NonStop Tuxedo
Agenda

- High level description of Tuxedo
- Goals motivating project
- More about Tuxedo
- NonStop Tuxedo on Tandem
- Why buy NonStop Tuxedo
- Project information
- Coexistence and migration
- Summary
What is Tuxedo

Tuxedo is a leading open transaction monitor providing a framework for Enterprise Transaction Processing

- Consistent application programming interface for client/server communication
- OLTP performance characteristics
- Configuration and management of OLTP applications
- Distributed transaction processing, hiding underlying network protocols
Important TUXEDO Attributes

• Supports multiple environments
  – Workstation front ends (DOS, Windows, OS/2, Mac, Unix)
  – Networks of Unix system servers
  – Proprietary (IBM) servers

• Modularly constructed so an application can:
  – Plug in DBMS systems
  – Plug in front-ends
  – Plug in network interfaces
  – Plug in various hardware platforms

• Implements standards and drives emerging standards
  – XATMI
  – Tx
  – XAP-TP (OSI-TP) (Future)
  – TxRPC (Future)
Goals Driving NonStop TUXEDO Project

• Provide real Tuxedo APIs and FAPs
  – ATMI API and network protocols
  – Portability of programs and programming skills
  – Heterogeneous application interoperability
  – Standards compliance with XATMI, TxRPC

• Exploit Tandem differentiation
  – Availability, scalability, networking, manageability
  – PATHWAY, TMF, NS Kernel

The same application runs better on Tandem systems.
TUXEDO Components

Terminals

Workstations

DES/MIO

GW

/T

/WS

GW

/HOST

/D

/Q

/DOMAIN

Application Servers

IBM CICS

Other Tuxedo Applications and non-Tuxedo (XATMI) Applications
TUXEDO Component Descriptions

• /T Core services, Application Transaction Monitor Interface (ATMI)

☒ DES/MIO Data Entry System - terminal handler
• /WS Client-Side ATMI for workstations

☒ /D OLTP DBMS

☒ /HOST Communication to MVS/CICS
• /Q Queued messages
• /Domain Communication to separately administered appls
  – /TDomain - TUXEDO to TUXEDO

☒ /OSI-TP - TUXEDO to TP system over OSI-TP

☒ We are not currently pursuing these components
Core Services of Tuxedo /T

**Client Modules**

TPBEGIN
TPCALL "X"
TPCOMMIT

TPBEGIN
TPCALL "Y"
TPCOMMIT

TPBEGIN
TPCALL "X"
TPCALL "Z"
TPCOMMIT

TPBEGIN
TPCALL "Z"
TPCOMMIT

**Tuxedo System/T**

- Name Server
- Load Balancing
- Data Dependent Routing
- Buffer Management
  - Transparent Data Conversion
- Communications
  - Hides Network Protocols
- Transaction Control
- Configuration and Administration
- Monitor and Manage Running System

**Server Modules**

PROCESS REQUESTS "X", "Y"

PROCESS REQUESTS "X", "Z"
ATMI Highlights

- Demarcate transactions
- Call synchronously and block while waiting for reply
- Call asynchronously and do more work while waiting for reply
- Establish a conversation with a service
  - Open a connection
  - Send and receive multiple messages
  - Close the connection
- Reliably queue request for future execution
- Buffer management
- Services
ATMI and Tx Verbs

• Insert Page from MS Word Here
ATMI Conversation Example

Client

\[ \text{cd} = \text{tpconnect("svc1", data \ldots TPSENDONLY}); \]
\[ \text{tpsend( cd, data \ldots )} \]
\[ \text{tprecv( cd, &buf, \ldots &event );} \]
\[ \text{tprecv( cd, &buf, \ldots &event );} \]
\[ \text{tprecv( cd, &buf, \ldots &event );} \]
\[ /* \text{event} = \text{SVCSUCC} */ \]

Server

\[ \text{svc1( svcinfo )} \]
\[ \text{tprecv( svcinfo->cd, &buf,\ldots,&event );} \]
\[ /* \text{event} = \text{SENDONLY} */ \]
\[ \text{tpsend( svcinfo->cd, data \ldots )} \]
\[ \text{tpsend( svcinfo->cd, data \ldots )} \]
\[ \text{tpreturn( TPSUCCESS, data \ldots )} \]
NonStop TUXEDO Externals

- Full TUXEDO API
  - ATMI (superset of X/Open's XATMI)
  - C and COBOL bindings
  - X/Open's Tx API
  - FML and VIEW functions (80+ calls)
  - Integration with I18N
  - Miscellaneous utility functions
- Tuxedo application administration tools
  - tmloadcf, tmboot, tmconfig, tmadmin, etc.
- /WS and /DOMAIN interoperability
- TUXEDO development tools
  - buildclient, buildserver, mkfldhdr, viewc, viewdis, etc.
NonStop TUXEDO and TSA

Application

ATMI, Tx APIs

Tuxedo Added Value
- Name Server
- Buffer Mgmt
- Data Dependent Routing

Pathsend, TMF, NS Kernel APIs

TSA
- Link Mgmt
- Load Balancing
- Transaction Control
- Process Monitoring
- System Mgmt
- Dialog Servers

The same application runs better on Tandem systems.
Architecture

- Multiple CPUs in one Logical Machine
- BB, BBL in each CPU
- Spans nodes
- Pathway servers
- Pathway manages processes
- No process pairs (other than Pathmon)
- No TMS
- No Bridge
Porting Attributes

• Expose Tandem fundamentals
• Performance similar to native calls
  – TPCall ≈ Pathsend
  – No extra messages
• Future releases readily move to Tandem
  – Well defined interfaces between OS dependent and independent code
  – Novell incorporates interface changes into base product
Why Would a Tandem Customer Use NonStop TUXEDO?

- Openness and standards prevent vendor lock in
- Interoperability with other systems (including Integrity)
- Tuxedo value added
  - Logical service names
  - Location independent naming
  - Data dependent routing
  - Buffer management
    - Automatic data conversions
  - Single service can span Tandem nodes
  - Reliable queued messages for long operations
Why Should a Non-Tandem Customer Use NonStop TUXEDO?

- Can use the leading open Transaction Monitor
- Can use the best OLTP platform
  - Scalability
  - Availability
    - No single point of failure
    - NonStop availability, 7 x 24 x Forever
  - Manageability
    - Single log per system
    - Easier/automatic tuning characteristics
    - Load balancing over multiple CPUs
    - Richer system management tools
  - Process Management
    - Process restart after CPU failure
    - Better failure detection
Why Should a TUXEDO Customer Switch to NonStop TUXEDO?

• Because they can!!!!
• NonStop Tuxedo removes another road block preventing an application from moving to the best OLTP platform
  – Scalability
  – Availability
  – Manageability
  – Process management

The same application runs better on Tandem systems.
Project/Product Phases

- Prototype the chosen design approach (done)
- Release 1 - One Tuxedo system, many Tandem nodes and cpus
- Release 2 - Tuxedo interoperability between Tandem & "other" Tuxedo domains
NonStop Tuxedo - Release 1

- Single, NS Kernel domain – no non-Tandem nodes
- ATMI (superset of XATMI) – client, server
- /WS, WMIO clients
- PATHWAY server classes, TMF transactions
- Tuxedo Administration tools
- C and COBOL API - clients and servers in both languages
- Unix (including Integrity) to NS Kernel via /WS
- NS Kernel to Unix via /WS
- No /Q, no /DOMAIN, no interoperability
NonStop Tuxedo - Release 2

• /Domain support
  – /TDomain - Tuxedo to Tuxedo (possibly not on Tandem) over Tuxedo FAP
• Interoperability with non-NS Kernel Tuxedo Domains
• Heterogeneous Transactions (needs Open TMF)
• /Q support

• Outplan items
  – TxRPC (lack of funding)
  – /OSI-TP Domain - Tuxedo to TP system over OSI-TP (requires OSI-TP services)
Open TMF

• Heterogeneous Transaction
  – Import Transaction - Tell TMF to be subordinate
  – Export Transaction - Include Gateway in 2-phase commit
  – Logging of Gateway Info
  – Recovery Service

• Follow on to TMF3
NonStop Tuxedo Platform

- RISC only, (need multiple active extended segments)
  - CLX/R, Cyclone/R, Himalaya
- D30 based, (Various dependencies, including semaphores)
- OSS clients and servers
  - Pathway to support OSS servers
Coexistence - Positive

- Tandem client can both Pathsend to Pathway servers and ATMI to Tux Servers
- Similarly for WS clients using RSC and ATMI
- Pathway servers can TPCall to Tux Servers
- Tux Servers can Pathsend to Pathway server
- Transactions flow across both sends and TPCalls
Coexistence - Negative

- Cannot Pathsend (or SCOBOL send, or RSC send) directly to Tux Server
- Cannot TPCall directly to a Pathway server
- WS client cannot do both RSC and TPCall under same transaction
Coexistence - Translation Servers

- Can translate an old-style send to a TPCall
- Converts DDL described message into Tuxedo Buffer (FML or VIEW)
- Can be used for migration
- Terminal support is a requirement
- Simple translation server will be provided
Migration

- Can write servers in anticipation of Tuxedo
- Will require less that 10% of code to be converted
- Can provide simple examples
Summary

• NonStop Tuxedo will give Tandem users an Industry standard Transaction monitor
  – Standard APIs for portability (programs and skills)
  – Standard FAPs for interoperability
  – Additional capabilities for Tandem OLTP applications
• NonStop Tuxedo is built on top of Tandem's core OLTP products
  – Pathway, TMF, NS Kernel
• NonStop Tuxedo inherits Tandem Fundamentals
  – Scalability, Fault Tolerance, Availability, Manageability . . .

The same application runs better on Tandem systems.