System Integration

Mini Case Studies © 2010

Concurrency and Case Study 2

Shawn A. Butler, Ph.D.
Senior Lecturer, Executive Education Program
Institute for Software Research
Carnegie Mellon University

Objectives

- Review the ACID properties of concurrency
- Understand why concurrency issues are important to a system integrator

Distributed Systems

- Today’s systems can be located almost anywhere and users are able to access their systems remotely
- Multiple data sources can provide service to remote clients
  - For load balancing
  - Performance
- Users want transparent access to data
- Updates in one source must be replicated to other sources
- Consistency is all about timing!

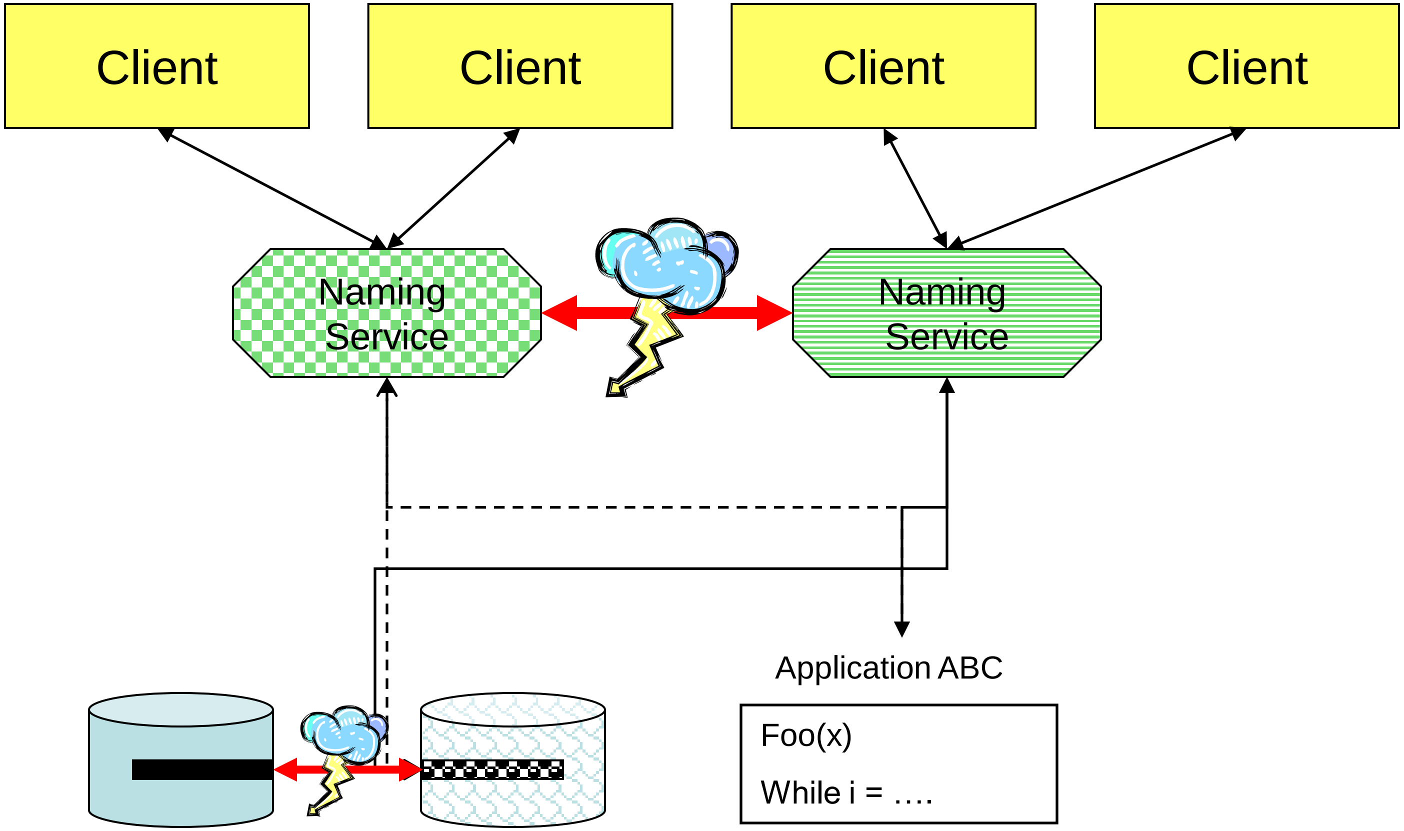
Replicated Data and Services

- Reduced risk through disaster recovery
- Provides load balancing
- Increases performance for the user
- Ensures availability
- Is much easier in today’s systems
- External media is very cheap
- Virtualization allows very fast recovery of services

Consistency

- Autonomous data sources are in a consistent state when a user can access data at any source and it will be the same
- After an update, how long before the system will be in a consistent state?
- What is the requirement? Immediately, within a few minutes, hours, or next day?
- Consistency requirements must be determined in distributed systems based on user requirements!
- The faster the consistency must be achieved, the more expensive the system

Maintaining Consistency



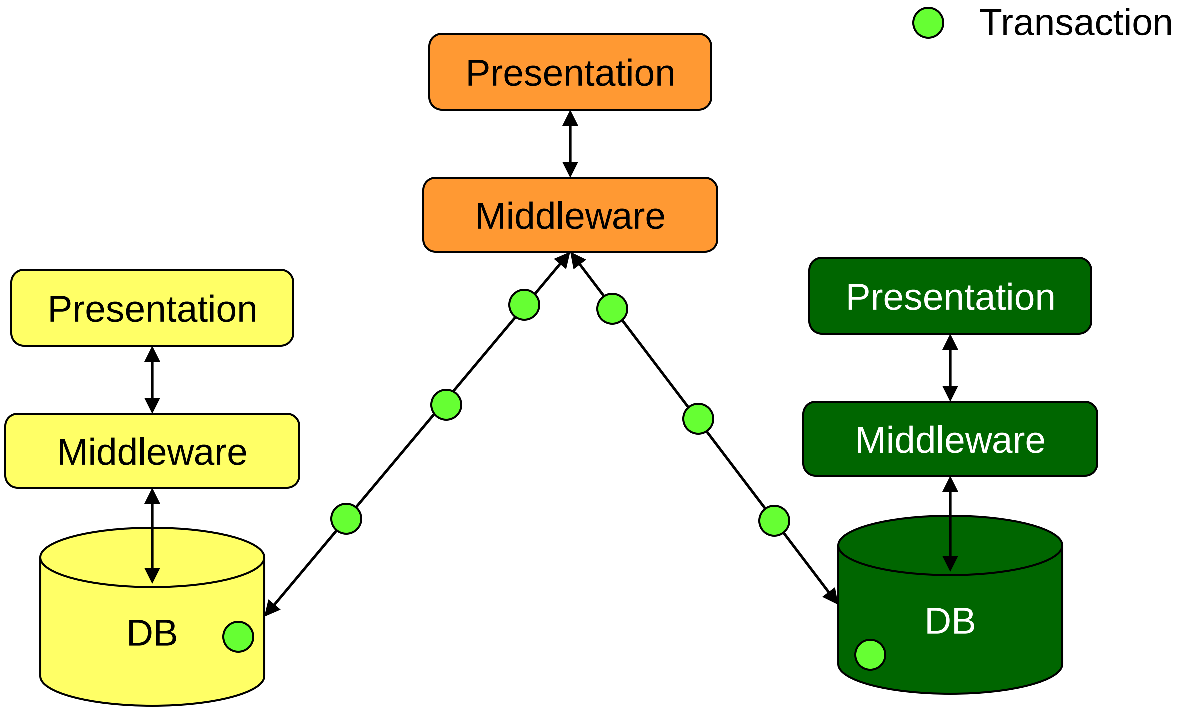
Information Consistency - ACID

- Atomicity – All tasks within a transaction are completed or no task is completed
- Consistency – Information must be in a ‘legitimate’ state at the end of a transaction
- Isolation – A transaction appears as a unit to other transactions
- Durability – Once the transaction is finished, it will persist

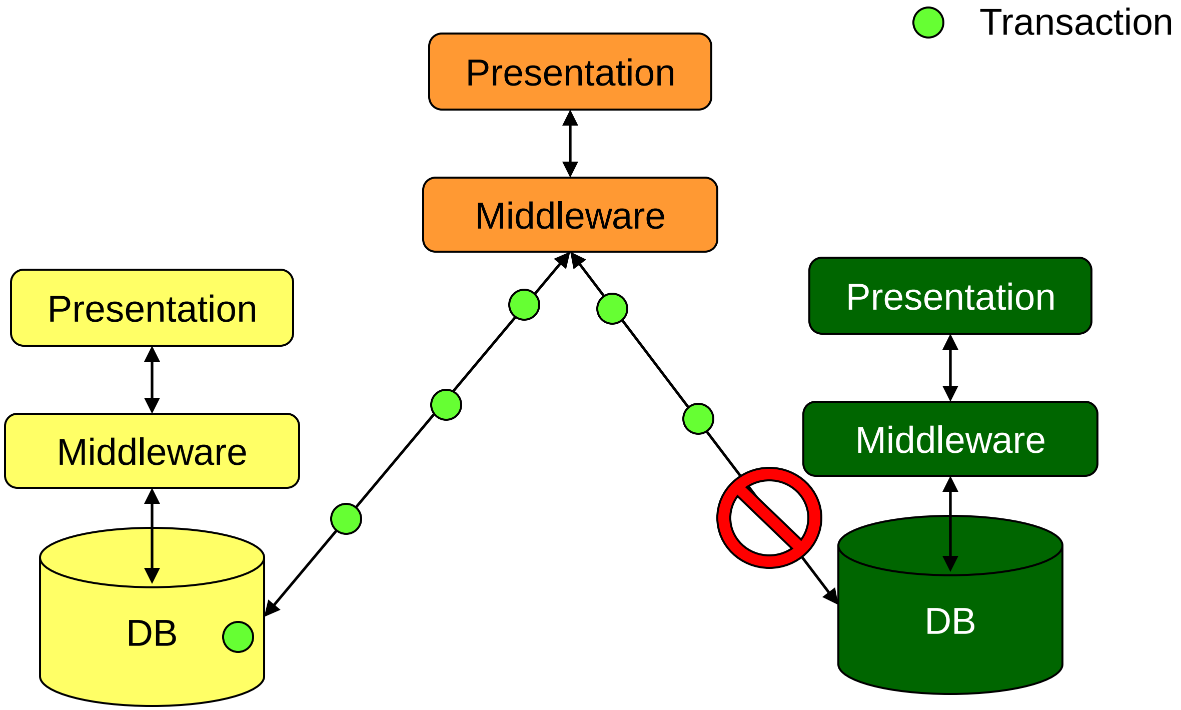
Why Do We Care?

- Modern Transaction Processing products ensure information consistency
  - Not always – Not all system objects covered
  - You may want to relax consistency requirements to improve performance
  - Cross domain transactions create integration problems
    - Different transaction processing products
    - Different assumptions about consistency

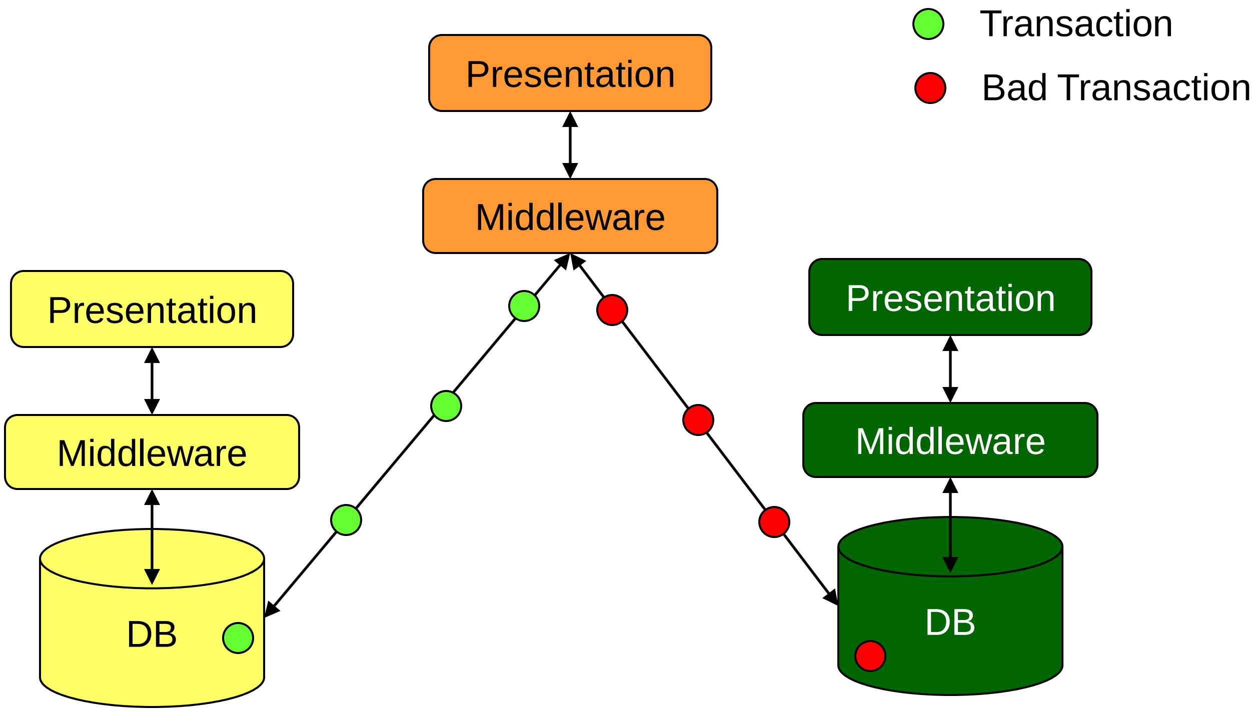
Atomicity


● Transaction
Presentation
Middleware
Presentation
Middleware
DB
Presentation
Middleware
DB

Atomicity


Transaction
Presentation
Middleware
Presentation
Middleware
DB
Presentation
Middleware
DB

Consistency



🟢 Transaction

🔴 Bad Transaction

Presentation

Middleware

Presentation

Middleware

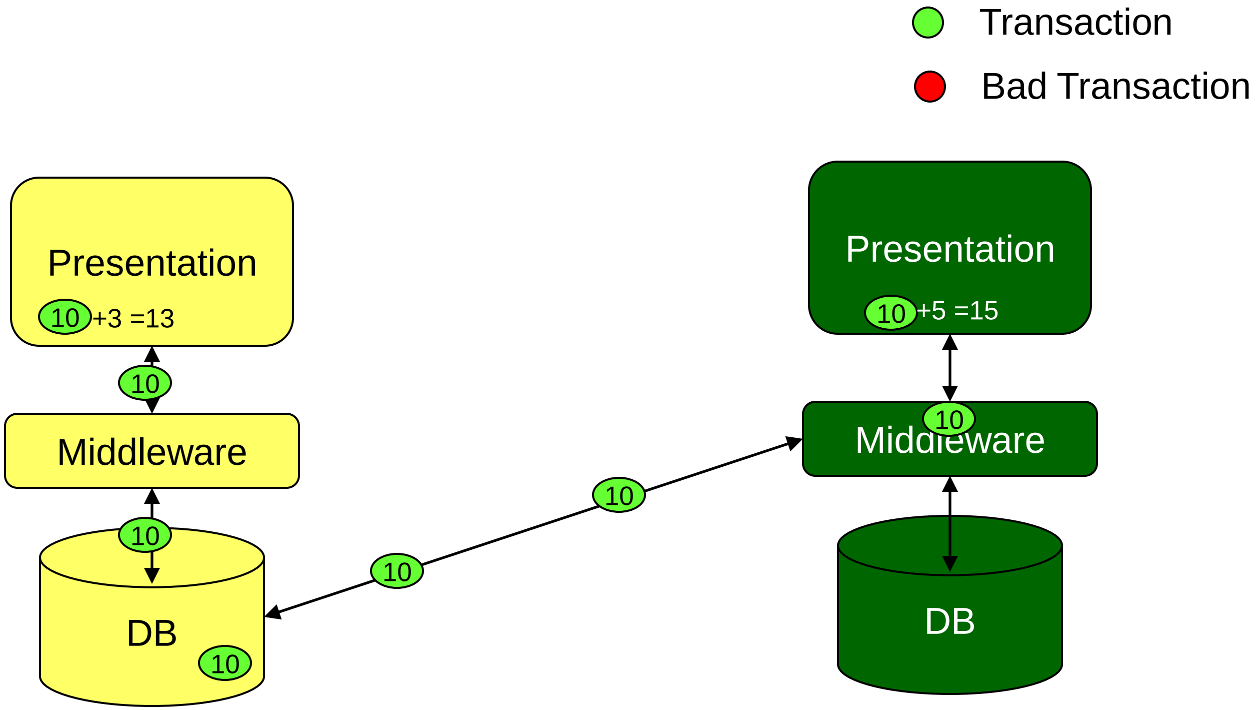
DB

Presentation

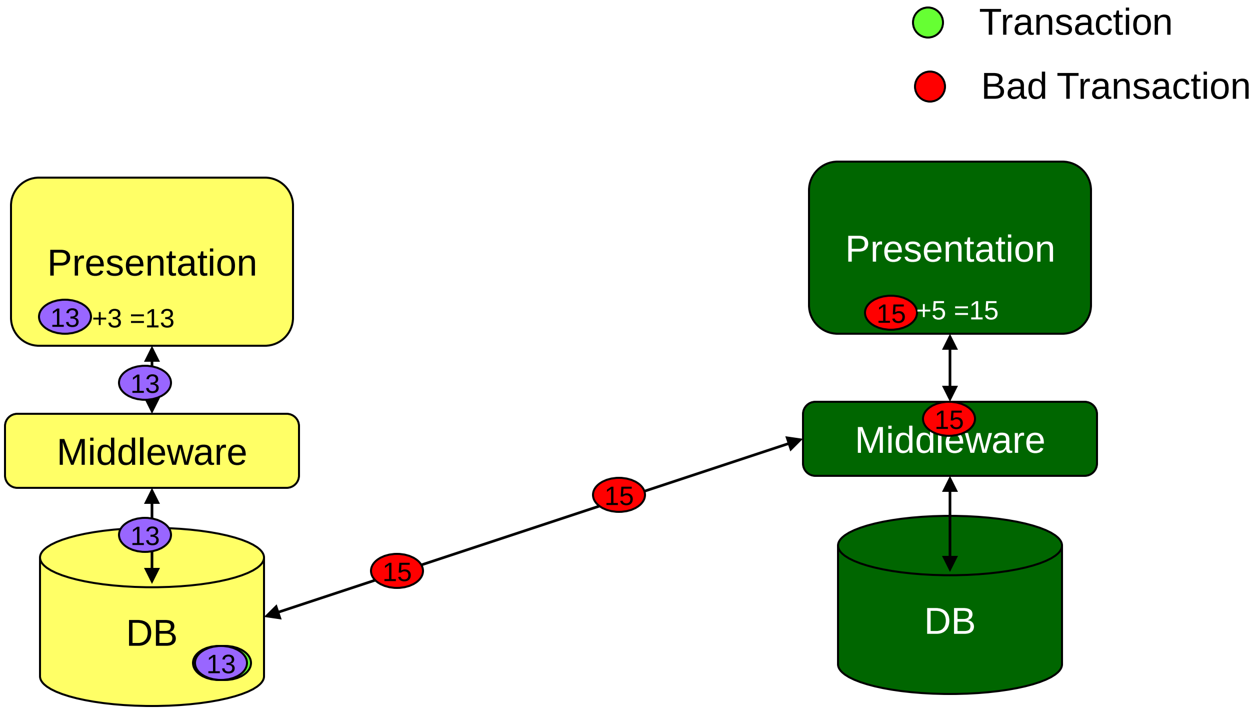
Middleware

DB

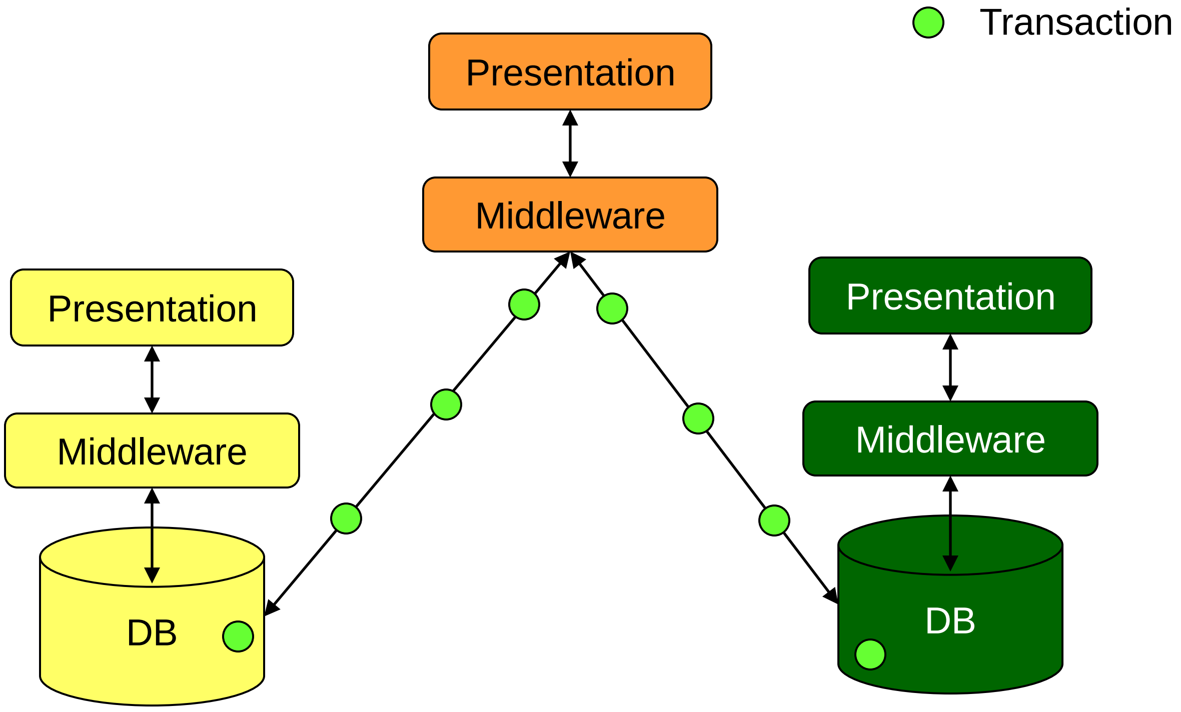
Isolation


● Transaction
● Bad Transaction
Presentation
10 +3 =13
10
Middleware
10
DB
10
10
10
Presentation
10 +5 =15
10
Middleware
DB

Isolation


● Transaction
● Bad Transaction
Presentation
13 +3 =13
13
Middleware
13
DB
13
15
15
Presentation
15 +5 =15
15
Middleware
DB

Durability


Transaction
Presentation
Middleware
Presentation
Middleware
DB
Presentation
Middleware
DB

Heuristic 2

Build and maintain options as long as possible
in the design and implementation of complex systems.
You will need them.

Summary

- System engineers have to ‘engineer’ for concurrency issues
- Systems engineers have to make tradeoffs between performance and global consistency
- ACID properties are generally associated with databases and transaction processing, but they are important when integrating different types of systems
- Distributed and redundant services improve reliability, but increase complexity

Case Study 2: Integration

- ACME Executive Management would like to see a ‘dashboard’ of important information that will help them make informed decisions about ACME’s employees.
- ACME provided a list of key requirements for the ‘dashboard’, however, the design of the GUI is the responsibility of the team.
- The Executive Management team will not update the data, the data is for information purposes only.
- The Executive Management team will want to “drill down” into the data presented.
- The payroll and HR applications must remain unchanged!
- The application should be thoroughly tested since it would not look good for the development team to have buggy software for such an important level of user.

Deliverables

- Develop an integrated system using the presentation model
- Test the system to determine the limitations of this model and data consistency
- Demonstrate your integrated system and show that the system meets the CEO’s requirements
- Turn in your documentation, software and test results