Transactions on the Internet

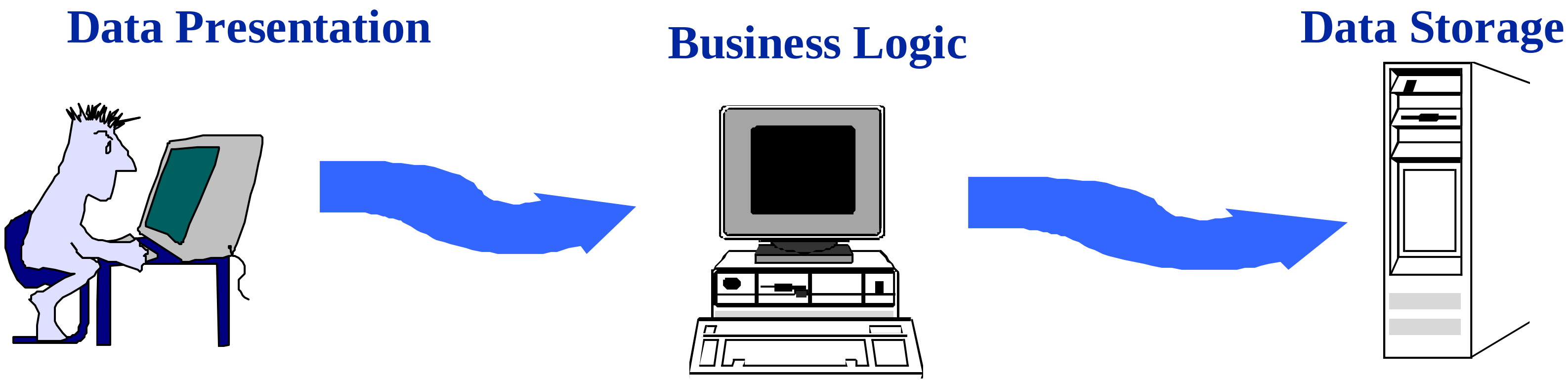
Zhixue Wu

APM Ltd.

Oct. 14 1997

Internet Applications

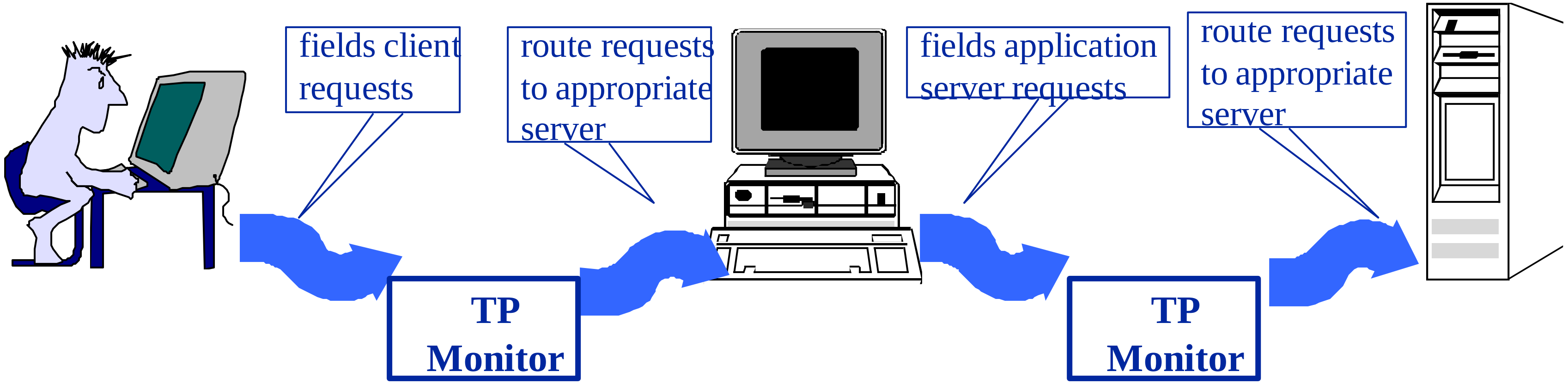
- Internet applications typically have a three-tier structure
- Second tier supports “business logic”
  - a sophisticated infrastructure
  - scalability: thousands of clients
  - performance: fine granularity of concurrency control
  - integrity: reliable and secure
  - deployment: demanding development, integration and configuration needs



Data Presentation **Business Logic** **Data Storage**

Transactions

- Transactions are used in OLTP to attack similar problems
- Transaction Process (TP) Monitor offers a middleware environment oriented to handling transactions on the Internet
  - Microsoft Transaction System, BEA TUEXDO, JavaSoft JDBC
- Support component-computing for the middle-tier



fields client requests

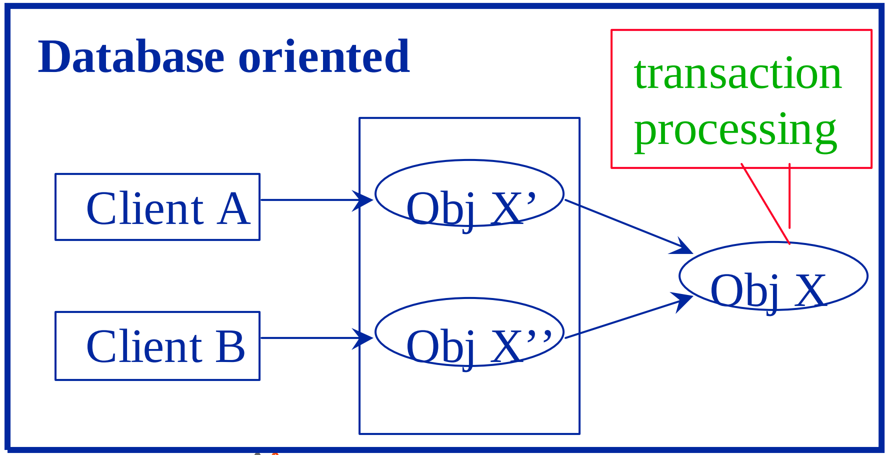
route requests to appropriate server

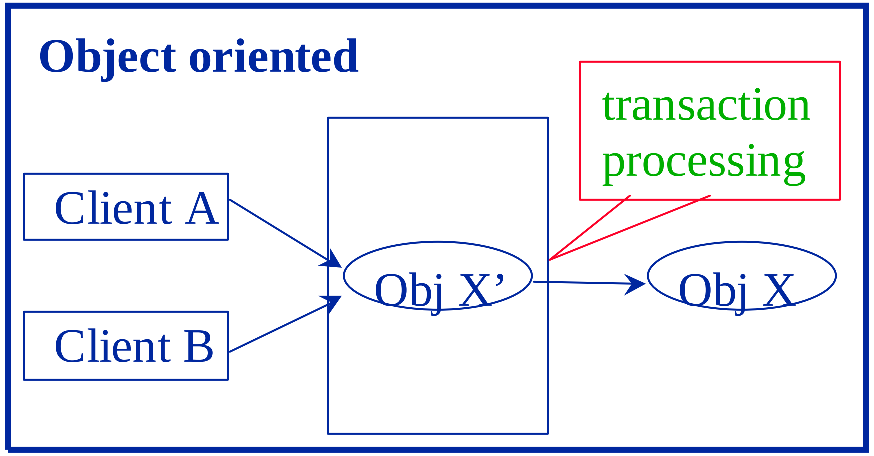
fields application server requests

route requests to appropriate server

Problems of Current Solutions

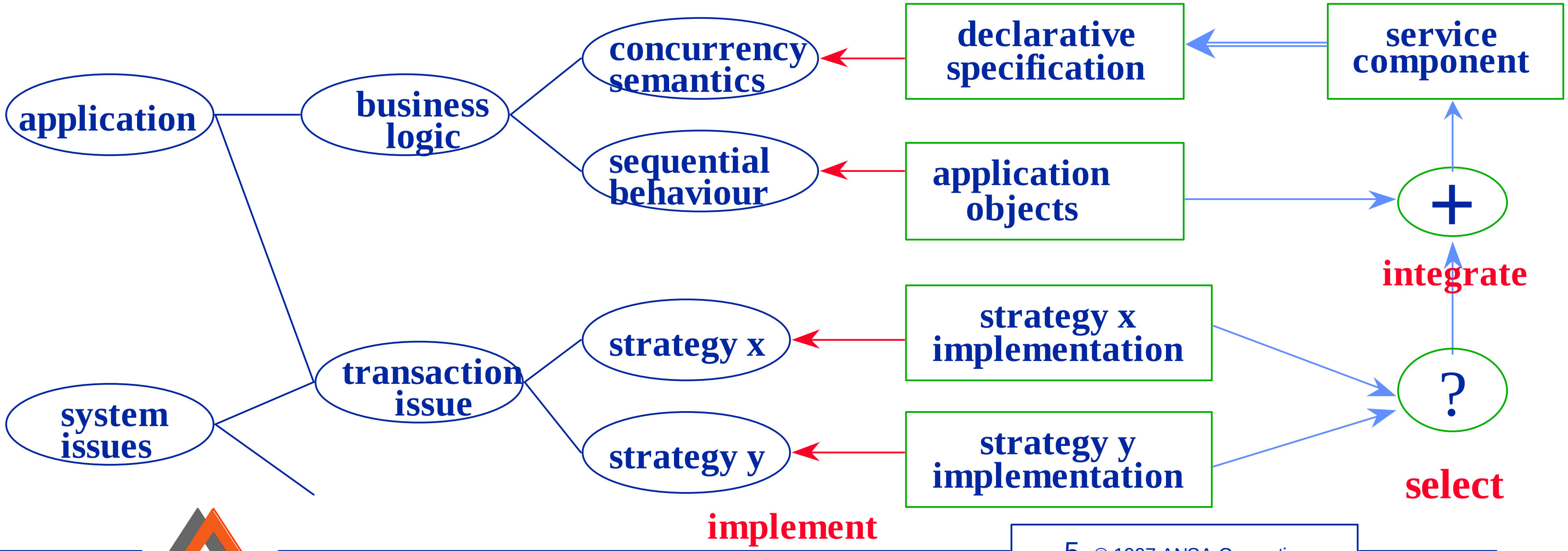
- Database-oriented, focus mainly on messaging
  - leverage existing products
  - database system responsible for concurrency control, recovery and persistence
  - file / record rather than object semantics
- Fixed and closed implementation
  - hard to upgrade, to customise, to apply different strategy
- Weak support for middle-tier development
  - no integration of transactions in objects





Open Approach

- Separate business logic and system issues
  - wrap application objects inside transaction frames
- Separate sequential behaviour and concurrency semantics
  - concurrency semantics specified declaratively
- Choose concurrency control strategies statically or dynamically



Concurrency Semantics

- Object semantics can be used to increase concurrency
- Single operation only
- Multiple read / single write
- Semantics-based
  - commute relation:
    - H0 * op1 * op2 = H0 * op2 * op1
    - op1 and op2 are commutative
  - invalidated-by relation:
    - op2(H0) != op2(H0 * op1)
    - op2 is invalidated-by op1

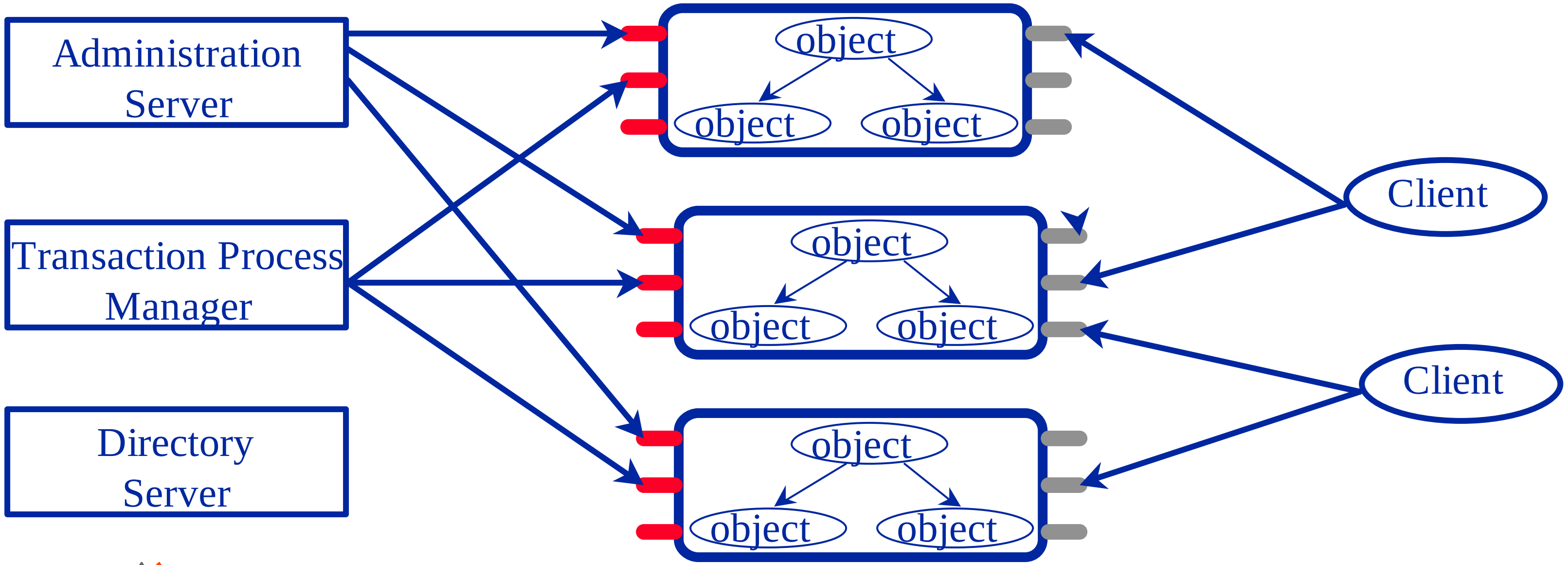
```
class Account {
            private  Money   amount;
            public           Account( );
            public   boolean credit(Money in);
            public   boolean debit(Money out);
            public   Money   balance( );
}

commute operations:
            (balance, balance)
            (credit,  credit)

invalid-by relations:
(credit,  balance)          (credit,  debit)
(debit,   balance)          (debit,   debit)
```

High Level Goals

- Focus on middle-tier (where less support is available)
- High transparency to application developers
- Component based approach
  - making use of JavaBeans
  - compositional
- Easy integration of business logic to transaction framework
- Easy component assembly
- Conform to Java standards as much as possible
- Fit to other parts of FlexiNet

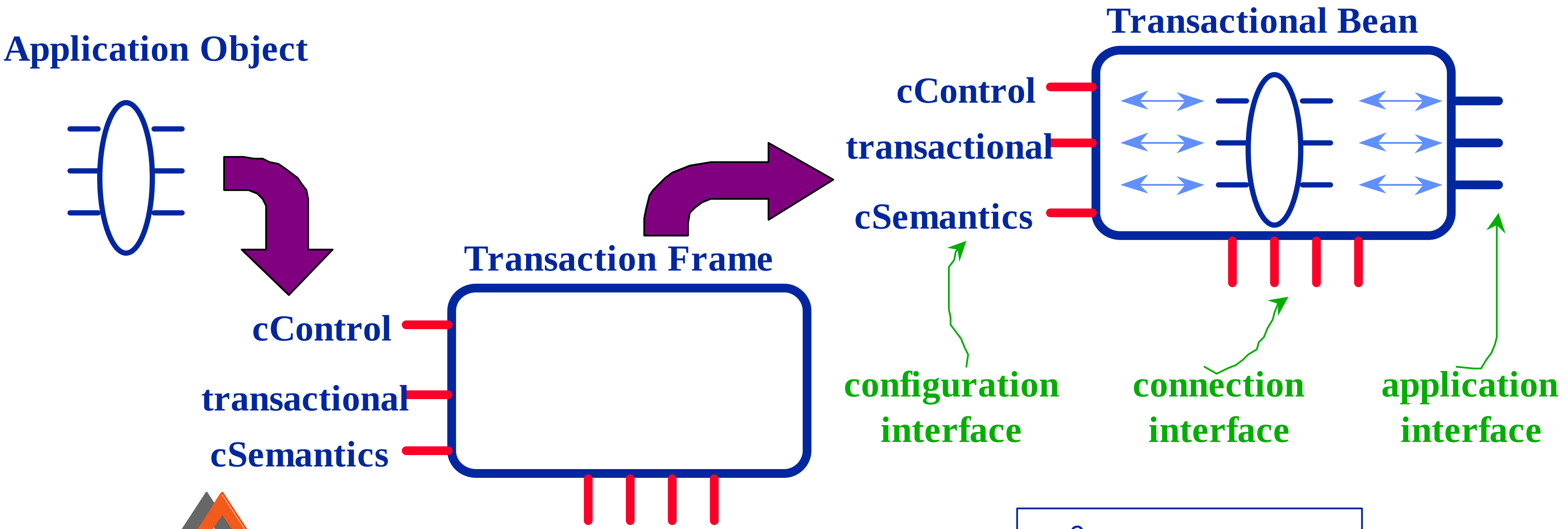
Infrastructure (middle-tier)

- Transactional Beans
  - a transaction frame + a set of objects
  - unit for concurrency control, deployment, and management
- Communication with other Beans and client via application interface
- Meta interface is used by TP Managers and administration servers



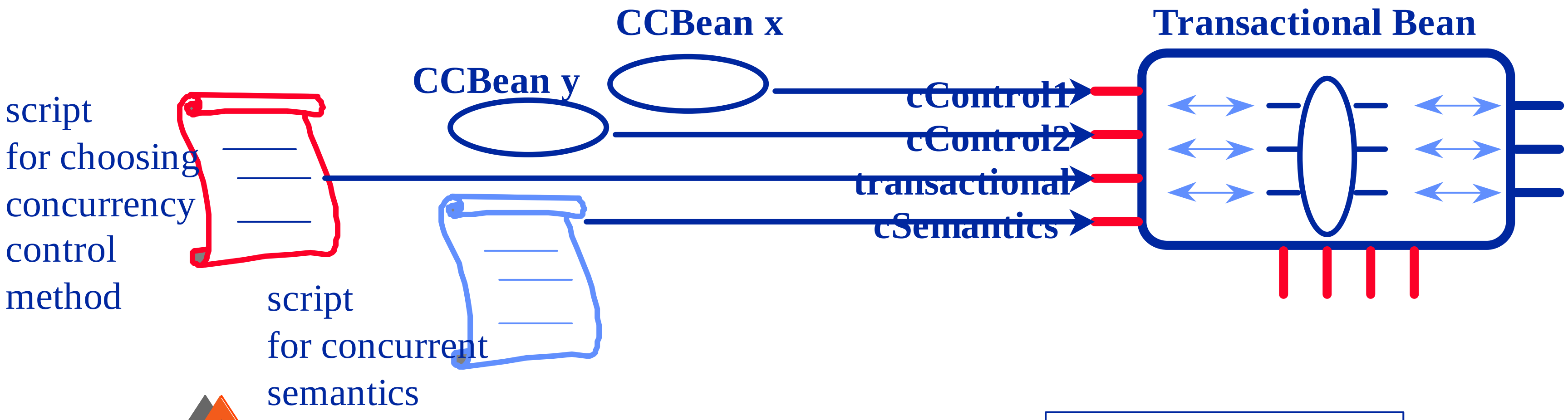
Construction of Transactional Beans

- No special rules for application object
- The application interface and default concurrency control semantics are generated automatically when putting objects inside a frame
- Transactional Beans can be customised via configuration interface
- Connection interface is used for connecting to TP Manager



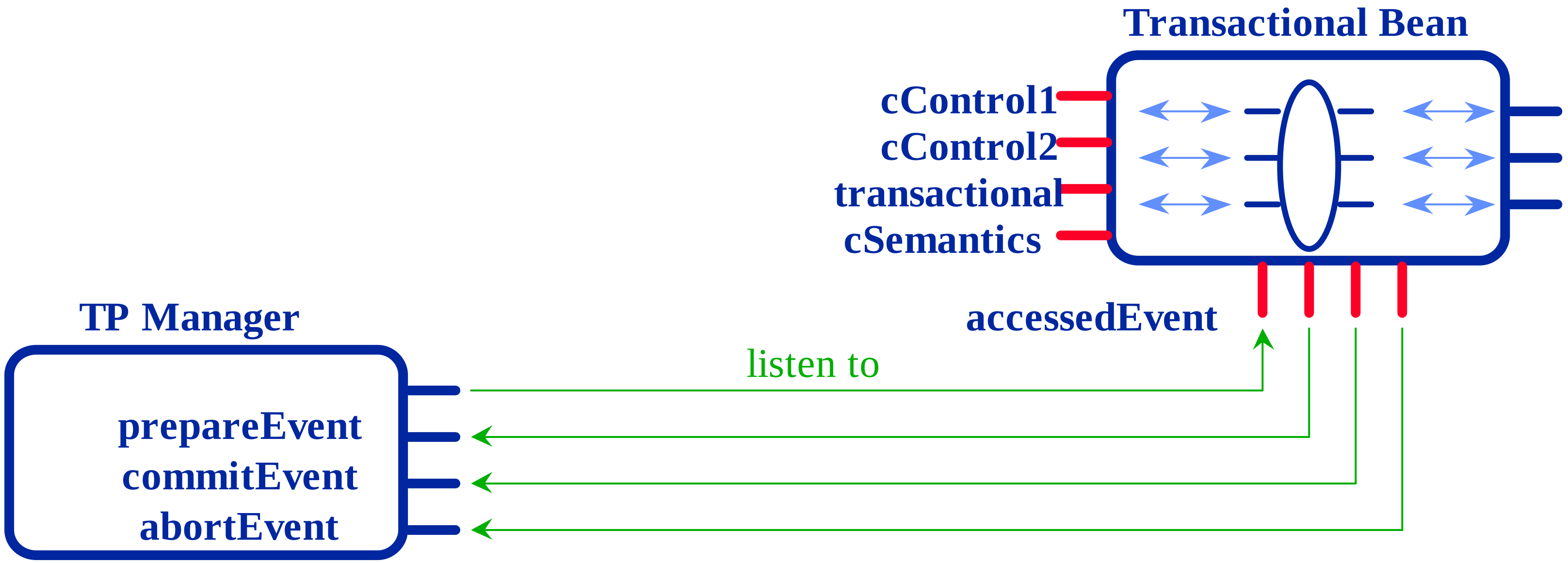
Configure Transactional Beans

- A CCBean provides a concurrency control & recovery method
- Several CCBeans can be connected to a Transactional Bean
- A transaction script describes the policy to choose a CCBean
- A semantics script describes the concurrent semantics of a bean



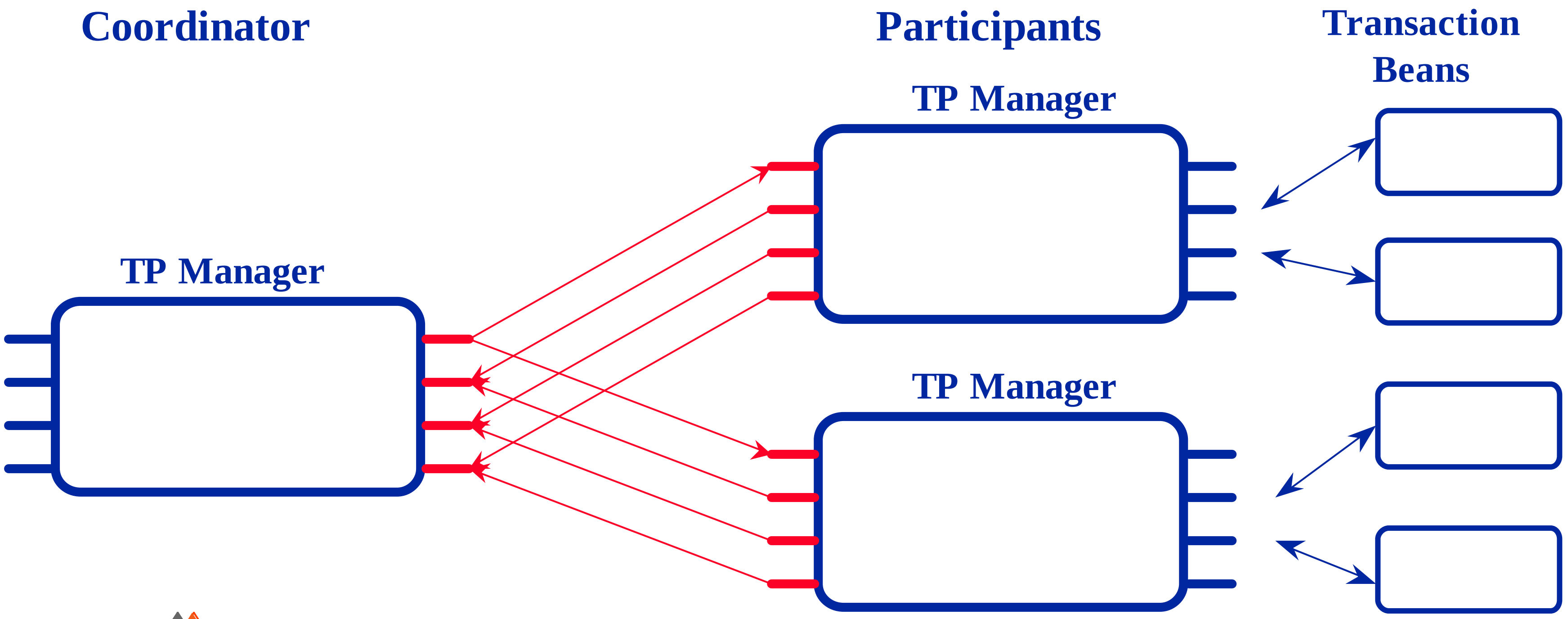
Connection to TP Manager

- Implements the 2-phase-commitment protocol
- Using JavaBeans event model for connection to TP Manager Bean
- Each Bean runs in an individual process
- Each domain has one TP Manager (scalability)



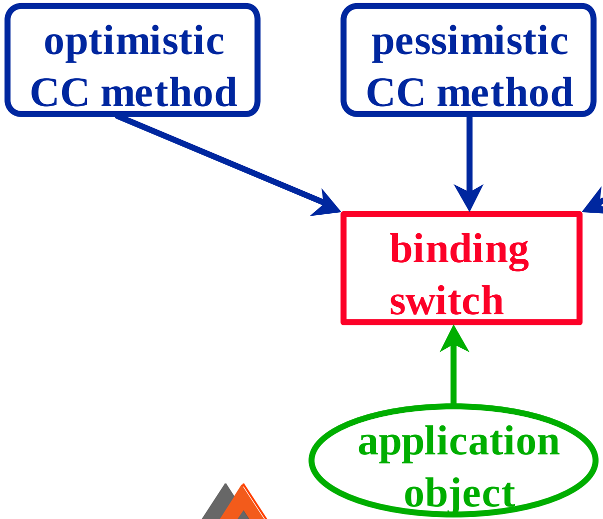
Connections between TP Managers

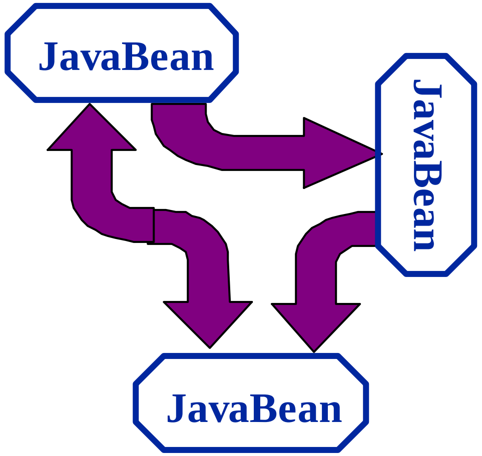
- Implements the 2-phase-commitment protocol
- The coordinator: originate a transaction
- Participants: some beans in its domain involved in the transaction



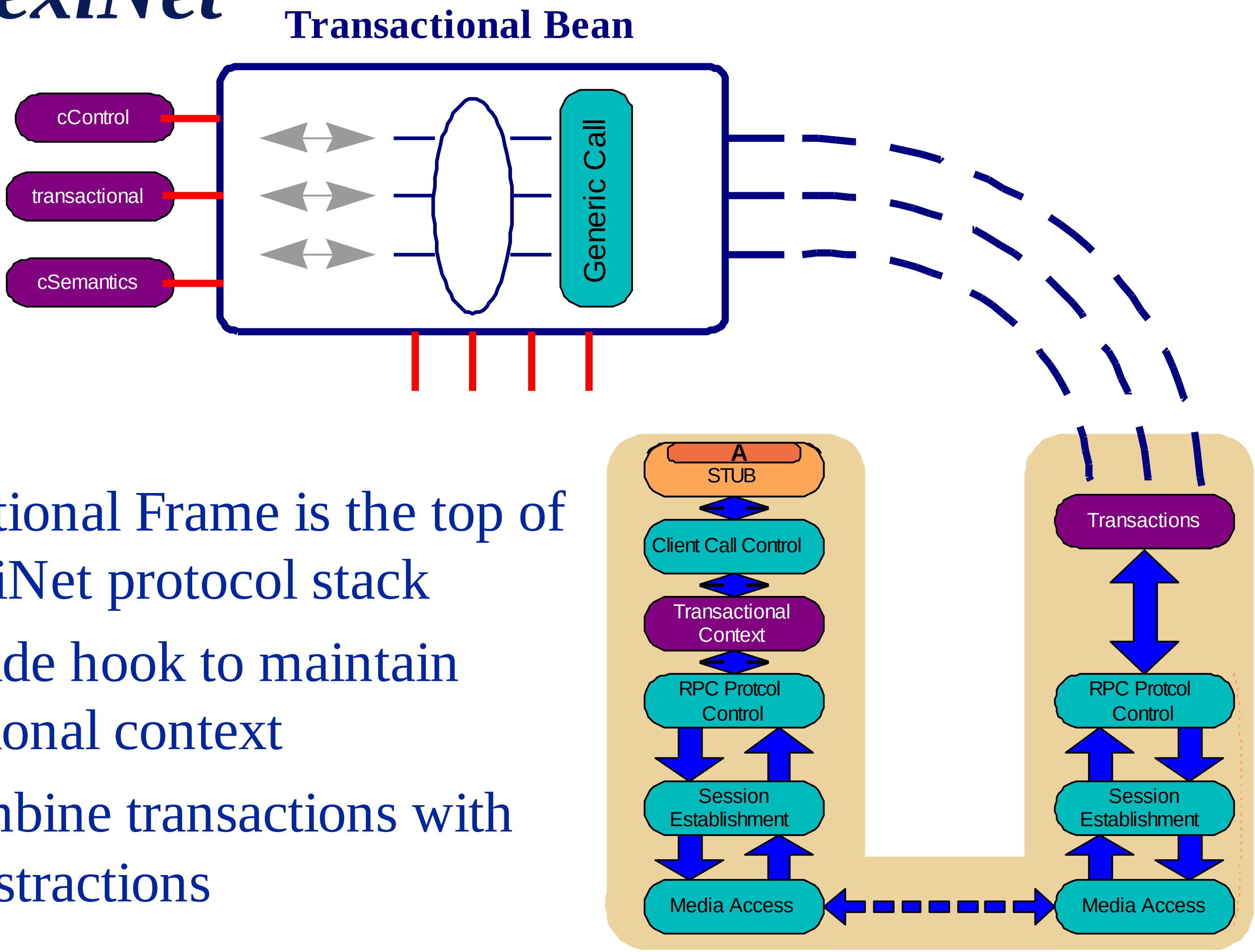
Enabling Technology

- Reflection and meta programming
  - business logic is implemented in application objects
  - CC methods are implemented in meta objects
  - integration through binding (static or dynamic)
  - changing CC methods is done through changing binding
- JavaBeans
  - component approach
  - reusable components
  - easy assemble (visual building tools)





Link to FlexiNet



- Transactional Frame is the top of the FlexiNet protocol stack
- Client-side hook to maintain transactional context
- Can combine transactions with other abstractions

Research Issues

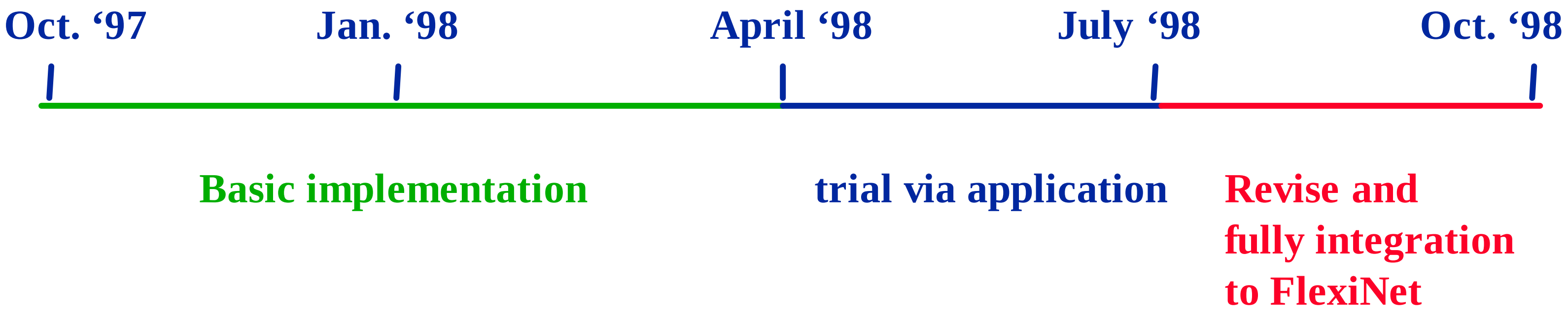
- The specification and use of concurrent semantics
- The specification of a policy for choosing CC method
- Dynamic choose and change CC method at run time
- Integration of CC methods with a transaction frame
- The standard interface for CC and Transactional Beans
- Pure Java implementation
- Total transparency

Benefits

- Powerful support for developing middle-tier
- Easy to specify and use application semantics
- Pure Java implementation
- To business logic developer
  - high transparency, thus easy implementation
  - easy integration with other components
  - reusability and using off-shelf products
- To system software developer
  - wide usability, and reusability
  - easy to meet specific requirements
  - easy to meet new challenge
- To system assembler
  - easy to inspect the properties of each component
  - standard procedure for configuration and assembling
  - free choice of product from any vendors
  - easy and flexible for customisation and upgrade

Milestones

- Demonstrate transaction frame
- Demonstrate declarative concurrency control in frames
- Demonstrate integration with FlexiNet binding



Oct. ‘97

Jan. ‘98

April ‘98

July ‘98

Oct. ‘98

Basic implementation

trial via application

**Revise and
fully integration
to FlexiNet**

Deliverables

- Specification & implementation of the Transaction Frame
- Specification of the CCBean
- Script language for specifying concurrent semantics
- Script language for specifying policy of choosing CCBean
- Code generator for producing Transactional Bean wrapper
- Specification & implementation of the transaction system
- Some CCBean samples
- A demonstration application