

Transactions in Data and Process Management

dr.ir. P.W.P.J. Grefen

Computer Science Department

University of Twente

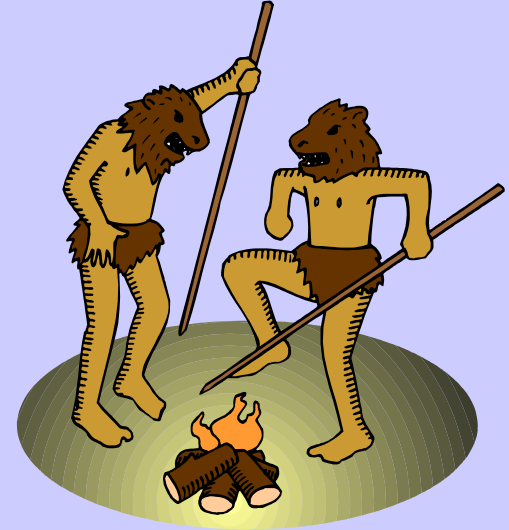
A Piece of History



Transaction Management Eras

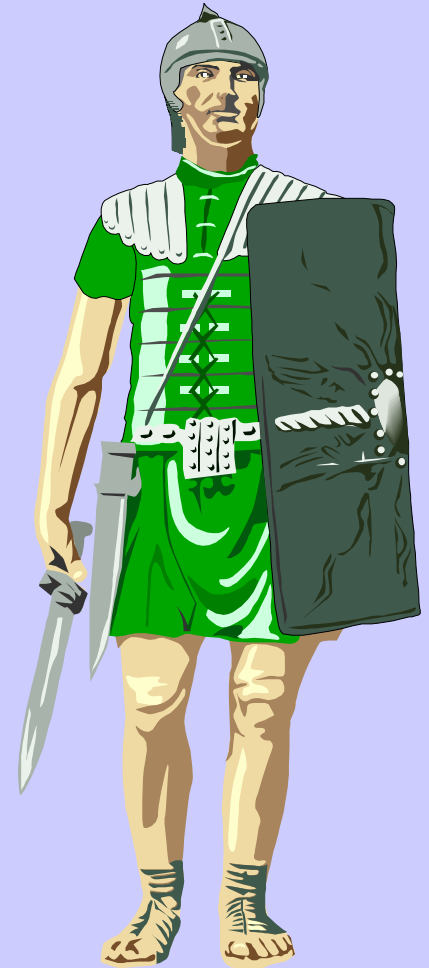
- Stone Age
 - no transactional applications
- Classic History
 - transactions in debit/credit type applications
- Middle Ages
 - transactions in advanced applications
- Modern Times
 - transactions in cross-organizational applications

- No transactions



Classic History

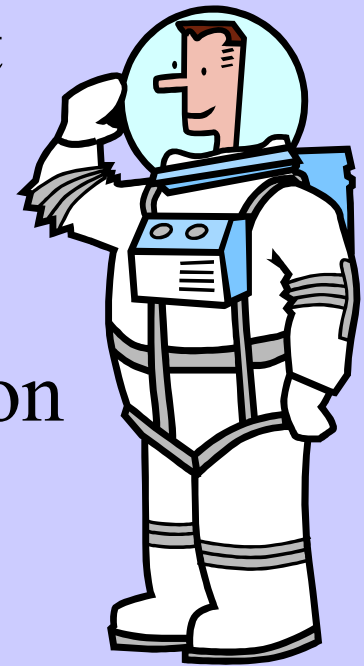
- Flat transactions
 - Debit/Credit type
 - Strongly database-oriented
- ACID properties
 - Rigid semantics
- Towards distributed transactions
 - Extension of flat ACID transactions



- Applications with advanced transactional requirements
 - CAD/CAM, Groupware, WFM
- Many extended transaction models
- Complex (to very complex) semantics
- Usually limited support
 - only available on paper
 - only available in academic prototype



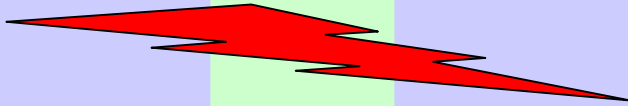
- Cross-organizational applications
 - XO process support, Electronic commerce
- Cross-organizational transactions
 - linking of local transaction support
- Autonomy, encapsulation
 - protocols, mapping
- Dynamic transaction configuration
 - dynamically configurable support



The Stone Age Problems

Incomplete Transaction Problem

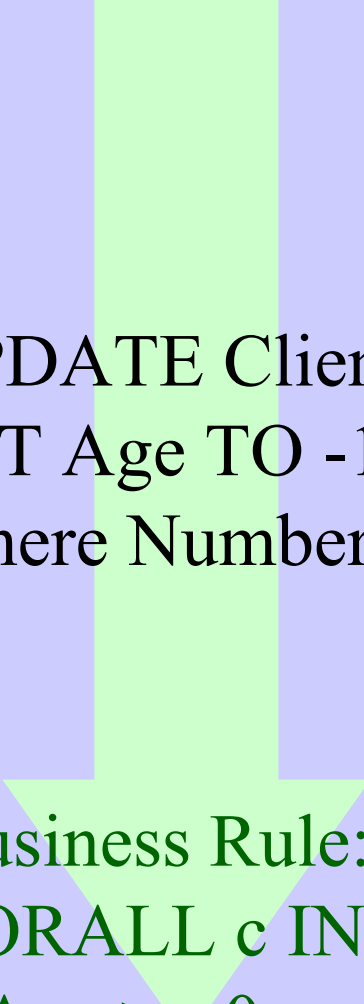
UPDATE Account
SET Balance TO Balance - Transfer
Where Number = 1234



crash!!!

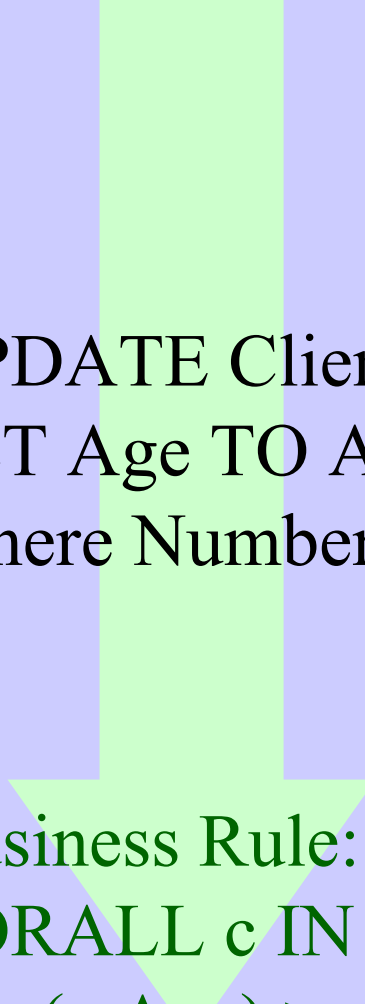
UPDATE Account
SET Balance TO Balance + Transfer
Where Number = 4321

Incorrect Transaction Problem



```
UPDATE Client  
SET Age TO -10  
Where Number = 1234
```

Business Rule:
FORALL c IN Client
c.Age \geq 0



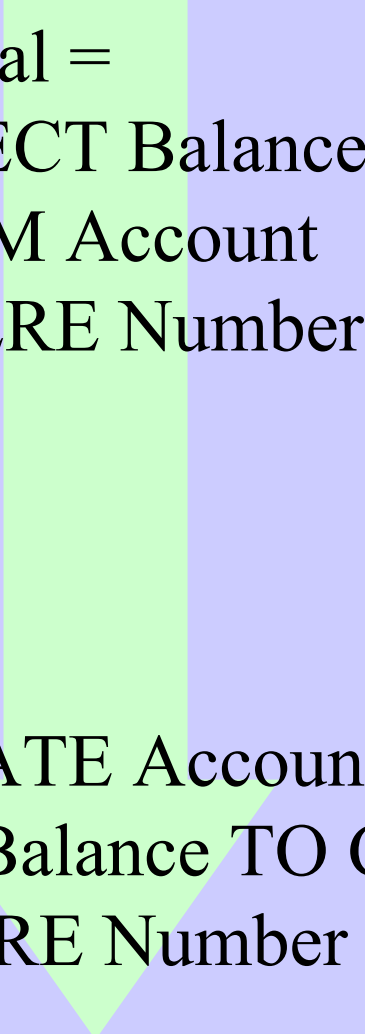
```
UPDATE Client  
SET Age TO Age-2  
Where Number = 1234
```

Business Rule:
FORALL c IN Client
new(c.Age) \geq old(c.Age)

Lost Update Problem



```
UPDATE Account  
SET Balance  
TO Balance * 1.05
```



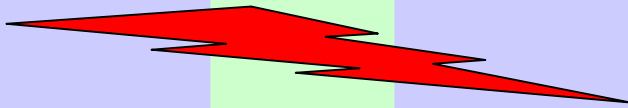
```
CurBal =  
SELECT Balance  
FROM Account  
WHERE Number = 1234
```

```
UPDATE Account  
SET Balance TO CurBal+input  
WHERE Number = 1234
```

Lost Effects Problem

UPDATE Account
SET Balance TO Balance - Transfer
Where Number = 1234

END



crash!!!

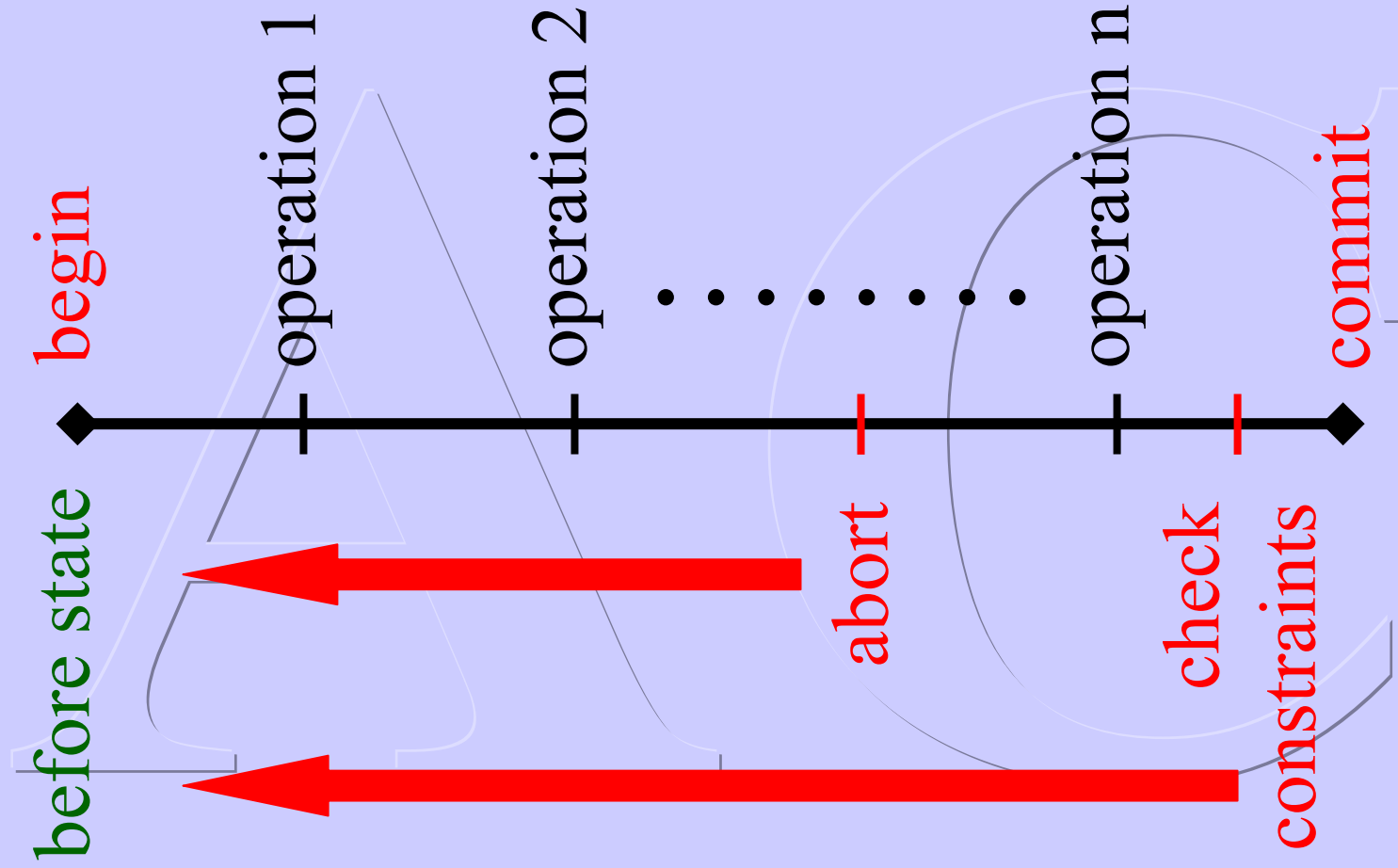
WRITE new(Account)
TO STABLE STORAGE

The Classic Transactions

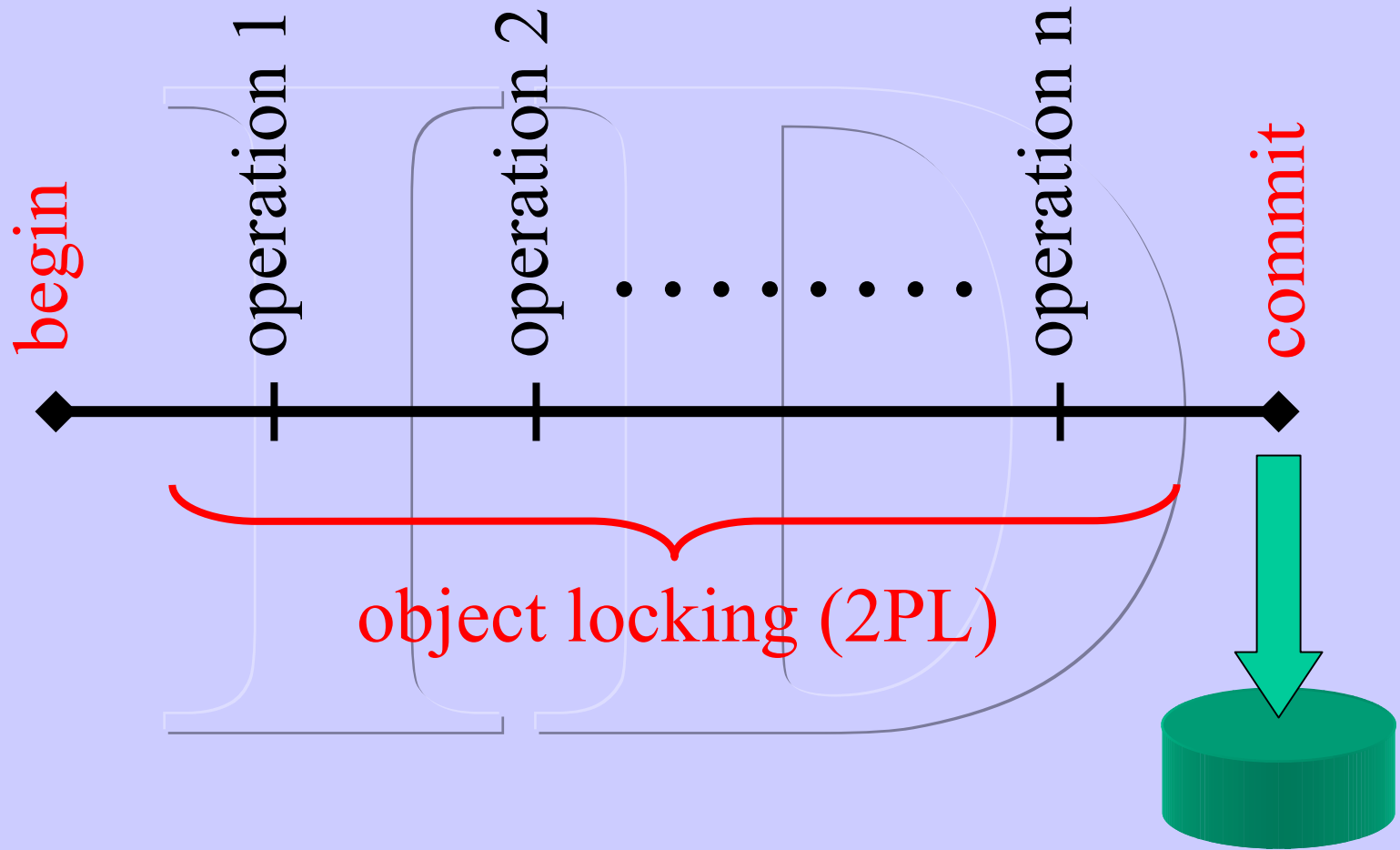
ACID Properties

- **A**tomic: a transaction has all-or-nothing semantics: either it completes or is undone
- **C**orrect: a transaction performs a correct transition resulting in a correct state
- **I**solated: the intermediate results of a transaction are only visible when it commits
- **D**urable: the committed results of a transaction are permanent

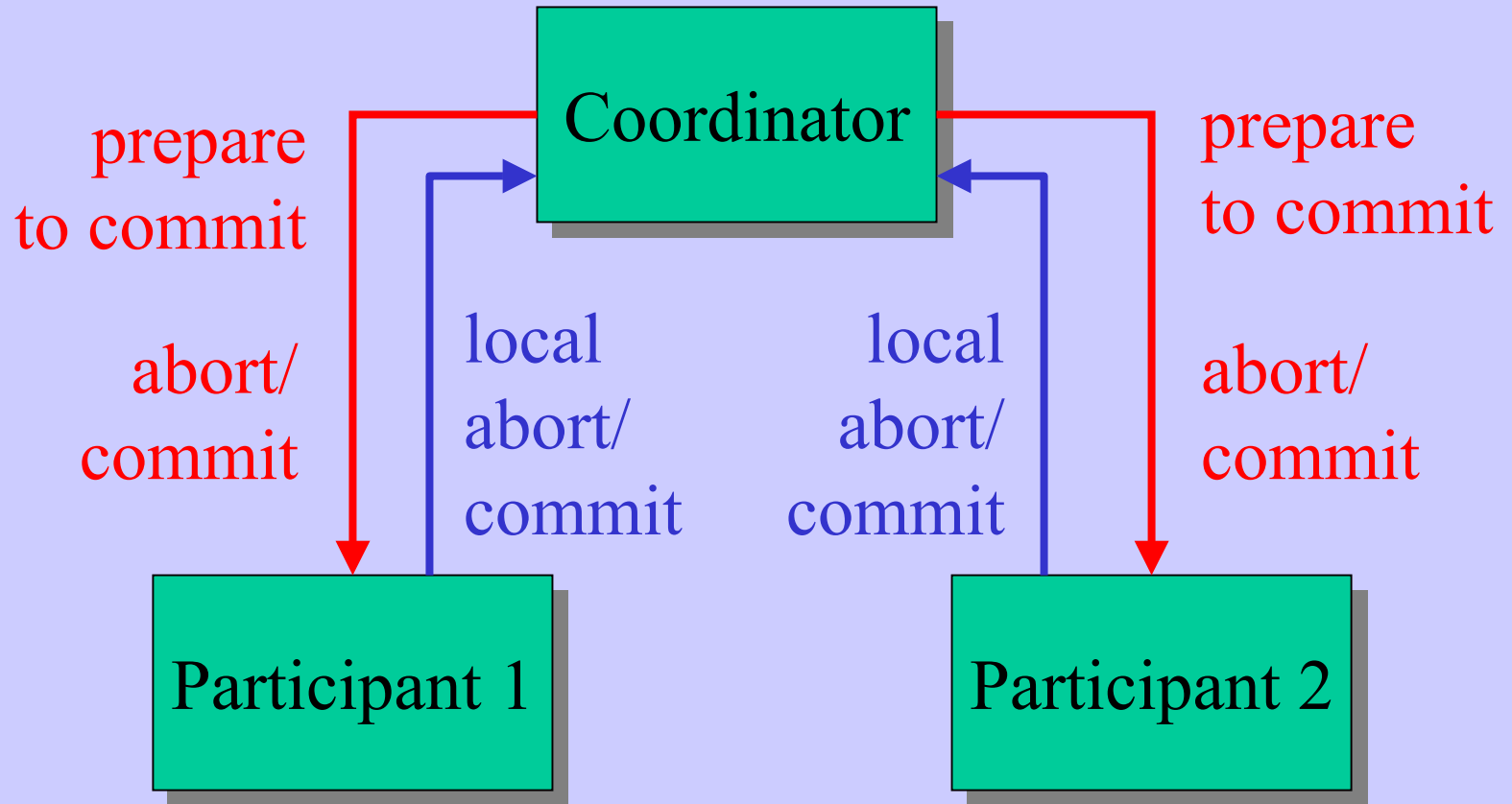
Classic Transaction Mechanisms (1)



Classic Transaction Mechanisms (2)

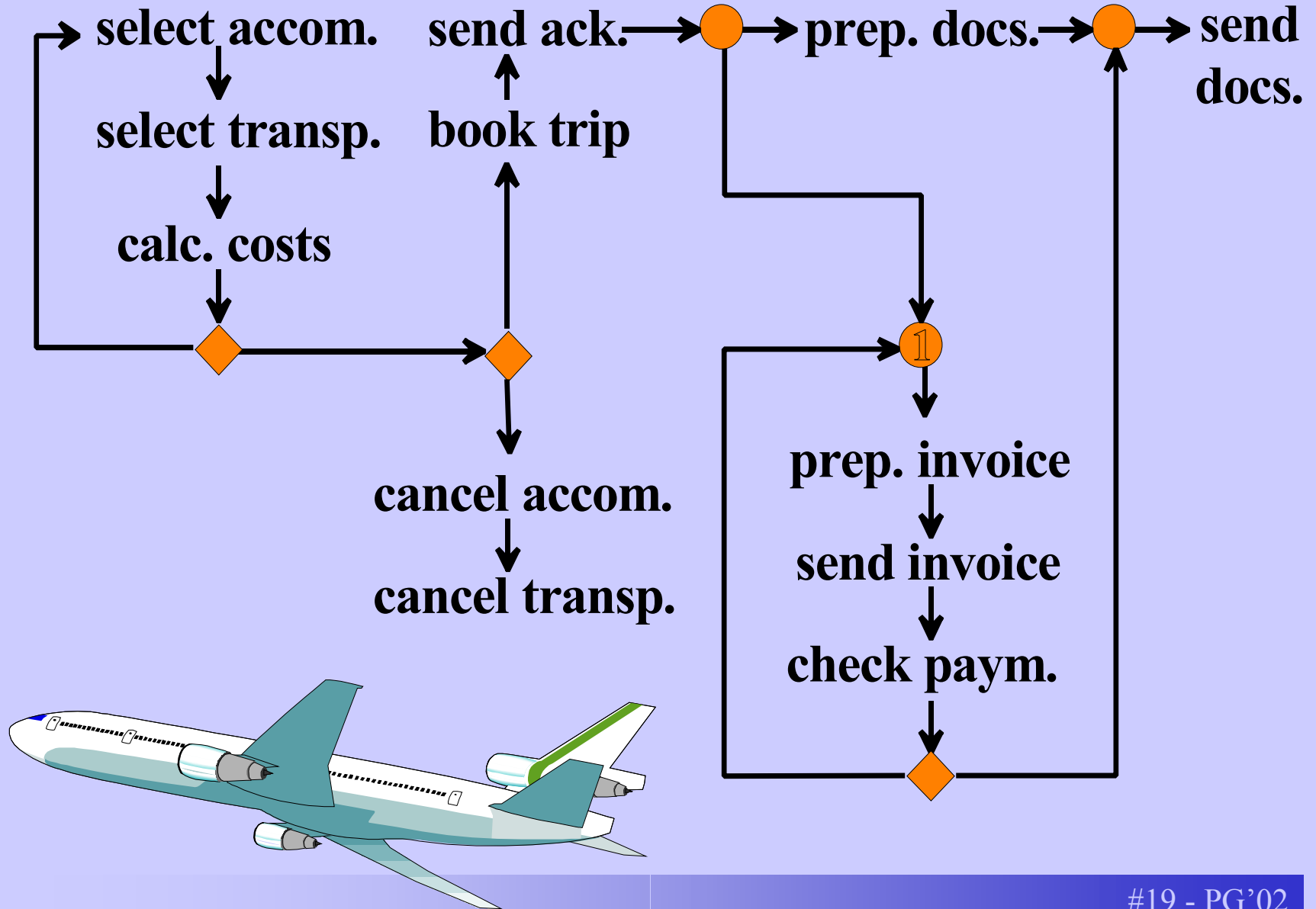


2PC for Distributed Transactions

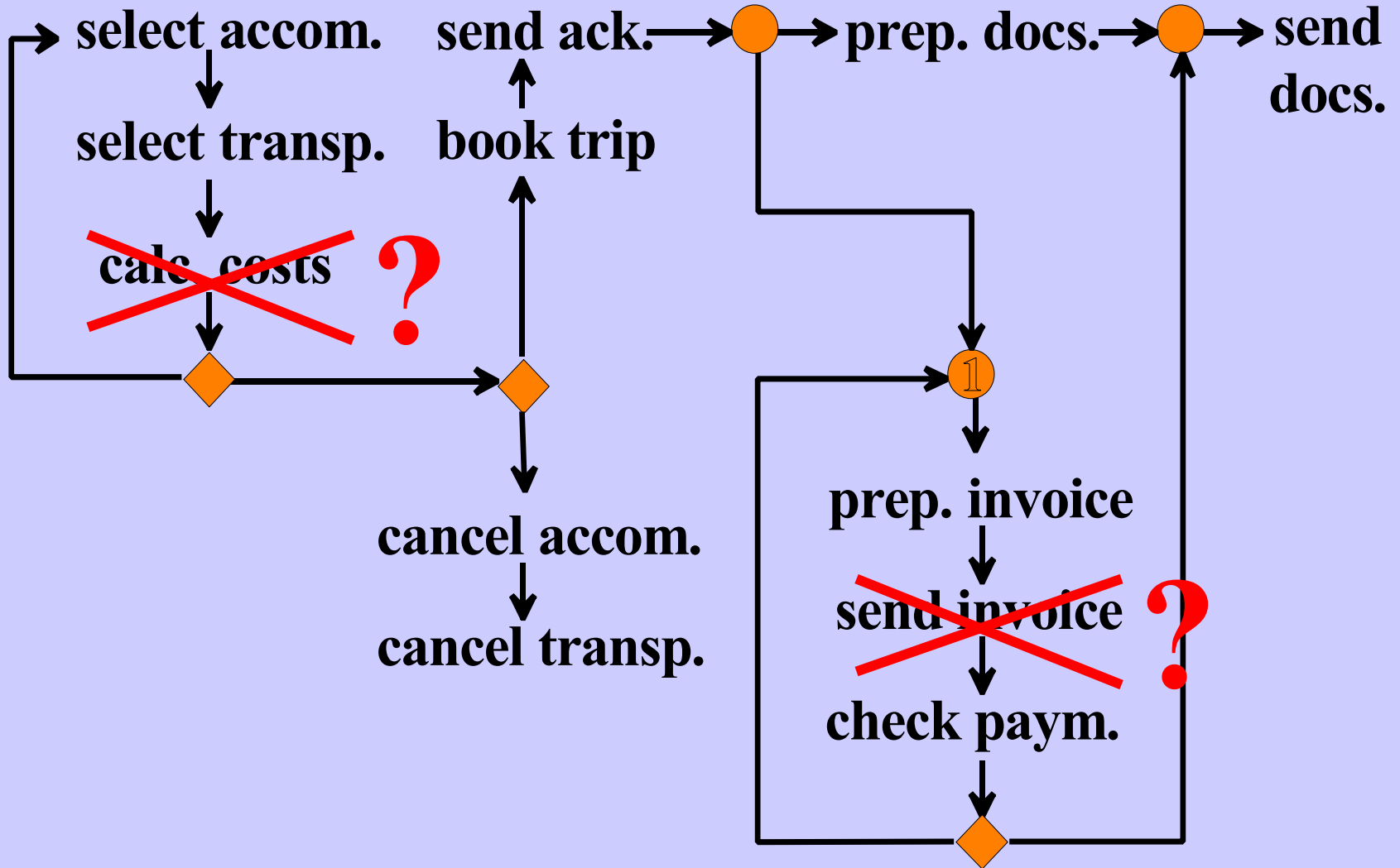


Middle Age Approaches to Process Support

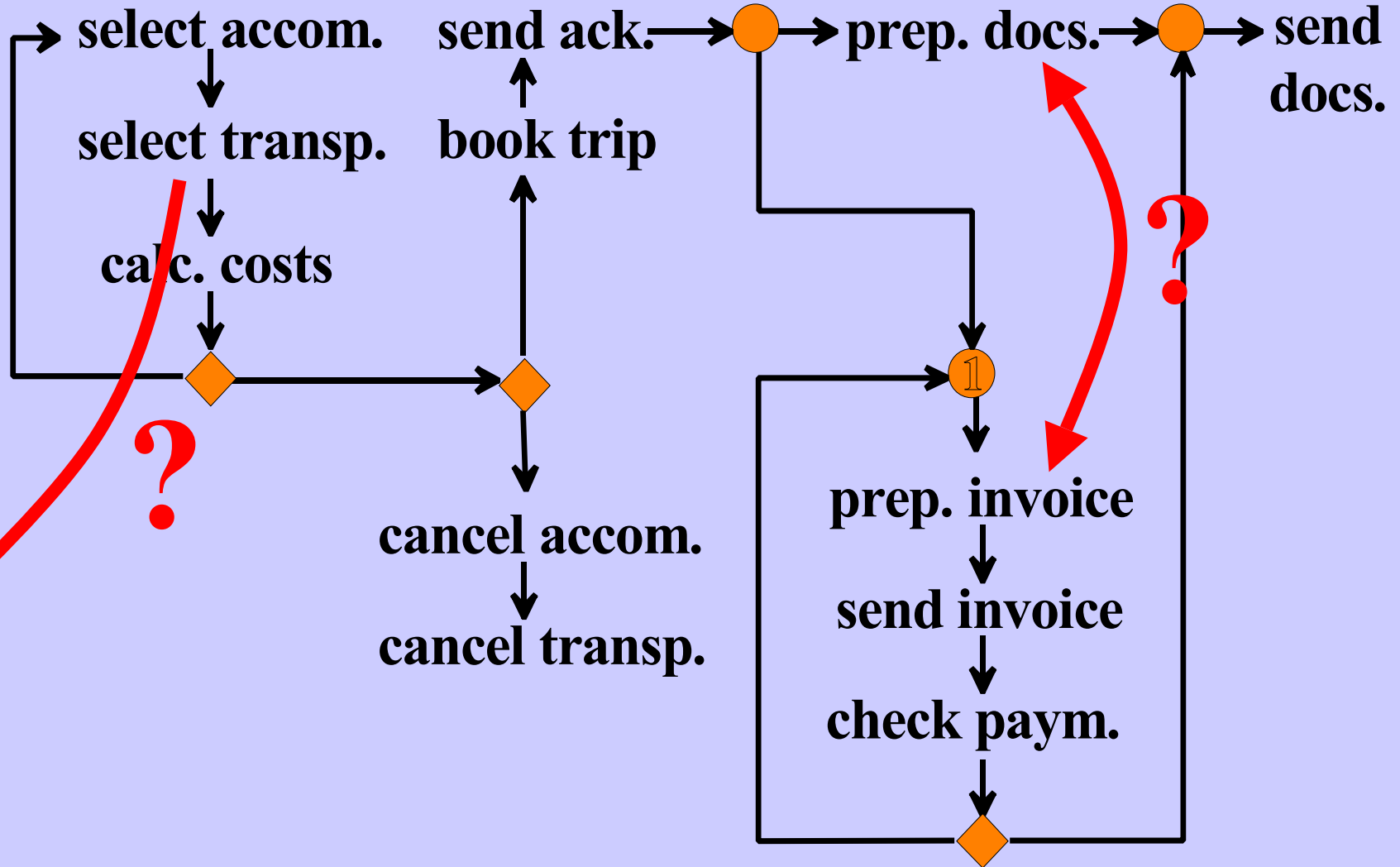
Example Process



Atomicity?



Isolation?



Transactional Requirements

- Need to specify failure semantics: which parts of process behave as a whole (atomically)?
- Need to specify data sharing semantics: which parts of process are shielded (isolated) from their context w.r.t. their variables?
- Reliable process semantics requires transaction mechanism!

Further Requirements

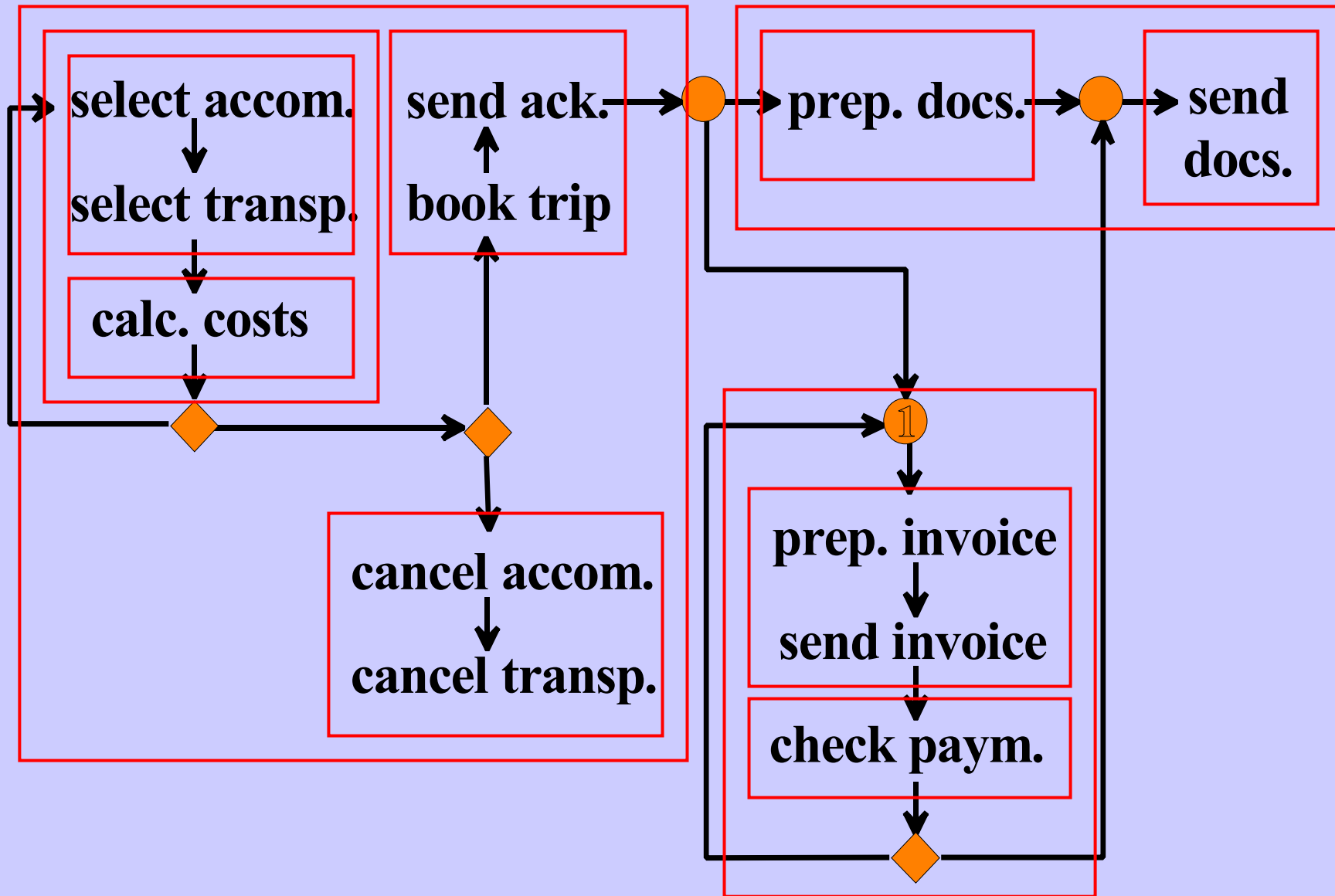
- long transaction support
- friendly resource sharing
- visibility some intermediate results
- partial rollback
- application-dependent rollback
- complex action structures
- intra-transaction parallelism
- multiple actors per transaction

Extended Transaction Models

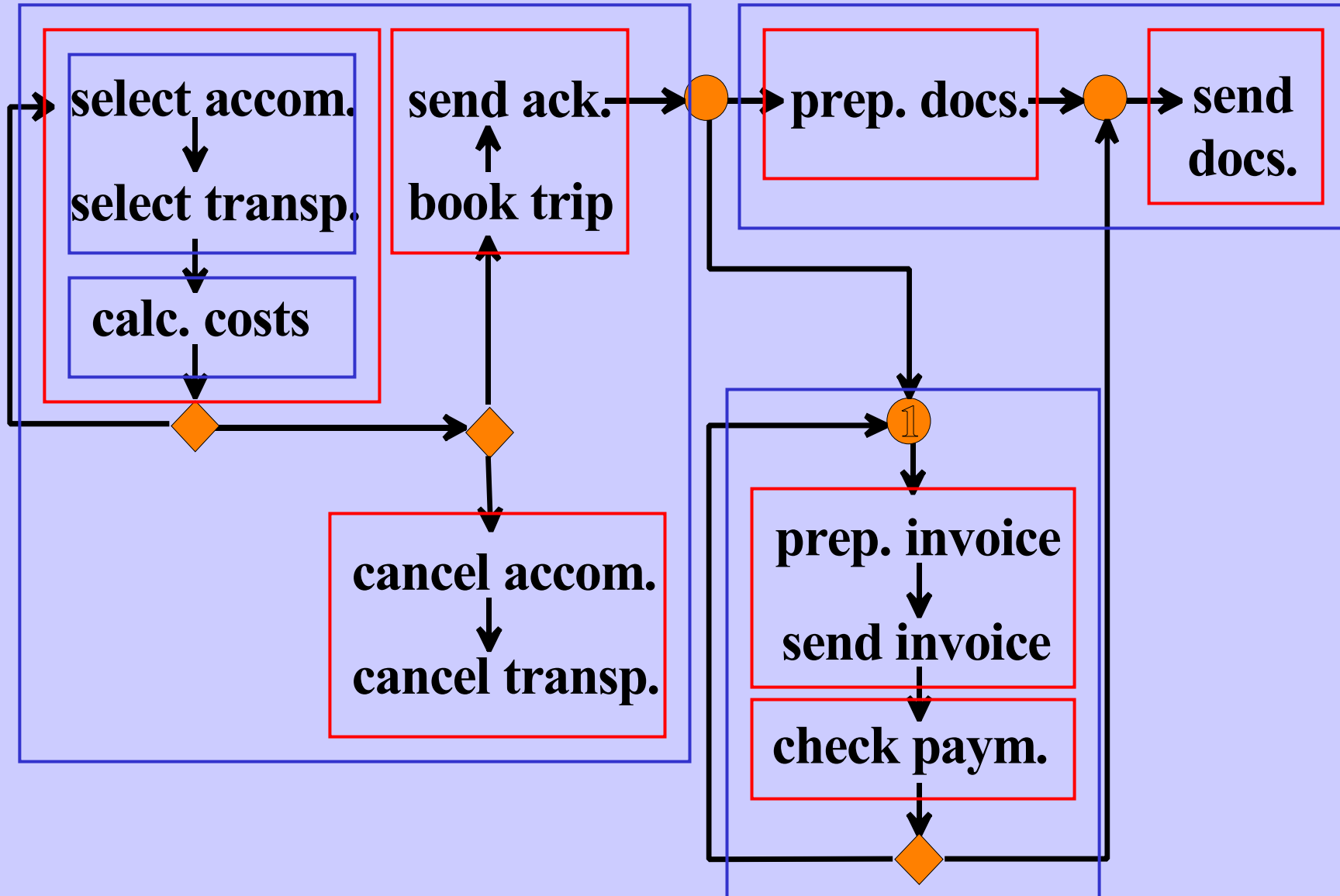
- Various models proposed, e.g.:
 - multi-level transaction (Weikum '91)
 - transaction hierarchies (Chen '97)
- Various approaches proposed, e.g.:
 - Exotica (Alonso '96)
 - transaction adapters (Barga '95)
- Limitations often:
 - complex model and semantics
 - concept or prototype status only
 - partial solution to problem

The WIDE Approach to Workflow Support

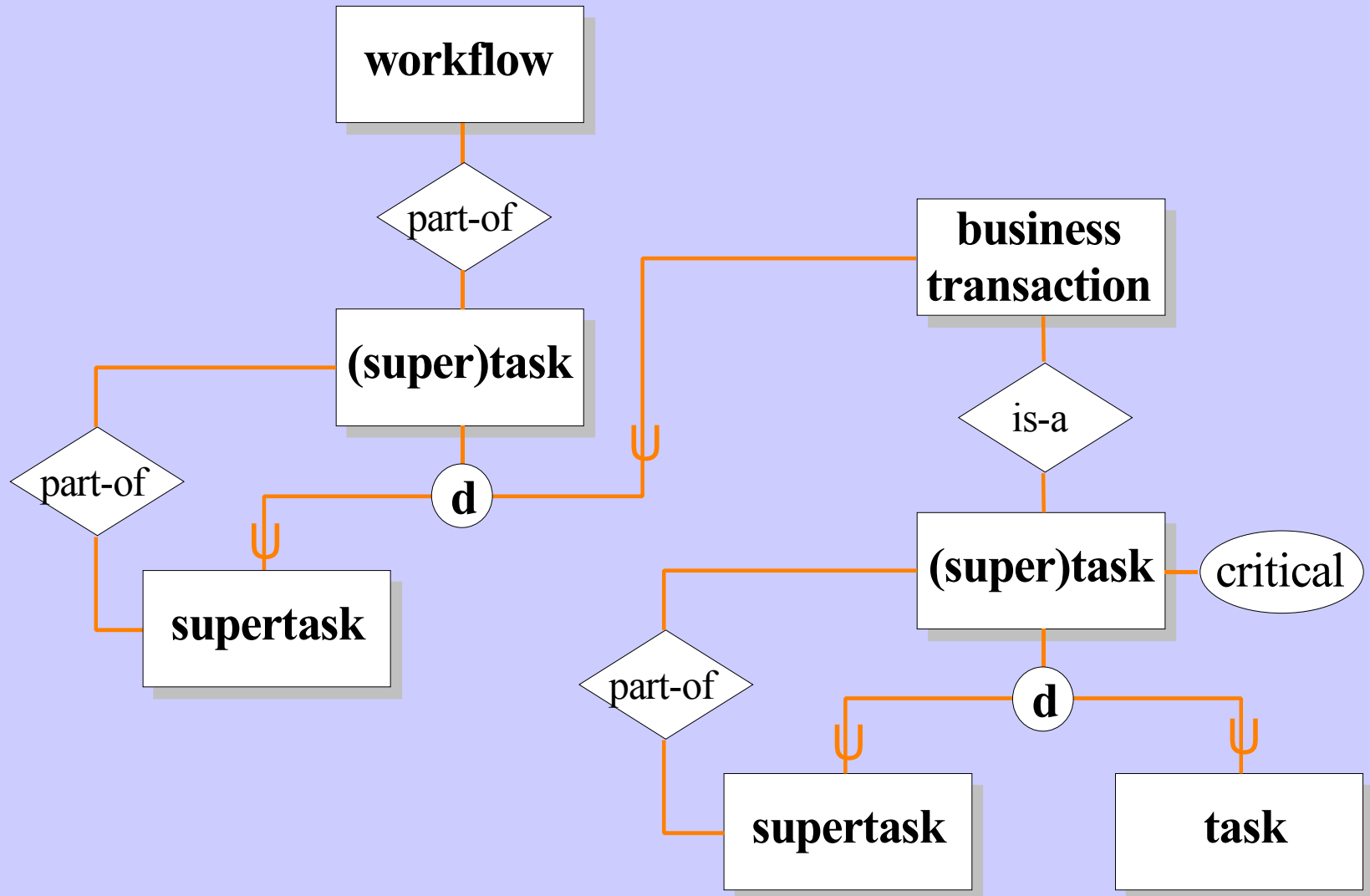
Example Workflow



Example Workflow



Process Model



Two-Layer Transaction Model

Upper layer processes:
long duration
relaxed isolation
no atomicity
application-based rollback



Global transactions:
based on saga
transaction model

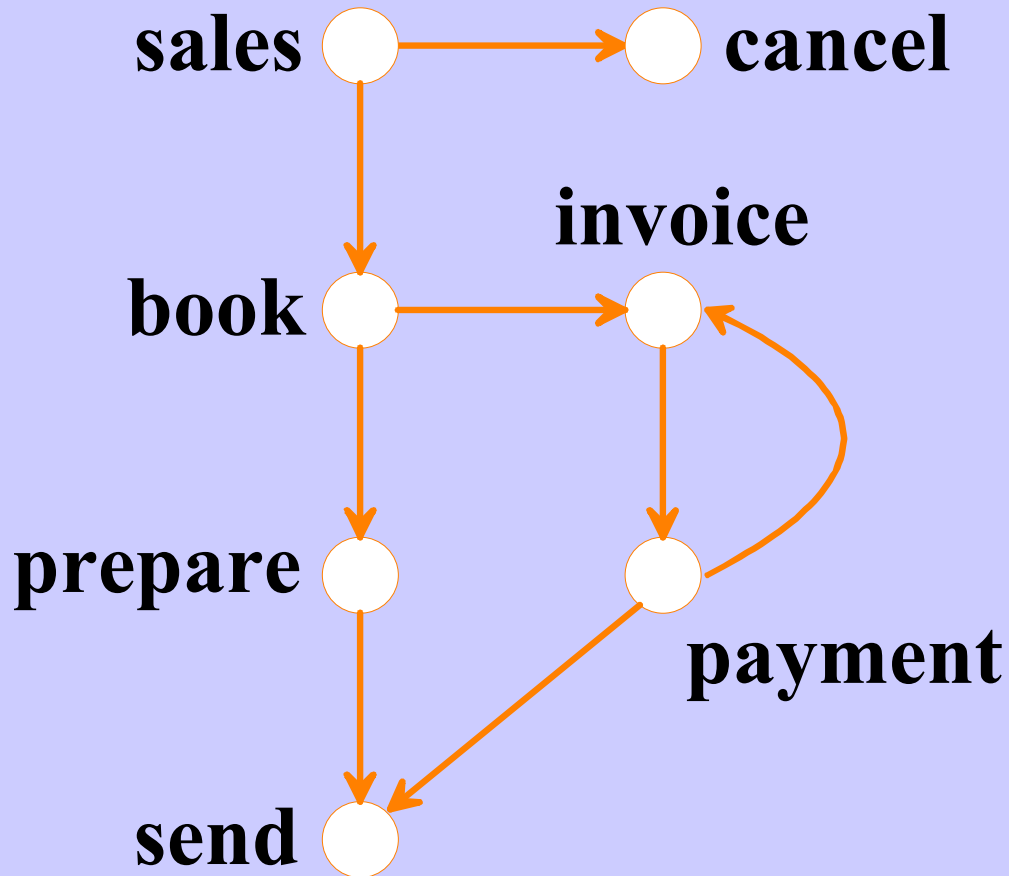
Lower layer processes:
short duration
strict isolation
flexible atomicity
state-based rollback



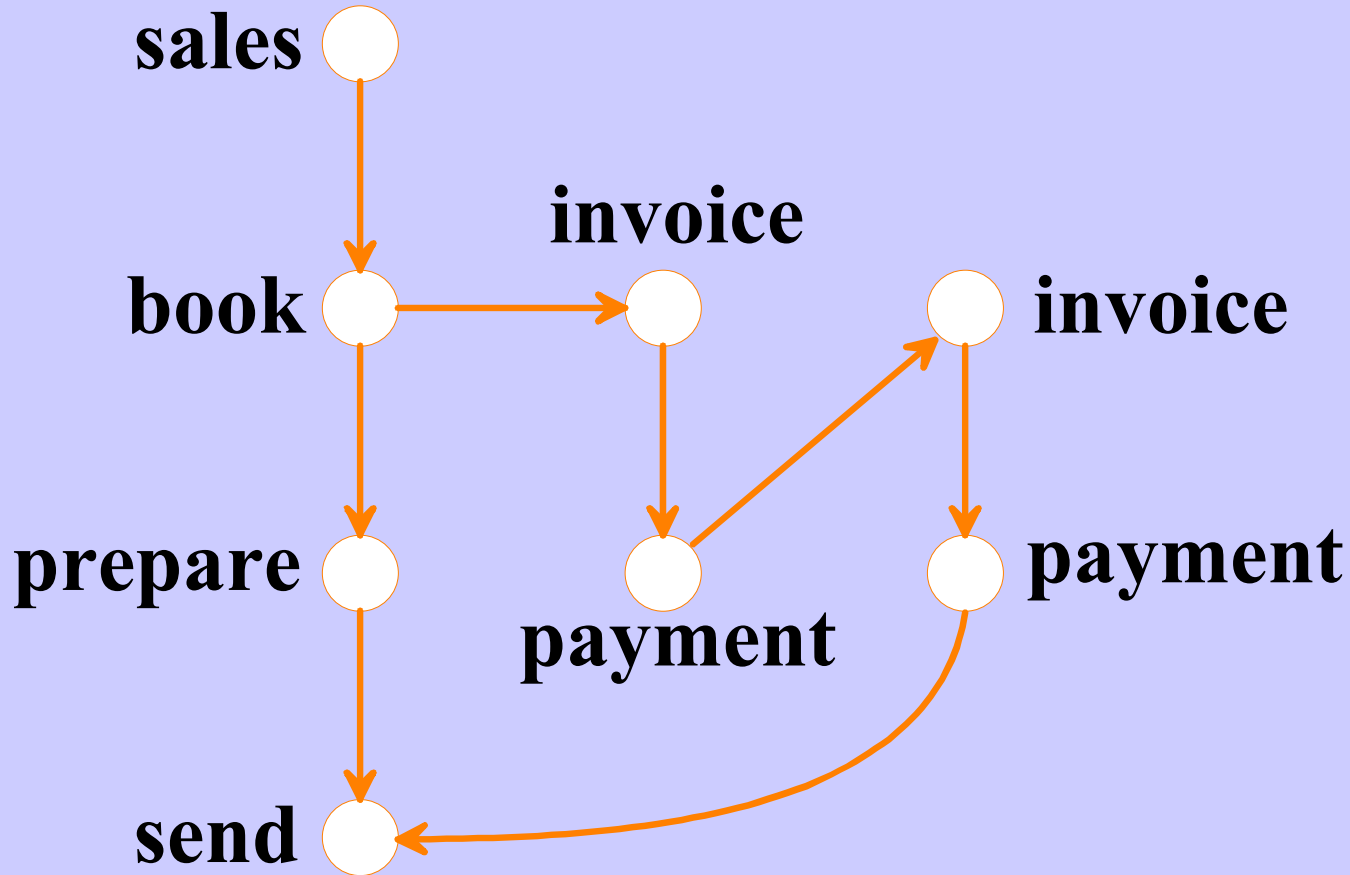
Local transactions:
based on nested
transaction model

- Graph of ‘black box’ local transactions
- Intermediate results visible to other transactions (non-isolated)
- Abort based on process semantics through compensating actions
 - brings application back to semantically equivalent state, not same state
 - compensation of funds transfer is reverse transfer
 - compensation of sending letter is sending another letter

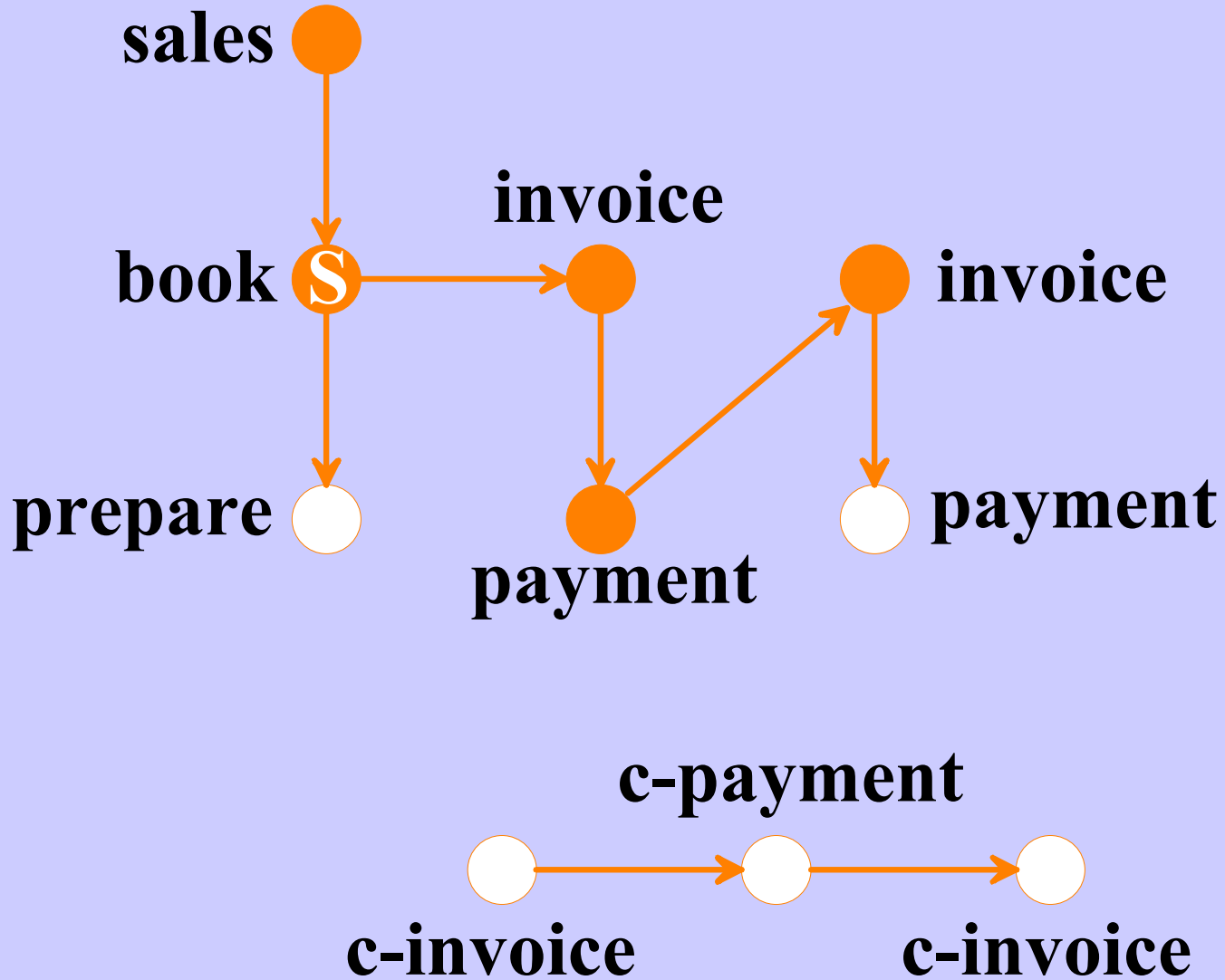
Specification Graph



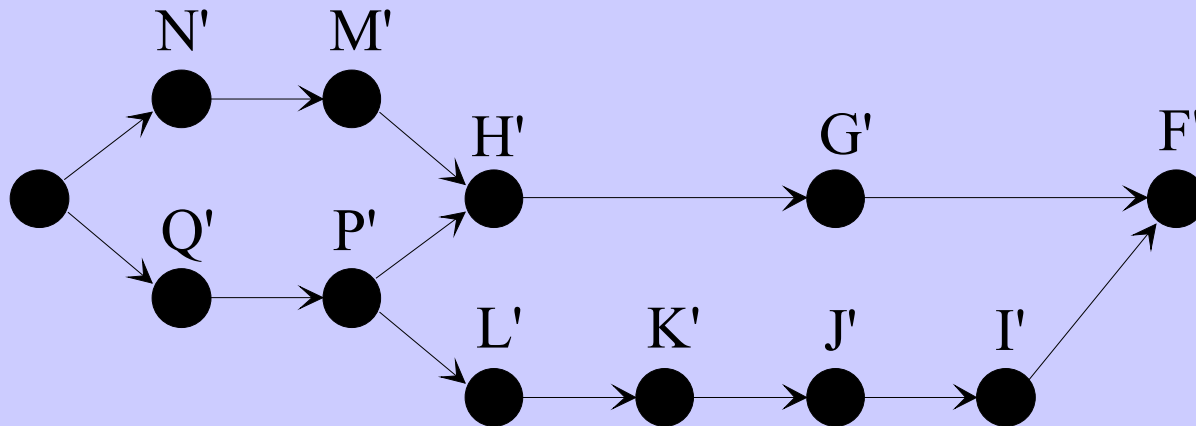
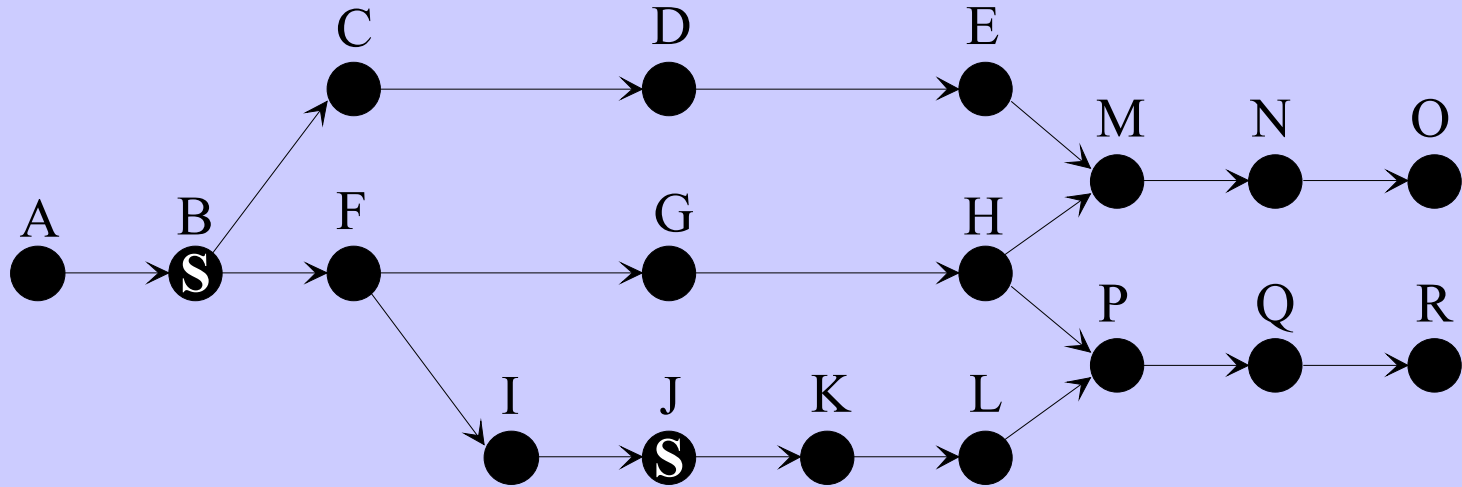
Execution Graph



Compensation Graph



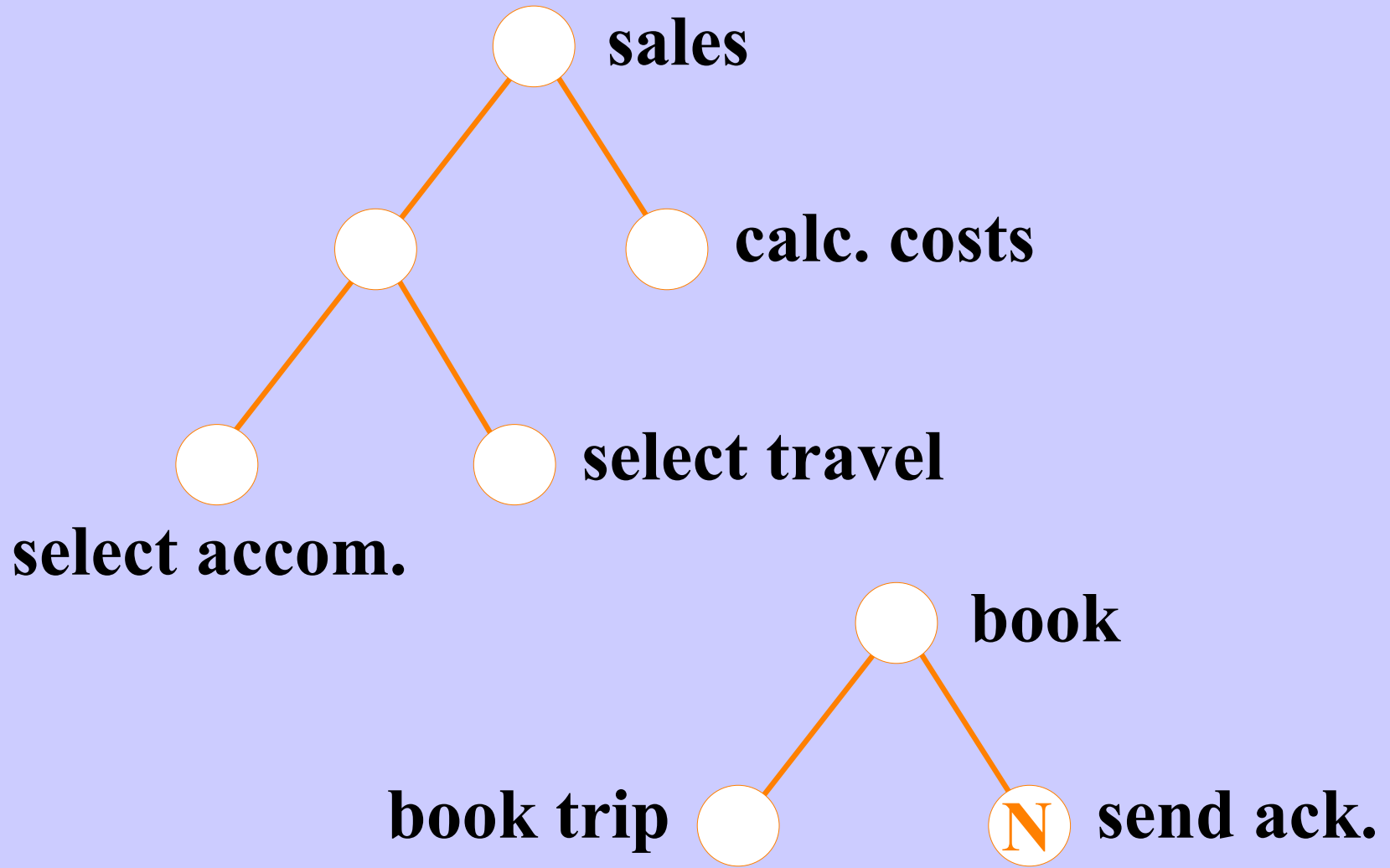
More Complex Examples



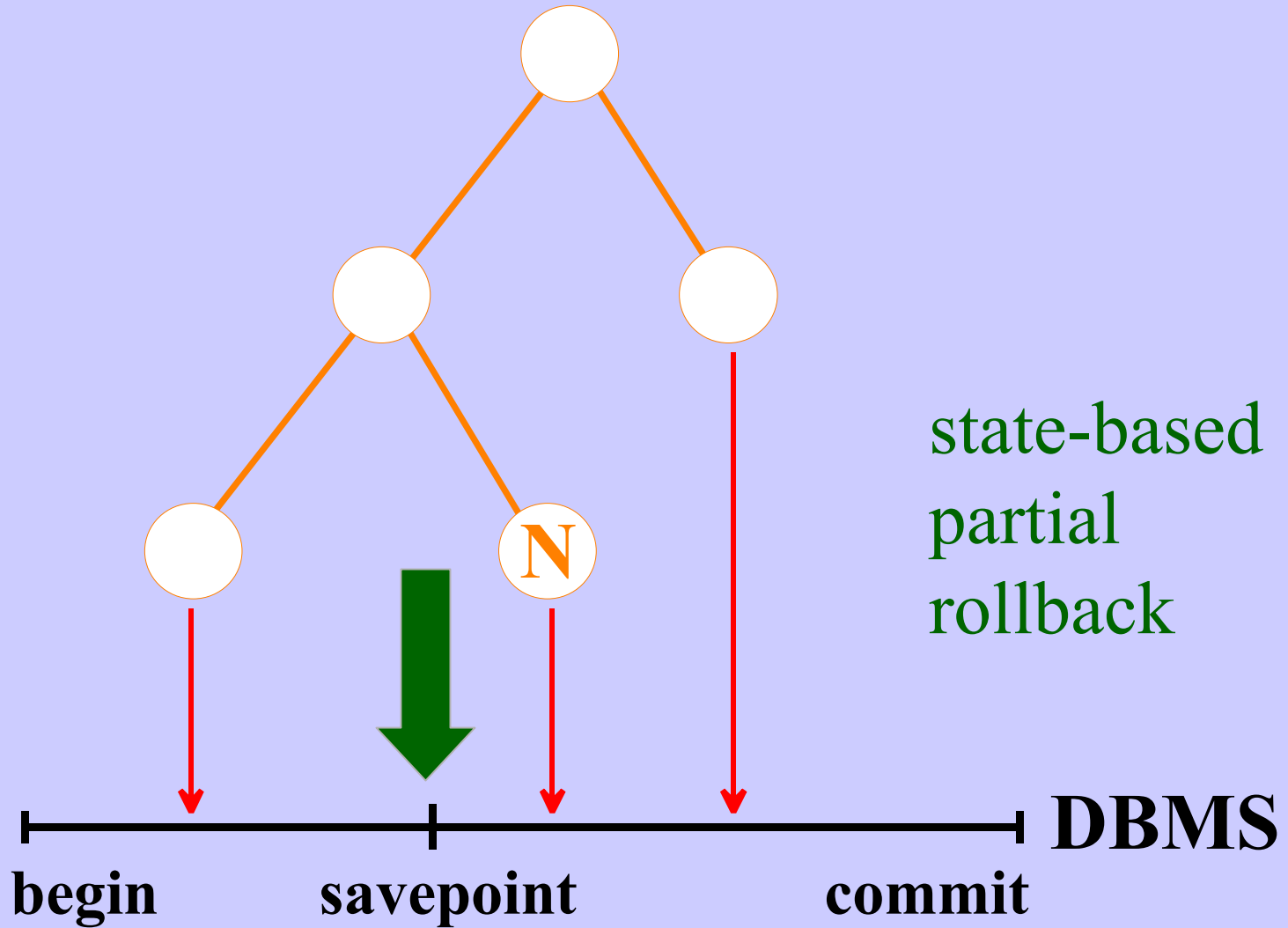
Local Transactions

- Tree of subtransactions, leaves are basic actions
- Intermediate results not visible to other transactions (isolated)
- Critical and non-critical subtransactions
 - Abort in critical propagates to parent
 - Abort in non-critical does not propagate
- Abort based on database states
 - Full abort to before state
 - Partial abort to explicit savepoint

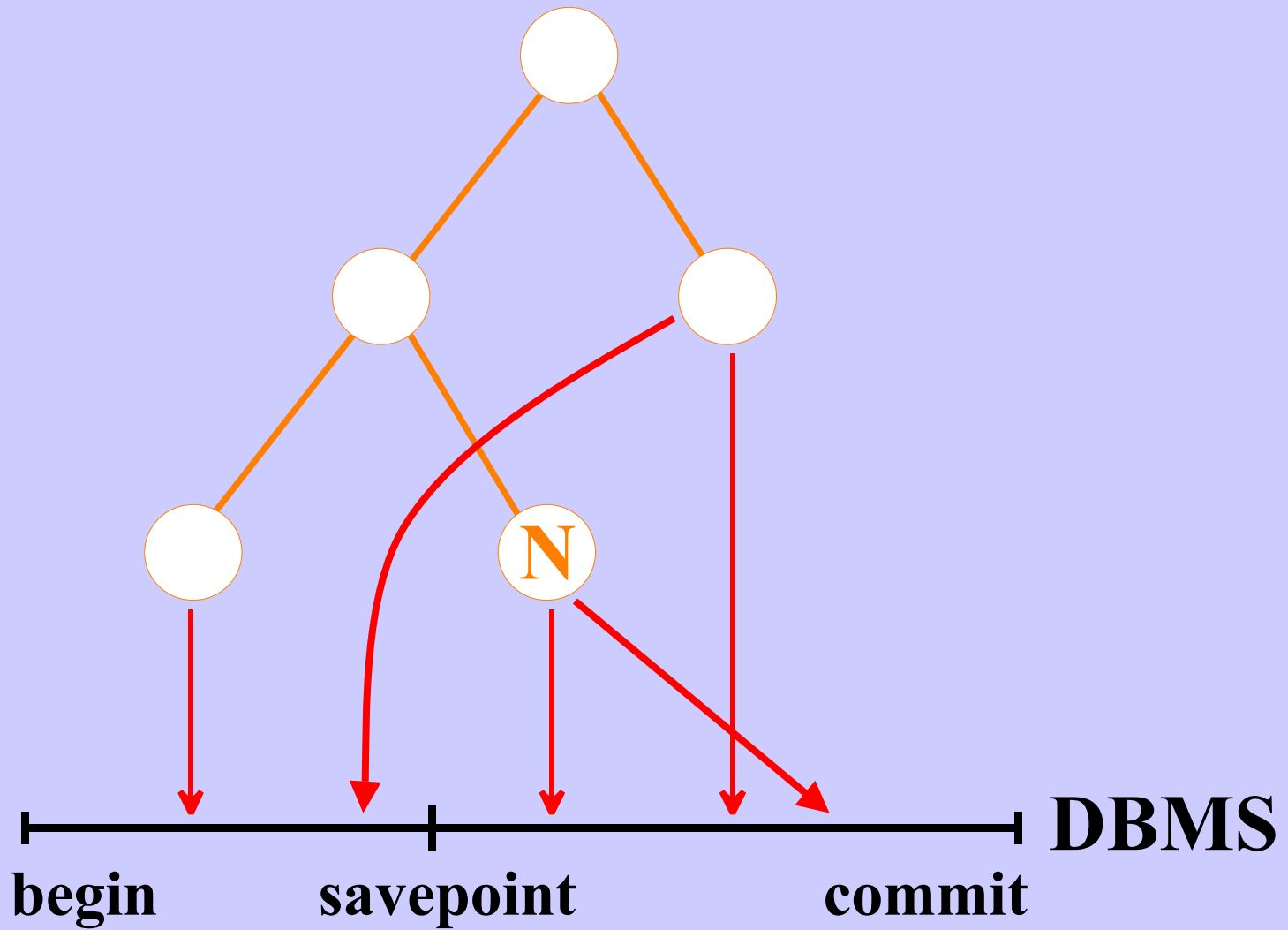
Example Local Transactions



Mapping to DBMS (1)



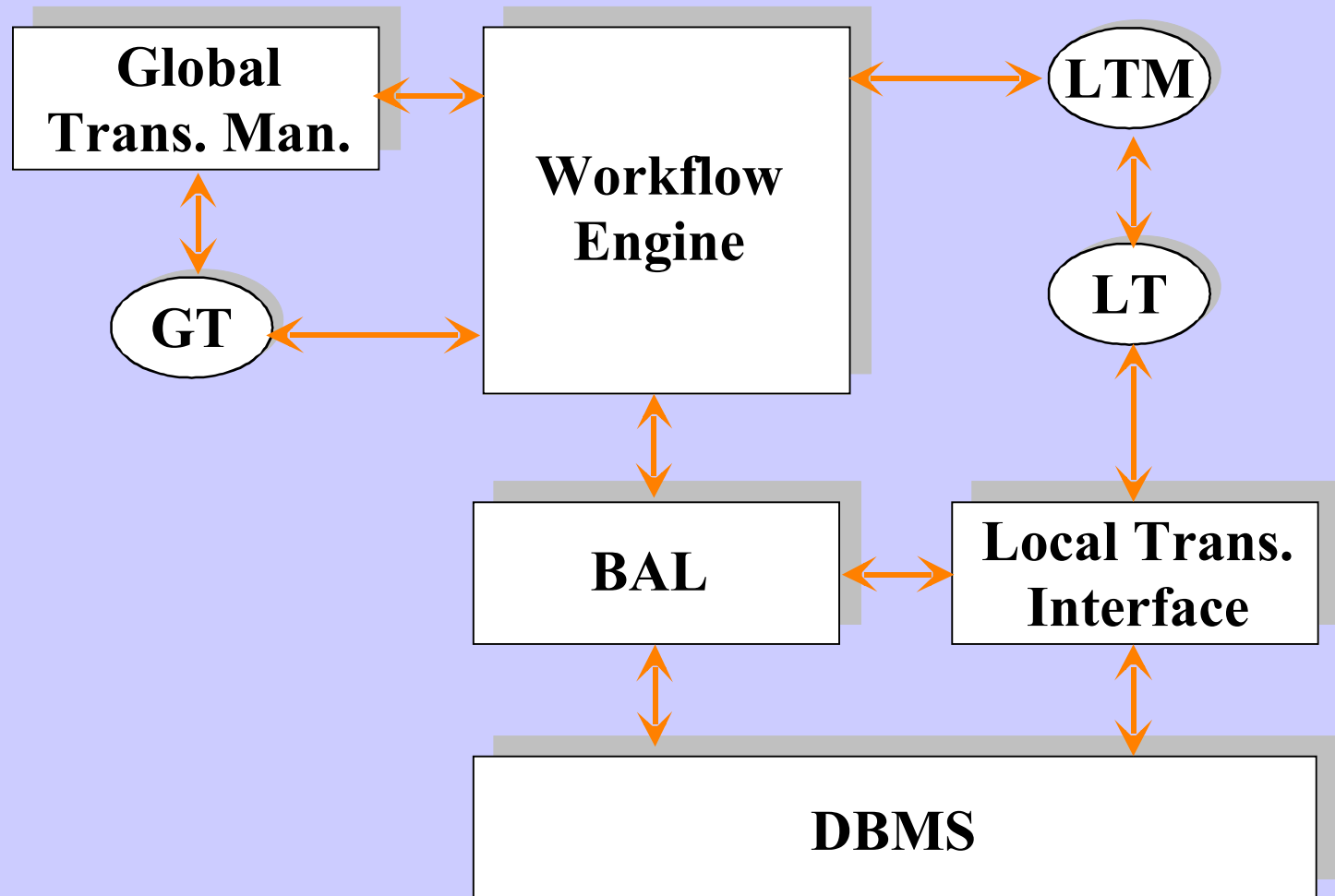
Mapping to DBMS (2)



Two-Level Transaction Model

- Global and local transaction model completely orthogonal
 - Local transactions are ‘black boxes’ in global transactions
 - Local transactions are ‘unaware’ of global transactions
- Modular support for global and local transactions
 - Separation of concerns in development
 - Flexible configuration & adaptation

Two-Level Transaction Management Architecture

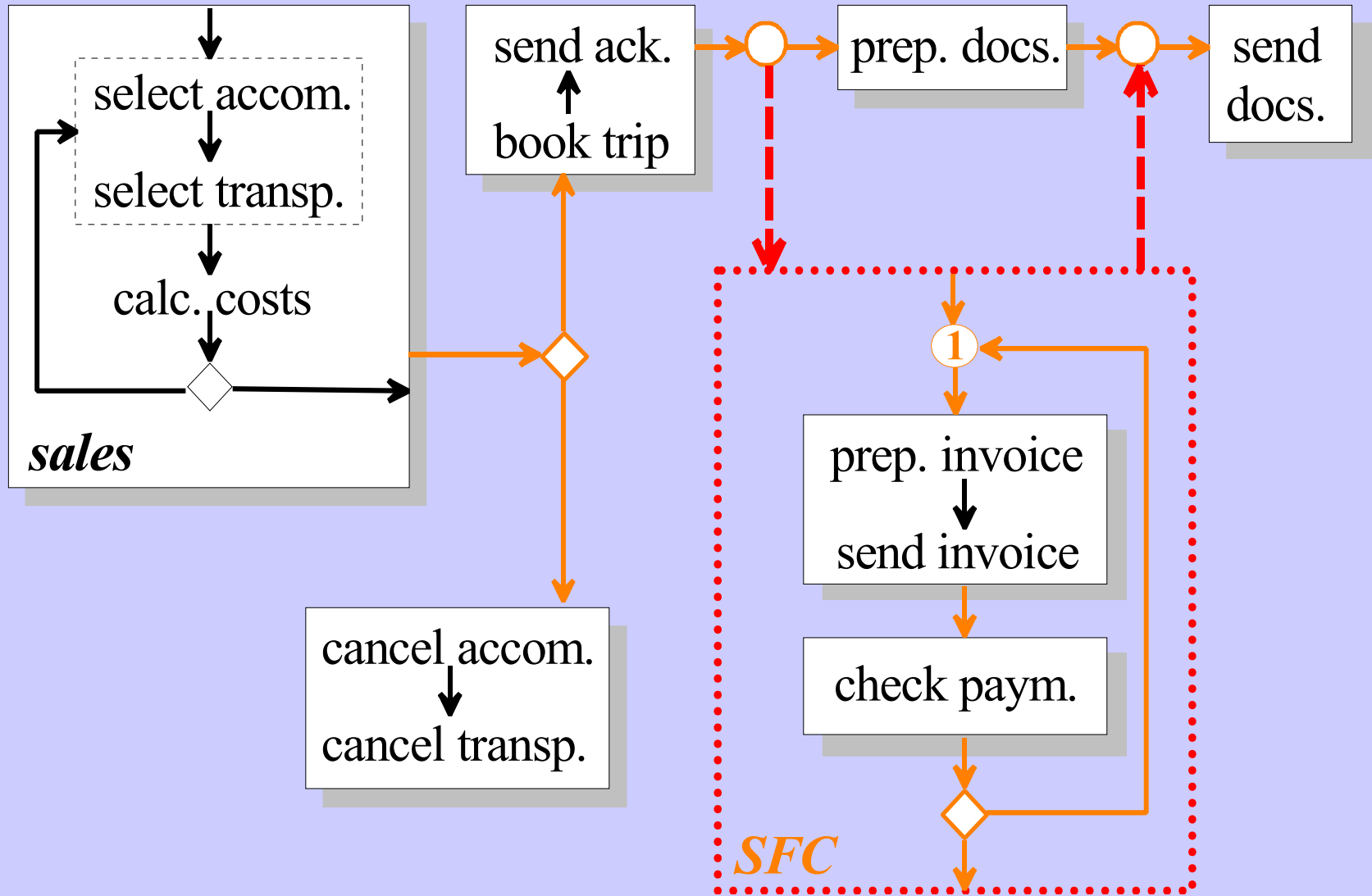


A Modern Times Approach to WF/EC Support

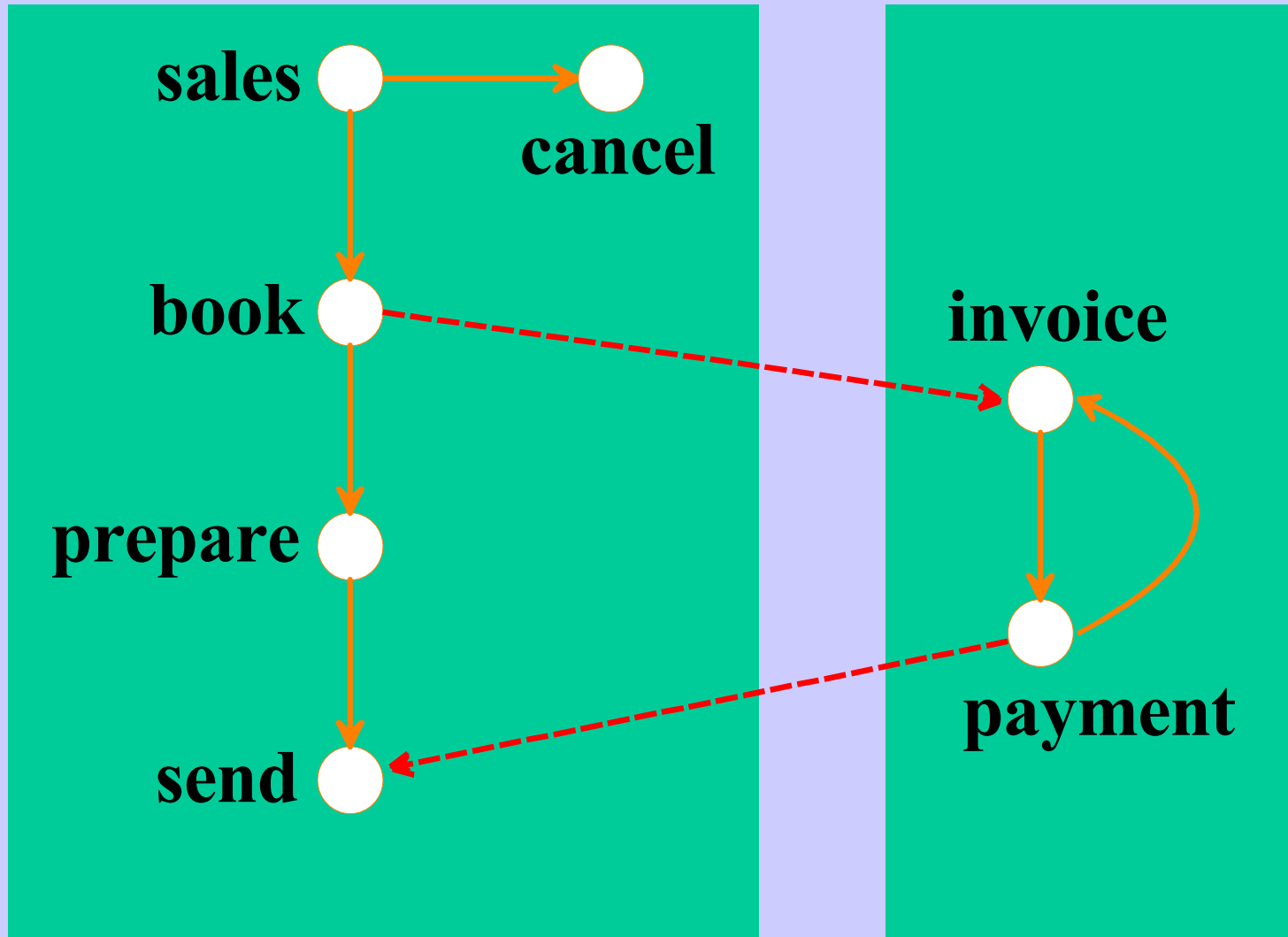
Cross-Organizational Workflows

- Cross-organizational workflows required in tightly-coupled virtual enterprises
 - e.g. production/value chains
 - e.g. service outsourcing
- Autonomy of organizations is main aspect
 - well-specified coupling of processes
 - hiding of non-public internals
- Dynamic virtual enterprises require dynamic cross-organizational workflow composition

Example XO Workflow



Example XO Global Transaction



Cross-Organizational Transactions

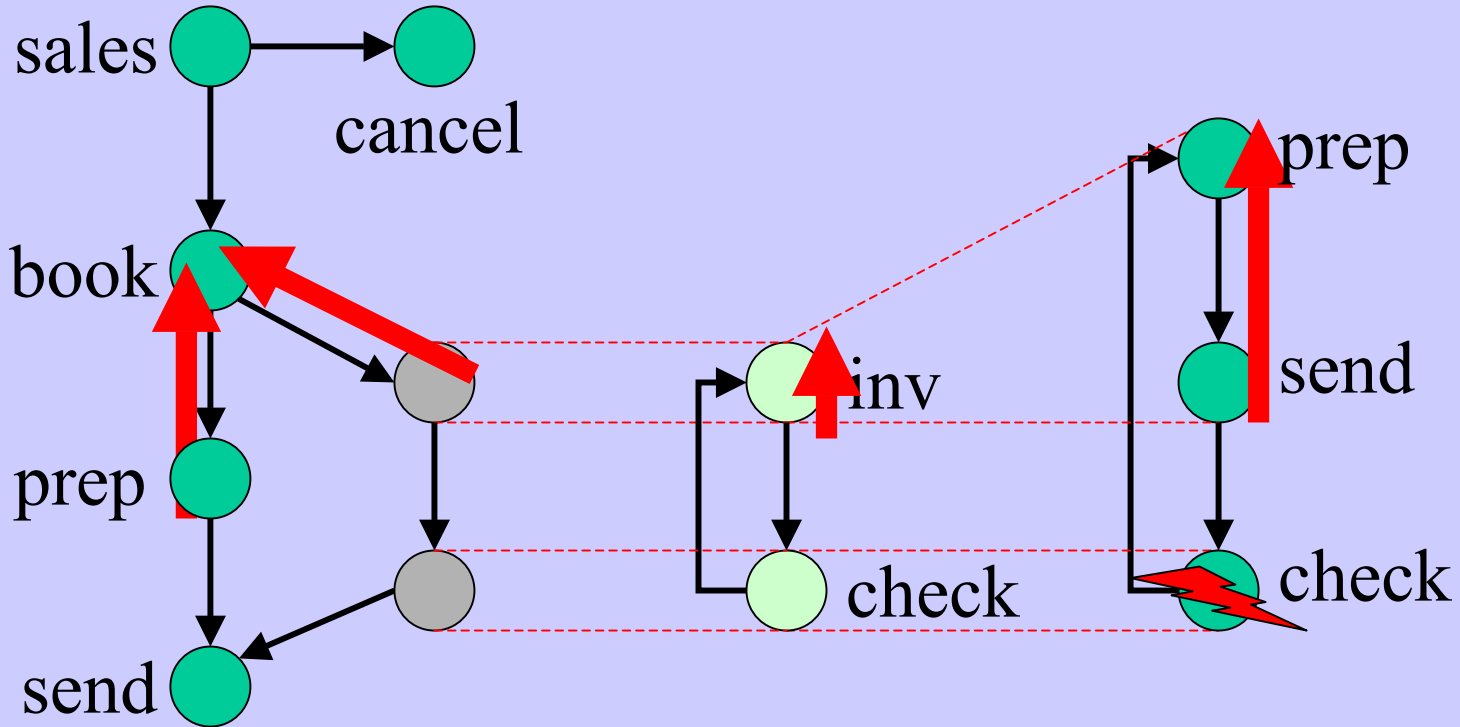
- Reliable workflow execution requires cross-organizational transaction management
- Autonomy of organizations is main aspect
 - ‘loose’ transaction models avoid too tight coupling of organizational processes
 - specification of allowed cross-organizational transactional operations
 - inter- and intra-organizational transactional semantics have to be distinguished

The CrossFlow Approach to WF/EC Support

CrossFlow XO Workflows

- Contract-based virtual organizations
- Service consumer ‘buy’ services from service providers via service traders
- Service enactment is based on dynamically linking workflow systems
- Contract states common process view of service
- Contract view embedded at consumer side
- Contract view implemented at provider side

CrossFlow Example Workflow

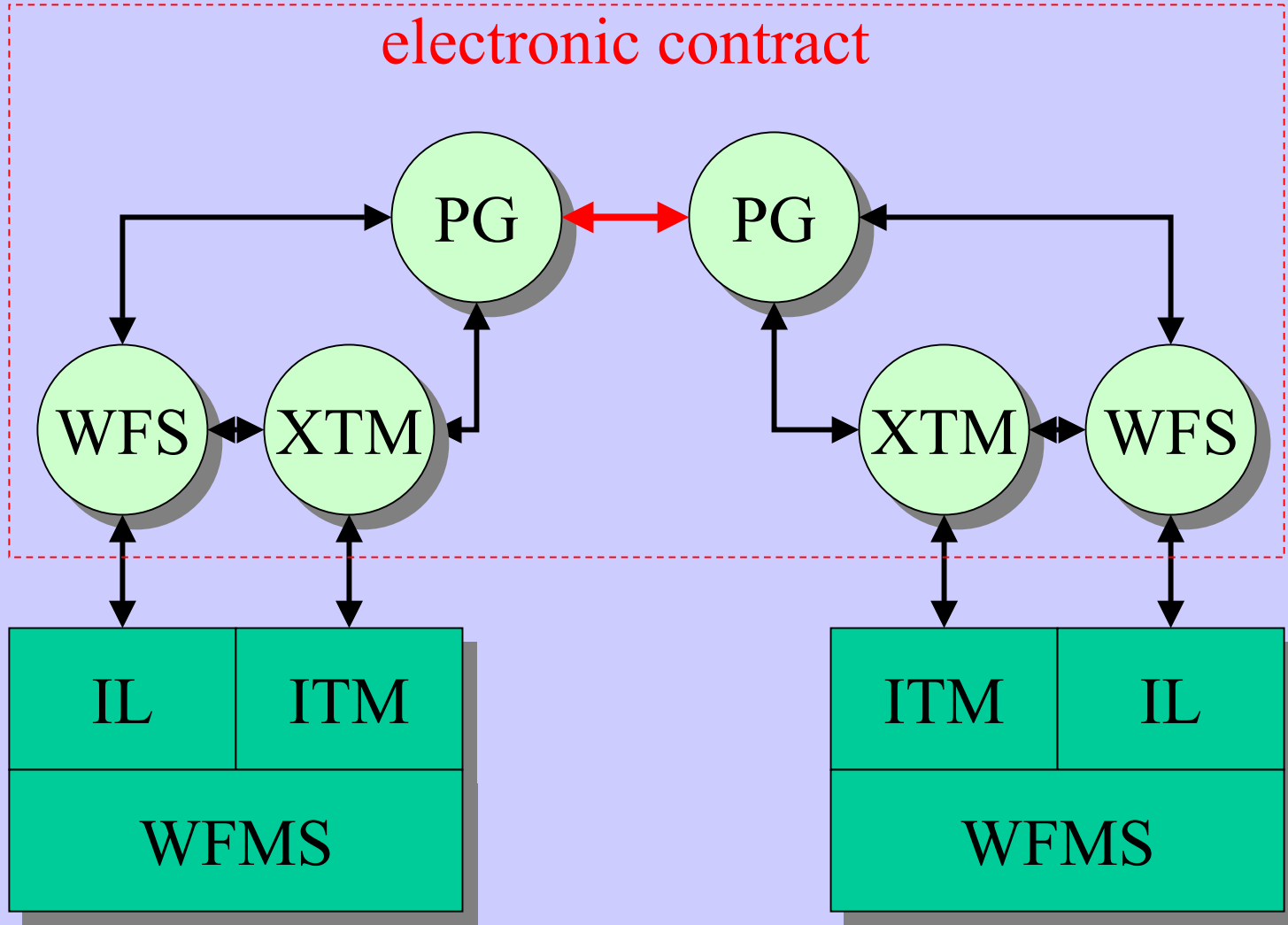


consumer
process
IO Transaction

provider
process
XO Transaction

provider
process
IO Transaction

CrossFlow Transaction Support



Conclusions from a History of Transactions

Conclusions (1)

- Transactions are a necessary component of the process aspect of all (database) applications in the business field
- So: transactions are indispensable for reliable process semantics
- Clear, not too complex semantics are required, both for system developers and application designers

Conclusions (2)

- High-level (workflow) processes require more relaxed transaction support, based on process semantics - e.g. saga-based transactions
- Low-level (workflow) processes require more strict transaction support, based on underlying database system - e.g. nested transactions
- Cross-organizational (workflow) processes require dynamic, inter-organizational transaction support