Implementing Multi-Tier Client/Server Systems using Composer

Session 440

Jeff Chancellor
Texas Instruments

Contents

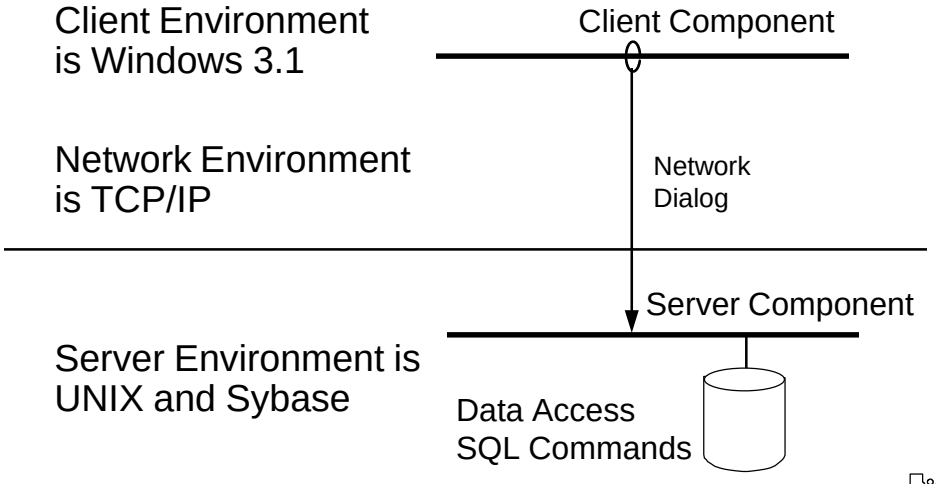
- Comparisons of various client/server styles
- Scenarios and their implementation
- Current Composer practices
- Transitioning to Composer 4
- Summary

Contrasting Styles for Client/Server

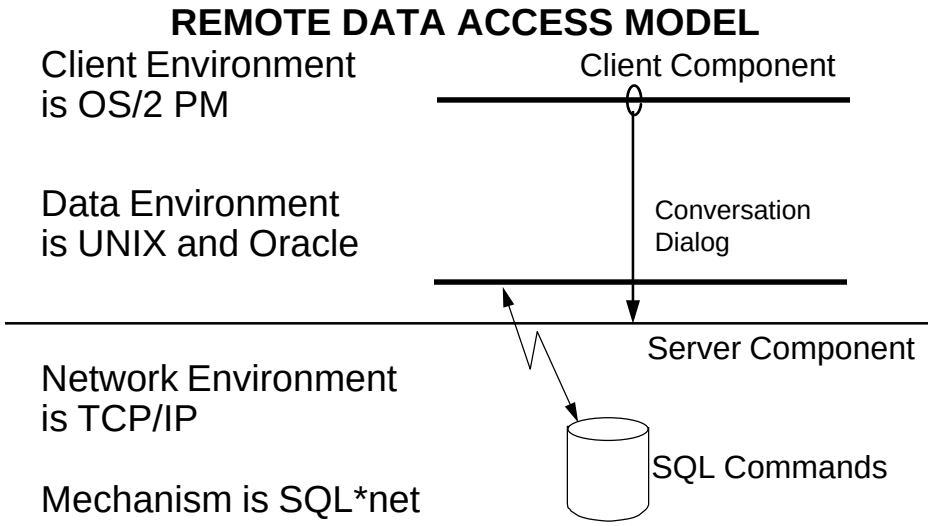
- Distributed Process features
- Remote Data Access features
- Performance differences of RDA versus DP styles
- Combining the two styles in a single application

Contrasting Styles

DISTRIBUTED PROCESS MODEL



Contrasting Styles



Comparing Distributed Process with Remote Data Access

- Network performance
- Transaction performance using a real-life example
  - Projections of performance on group views
  - Projections of performance on single view
- Database connection contention

Performance Differences between RDA and DP using MVS/DB2

	Remote Data	Distributed Process
Transaction Time (Wall Clock/User Time)	2 m 47 s	16 seconds
Application Elapsed Time (DB2 Summary)	2 m 33.2 s	5.4 seconds
DB2 Elapsed Time	3.5 seconds	3.4 seconds
# DB2 Fetches	2724	999
# Selects	473	473
# Packets Transmitted across Network	7796	281

Sample Network Performance Table

			Records Per Second				
Data Length	With Overhead	9600 Baud	14400 Baud	19200 Baud	56000 Baud	16 Mbit LAN	4.5 Mbyte Channels
10	22	55	82	109	318	11364	204545
25	37	32	49	65	189	6757	121622
80	92	13	20	26	76	2717	48913
100	112	11	16	21	63	2232	40179
125	137	9	13	18	51	1825	32847
150	162	7	11	15	43	1543	27778
200	212	6	8	11	33	1179	21226
250	262	5	7	9	27	954	17176
350	362	3	5	7	19	691	12431
500	512	2	4	5	14	488	8789
1800	1812	1	1	1	4	138	2483

Performance Projections using Single Views

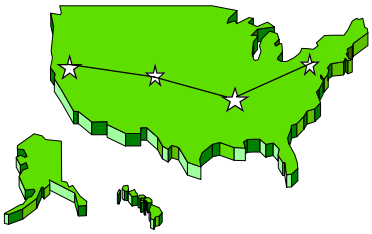
- LAN-based single view fetches are typically equivalent in both Remote Data Access and Distributed Process
  - Isolate the data access component into its own action block
  - Identify the concerns of the “build” if desire is to include the action block on the client package
- WAN-based single view fetches are typically best managed by a Distributed Process style

Performance Projections using Group Views

- LAN-based group views are typically accessed over ethernet or token-ring networks operating at 10Mbps+
  - Time to retrieve 100 cardinality group at 100 bytes/record based on network contention
  - Opening cursor at DBMS server
  - Accessing individual rows requires network hop on RDA versus DP
- WAN-based applications should always process group views in a DP manner

Understanding Bandwidth as a Performance/Partitioning Criteria

- T1 trunks installed between data centers
  - 1.544 Mbps or 172K bytes/sec
  - Not much worry on distributed data access
  - Reduced number of servers is possible
- 56Kbps trunks installed between data centers
  - 6200 bytes/sec
  - Concern increases with greater distributed access
  - Increased number of servers is desireable

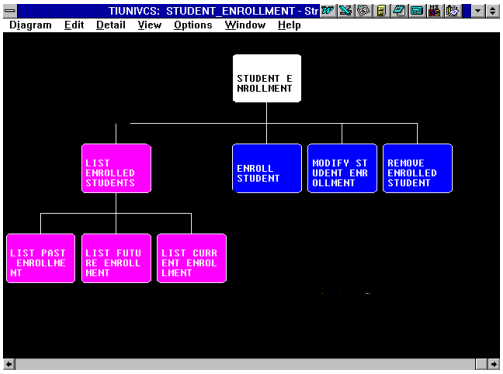


DBMS Contention Considerations

- Remote Data Access requires each application to connect directly to the DBMS
  - Must be concerned with maximum user connections per DBMS kernel
  - May result in costlier DBMS licensing
- Composer Distributed Application connects to the DBMS via the Transaction Enabler/ Teleprocessing Monitor
  - Multiple TE's can connect to the DBMS if required

Partitioning the Application

- Need to define criteria for partitioning
  - Client-side functionality
  - Server-side functionality
  - When there is not a clear reason for one or the other



Sample Scenarios

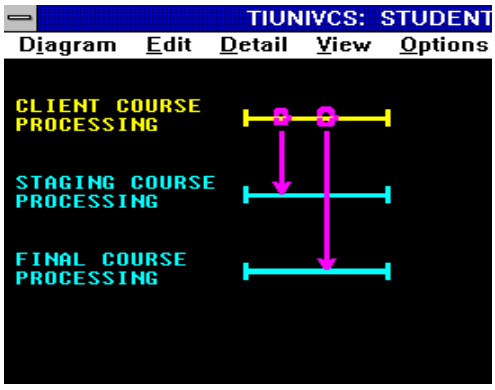
- Rehosting applications off of a mainframe
- Distributed databases supported by client access
- Rehosting applications on a server, but keeping mainframe data on the mainframe
- New applications that need to access/execute existing legacy transactions
- Decision support systems development

Scenario 1: Staging Server for Capacity-Limited Mainframe

- Criteria for investigation
  - High number of invalid transactions reduces capacity of mainframe
  - Invalid transactions cost $$$
  - Easy to replicate/distribute data to validate transaction
- Solution is to “Prevalidate” transactions on a staging server using a lower cost platform

Scenario 1: Solution Example

- Transaction Routing
  - Replicated transaction
    - »Production server
    - »Staging server
  - Different EXIT STATE triggers flow to servers
- Client initiates transaction

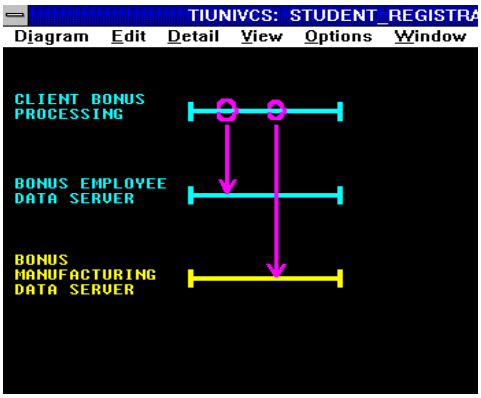


Scenario 2: Distributed Systems

- Criteria for investigation
  - Communication bandwidth prevents WAN access
  - Distribution of data is easily managed
- Solution is to “Push” transactions to each distributed server
- Solution includes the “Push” of data to all servers through DBMS replication mechanism
- Solution allows distributed servers to be sized for regional requirements

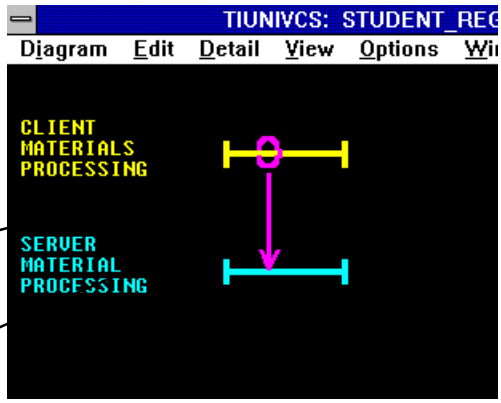
Scenario 2: Solution Example

- Each server includes a transaction envelope
  - Server-managed commits
  - Client-controlled transactions
  - Different EXIT STATES and TRAN CODES support DIRECTORY SERVICES



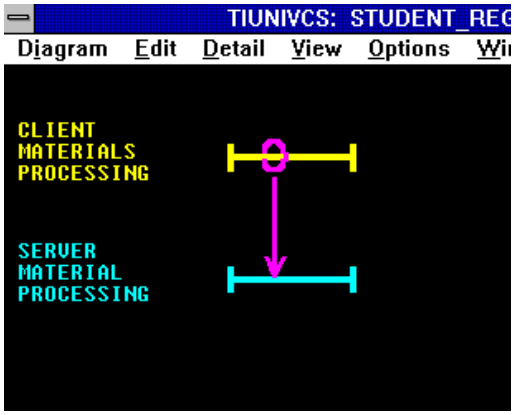
Scenario 3: Distributed Data using Directory Services

- Client Manager USER EXITS permits routing to different platforms
- Action diagram routing (Application Routing)
  - Set NEXT LOCATION feature
  - Same EXIT STATE and TRAN CODE



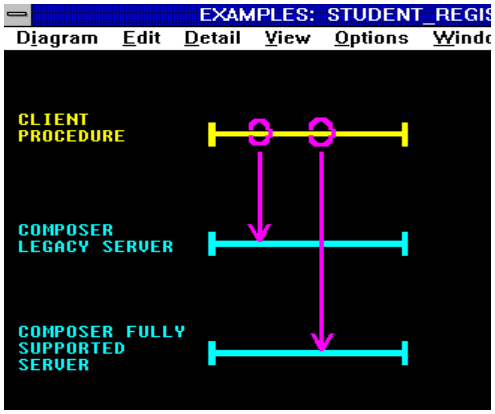
Scenario 4: Moving Processing, but not Data, off the Mainframe

- Recommendation is APPLICATION SERVER
  - Business Logic resides on Application Server
  - Use data gateway to access mainframe data
- No modification to legacy transaction
- No need for “wrapper” procedure



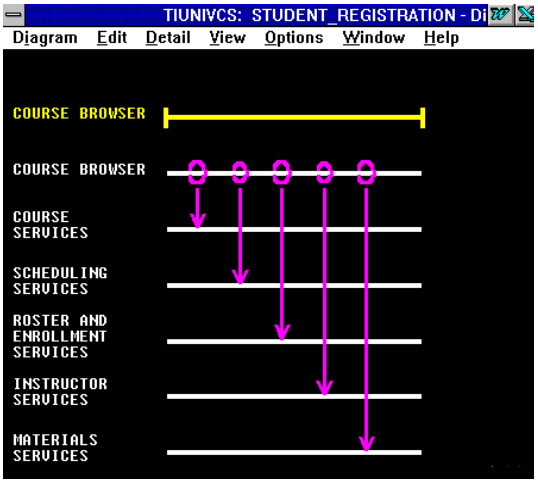
Scenario 5: Using Legacy Transactions in a Composer Distributed Application

- Uses existing transaction on the specified legacy server
- Use Composer online, no display procedure to wrap the transaction
- Use External Action Block (EAB) to interface to legacy transaction



Scenario 6: A Decision Support System Architecture

- Data capture
- Data analysis
- Data preservation
- FAT CLIENT concept
  - Event-driven data capture
  - Large window, multiple data areas



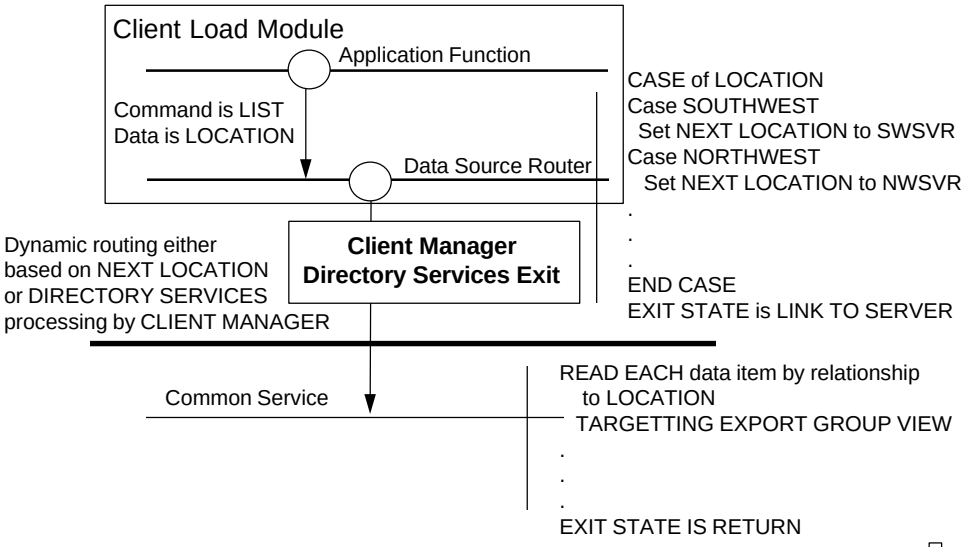
Benefits of Matching Architectures to Applications

- Performance should be optimal
- Adaptability is easily adjusted should architecture require change
- Flexibility is embedded in design
- Able to move “transaction control” to correct point if required to change

Composer 3 Recommendations

- Develop all data access components in separate action blocks
- Develop all reusable components in separate common action blocks
- Partition the application into procedures to match layers of the Technical and Application Architecture
- Package into the specific client/server style for Target Environment

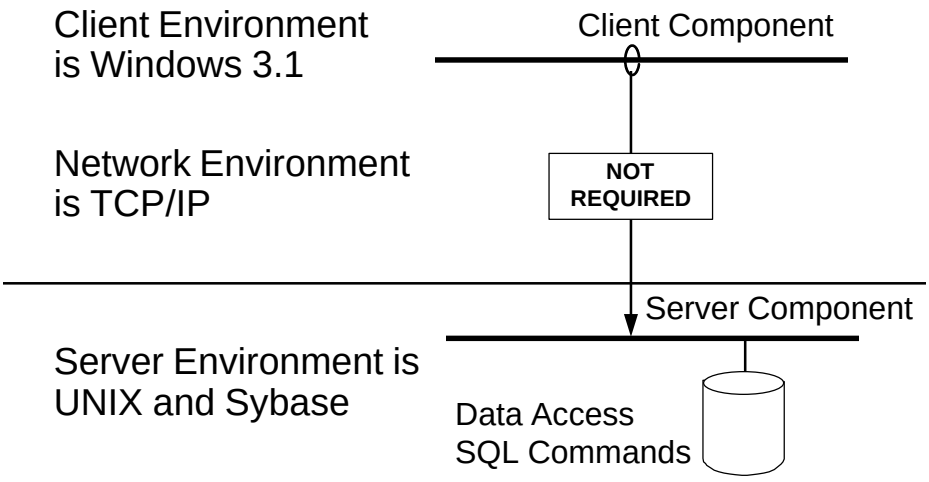
A Composer 3 Sample



Composer 4 Client/Server Features

- Remote USE
  - Cleans up the Dialog Design Diagram
  - Returns to action block after the USE statement
  - Client can “use” any service on any platform
  - Server can “use” other service in a similar transaction environment (e.g., CICS, Tuxedo)

Composer 4 Dialog Design



Transition Strategies for Composer 4 Migration

- Design for Remote Data Access using action blocks
  - Rapid design approach does not require user-designed partitioning through Window Hierarchy Diagram
  - Action blocks moved to procedures very rapidly without having dialog design view matching
- Distribute the application using the Packaging Diagram
  - End up with additional procedures
  - Much enhanced client/server flexibility

Summary

- Distributed architectures are possible *today*
- Performance requires understanding of environment
- Flexibility mandates structured action block usage
- Success requires total plan
- Composer 4 allows for extremely fast client/server design (remote data with Composer 3 today)

Implementing Multi-Tier Client/Server Systems using Composer

Session 440

Jeff Chancellor

Texas Instruments