Building Modern Web Applications

Evan Roth
Affinity Director
Software Architecture
Productive Edge
Productive Edge
At-a-Glance

2008
Founded as a software engineering firm focused on Web application development with agile processes and open source technologies

“Enterprise Boutique”
Built on quality and personal touch technologies

150+ resources

45% onshore
55% offshore

80% employees
20% contractors

2013/2014

Onshore
45%

Offshore
55%

Employees
80%

Contractors
20%

Service offerings now span:
- WEB
- MOBILE
- DIGITAL
- MANAGED

Office in 4 countries
Headquarters in Chicago
What we do

We provide end-to-end, full stack, high quality web and mobile solutions.

end-to-end solutions
strategy, visual design, user experience, business analysis, development, testing, analytics, support

full stack high quality
front end, back end, web, mobile, integration, data

user validated, thoroughly tested
Brief History of the Evolution of Apps

- Pre 1990’s
  - Shared systems
  - Console-based Applications
  - In-house infrastructure
  - Birth of the internet

1990’s
Brief History of the Evolution of Apps

Client-Server Apps

CSS allows for quick improvements beyond simple text

Enterprise embraces frameworks: Java & Microsoft’s ASP
Brief History of the Evolution of Apps

- **Pre 1990’s**
  - PHP released—now 80% of web sites run it
  - Co-location facilities

- **1990’s**
  - AJAX allows for building rich web applications
  - Browsers become the de facto client

- **2000’s**
  - Bring on the Javascript frameworks
  - Browsing on phones becomes commonplace

- **NOW**
Modern development requires modern processes.

Focus on performance and quality.

Applications require a mixed skillset.
Development Processes

**Waterfall**
- Requirements up front
- Rigid budget and timeline
- Project phases for Development and testing

**Agile**
- Embrace change
- Iterative development
- Continuous delivery
Hybrid – Practical Agile

Productive Edge’s model for working with our clients

- Early Discovery phase to understand requirements
- Early technical spikes to mitigate risk

Established deadline and budget
Iterative development
Continuous delivery
Continuous Delivery

Working towards:
- Reacting faster
- Reducing risk & cost
- Flexible releases

Separate environments
- Test
- Staging
- Production

Automated builds
- Test builds triggered by code check-ins
- Contain automated quality gates
  - Build success
  - Unit tests Code that tests other code
  - Other rules available
Cloud Services

Infrastructure-as-a-Service
- Servers
- Databases
- Caches
- Specific services
  - Notifications such as Email or mobile messaging
  - Message Queueing

Usable On-Demand
- Scriptable, burstable configuration
- Redundancy
- Geographical shortest hop
- Leading providers
  - Amazon Web Services
  - Microsoft Azure
  - OpenStack - RackSpace / NASA project open-sourced with wide support
Anatomy of a Simple Application

Server “Back End”
- Computation
- Persistence

Browser “Front End”
- Rendering
- Interaction

Diagram:
- Server
  - Application Framework
  - Database
- Browser
  - CSS
  - HTML
  - Javascript
Common Back-End Frameworks

Java
Runs anywhere, but typically Linux

Microsoft .Net
Windows-based
ASP.NET MVC widely popular

PHP
Commonly with the LAMP stack (Linux, Apache, MySQL & PHP)
All open source
Huge adaptation
Service-based Architecture

Applications become a mash-up of multiple services
• Enable multiple clients, whether web or mobile

Microservice
• Independently deployable applications
• Focus on specific domain

Services focus on RESTful design
• Representational State Transfer the architecture of the web
• Semantic use of endpoints
Client

GET /authors/Wilczek

200 OK
{
  count: 48,
  publications: [
    { id:1, title:"Superheavy Light Quarks and the Strong P, T Problem" },
    { id:2, title:"Oscillatory Attractors: A New Cosmological Phase" },
    ...
    { id:48, title:"Asymptotically Free Gauge Theories. 1” } ]
}

High-Energy Physics Publications Service
What’s a Database Today?

Relational most common understanding, row oriented
• Ex: MySql or SQL Server

“NoSQL” various non-relational databases that solve certain problems well
• Document – most common NoSQL, store objects as documents [MongoDB]
• Column – highly efficient aggregation of data (ex: avg height of users) [Cassandra]

• Graph – Nodes with relationships [Neo4J]
  • Solves problem queries like “Show me pizza restaurants in Chicago that my friends like”
• Key-Value store – Highly efficient retrieval of objects by key [memcached]
Search

Providing near real-time results

Solve problems where databases struggle:
  • Geospatial searching
  • Faceting
  • Spellcheck

Lucene
  • Java-based indexing of documents

Apache Solr & Elasticsearch
  • Lucene-based search engines
  • Offer updated integration and ease of use
  • Cutting-edge scalability
  • Elasticsearch completely controlled by modern service
Javascript: Language of the Web

jQuery
- Cross-platform library for manipulating a web page
- Most popular Javascript library by a wide margin
- Included in many frameworks and platforms

Backbone.js
- Lightweight templating engine for binding data models to HTML

AngularJS
- Web application framework for single page applications
- Enables two-way data-binding between HTML and back-end services

React
- Latest framework gaining popularity
- Renderings beyond HTML allow for HTML5 elements and
Simplifying Design Execution

Frameworks for streamlining HTML, CSS and JS components

Most popular: Bootstrap & Foundation

Responsive
Grid system
Stylized, themeable
CSS components
Optional Javascript extensions
Ex: Pop-up modal or fixed sidebar
Javascript from Front to Back

Node.js
Runtime environment for writing Javascript applications
Fast execution using Google V8 engine
Enables full-stack development in Javascript
Built in package manager

MEAN stack
Trending Javascript-based development stack
Compare to LAMP stack

Client

Server

Database
Caching

Client-side cache
- Browser does not request resource from server
- Commonly used for static resources

Server-side cache
- Web server caches outgoing requests
- Does not re-process upon further request

Distributed application caches
- Distributed cache consists of multiple nodes across infrastructure
- Eliminate redundant calls to long running processes
  - Database queries
  - Intense calculations
- Typically NoSql Key-Value store
- Various patterns for loading data into the cache
The Result

SQL → Service One .NET
Oracle → Service Two Java
MongoDB → Service Three .Net

HTML | CSS
---|---
Angular JS