Next Generation APIs Powering Web, Mobile, TV

Manny Pelarinos
Sr. Director of Distribution Platforms

Agenda

QCON 2016

1. Back Story
2. Next Gen API Platform
3. Real time consumer messaging…
   at extreme scale
4. Q&A

About me

- **Manny Pelarinos -** *Sr Director of Distribution Platforms*
- **Second time at QCon**
- **9 year career with ESPN**
- **We're hiring! - jobs.espncareers.com**

Backstory

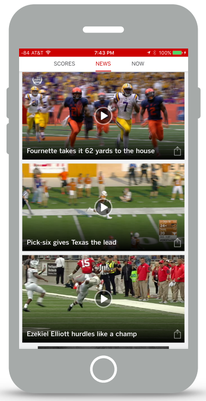

Our Mission
To Serve Sports Fans.
Anytime. Anywhere.
How
• Personalization
• Globalization
• Product-first design
ESPN

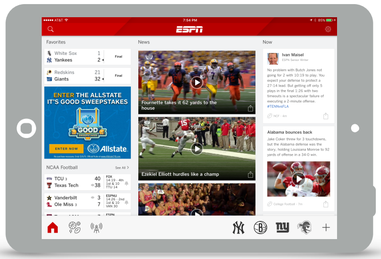
ESPN Facts and Figures

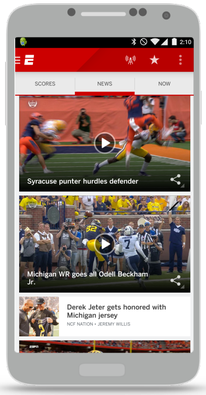
- > 400 Million API calls per day
- Peaks of over 20K RPS
- 1 Million WS Updates to Fans in < 100MS

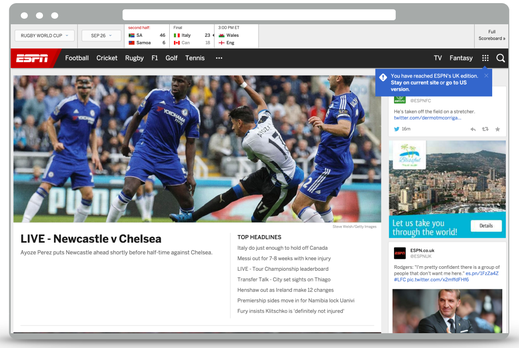
Product Galore

So Many Screens





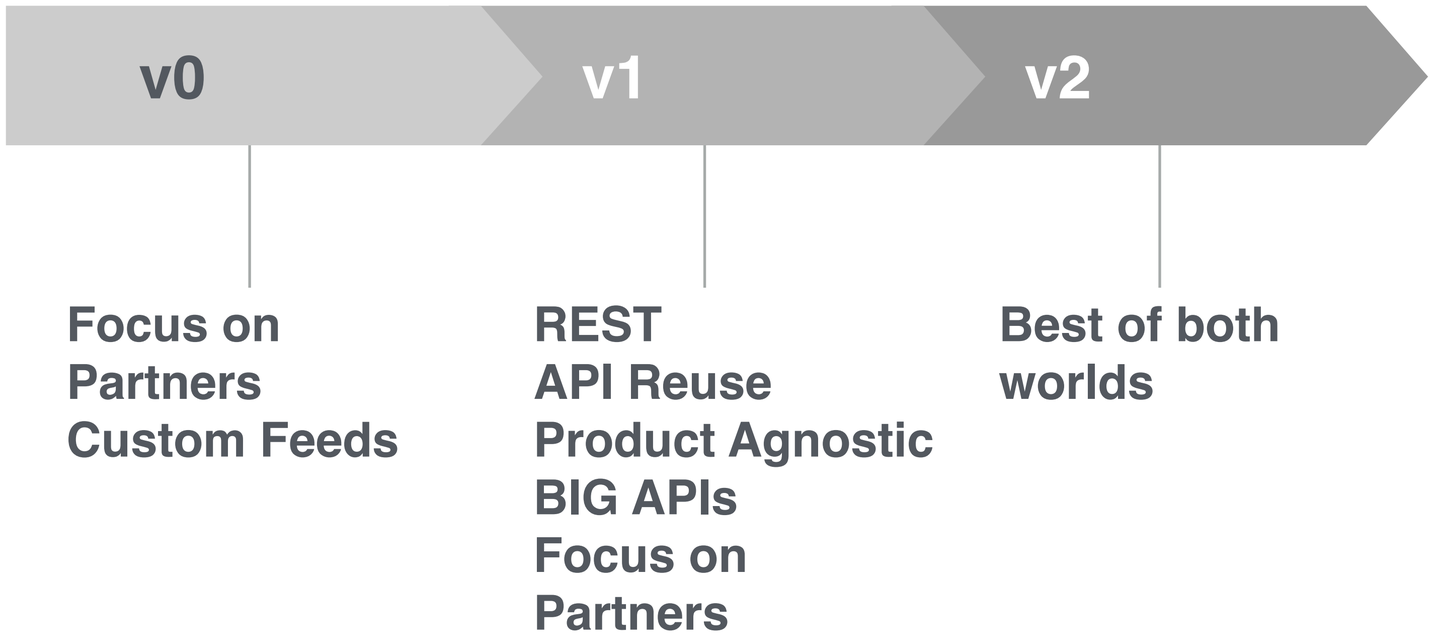




The Good, The Bad
and Mobile

- Network latency and multiple requests are bad
- Data plan restrictions and bandwidth

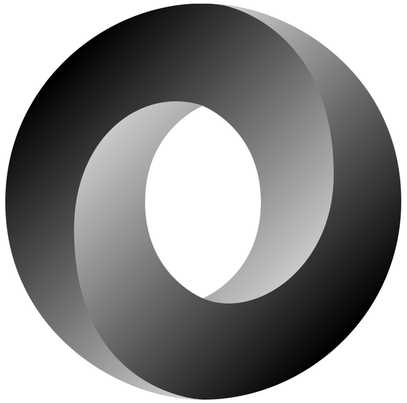
API Evolution



Next Generation API Platform

Say Hello to Binder

Core APIs



Product APIs

Architecture

Products & Screens

Core APIs
- Business logic
- Service tier
- Lightweight
- Data
- References

Product APIs
- Composition
- Trimming
- Custom DSL
- Efficient

```
/**
 * Portrays a simple way of aggregating data.
 *
 * @title Simple Example
 * @timeToLive 60
 * @options timeout 1000
 */
define '/ignore' { model, request, response ->
    get("${CORE_HOST}/v2/sports/baseball/leagues/mlb/events") {
        root 'items'
        expand('*') {
            exclude '$ref'
            exclude '*..$ref'
        }
    }.then { events ->
        model.events = events
    }
}
```

```
{
  - events: {
        id: "360609113",
        uid: "s:1~l:10~e:360609113",
        date: "2016-06-09T18:05Z",
        name: "Houston Astros at Texas Rangers",
        shortName: "HOU @ TEX",
        timeValid: true,
      - competitions: [
          - {
              id: "360609113",
              uid: "s:1~l:10~e:360609113~c:360609113",
              date: "2016-06-09T18:05Z",
              attendance: 30145,
              necessary: false,
              timeValid: true,
              neutralSite: false,
              divisionCompetition: false,
              conferenceCompetition: false,
              previewAvailable: false,
              recapAvailable: true,
              boxscoreAvailable: true,
              lineupAvailable: false,
              gamecastAvailable: true,
              playByPlayAvailable: true,
              conversationAvailable: true,
              commentaryAvailable: false,
              pickcenterAvailable: true,
              summaryAvailable: true,
              liveAvailable: false,
              ticketsAvailable: false,
              shotChartAvailable: false,
              timeoutsAvailable: false,
              possessionArrowAvailable: false,
              onWatchESPN: false,
              recent: false,
              wasSuspended: false,
            + boxscoreSource: {…},
            + playByPlaySource: {…},
            + venue: {…},
            + competitors: […],
              notes: [ ],
              situation: { },
              status: { },
              odds: { },
              broadcasts: { },
            + series: […],
              officials: { },
              details: { },
              leaders: { },
            + links: […]
            }
        ],
      + links: […]
      }
}
```

```
/**
 * Portrays a simple way of aggregating data.
 *
 * @title Simple Example
 * @timeToLive 60
 * @options timeout 1000
 */
define '/ignore' { model, request, response ->
    get("${CORE_HOST}/v2/sports/baseball/leagues/mlb/events") {
        root 'items'
        expand('*') {
            include 'id', 'name', 'date'
        }
    }.then { events ->
        model.events = events
    }
}
```

```
{
  - events: [
      - {
            id: "360609113",
            date: "2016-06-09T18:05Z",
            name: "Houston Astros at Texas Rangers"
        },
      - {
            id: "360609127",
            date: "2016-06-09T21:10Z",
            name: "Pittsburgh Pirates at Colorado Rockies"
        },
      - {
            id: "360609110",
            date: "2016-06-09T23:05Z",
            name: "Los Angeles Angels at New York Yankees"
        },
      - {
            id: "360609114",
            date: "2016-06-09T23:07Z",
            name: "Baltimore Orioles at Toronto Blue Jays"
        }
    ]
}
```

Personalization

- Actual Fan preference data is small - IDs
- Content is big - News, Scores, Etc.
- Solution - Product API that conflates Fan Preferences with sports data

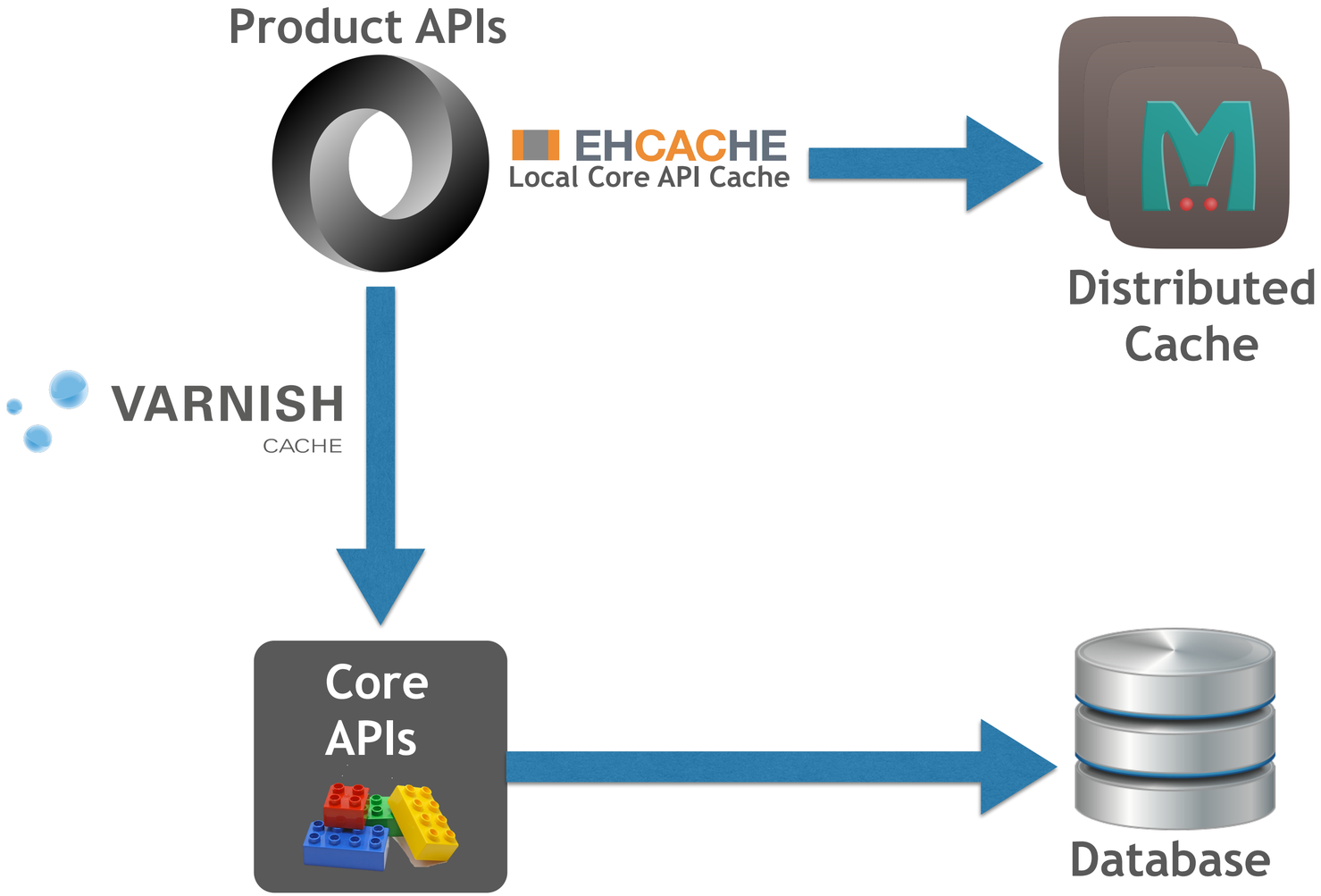
ESPN Now

- Many disparate sources of content - CMS, Blogs, Twitter, Facebook, etc.
- Search API alone doesn’t get you all the details
- Solution - Product API that conflates search and all our various content APIs

Key Components

- Caching
- Asynchronous
- DSL + Groovy
- Tools & Dashboards

Caching


Product APIs
EHCACHE
Local Core API Cache
Distributed Cache
VARNISH
CACHE
Core APIs
Database

Promises

- Make requests in parallel – Product API should only be as slow as its slowest core API call
- But, mixing synchronous programming with asynchronous is hard
- Enforce asynchronous all the time but needed convenience and rails
- Ported JS Promises to Java (With RxJava)

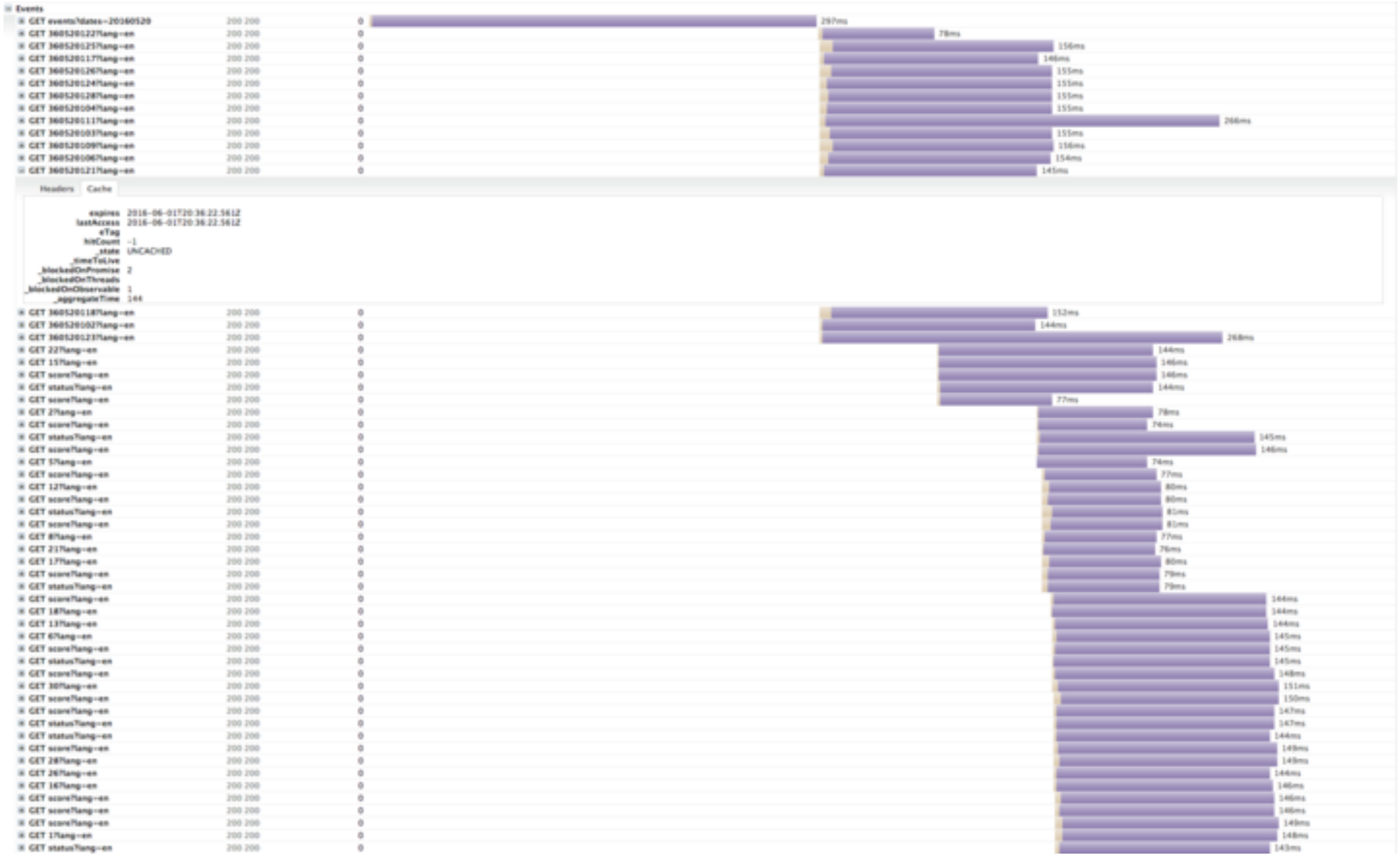
Groovy

- Powerful Dynamic Language
- Java++
- Performance issues mitigated by caching

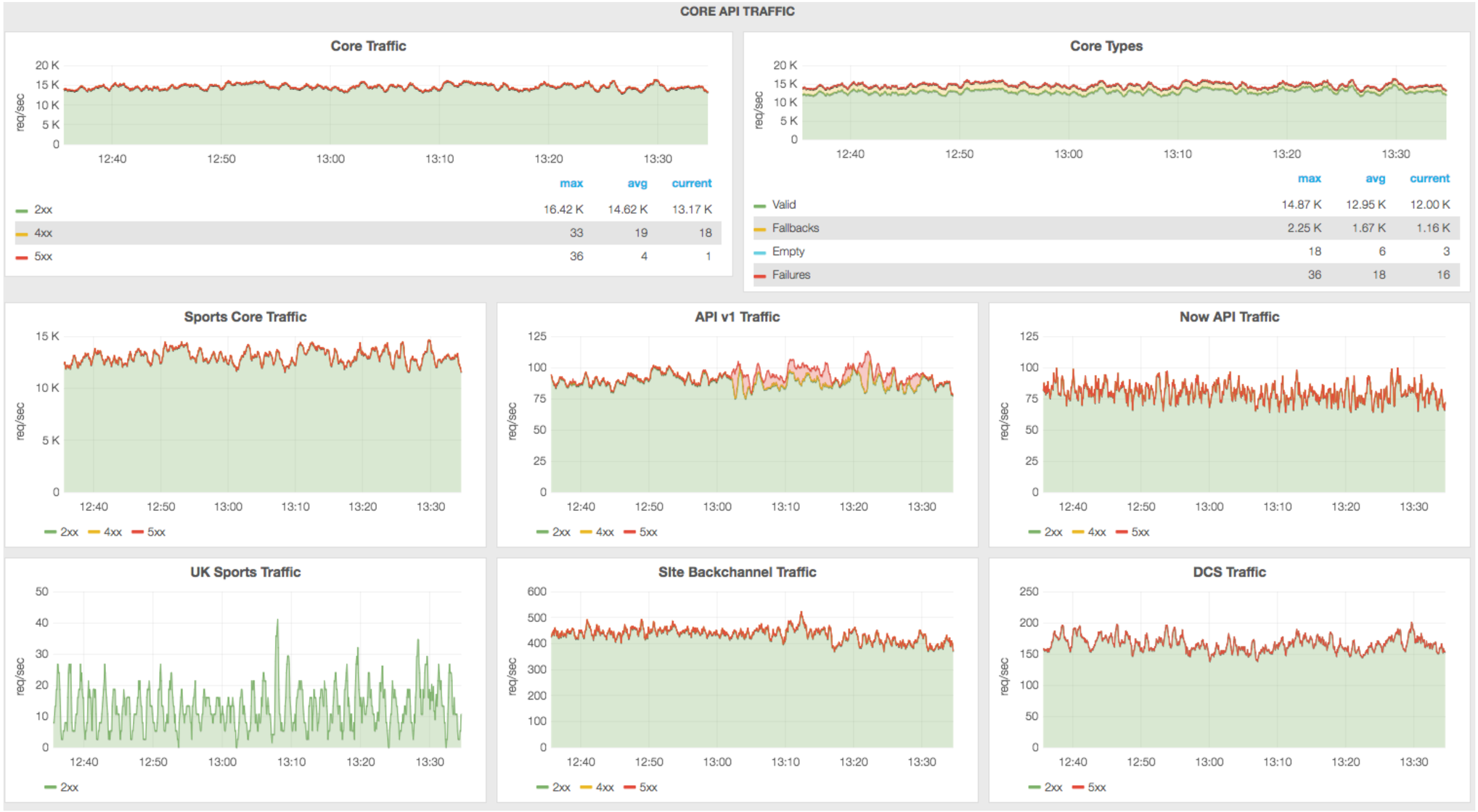
Tools and Dashboards - Trace


```
  duration: 332,
  timeToLive: 65253,
  timeToLiveRemaining: 64918,
- statistics: {
    count: 91,
    cached: 0,
    stale: 0,
    misses: 91,
    forced: 0,
    errors: 0
  },
- transactions: [
  - {
      uri: "http://{{CORE_HOST}}/v2/sports/baseball/leagues/mlb/events/360520102/competitions/360520102/status?lang=en",
      duration: 145,
      state: "CACHE_MISS",
      timeToLive: 162000,
      timestamp: "2016-06-01T20:36:57Z",
      requestDate: "2016-06-01T20:36:57Z",
      statusCode: 200,
      errorCode: 0
    },
  - {
      uri: "http://{{CORE_HOST}}/v2/sports/baseball/leagues/mlb/events/360520102/competitions/360520102/competitors/5/score?
      lang=en",
      duration: 142,
      state: "CACHE_MISS",
      timeToLive: 162000,
      timestamp: "2016-06-01T20:36:57Z",
      requestDate: "2016-06-01T20:36:57Z",
      statusCode: 200,
      errorCode: 0
    },
  - {
      uri: "http://{{CORE_HOST}}/v2/sports/baseball/leagues/mlb/seasons/2016/teams/5?lang=en",
      duration: 141,
      state: "CACHE_MISS",
      timeToLive: 316000,
      timestamp: "2016-06-01T20:36:57Z",
```

HAR (HTTP Archive)



Grafana and OpenTSDB



Technology


VARNISH
CACHE


swagger


EHCACHE


spring
by Pivotal


Groovy


Java


Jersey


HYSTRIX
DEFEND YOUR APP


{JSON}

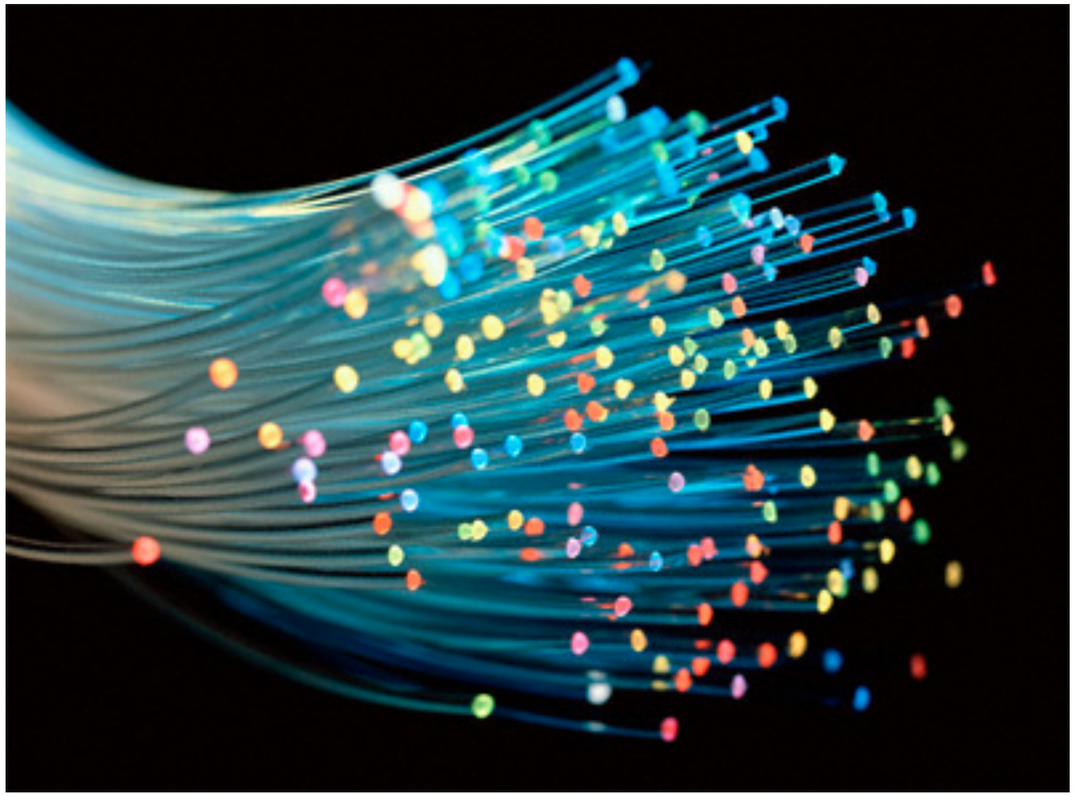
ESPN TECHNOLOGY

Real Time Messaging

Real-Time Data at Sports Scale

Massive Scale

- How do we enable real time data with REST APIs?
- How can we scale to millions of concurrently connected Fans?



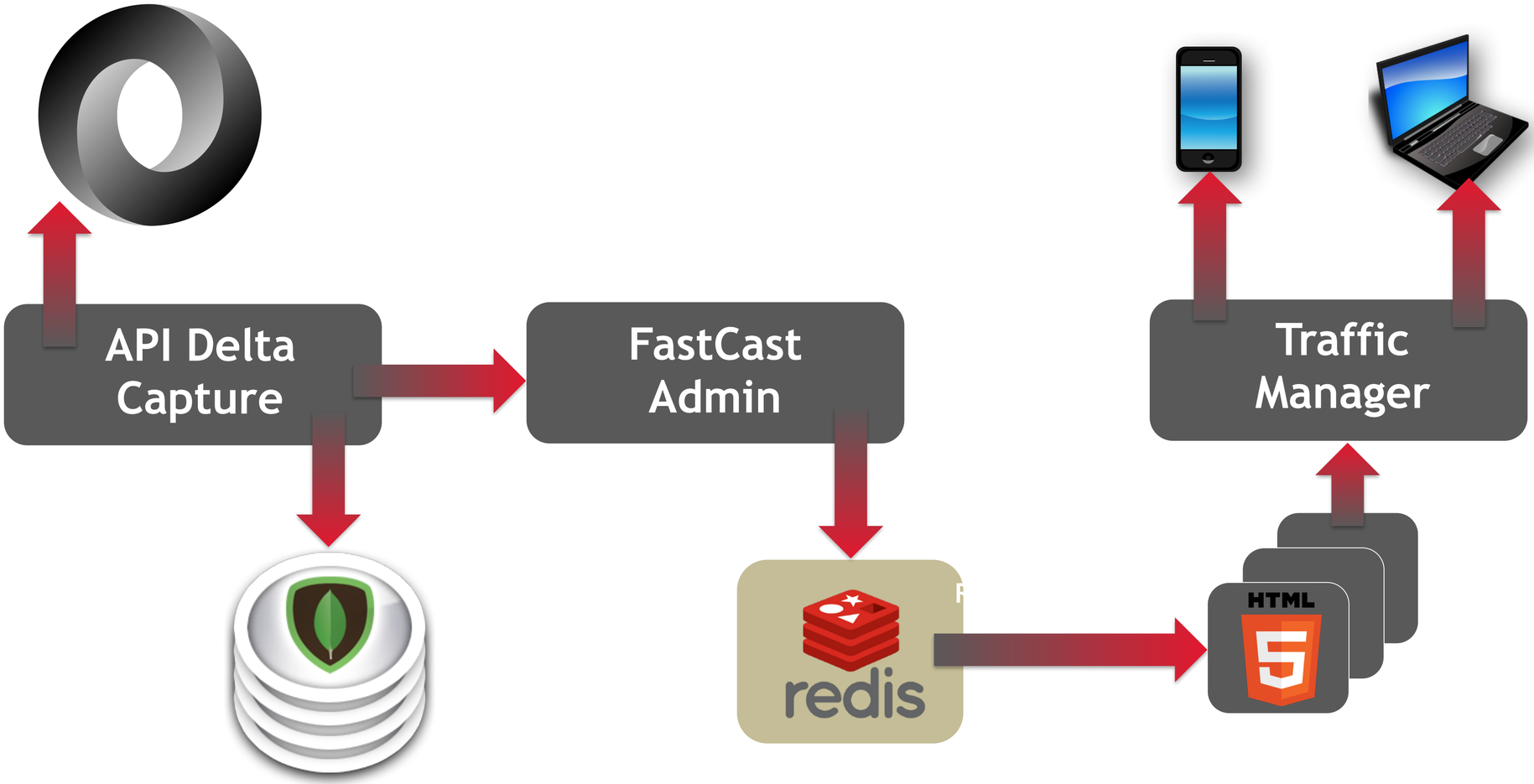
Say Hello to FastCast

- FastCast is a real-time high-scale multicast publishing platform that uses web sockets as its underlying technology, allowing it to work in both web and mobile spaces
- Client sets up a “Topic” via Admin Service
- Client publishes to Topic via Admin Service
- Fans subscribe to Topic via FastCast Core
- Fans receive messages as they are published

The Details

- Reliable core node replication handled thru Redis
- Built-in checkpoint/snapshot feature
- Uses generic JsonDiff/Patch + payload compression
- Client/server healthCheck + Better analytics captures
- AutoScaling in AWS with multi-region latency based routing

FastCast Architecture



Summary

What Did We Learn?

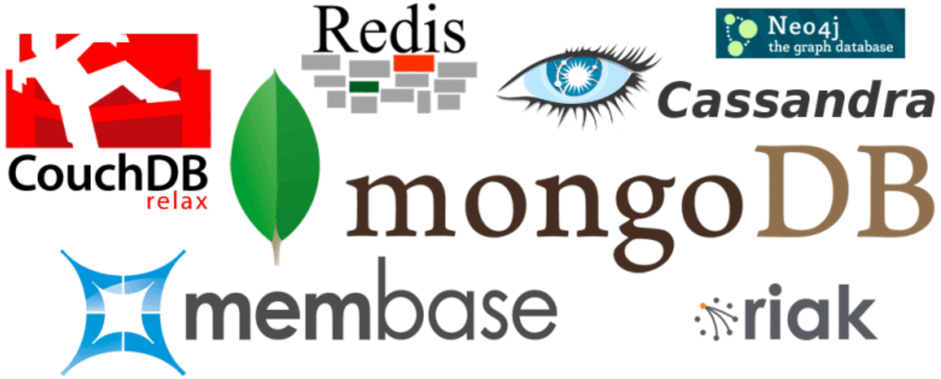
- One size does NOT fit all - Need to separate Core from Product
- Caching is Key
- Operational Tools are awesome
- REST APIs + WebSockets = COOL

Playbook for Errors

- Error handling is complicated…
- What to do when circuit breaker trips and nothing is in cache
- 50x vs. NULL

NoSQL = ~~Joins~~

- Binder as an in-memory application side ‘join engine’
- On going discussion around pre-expanding more



ESPN TECHNOLOGY