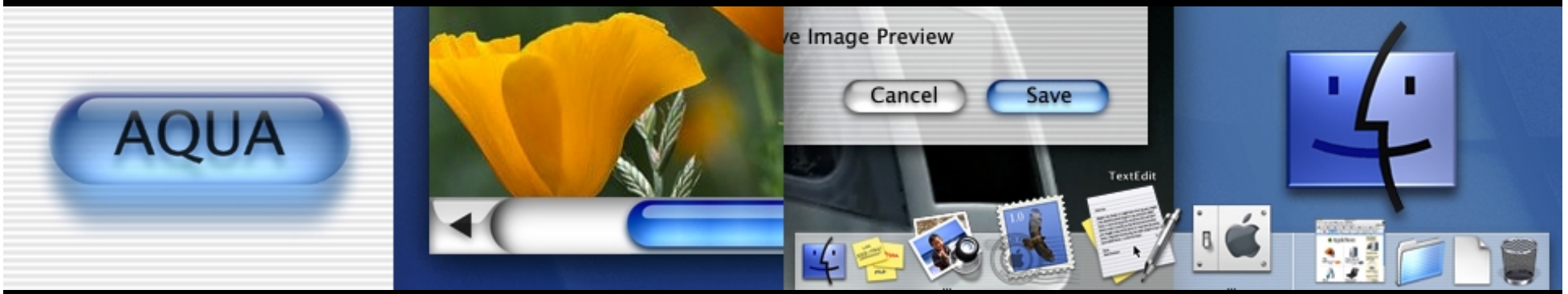
Session 417

Building Large-Scale WebObjects Applications



Steven Meyer
Consulting Engagement Manager, Apple iServices

Introduction

- Large, complex projects are becoming more common as corporations and government embrace the Web
- Large-scale projects are not just big simple projects

What You'll Learn

- How WebObjects fits into traditional large-scale development
- How to optimize WebObjects development for large-scale projects
- Pitfalls to avoid in developing large-scale projects

Large-Scale Projects

If your project has. . .

- Large project team
- Large code base
- Large amounts of data
- Large user base
- Long development/deployment time frame

. . . you have a large-scale project

General Engineering Practices

- Always needed for large projects
- WebObjects fits right in

General Engineering Practices

Project management

- Staffing
- Communication
- Scheduling
- Requirements tracking

General Engineering Practices

Source code control

- Essential for large projects
- Simplifies backup
- CVS the most common system in our projects
  - Optimistic locking—much more convenient for large projects

General Engineering Practices

Bug tracking

- Need the following features
  - Web access
  - Problem priority, Fix in release/Fix priority, responsibility, audit trail
  - Should support both bugs and feature requests

General Engineering Practices

Build/release management

- Build/Test team
- Fixed internal build schedule
  - Weekly or biweekly
  - Helps keep development on track
- Separate external release schedule

WebObjects Engineering Practices

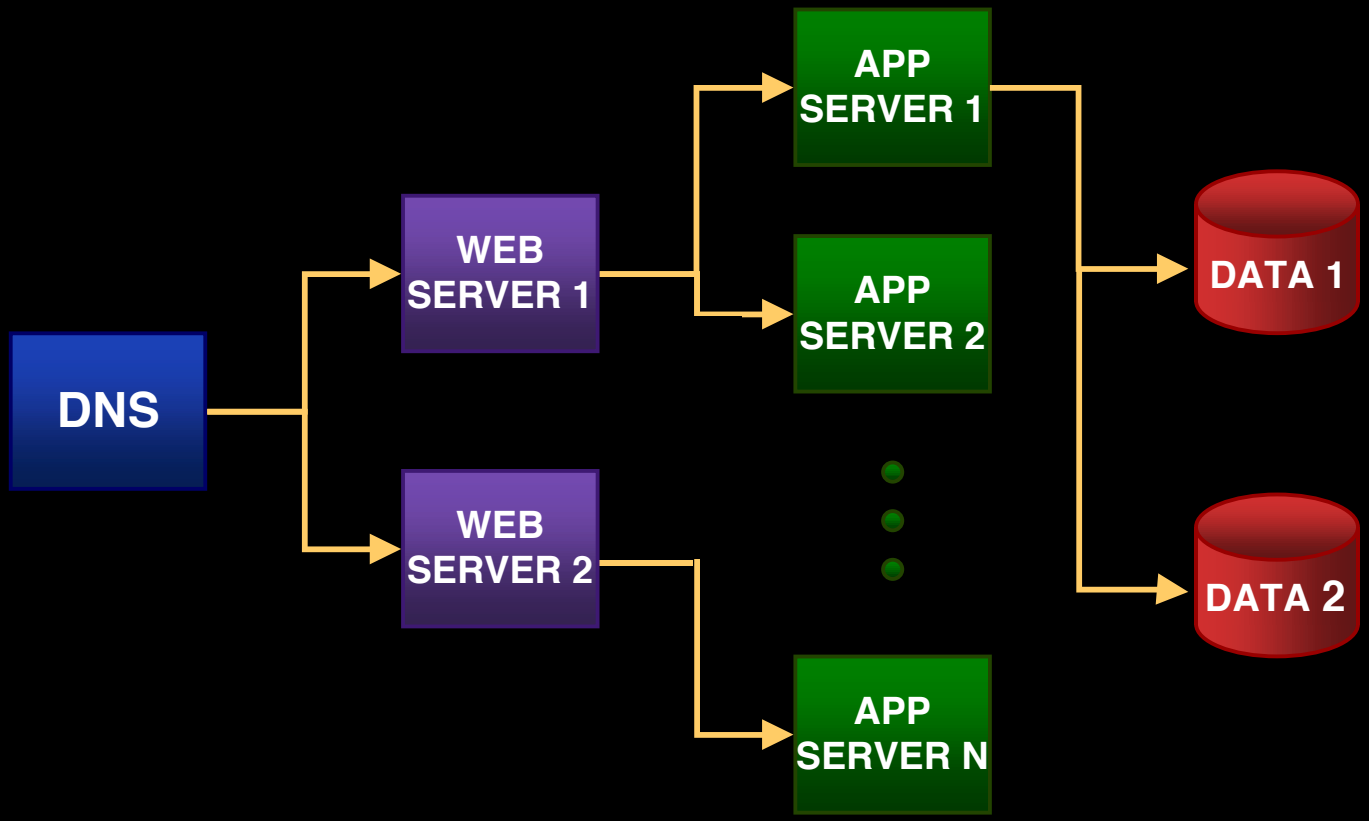
Documentation

- Requirements
- Specifications
- Design
- Deployment/System administration
- Coding
- User

Hardware Architecture

- Multiple web servers
- Multiple application servers
- Multiple data sources

Hardware Architecture



Software Architecture

- Already dealing with data caching and synchronization issues due to hardware architecture
- Best to separate functionality into multiple applications
  - Read-Only/Read-Write
  - Public/private/admin
- Separately configurable numbers of instances to match different kinds of load

Application Design

Typical Code Organization

- Data frameworks
- "Fixes" frameworks–keep an eye on this
- Architectural frameworks
- Application # 1
- Application # 2

Team Organization

- Typically architecture, data modeling, application # 1, application # 2
- Make sure to spread the senior staff around the teams
- Make sure you have enough *planned* communication among the teams

Data Modeling

- Model the business process, not just the business data
- If you're dealing with relational data sources, don't over-normalize
- Be cautious with entity inheritance

Abstraction of Data Interfaces

- Generally a good idea
- Weaknesses
  - Too much abstraction limits performance optimization
  - Attempting to abstract dissimilar sources leads to a lowest common denominator
  - Tendency is to abstract just to match your current needs

Design for Reuse

- Best for reusable interface components
  - Custom versions of elements
  - Configurable combinations of elements
  - Examine the WOExtensions framework
- Enterprise Objects
- Organize into distinct frameworks—try to limit dependencies

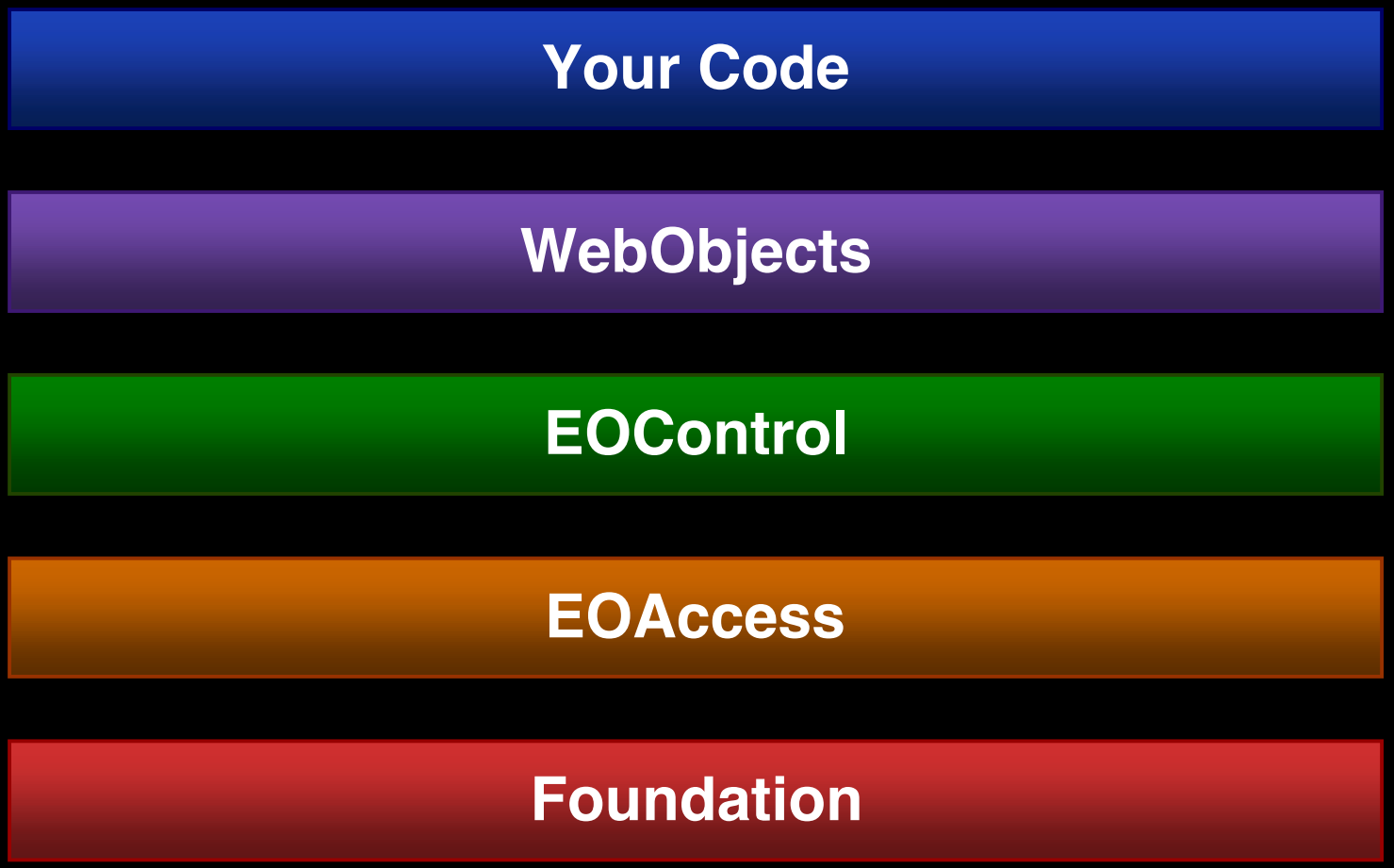
Multithreading

- As a general software technique, multithreading is difficult
  - Hard to write
  - Even harder to debug
- Benefits of multithreading are hard to predict
  - Need to be aware of bottlenecks in both your system and external systems
- If you have to do it, WebObjects supports it

Potential Warning Signs

- You're subclassing a lot of Apple's classes
- Your classes are intertwined with Apple's classes

Ideal Code Hierarchy



Ideal Code Hierarchy

More Potential Warning Signs

- You have large Application and Session classes
- You're using Objective-C in a Java project
- You're working with the EOAccess framework
- You have custom key-value coding handling

Prototyping

- User interface—part of requirements and specification
- Technical—keep it independent of interface
  - Small, single function
  - Throw it away

Construction

- Assume the project will continue after you're gone
- Standardize keywords for comments
- Document, document, document

Comment Example

- The kind of comment I don't like to see

```
...
/* Hack to fix index problem */
/* Hmm - we may not need this anymore - SDM */
myEODatabaseSubclass.fixUpArticleIndexCache();
...
```

Comment Example

- The kind of comment I like to see

```
...
/* FIXME   WEBOBJECTS_3.5  SYBASE_10.3
   Need the following to fix a problem where the
   caches get out of sync after the stored procedure
   call to rebuild our article indexes.  Can be removed
   if we go to Sybase 11, since we won't need the stored
   procedure call.  Needs to be reviewed  after next
   WebObjects release to see if they add an official way
   to do this. Steven Meyer, 4/11/1998 */
myEODatabaseSubclass.fixUpArticleIndexCache();
...
```

More Construction

- The meetings have a purpose
- Clever is bad

Clever Is Bad

- You can fool yourself
- You can fool your team
- You can fool us

Testing

- Properly needs a complete staging environment
- Use automated tools for regression test and load testing
- Use humans for functional testing
- Schedule regular bug tracking meetings

Performance

- “How many users can I support?”
- “How many machines do I need?”
- “How many instances do I need?”

Performance Enhancement

- Speed up what is slowest
- Look first to data access
  - Modify data model – especially pre-fetch and batch fetch
  - Consider custom versions of data model for certain applications
  - Manually fetch the data you need

Performance Enhancement (Cont.)

- Data access
  - Limit complicated bindings

```
MyComponent.wod
...
String1:WOString {
value = selectedEmployee.manager.spouse.car.color;
}
...
```

- Make sure you don't transfer large data to the browser unnecessarily

Deploying

- Deployment team
- Gather as many statistics as you can without impact to performance
- The application is not finished when you deploy

What Went Wrong?

- If you never make it to deployment, bad management
  - Poor planning
  - Poor tracking
  - Poor tool selection
- If you're stuck at version 1, bad engineering
  - Poor design choices
  - Poor documentation

What Went Right?

- If you field version 1, credit management
- If you field version 2 without a complete rewrite, credit engineering

Summary/Conclusion

- Big projects are hard
- Big projects are worth the effort

Roadmap

915 WebObjects Feedback Forum

Room J2
Next

For More Information

http://www.apple.com/webobjects

http://enterprise.apple.com/wwdc2000

Visit the WebObjects lab downstairs!
Everyday from 11:00 a.m.–2:00 p.m.

Try out your WebObjects 4.5 Evaluation CD!

Who to Contact

Toni Trujillo Vian
Director, WebObjects Engineering
wofeedback@group.apple.com

Ernest Prabhakar
Product Line Manager, WebObjects
webobjects@group.apple.com

Session 417

Q&A

WWDC

Worldwide Developers Conference 2000

Think different.