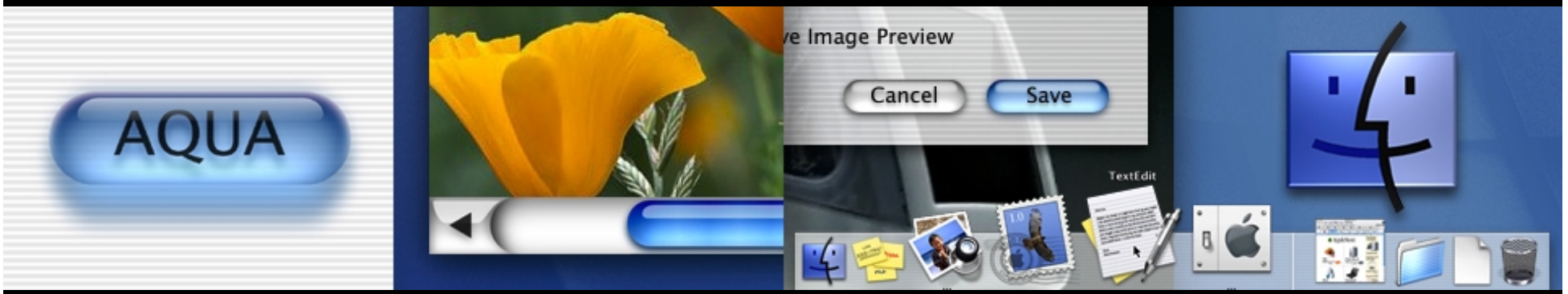
Session 414

WebObjects Performance Metrics



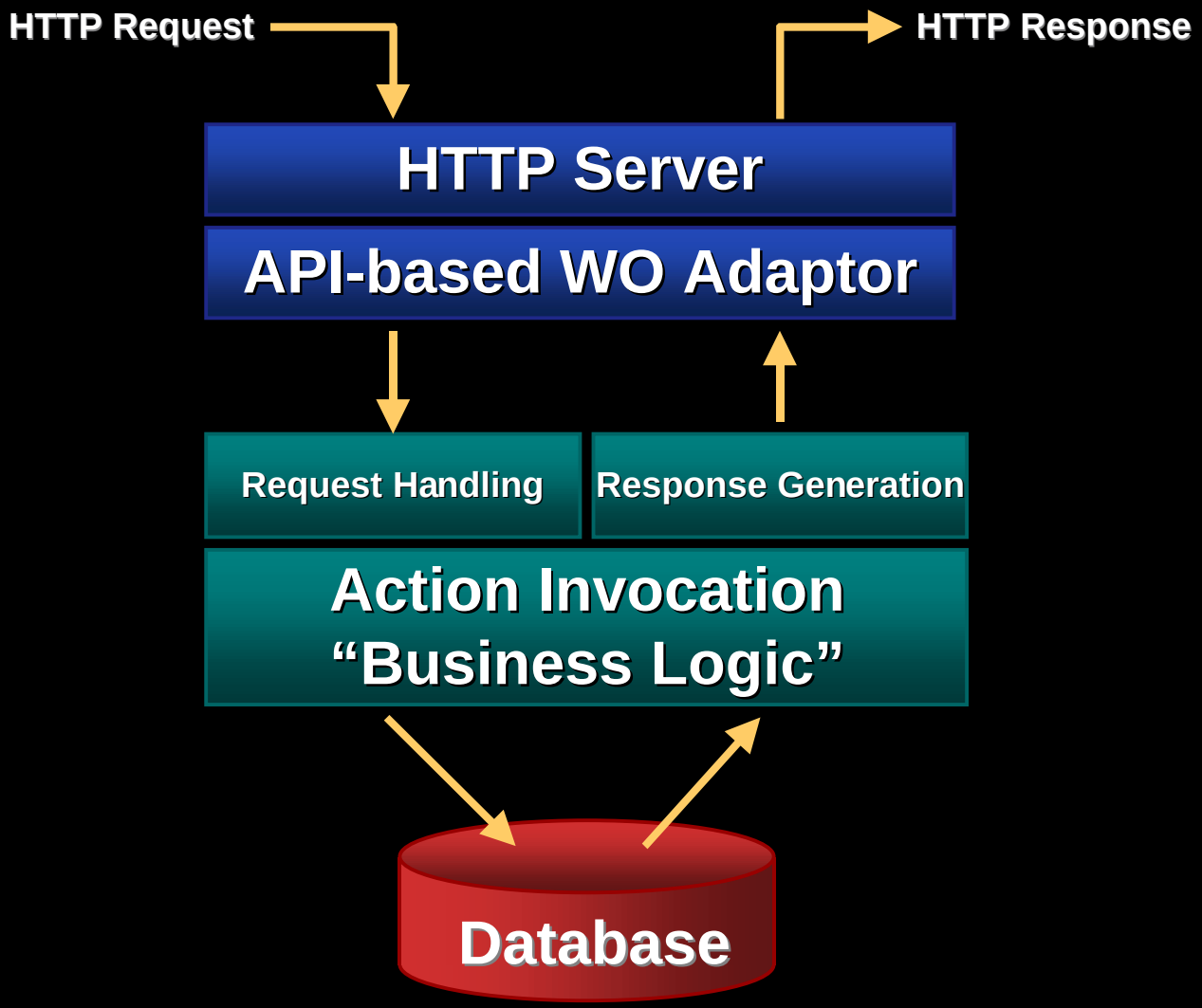
Eric Bailey
Internet Services, Apple

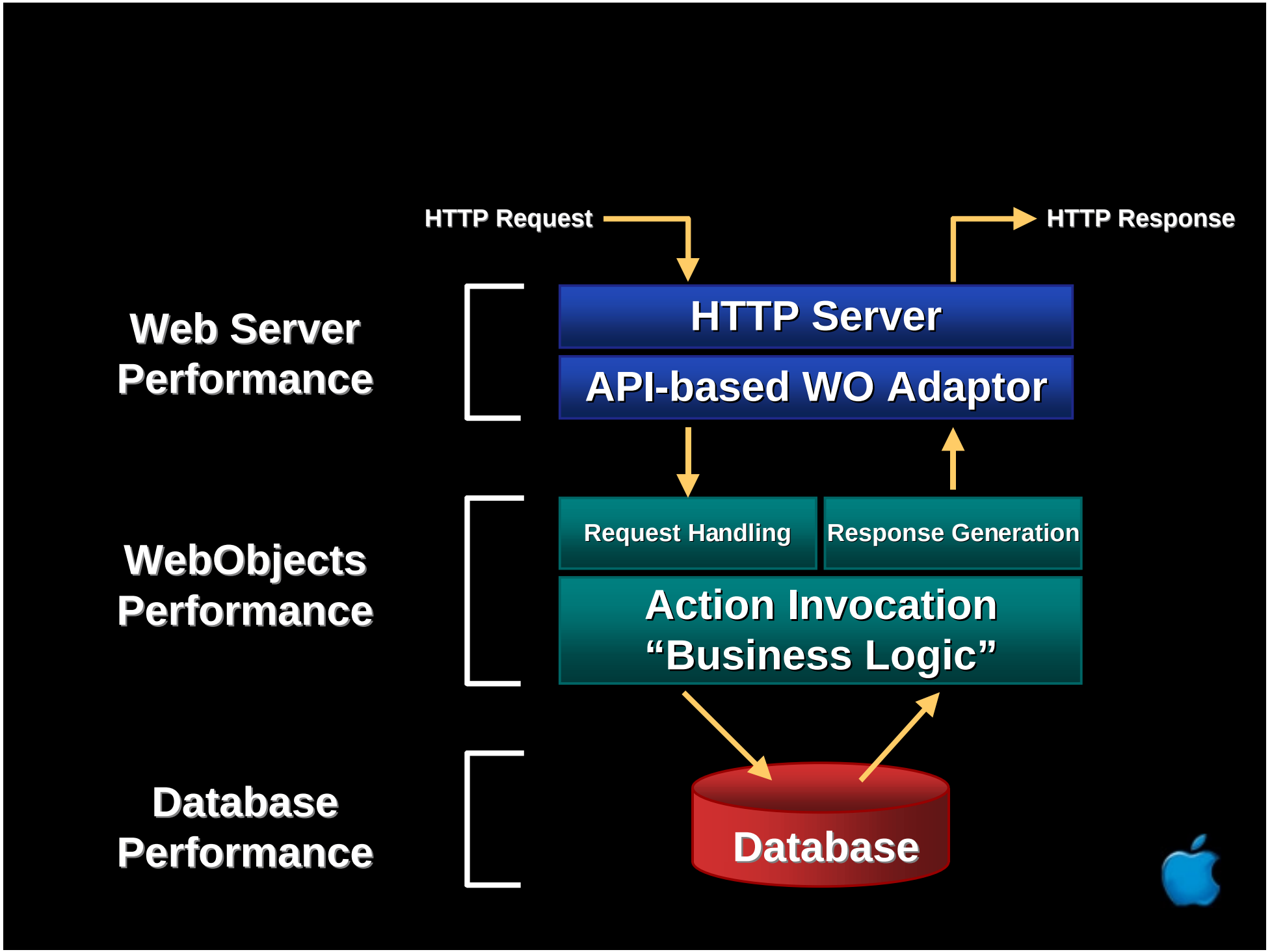
Stan Jirman
WebObjects Tools, Apple

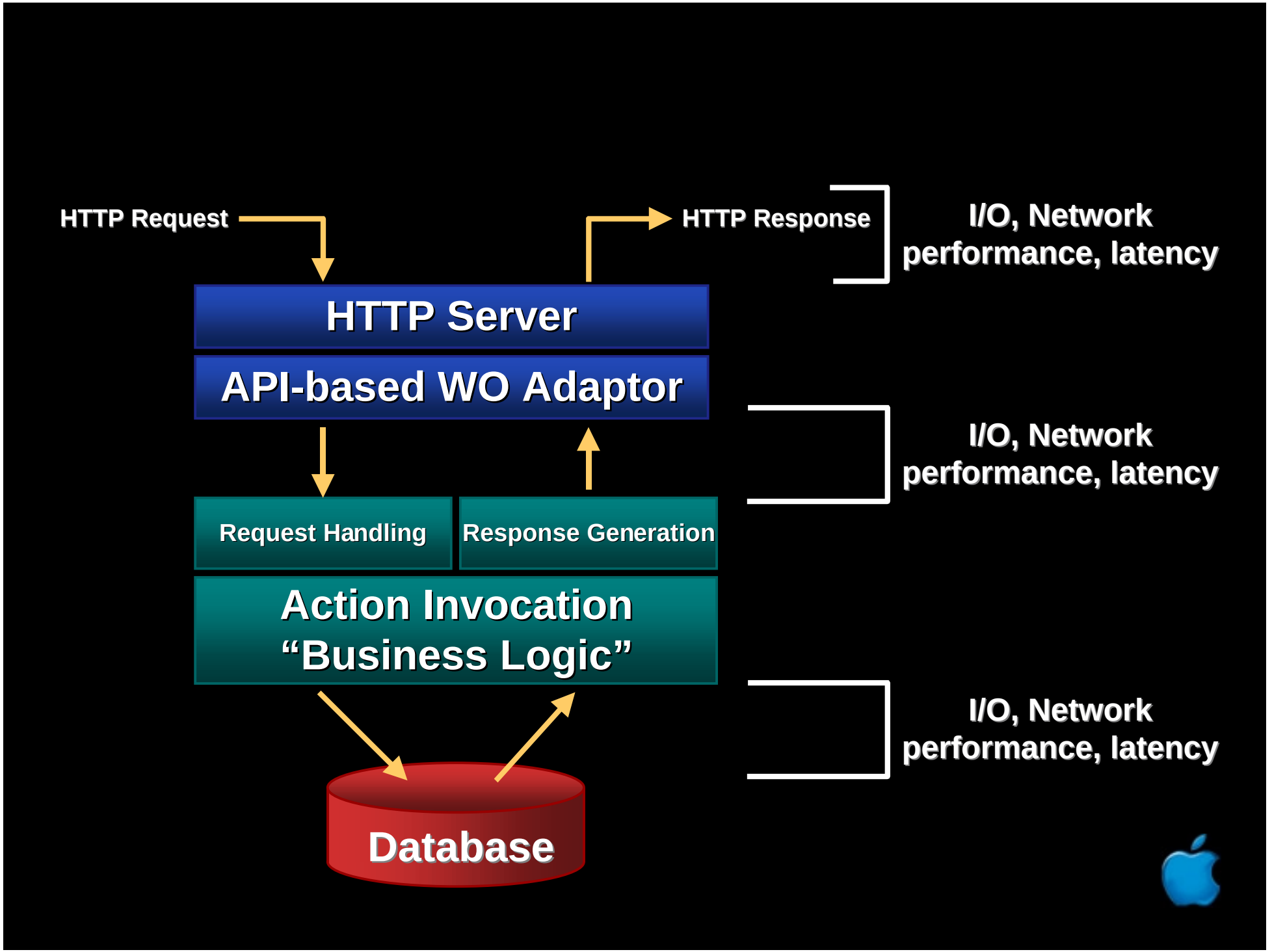
Introduction

- Use tools and techniques to measure your application's performance characteristics
- Discover factors that influence a WebObjects application's performance
- Pinpoint the critical code in your application that needs attention

The Big Picture




HTTP Request
HTTP Response
Web Server Performance
HTTP Server
API-based WO Adaptor
WebObjects Performance
Request Handling
Response Generation
Action Invocation
“Business Logic”
Database Performance
Database

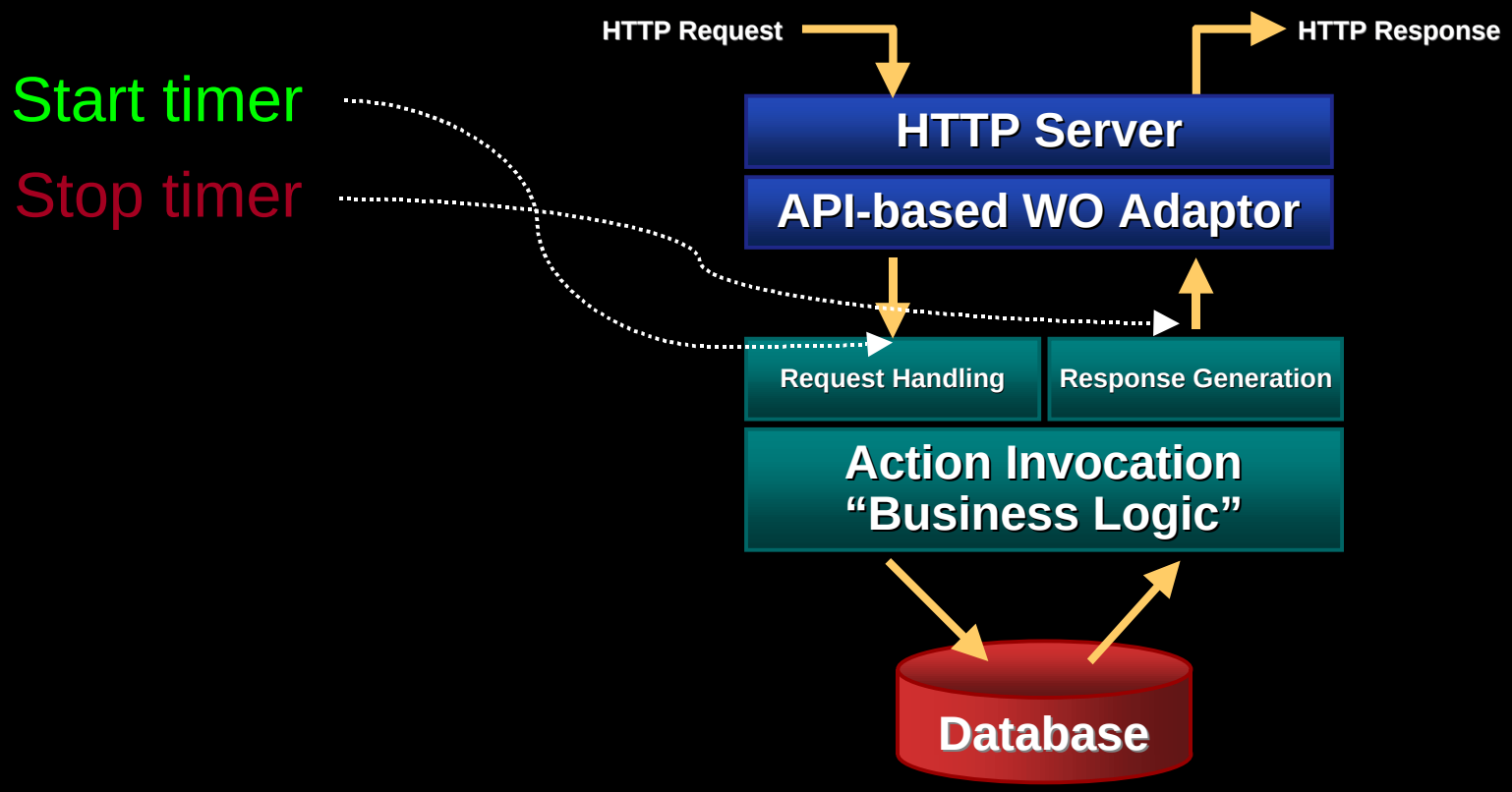

HTTP Request
HTTP Response
I/O, Network performance, latency
HTTP Server
API-based WO Adaptor
I/O, Network performance, latency
Request Handling
Response Generation
Action Invocation "Business Logic"
I/O, Network performance, latency
Database

WOStats

- Mechanism built in to WebObjects for reporting high-level statistics about an application
- Good starting point to get a general sense of whether your application is fast or slow

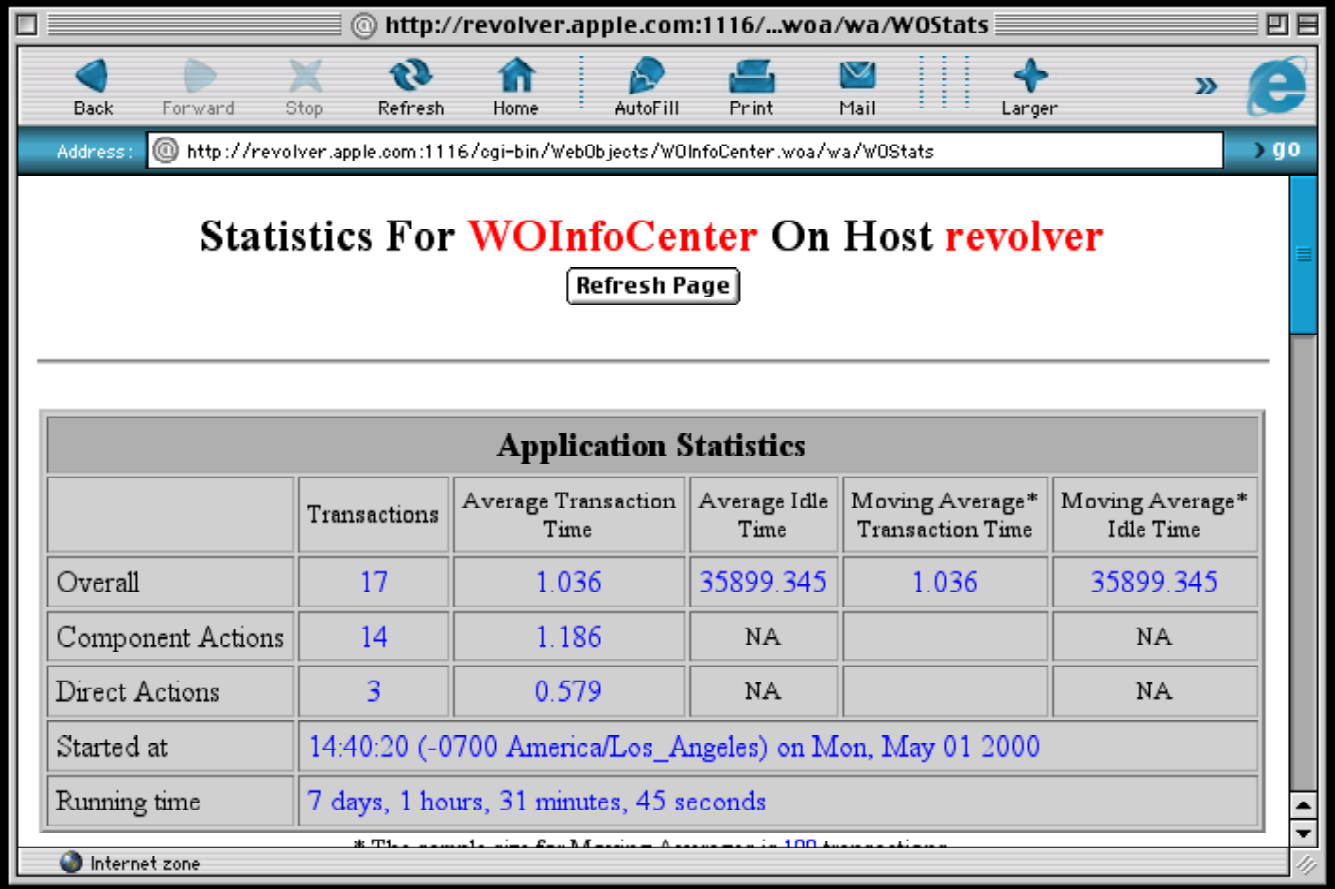
WOStats

- Measures transaction time at a high level

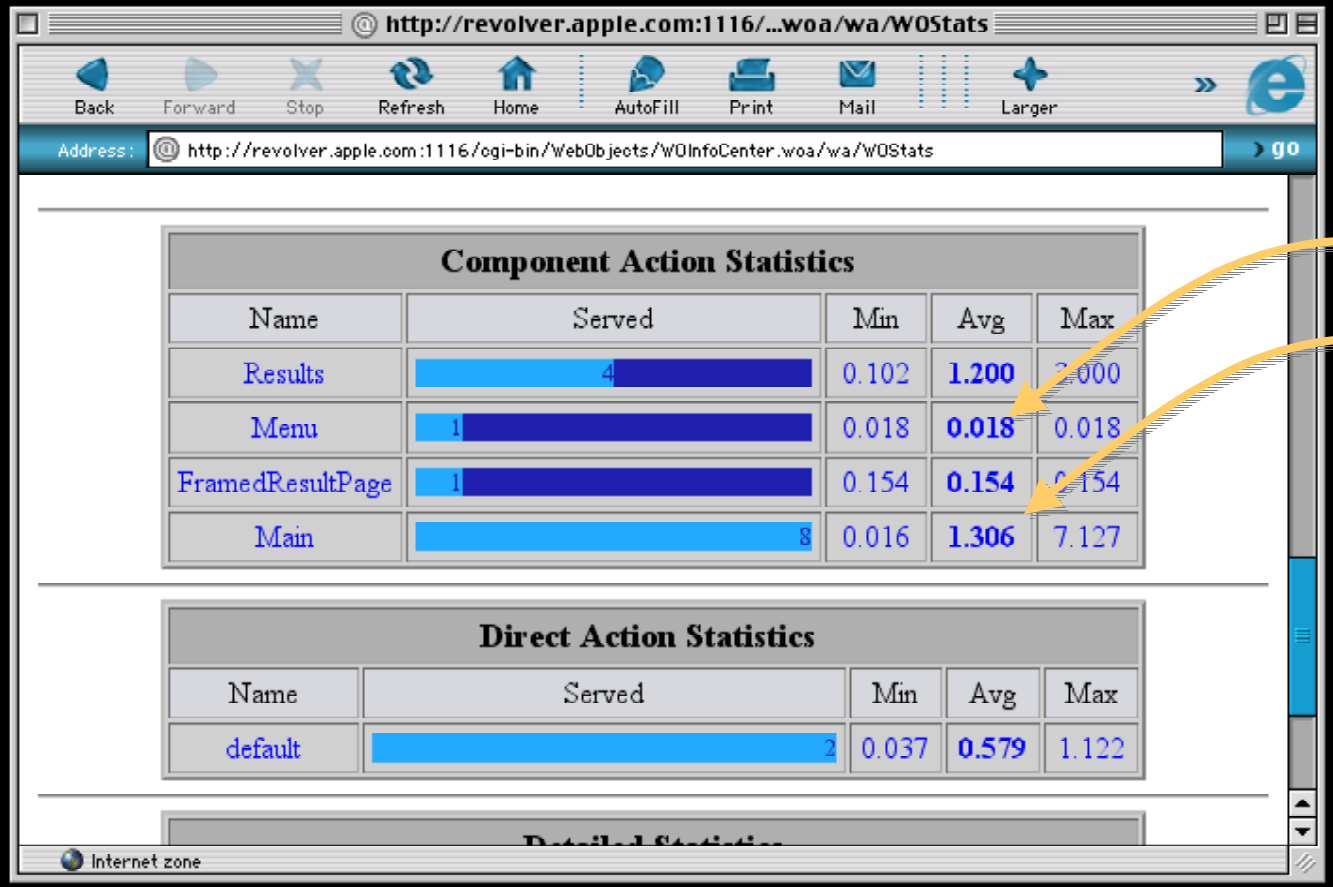


WOStats

http://localhost:1116/cgi-bin/WebObjects/WOInfoCenter.woa/wa/WOStats



WOStats



Fast!

Slow

Event Logging Introduction

- New in WebObjects 4.5
- More fine grained than WOStatistics
- Allows you to identify the bottlenecks of your app
- Built into any WO 4.5 App

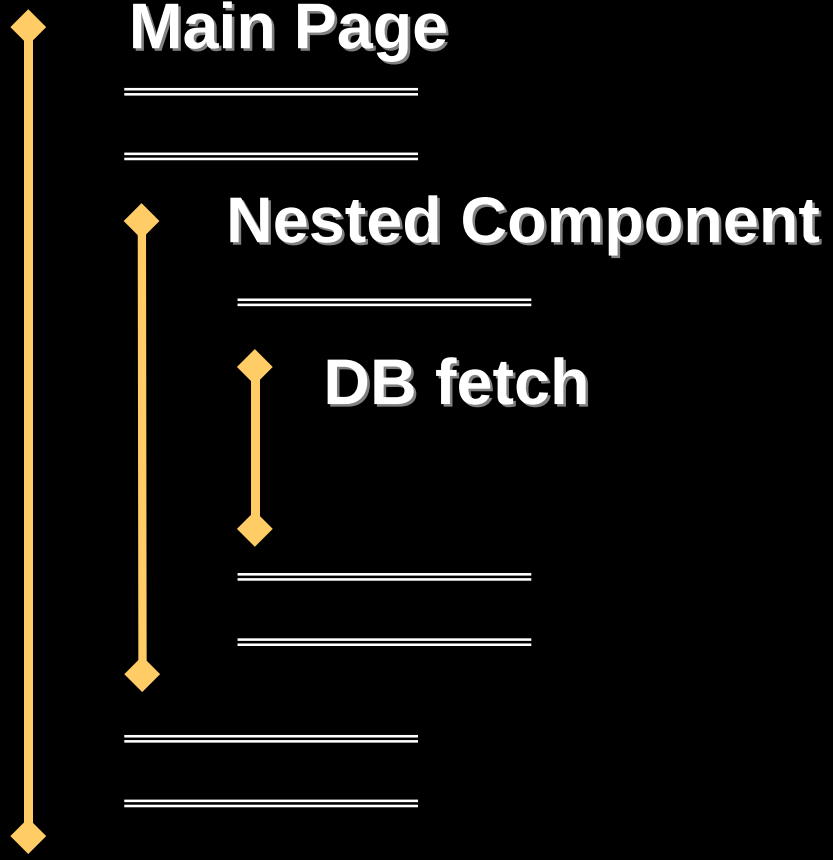
What You'll Learn

- What is an event
- Built-in events
- Enabling event logging
- Event log analysis
- Defining custom events for your own needs

Prerequisites

- WO 4.5 App on any platform
- Simple Web Browser
- No preparations necessary, just connect to your app when ready/when it looks slow

Example Call Stack



Efficient

- No special framework/library needs to be linked
- All event logging is programmed in-line
- All cost is at analysis time, not collection time
- Memory overflow protection

Performance Numbers

- 300MHz G3 can log over 300,000 events per second
- No disk I/O
- Pure C/public static Java methods for maximum performance

Efficient–Some More Numbers

- Custom event memory manager and garbage collector
- Worst case scenario:
  - You forget the app running with logging enabled…
    ~4MB memory wasted per thread (settable)
  - Memory will be recycled once this high water mark is reached
- User defaults for tunable parameters

Robustness

- Graceful handling of exceptions in your code (no memory or performance loss)
- Self-diagnosis: logging turns itself off automatically in a number of crisis situations (settable)

Completeness—Built-in Events

- Dozens of events for all common operations are built in, such as
  - EOAdaptor access
  - EOEditingContext
  - WOComponents and WOPages
  - WO Bindings

User Interface

- The UI is accessed through a web browser
- Two Direct Action web pages:
  - WOEventSetup
  - WOEventDisplay

Event Groups

- Built-in events collected into logical, atomic groups
- Example:
  - EODatabaseContext Event (group)
    - Objects with fetch specification (event)
    - Save changes (event)

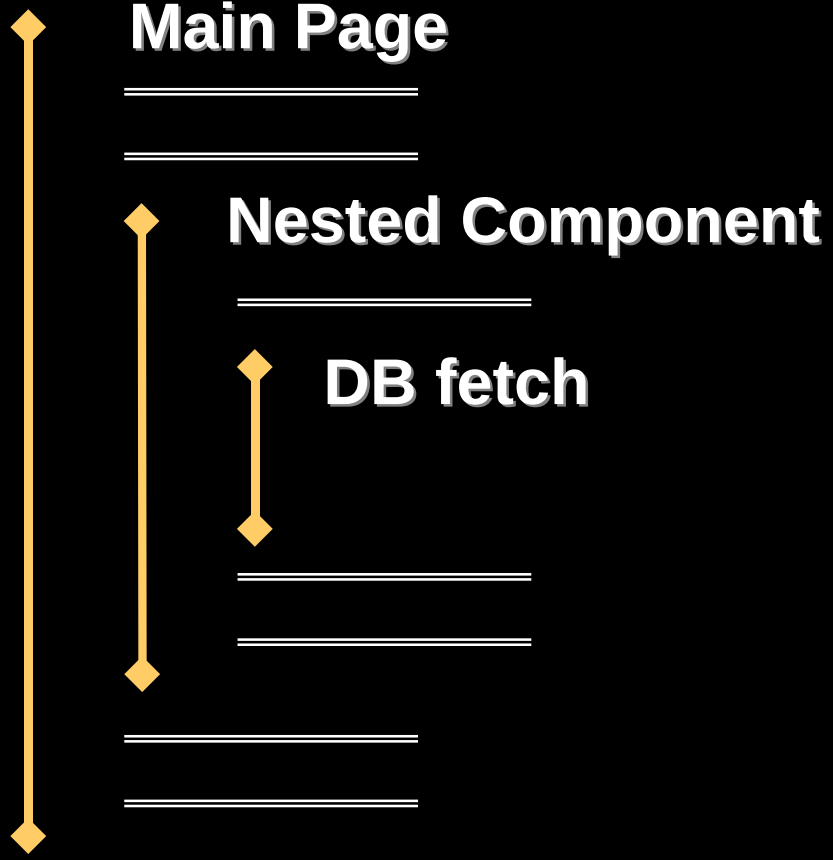
Event Setup Page

- WOEventSetup Direct Action
- Allows you to turn on/off event logging by Event Group
- All Event Groups applicable for your app are shown, and only those

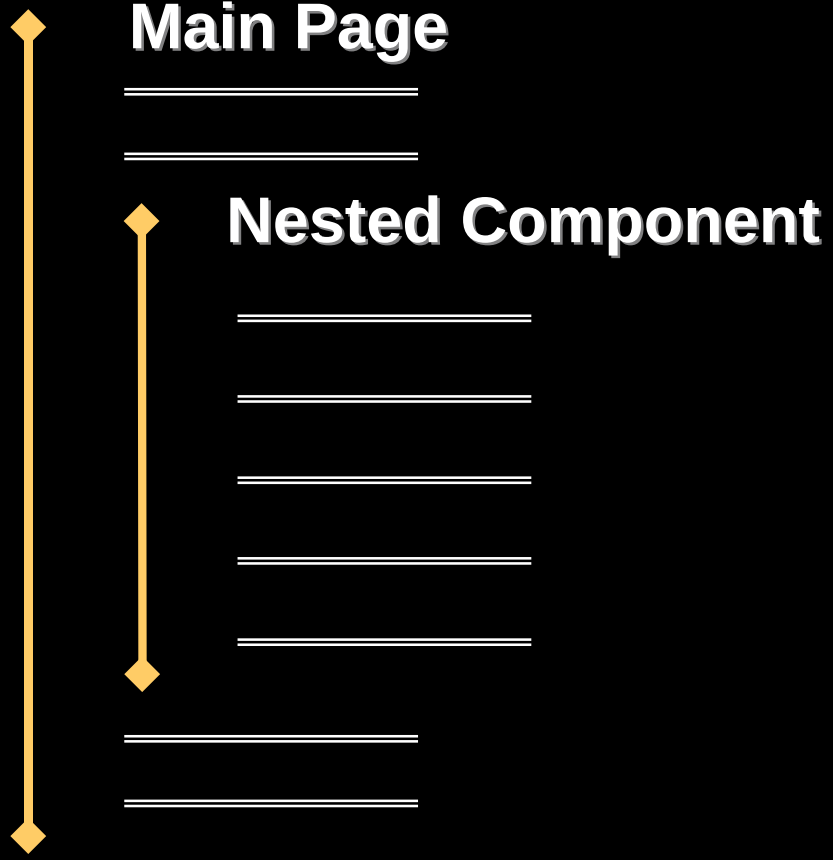
Event Display Page

- WOEventDisplay Direct Action
- Allows you to view the collected data in five different ways, depending on your needs

Example Call Stack—Case 1

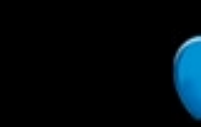


Example Call Stack—Case 2



Aggregated View of Both Cases

Main page	70 ms	2
Nested Component	60 ms	2
DB fetch	50 ms	1



View By Page, Component

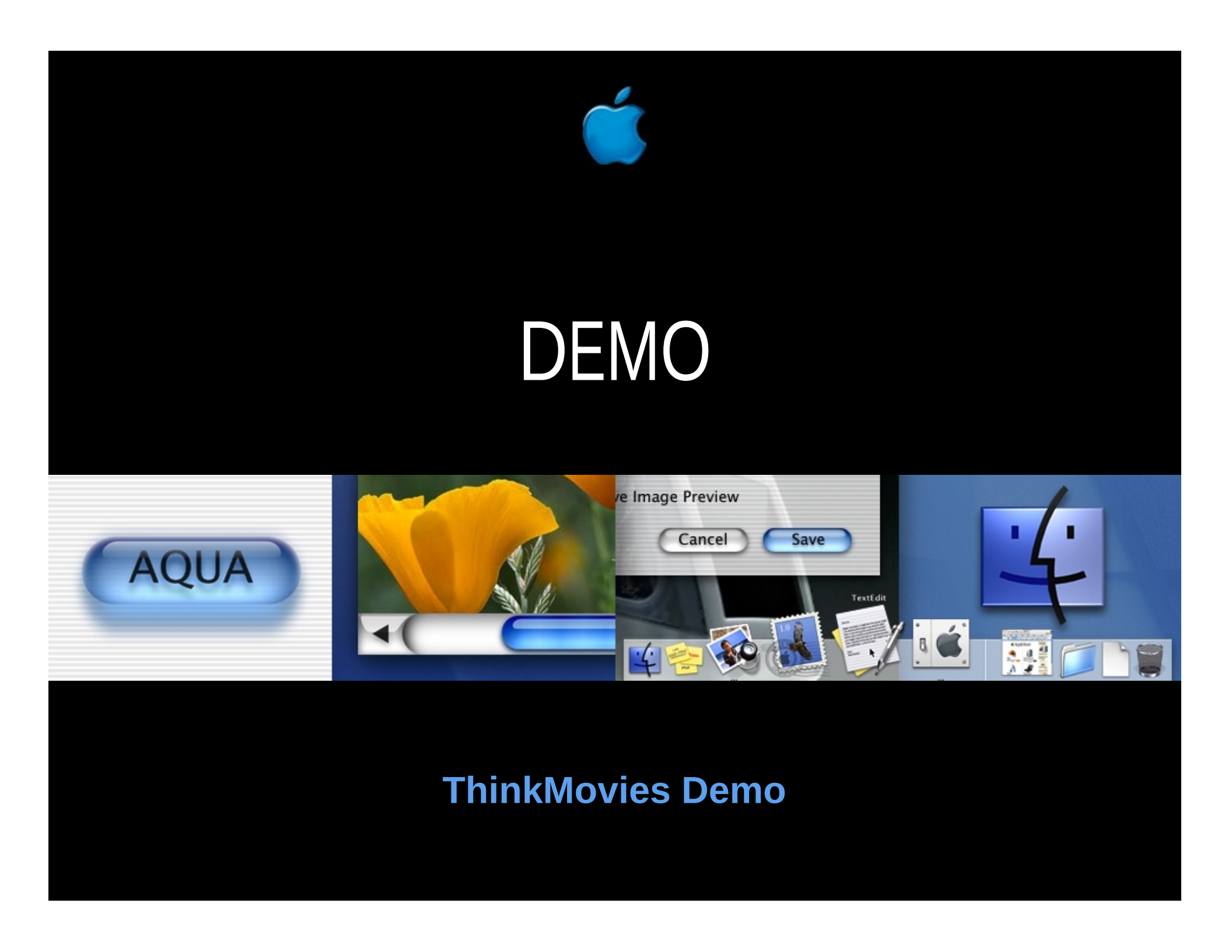
- Easiest, most common starting point
- Root level shows all pages touched during the run
- Child level shows individual components of that page
- Each sublevel shows sub-components, and so on

View By Page

- Root level shows all pages
- Useful if you want to flatten all components, without seeing a hierarchy (more info at one glance, but more confusing)
- You still can “drill down” into subevents

Unsorted View

- Shows events in an aggregated way, i.e., identical events are merged into one and counted
- Shows events nested like a call graph
- “Drill Down” to nested events by clicking on hyperlinks


DEMO
AQUA
Image Preview
Cancel
Save
TextEdit
ThinkMovies Demo

Customization

- Events are very general
- They are everywhere in the EO and WO frameworks where it makes sense
- You can customize them for your custom classes

Using WOEvent for Your Code

```
public void awake() {
    super.awake();
    WOEvent event = null;
    if (isEventLoggingEnabled()) {
        event = (WOEvent)
                EOEventCenter.newEventOfClass (
                WOEvent.class, comment);
        EOEventCenter.markStartOfEvent (event, “awake”);
    }
    // your profiled code here
    if (event != null)
        EOEventCenter.markEndOfEvent (event);
}
```

Alternative: Subclass

- All you need is a custom event class
- Example: MyComponentEvent

```
public class MyComponentEvent extends WOEvent {
    public MyComponentEvent () {
        super();
    }
}
```

Required Description File

- MyComponentEvent.description

```
{
    EOEventGroupName = “MyComponent Event”;
    doThis = “Operation #1”;
    doThat = “Another operation to be logged”;
}
```

- Key/Value name mapping
- Only group name mandatory
- Place in project Resources suitcase

Playback/Recording

- Automated load testing built into WebObjects
- Record HTTP request/response interactions with your WebObjects application
- Playback recorded session repeatedly to generate load

What It's Intended For…

- Good starting point for prototyping load
- "Get the feel" of your application's performance in a deployment scenario
- Free—comes with the product
- Powerful enough to prove stability of The Apple Store—a high-volume Internet site

Limitations

- Not a coverage tool
- Not for functional testing
- Not scriptable
- Only compares page lengths; does not do HTML matching
- Managing a high volume of virtual clients with Playback can be difficult

Recording

- To record, launch app with special argument:

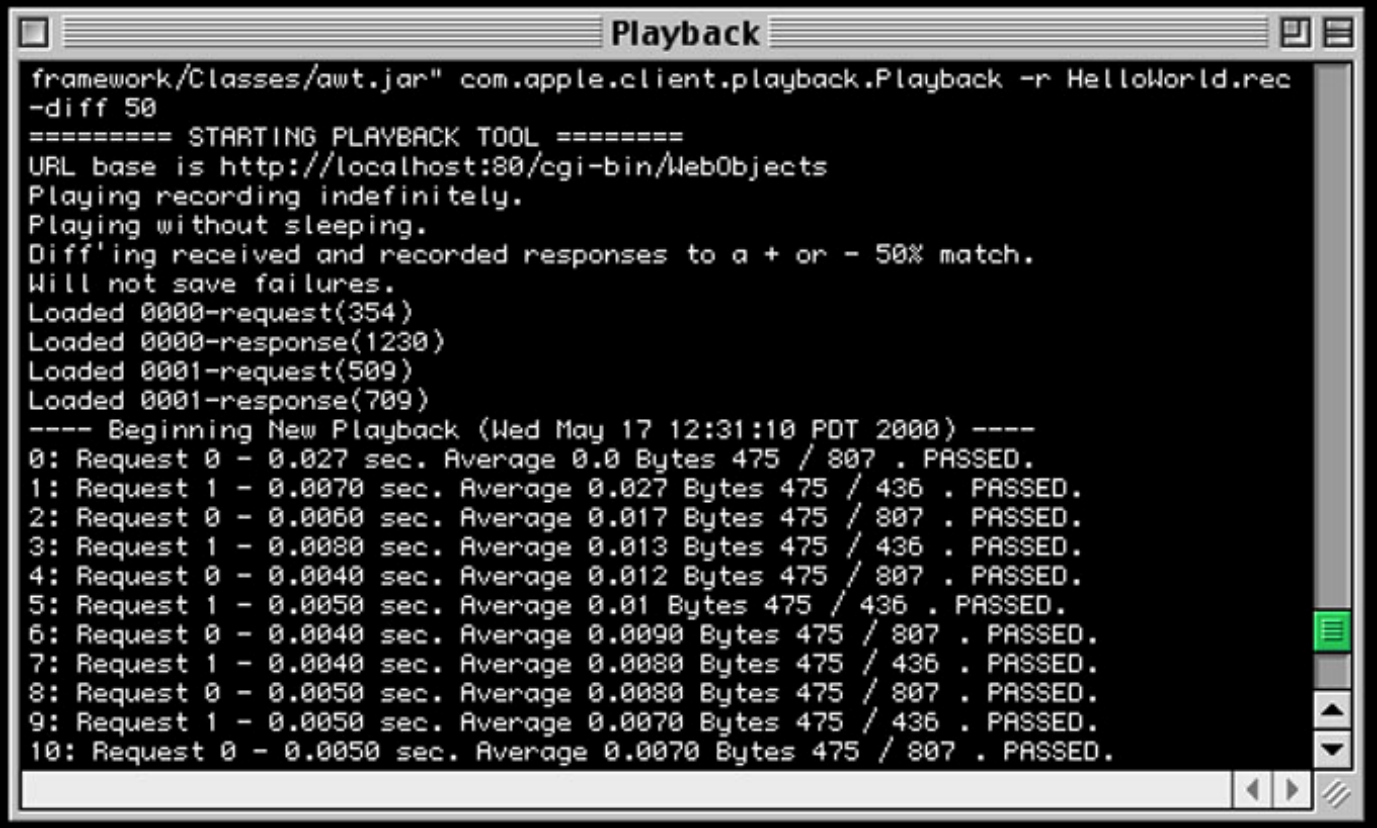
```
ThinkMovies  -WORecordingPath /tmp
```

- Use browser as client and click-through application's interface WebObjects will write requests and response to path specified above

Playback

- Playback the recorded interactions
- Two approaches:
  - Command-line interface
  - PlaybackManager

Command-Line Interface



Third-Party Tools

- Better for coverage
  - Scriptable for automating coverage
- Better for regression testing
  - More flexible response verification
- Automated site monitoring

Factors That Influence Performance

- Hardware
  - CPU
  - RAM
- Database
  - EOF
  - Schema and server
- Singlethreaded vs. Multithreaded
- Factors in the larger deployment scenario

For More Information

http://www.apple.com/webobjects

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

Try out your WebObjects 4.5 Evaluation CD!

WebObjects Community BOF
Wed., 6:30 p.m.–8:00 p.m.

Session 414

Q&A

Who to Contact

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

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

WWDC

Worldwide Developers Conference 2000

Think different.