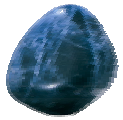
Pure Java Transaction Management for Tomorrow's Enterprise Applications

James G. Lynn

August 24, 2001

Agenda

- What is transaction processing?
  - Transaction ACID properties
  - Distributed transactions
  - JTA, JTS, J2EE and OMG
- Configuration Features
- Complex Transaction Features
- Standards

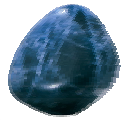


What is transaction processing?

- In the simplest terms:
  - A request for a service of some kind with immediate confirmation or denial back to the requester. In between the request and response, resources (e.g. files, databases) are read and updated as required

- Distributed Transaction Processing
  - Transactions which consist of reads and/or updates to various resources spread over several systems and/or databases

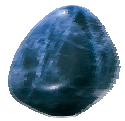
Software technology to assure complete, accurate business transactions



Transaction ACID properties

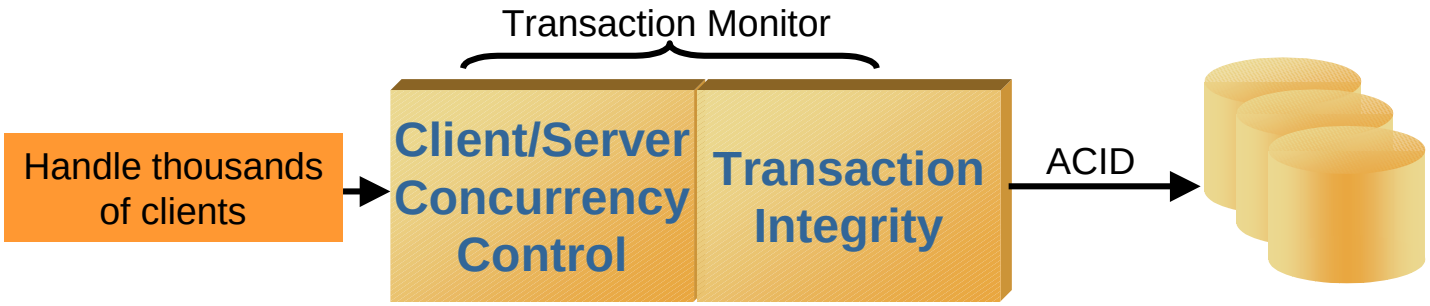
- Atomicity
  - the transaction completes (commits) or if it fails (aborts) then all effects are undone (rollback)
- Consistency
  - transactions produce consistent results
- Isolation
  - intermediate results are not visible, and transactions appear to execute serially even if done concurrently
- Durability
  - the effects of a committed transaction are never lost

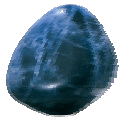
The result of a transaction must be predictable and stable



World of Transactioning

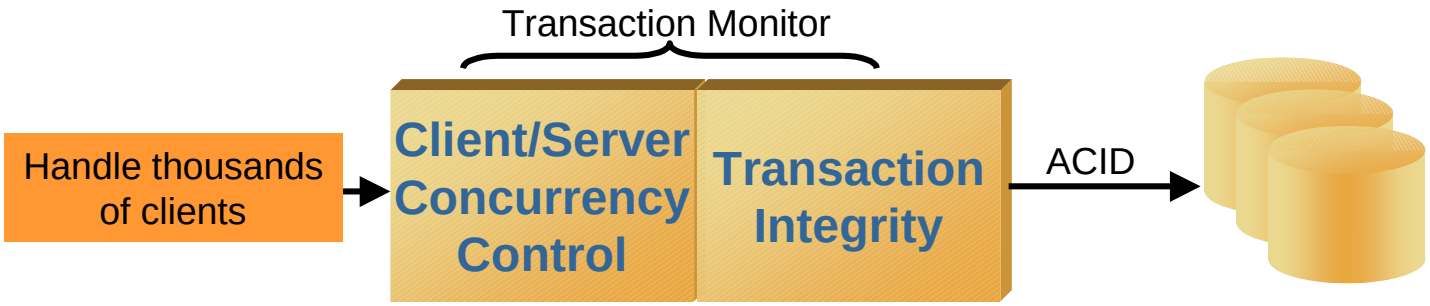
- Traditional transaction systems



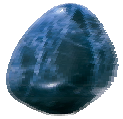


New World of Transactioning

- Traditional transaction systems



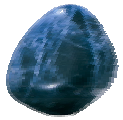
- New Transaction Systems



Marketplace

- Financial and Telecommunications markets
  - Banking
  - Insurance
  - Mobile services
- ISV’s
- Other J2EE vendors without a JTS

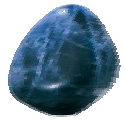
- Sophisticated end users
  - Application component builders
  - Application service integrators



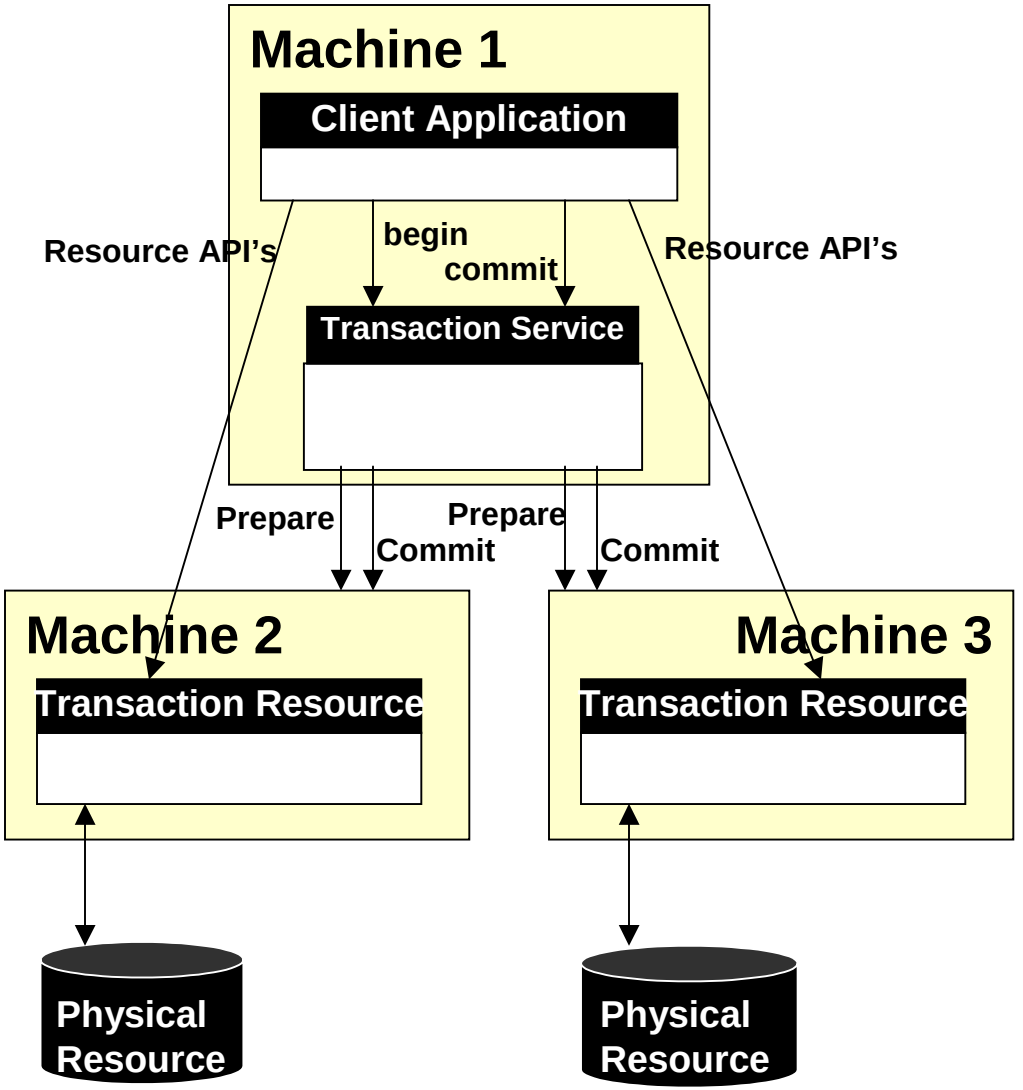
Why do we need it?

"Midtier application server companies have to gain transaction skills or risk being left behind. ... there will be less and less reason to buy application servers and transaction monitors separately—as well as less and less reason to buy application servers without transaction services."

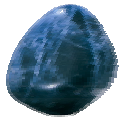
"App Servers vs. Transaction Monitors",
Timothy Dyck, July 24, 2000.



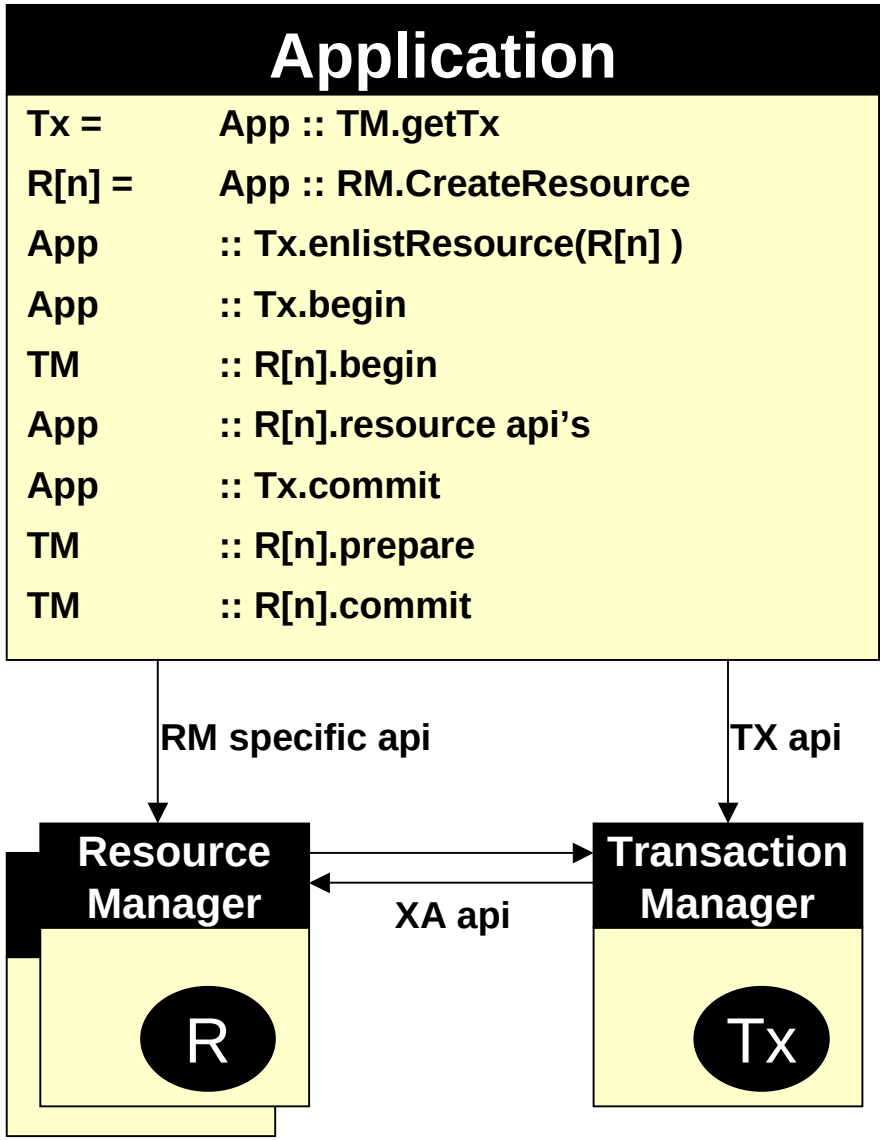
Distributed Transactions



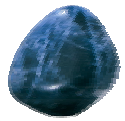
- Transactions can span:
  - machines
  - domains
  - software languages



Distributed transaction process model

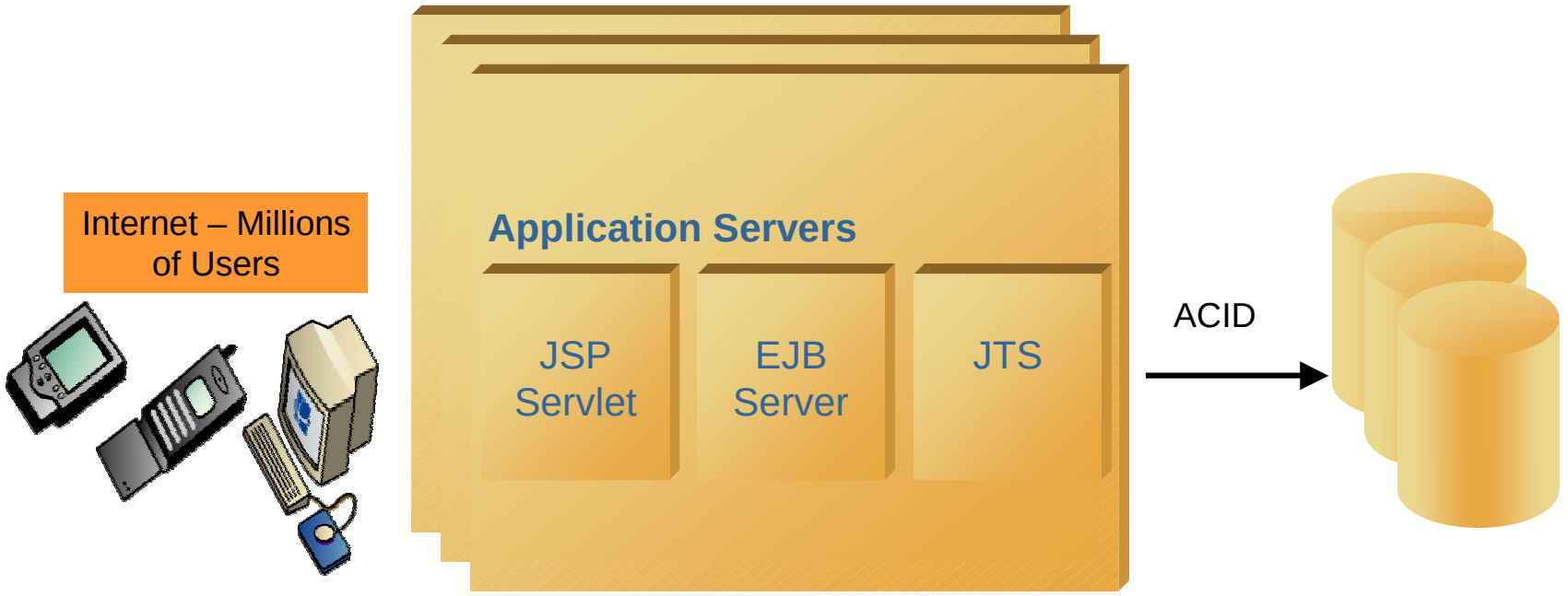


- X/Open and OSI define the DTP model
- Defines the basics of transaction processing
  - Supports ACID properties
  - Defines two phase commit (2PC) protocol



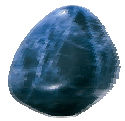
J2EE model for Transaction Integration

New model for App Server – Pure Java JTS

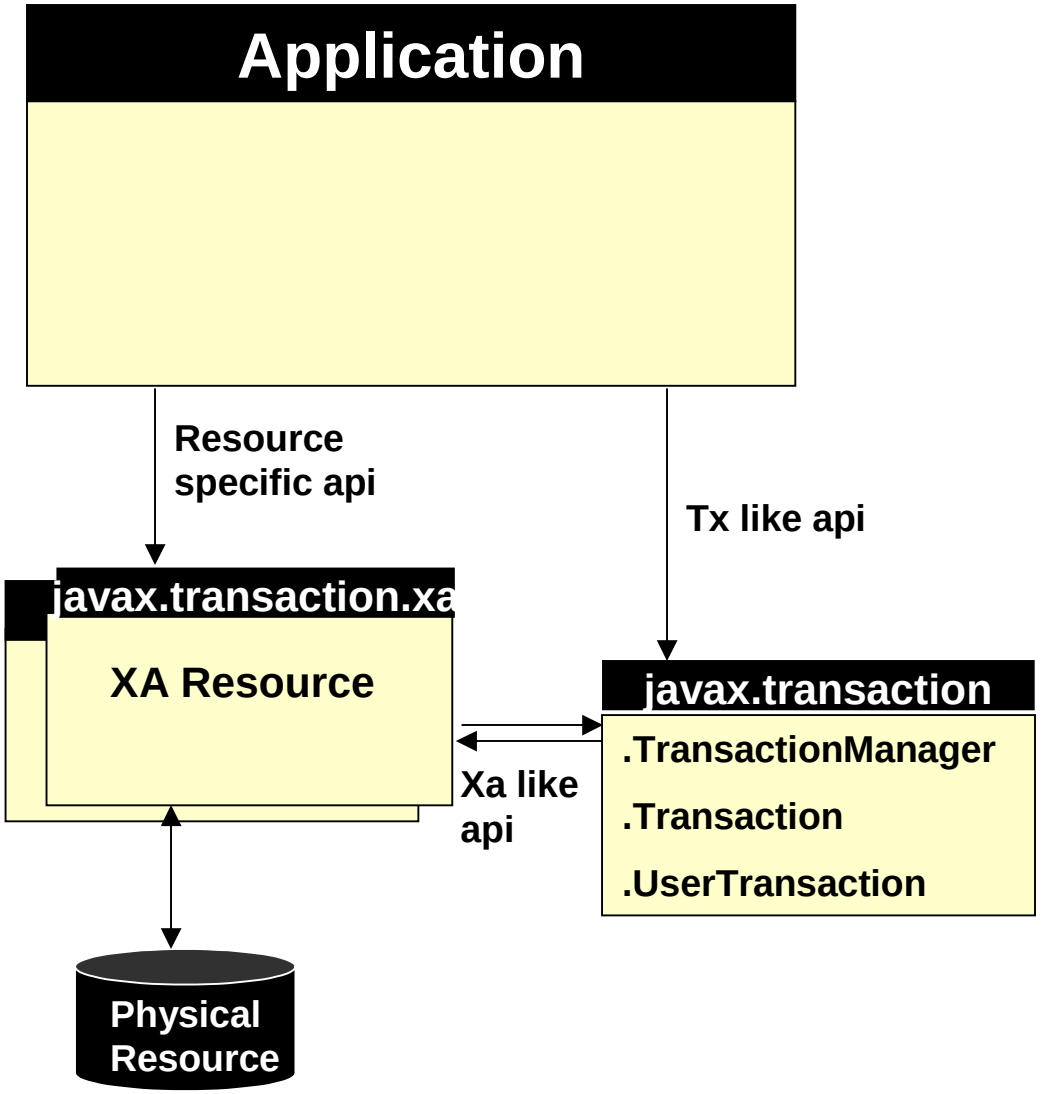


What is Java Transaction Service?

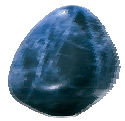
- JTS is an implementation of a Transaction Manager
- JTS implements
  - Java Transaction API (JTA) 1.0 Specification
  - Java mapping of the OMG Object Transaction Service (OTS) 1.1 Specification
- JTA is a required part of J2EE
- JTS is an optional part of J2EE and EJB *today…*
- A JTS Transaction Manager provides transaction services to the parties involved in distributed transactions



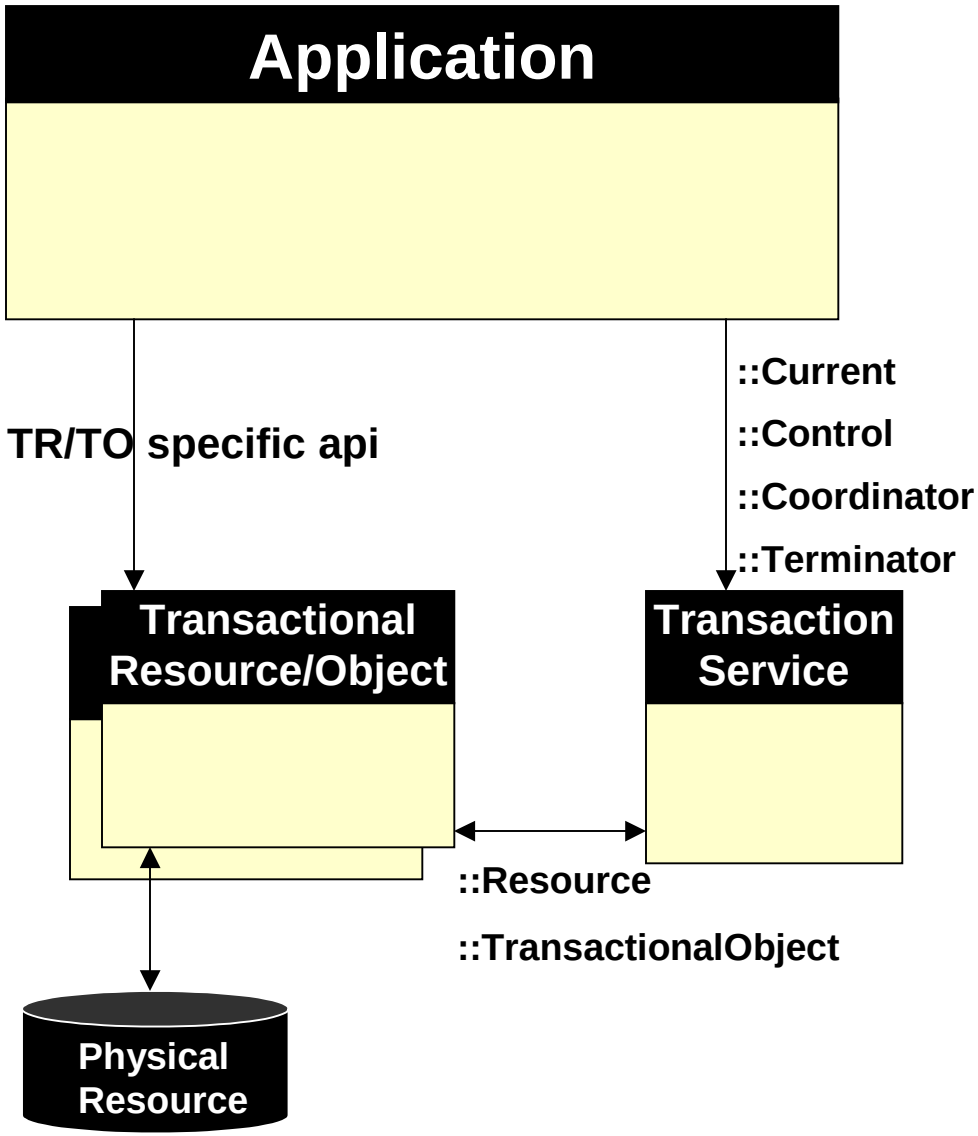
Java Transaction API (JTA)



- Sun Microsystems specification
- Required for J2EE
  - Gives easy API to J2EE developers
- XA architecture
  - Similar to XA
  - Supports XA compliant resources

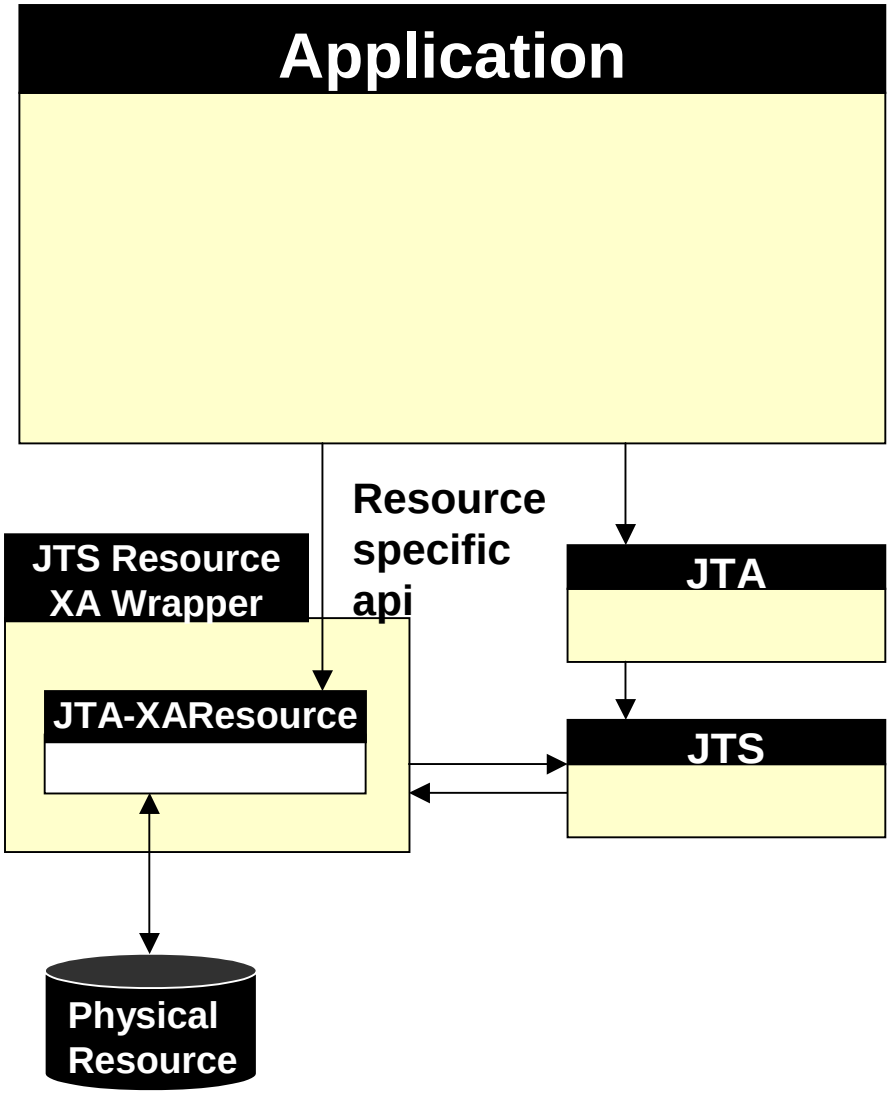


Object Transaction Service (OTS)

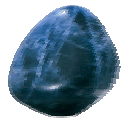


- OMG defines standards for object transaction service
- Standard provides IDL (interface definition) for transactions
- Language neutral specification
- Specifications
  - OTS 1.1 Released
  - OTS 1.2 May 2000

JTA to JTS

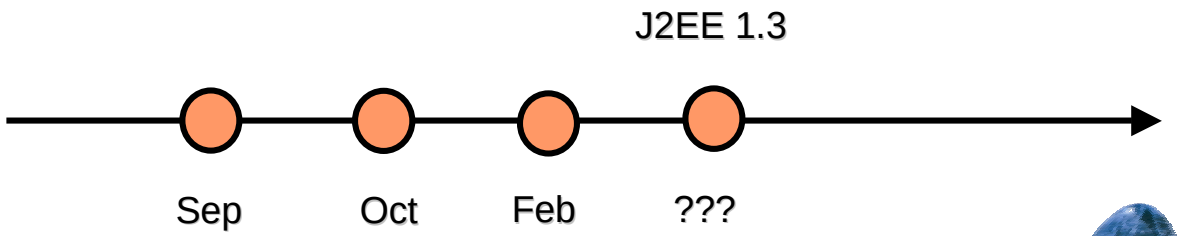


- JTA implemented via JTS (an OTS mapping)
- Mature and proven software technology
- Offers benefits of JTS to JTA
  - distributed transactions
  - nested transactions

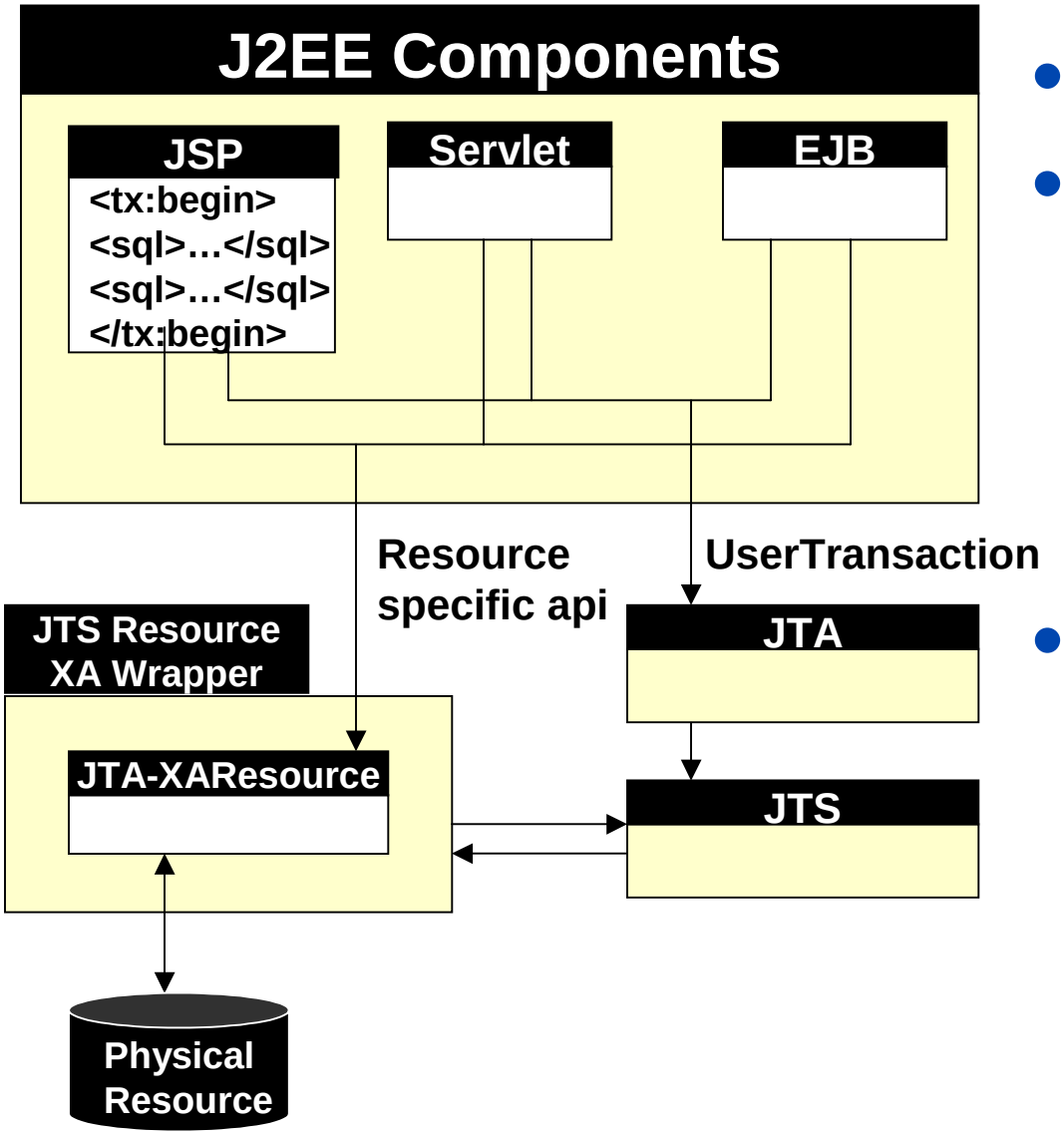


J2EE 1.3

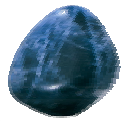
- Sept 2000 — First 1.3 source drop
- Oct 2000 — Specification for proposed final
- Feb 2001 — Beta 1.3 (source, binary, CTS)
  - API feature complete
- ??? 2001 — GA



J2EE transaction platform



- J2EE requires a JTA
- Transactions can be started by
  - J2EE components
  - J2EE application client
- Transactions can be propagated from one J2EE platform to other J2EE platforms

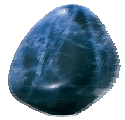


Java vs. Other Implementations

…a commercial pure Java transaction service, an essential requirement for e-commerce, especially wireless e-commerce.

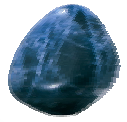
…an object based model that is highly proprietary and based on older technology… is not highly scaleable, nor is it easily integrated.

Hurwitz Trend Watch – 7/13/00

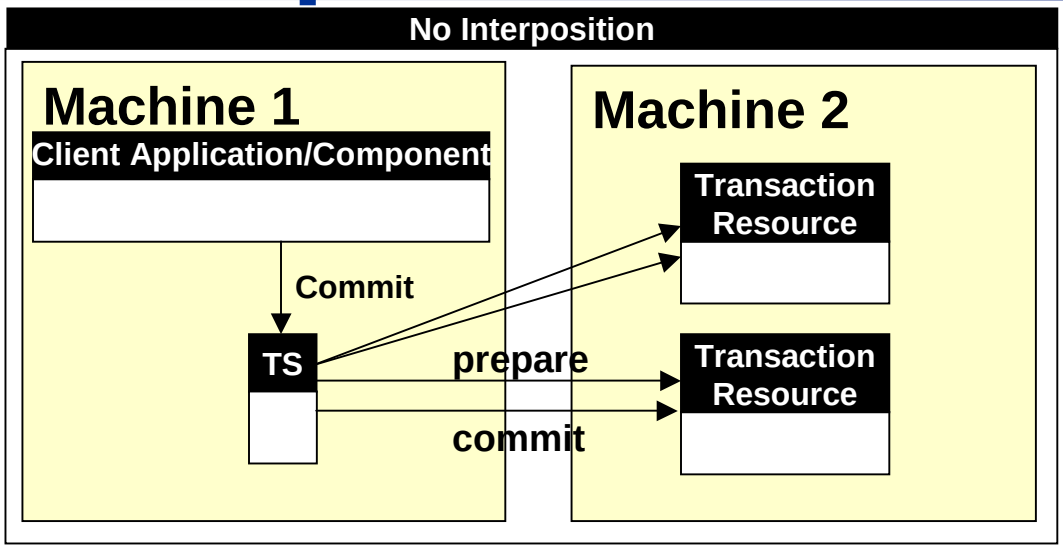


Agenda

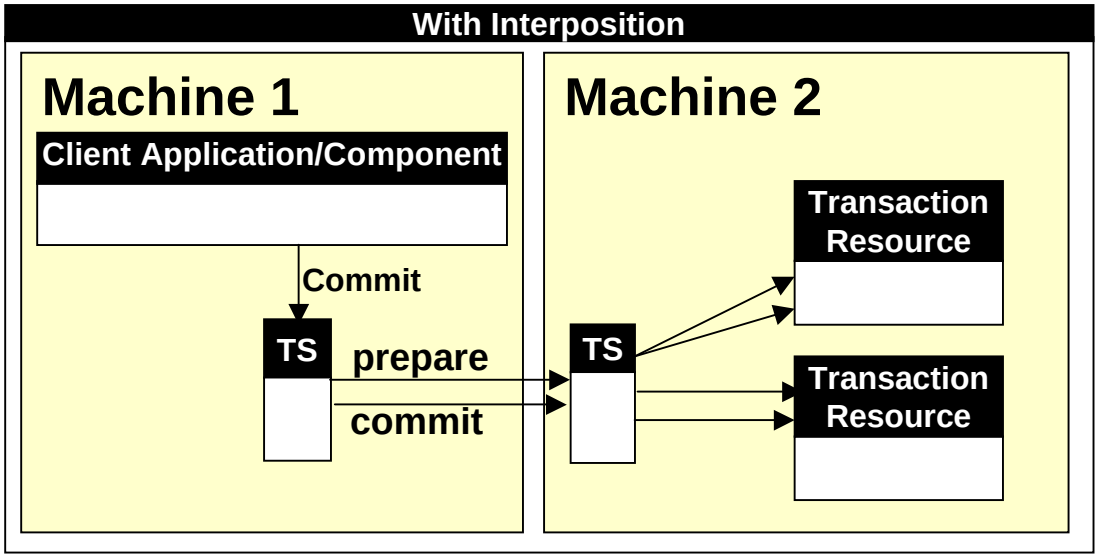
- What is transaction processing?
  - Transaction ACID properties
  - Distributed transactions
  - JTA, JTS, J2EE and OMG
- Configuration Features
- Complex Transaction Features
- Standards



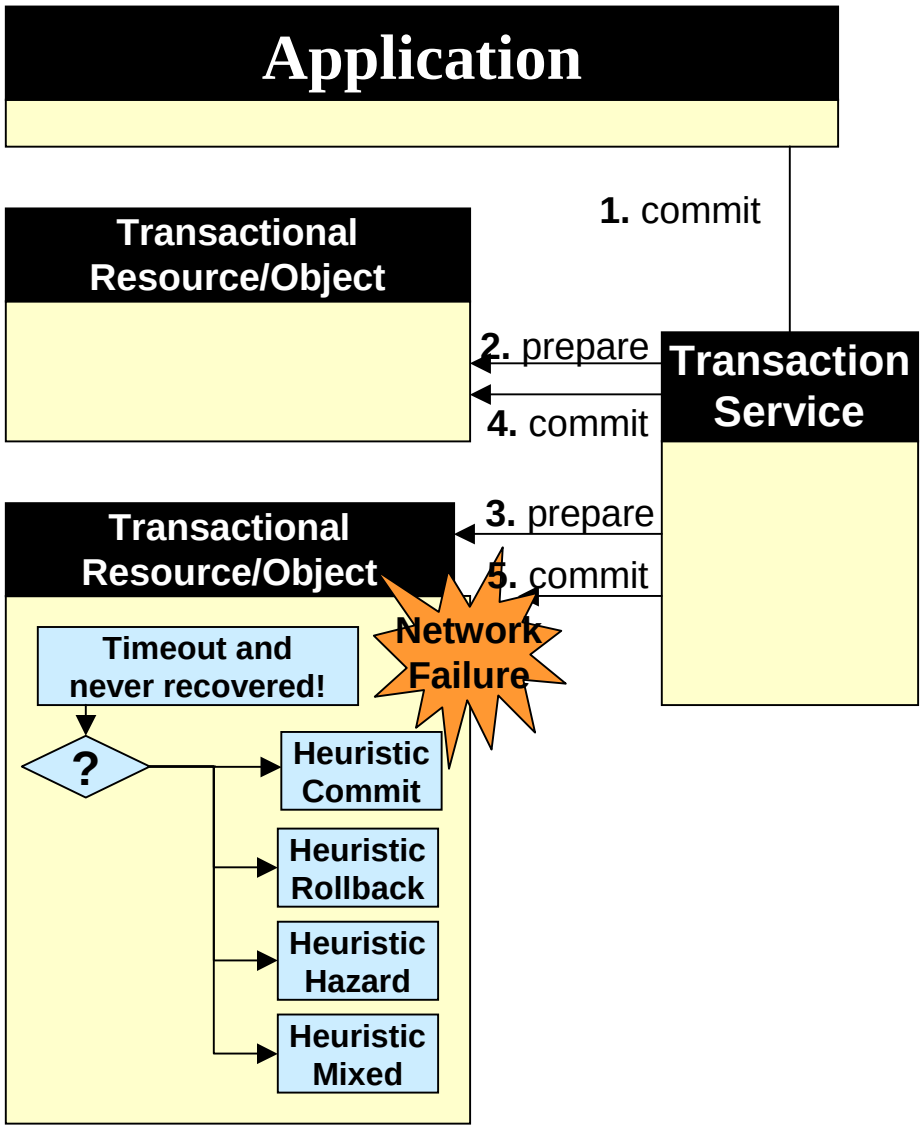
Interposition



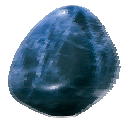
- Reduces network resources
- Optimized orchestration of 2PC



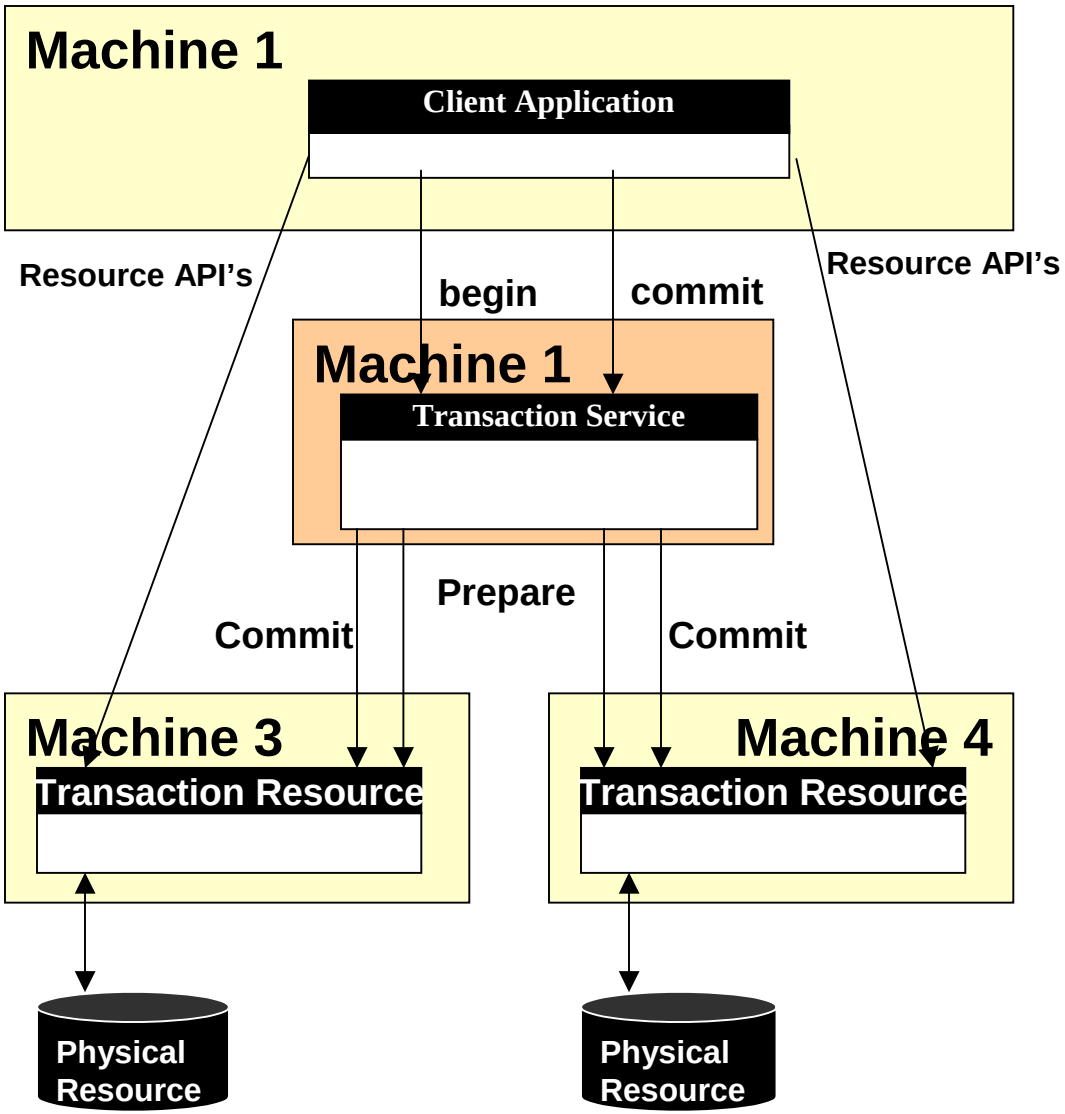
Transaction Heuristics



- Independent transaction completion
- Available for unusual circumstances (e.g. network failure)



Transaction Manager Server Model



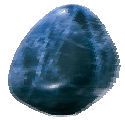
- Transaction service can run standalone
- Runtime model is configurable

Check/Unchecked behavior

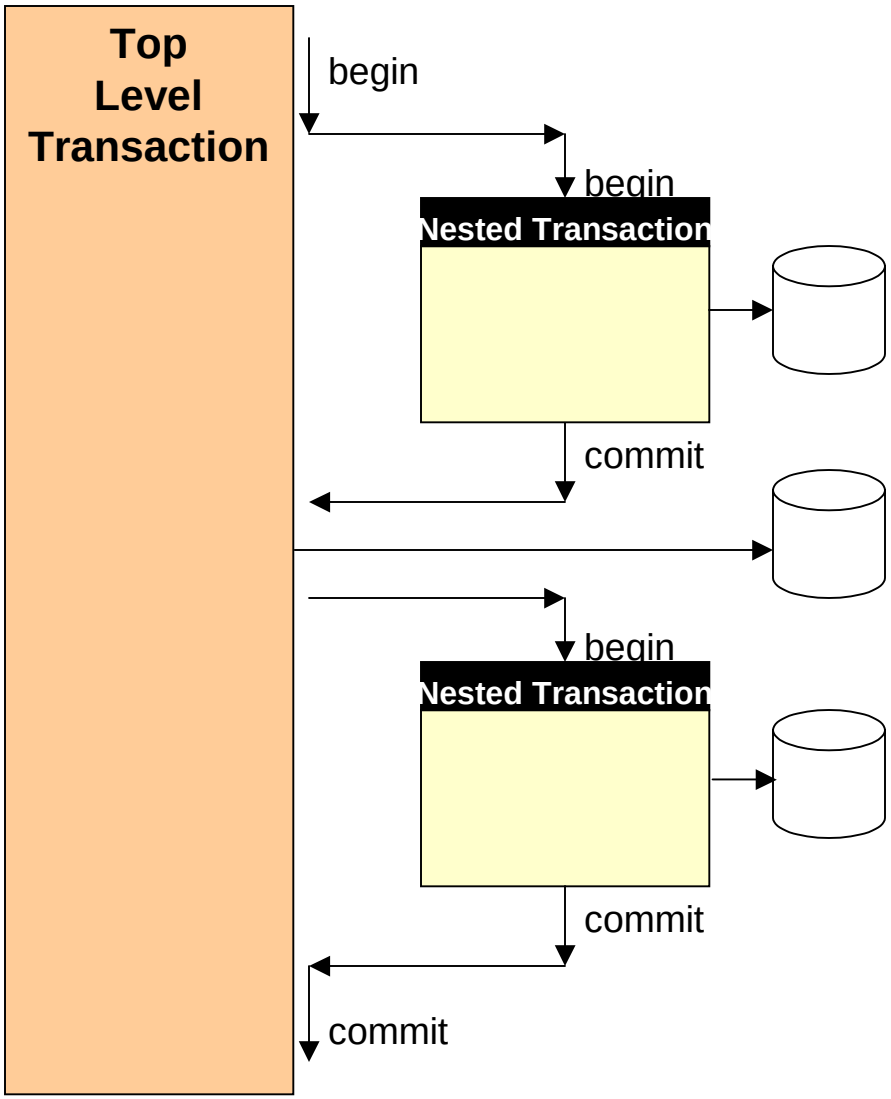
- Transaction originator is the only able to commit transaction
- Transaction commits only after all transactional objects have completed requests
- These may be configurable

Agenda

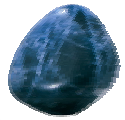
- What is transaction processing?
  - Transaction ACID properties
  - Distributed transactions
  - JTA, JTS, J2EE and OMG
- Configuration Features
- Complex Transaction Features
- Standards



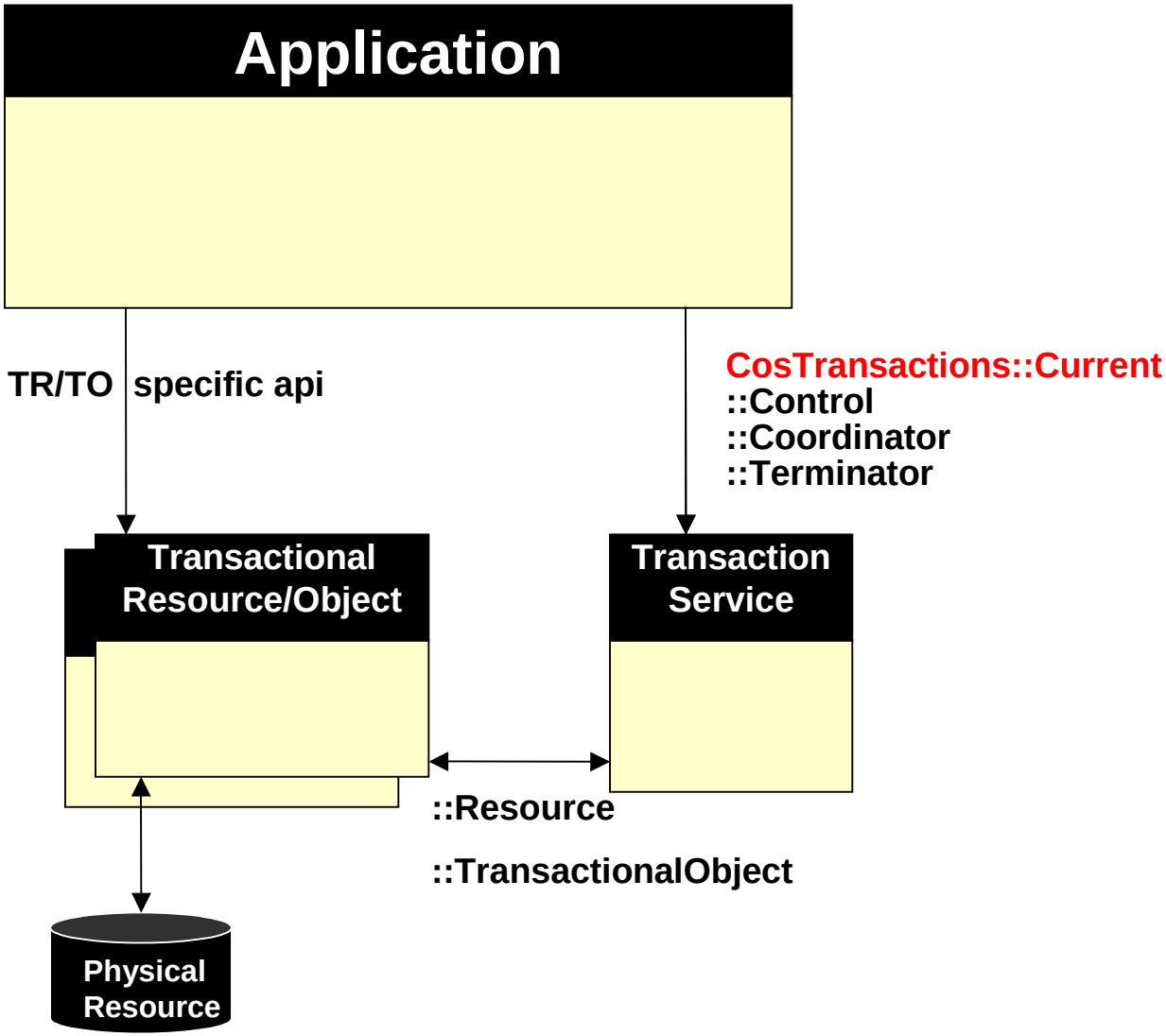
Nested Transactions



- Nested transactions are supported
- Nested Transactions with 2PC are supported
- Nested transactions allow for pieces to complete without hurting the entire transaction

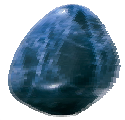


Transaction's Current

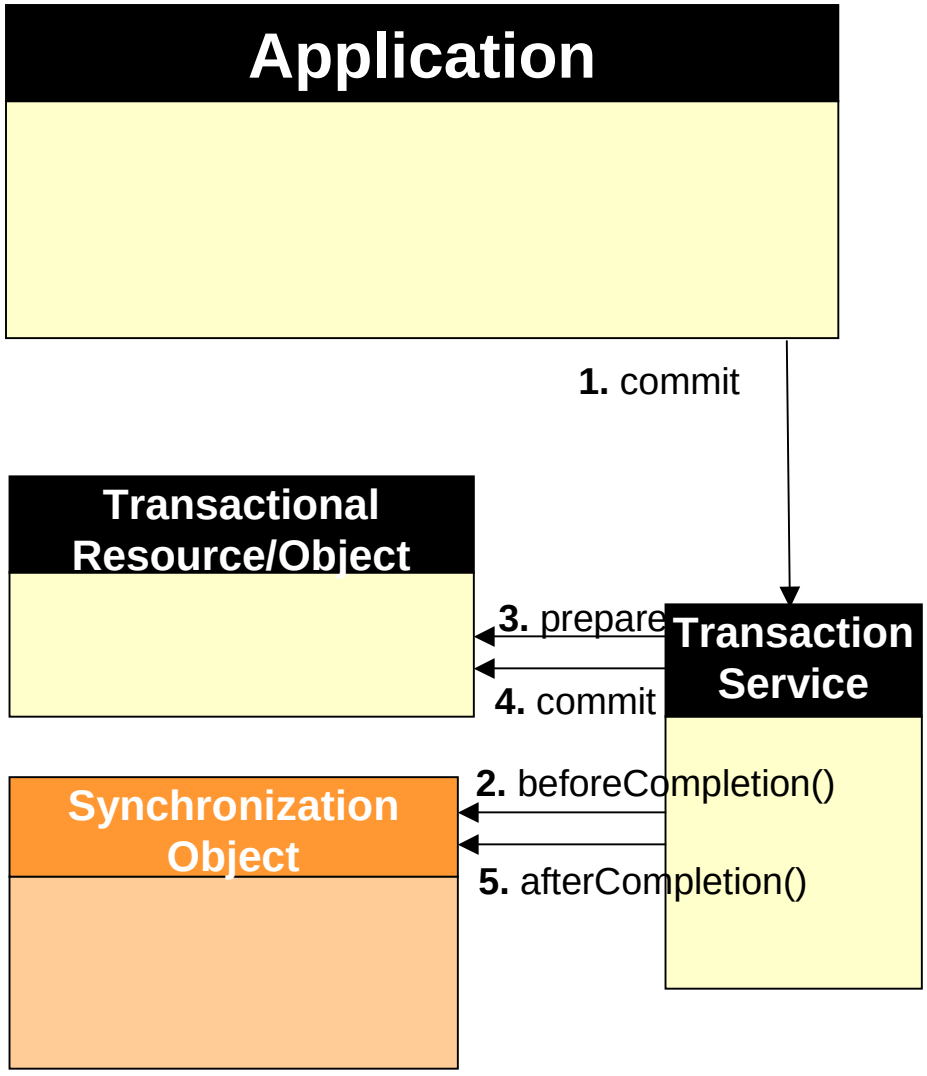


Direct/Indirect management

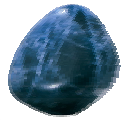
- Direct
  - The developer uses the following services to work the transaction
    - Control
    - Coordinator
    - Terminator
- Indirect
  - Transaction control is done through the Current object
  - Similar to using the JTA where the transaction control and creation is abstracted from the user



Synchronization object

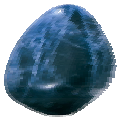


- Allow objects to monitor transactions
- Interface supplies methods
  - beforeCompletion( )
  - afterCompletion( )
- Can be utilized for notifications when transactions commit



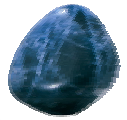
Implicit/Explicit propagation

- Explicit propagation
  - Transaction propagated as parameter for method
  - Programmer must implement
- Implicit propagation
  - Transaction propagated by system with transactional objects
  - JTS responsible for ensuring propagation

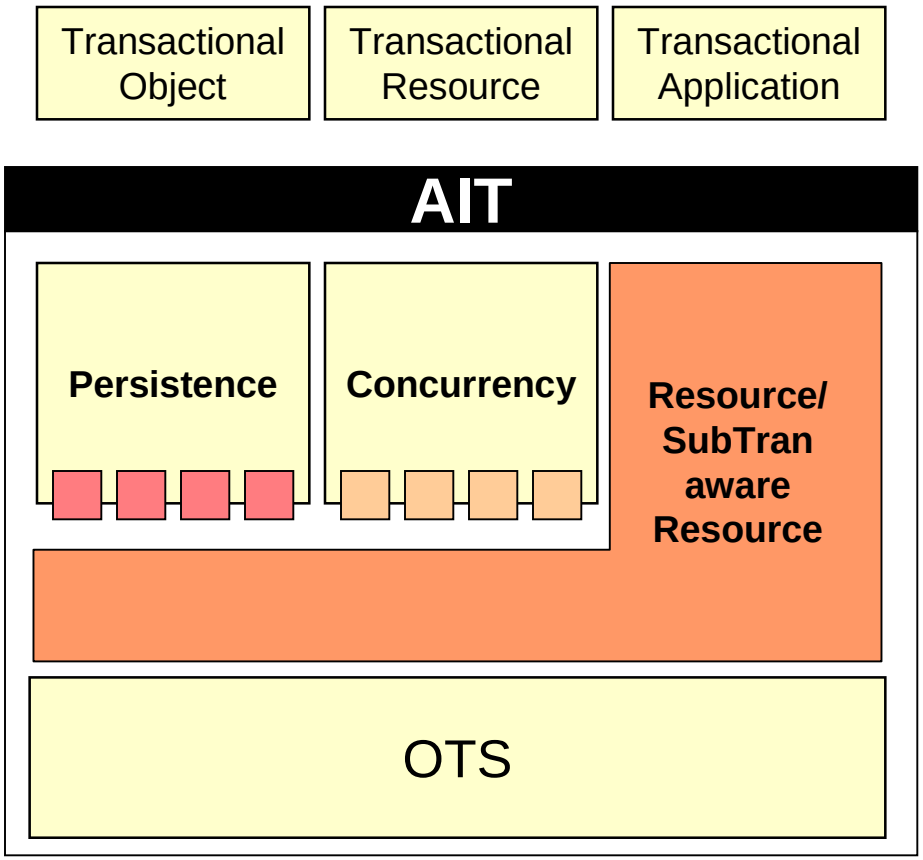


Multi-threaded aware

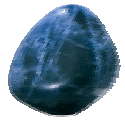
- Allows for transactions to participate across multiple threads
- JTS implementation is thread safe



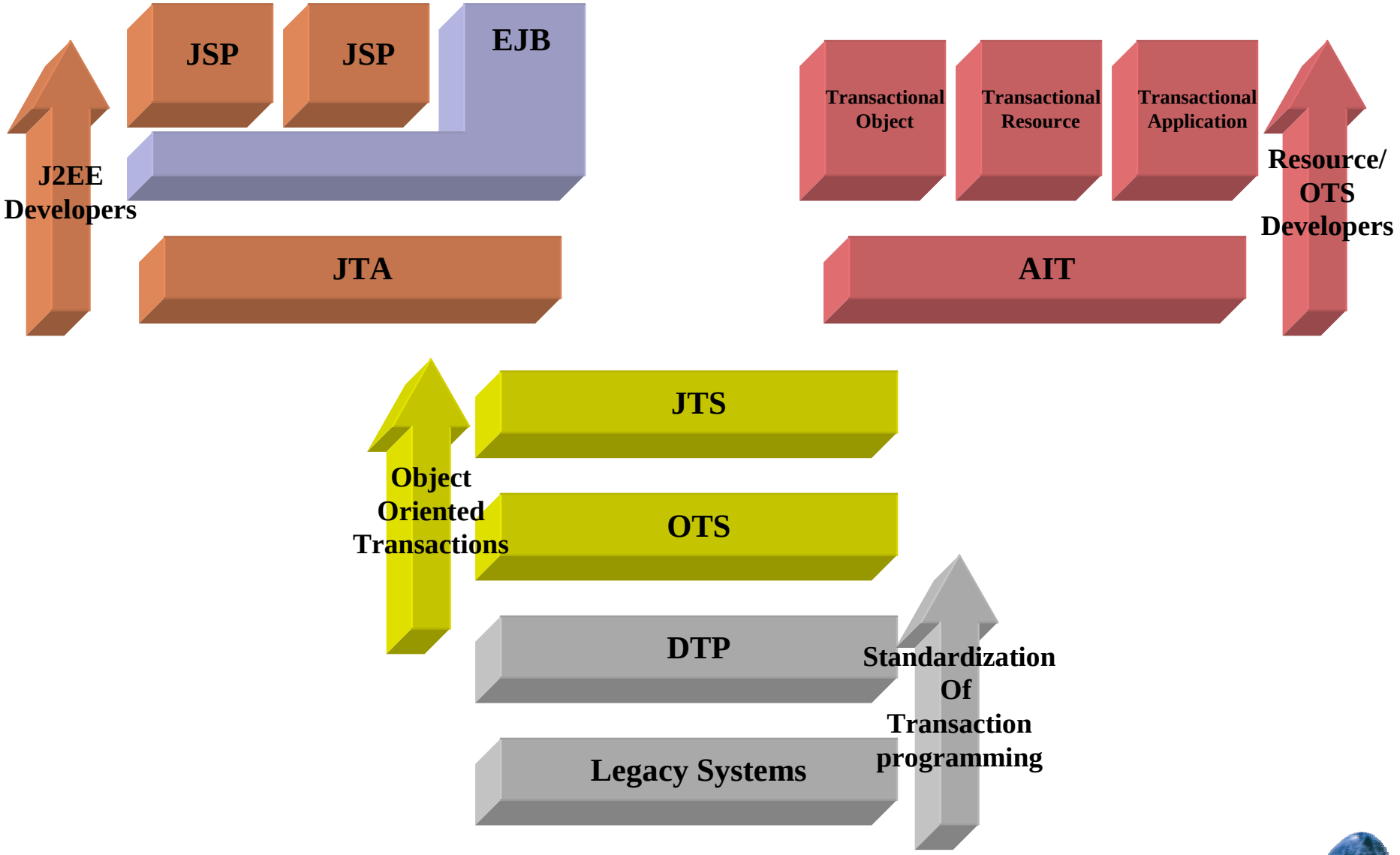
Advanced Integrated Transactions (AIT)



- Complete framework for developing transactional applications and components easily
- Provides concurrency control, persistence and crash recovery
- Provides interfaces and implementations for persistence and concurrency
- Provides level of abstraction above raw OTS programming

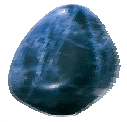


Enterprise Transaction Programming

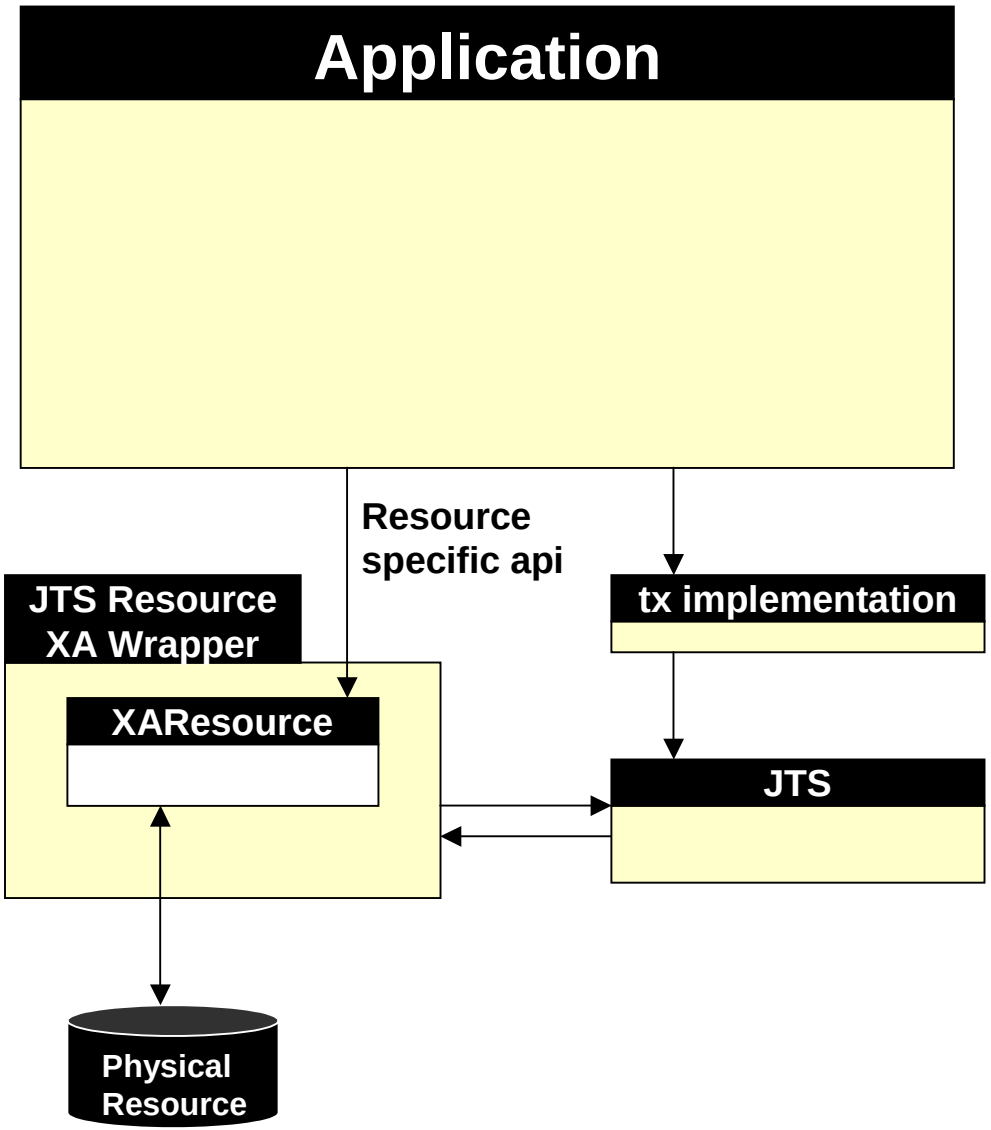


Agenda

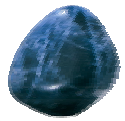
- What is transaction processing?
  - Transaction ACID properties
  - Distributed transactions
  - JTA, JTS, J2EE and OMG
- Configuration Features
- Complex Transaction Features
- Standards



XA compliance

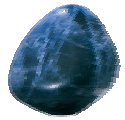


- JTS provides Tx layer compliant with XA
- Supports XA resources



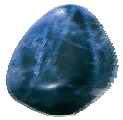
JDBC Support

- JDBC 1.0
  - XA Wrapper for drivers is provided
  - Resources cannot participate in 2PC
- JDBC 2.0
  - Supports drivers
  - Supports XA resources and 2PC

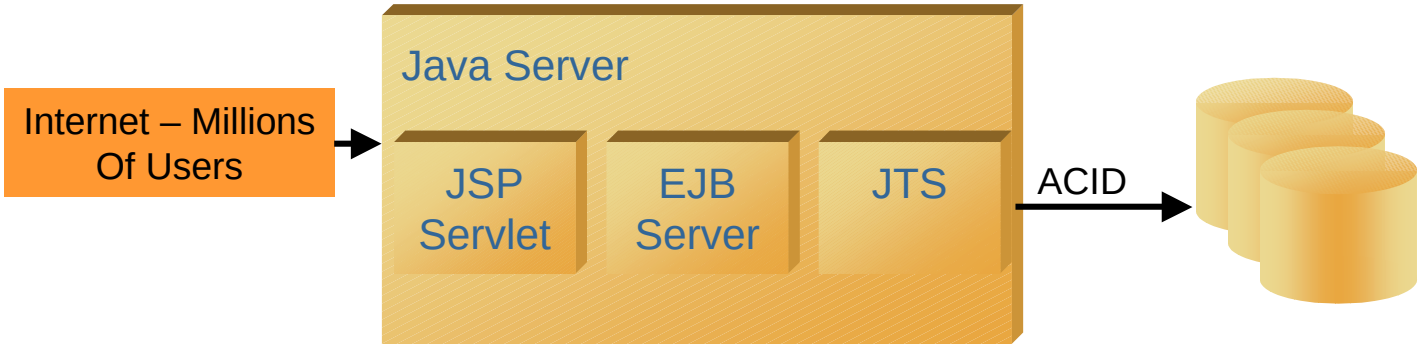


ORB Portability Harness

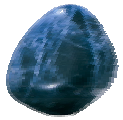
- Runtime
  - Layer between JTS and ORB
  - Abstractions for
    - BOA Initialization
    - BOA Shutdown
    - Initialization code
    - Locating objects and services
    - Threading (C++)
- Development
  - Make system to build targeting multiple orbs



Product integration

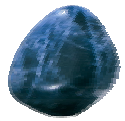


- High-performance transaction capability built directly into Single Process



JTS Desirable Features

- 100% Pure Java JTS 1.0.1 compliant product with full JTA support
- Configuration features
  - Interposition
  - Transaction Heuristics
  - Distributed Transaction Manager or Transaction Manager Server
  - Supports check/unchecked behavior
- Complex transaction features
  - Nested Transactions (also with 2PC)
  - Support for CosTransaction::Current
  - Direct/Indirect Transaction Management
  - Synchronization object support
  - Explicit/Implicit propagation
  - Crash Recovery
  - Multi-threaded aware
  - AIT
- Standards
  - XA Compliance
  - Support for JDBC 1.0 and 2.0
  - ORB Portable



Recommended reading

- “Principles of Transaction Processing”

  P.A. Bernstein and E. Newcomer

  1997, Morgan Kaufmann, San Francisco CA USA

  ISBN 1-55860-415-4

- “Enterprise Transaction Processing Systems: Putting the CORBA OTS, Encina++ and OrbixOTM to work”

  I. Gorton

  2000, Addison-Wesley, Harlow, England

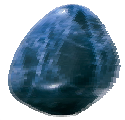
  ISBN 0-201-39859-1

- “Enterprise CORBA”

  D. Slama, J. Garbis, P. Russell

  1999, Prentice Hall PTR, Upper Saddle River, NJ, USA

  ISBN 0-13-083963-9



Pure Java Transaction Management for Tomorrow's Enterprise Applications

James G. Lynn **jlynn@bluestone.com**