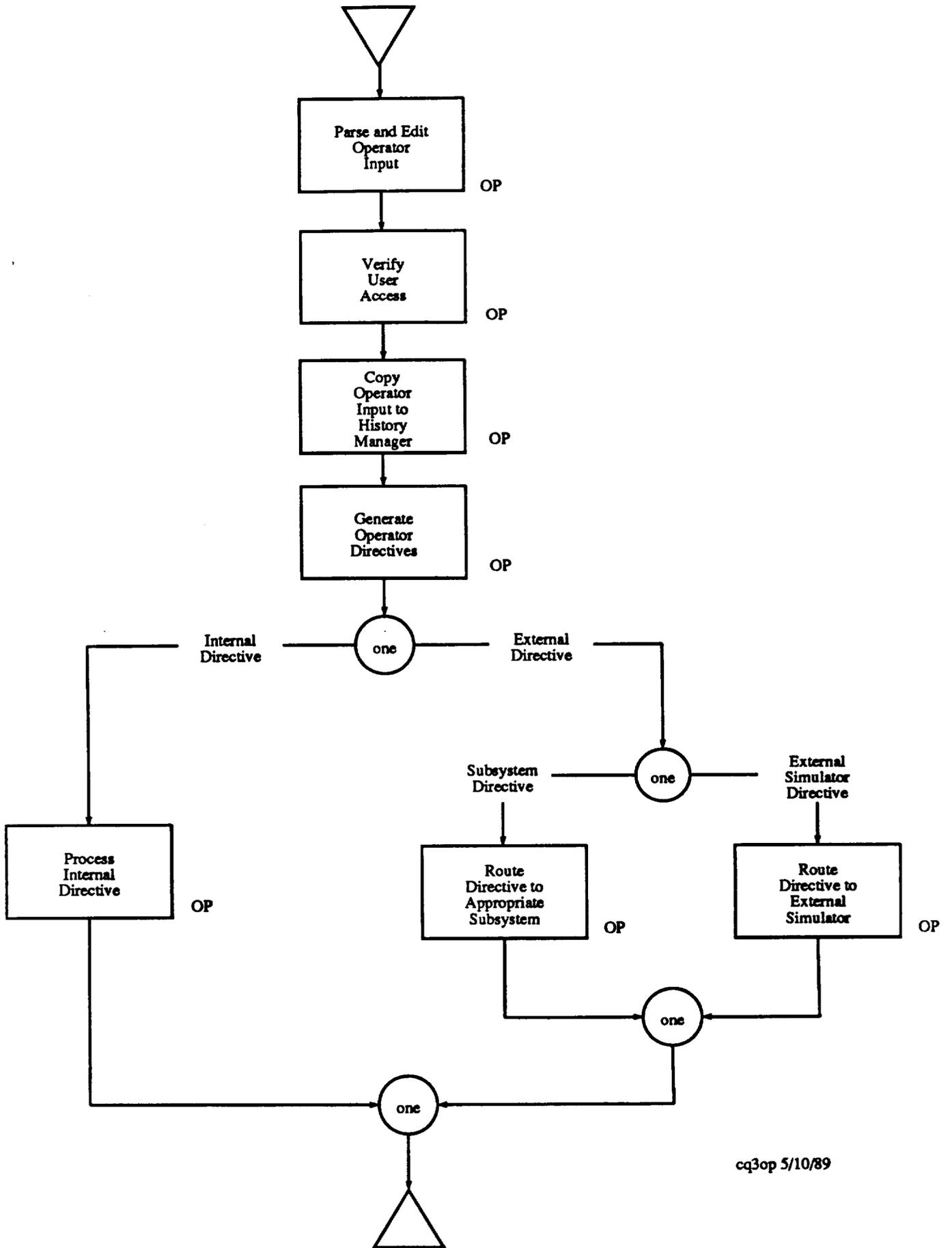
Process Data to be Sent to External Systems via NASCOM



Process NCC Block



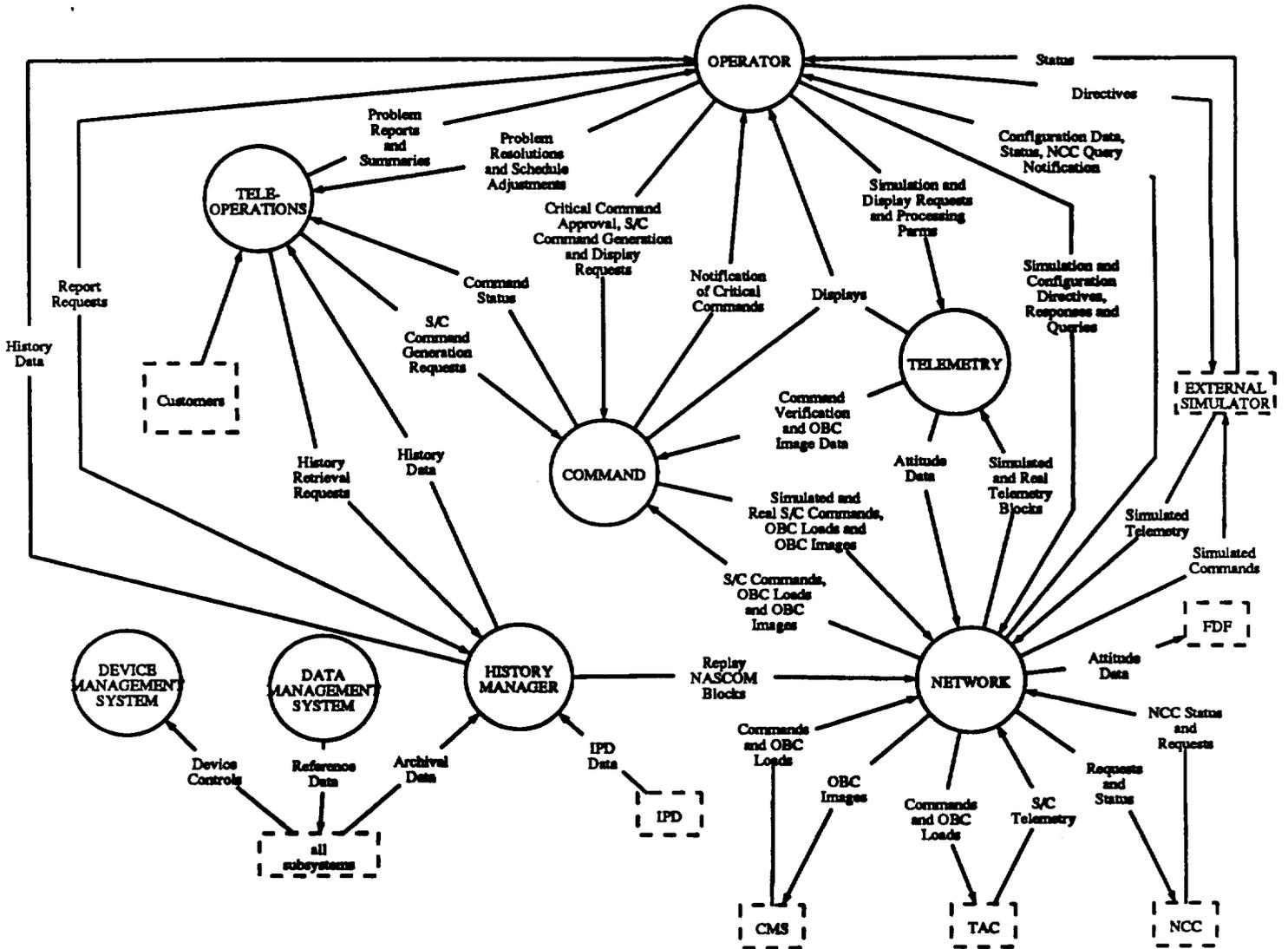
Process Operator Input



**POCC AP - GENERIC ARCHITECTURE**

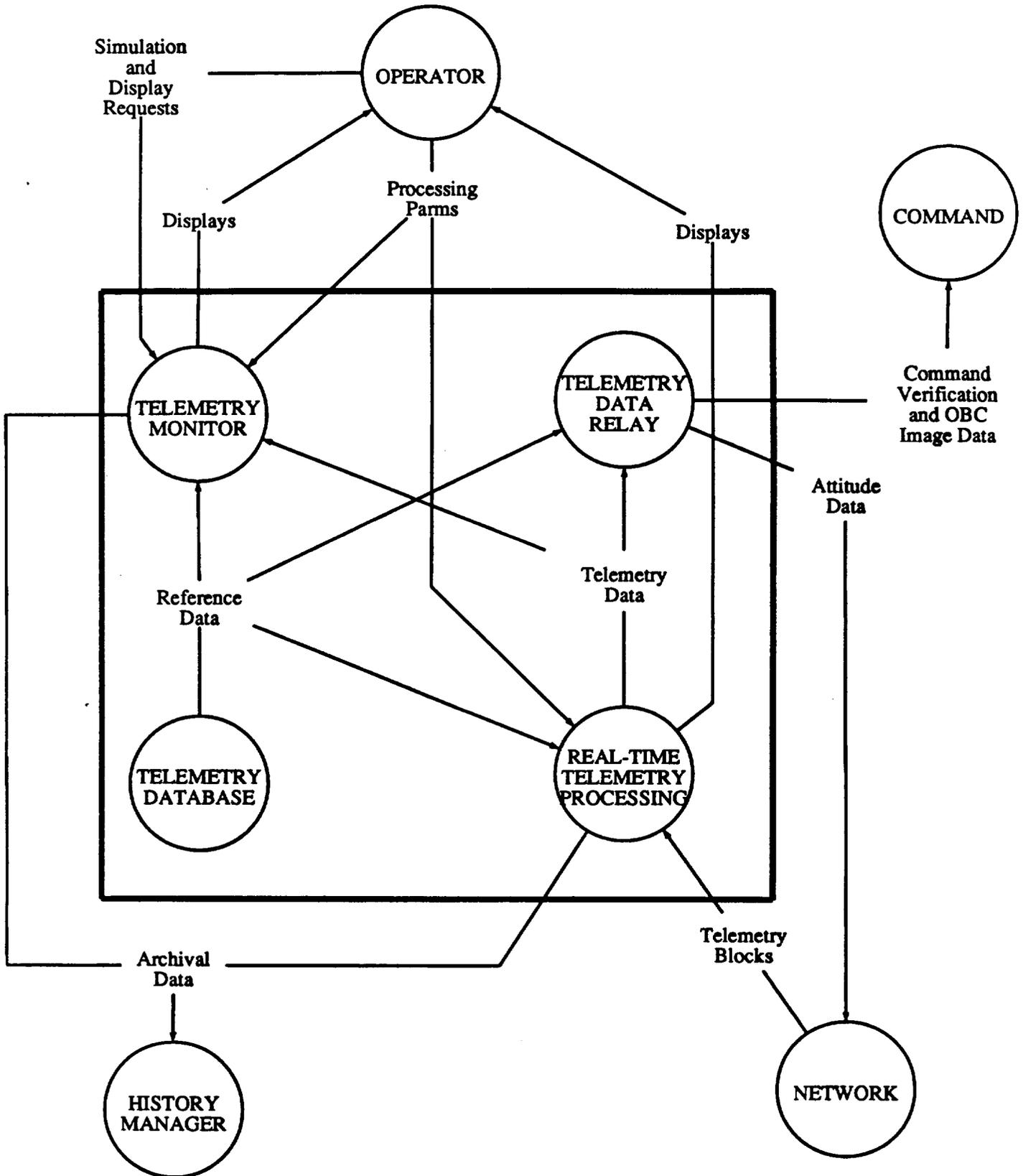
**Dataflow Diagrams**

POCC AP--GENERIC ARCHITECTURE LEVEL 0 DATA FLOW



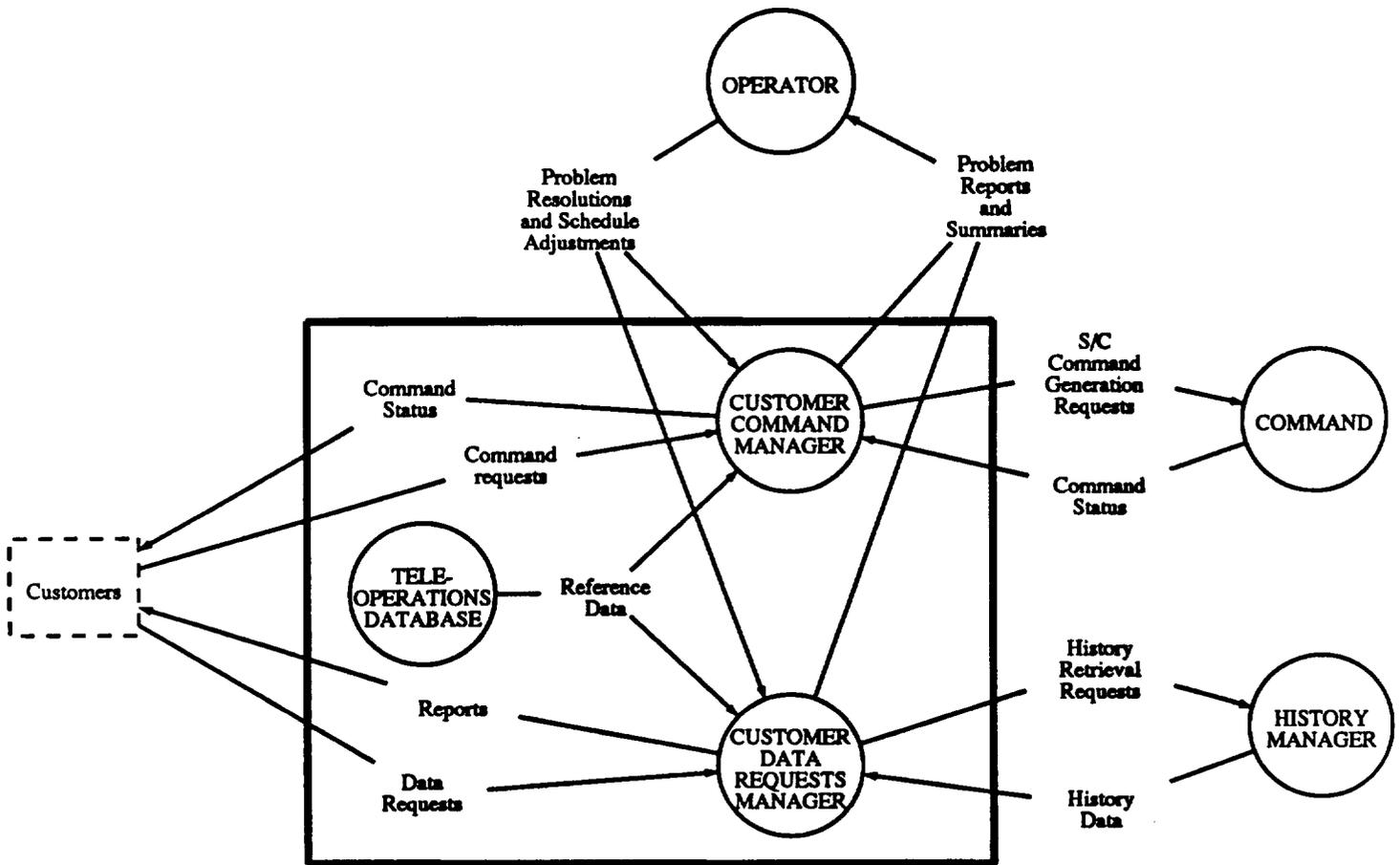
POCC AP--GENERIC ARCHITECTURE LEVEL 1 DATA FLOW

TELEMETRY



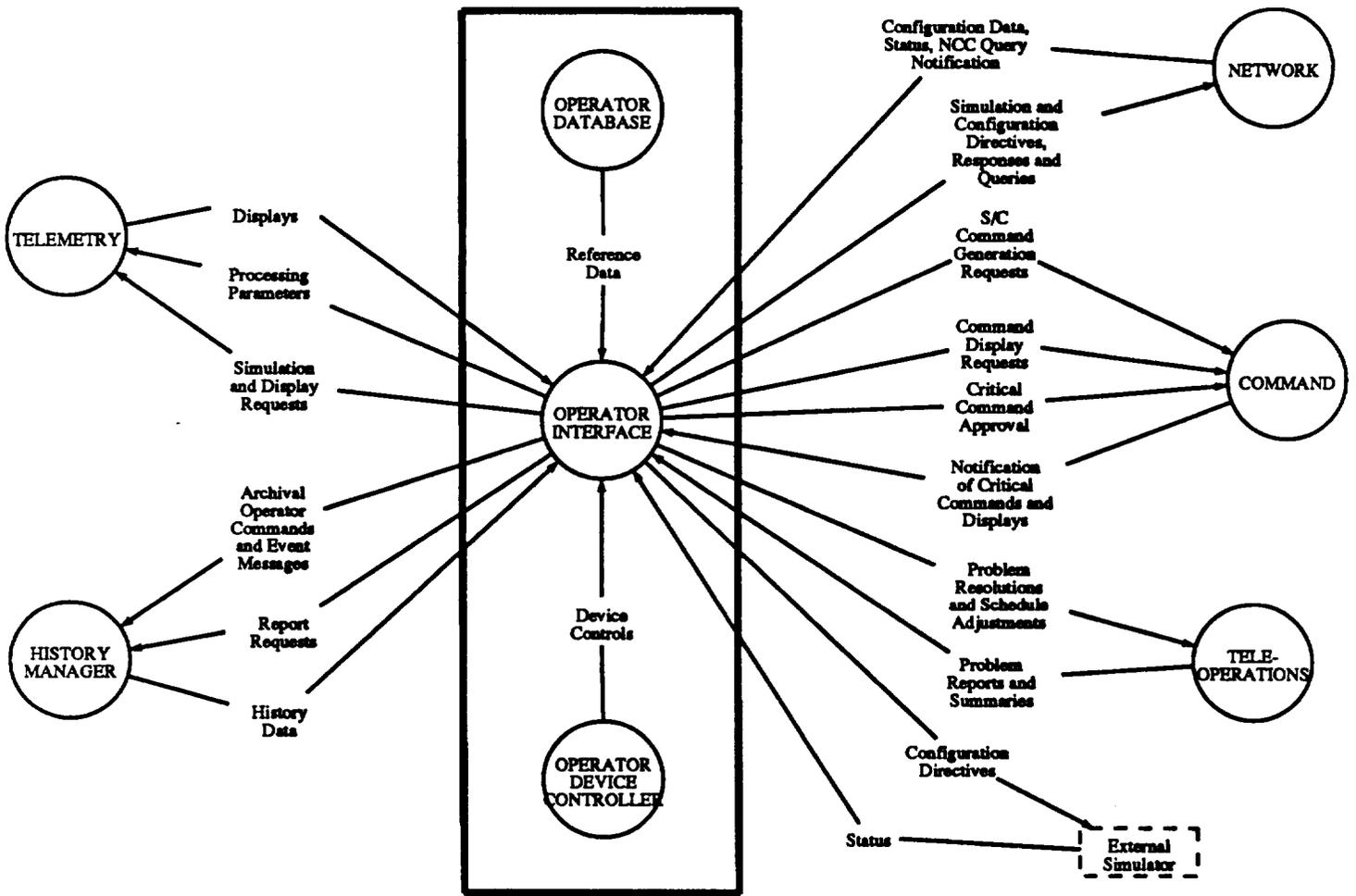
POCC AP--GENERIC ARCHITECTURE LEVEL 1 DATA FLOW

TELEOPERATIONS



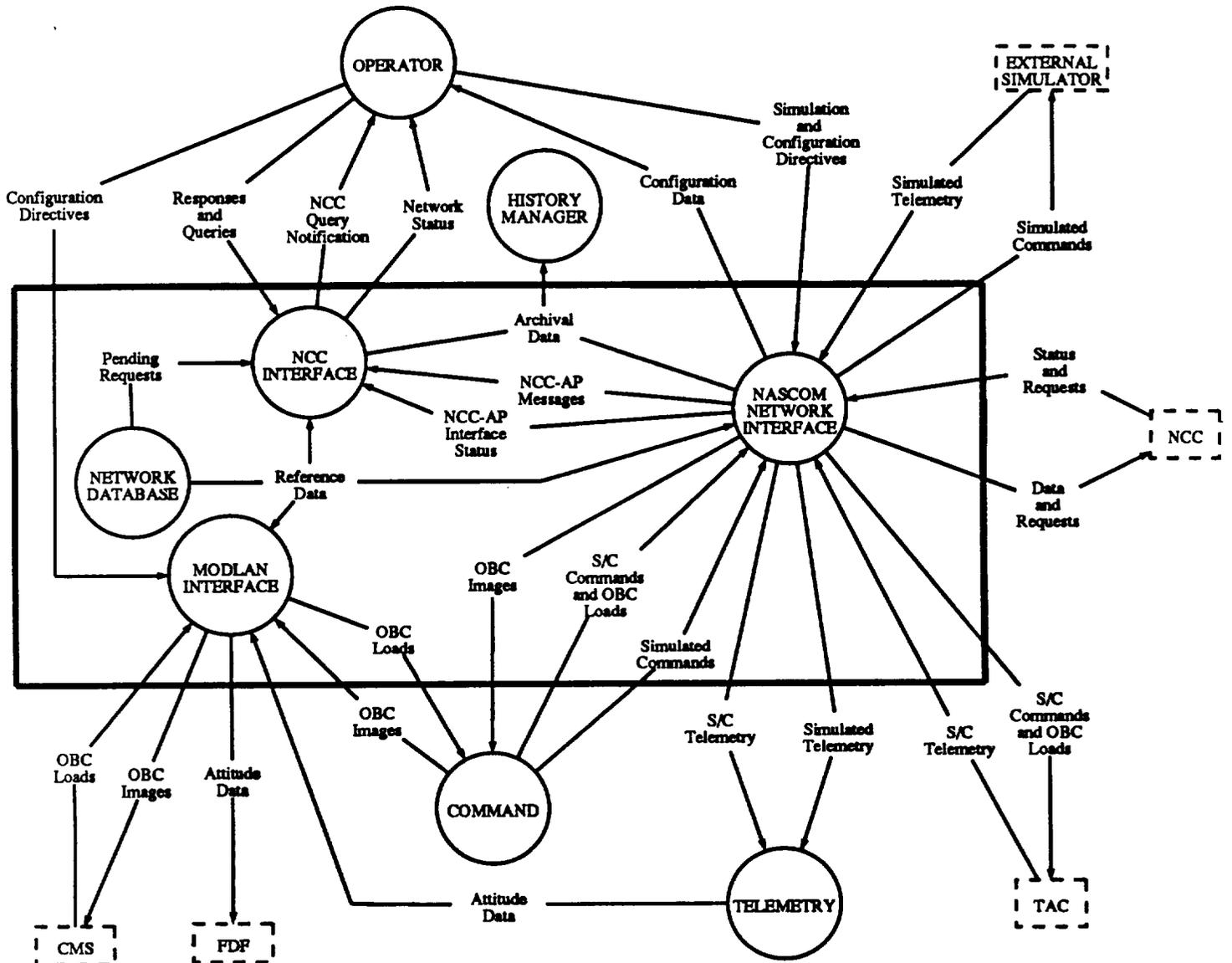
POCC AP-GENERIC ARCHITECTURE LEVEL 1 DATA FLOW

OPERATOR



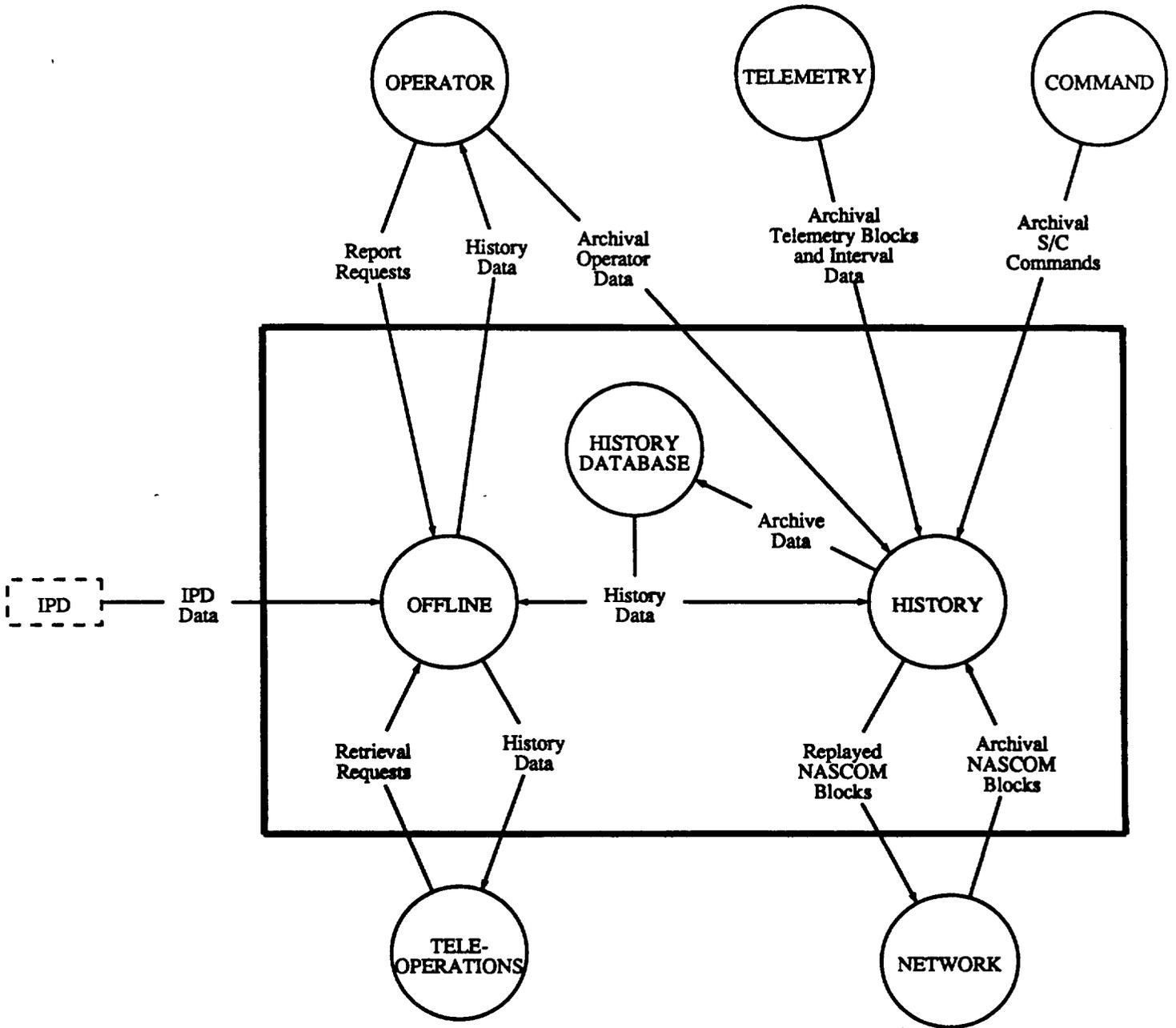
POCC AP--GENERIC ARCHITECTURE LEVEL 1 DATA FLOW

NETWORK



POCC AP-GENERIC ARCHITECTURE LEVEL 1 DATA FLOW

HISTORY MANAGER



POCC AP-GENERIC ARCHITECTURE LEVEL 1 DATA FLOW  
COMMAND

