#SHAREorg

Experiences with Mobile from Mainframe to Multiple Devices

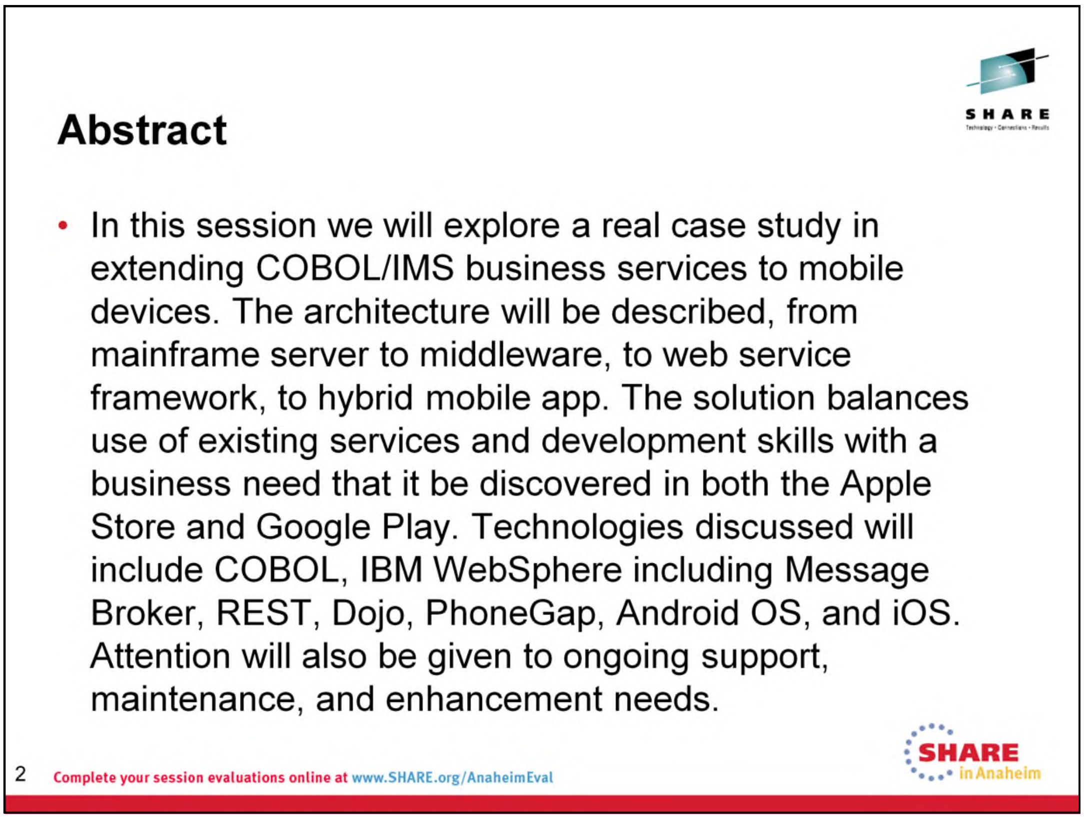
#SHARE2014MobExp

John Mallonee
Highmark

March 13, 2014
Session 14485

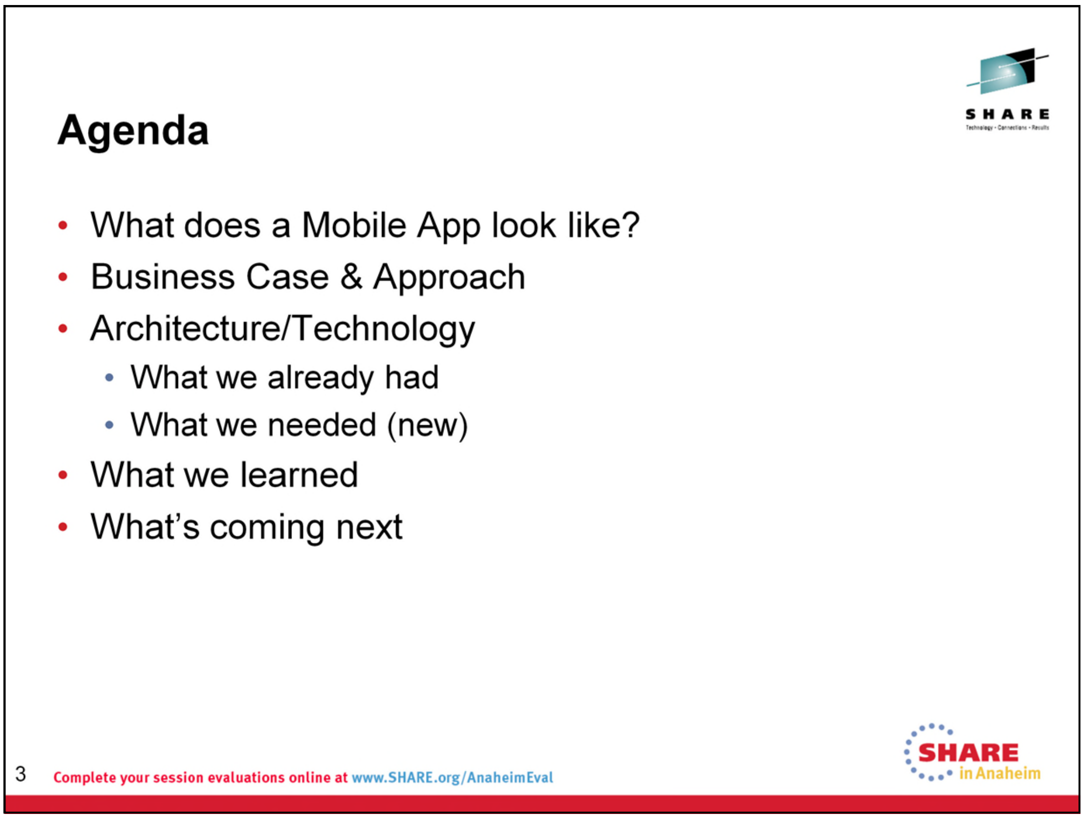
Test link: www.SHARE.org

SHARE in Anaheim



Abstract

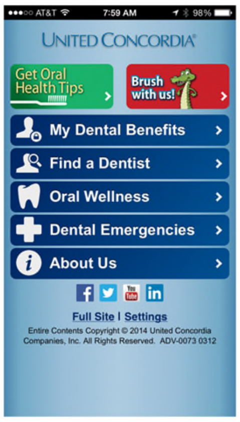
- In this session we will explore a real case study in extending COBOL/IMS business services to mobile devices. The architecture will be described, from mainframe server to middleware, to web service framework, to hybrid mobile app. The solution balances use of existing services and development skills with a business need that it be discovered in both the Apple Store and Google Play. Technologies discussed will include COBOL, IBM WebSphere including Message Broker, REST, Dojo, PhoneGap, Android OS, and iOS. Attention will also be given to ongoing support, maintenance, and enhancement needs.



Agenda

- What does a Mobile App look like?
- Business Case & Approach
- Architecture/Technology
  - What we already had
  - What we needed (new)
- What we learned
- What's coming next

What does a Mobile App look like?


●●●○○ AT&T 7:59 AM 98%

UNITED CONCORDIA®

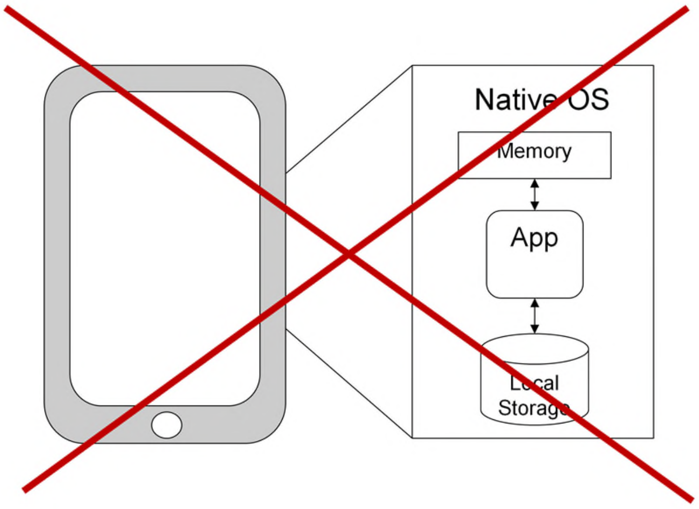
Get Oral Health Tips

Brush with us!

- My Dental Benefits
- Find a Dentist
- Oral Wellness
- Dental Emergencies
- About Us

Full Site I Settings

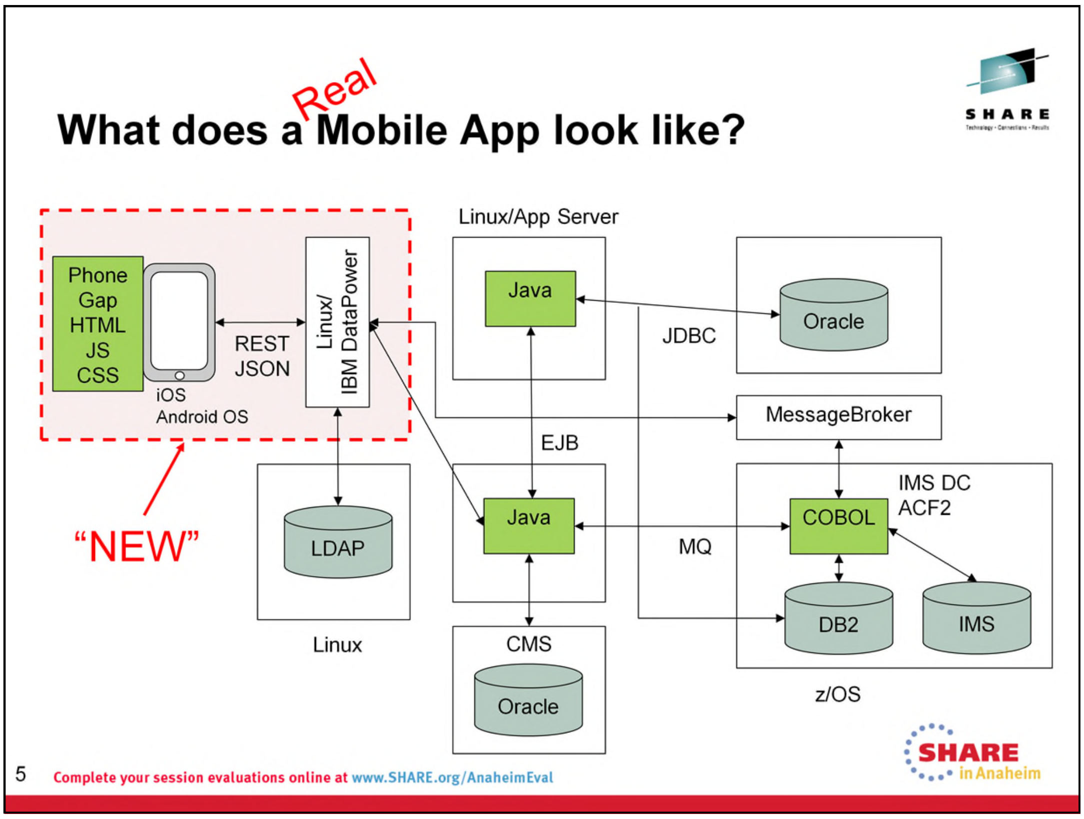
Entire Contents Copyright © 2014 United Concordia Companies, Inc. All Rights Reserved. ADV-0073 0312



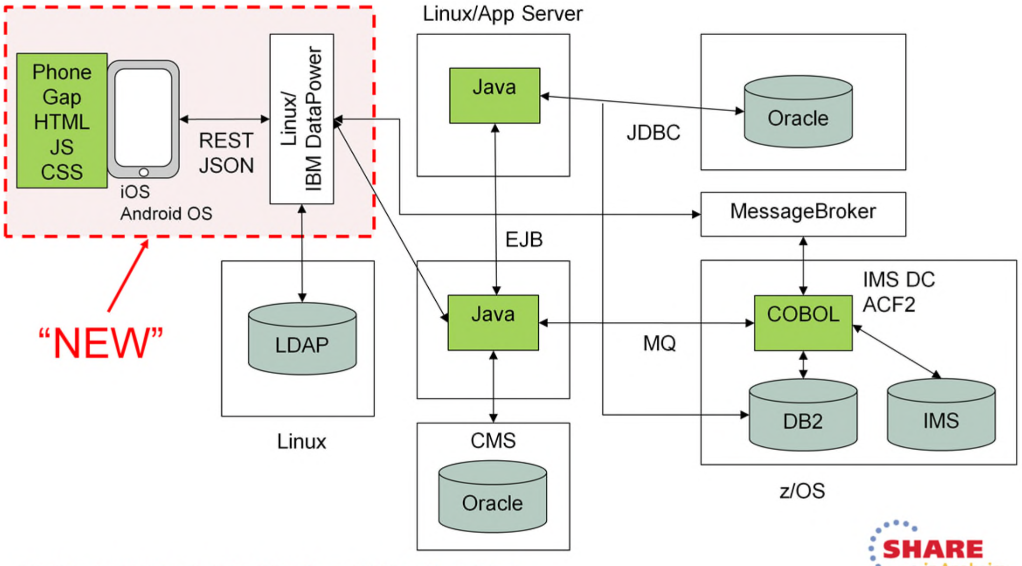
4 Complete your session evaluations online at www.SHARE.org/AnaheimEval

The point here is that for real business applications, there is a connected back-end for services. The "mobile" part of the app is just a presentation layer that is unique for the mobile environment.

Luckily, our enterprise had most of the back-end (services, middleware, business logic) already.



What does a Real Mobile App look like?



5 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

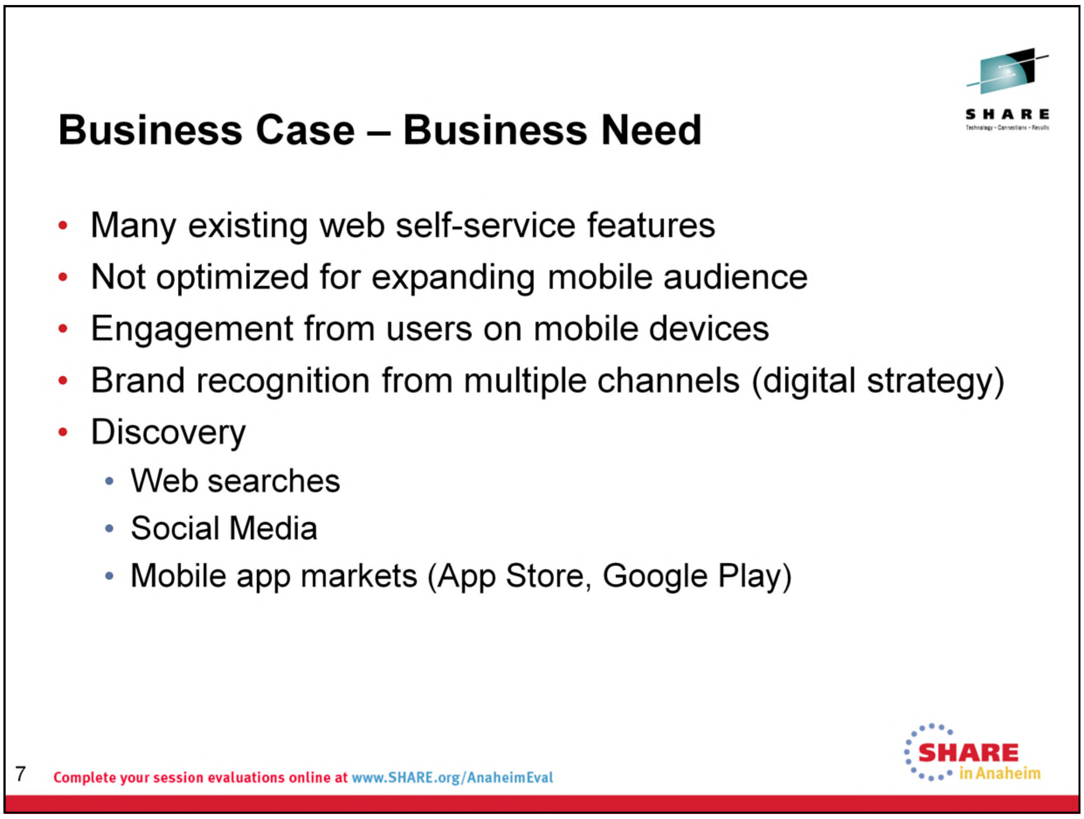
This is a high level depiction of how a mobile client can interact with an enterprise back-end environment.

The back-end already existed for us as a result of our existing web applications and was a combination of mainframe and open systems infrastructure.

Business Case

- Business Need
- Research & Planning
- Collaboration
- Technical Implications & Outcome

SHARE in Anaheim

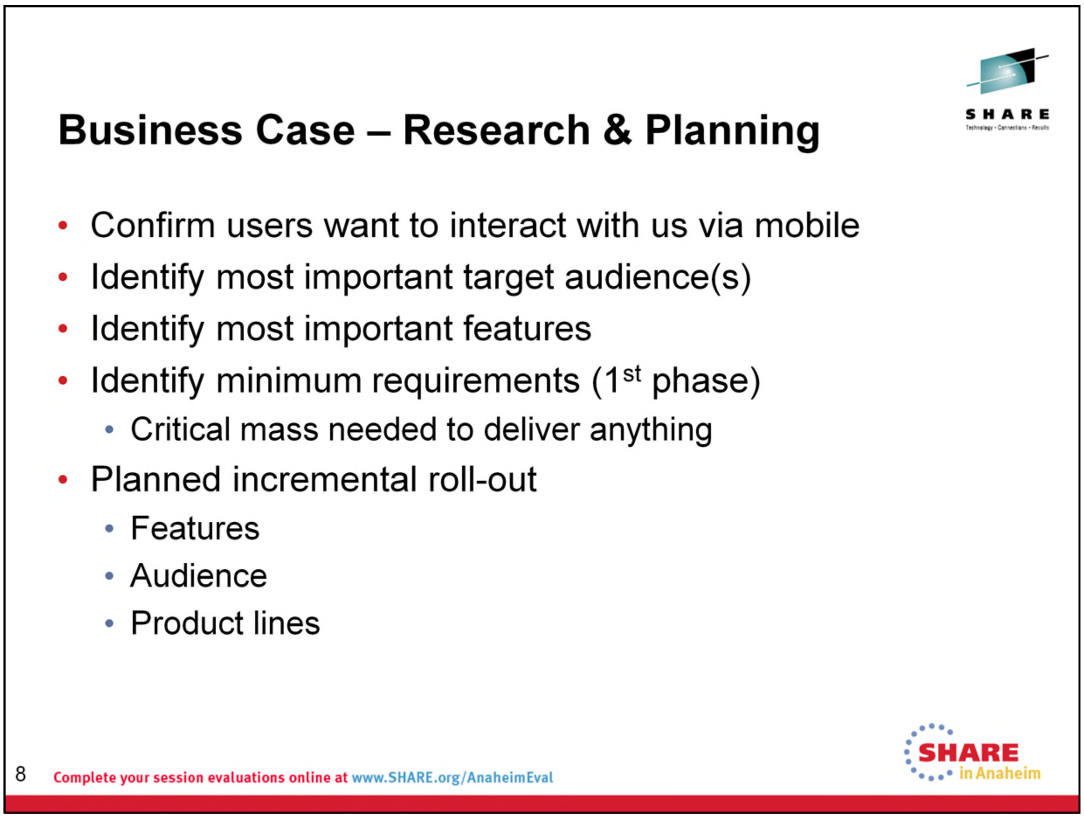


Business Case – Business Need

- Many existing web self-service features
- Not optimized for expanding mobile audience
- Engagement from users on mobile devices
- Brand recognition from multiple channels (digital strategy)
- Discovery
  - Web searches
  - Social Media
  - Mobile app markets (App Store, Google Play)

7 Complete your session evaluations online at www.SHARE.org/AnaheimEval

The primary business need was to reach people who are more and more engaging service providers from a mobile channel. Ultimately, this led to the discussion of how to provide mobile application support for existing self-service features.



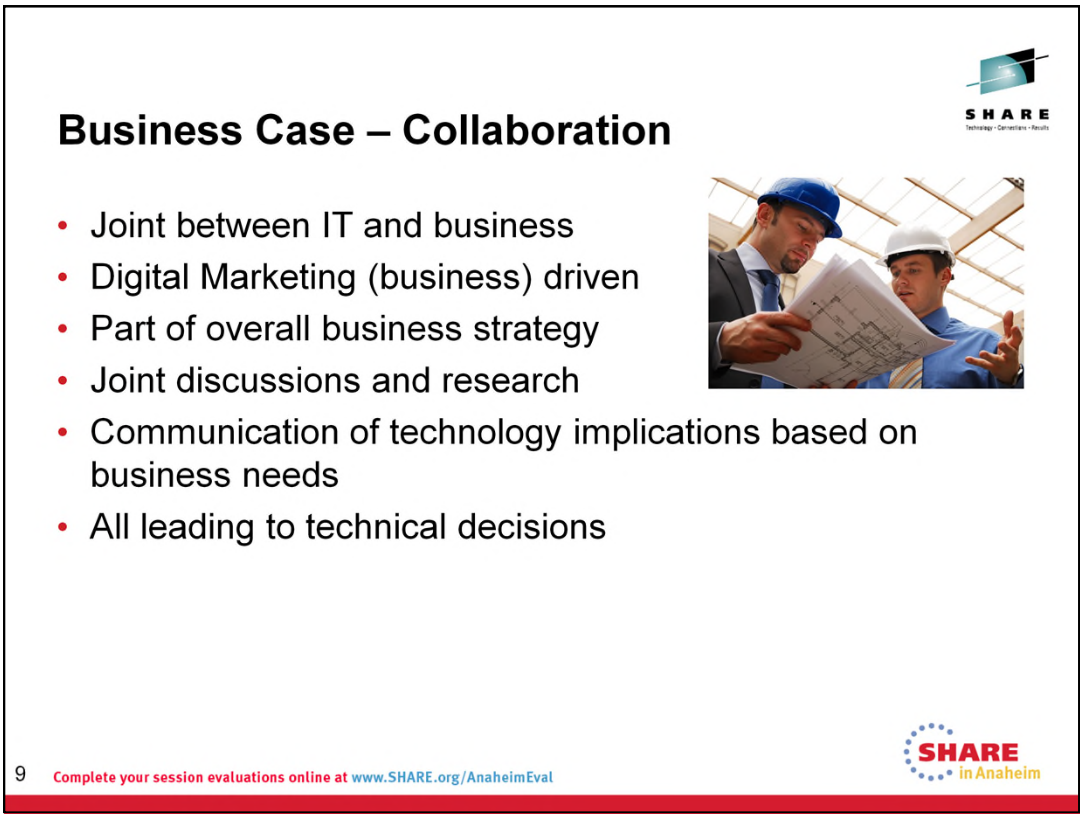
Business Case – Research & Planning

- Confirm users want to interact with us via mobile
- Identify most important target audience(s)
- Identify most important features
- Identify minimum requirements (1st phase)
  - Critical mass needed to deliver anything
- Planned incremental roll-out
  - Features
  - Audience
  - Product lines

SHARE in Anaheim

The research phase was critical to determine the direction of the app including whether it was really needed and if so what the features should be.

Ultimately, we determined that there was great interest in having features exposed in a mobile channel and which features were the minimum requirements. This helped us to shape a plan to get some features our first with an opportunity to phase additional features in over time (subsequent releases).



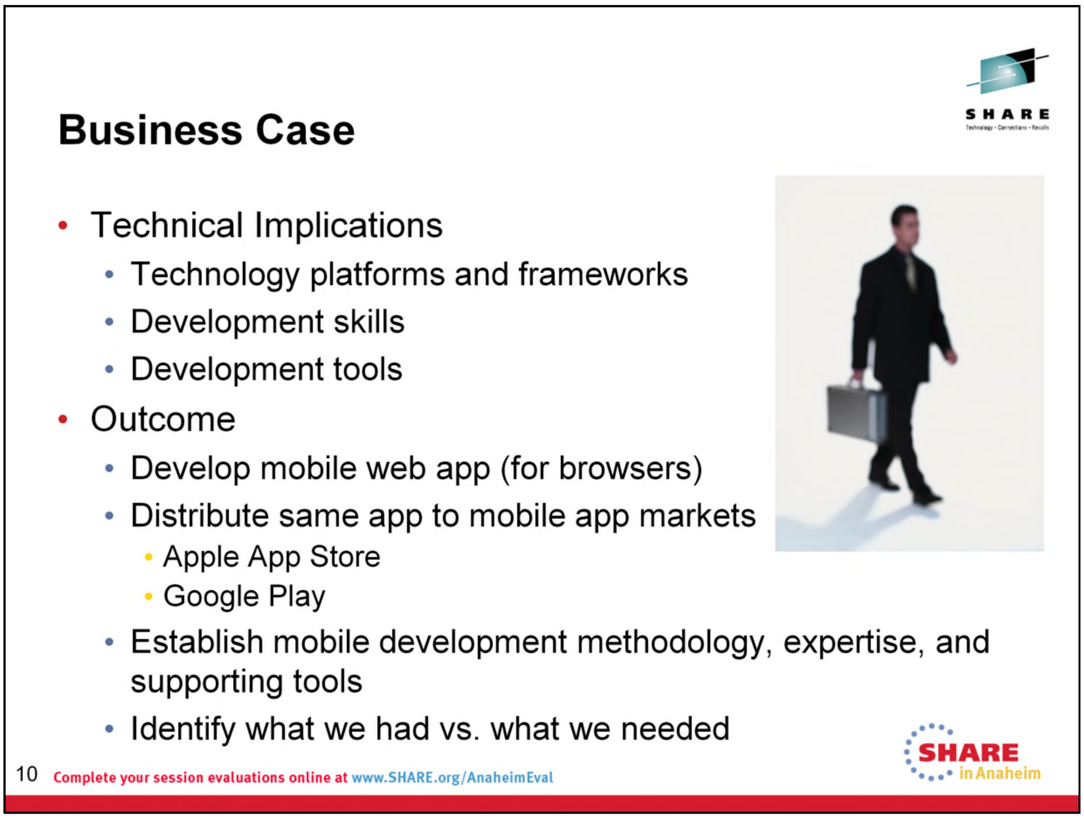
Business Case – Collaboration

- Joint between IT and business
- Digital Marketing (business) driven
- Part of overall business strategy
- Joint discussions and research
- Communication of technology implications based on business needs
- All leading to technical decisions

9 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

The discussion between IT and the business was very productive and ultimately shaped the technical decisions to be made. Fortunately, we were able to clearly articulate the implications of business choices which allowed the business owners to make more informed business decisions.



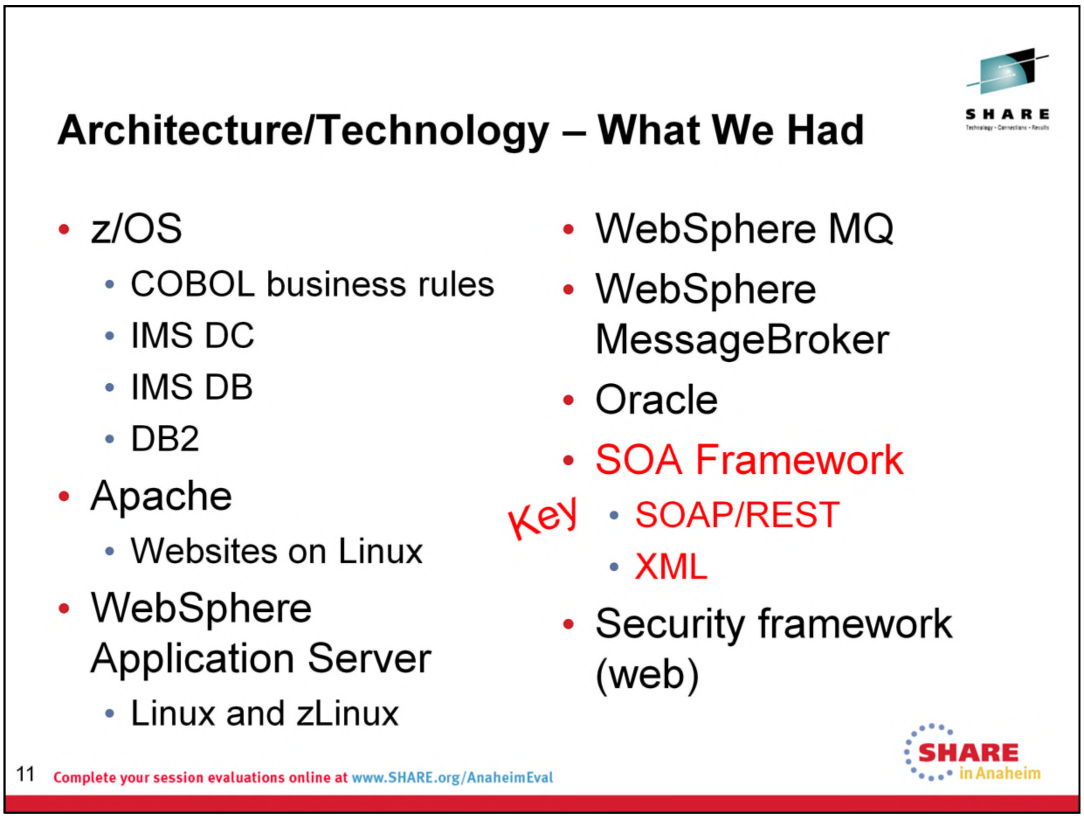
Business Case

- Technical Implications
  - Technology platforms and frameworks
  - Development skills
  - Development tools
- Outcome
  - Develop mobile web app (for browsers)
  - Distribute same app to mobile app markets
    - Apple App Store
    - Google Play
  - Establish mobile development methodology, expertise, and supporting tools
  - Identify what we had vs. what we needed

10 Complete your session evaluations online at www.SHARE.org/AnaheimEval

After providing implications around things such as cost, scope of effort, and device use, we were able to make technical choices around the needed technology stack, skills, and development approach.

This was new ground for us, but we started off by identifying what we could leverage from what we had and what we needed to add.



Architecture/Technology – What We Had

- z/OS
  - COBOL business rules
  - IMS DC
  - IMS DB
  - DB2
- Apache
  - Websites on Linux
- WebSphere Application Server
  - Linux and zLinux
- WebSphere MQ
- WebSphere MessageBroker
- Oracle
- SOA Framework
  - SOAP/REST
  - XML
- Security framework (web)

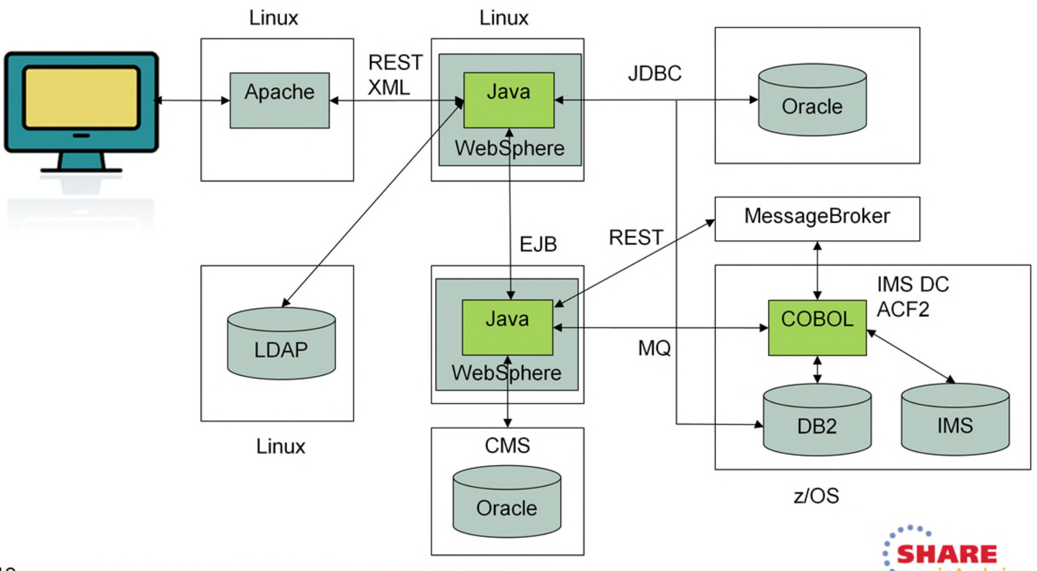
Key

11 Complete your session evaluations online at www.SHARE.org/AnaheimEval

We were already a large enterprise shop with support for mainframe and open systems applications. Much of what we already had in place to support web application development could be leveraged for mobile development.

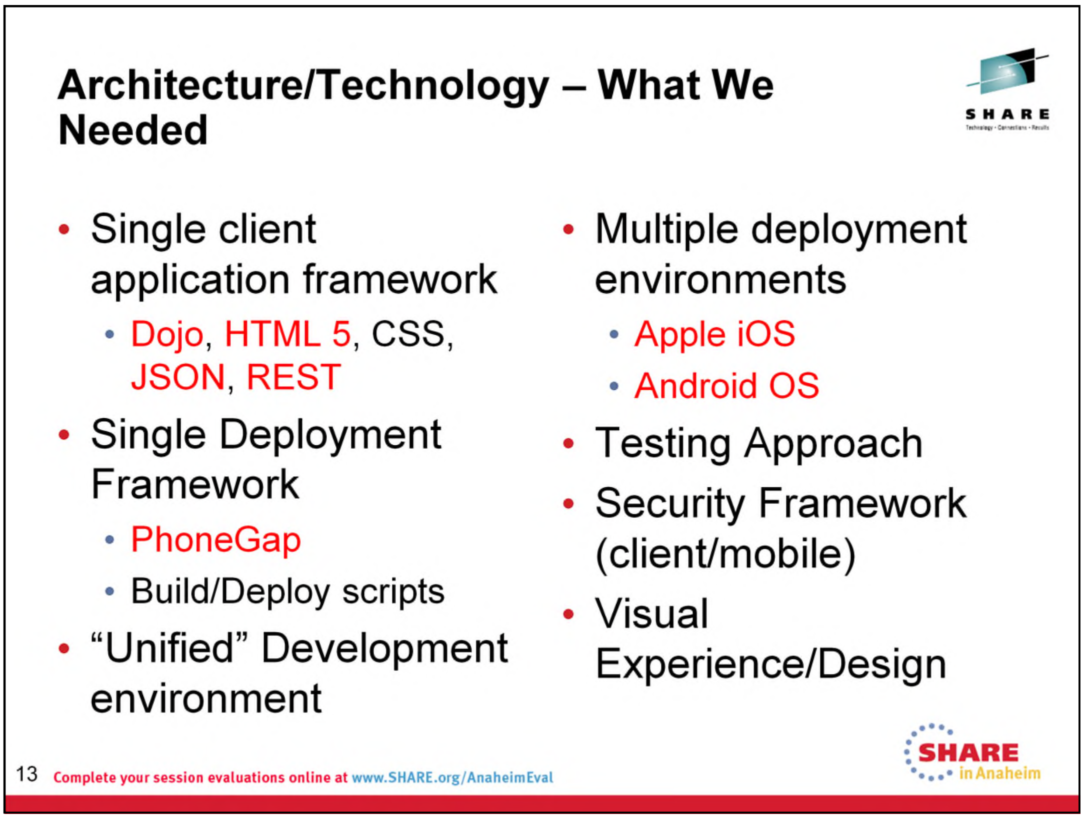
Having a SOA framework proved to be key to the transition from web development, although with some slight enhancements.

Architecture/Technology – What We Had



Complete your session evaluations online at www.SHARE.org/AnaheimEval

Visual representation of our existing infrastructure/architecture.



Architecture/Technology – What We Needed

- Single client application framework
  - Dojo, HTML 5, CSS, JSON, REST
- Single Deployment Framework
  - PhoneGap
  - Build/Deploy scripts
- "Unified" Development environment
- Multiple deployment environments
  - Apple iOS
  - Android OS
- Testing Approach
- Security Framework (client/mobile)
- Visual Experience/Design

13 Complete your session evaluations online at www.SHARE.org/AnaheimEval

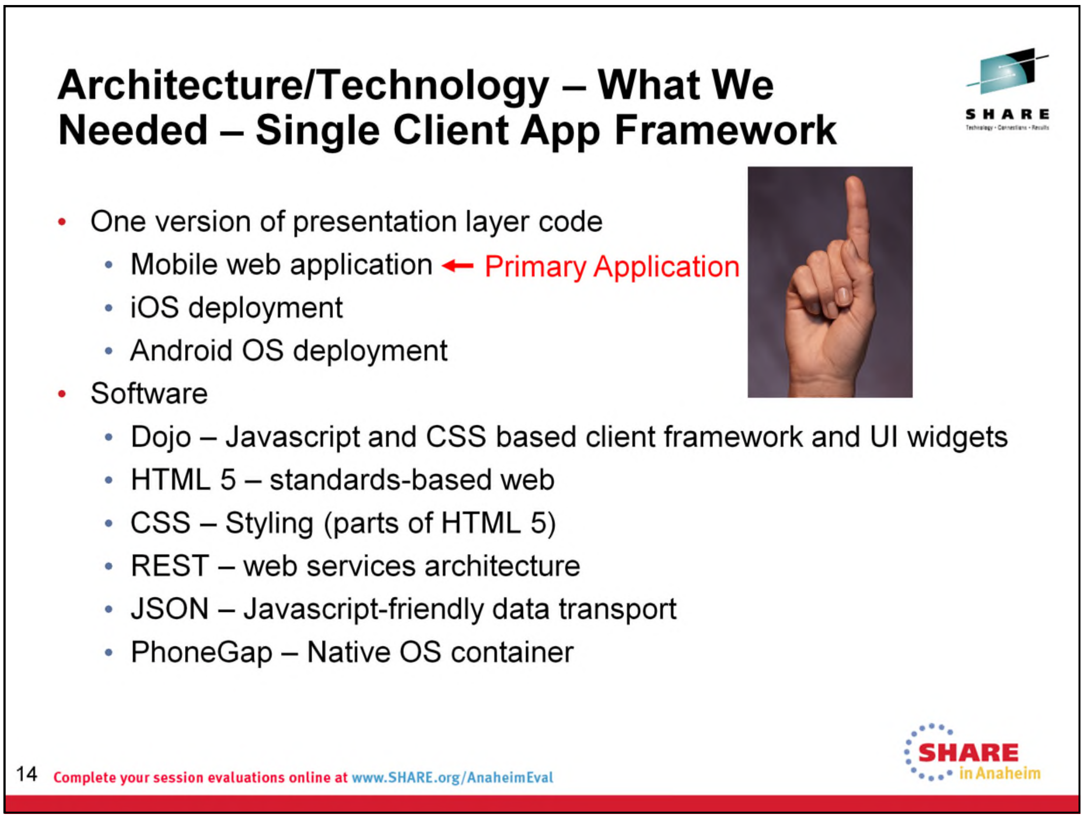
SHARE in Anaheim

While the items that we needed were incremental to what we already had, there were still a lot of details.

Items highlighted as RED are just to point out some high level concepts development technologies that were new.

Thinking from the perspective of a client application was a key departure from our current web applications. The development of the client application reused some existing skills that we had including HTML, CSS, and Javascript, but to a much larger degree than we had used in the past. The Dojo framework was new and as such had its own complexities and nuances.

It was also important to think in the mobile operating system mindset for both iOS and Android. Concepts between the two are similar, but there are real differences between the two platforms. PhoneGap helped to insulate differences from a coding perspective, but the deployment and testing were distinctly different.



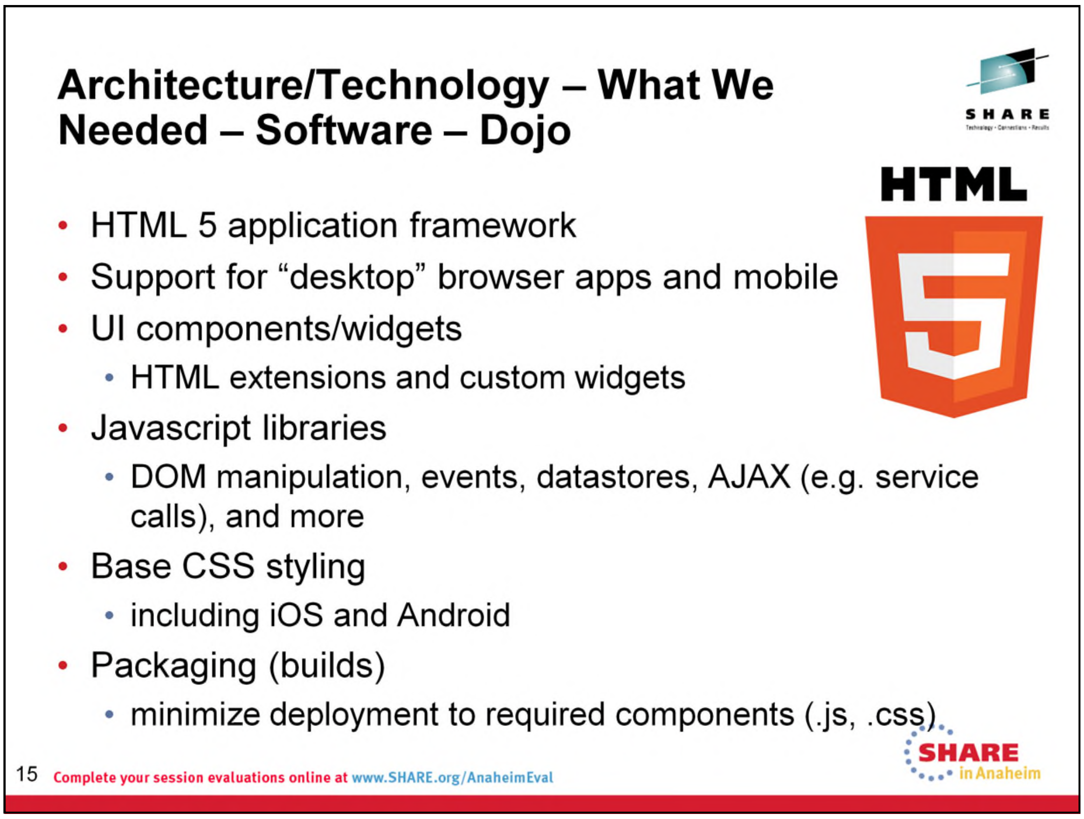
Architecture/Technology – What We Needed – Single Client App Framework

- One version of presentation layer code
  - Mobile web application ← Primary Application
  - iOS deployment
  - Android OS deployment
- Software
  - Dojo – Javascript and CSS based client framework and UI widgets
  - HTML 5 – standards-based web
  - CSS – Styling (parts of HTML 5)
  - REST – web services architecture
  - JSON – Javascript-friendly data transport
  - PhoneGap – Native OS container

14 Complete your session evaluations online at www.SHARE.org/AnaheimEval

Ultimately, we were able to code one application code base that was usable for 3 platforms:

- Mobile web – this was HTML 5, CSS, Javascript, and Dojo and was capable of running in a web browser (though not in older versions of Internet Explorer)
- iOS and Android versions embedded the mobile web app in PhoneGap with unique configuration for each mobile OS



Architecture/Technology – What We Needed – Software – Dojo

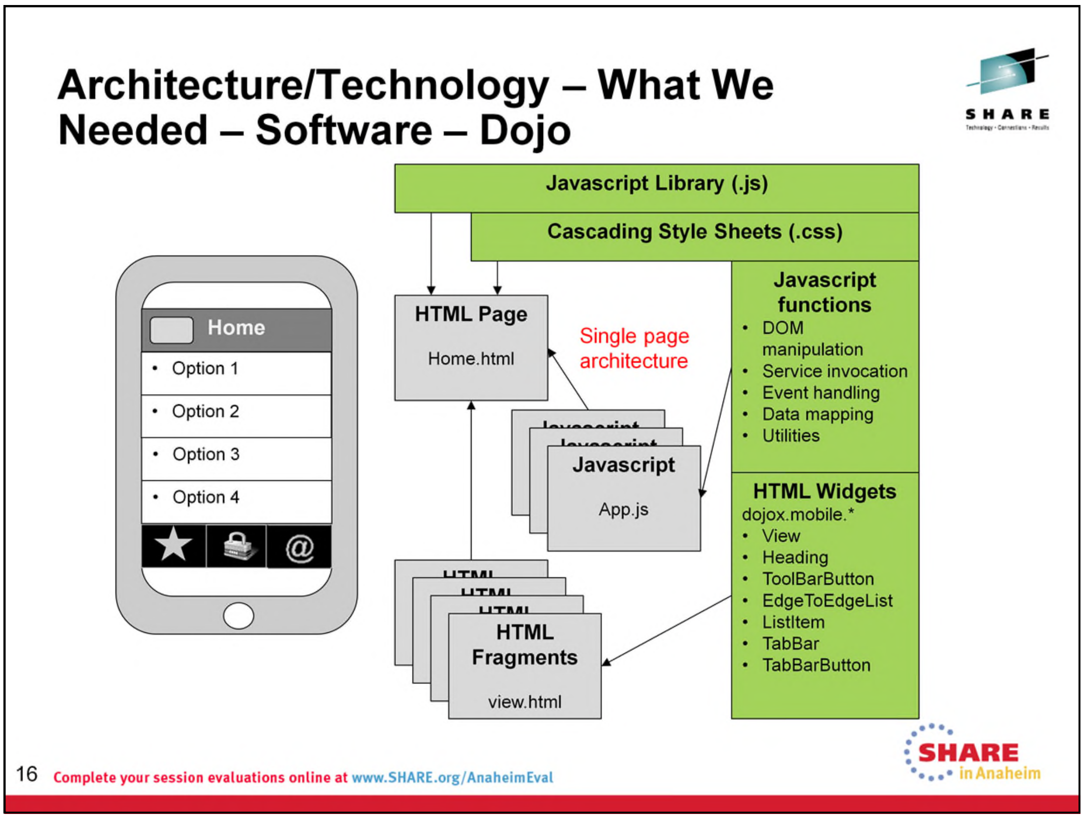
- HTML 5 application framework
- Support for "desktop" browser apps and mobile
- UI components/widgets
  - HTML extensions and custom widgets
- Javascript libraries
  - DOM manipulation, events, datastores, AJAX (e.g. service calls), and more
- Base CSS styling
  - including iOS and Android
- Packaging (builds)
  - minimize deployment to required components (.js, .css)

15 Complete your session evaluations online at www.SHARE.org/AnaheimEval

Dojo

- Just one of many HTML 5 frameworks
- Javascript/CSS-based

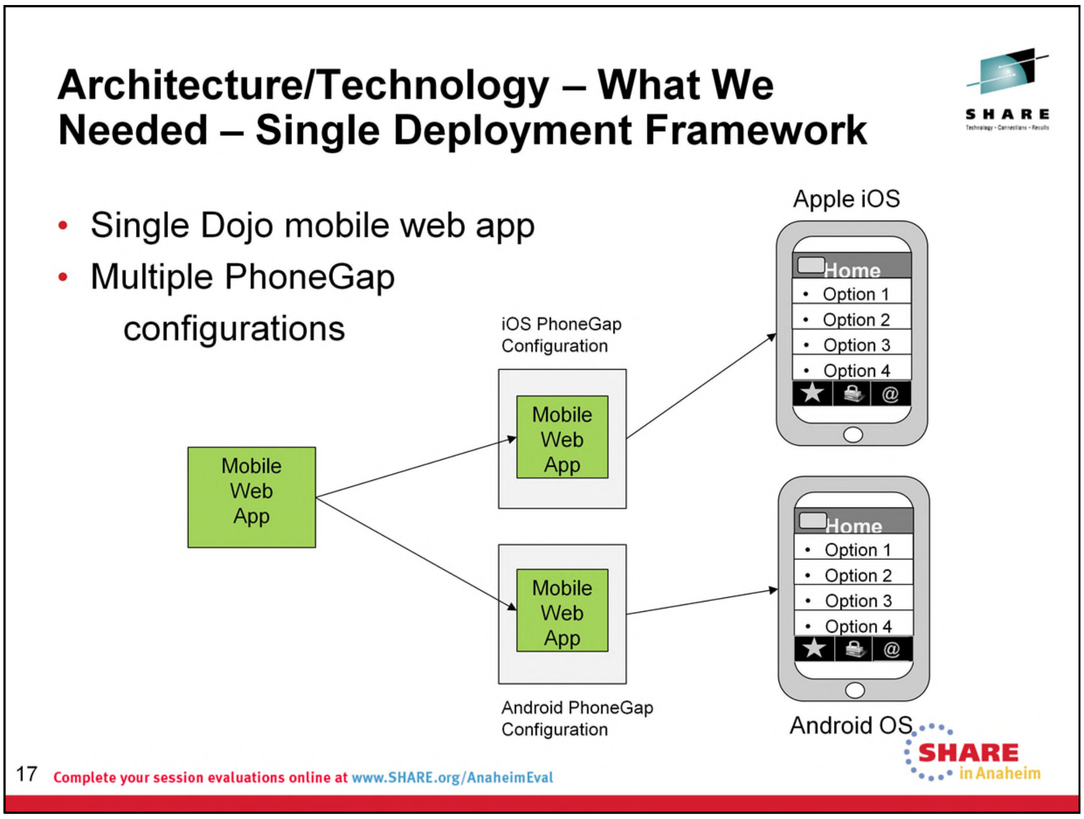
Like any framework, Dojo provides a lot of functionality "out of the box". It was challenging at times to get the right level of documentation even though there is a large site dedicated to the product (dojotoolkit.org). Additionally, the product was still evolving, so understanding the capabilities of a given version was important.



Architecture/Technology – What We Needed – Software – Dojo

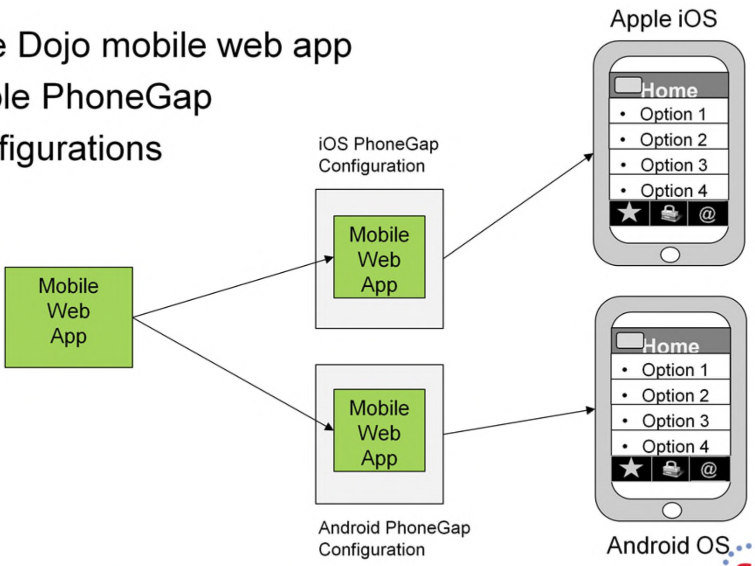
High level depiction of the primary components of Dojo (visual components, Javascript libraries, stylesheets).

An important item to note is that mobile apps are typically built using a single page architecture. This means that there is a single HTML page with additional page fragments (views) that are loaded dynamically based on user interaction. This is not required, but is a standard pattern of use.



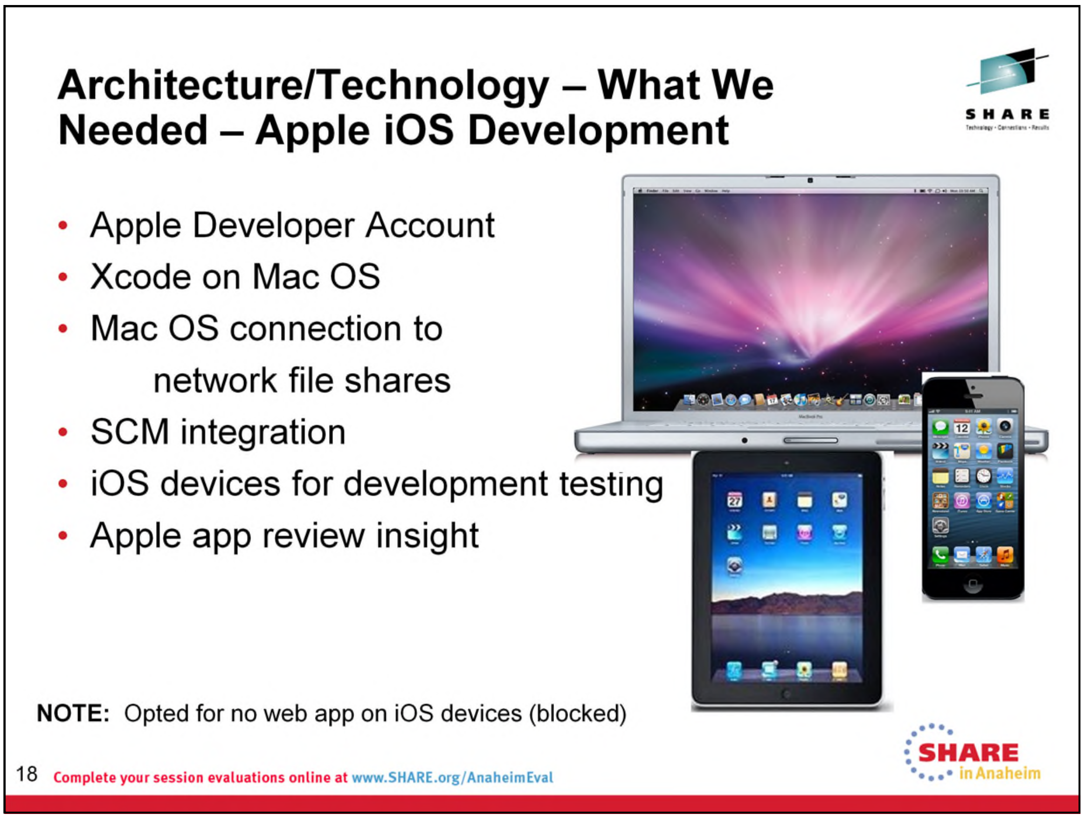
Architecture/Technology – What We Needed – Single Deployment Framework

- Single Dojo mobile web app
- Multiple PhoneGap configurations



A single mobile web app is created and then all of the web files (.html, .css, .js, images) are embedded in native OS projects:

- iOS – Xcode PhoneGap project (see subsequent pages)
- Android – Java PhoneGap project (see subsequent pages)



Architecture/Technology – What We Needed – Apple iOS Development

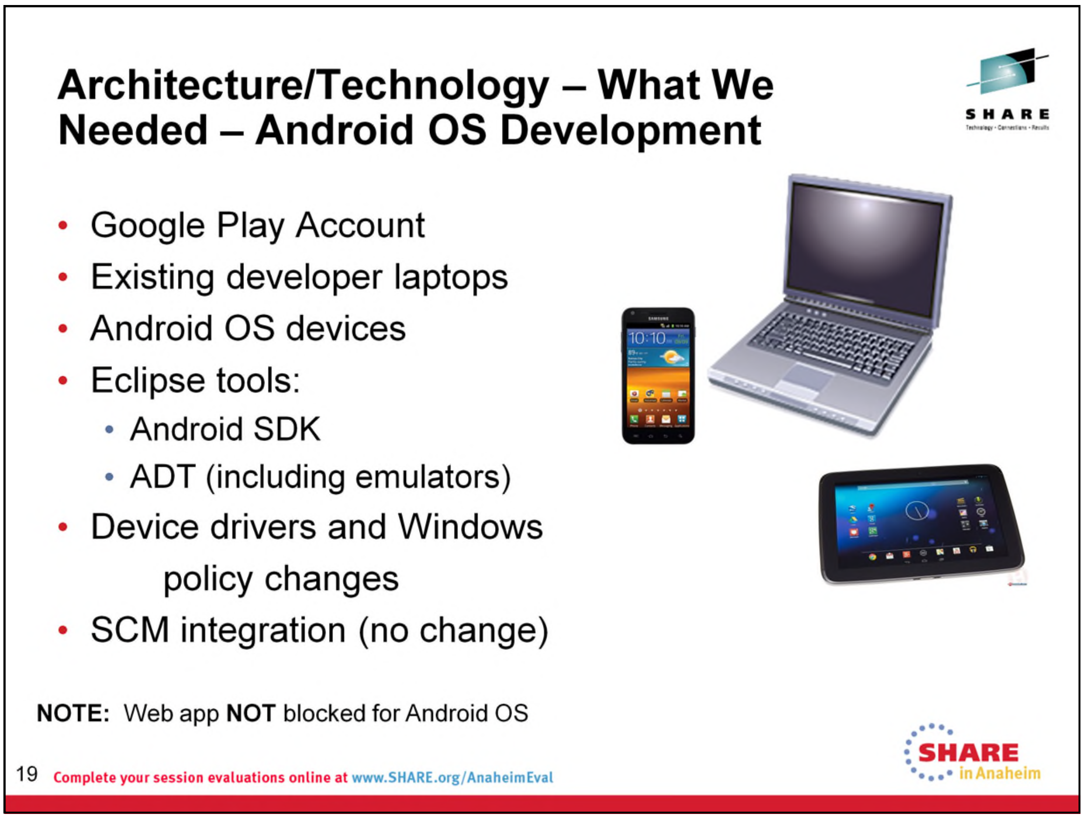
- Apple Developer Account
- Xcode on Mac OS
- Mac OS connection to network file shares
- SCM integration
- iOS devices for development testing
- Apple app review insight

NOTE: Opted for no web app on iOS devices (blocked)

18 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- A developer account is needed with Apple and needs to be owned by someone. This requires purchasing and acceptance of terms which can involve procurement and legal collaboration.
- Mobile devices not easy to manage – bill payment and reimbursement, data plan, ownership and governance of devices, physical security
- Developers should become familiar with Xcode and the configuration settings for PhoneGap
- We had to develop procedures related to connectivity and software versioning since Macs are used very sparingly in our enterprise
- Reviewing and understanding Apple’s extensive review and acceptance policies and distribution methodology was key



Architecture/Technology – What We Needed – Android OS Development

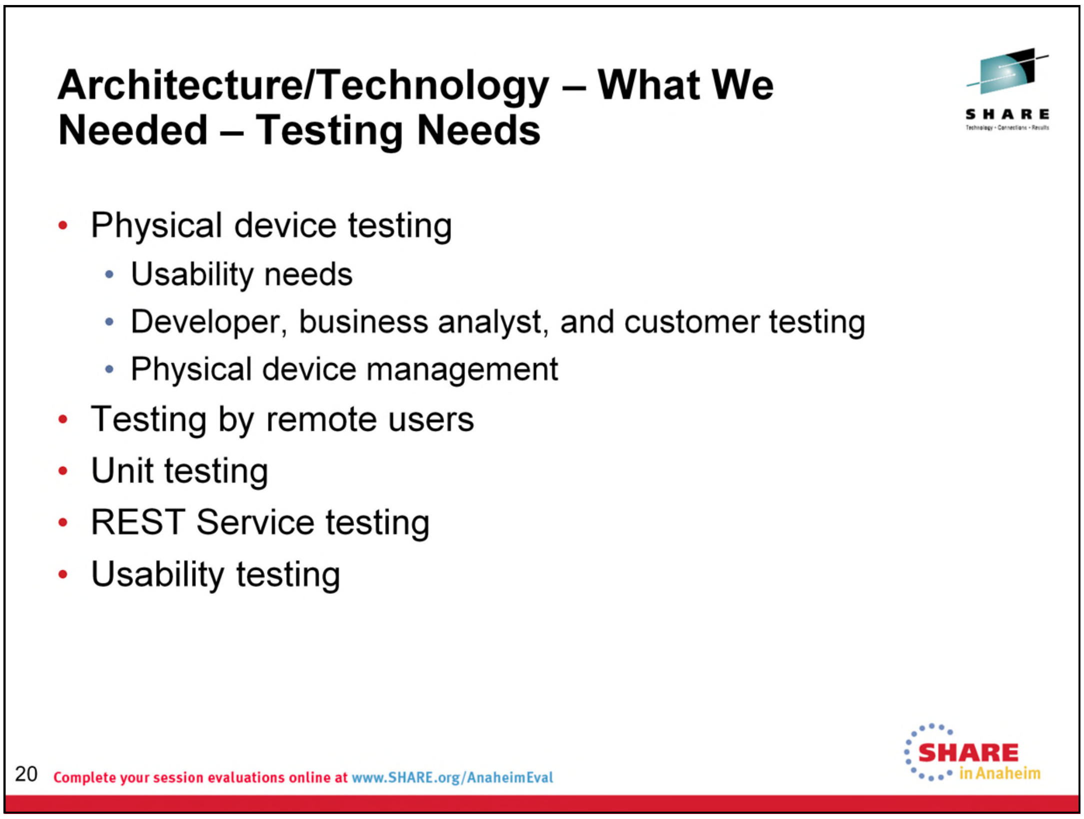
- Google Play Account
- Existing developer laptops
- Android OS devices
- Eclipse tools:
  - Android SDK
  - ADT (including emulators)
- Device drivers and Windows policy changes
- SCM integration (no change)

NOTE: Web app **NOT** blocked for Android OS

19 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- Needed to establish an account with Google, similar to what we did with Apple with similar setup and ownership considerations
- Also similar to Apple, there was a need for developers to be familiar with PhoneGap configuration unique to Android
- We were able to leverage existing desktops/laptos and development tools, although we did have to establish additional installations for Android development tools
- Our Windows policies blocked external devices, so we had to establish exceptions for the developers that connected Android devices to their laptops
- Software versioning was already established for our Eclipse environment, so no additional needs were required for Android development

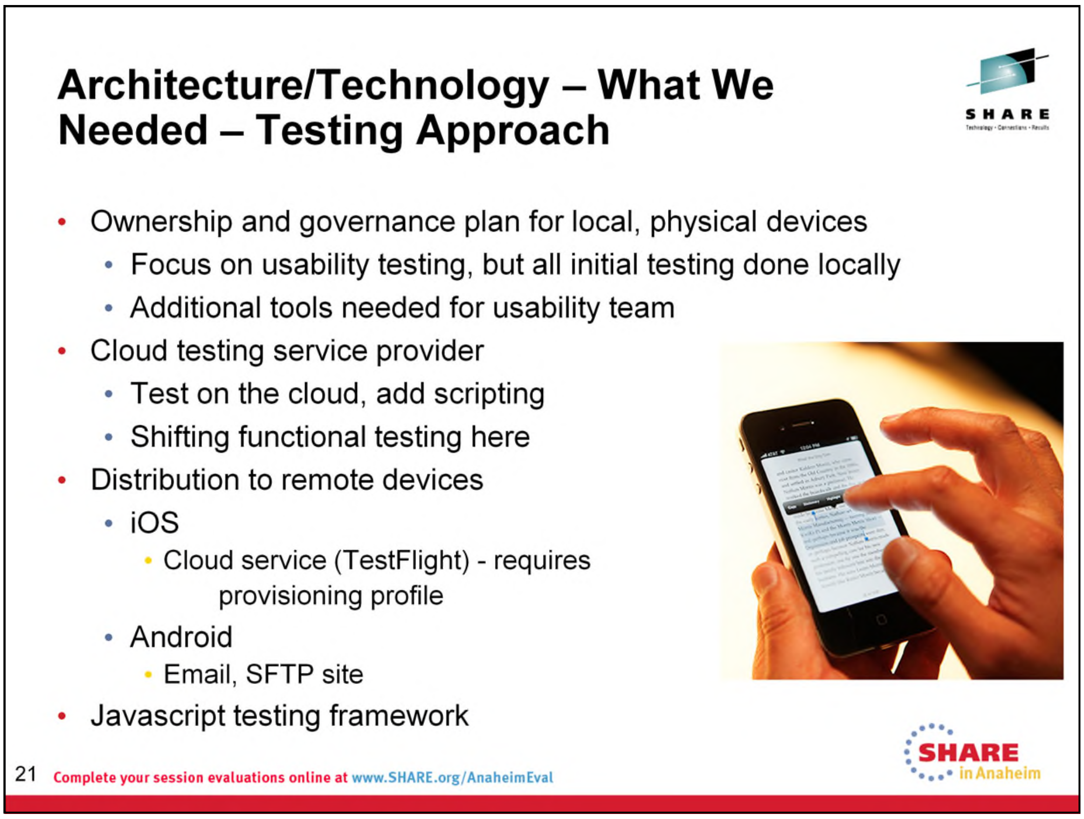


Architecture/Technology – What We Needed – Testing Needs

- Physical device testing
  - Usability needs
  - Developer, business analyst, and customer testing
  - Physical device management
- Testing by remote users
- Unit testing
- REST Service testing
- Usability testing

20 Complete your session evaluations online at www.SHARE.org/AnaheimEval

- This lists some of the testing needs that we identified from different perspectives
- Physical device management
  - Driver installs, policy settings, cables, and more
  - Governance of physical devices
  - Physical device management is not easy



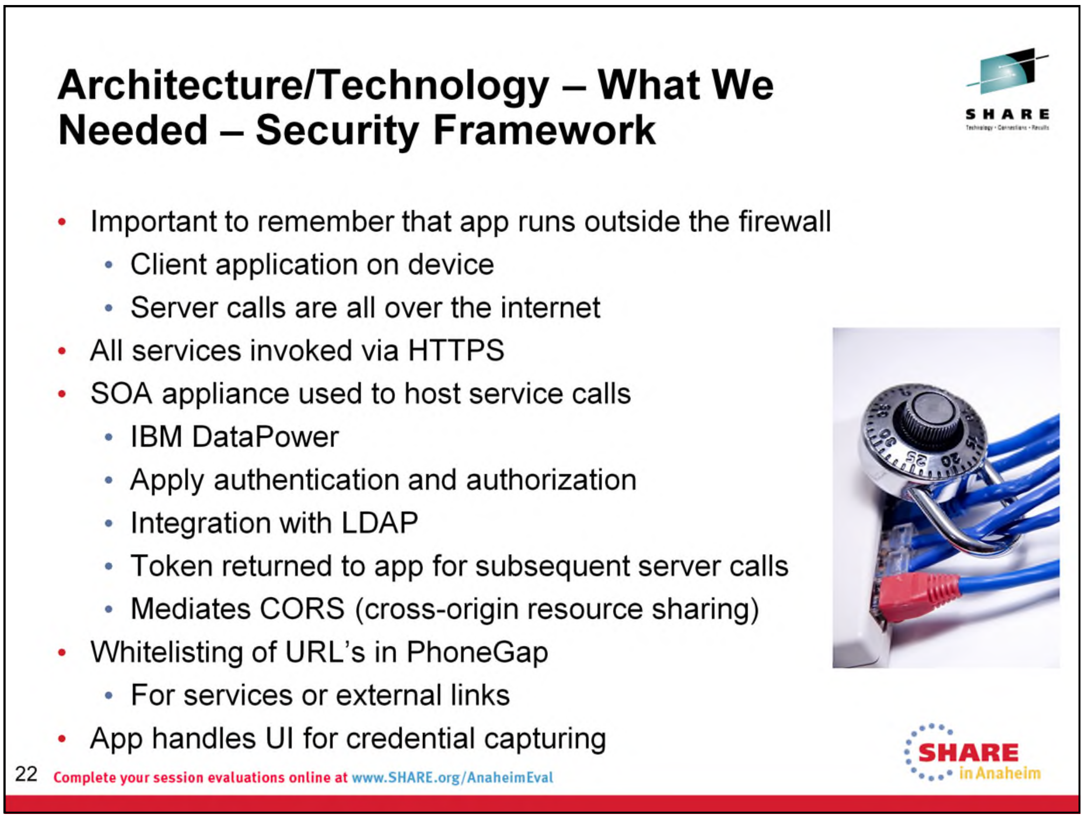
Architecture/Technology – What We Needed – Testing Approach

- Ownership and governance plan for local, physical devices
  - Focus on usability testing, but all initial testing done locally
  - Additional tools needed for usability team
- Cloud testing service provider
  - Test on the cloud, add scripting
  - Shifting functional testing here
- Distribution to remote devices
  - iOS
    - Cloud service (TestFlight) - requires provisioning profile
  - Android
    - Email, SFTP site
- Javascript testing framework

21 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- For our initial foray into mobile development and testing, we went the route of acquiring on hand, physical devices for sharing across developers, testers, and user experience staff
- As an enterprise, we also established a cloud-based service for testing internet-connected devices
  - The biggest value proposition for this was the ability to use automated scripting to replay a script multiple times, including sharing parts of the scripts across devices
- For remote testers we needed to have a method for distributing application versions to other devices
  - We had differences between iOS and Android, but there are online services that assist with this (free)
- Unit testing is part of our development culture, so we also made attempts to build out our unit testing capabilities for Javascript. Dojo has a framework included called DOH (Dojo Objective Harness), but there are others as well.

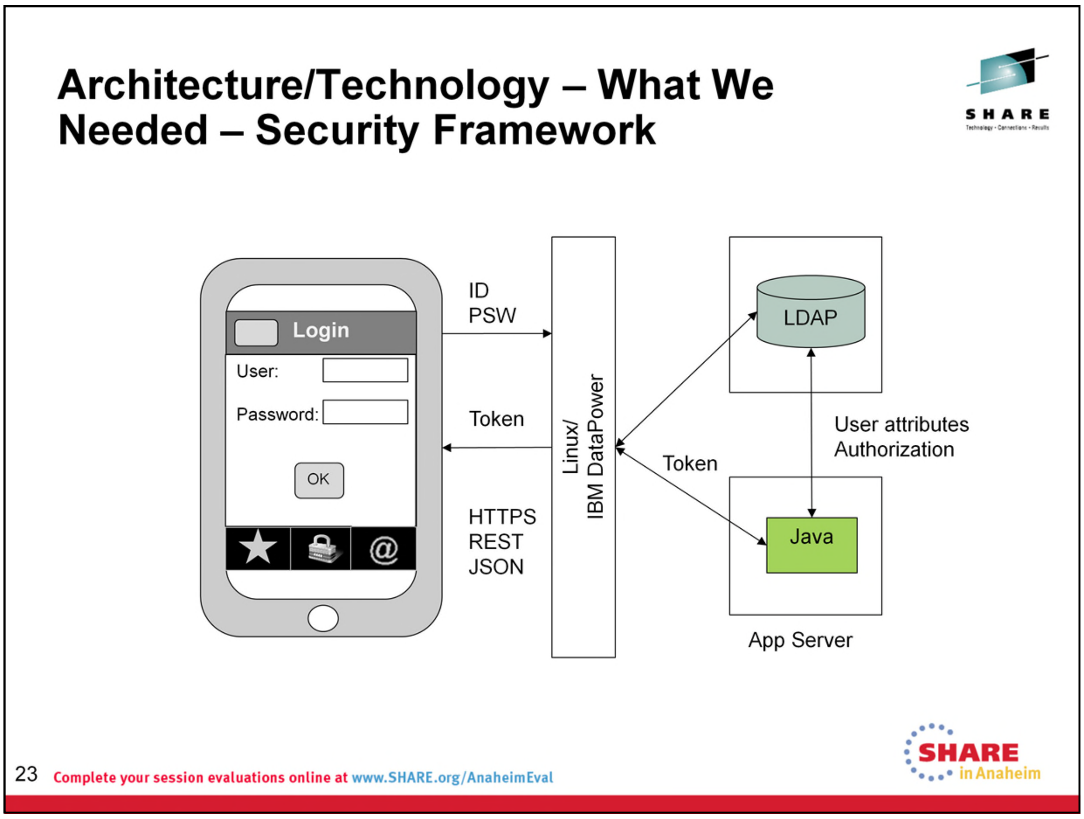


Architecture/Technology – What We Needed – Security Framework

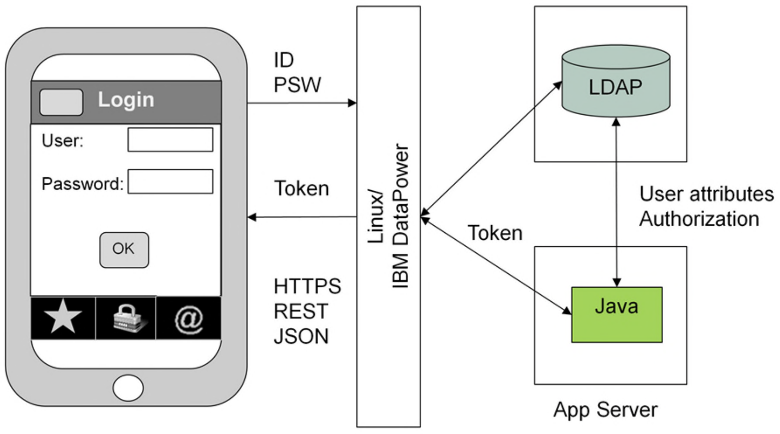
- Important to remember that app runs outside the firewall
  - Client application on device
  - Server calls are all over the internet
- All services invoked via HTTPS
- SOA appliance used to host service calls
  - IBM DataPower
  - Apply authentication and authorization
  - Integration with LDAP
  - Token returned to app for subsequent server calls
  - Mediates CORS (cross-origin resource sharing)
- Whitelisting of URL's in PhoneGap
  - For services or external links
- App handles UI for credential capturing

22 Complete your session evaluations online at www.SHARE.org/AnaheimEval

- Key point was that our devices were operating outside our firewall since the app on the device would make REST service calls to our back end systems
- Having an SOA appliance (DataPower) was key to mediating authentication and authorization for our applications as well as handling CORS
- CORS – Cross-origin resource sharing
  - Normally, a web page cannot make a request to a URL (such as a service call) that is not on the same domain as that from which the page was loaded (security vulnerability)
  - Some SOA appliances will mediate this concern to expose the service on the same domain as the website
  - CORS is still an issue for desktop testing since local server is not the same as the web service sever
    - Started with JSONP and JSON, but required extra coding
    - Implemented local IBM HTTP Server with configuration for desktop testing

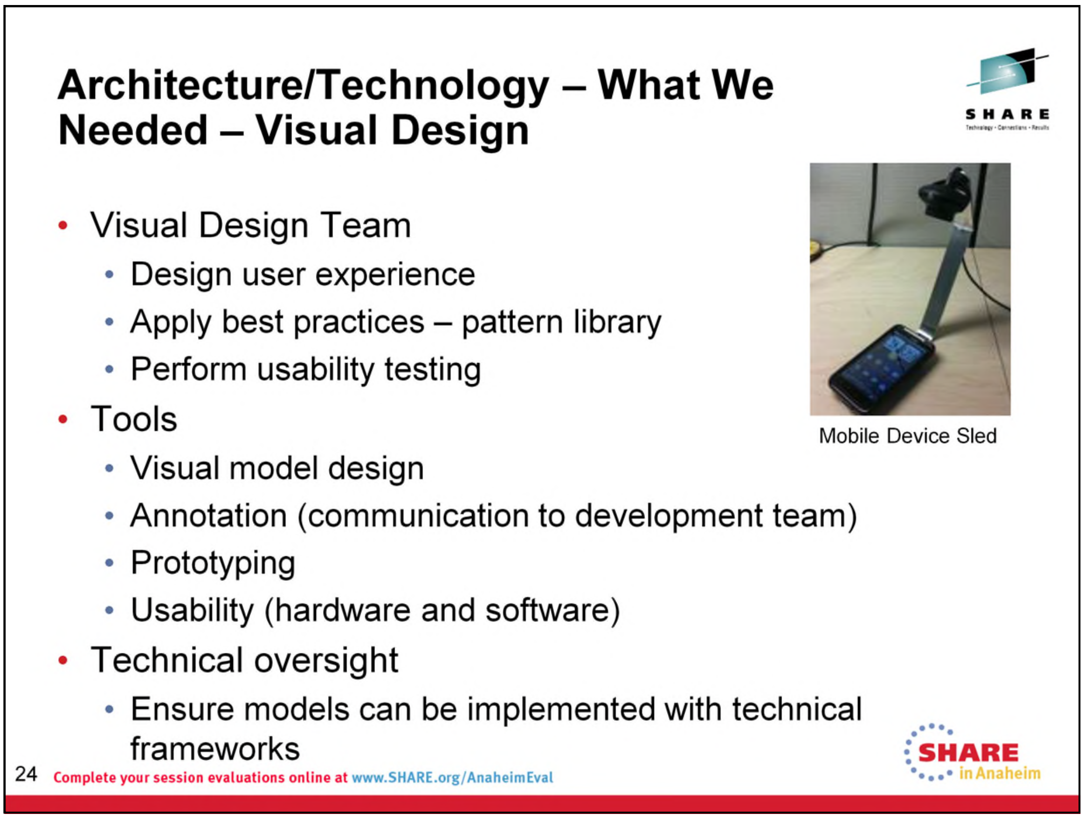


Architecture/Technology – What We Needed – Security Framework



23 Complete your session evaluations online at www.SHARE.org/AnaheimEval

Basic diagram depicting the interaction between the application on the device and access to back-end services through DataPower



Architecture/Technology – What We Needed – Visual Design

- Visual Design Team
  - Design user experience
  - Apply best practices – pattern library
  - Perform usability testing
- Tools
  - Visual model design
  - Annotation (communication to development team)
  - Prototyping
  - Usability (hardware and software)
- Technical oversight
  - Ensure models can be implemented with technical frameworks

Mobile Device Sled

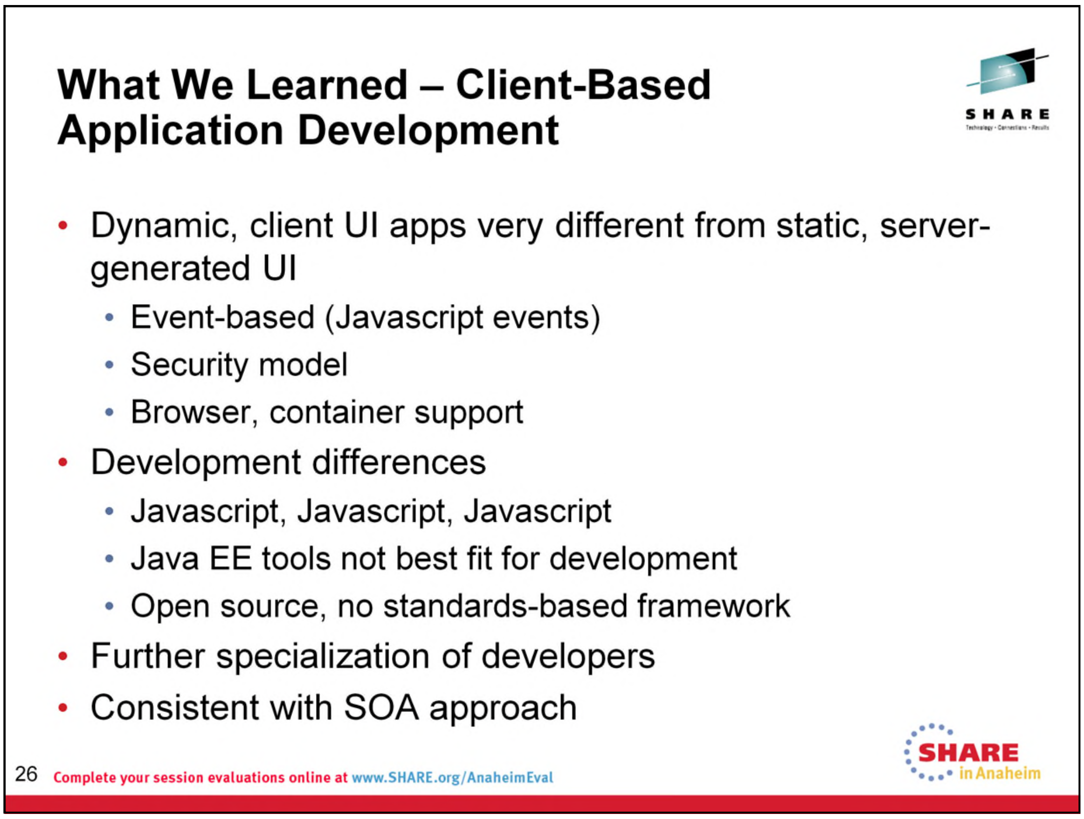
24 Complete your session evaluations online at www.SHARE.org/AnaheimEval

- Visual design for mobile optimized apps was new for our team
  - Mobile actually helped to build visual design experience along with other web initiatives
- Tools
  - Mobile Device Sled - DIY constructed bracket to focus web cam on device for usability testing
  - Tried maqetta.org as a Dojo-based prototyping tool. It was good for trying concepts, but didn't work well for visual designers or conversion from prototype into development
- It was very important to monitor visual design throughout the process to ensure that we could implement the visual design without a lot of customization. In general, we tried to stick to the out of the box Dojo mobile widgets.

What We Learned

- Client-based application development
- Security Implications
- Development & Tools
- Software & Frameworks
- Ongoing Maintenance & Support
- Testing
- Architecture

SHARE in Anaheim

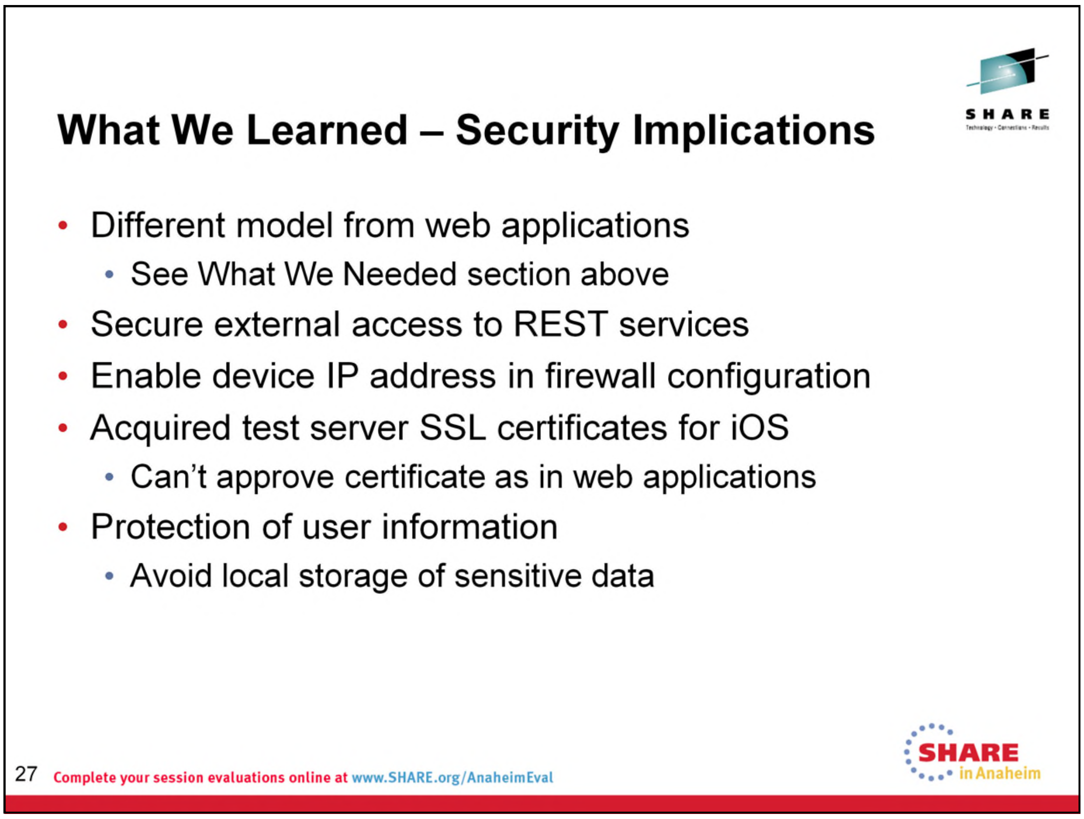


What We Learned – Client-Based Application Development

- Dynamic, client UI apps very different from static, server-generated UI
  - Event-based (Javascript events)
  - Security model
  - Browser, container support
- Development differences
  - Javascript, Javascript, Javascript
  - Java EE tools not best fit for development
  - Open source, no standards-based framework
- Further specialization of developers
- Consistent with SOA approach

26 Complete your session evaluations online at www.SHARE.org/AnaheimEval

- This was a shift in development thinking – client-based apps vs. server-based (at least as compared to previous web development)
- Javascript was the key “language” skill needed by developers
- Open source
  - Versions change regularly
  - Emerging frameworks
- A positive was that our existing SOA approach fit well into a client app that was just for presentation and navigation and we could leverage our existing services
  - Biggest change (with simple implementation) was to adjust services to return JSON instead of XML

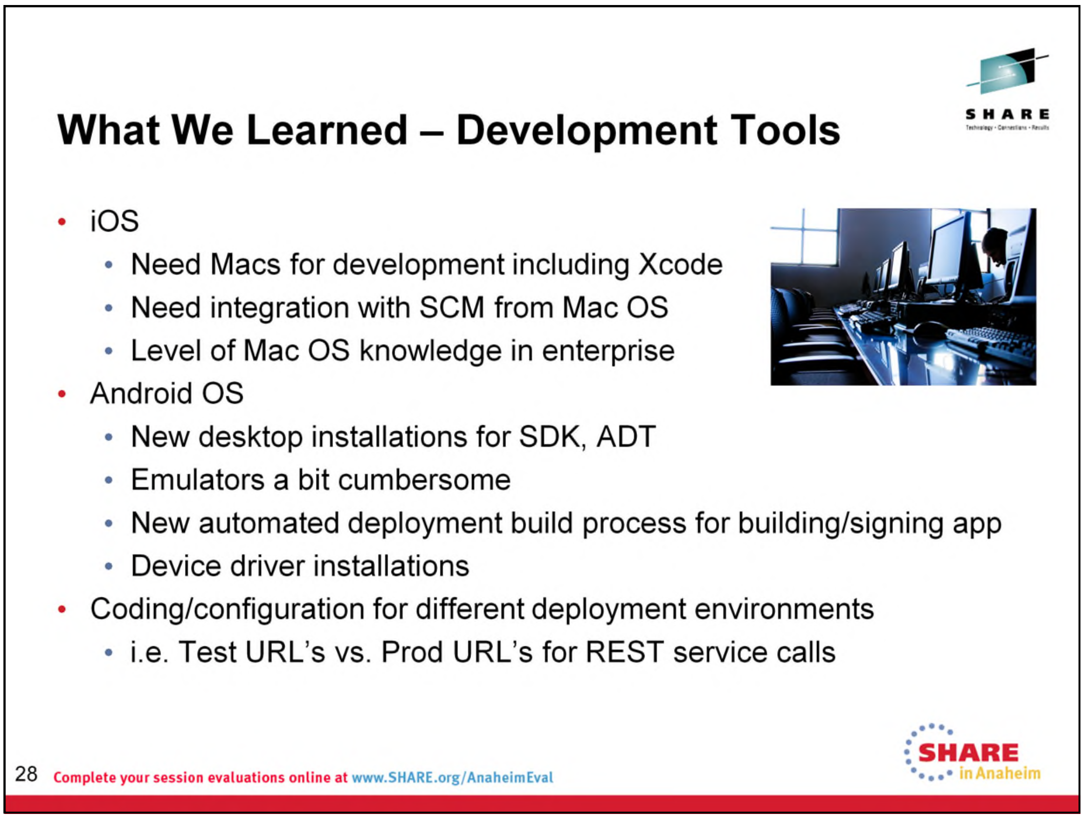


What We Learned – Security Implications

- Different model from web applications
  - See What We Needed section above
- Secure external access to REST services
- Enable device IP address in firewall configuration
- Acquired test server SSL certificates for iOS
  - Can’t approve certificate as in web applications
- Protection of user information
  - Avoid local storage of sensitive data

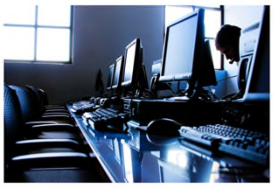
27 Complete your session evaluations online at www.SHARE.org/AnaheimEval

- This was an implication of the application residing on a client device
  - Security calls were from the client instead of server to server (more security required)
- Where we used to have REST service calls from sever-side Java code internal to the network, now the services were invoked by the client app requiring additional security to avoid unauthorized external access
- We had to work with our firewall administrators to open IP addresses for the mobile devices used for testing (blocks of IP addresses based on carrier)
- iOS doesn’t trust non-certified SSL calls, so we had to purchase certificates for our test servers
  - There is a workaround in the Objective C code that we used temporarily until the certificates were applied (found via internet search)
- We also had to be concerned about the protection of end user information and avoid any coding that would store any user information locally on the device



What We Learned – Development Tools

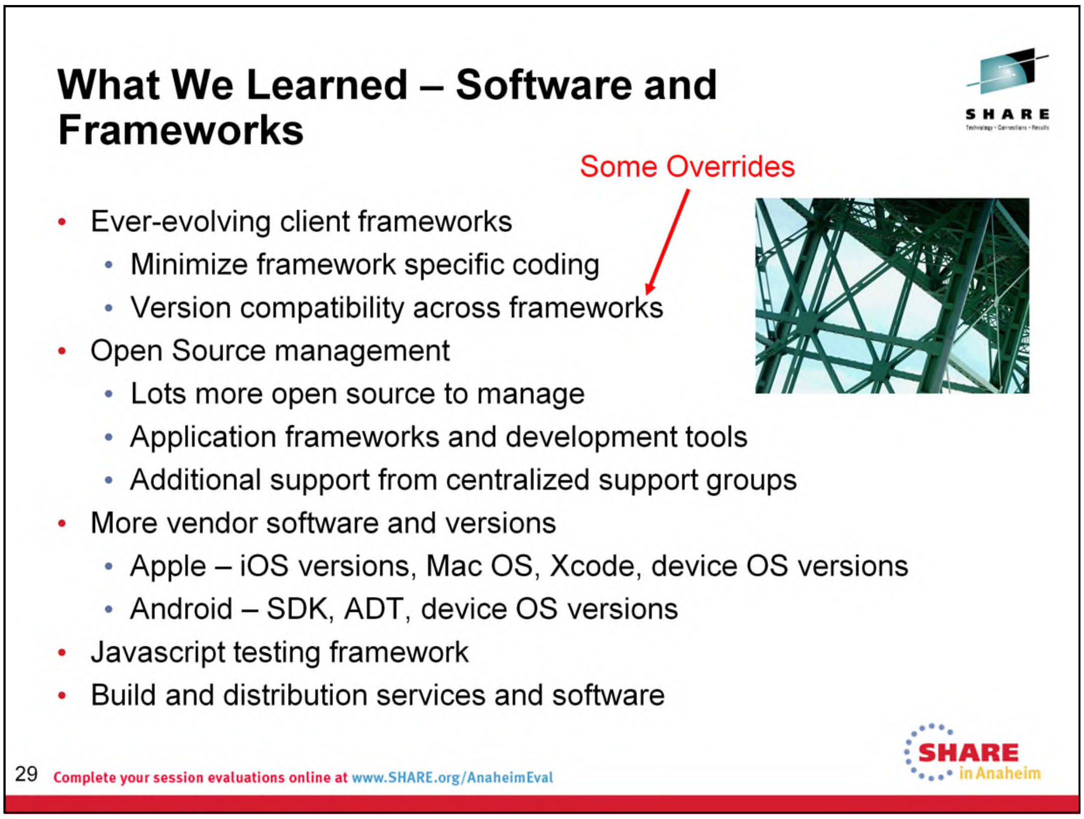
- iOS
  - Need Macs for development including Xcode
  - Need integration with SCM from Mac OS
  - Level of Mac OS knowledge in enterprise
- Android OS
  - New desktop installations for SDK, ADT
  - Emulators a bit cumbersome
  - New automated deployment build process for building/signing app
  - Device driver installations
- Coding/configuration for different deployment environments
  - i.e. Test URL’s vs. Prod URL’s for REST service calls



28 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- For iOS we needed to provide our own support for Macs in the enterprise. We had access to developers who were already familiar with Mac OS and we developed additional expertise internally. It’s a good idea to involve some existing expertise in this area.
- To automate the Android build process, we had to modify our existing scripts and builds to augment with software from Google.
- Ultimately, we leveraged existing knowledge in both the iOS and Android OS arenas.
- Designing the Javascript code to be configurable from environment to environment is key and not necessarily trivial.
  - Our builds generated different environment variables based on the IT or QA deployment environments as well as production
  - Our goal (as in any other app) was to ensure no coding changes were required between deployment environments



What We Learned – Software and Frameworks

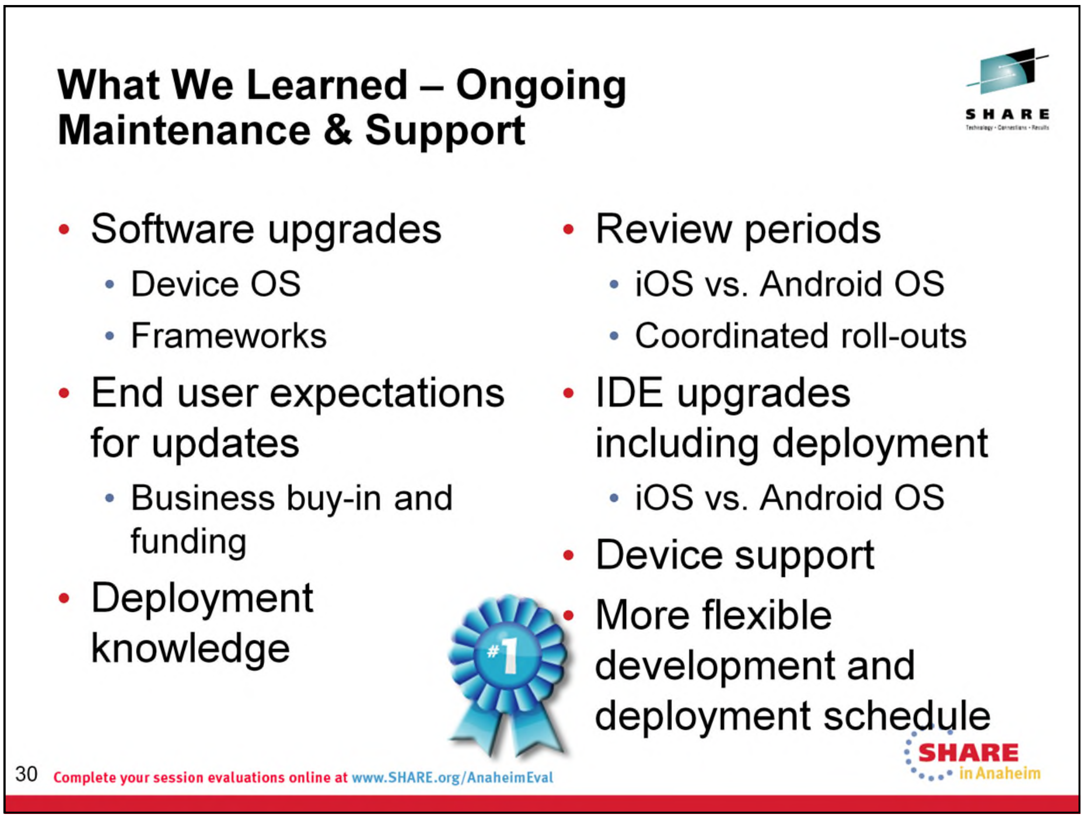
Some Overrides

- Ever-evolving client frameworks
  - Minimize framework specific coding
  - Version compatibility across frameworks
- Open Source management
  - Lots more open source to manage
  - Application frameworks and development tools
  - Additional support from centralized support groups
- More vendor software and versions
  - Apple – iOS versions, Mac OS, Xcode, device OS versions
  - Android – SDK, ADT, device OS versions
- Javascript testing framework
- Build and distribution services and software

29 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- With frameworks ever changing, there is a tendency to shift to new UI frameworks over time. It's helpful to strike a balance between leveraging the framework, while minimizing dependence for future migrations.
- Coordinating versions (i.e. Dojo vs. PhoneGap vs. iOS/Android OS) could be challenging. We did experience some compatibility issues at some points between the different layers. This required some overriding of the framework code based on fixes found on the internet.



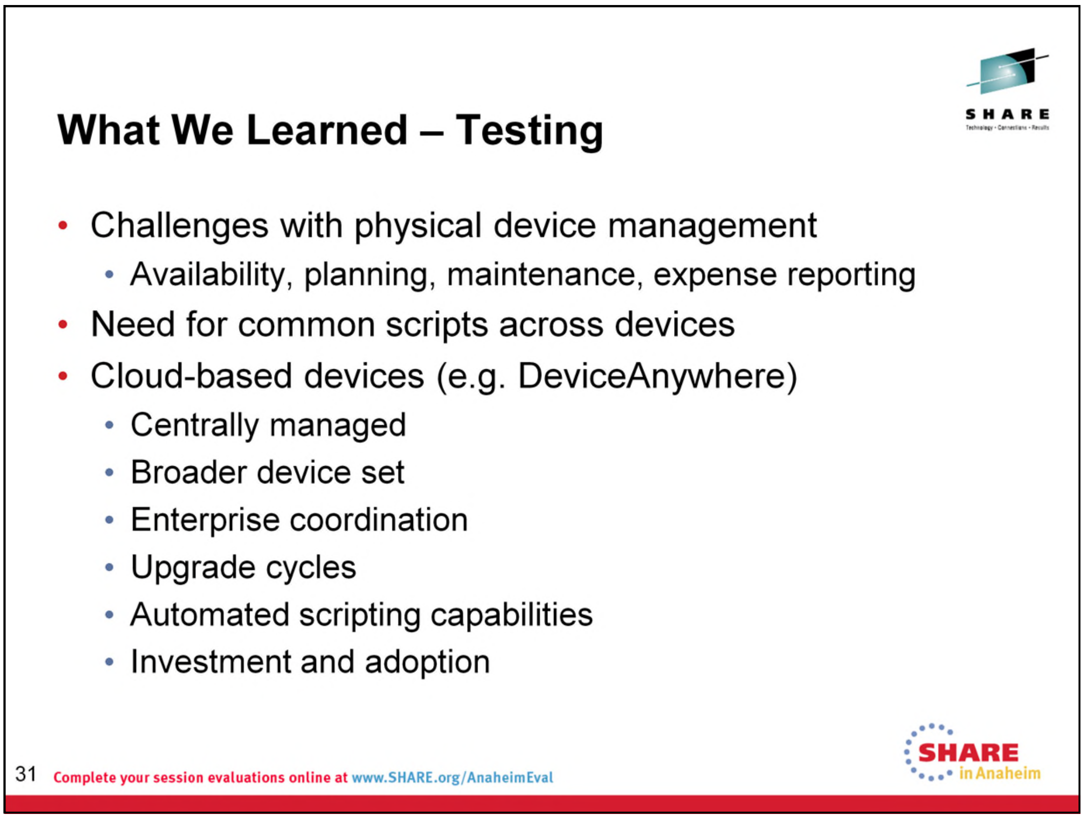
What We Learned – Ongoing Maintenance & Support

- Software upgrades
  - Device OS
  - Frameworks
- End user expectations for updates
  - Business buy-in and funding
- Deployment knowledge
- Review periods
  - iOS vs. Android OS
  - Coordinated roll-outs
- IDE upgrades including deployment
  - iOS vs. Android OS
- Device support
- More flexible development and deployment schedule

30 Complete your session evaluations online at www.SHARE.org/AnaheimEval

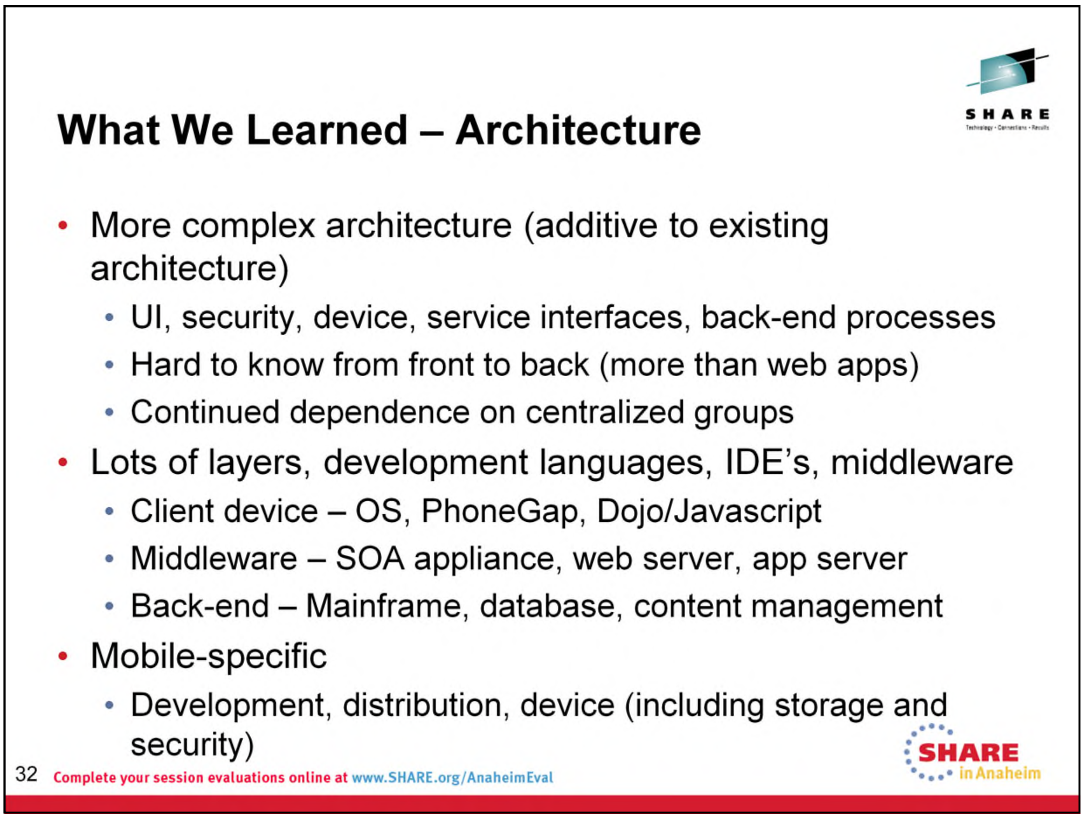
SHARE in Anaheim

- To coordinate iOS and Android releases, we would stage iOS to be reviewed, but requiring manual release. In this way, once the review was accepted by Apple, we could basically release the iOS release and Android simultaneously.
- A side benefit was that we had more flexibility in scheduling development and deployment since the mobile app was mostly stand-alone. The only dependence was on the service interfaces.



What We Learned – Testing

- Challenges with physical device management
  - Availability, planning, maintenance, expense reporting
- Need for common scripts across devices
- Cloud-based devices (e.g. DeviceAnywhere)
  - Centrally managed
  - Broader device set
  - Enterprise coordination
  - Upgrade cycles
  - Automated scripting capabilities
  - Investment and adoption

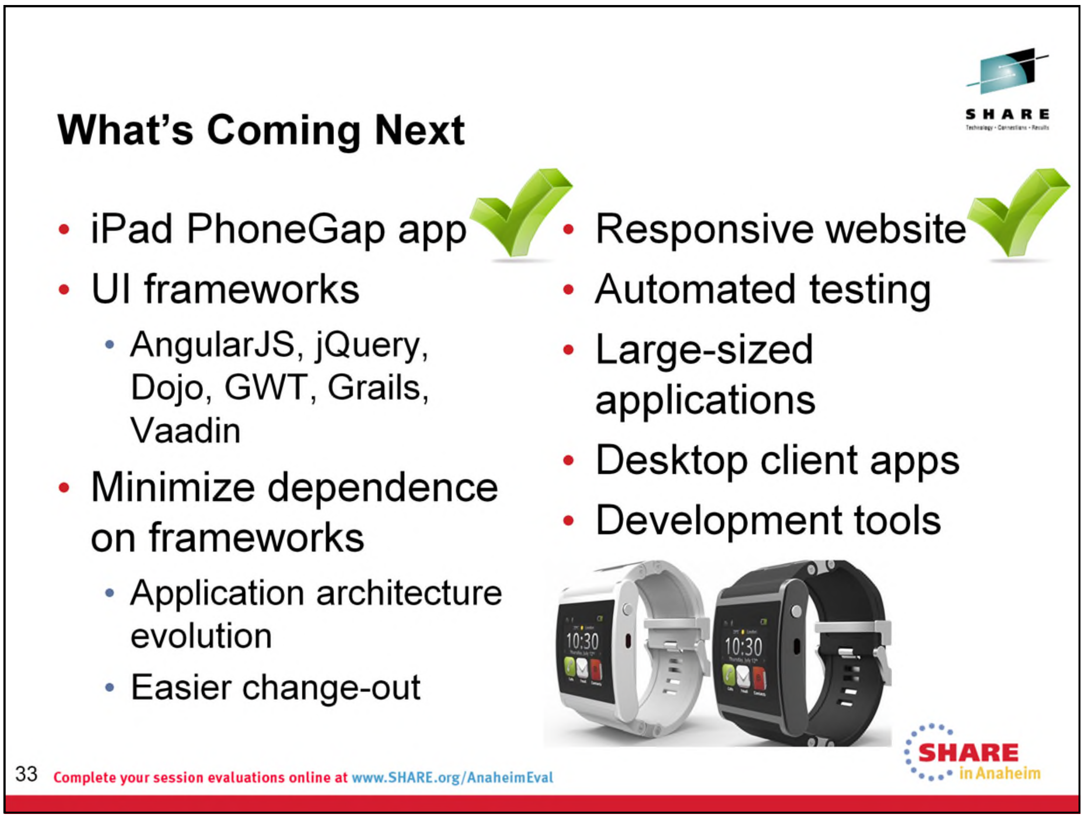


What We Learned – Architecture

- More complex architecture (additive to existing architecture)
  - UI, security, device, service interfaces, back-end processes
  - Hard to know from front to back (more than web apps)
  - Continued dependence on centralized groups
- Lots of layers, development languages, IDE's, middleware
  - Client device – OS, PhoneGap, Dojo/Javascript
  - Middleware – SOA appliance, web server, app server
  - Back-end – Mainframe, database, content management
- Mobile-specific
  - Development, distribution, device (including storage and security)

32 Complete your session evaluations online at www.SHARE.org/AnaheimEval

- We found that not only did the developers need to understand mobile technology, but our architects needed to understand how the mobile client app fits in with all of the rest of our layers of architecture.
- In other words, our architecture was already complex, and we added another layer/variation to it.



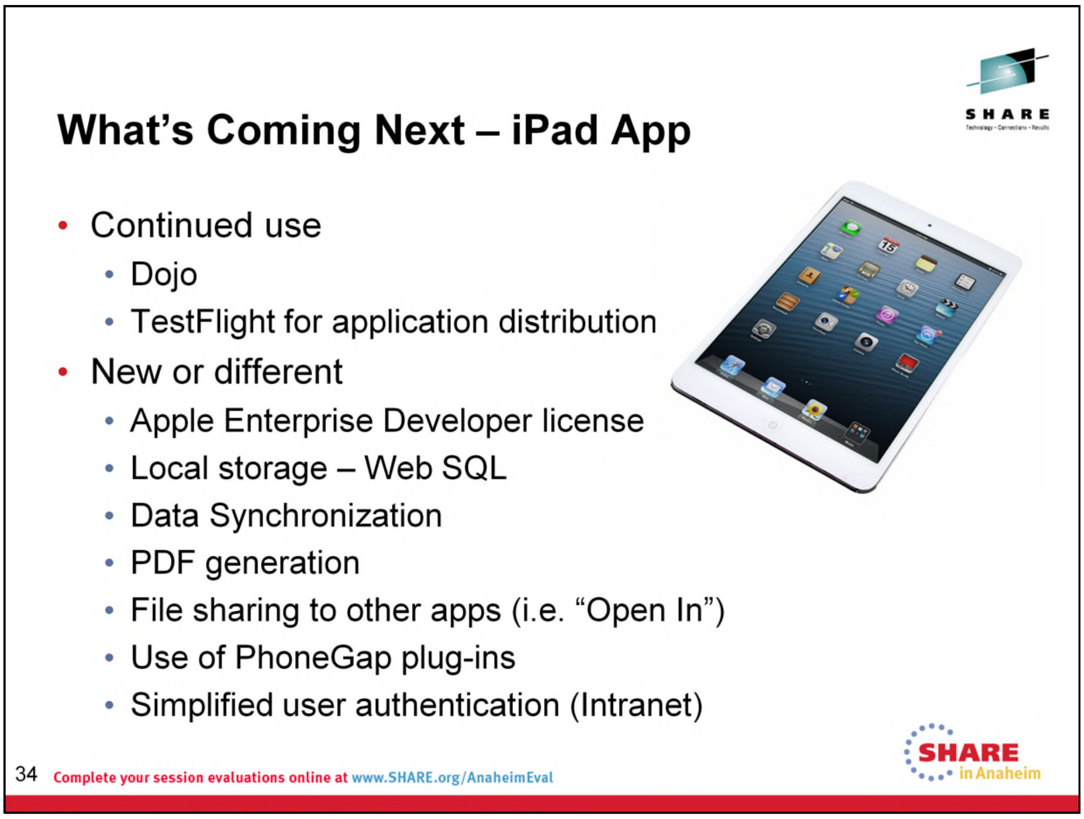
What's Coming Next

- iPad PhoneGap app ✓
- UI frameworks
  - AngularJS, jQuery, Dojo, GWT, Grails, Vaadin
- Minimize dependence on frameworks
  - Application architecture evolution
  - Easier change-out
- Responsive website ✓
- Automated testing
- Large-sized applications
- Desktop client apps
- Development tools

33 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- AngularJS is a promising, emerging UI framework that we are beginning to use for enterprise development (and which has support for mobile)
- As stated earlier, we see a need to limit framework dependence in our code to facilitate future migration to other UI frameworks



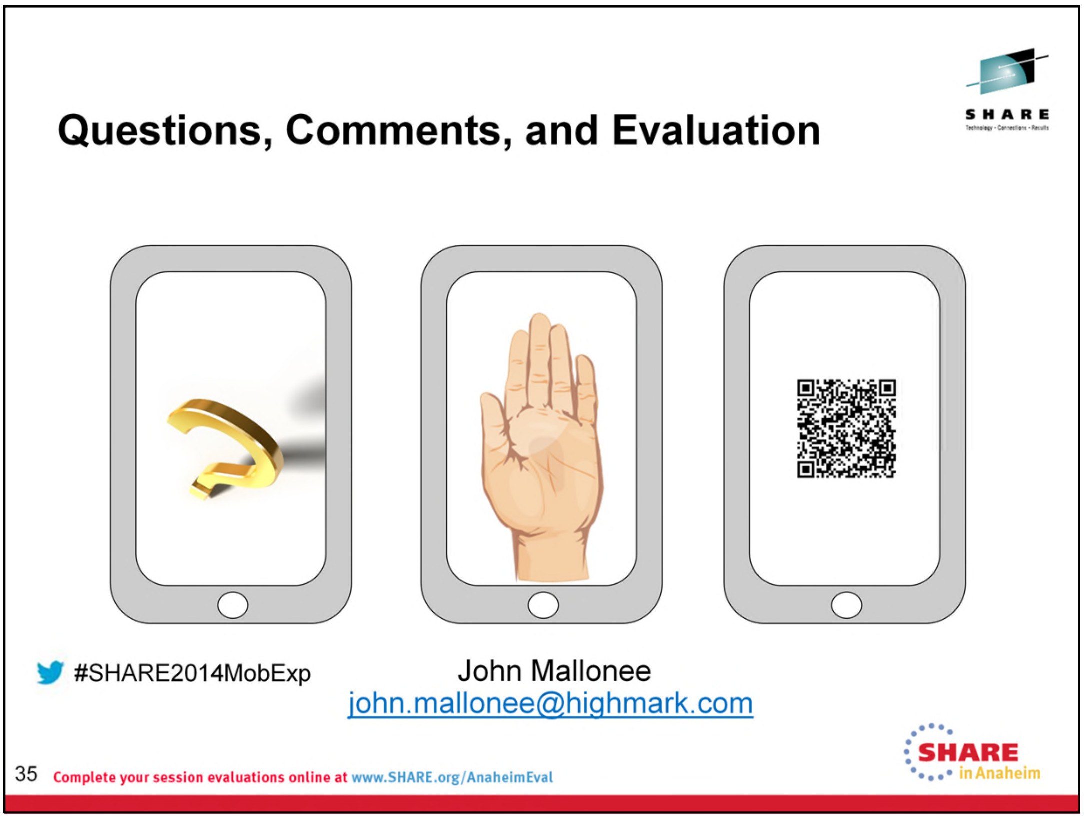
What's Coming Next – iPad App

- Continued use
  - Dojo
  - TestFlight for application distribution
- New or different
  - Apple Enterprise Developer license
  - Local storage – Web SQL
  - Data Synchronization
  - PDF generation
  - File sharing to other apps (i.e. "Open In")
  - Use of PhoneGap plug-ins
  - Simplified user authentication (Intranet)

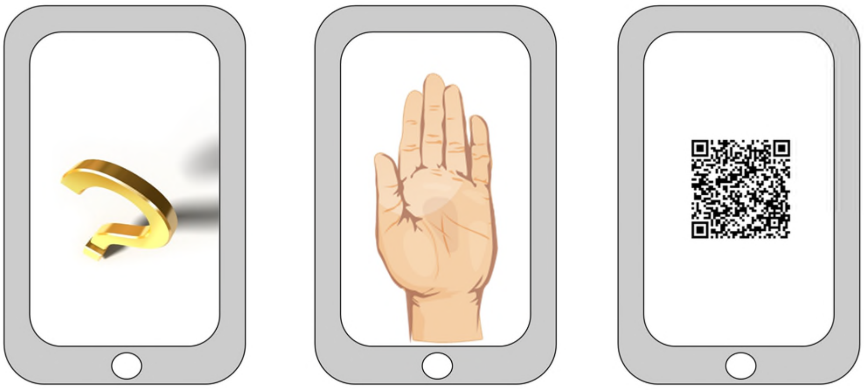
34 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim

- Developing "mobile" applications for internal staff does require an additional license from Apple
  - Developer license is for the App Store
  - Enterprise license is for employees
  - There is also a B2B extension to the App Store for sharing apps with external partners or customers



Questions, Comments, and Evaluation



#SHARE2014MobExp

John Mallonee
john.mallonee@highmark.com

35 Complete your session evaluations online at www.SHARE.org/AnaheimEval

SHARE in Anaheim