ESPN DIGITAL MEDIA

Max Protect: Scalability & Caching at ESPN.com

About the speaker

- **Sean Comerford - Site Architect, ESPN.com**
  - Previous gigs at MLB.com, Sun Micro and IBM
- sean.comerford@espn.com , @scc1976 on Twitter
- First time at QCon!

WHO NEEDS TRAINING TO KEEP UP WITH TECHNOLOGY TRENDS?
ME.
Dilbert.com DilbertCartoonist@gmail.com
YOU'RE FIRED. I ONLY WANT PEOPLE WHO ALREADY KNOW HOW TO DO THEIR JOBS.
11-16-09 ©2009 Scott Adams, Inc./Dist. by UFS, Inc.
I DID NOT SEE THAT COMING.
THEY DON'T HAVE A CLASS TO FIX THAT.

Agenda

- High Level Architecture
- Technology Deep Dive
  - ESPN.com Re-Arechitecture
    - Cache Push
    - SOA
  - Dynamic Content System
  - Live Scores
  - Personalization
- Coming Attractions & Help Wanted
- Q&A

ESPN.com Facts & Figures

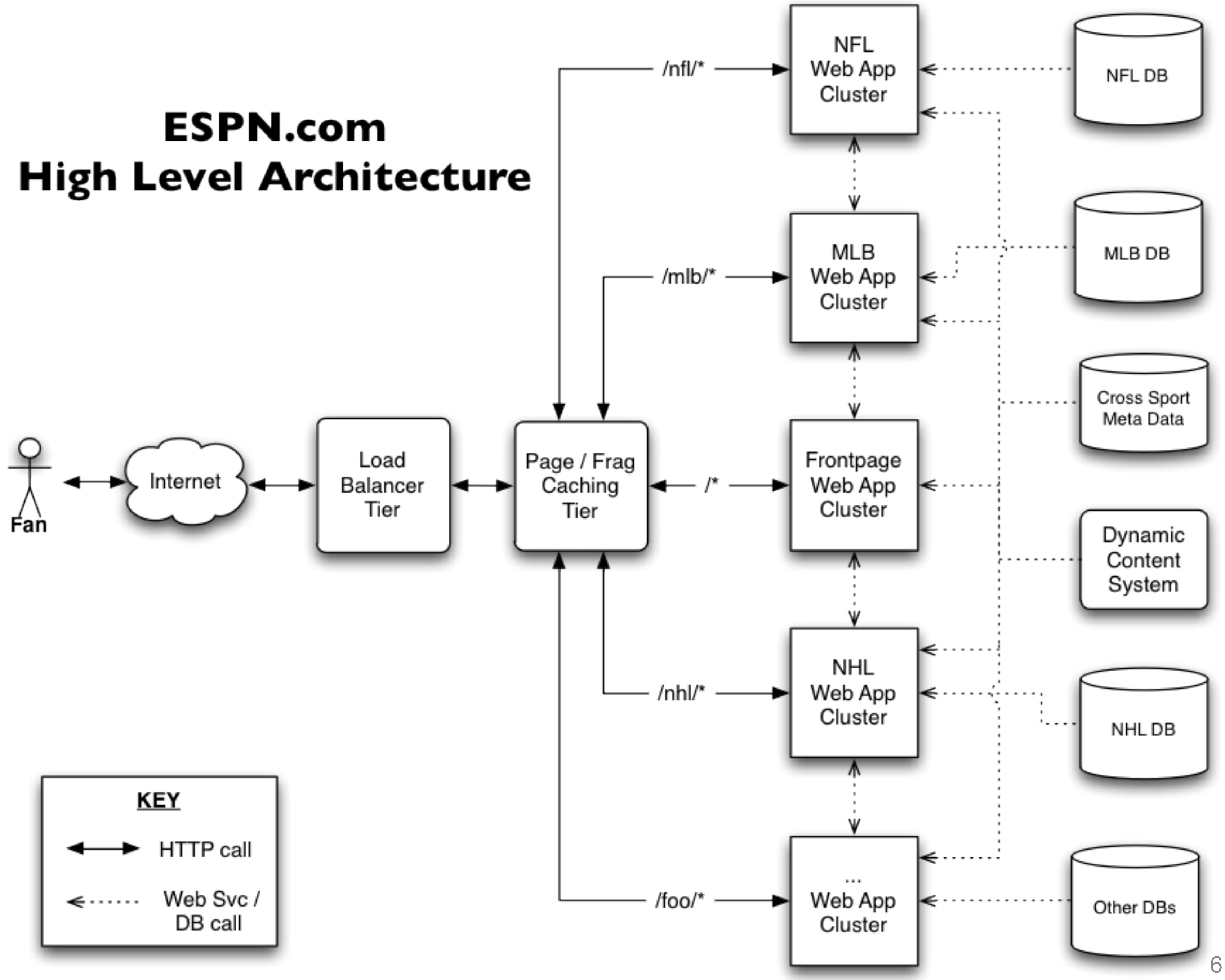
- Internet's #1 sports web site
- Top 10 (all sites) in terms of viewership
- Almost entirely Java based
- Serves 10s of thousands to 100s of thousands of requests per second with a relatively small number of servers
- ESPN digital properties include
  - ESPN.com
  - Fantasy games
  - Mobile
  - WatchESPN
  - ESPN the Ocho
    - No, not yet but others (Deports, W, HS, etc)

ESPN.com Mission

- Serve sports fans anytime, anywhere on any device
- Availability & accuracy of the utmost important
  - You wouldn't tolerate ESPN going out for 10 seconds on your TV
  - You shouldn't tolerate it for ESPN.com either
- Bring fan all stats and scores + more and deeper content
- ESPN has a deep appreciation for technology…
  - But we're not a technology company – we're a media & content provider

ESPN DIGITAL MEDIA

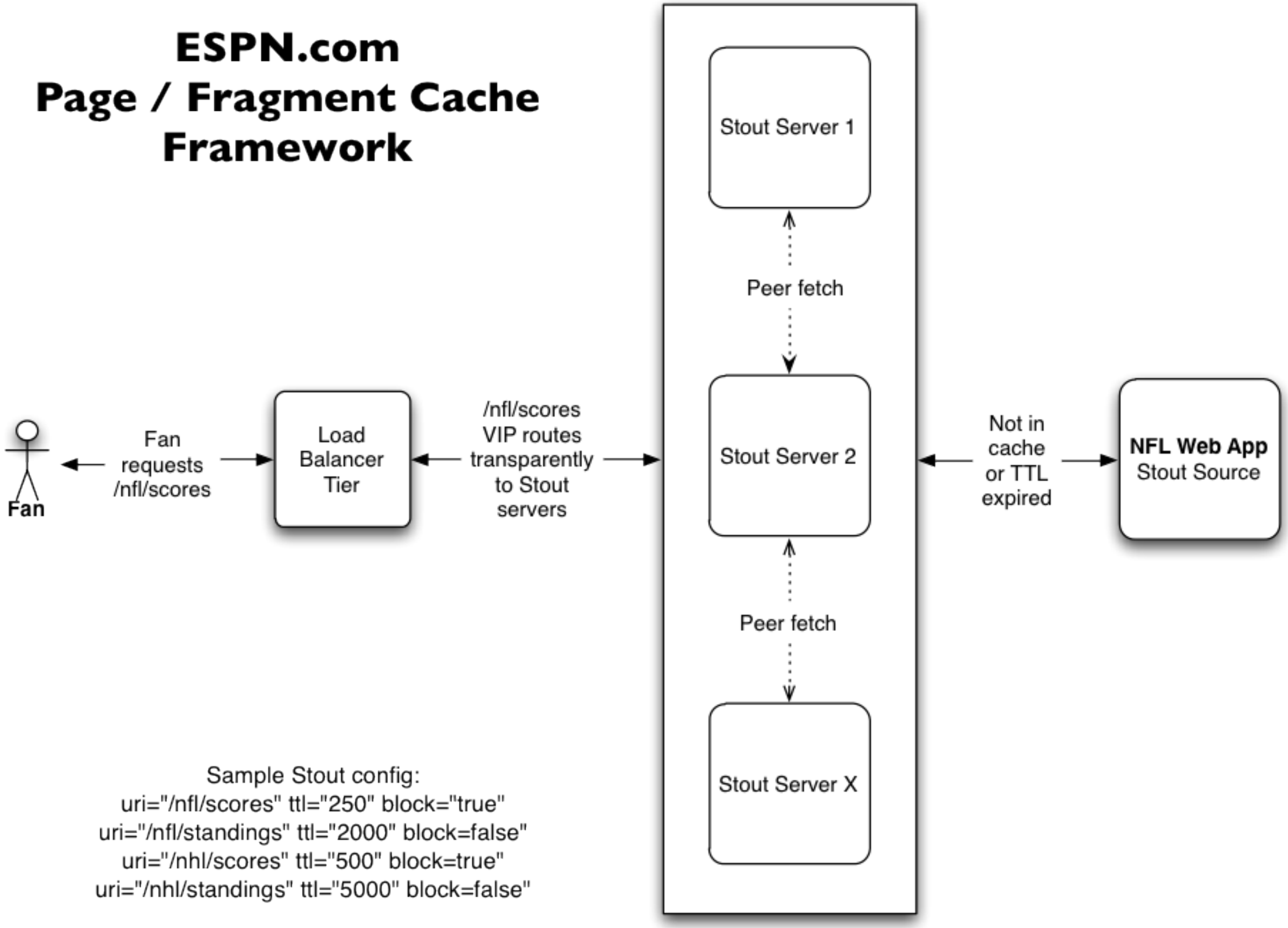
ESPN.com High Level Architecture



Page Caching Framework

- High performance page and fragment caching
- Replicated / peer fetch enabled
- Per URI, TTL based expiration
- Blocking and non-blocking source fetch
  - Low TTL, block for scoreboard → highest accuracy
  - High TTL, don't block for schedule → not updated frequently
- Automatically demotes unresponsive source servers
- Runs on cheap / low end hardware
  - 100s of thousands of requests to load balancer
  - 100s of requests to actual app server
  - 10 MLB servers instead of 50 → save big $$$

ESPN.com Page / Fragment Cache Framework



Sample Stout config:

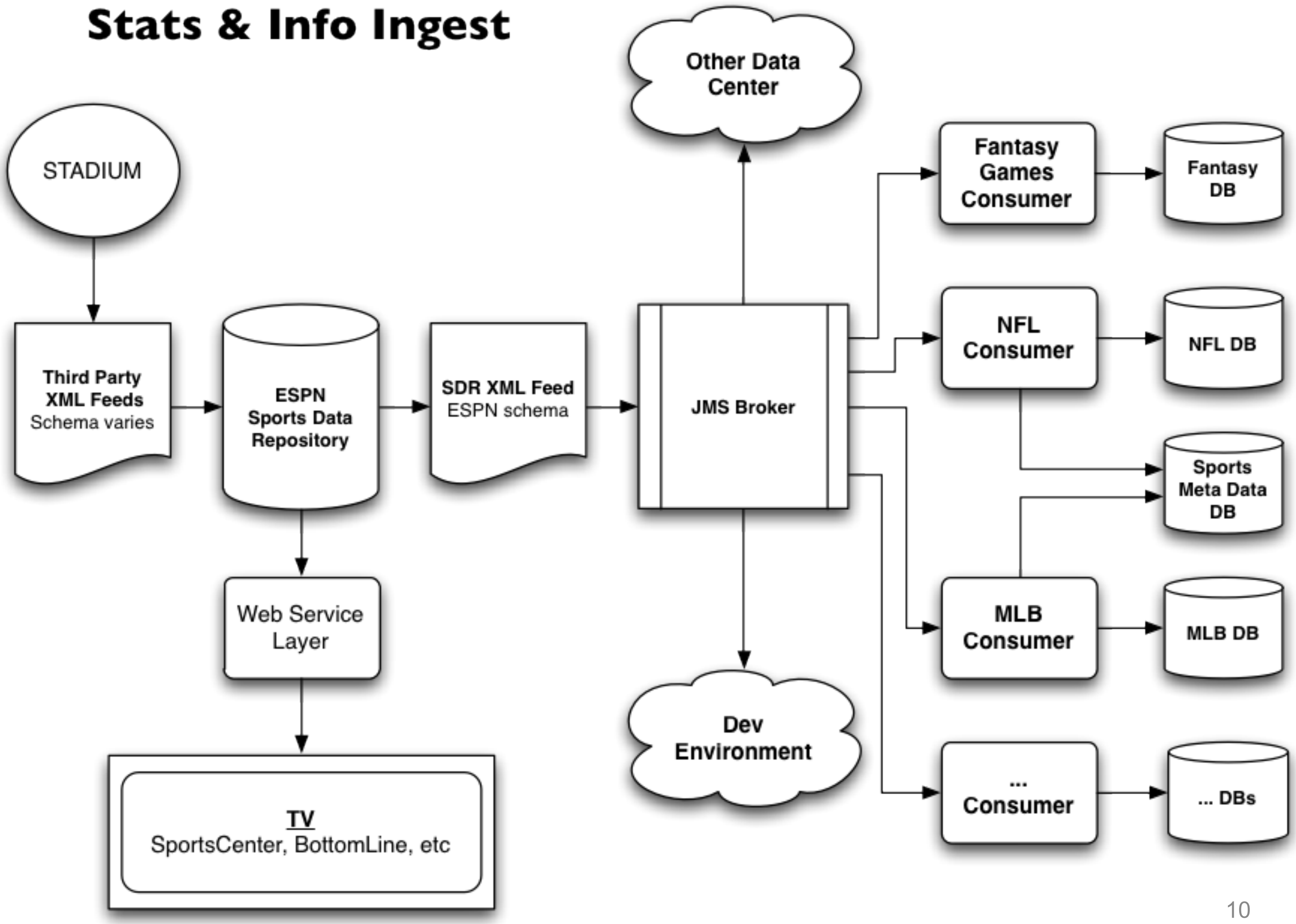
```
uri="/nfl/scores" ttl="250" block="true"
uri="/nfl/standings" ttl="2000" block=false"
uri="/nhl/scores" ttl="500" block=true"
uri="/nhl/standings" ttl="5000" block=false"
```

ESPN.com Data Ingest

- Most stats come from 3rd party vendors or the pro leagues themselves
- Some stats entered by ESPN stats team
- Almost all are overwritten nightly with “official” stats from a 3rd party
- Same stats that power .com also power TV
    - But not accessed in same way
- Relatively speaking, message rates are not very high…
- But complicated by fact almost all in game events need to be processed in order

ESPN DIGITAL MEDIA

ESPN.com Stats & Info Ingest



ESPN.com Application Architecture

- Proprietary, high performance templating framework
  - Think stripped down JSP with built in Spring-like service injection framework
  - Page latency very important for scaling and fan experience
    - Slow page → out of date scores
    - Limit what web devs can do so they don't take down the site
  - Looking at switching to Grails
    - Performance not great in V1.x
    - Looking into V2 which is better

ESPN.com Application Architecture: Application Level Caching

- Historically a “table in memory” view of DB
  - No composition so tons of logic in templates to cobble together a boxscore from 25 different tables
- Replicated (per server) in memory HashMap cache
  - Cache expiration by DB sending expire msg → webapp fetching again from DB
  - Works and simple but O^n performance
    - DB becomes bottleneck as # of servers goes up (expire stampede)

ESPN DIGITAL MEDIA

Re-Architecting ESPN.com

I NEED YOU TO WORK ON A HIGHLY CONFIDENTIAL PROJECT.
Dilbert.com DilbertCartoonist@gmail.com
WHEN YOU'RE DONE, I WANT YOU TO DIG YOUR OWN SHALLOW GRAVE AND BEAT YOURSELF TO DEATH WITH THE SHOVEL.
8-9-10 © 2010 Scott Adams, Inc./Dist. by UFS, Inc.
WHY DOES IT FEEL AS IF MY ENTIRE CAREER HAS BEEN PREPARATION FOR THIS PROJECT?
YOU'RE WELCOME.

Re-Architecting ESPN.com

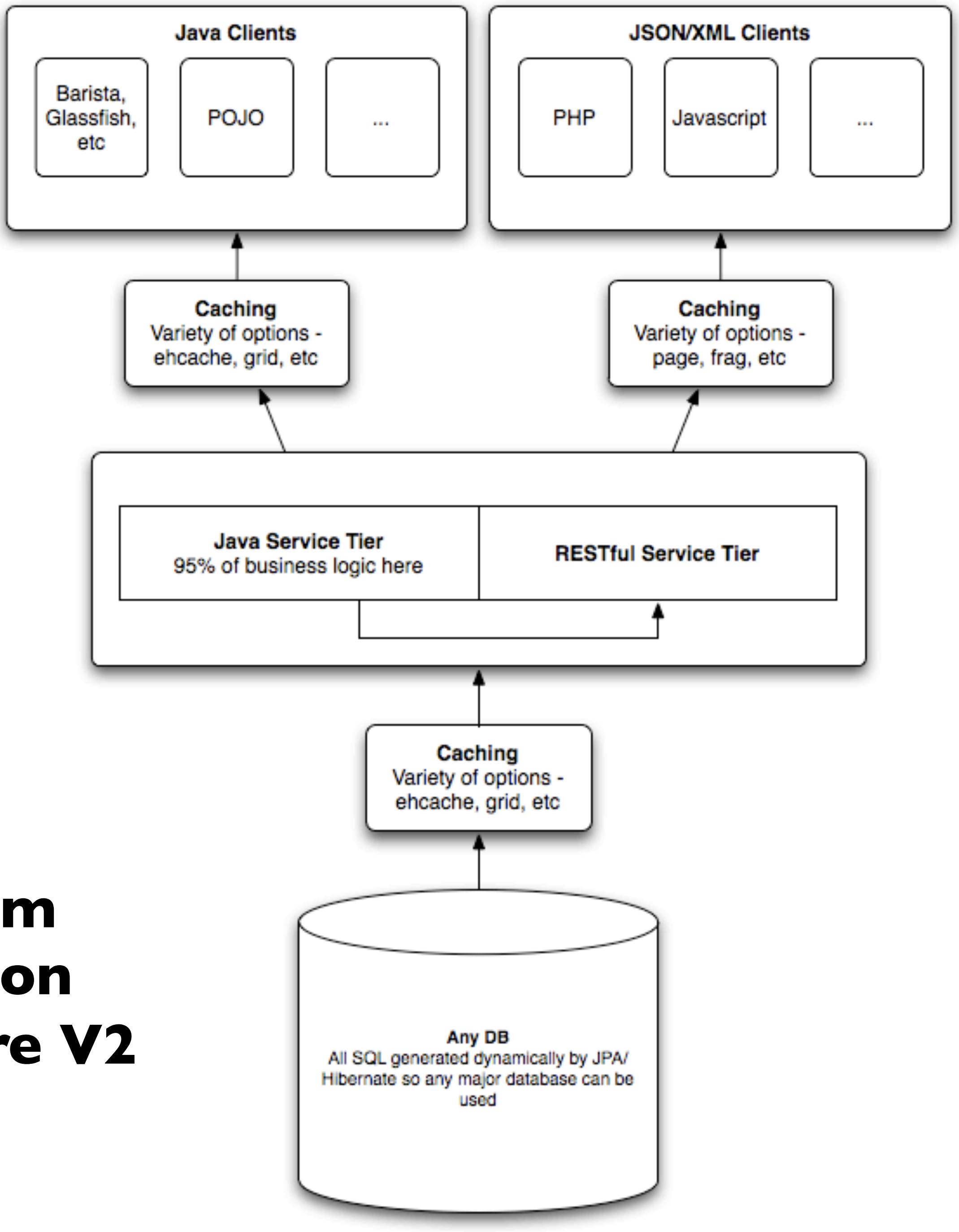
- Replace existing per sport data model with new “common” sports model
  - Common APIs for all data
  - Common APIs for lookup of that data
  - Sport specific extensions where necessary
- Rich JPA/Hibernate based domain model allows us to:
  - Store/retrieve NFL game same as MLB same as...
    - Simplify aggregation & display of all sports data
  - Automagically create RESTful end points via JAX-RS
  - Easily build a more service oriented architecture
    - EJB w/ Hessian encoding + client cache for Java
    - REST for non-Java / lower performance

Re-Architecting ESPN.com
Current State of SOA

HOW'S YOUR PROJECT COMING ALONG?
IT'S A STEAMING PILE OF FAILURE.
scottadams@aol.com
www.dilbert.com
IT'S LIKE FIFTEEN DRUNKEN MONKEYS WITH A JIGSAW PUZZLE.
© 2007 Scott Adams, Inc./Dist. by UFS, Inc.
HOW'S YOUR PROJECT COMING ALONG?
FINE.
© Scott Adams, Inc./Dist. by UFS, Inc.

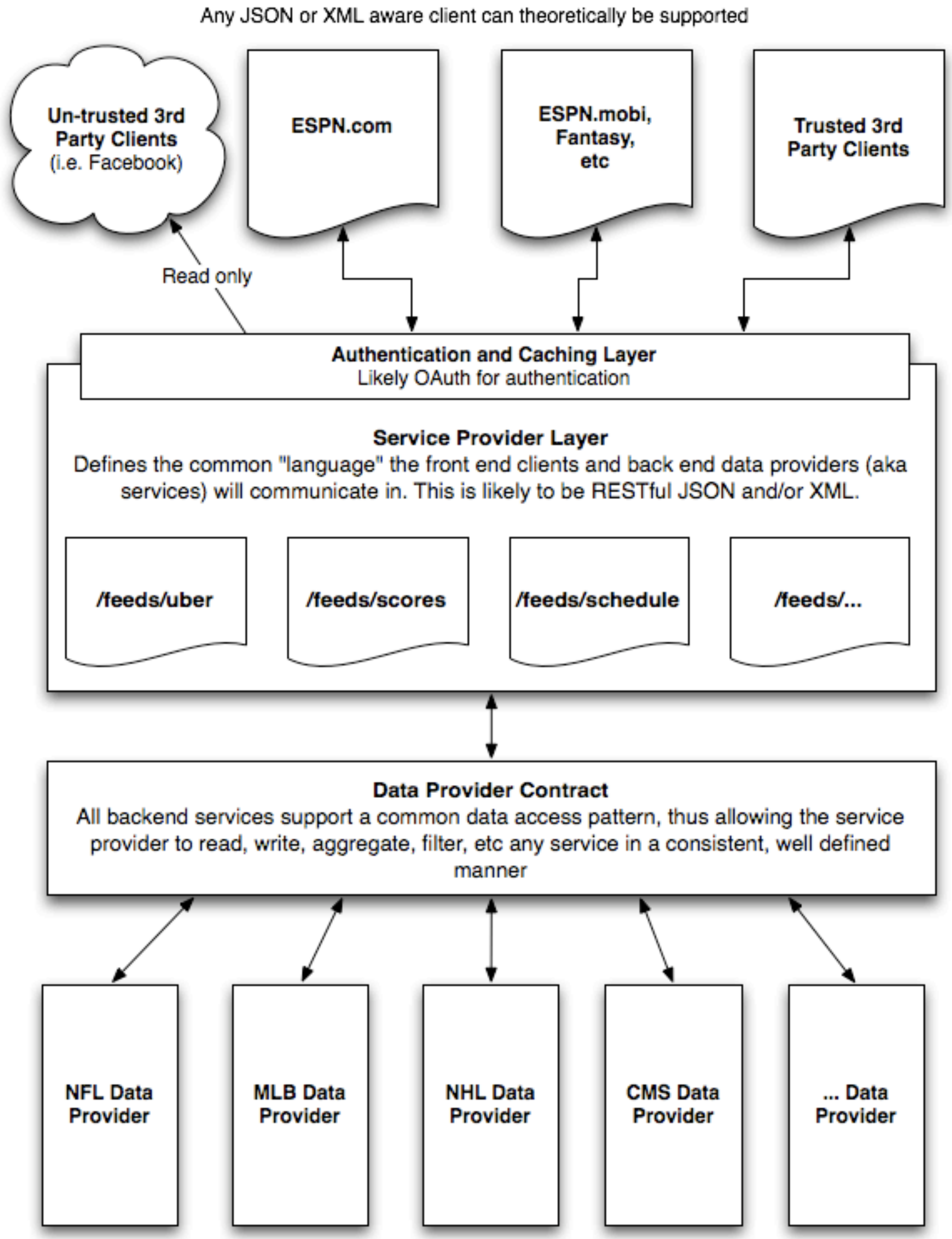
Re-Architecting ESPN.com More SOA-like

- Decouple our services – all new apps can work:
    - Locally (same JVM)
    - Via Java remoting (EJB 3 w/ custom serialization)
    - RESTfullly (via JAX-RS)
  - –Can change with no modification to front end code
- All three methods leverage same persistence & DAO layer
- Have successfully converted a few of our major apps
  - –Leveraging generics and code gen to expedite process of converting the rest
- Move away from having per sport back end service
  - –Create one service for all sports
  - –Modular / flexible front end presentation to have a single scoreboard for all sports


Java Clients
Barista, Glassfish, etc
POJO
...
JSON/XML Clients
PHP
Javascript
...
Caching
Variety of options - ehcache, grid, etc
Caching
Variety of options - page, frag, etc
Java Service Tier
95% of business logic here
RESTful Service Tier
Caching
Variety of options - ehcache, grid, etc
Any DB
All SQL generated dynamically by JPA/ Hibernate so any major database can be used

ESPN.com Application Architecture V2

ESPN.com App Arch V3


Any JSON or XML aware client can theoretically be supported
Un-trusted 3rd Party Clients (i.e. Facebook)
ESPN.com
ESPN.mobi, Fantasy, etc
Trusted 3rd Party Clients
Read only
Authentication and Caching Layer
Likely OAuth for authentication
Service Provider Layer
Defines the common "language" the front end clients and back end data providers (aka services) will communicate in. This is likely to be RESTful JSON and/or XML.
/feeds/uber
/feeds/scores
/feeds/schedule
/feeds/...
Data Provider Contract
All backend services support a common data access pattern, thus allowing the service provider to read, write, aggregate, filter, etc any service in a consistent, well defined manner
NFL Data Provider
MLB Data Provider
NHL Data Provider
CMS Data Provider
... Data Provider

Re-Architecting ESPN.com Cache Push

- Moving to cache push model
- Our data ingest process has already converted incoming XML message into JPA POJO
  - Inefficient to have DB send expire
  - MDBs and webapp both talk POJO so just push it
  - Remove biggest bottle neck (our DB) from the equation
- All live event data gets delivered to the web application by ingest (MDB) process
  - Eliminates millions of DB calls per hour during peak times
  - Once an application has primed its caches with historical & meta data, DB could theoretically be turned off

Deep Dive: Cache Push

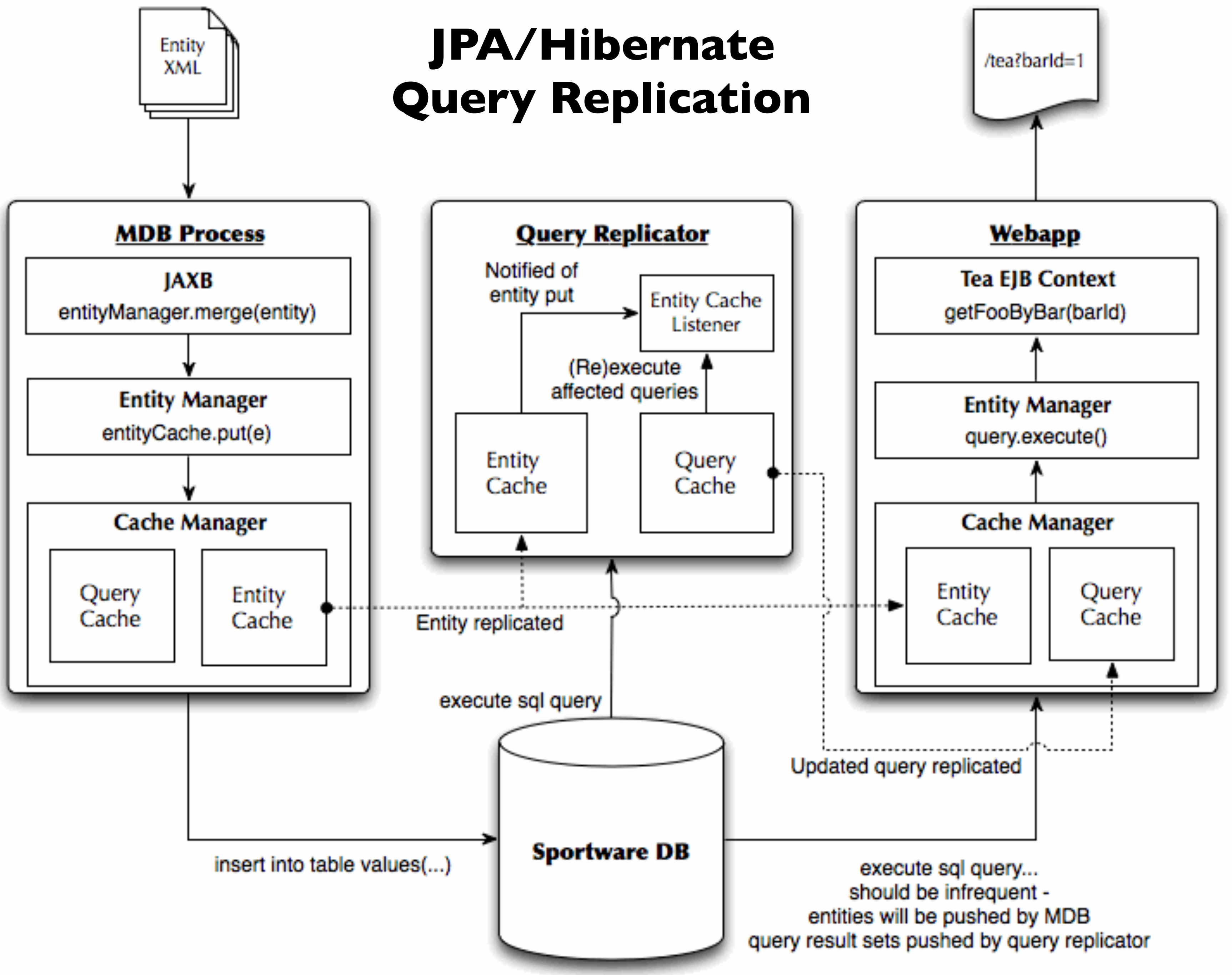
- Using Ehcache as 2nd level cache provider with cache replication enabled
- Works great for entity updates
- **BIG PROBLEM: almost everything on .com looked up via query…**
  - Hibernate support for query caching inefficient
  - Can’t have getPlaysForGame(1234) query banging on DB all day b/c query caceh only supports TTL or dopey last update timestamp
  - What to do….

ESPN DIGITAL MEDIA

Deep Dive: Cache Push – Query Replication

- Enter the “Query replicator”
- Basically a rules engine to dictate what queries to update when an event happens
  - Example: play is inserted for game w/ ID 1234, re-run the query getPlaysForGame(1234)
- Configured via XML mappings files that defines
  - What entities to act upon and for which actions (insert, update, delete)
  - The Java DAO class and method to invoke so query for affected entity are refreshed in cache
- Uses standard mappings file XSD and reflection APIs to work for any use case
- Run the query ONCE and replicate to all apps

JPA/Hibernate Query Replication



Deep Dive: Cache Push Demo

- Let’s do something fun

Deep Dive: Dynamic Content System

- GoPublish system used by majority of Disney family of web sites for content management
- Consists of two major pieces
  - Content Management Service(CMS)
  - Dynamic Content Service (DCS)
- Content can be input either manually (a writer types it in) or via feed consumption (scraping an RSS feed)
- Customizable support for workflow tasks
  - From basic stuff such as writer enters story but not published until editor approves…
  - To detailed user management such as Bob is allowed to create content types but NOT actual content and only for NHL

ESPN DIGITAL MEDIA

Deep Dive: Dynamic Content System - CMS

- Content Management Service
  - Stores both published and unpublished content & types
  - Content Editor (CE): GUI for writers to input content and types
    - Also for basic WYSIWYG layout and formatting
  - Content types are used to group things and in hash map style look ups
    - Give me all content with type “NFL recap” and event date of today
    - Example content types include: preview; recap; frontpage carousel item;
  - Content tagging and aggregation a big focus
  - Backend is standard SQL DB

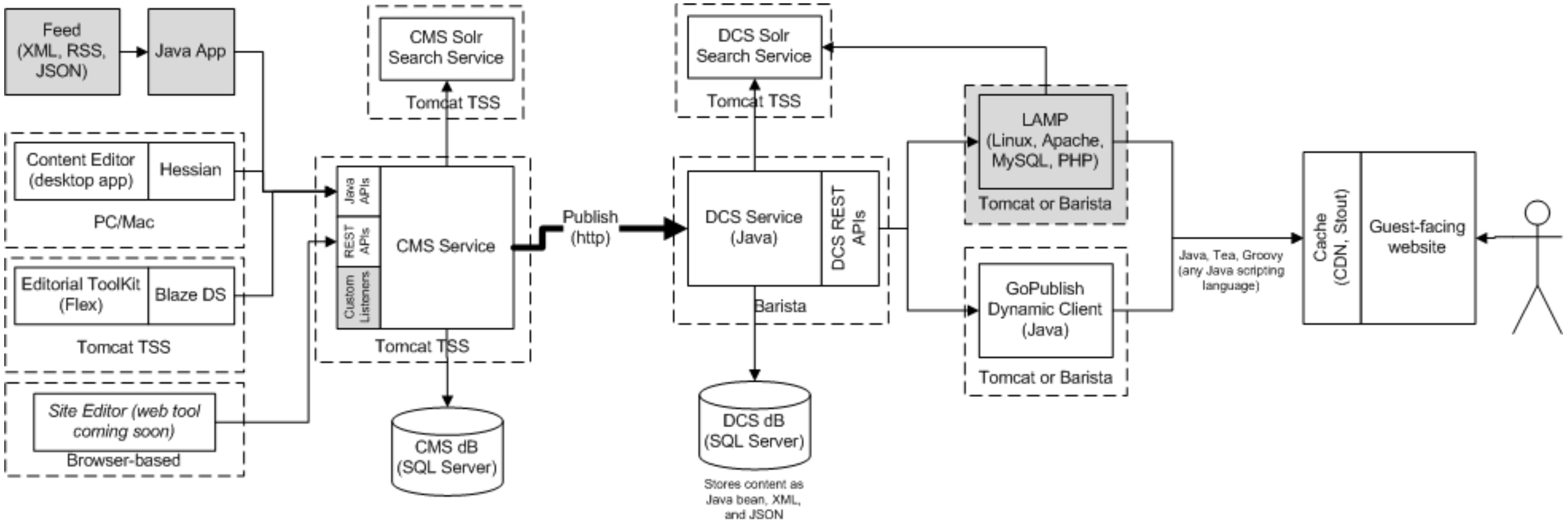
ESPN DIGITAL MEDIA

Deep Dive: Dynamic Content System - DCS

- Dynamic Content Service
  - Collection of Java and REST APIs for accessing published content & content types
  - Content can be grouped by environment (ie QA, UAT, PROD)
  - Java clients get content & type push via serialized beans when content published to the CMS
  - SOLR search service
  - Backend DB stores serialized Java objects or XML
  - Provides access to MILLIONS of content items with less than 50 ms latency (generally)

ESPN DIGITAL MEDIA

ESPN.com / Disney Dynamic Content System

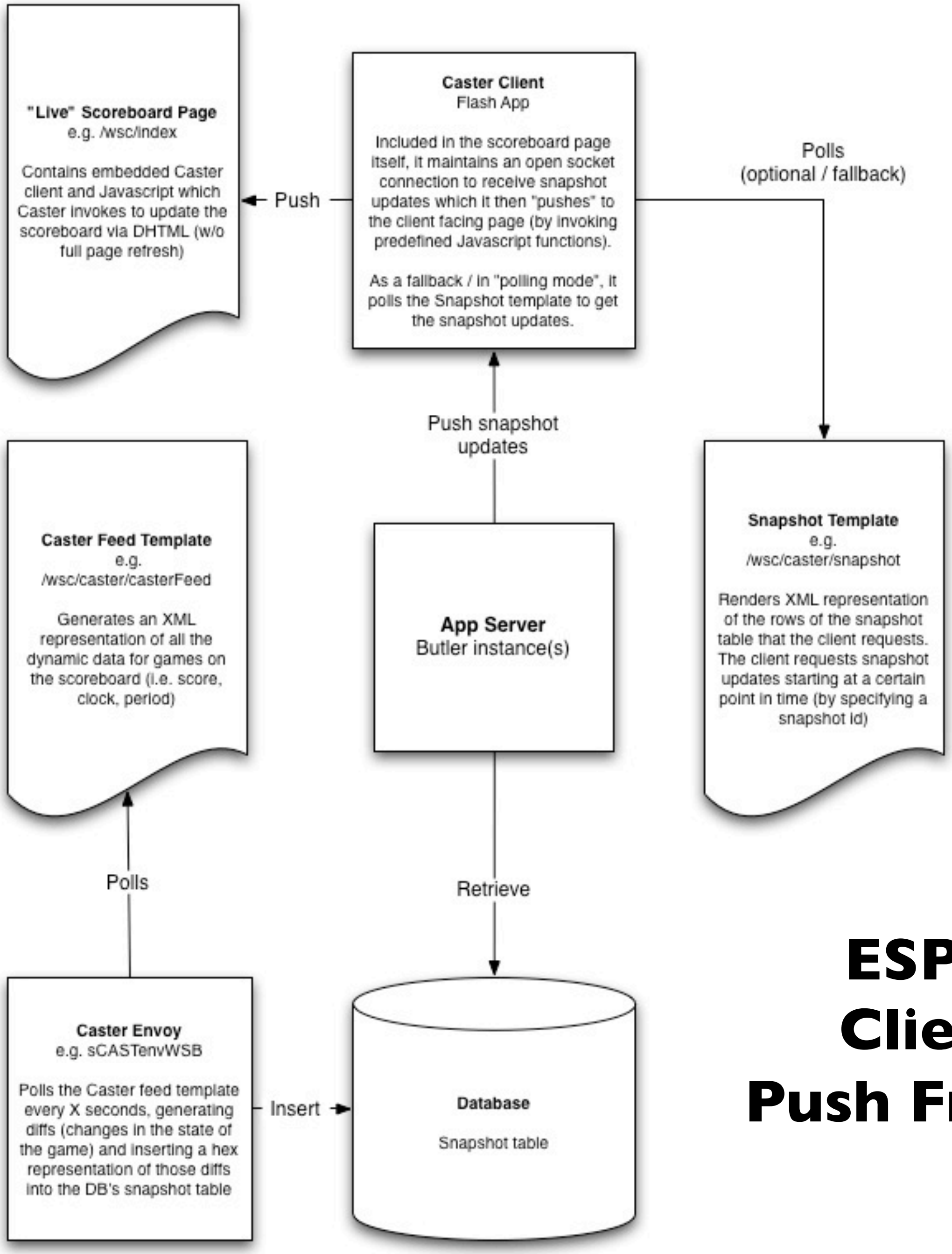


Deep Dive: Live Scores

- Don’t want fan to have to reload for scoreboard to update
- Need client side push of live scores & data
- Websocket before there were websockets
- Three primary components
  - Feed template: XML representation of all dynamic game data
  - Feed Monitor: polls feed template, creates diffs and stores those as a data “snap shot”
  - Caster client: Flash client that keeps connection to server open for push of snap shots. Hands snap shots off to Javascript

ESPN DIGITAL MEDIA

ESPN.com Client Side Push Framework


"Live" Scoreboard Page
e.g. /wsc/index
Contains embedded Caster client and Javascript which Caster invokes to update the scoreboard via DHTML (w/o full page refresh)
Push
Caster Client
Flash App
Included in the scoreboard page itself, it maintains an open socket connection to receive snapshot updates which it then "pushes" to the client facing page (by invoking predefined Javascript functions).
As a fallback / in "polling mode", it polls the Snapshot template to get the snapshot updates.
Polls (optional / fallback)
Push snapshot updates
Caster Feed Template
e.g. /wsc/caster/casterFeed
Generates an XML representation of all the dynamic data for games on the scoreboard (i.e. score, clock, period)
App Server
Butler instance(s)
Snapshot Template
e.g. /wsc/caster/snapshot
Renders XML representation of the rows of the snapshot table that the client requests. The client requests snapshot updates starting at a certain point in time (by specifying a snapshot id)
Polls
Retrieve
Caster Envoy
e.g. sCASTenvWSB
Polls the Caster feed template every X seconds, generating diffs (changes in the state of the game) and inserting a hex representation of those diffs into the DB's snapshot table
Insert
Database
Snapshot table

Deep Dive: Personalization

- Fan desire to have personalized content and presentation
  - And have it follow you everywhere (.com, mobile, TV, etc)
- Need to build scalable, high performance, distribute cache
  - 200 GB of data now
  - Will likely triple over next year
  - Lookup primarily by ID
  - Map-reduce style needs growing
- But site is heavily page cached… not conducive to personalized user experience

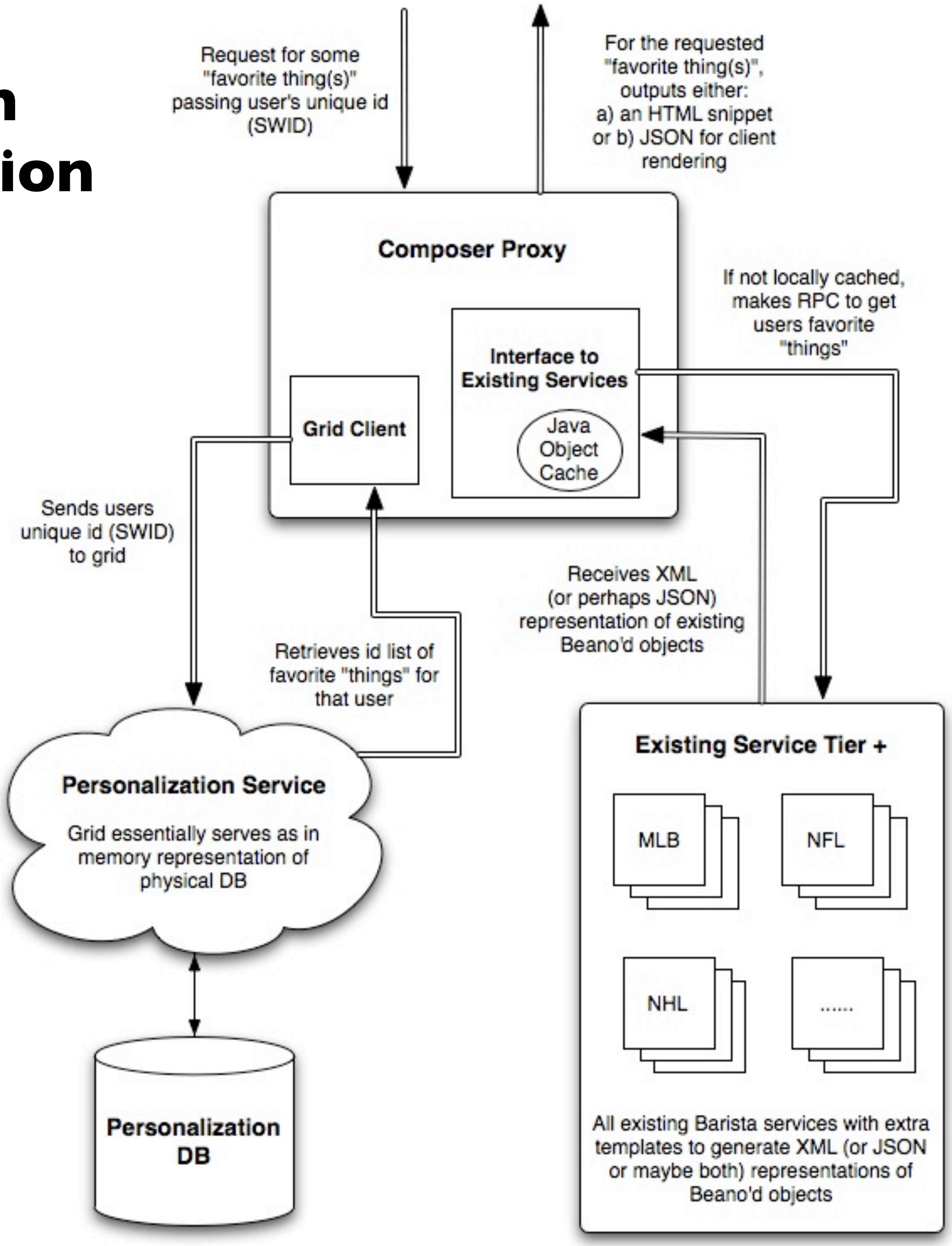
ESPN DIGITAL MEDIA

Deep Dive: Personalization Web Service

- Developer client side framework & high performance, extremely low latency web service
  - Sustained load = 1000s of requests per second per instance
  - Single millisecond latency
  - Used to build your nav bar on front page which gets a TON of traffic
- Lots of GC tuning
  - Set large eden sizes and occupancy fractions

ESPN DIGITAL MEDIA

ESPN.com Personalization Service



Request for some "favorite thing(s)" passing user's unique id (SWID)
For the requested "favorite thing(s)", outputs either: a) an HTML snippet or b) JSON for client rendering
Composer Proxy
If not locally cached, makes RPC to get users favorite "things"
Interface to Existing Services
Java Object Cache
Grid Client
Sends users unique id (SWID) to grid
Receives XML (or perhaps JSON) representation of existing Beano'd objects
Retrieves id list of favorite "things" for that user
Personalization Service
Grid essentially serves as in memory representation of physical DB
Existing Service Tier +
MLB
NFL
NHL
......
All existing Barista services with extra templates to generate XML (or JSON or maybe both) representations of Beano'd objects
Personalization DB

Deep Dive: Personalization Demo

- Changing your ESPN.com navigation

Coming Attractions

- Some other cool projects in the works
  - Client side push framework migration to XMPP
    - Deliver more customizable, personalized, dynamic data feed to fans
  - Evaluating NoSQL solutions for distributed caching solutions
    - We have a DAO framework that decouples us from JPA / Hibernate but interested in Hibernate OGM project other non-relational solutions for Hibernate
  - ESPN APIs Project
    - Provide developer access to ESPN's unparalleled suite of data and content

ESPN DIGITAL MEDIA

Ultimate End Goal

- Get the architecture right to provide best fan expierence
- So we can concentrate on what we do best…

ESPN DIGITAL MEDIA

Help Wanted!

- **We are hiring!!!**
- Go apply at http://espncareers.com … or better yet talk to me afterward

Q&A

- Questions?
- Feedback on anything you heard?
- Suggestions for ESPN.com features?
- Click that little happy face for my rating and I'll fix your fantasy football scores!

ESPN DIGITAL MEDIA