Title: **Practical Points on Simultaneously Leveraging and Migrating Image Applications**

Presentation: 037

Authors: Matt Street & Donna Faudree

Company: Orion Group Software Engineers
5770 Nimtz Parkway
South Bend, IN 46628

Phone: (574) 233-3401

E-mail: mstreet@ogse.com
dfaudree@ogse.com

What We'll Cover Today

- Introduction
- EasyStep Approach
- N-Tier Architecture Design
- Data Access Approach
- Transactionalizing Entity Data
- Case Study
- Questions

Migration vs. Re-engineering

- Migration – the translation and rehosting of source code and data to run on Unix or other platforms
- Re-engineering – “modernizing” of applications using current technology, including the client-server model, graphical user interface (GUI), web-based, and relational databases

Migration vs. Re-engineering

Migration:

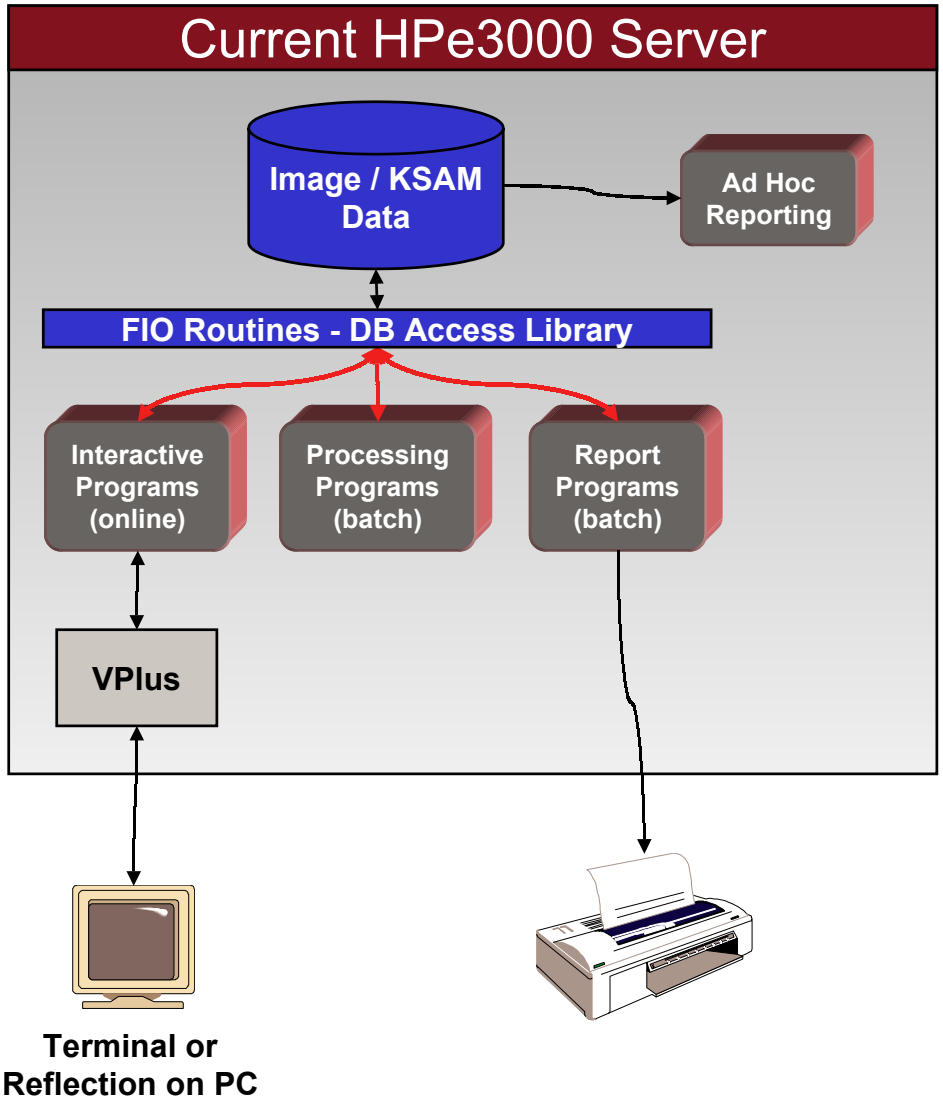
- COBOL reports and batch processes
- Image data

Re-engineering:

- VPLUS screens/applications

EasyStep Approach – Step 1

Current Environment

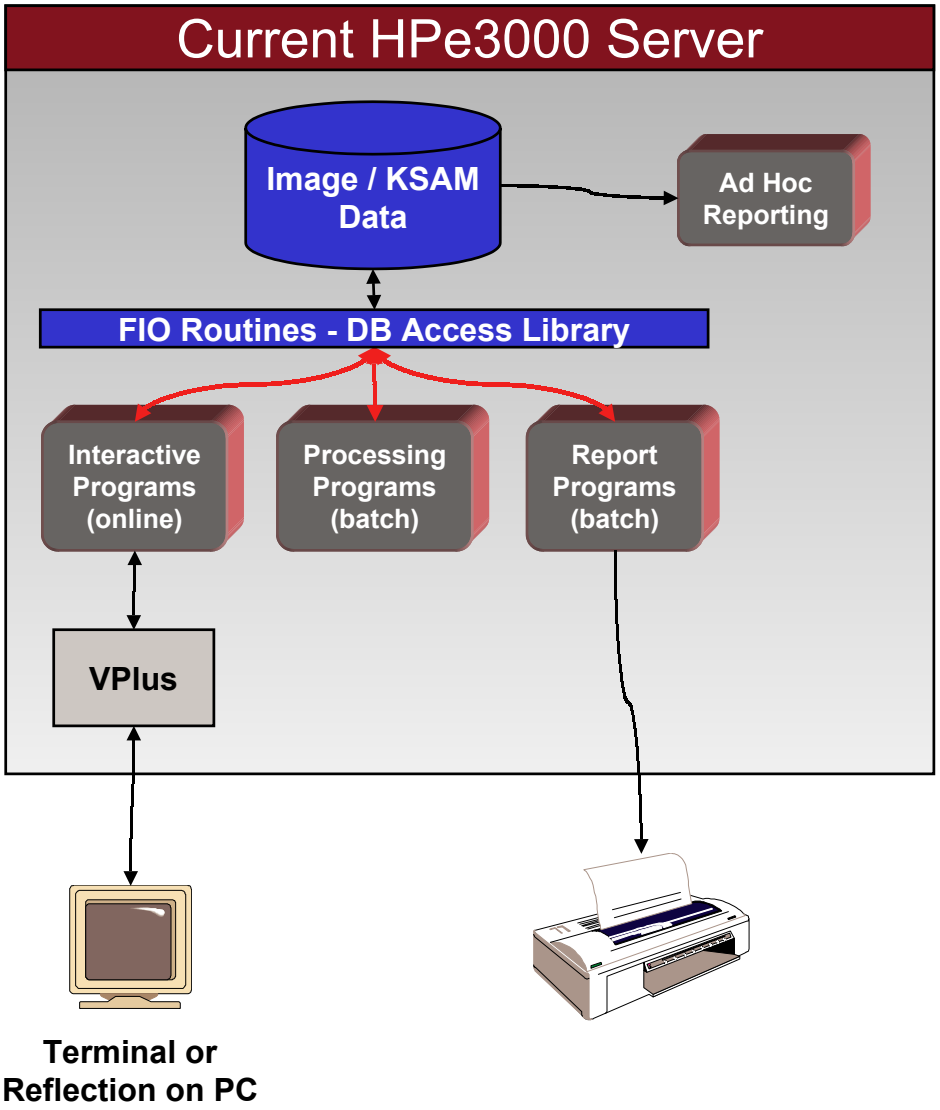


May Consists of:

- ❑ Interactive programs
- ❑ Background processes
- ❑ Report programs
- ❑ Image databases
- ❑ KSAM files
- ❑ VPLUS screens
- ❑ COBOL programs.

EasyStep Approach – Step 2

Develop Relational Model

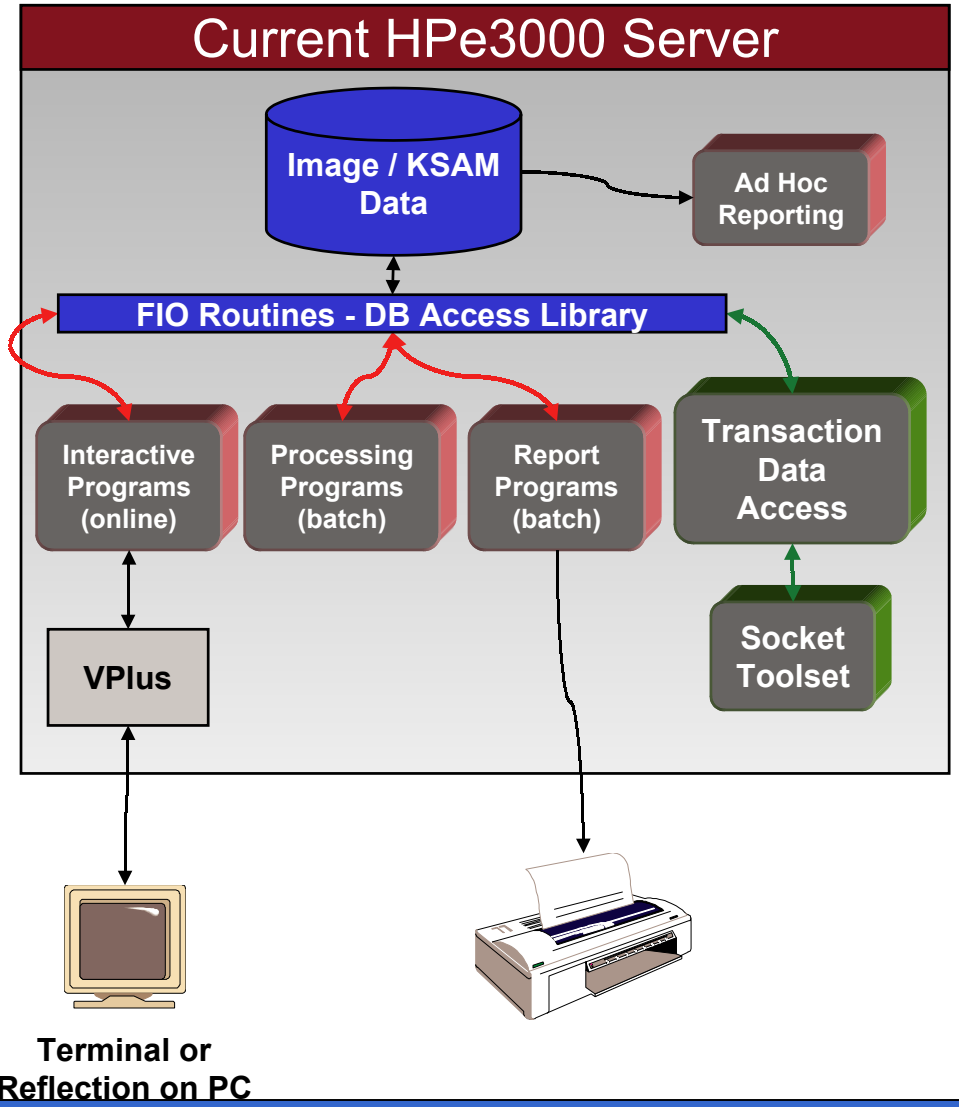


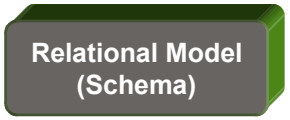
Relational Model (Schema)

This phase involves mapping the data structures on the HPe3000 into a relational database model. For the most part, this should be straight forward for well designed Image databases. Care should be taken to determine unique table keys, proper usage of data types, and to construct efficient relationships between tables.
All data which the application uses should be modeled at this time. It is not necessary to convert the data at this time, or choose the relational database / server platform the data will reside in.

EasyStep Approach – Step 3

Transactionalize HPe3000 Data

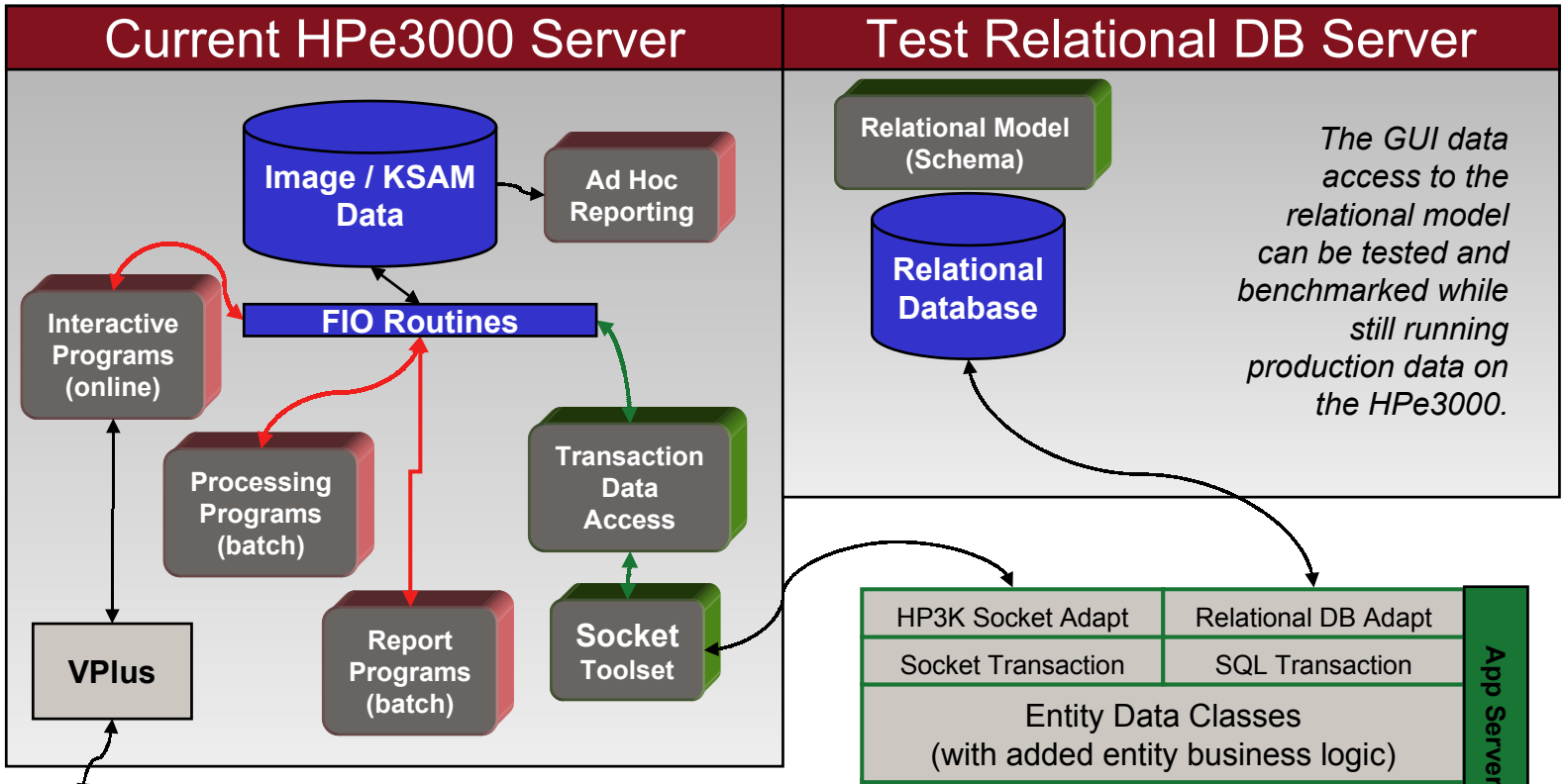


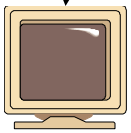


This phase involves creation of Transactions to access data on the HPe3000. These transactions will map to the entities (tables) defined in the relational model. Future GUI development and data migration (conversion) programs will utilize the data provided by these transactions.

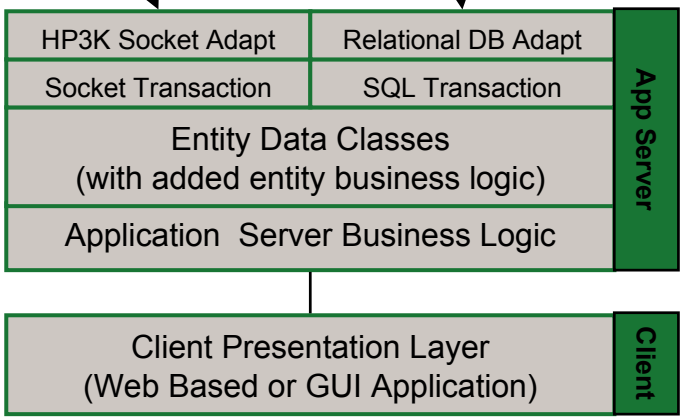
EasyStep Approach – Step 4

GUI Development



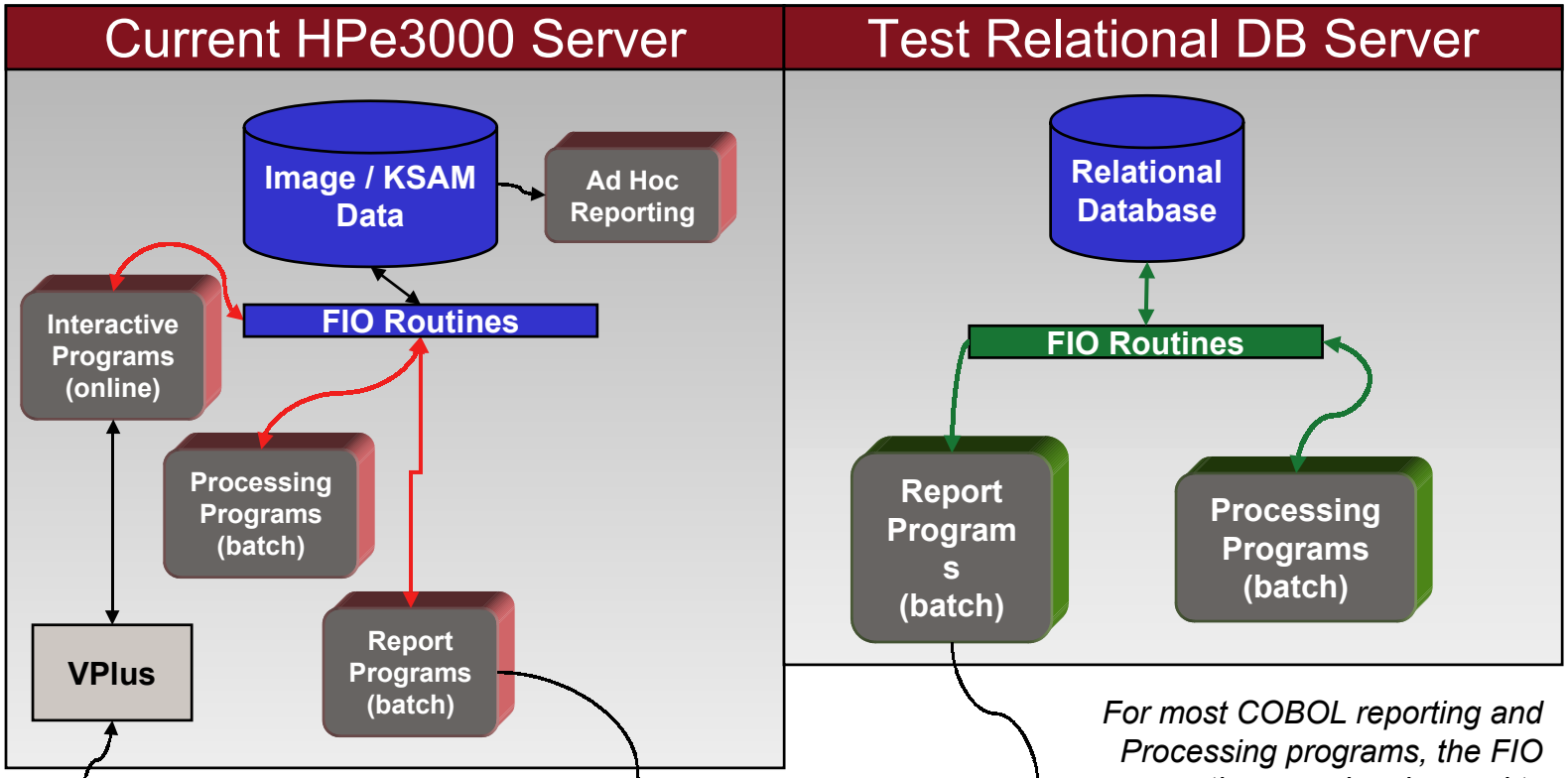


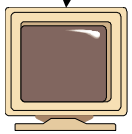
This phase involves creation of GUI or Web based screens that provide the functionality of the HPe3000 interactive programs. These screens can be constructed individually or in batches. Users should be able to use the existing HPe3000 based interactive programs simultaneously with the new GUI screens. The GUI screens should be developed to use the transaction data provided by the HPe3000 data structures or data from a relational database as defined by the relational model.



EasyStep Approach – Step 5

Batch Program Development



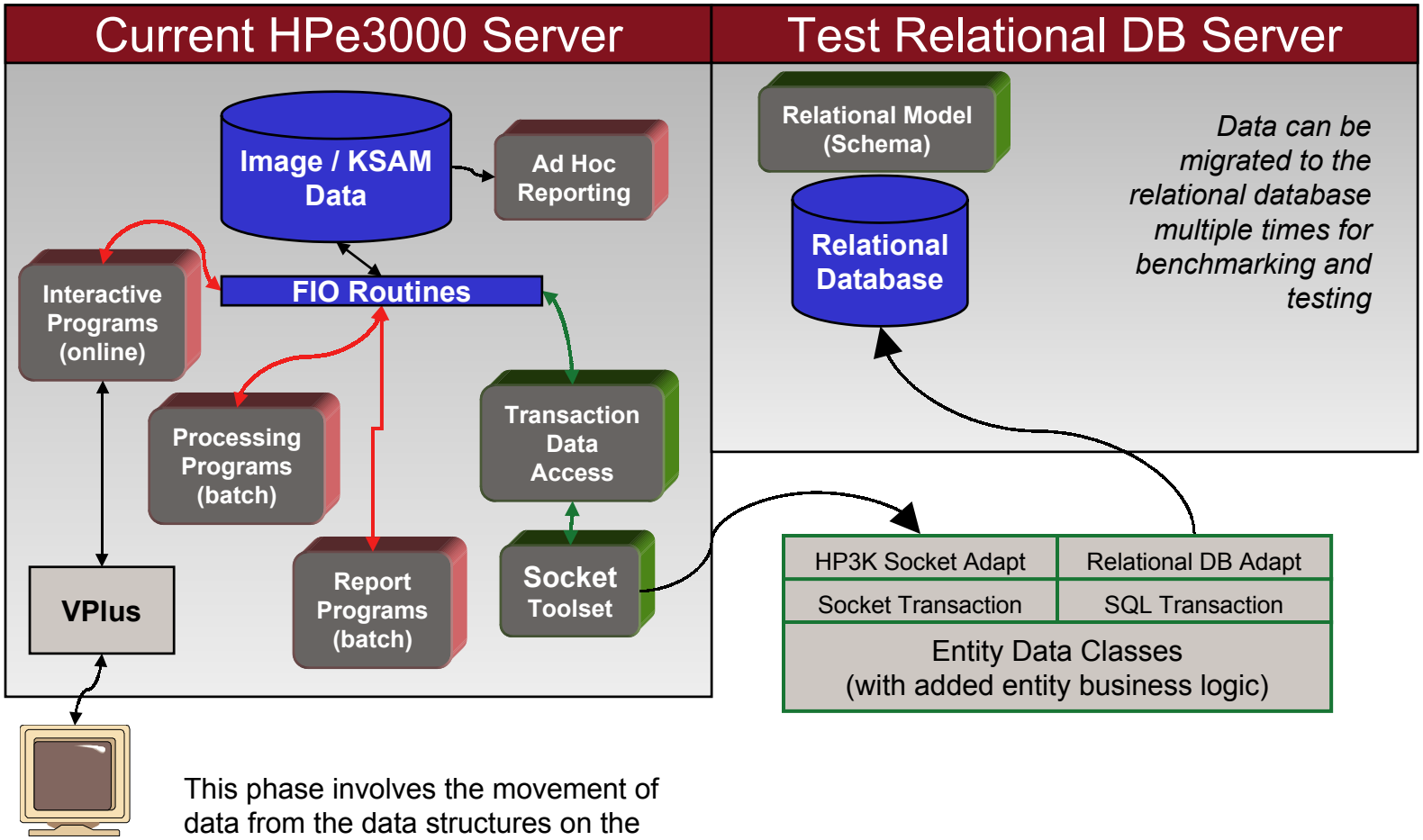


This phase involves the migration of batch programs which provide the reporting, posting or background transaction processing functions of the system. This phase can be done in conjunction with or separate from GUI development.

For most COBOL reporting and Processing programs, the FIO routines can be changed to access the new database and the COBOL migrated to a new COBOL compiler supported by the intended Database Server.

EasyStep Approach – Step 6

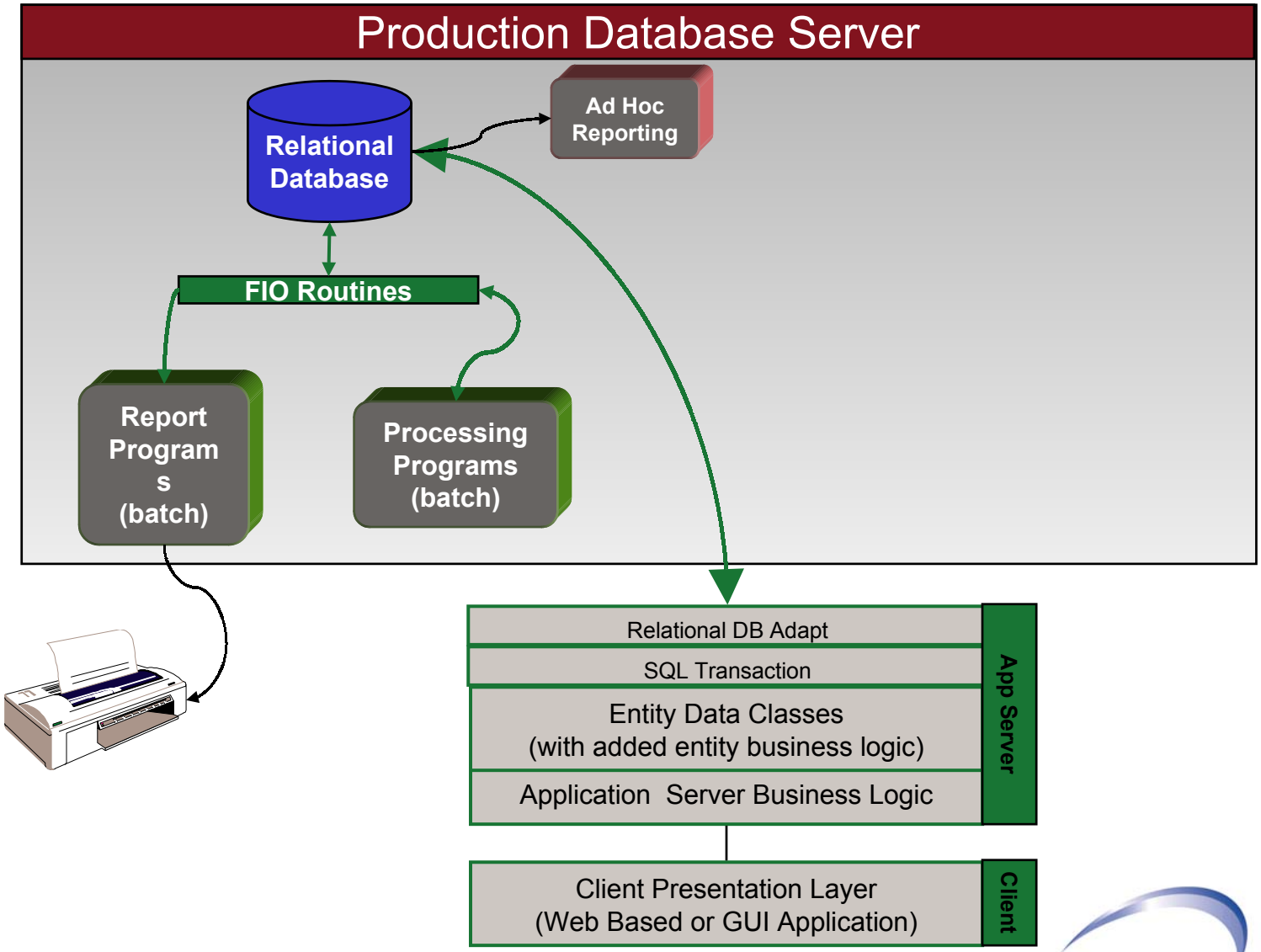
Data Migration (Conversion)



This phase involves the movement of data from the data structures on the HPe3000 to the relational database. This migration can occur for specific entities multiple times during the testing process and form the basis of the final conversion of data to the new database.

EasyStep Approach – Step 7

Final Solution



Benefits and Advantages with EasyStep Approach

- Gradual approach is possible
- New GUI screens can be used with current HP e3000 data
- Same screens can access Image data on HP e3000 and new platform (e.g., Oracle data on HP 9000)
- Parallel testing can take place

N-tier Architecture Design

- “n” represents any number of physical or logical layers
- Each layer is made up of software components that interact with other components within and across layers
- Consists of at least three layers: presentation, business, and data

N-tier Architecture Design (Cont.)

- Presentation Layer: provides software interface that end user sees
- Business Layer: provides the foundation for components that provide the processing, business rules, validation, data manipulation, data retrieval, error handling, and other core logic that makes the system “smart”
- Data Layer: Provide the storage platform

Benefits and Advantages with N-tier Architecture

- Scalability – Each component can reside on a separate machine to increase performance
- Flexibility – Components are interchangeable

Data Access Approach

Types of Data Processing

Online Transaction Processing (OLTP)

- High transaction throughput
- Add/change/delete data
- Predefined transactions
- Response times critical

Types of Data Processing (Cont.)

Information on Demand - Ad Hoc Reporting

- Inquire and Report data only
- Undefined transactions - different each time
- Response times not as critical

ODBC Defined

- Open Database Connectivity (ODBC)
- Used for standard access to multiple types of databases (Image, Oracle, etc)
- Allows PC access from applications like Excel to HP e3000 TurboImage data
- Converts SQL/ODBC calls to TurboImage calls
- Good for Ad Hoc Reporting and extracting data to other applications

Sockets Defined

- Sockets are a method of establishing a connection between different machines and/or operating systems
- Socket ports are similar to phone numbers for a machine
- Uses low level TurboImage native database access routines that are already developed
- Good for high volume OnLine Transaction Processing (OLTP) performance

ODBC Benchmarks

- Higher HP e3000 CPU usage
- Low concurrency of users, slower response times under load
- Database locking problems with other applications, not controlled by the application

Socket Technology Benchmarks

- Low HP e3000 CPU usage (similar to current native HP e3000 applications)
- High concurrency of users
- Database locking controlled by the server program on the HP e3000

Socket Technology Application Goals

- Fastest response on HP3000-based data
- Read and Update ability from the PC application to the HP e3000
- No performance hit to the production HP e3000 environment

Transactionalize HP e3000 Data

Socket Communications

- Socket communications are a way to connect different machines without understanding the different network protocols
- All connectivity between machines uses sockets at the low level
- Berkley Software Distribution Interprocess Communications (BSD IPC) is a standard available on almost all machines

Socket Communications (Cont.)

- Network Interprocess Communications (NetIPC) is similar and compatible to BSD IPC, but has additional functionality on the HP e3000
- Perform system calls to establish connections and transfer data between machines

Socket Communications (Cont.)

Processes required for socket connections:

- Listener Process (Waits for new connection requests)
- Server Process (Handles requests once connected)
- Client Process (Asks for connection, sends requests, accepts returned data)

Listener Process

- A background job on the HP e3000 waits for “calls” on a specified port from clients
- Define the port in the services.net.sys file with a port over 20,000
- Once a request is received, create a separate socket connection for the client to use when sending and receiving information across the socket and start an individual server process

Listener Process Example

```
MAIN.
  PERFORM OPEN-LISTENER.
  PERFORM LISTEN-FOR-CONNECTION UNTIL DONE.

OPEN-LISTENER.
  CALL INTRINSIC "IPCCREATE" USING SOCKETKIND,
       PROTOCOL, FLAGS, OPT, CALLDESC, RESULT.

LISTEN-FOR-CONNECTION.
  CALL INTRINSIC "IPCRECVCN" USING CALLDESC,
       VCDESC, FLAGS, \\, RESULT.
  <<Create and Activate Server Process>>
  CALL INTRINSIC "IPCGIVE" USING VCDESC,
       SOCKETNAME, NLEN, FLAGS, RESULT.
```

Server Process – HP e3000 Data Layer

- A unique child process is started for each client connection
- All socket communications are handled by the server process
- The server process handles transaction requests and enforces business rules
- Calls sub-routines to handle individual transaction requests

Server Process Example

```
MAIN.
  CALL INTRINSIC "IPCRECV" USING
       IPC-VCDESC, WS-SOCKET-IN,
       IPC-DLEN, IPC-FLAGS, \\,
       IPC-RESULT.
  IF TRAN-IN OF WS-SOCKET-IN = “CUSTI”
     CALL “CUSTOMER”.
  CALL INTRINSIC "IPCSEND" USING
       IPC-VCDESC, WS-SOCKET-OUT,
       IPC-DLEN, IPC-FLAGS, \\,
       IPC-RESULT.
```

Customer Transaction Example

```
CUSTOMER.
  MOVE CUST-KEY-IN OF WS-SOCKET-IN
          TO SEARCH-KEY.
  PERFORM READ-CUSTOMER.
  IF NOT CUSTOMER-FOUND
     PERFORM SET-READ-ERROR
  ELSE
     PERFORM LOAD-WS-SOCKET-OUT.
```

Presentation Layer

- Microsoft Excel & Visual Basic
- Microsoft Visual FoxPro
- Web Access

Client Process

- Use BSD IPC to connect to the HP e3000 on the predefined socket port
- Client initiates connection, then is moved to a separate unique socket connection for further transactions
- Each window on a client can have a unique socket connection or share one for the machine

Microsoft Visual FoxPro

- True Object Oriented Programming
- Using F1 Technologies' Visual Fox Express framework
- Uses C++ DLL for low level socket control

Visual FoxPro Example

OG Socket Cursor Class

Step: 4 - Define Cursor Behavior

Default Alias: Customer

Buffer Mode Override: 3 - Optimistic Row Buffering

Trans Code Length	8	Update Trans. Code	CUSTU
Add Trans. Code	CUSTA	List Trans. Code	CUSTL
Delete Trans. Code	CUSTD	List Next Trans. Code	CUSTN
Inquire Trans. Code	CUSTI	Previous Trans. Code	CUSTP

Finish... Cancel

Visual FoxPro Example (Cont.)

Customer Maintenance

Customer No. 100

Name Sunnyside Furniture

Address1 10736 Solar Blvd.

Address2 po box 123

City Los Angeles State AZ Zip Code 99999-9999

Country U.S.A.

Control Information | Contact Information | Financial Summary

Bal Method Open Item Accounting

Stmt Freq Quarterly

Location AT Atlanta, GA

Territory XX

Frt Pmt Cd Prepaid

Partial Ship? Yes No

Finance Chg? Yes No

Discount % 0.00

Credit Status Good Credit

Credit Limit 10,000

Credit Rate BBB

Terms Code N Cash with orde

Ship Via F Federal Express

Tax Code NY NEW YORK STATE SALES TAX

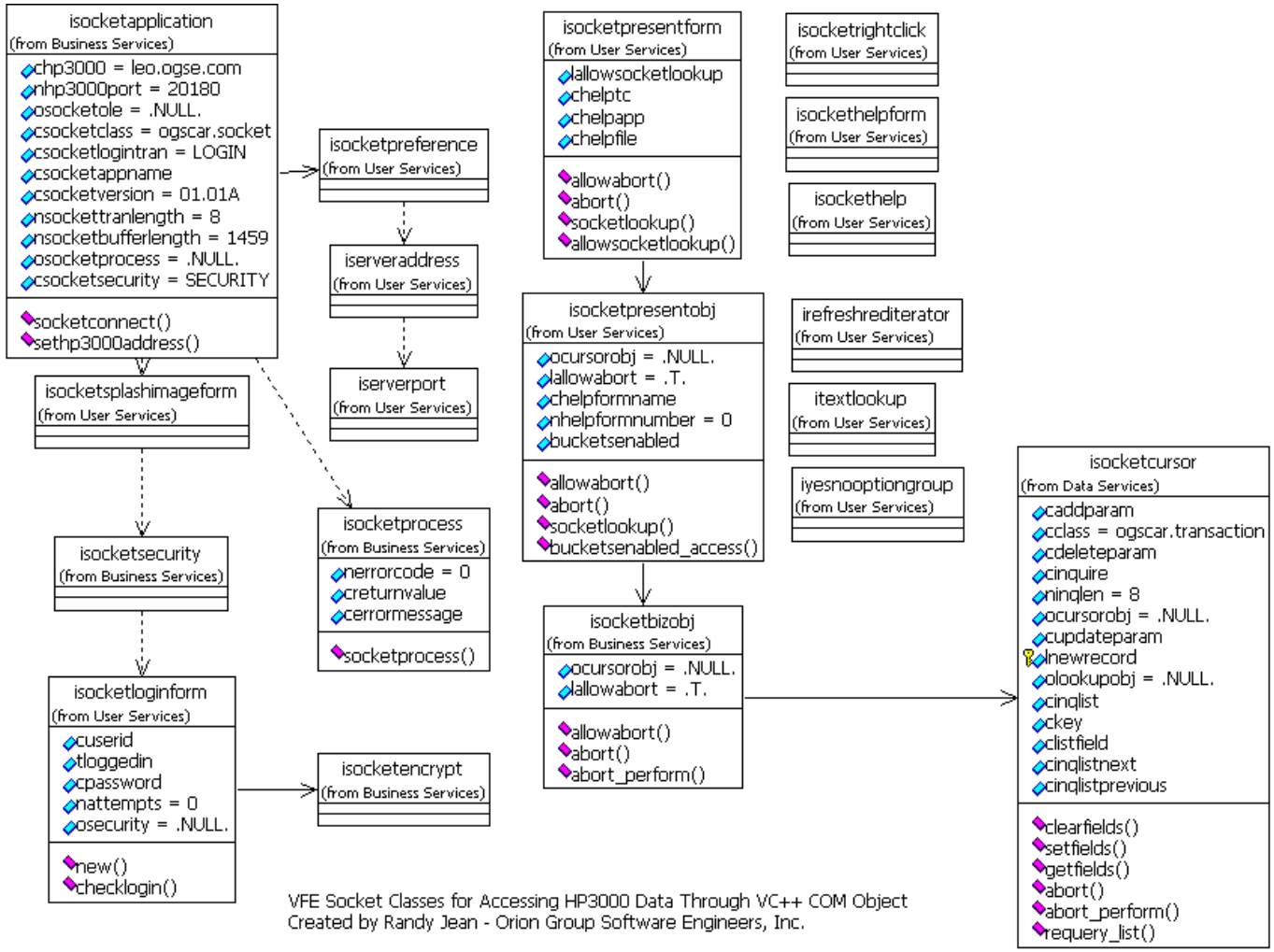
Sales Rep No 500 Norm Z. Goldstein

A/R Account FURNITUR-00100060-01000000

Accounts Receivable - West

Comment GOOD CUSTOMER

Visual FoxPro Example (Cont.)



VFE Socket Classes for Accessing HP3000 Data Through VC++ COM Object
Created by Randy Jean - Orion Group Software Engineers, Inc.

Web Access

- The transactions created on the HP e3000 can also be used to create web applications
- Works with HTML, Active Server Pages, Web Objects, etc.
- Orion Group has created socket frameworks in Java and C++ to aide in application development

Web Access Example

Account Information - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Shop Stop

Bookmarks Location: ebObjects/ECDemo.woa/32/wo/sI54DUDsZlql2C8ExTc6e5kpFIX/1.6.21.0 What's Related

Staff Locator Instant Message SmartUpdate

Cass Furniture Distribution

Products Search Quick Entry View Order Account Info Order History

Account Information

Address	Purchases		Payments	
Sunnyside Furniture (100)	Period to Date	$104.56	Last Payment	$50.79
10736 Solar Blvd.	Year to Date	$2,000.90	Payment Date	07/31/00
po box 123	Last Year	$344.45	Current balance	$344.56
Los Angeles, AZ 99999-9999				
			Other	
Phone number 213-657-4321			Credit Limit	$10,000.00

Document: Done

Case Study

Proof-Of-Concept Project

Scope:

- Migrate a subset of Image datasets to Oracle tables
- Migrate one COBOL batch program
- Re-engineer one screen [create a GUI and a Web application]

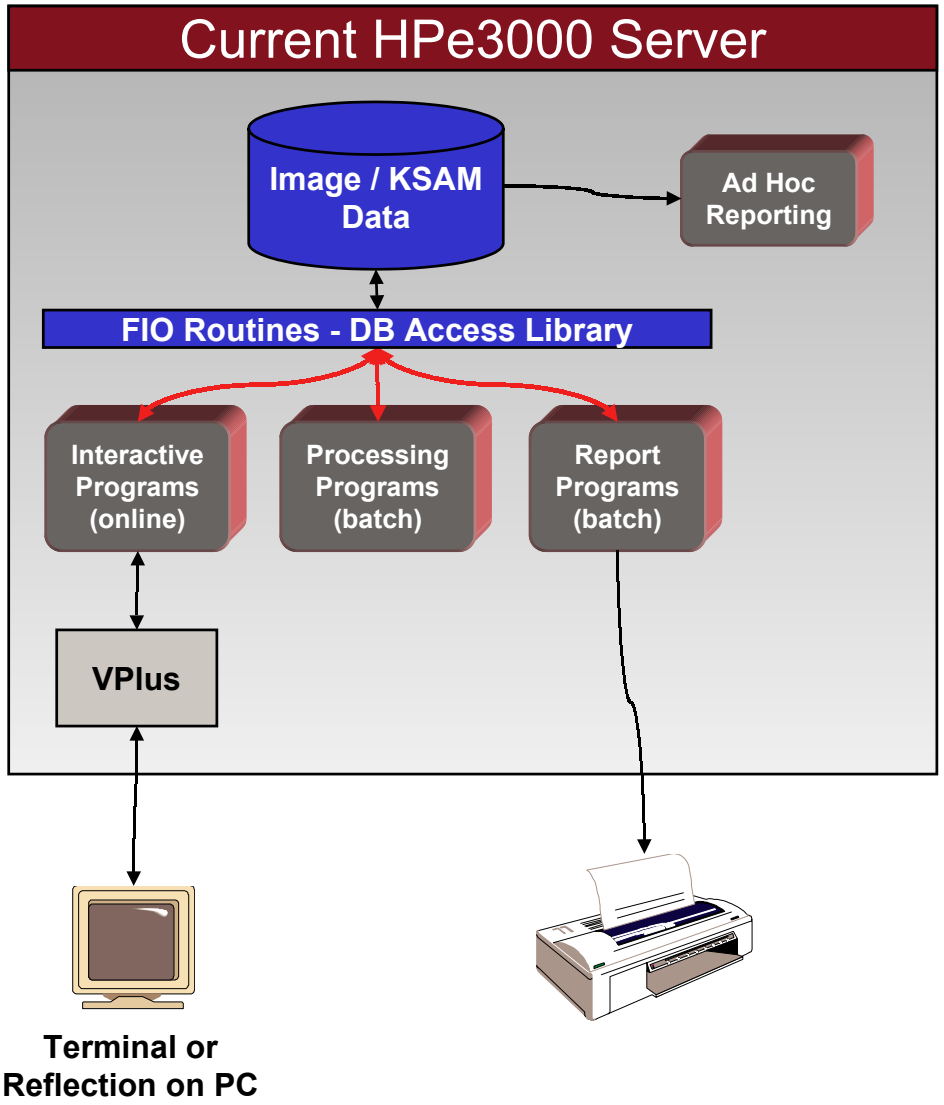
Proof-Of-Concept Project

Purpose:

- Determine technical issues related to migration of data and COBOL programs
- Help client determine online development tool (Visual FoxPro or WebObjects)
- Develop “framework” for migration projects to increase efficiency and productivity while decreasing cost

EasyStep Approach – Step 1

Current Environment



May Consists of:

- ❑ Interactive programs
- ❑ Background processes
- ❑ Report programs
- ❑ Image databases
- ❑ KSAM files
- ❑ VPLUS screens
- ❑ COBOL programs.

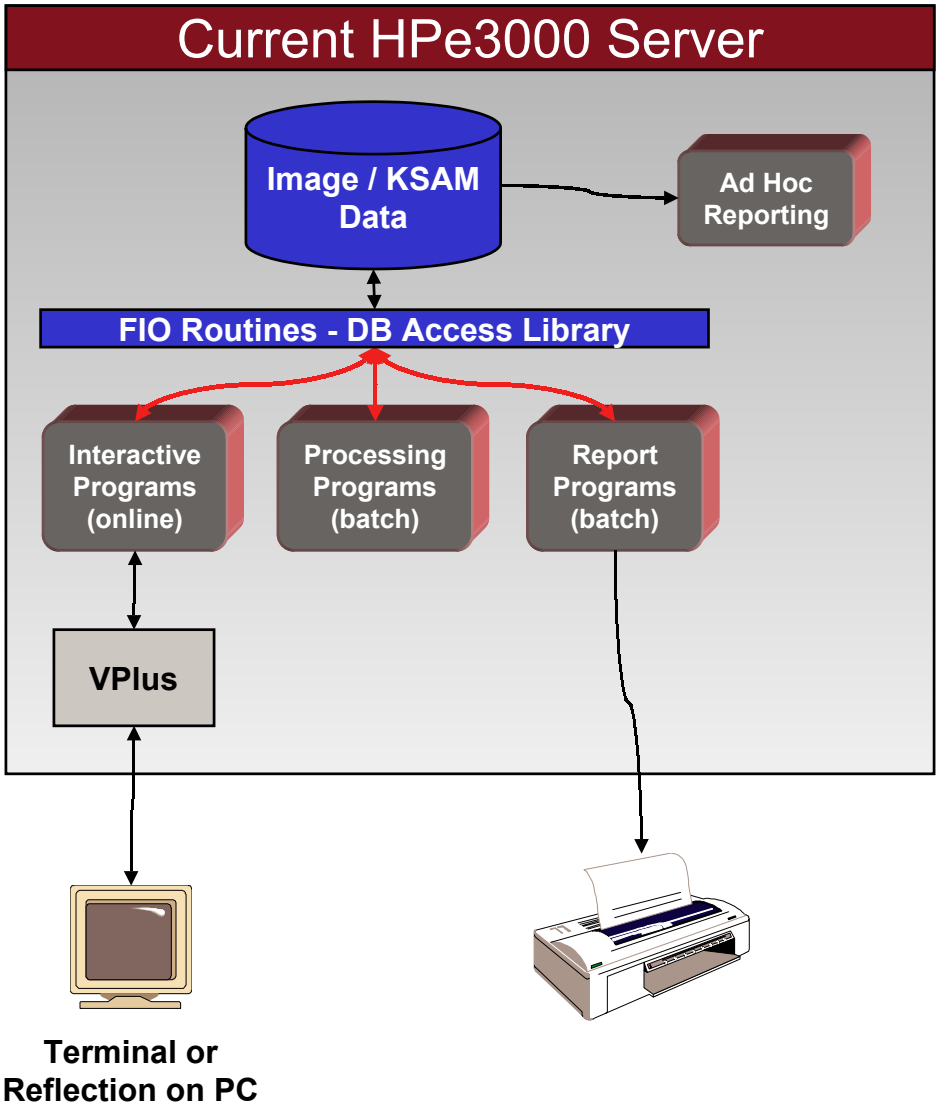
EasyStep Approach – Step 1

Current Environment

- Interactive programs developed in Pascal and COBOL
- Background processes developed in COBOL
- Report programs developed in COBOL
- FIO routines developed in Pascal
- Multiple Image Databases
- Vplus User Interface

EasyStep Approach – Step 2

Develop Relational Model



Relational Model (Schema)

This phase involves mapping the data structures on the HPe3000 into a relational database model. For the most part, this should be straight forward for well designed Image databases. Care should be taken to determine unique table keys, proper usage of data types, and to construct efficient relationships between tables.
All data which the application uses should be modeled at this time. It is not necessary to convert the data at this time, or choose the relational database / server platform the data will reside in.

Data Mapping Issues:

- Data buffer should be all ascii data
- Where appropriate, format data in data buffer (e.g., 23.99, 01/05/2002, (574)233-3401, etc.)
- Where possible, keep HP e3000 socket data buffers and relational database tables consistent
- Where necessary, include unique key [to make relational database tables more efficient]

EasyStep Approach – Step 2

Develop Relational Model

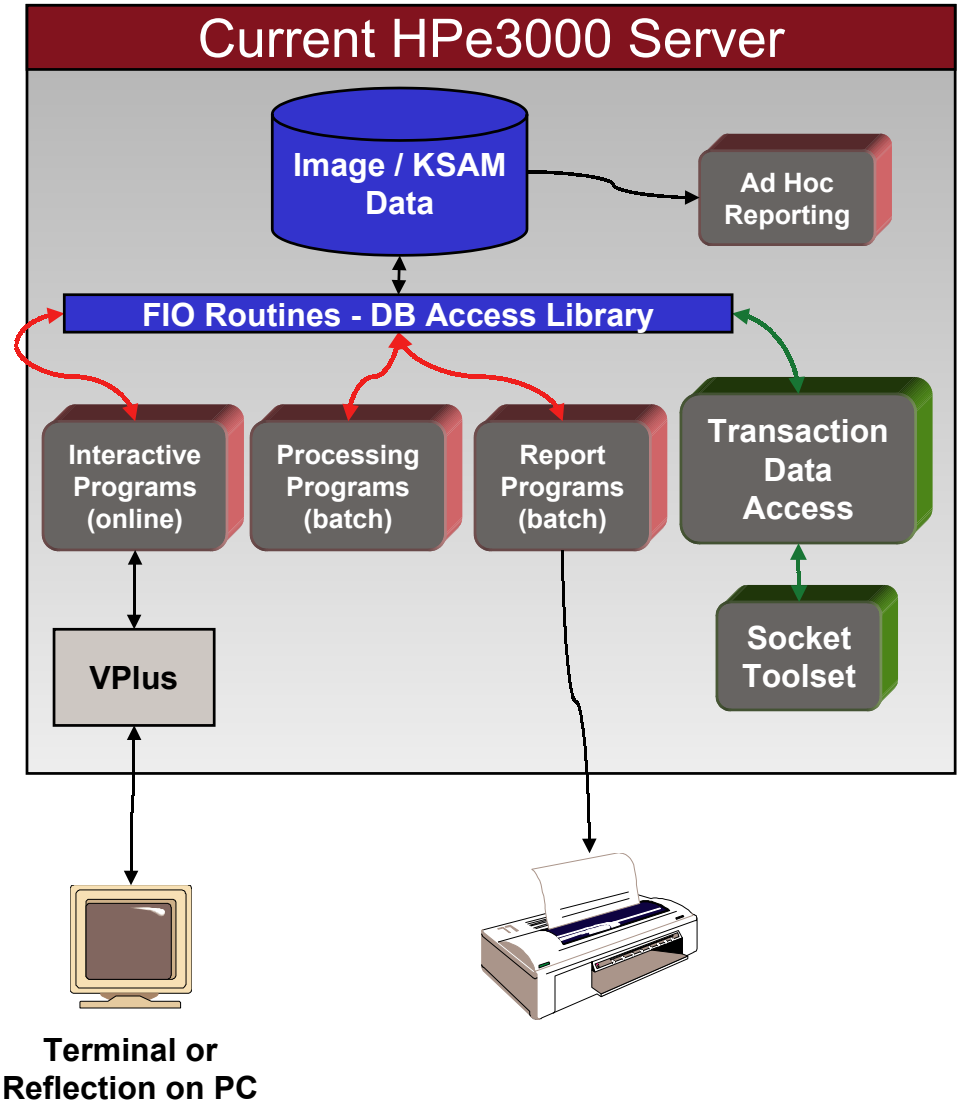
Image (Patient, Manual Master)	Oracle (Patient table)
PATIENT-ID, I2	PATIENTID, Number (9)
PATIENT-NO, X10	PATIENTNO, Char (10)
DOCTOR-ID, I2	DOCTORID, Number (9)
GUARANTOR-ID, I2	GUARANTORID, Number (9)
HICDA-ID, I2	HICDAID, Number (9)
BED, X2	BED, Char (2)
BIRTHD, I2	BIRTHD, Date
CLRKID, X8	CLRKID, Char (8)
COMENT, X60	COMENT, Char (60)
DOCTR2 ,I2	DOCTR2 , Number (9)
DOCTR3, I2	DOCTR3, Number (9)
DOCTR4, I2	DOCTR4, Number (9)
LMPDATE, J2	LMPDATE, Date
ENTER-DATE, J2	ENTERDATE, Date
FLAGS, X10	EMPLSTATUS, ISOLATION, MARITALSTATUS, PSEX, PTYPE, RACE, REL, SOFCR, STUDENTSTATUS, FREE Char (1) each
HICDA1-ID, I2	HICDA1ID, Number (9)
AUTHORIZATION-NO, X10	AUTHORIZATIONNO, Char (10)
DATE-LAST-ACTIVE, J2	DATELASTACTIVE, Date
DATE-LAST-MERGED, J2	DATELASTMERGED, Date
NAME1, X30	NAME1, Char (30)
PDOA, J2	PDOA, Date
PDOD, J2	PDOD, Date
ROOM, X4	ROOM, Char (4)
SERIES, I1	SERIES, Number (4)
SOFDATE, J2	SOFDATE, Date
HNA-LOCTN, X4	HNALOCTN, Char (4)
ADRES1, X34	ADRES1, Char (34)
ADRES2, X34	ADRES2, Char (34)
CTYST, X34	CTYST, Char (34)
PHONE, P12	PHONE, Char (13)
ZIP, X10	ZIP, Char (10)
INS-MEMBER-NO, X6	INSMEMBERNO, Char (6)
HICDA2-ID, I2	HICDA2ID, Number (9)
HICDA3-ID, I2	HICDA3ID, Number (9)

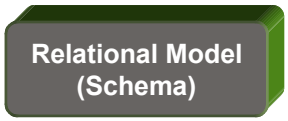
Patient Data Buffer:

PATIENTID	X9
PATIENTNO	X10
DOCTORID	X9
GUARANTORID	X9
HICDAID	X9
BED	X2
BIRTHD	XX/XX/XXXX
CLRKID	X8
COMENT	X60
DOCTR2	X9
DOCTR3	X9
DOCTR4	X9
LMPDATE	XX/XX/XXXX
ENTERDATE	XX/XX/XXXX
EMPLSTATUS	X1
ISOLATION	X1
MARITALSTATUS	X1
PSEX	X1
PTYPE	X1
RACE	X1
REL	X1
SOFCR	X1
STUDENTSTATUS	X1
FREE	X1
HICDA1ID	X9
AUTHORIZATIONNO	X10
DATELASTACTIVE	XX/XX/XXXX
DATELASTMERGED	XX/XX/XXXX
NAME1	X30
PDOA	XX/XX/XXXX
PDOD	XX/XX/XXXX
ROOM	X4
SERIES	X4
SOFDATE	XX/XX/XXXX
HNALOCTN	X4
ADRES1	X34
ADRES2	X34
CTYST	X34
PHONE	(XXX)XXX-XXXX
ZIP	X10
INSMEMBERNO	X6
HICDA2ID	X9
HICDA3ID	X9

EasyStep Approach – Step 3

Transactionalize HPe3000 Data





This phase involves creation of Transactions to access data on the HPe3000. These transactions will map to the entities (tables) defined in the relational model. Future GUI development and data migration (conversion) programs will utilize the data provided by these transactions.

EasyStep Approach – Step 3

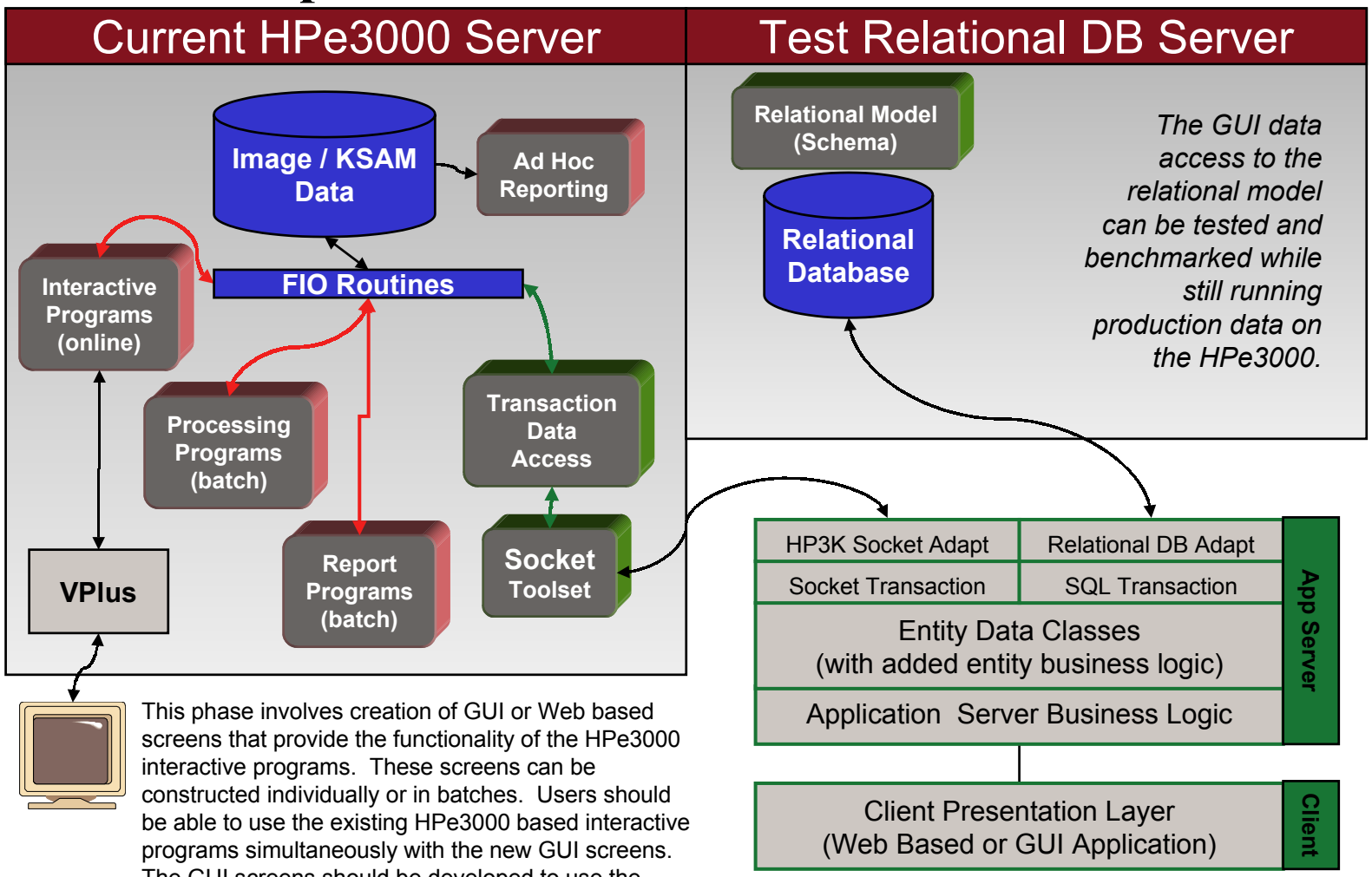
Transactionalize HPe3000 Data –Patient Transaction

```
PATIENT.
  MOVE PA-KEY-IN OF WS-SOCKET-IN
         TO SEARCH-KEY.
  PERFORM READ-PATIENT.
  IF NOT PATIENT-FOUND
     PERFORM SET-READ-ERROR
  ELSE
     PERFORM LOAD-WS-SOCKET-OUT.
```

*WS-SOCKET-OUT matches Patient Data Buffer

EasyStep Approach – Step 4

GUI Development



This phase involves creation of GUI or Web based screens that provide the functionality of the HPe3000 interactive programs. These screens can be constructed individually or in batches. Users should be able to use the existing HPe3000 based interactive programs simultaneously with the new GUI screens. The GUI screens should be developed to use the transaction data provided by the HPe3000 data structures or data from a relational database as defined by the relational model.

EasyStep Approach – Step 4

GUI Development – Visual FoxPro

OG Socket Cursor Class

Step: 4 - Define Cursor Behavior

Default Alias: Patient

Buffer Mode Override: 3 - Optimistic Row Buffering

Trans Code Length	8	Update Trans. Code	PATIENTU
Add Trans. Code	PATIENTA	List Trans. Code	PATIENTL
Delete Trans. Code	PATIENTD	List Next Trans. Code	PATIENTN
Inquire Trans. Code	PATIENTI	Previous Trans. Code	PATIENTP

Finish... Cancel

EasyStep Approach – Step 4

GUI Development – Visual FoxPro (Continued)

Patient Profile

Patient# A555555551 OTHER/UNK

Name TAYLOR, TOM Sex M Race X DOB 12/02/1950 Age 51

Authorz# Status: Marital S Student N Employment F Entered 05/16/2001

Address 4784 NORTH Rm-Bd -

City, St SOUTH BEND, IN

Zip 46614- Phone ( ) -

LAST

Active 05/16/2001

Series 2

Mod By MGR

Merged / /

Inpatient Services: Admission Date / / Discharge / /

Billing Information

Guarantor# A555555551 PA's Relationship to GU

Name TAYLOR, TOM PA SOF? Y 05/16/2001 GU SOF? Y 05/16/2001

Attn Mbr# Insur: GU

Additional Info Autho Auth# Optional

EasyStep Approach – Step 4

GUI Development – Visual FoxPro (Continued)

Image data access:

- Uses VFP tables to parse data buffers
- Application, cursor, business, presentation, and form classes based on custom “iSocket” classes

Oracle data access:

- Uses remote SQL views
- Application, cursor, business, presentation, and form classes based on standard Visual FoxPro classes

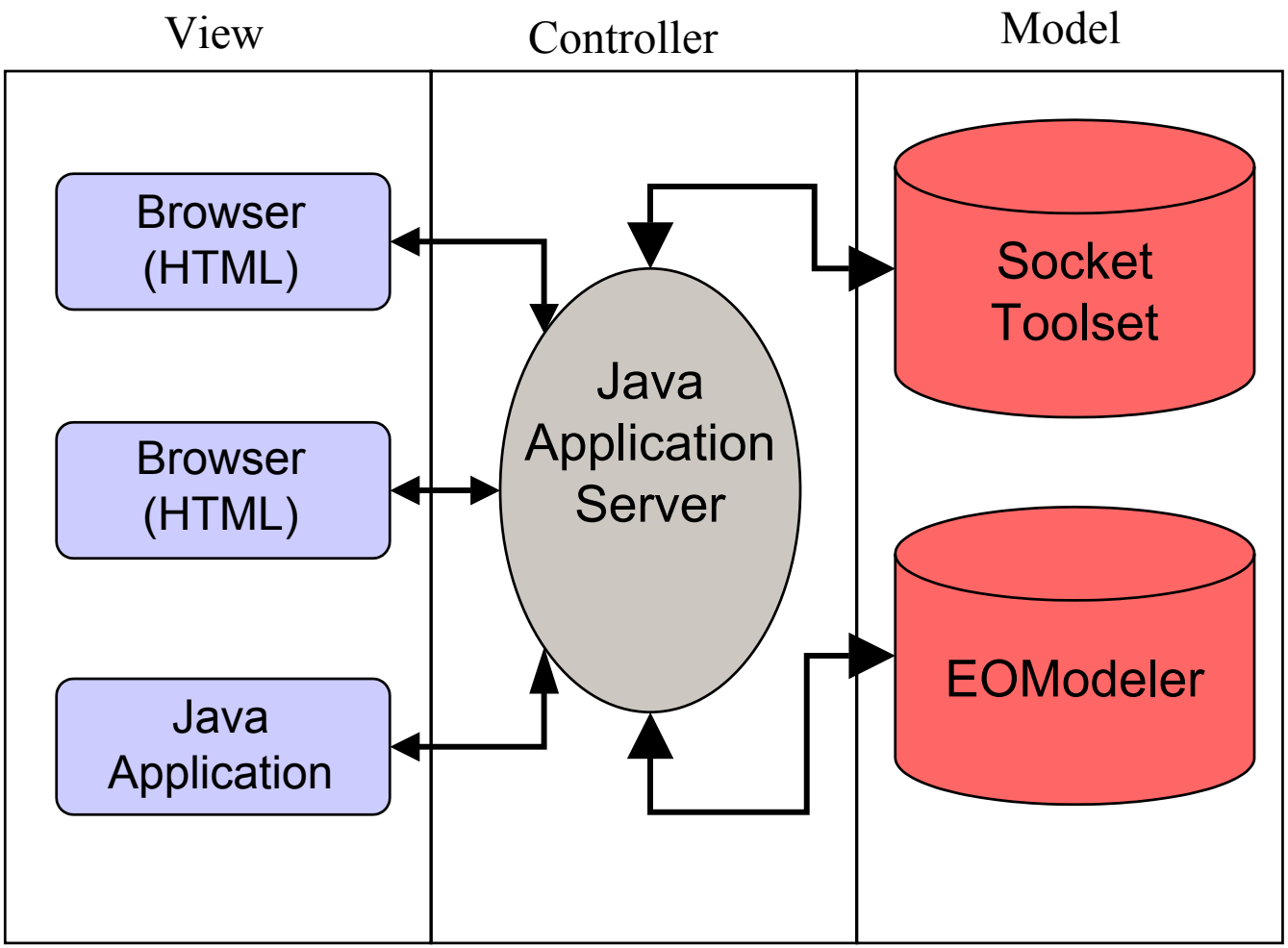
EasyStep Approach – Step 4

GUI Development – WebObjects

WebObjects Demo

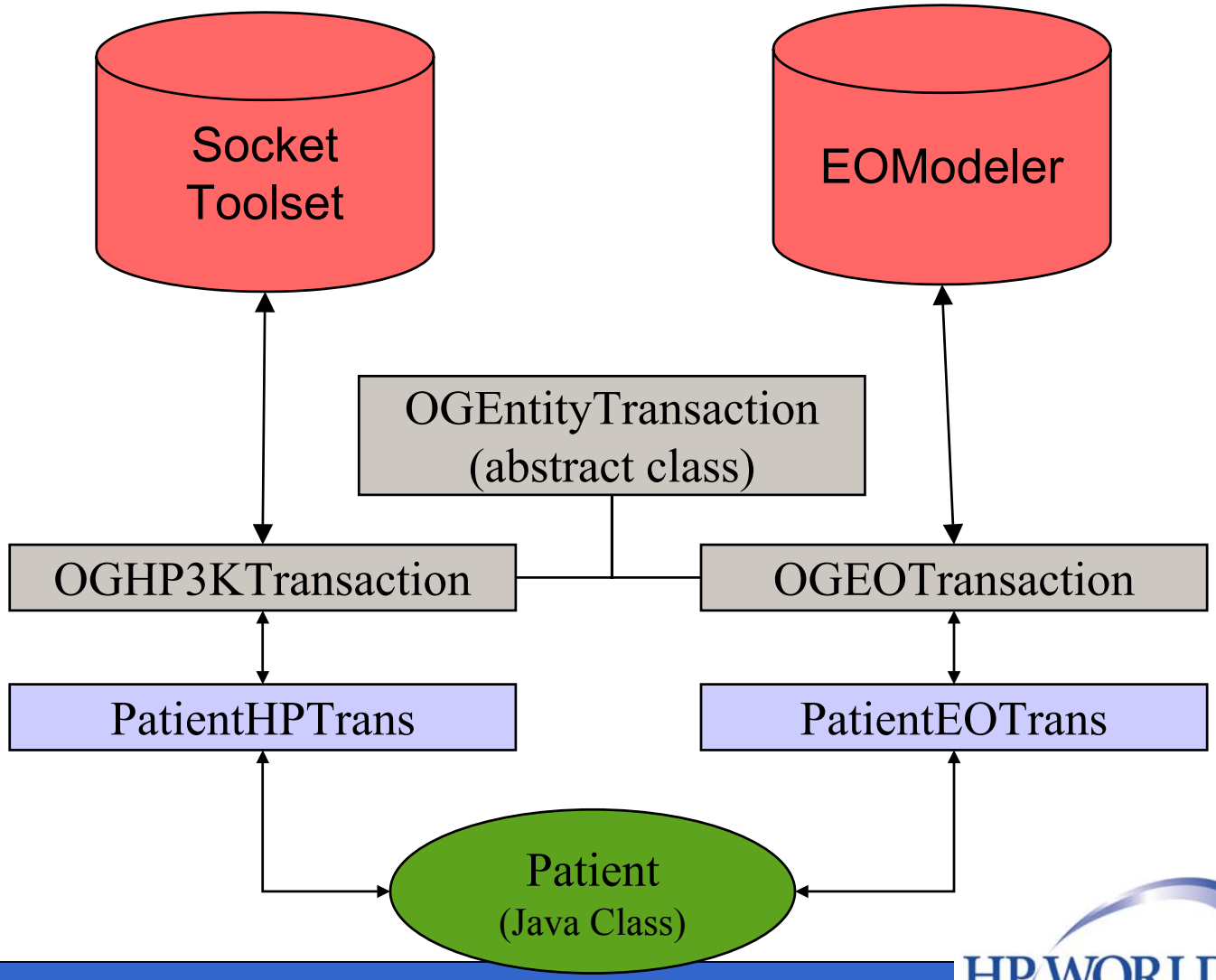
EasyStep Approach – Step 4

GUI Development – WebObjects



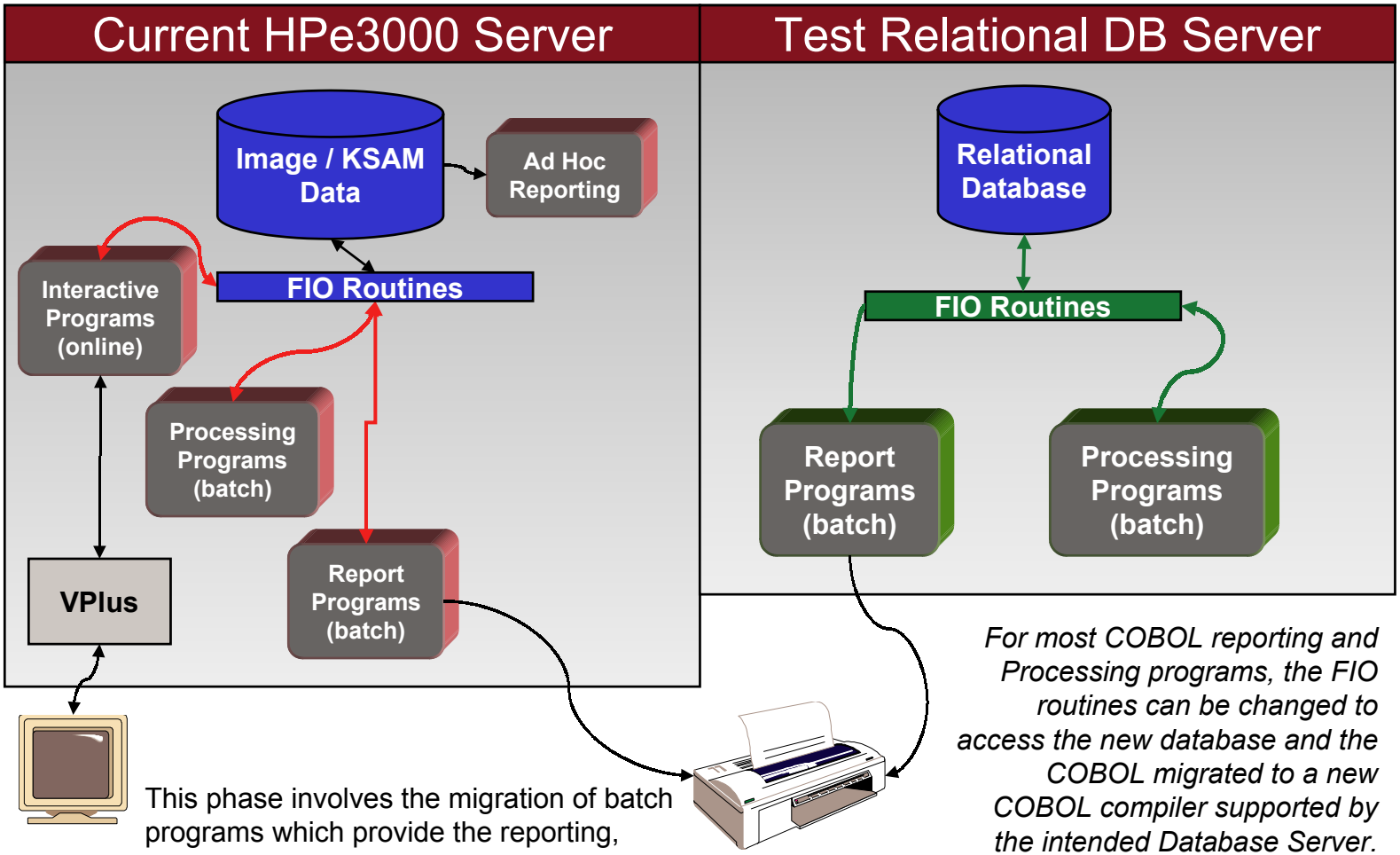
EasyStep Approach – Step 4

GUI Development – WebObjects



EasyStep Approach – Step 5

Batch Program Development



This phase involves the migration of batch programs which provide the reporting, posting or background transaction processing functions of the system. This phase can be done in conjunction with or separate from GUI development.

For most COBOL reporting and Processing programs, the FIO routines can be changed to access the new database and the COBOL migrated to a new COBOL compiler supported by the intended Database Server.

EasyStep Approach – Step 5

Batch Program Development - Options

- Third-party conversion/translator tools
- Migrate using compiler on target platform

EasyStep Approach – Step 5

Batch Program Development –
Using RMCOBOL & ODBC access to Oracle

Replace FIO routines calls with calls to COBOL subroutines containing SQL calls to access Oracle data on HP9000

EasyStep Approach – Step 5

Batch Program Development – FIO high level

[At high level keep same parameter list to minimize changes]

Image:

```
CALL "PATIENT_READ" USING ARDB-LINK, @SEARCH-
        KEY-FIELD, @SEARCH-STRING, @READ-METHOD,
        @LOCK-METHOD, @RECORD-BUF GIVING
        FIO-STATUS.
```

SQL:

```
CALL "PATIENTR" USING ARDB-LINK, SEARCH-KEY-
        FIELD, SEARCH-STRING, READ-METHOD, LOCK-
        METHOD, RECORD-BUF GIVING FIO-STATUS.
```

EasyStep Approach – Step 5

Batch Program Development – FIO low level

Image:

DBGET (base, dataset, mode, status, list, databuffer, argument)

SQL:

SQL PREPARE QUERY sql-QueryHandle,
sql-ConnectionHandle, sql-QrySQL

SQL START QUERY sql-QueryHandle

SQL FETCH ROW sql-QueryHandle

SQL GET DATA sql-QueryHandle, <<field parm list>>

* sql-QrySQL contains select command (e.g., select * from PATIENT where PATIENTNO = search-string)

EasyStep Approach – Step 5

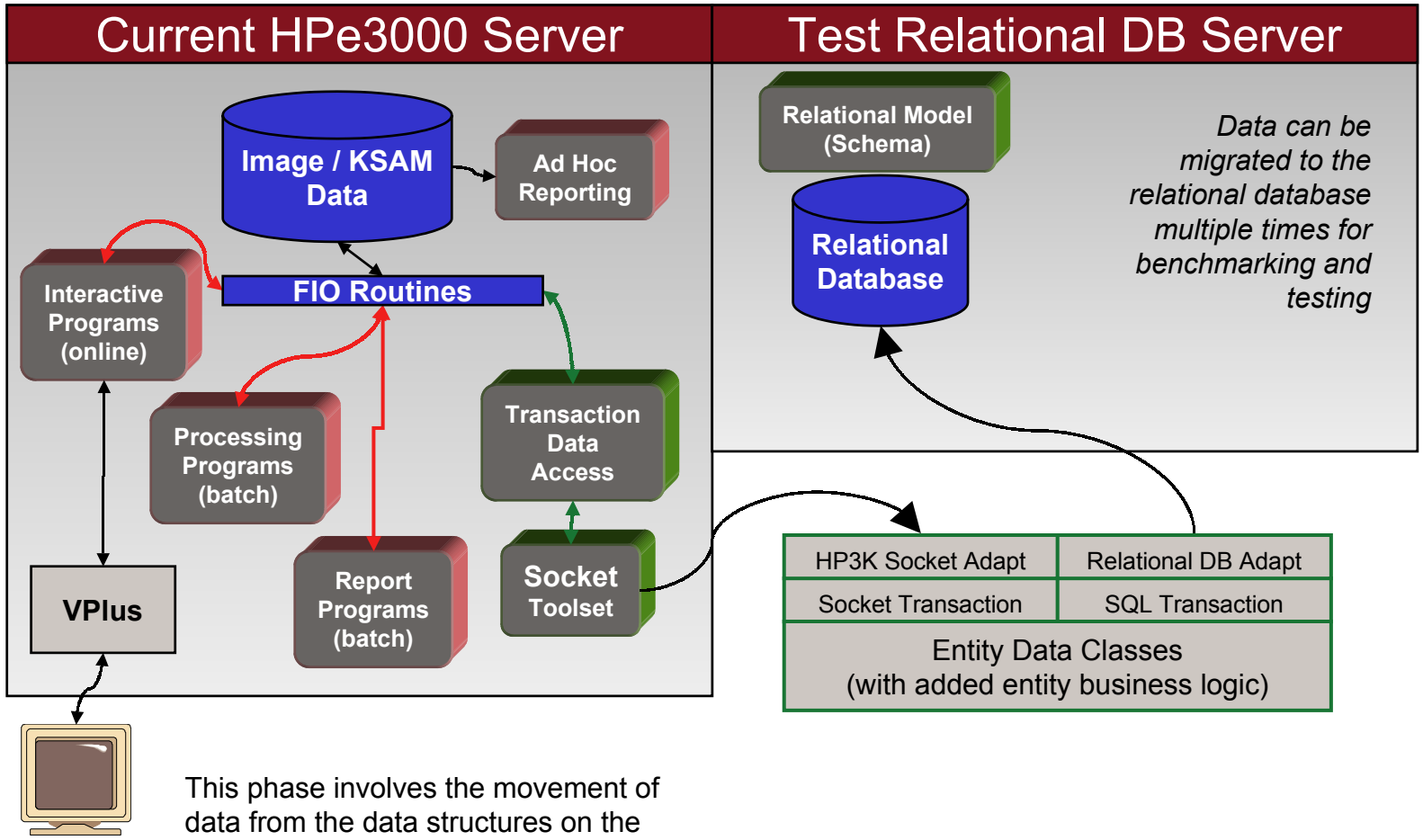
Batch Program Development – Using ProCOBOL & native Oracle access

ORACLE's precompiler for COBOL –

- Accepts source program as input
- Translates the embedded SQL statements into standard Oracle runtime library calls
- Generates a modified source program that can be compiled, linked, and executed in the usual way.

EasyStep Approach – Step 6

Data Migration (Conversion)



This phase involves the movement of data from the data structures on the HPe3000 to the relational database. This migration can occur for specific entities multiple times during the testing process and form the basis of the final conversion of data to the new database.

EasyStep Approach – Step 6

Data Migration (Conversion)

- Use RDBMS tool(s) to create relational database
- Use data access transactions developed in steps 3 and 4 to transfer data from Image to relational database

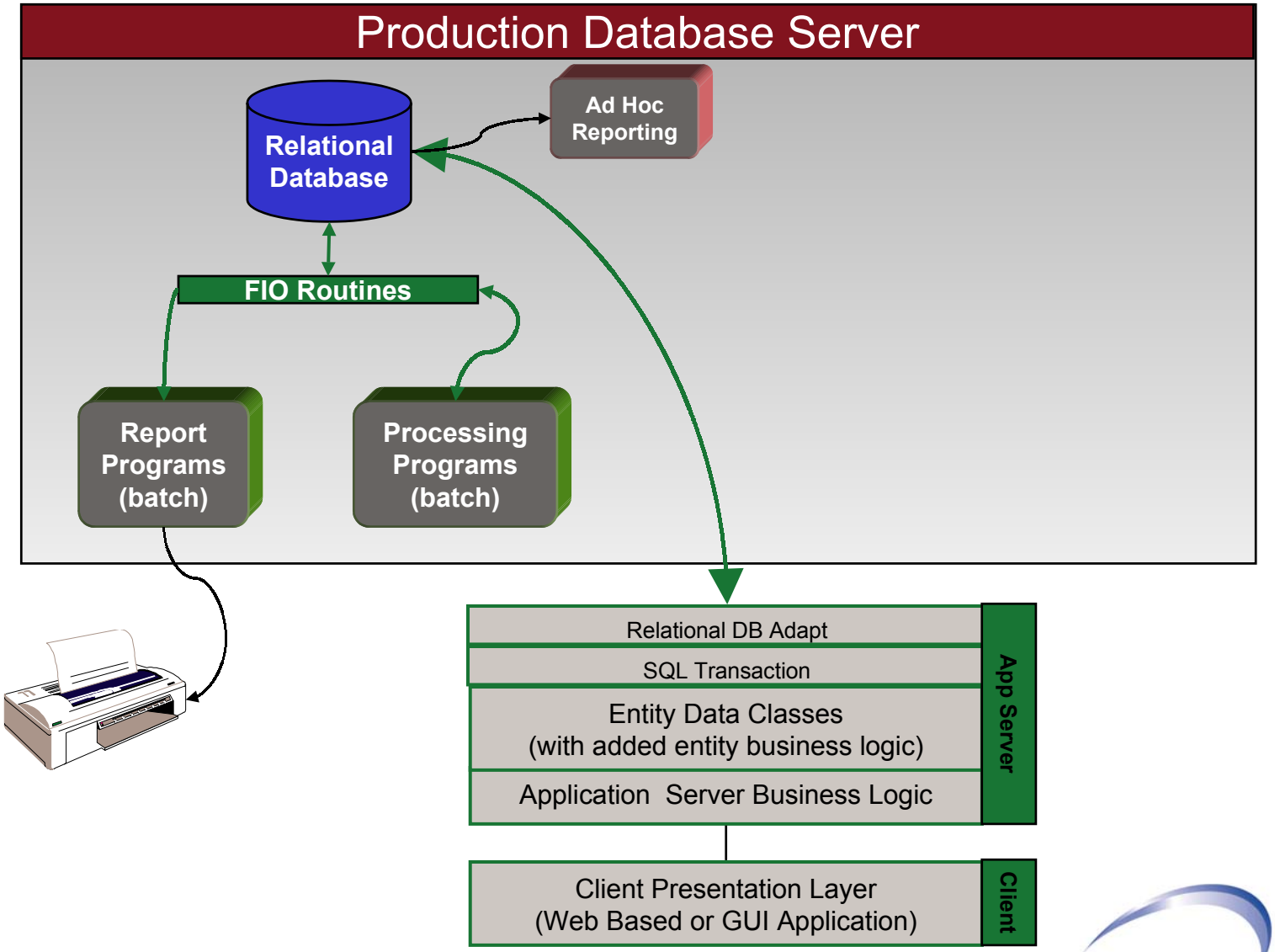
EasyStep Approach – Step 6

Data Migration (Conversion)

- WebObjects EOModeler (Enterprise Object Modeler) to create Oracle table structures based on the relational data model
- Visual FoxPro to transfer data to Oracle [“loop” to read Image data on HP e3000 and write to Oracle database on HP9000 until all entries are read]

EasyStep Approach – Step 7

Final Solution



EasyStep Approach – Step 7

Final Solution

- Interactive programs developed in Visual FoxPro and/or WebObjects
- Batch processes developed in COBOL
- Multiple Oracle Tables
- GUI User Interface

Questions?

Title: **Practical Points on Simultaneously Leveraging and Migrating Image Applications**

Presentation: 037

Authors: Matt Street & Donna Faudree

Company: Orion Group Software Engineers
5770 Nimtz Parkway
South Bend, IN 46628

Phone: (574) 233-3401

E-mail: mstreet@ogse.com
dfaudree@ogse.com