Webszolgáltatások implementációja

Ráth István

rath@mit.bme.hu

Tartalom

- Alaptechnológiák
  - Java Enterprise bevezető
- Futtatókörnyezet
  - Nyilvános cloud: Google App Engine
  - Privát cloud: Eclipse Gyrex
- Fejlesztőeszköz
  - Eclipse Web Tools Platform
- Konkrét technológiák
  - Java RESTful web services: JAXRS
  - Java SOAP web services: JAXWS
  - Java XML Bindings: JAXB

Források

- JAXWS, JAXB
  - http://www.slideshare.net/kverbert/soap-tutorial
- JAXRS
  - http://www.slideshare.net/kverbert/using-java-to-implement-rest-web-services-jaxrs
- Eclipse WTP
  - http://www.slideserve.com/fox/eclipse-wtp-web-service-tools
- Google App Engine
  - http://www.slideshare.net/ikailan/intro-to-app-engine-agency-dev-day-nyc-2011
- Eclipse Gyrex
  - http://www.eclipsecon.org/europe2012/sessions/developing-cloud-applications-eclipse-gyrex

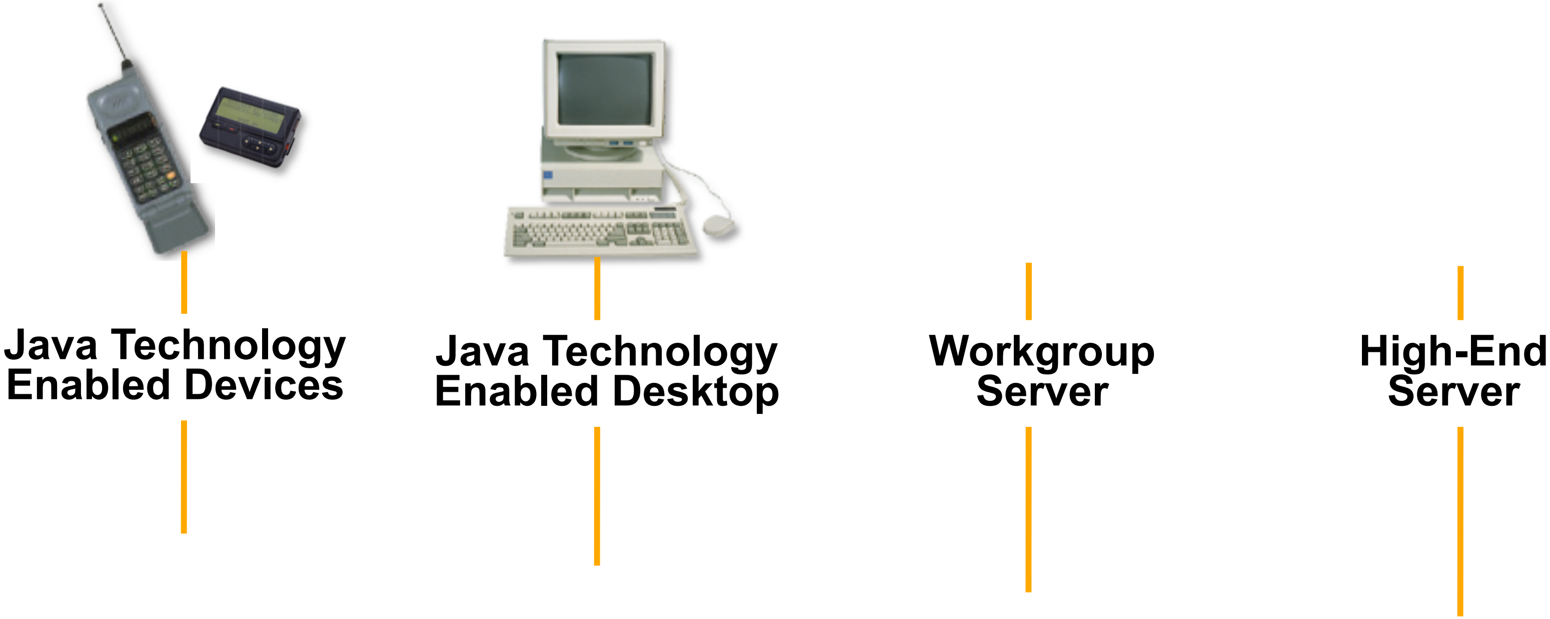
JEE Overview

What is JEE ?

- A set of technologies for developing enterprise applications in Java
- Specified by Sun and the Java Community Process (JCP).
- Implemented by JEE vendors.
- Implementations of JEE technologies are provided within Application Servers.
- Previously named J2EE (until version 1.4) current version is JEE 5.

What is JEE ?

- The Java Platform



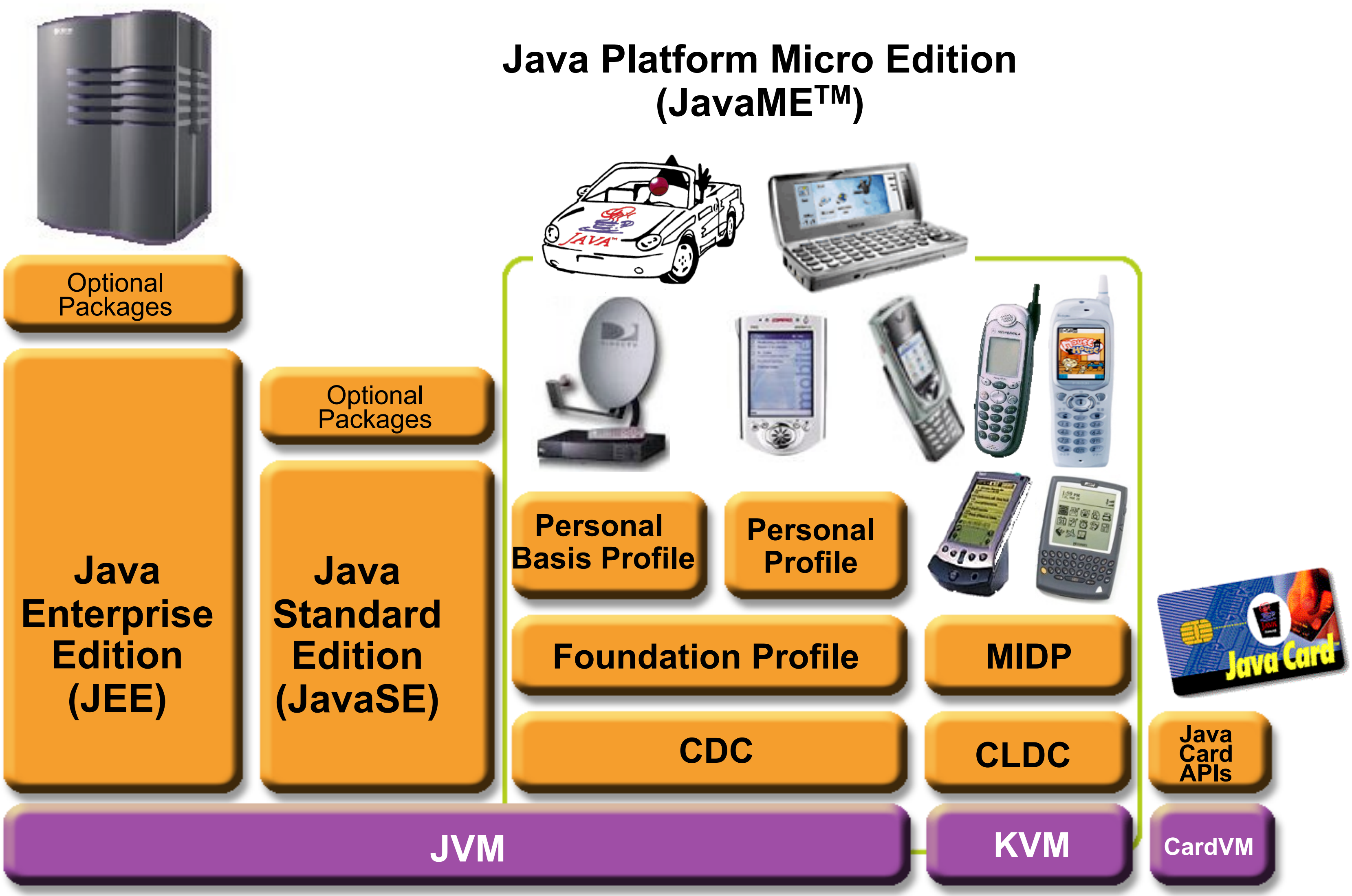
Java Technology Enabled Devices

Java Technology Enabled Desktop

Workgroup Server

High-End Server

The Java Platform



Java Platform Micro Edition (JavaME^{TM})

Optional Packages

Java Enterprise Edition (JEE)

Optional Packages

Java Standard Edition (JavaSE)

Personal Basis Profile

Personal Profile

Foundation Profile

MIDP

CDC

CLDC

Java Card APIs

JVM

KVM

CardVM

* Under development in JCP

Why do we need JEE ?

- Distribution
- Transactions
- Security
- Scalability
- Persistence

Value to developers.

- Can use ***any JEE implementation*** for development and deployment
  - Use production-quality standard implementation which is free for development/deployment
  - Use high-end commercial JEE products for scalability and fault-tolerance
- Vast amount of JEE ***community resources***
  - Many JEE related books, articles, tutorials, quality code you can use, best practice guidelines, design patterns etc.
- Can use off-the-shelf 3rd-party business components

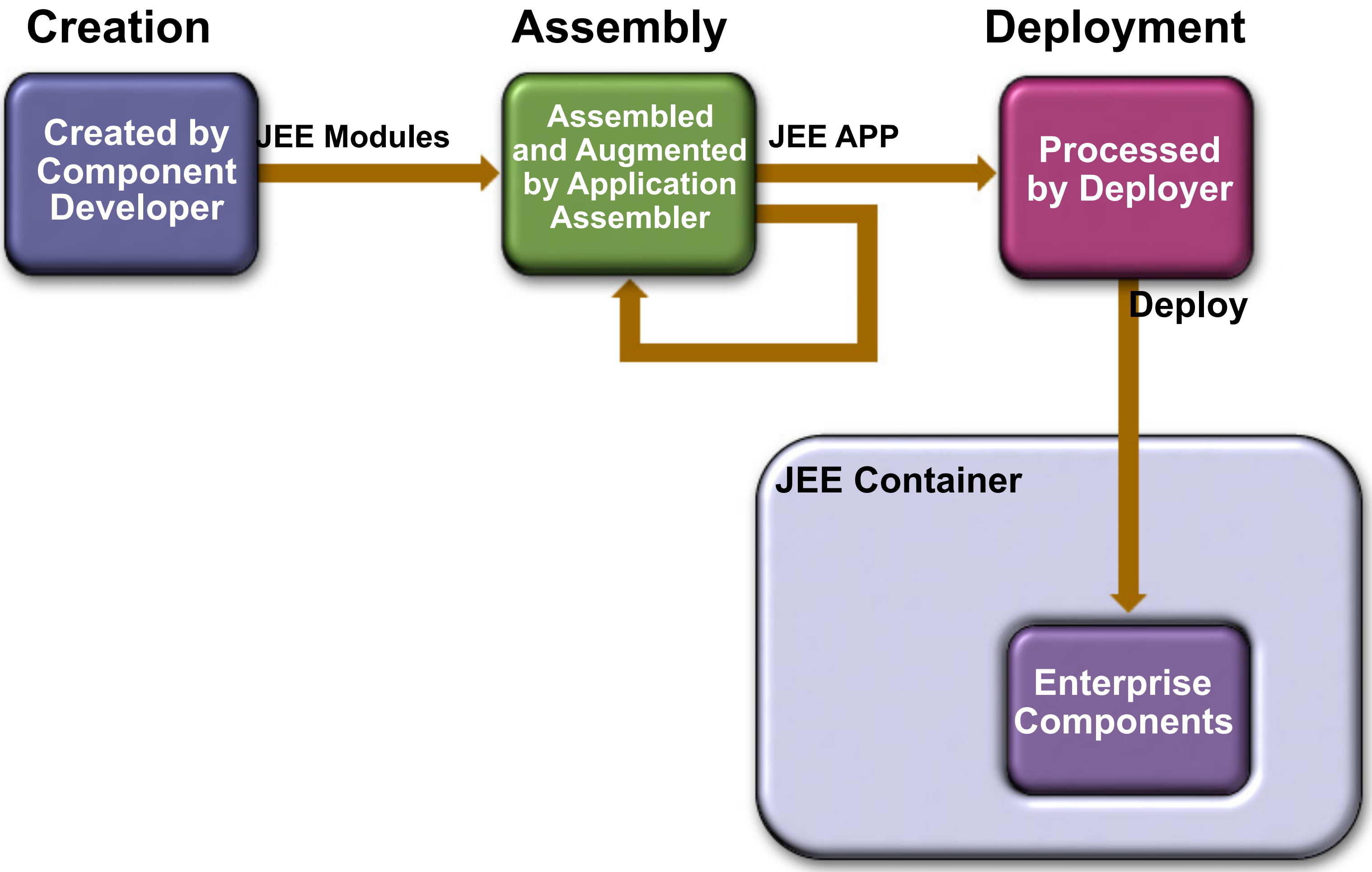
Value to vendors

- Vendors work together on specifications and then compete in implementations
  - In the areas of Scalability, Performance, Reliability, Availability, Management and development tools, and so on
- Freedom to innovate while maintaining the portability of applications
- ***Do not have create/maintain their own proprietary APIs***

JEE Development Roles

- Component provider
    - Bean provider
- Application assembler
- Deployer
- Platform provider
    - Container provider
- Tools provider
- System administrator

The JEE Life Cycle



The JEE Architecture

The JEE Architecture

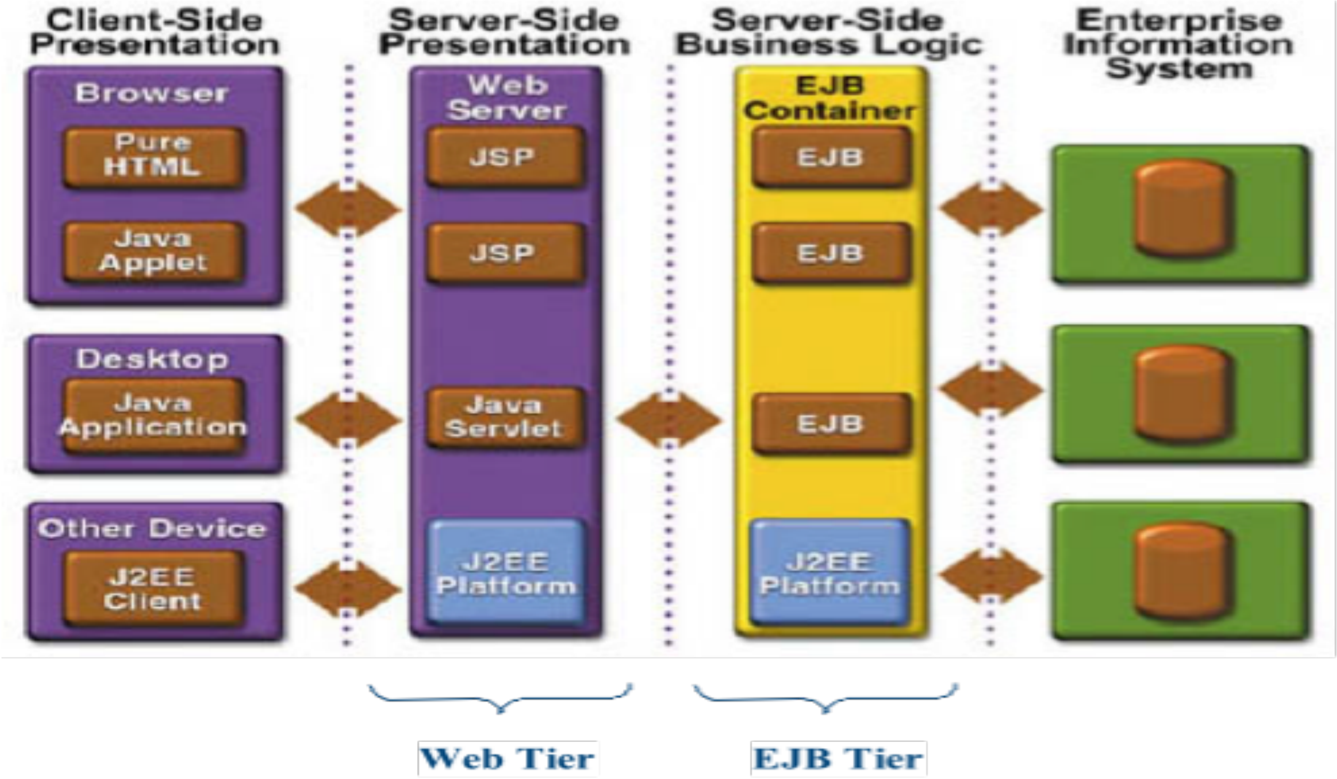
- N-tier architecture
- Comprised of technologies for the business tier the presentation tier and other system services.
- Runs within the application server and within specific containers (web container, EJB container) within the Application server.

The JEE Architecture

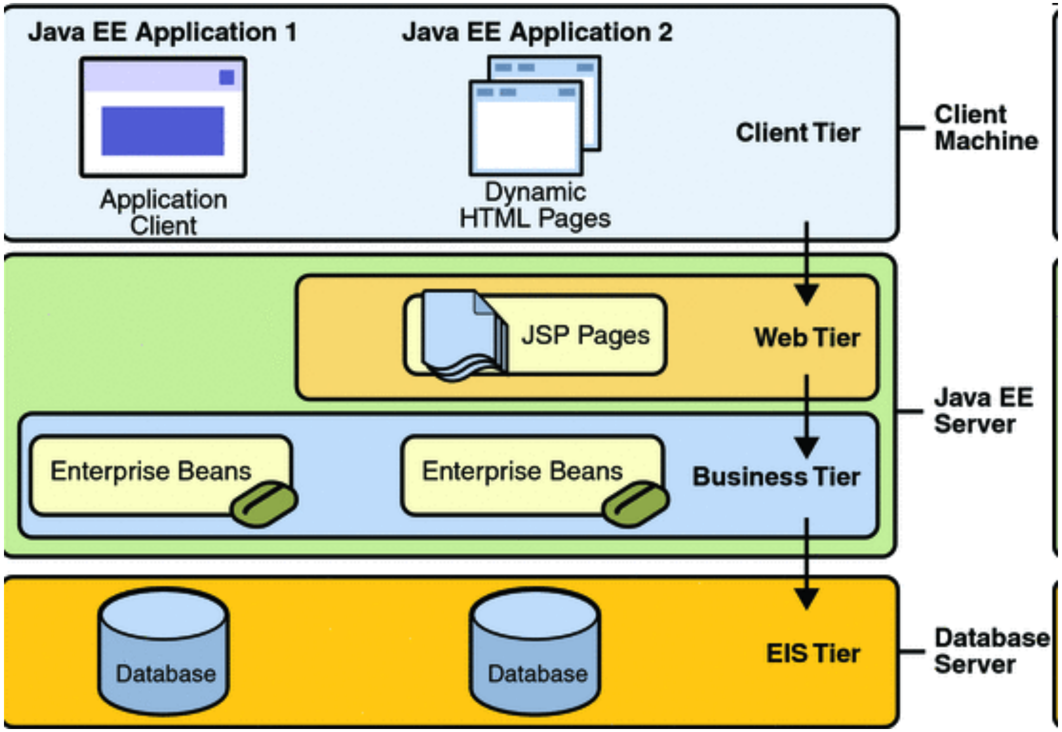
- Uses the "component and container" model in which container provides system services in a well-defined and as industry standard

- JEE is a standard that also provides portability of code because it is based on Java technology and standard-based Java programming APIs

Three-Tier architecture



JEE Tier Architecture

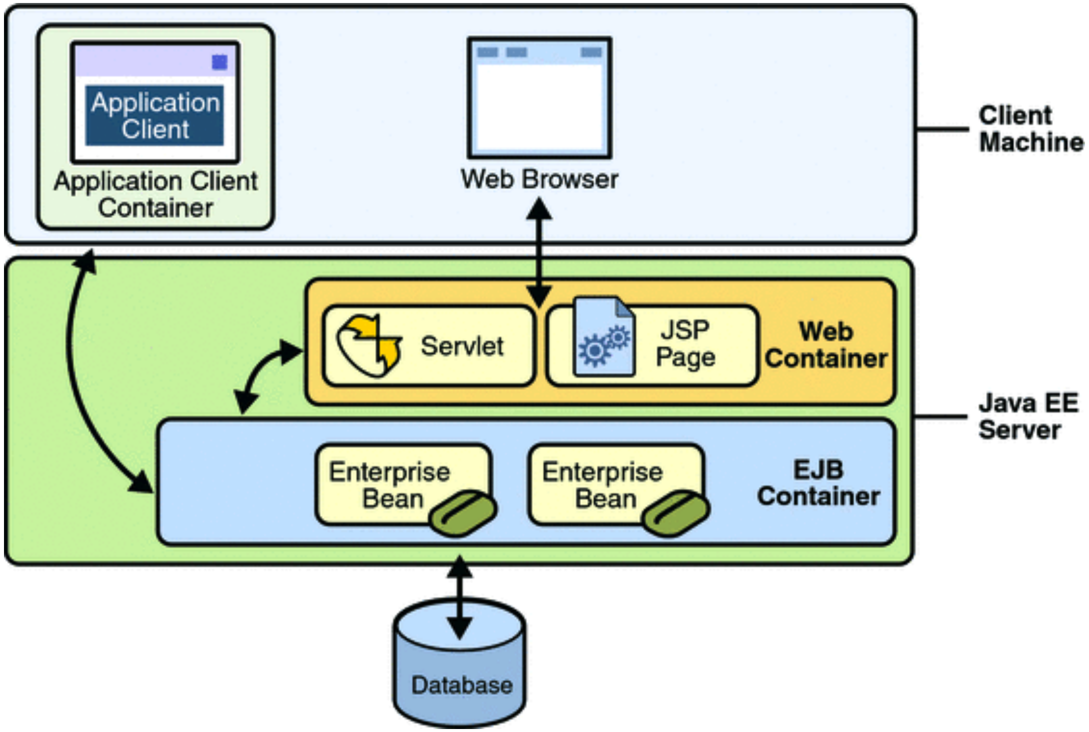


JEE Application Servers

JEE Application Servers

- JEE vendors provide their implementation of JEE technologies within an Application Server.
- Each application server has its own implementation of JEE standards as well as some proprietary features.
- Comprised of a Web Container,EJB Container and other server services.

The App server and JEE containers.



References

- Oded Nissan: JEE Overview
  - http://www.slideshare.net/odedns/jee-course-jee-overview
- Imre Gábor: Enterprise Java Beans
  - (UML bázisú modellezés és analízis tárgy anyagai, 2008-2010)

Developing Cloud Applications with Eclipse Gyrex

Gunnar Wagenknecht, @guw

Modern Server Applications

- **High traffic**

Million transactions per hour

- **Different** frontend **technologies** and **devices**

- **Modular** in development and deployment
- **Easy** to setup and run
- **Open** for new technologies
    - e.g. persistence

Eclipse Gyrex

A lightweight **application stack** for building server applications using **EclipseRT** technologies.

EclipseRT (RT = Runtime)

“EclipseRT is the collection of OSGi-based runtimes and frameworks built by the Eclipse open source projects. “

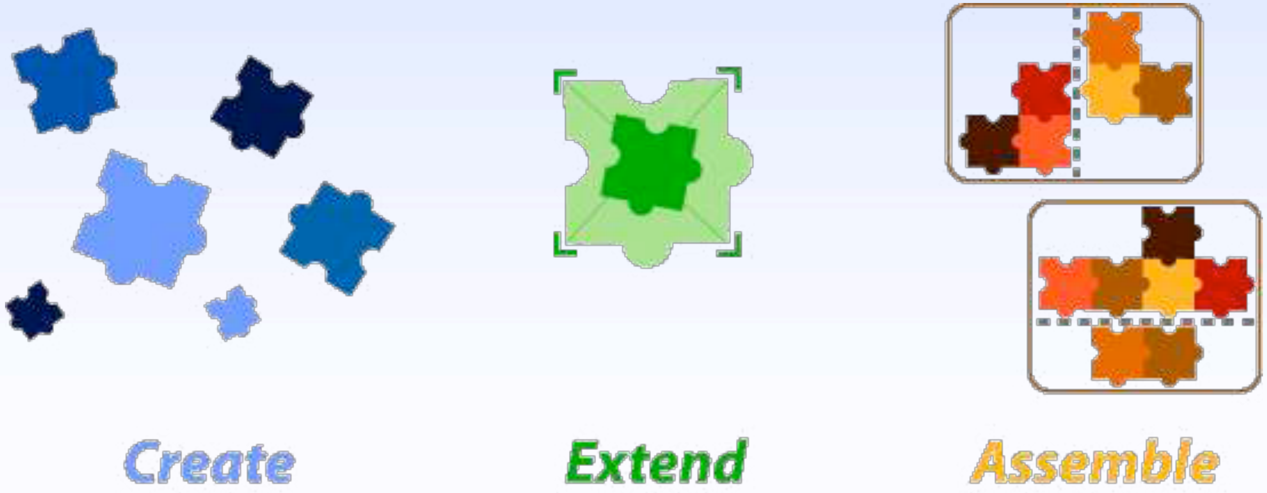
Containers, Middleware, EnterpriseFrameworks

Equinox

- **OSGi** reference implementation

- Foundation of EclipseRT

- **C**omponent **O**riented **D**evelopment and **A**ssembly



Jetty

- Asynchronous HTTP Server and Client
- Standards based Servlet Container
- Web Sockets server
- OSGi, JNDI, JMX, JASPI, AJP support

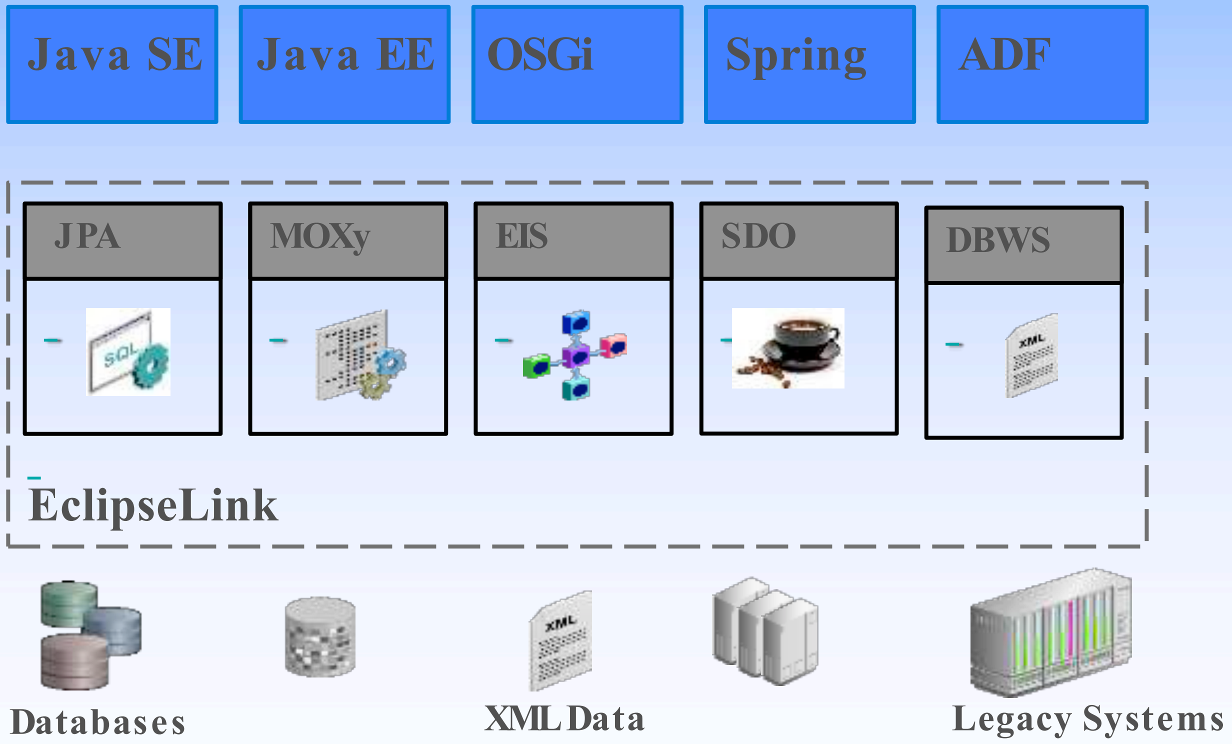
- Small foot print
- Excellent scalability

jetty://

- Runs in
    - Apache Hadoop
    - Google AppEngine

EclipseLink

Comprehensive Java persistence solution addressing relational, XML, and database web services.



Java SE

Java EE

OSGi

Spring

ADF

JPA

MOXy

EIS

SDO

DBWS

EclipseLink

Databases

XML Data

Legacy Systems

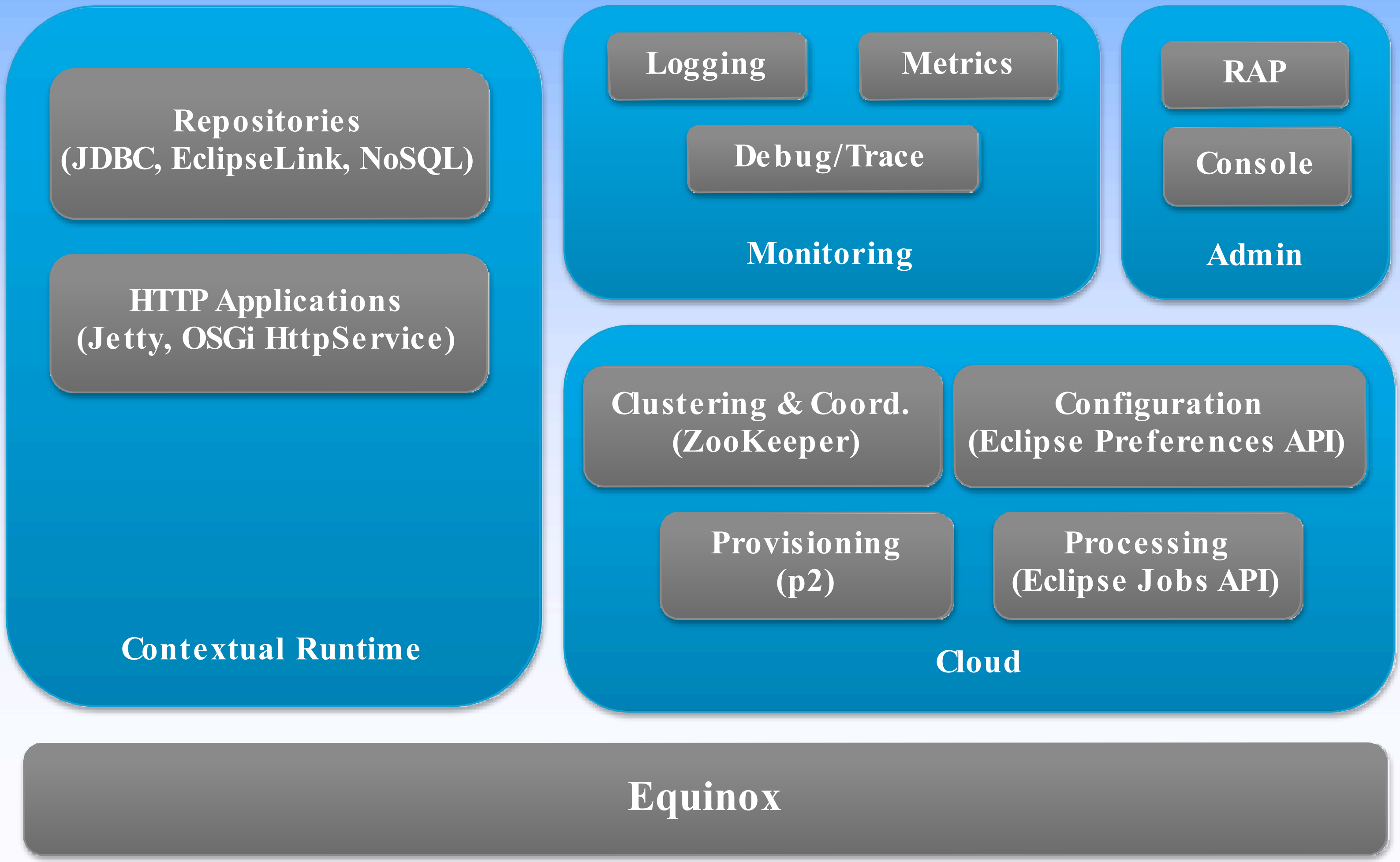
Gyrex

- built-in **clustering**
- built-in **web-based administration** UI
- built-in **multi tenancy**
- enhancements for **professional maintenance**
  - centralized logging
  - cluster provisioning UI

Gyrex Features

- lightweight **application stack**
- fast **100% OSGi** runtime
- central **cluster** configuration through Apache ZooKeep
- cluster aware **job scheduling**
- **automated deployment** through p2
- support for cluster **node roles**, e.g. “job worker node” and “api node”

Gyrex Components



Repositories (JDBC, EclipseLink, NoSQL)

HTTP Applications (Jetty, OSGi HttpService)

Contextual Runtime

Logging

Metrics

Debug/Trace

Monitoring

RAP

Console

Admin

Clustering & Coord. (ZooKeeper)

Configuration (Eclipse Preferences API)

Provisioning (p2)

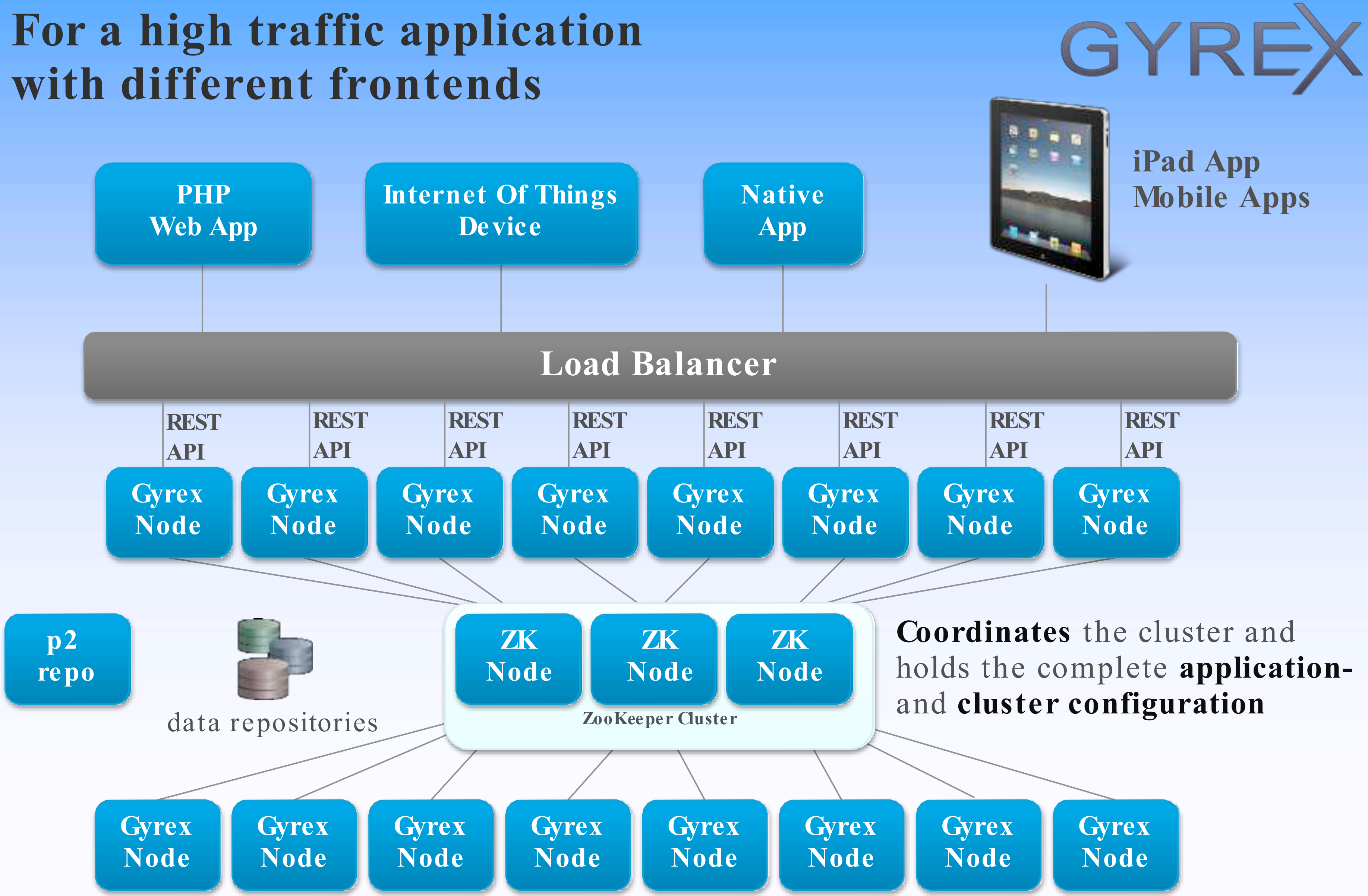
Processing (Eclipse Jobs API)

Cloud

Equinox

Gyrex Infrastructure Setup

For a high traffic application with different frontends



Coordinates the cluster and holds the complete **application- and cluster configuration**

Q&A

- Gyrex Newsgroup / Forum at **http://www.eclipse.org/forums/**

- Information hub at **http://www.eclipse.org/gyrex/**

- Session feedback / questions **gunnar@eclipse.org** **@guw**

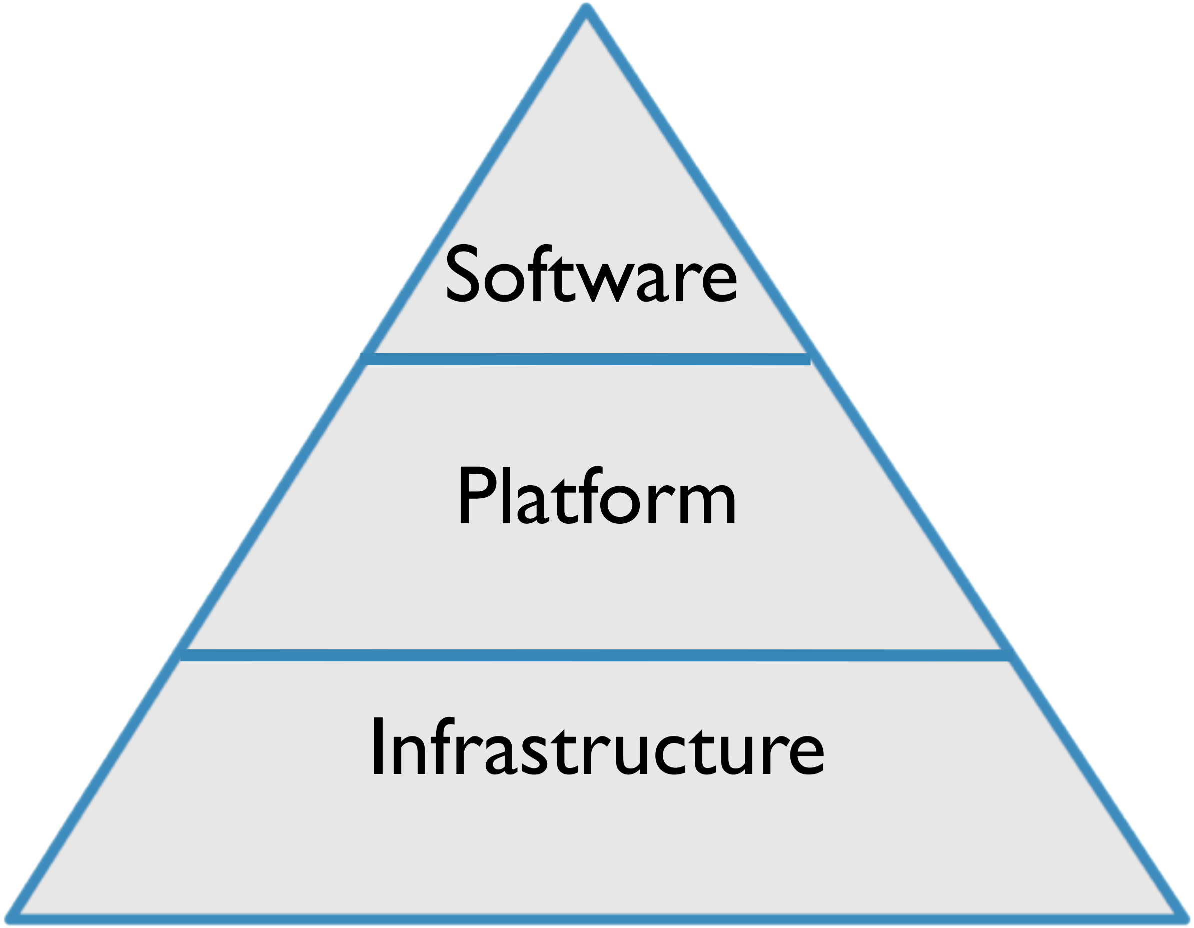
Google App Engine

Ikai Lan plus.ikailan.com NYC
Agency Day! July 25, 2011

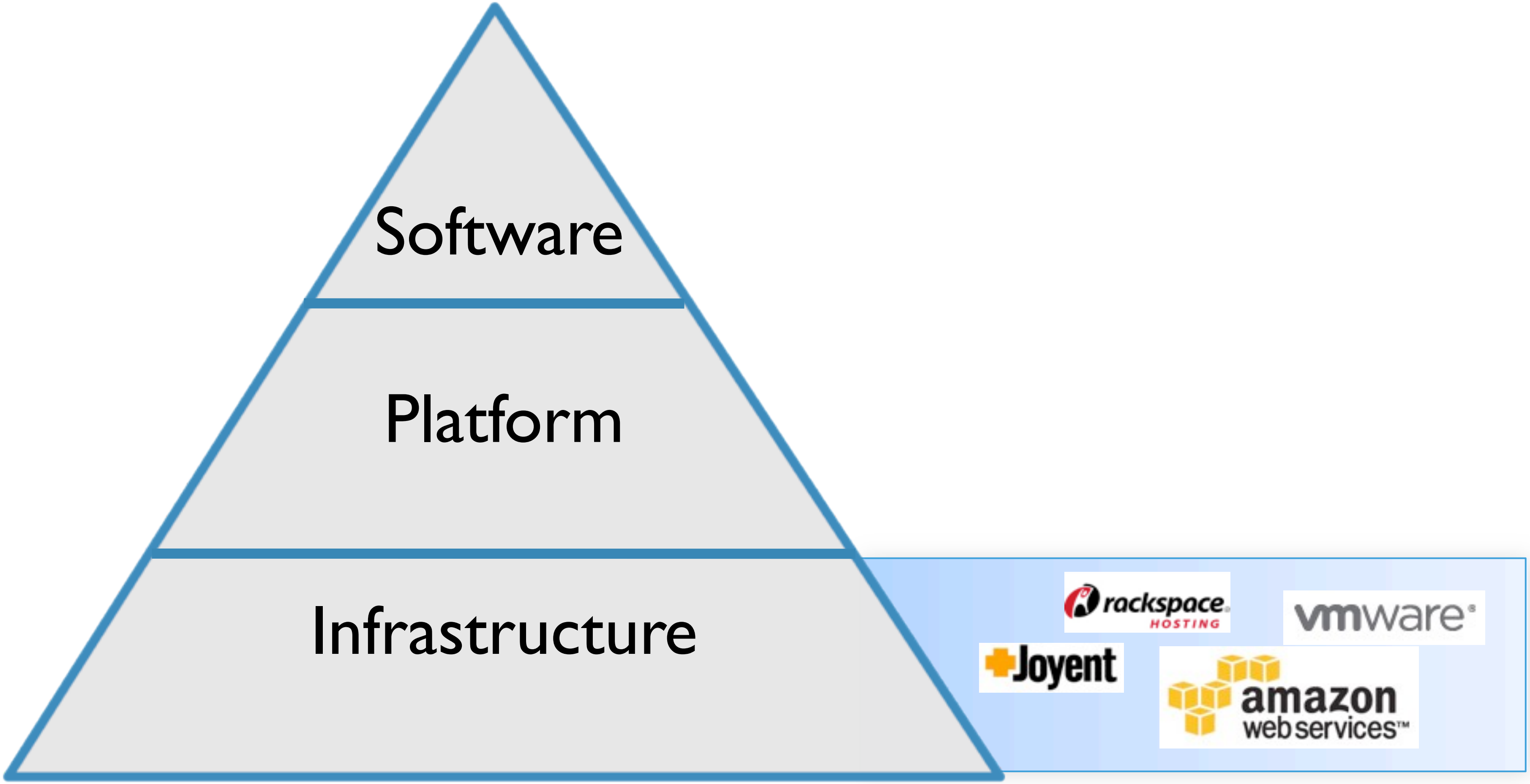
Agenda

- What is Google App Engine?
- Anatomy of an App Engine application
- Demo app and questions

What is App Engine?



Source: Gartner AADI Summit Dec 2009

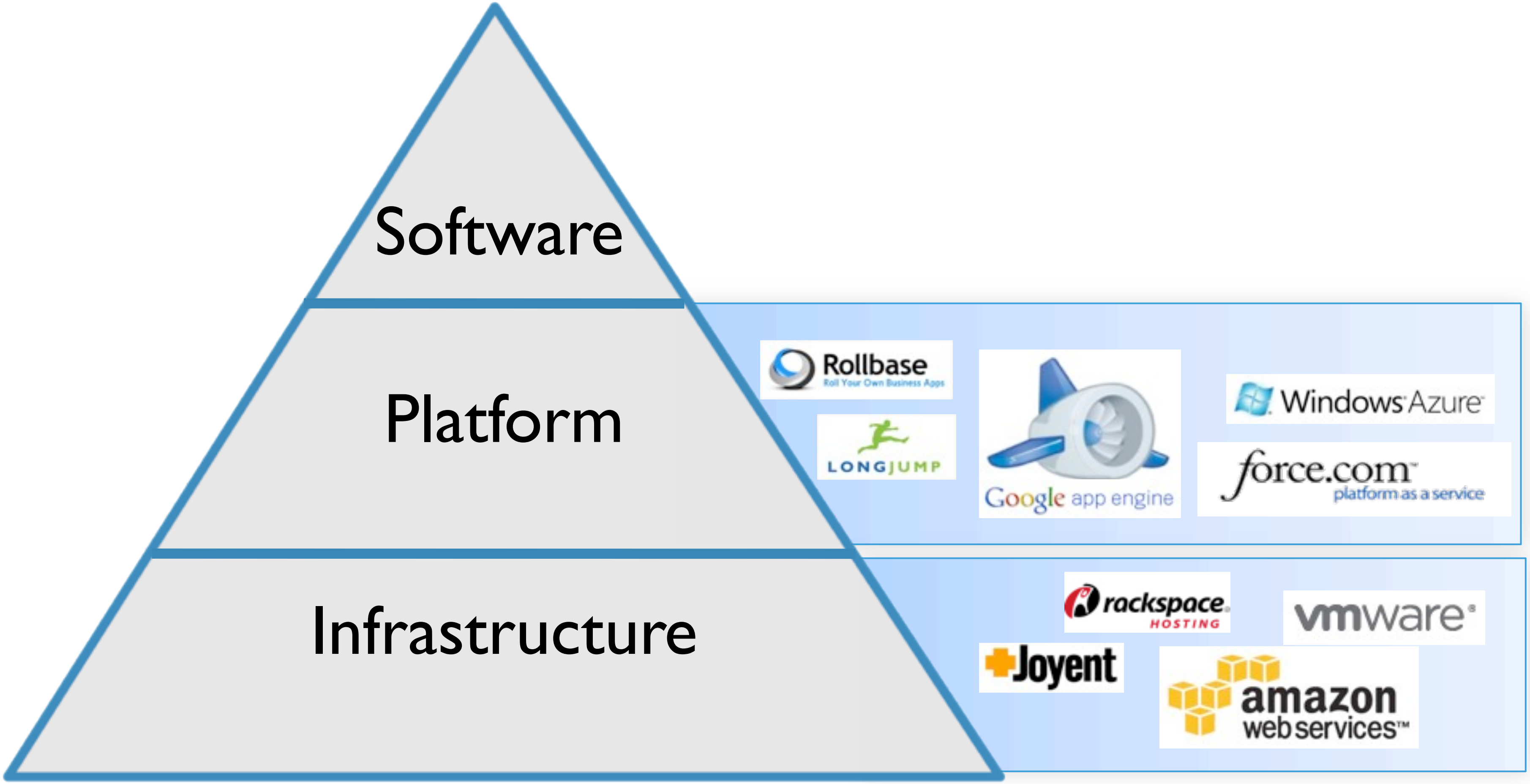


Software

Platform

Infrastructure

Source: Gartner AADI Summit Dec 2009

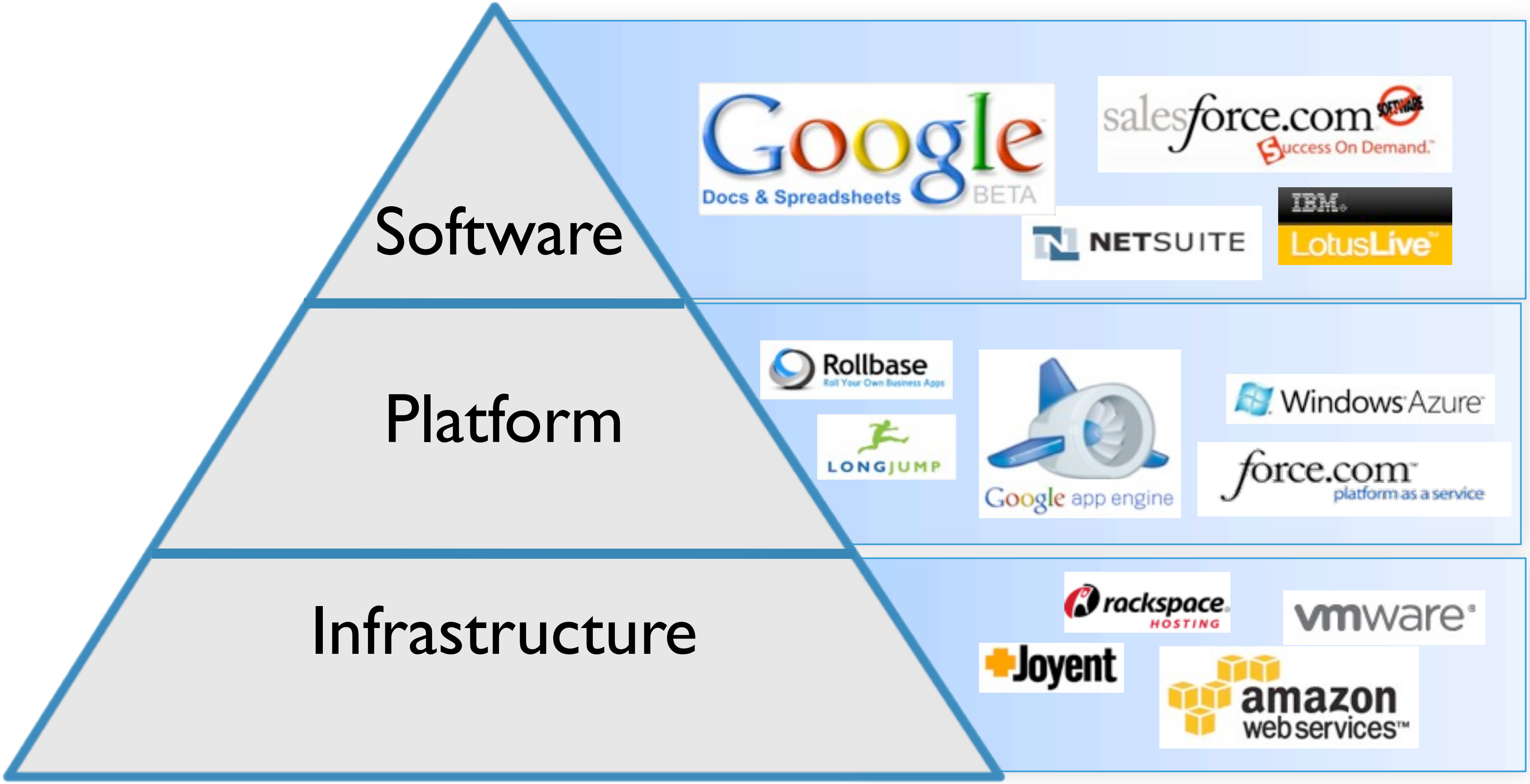


Software

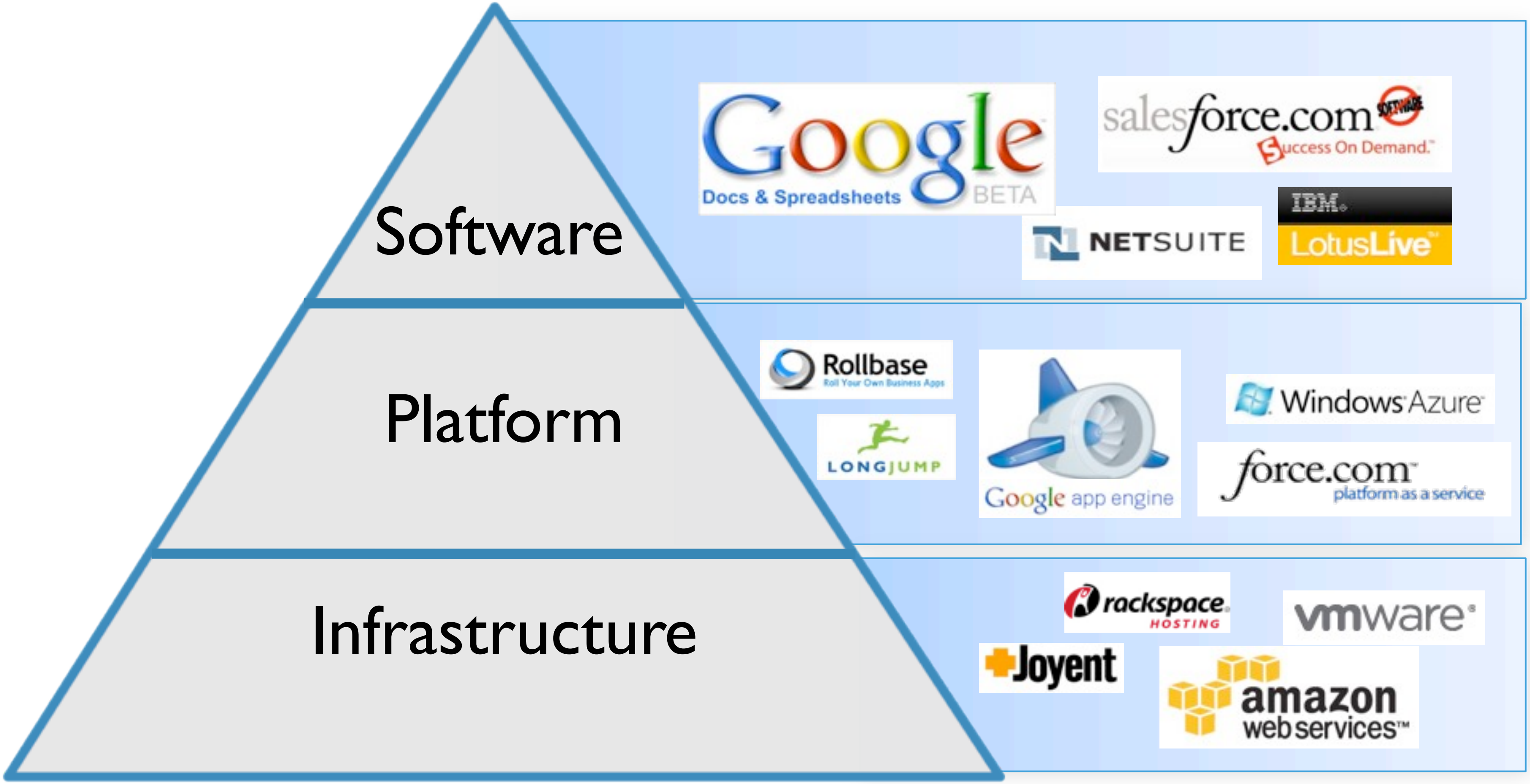
Platform

Infrastructure

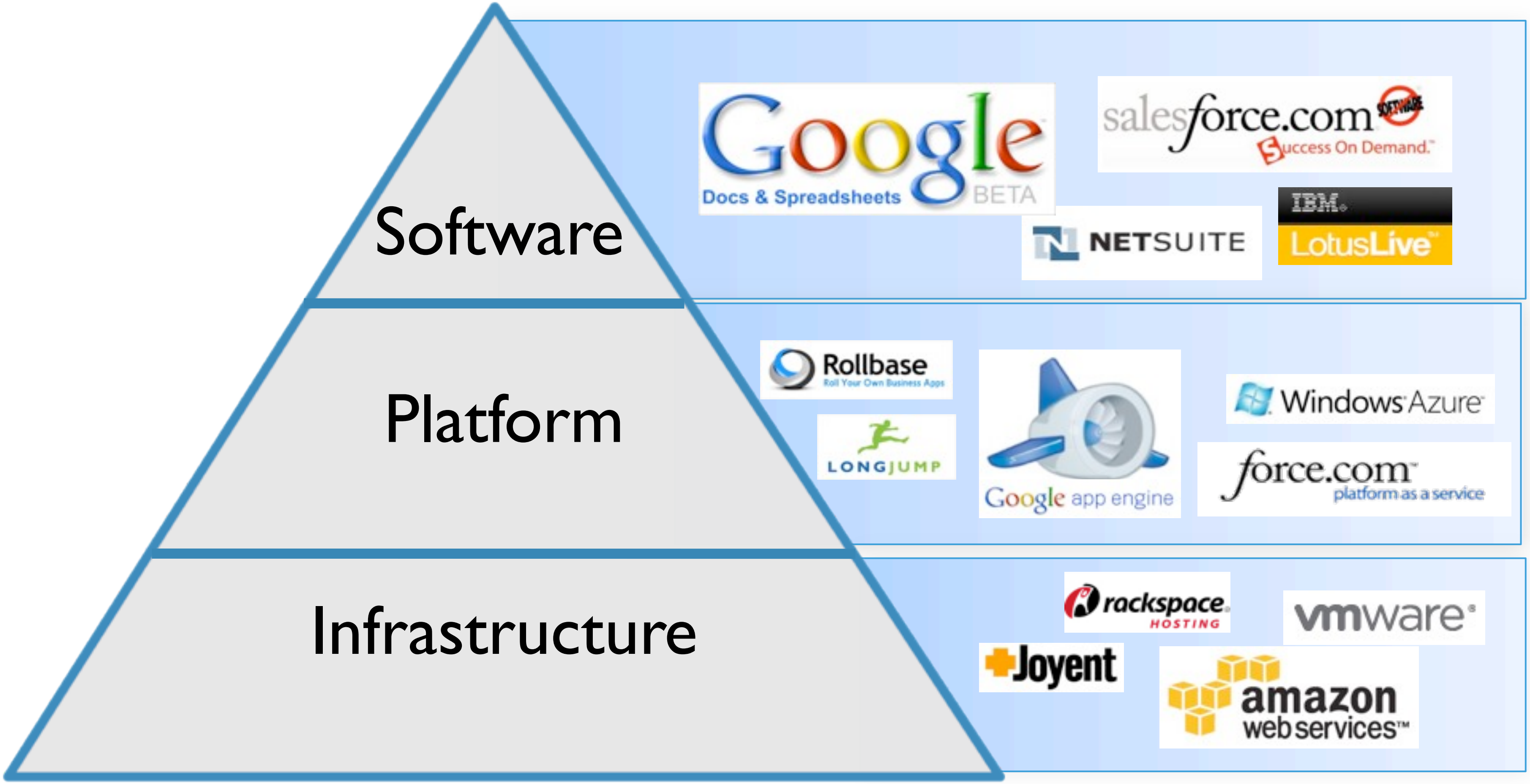
Source: Gartner AADI Summit Dec 2009



Source: Gartner AADI Summit Dec 2009



Source: Gartner AADI Summit Dec 2009



Software

Platform

Infrastructure

Source: Gartner AADI Summit Dec 2009

SDK & “The Cloud”

Hardware

Networking

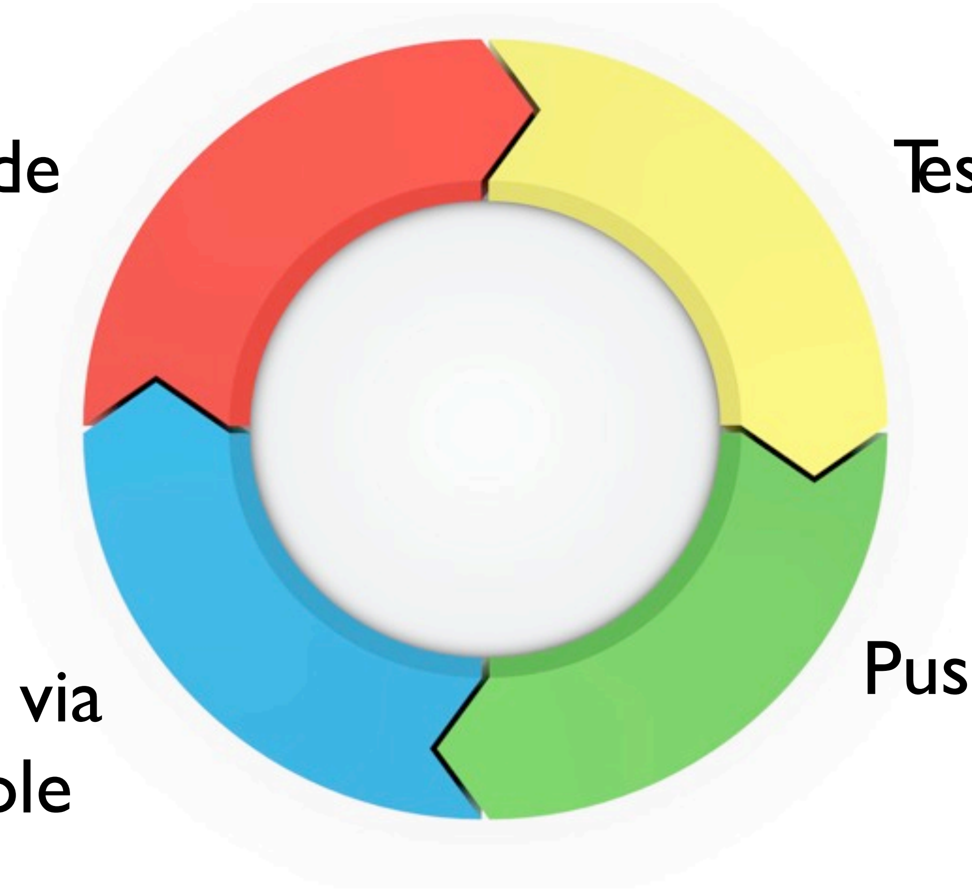
Operating system

Application runtime

Java, Python, Go

Static file serving

Development Cycle



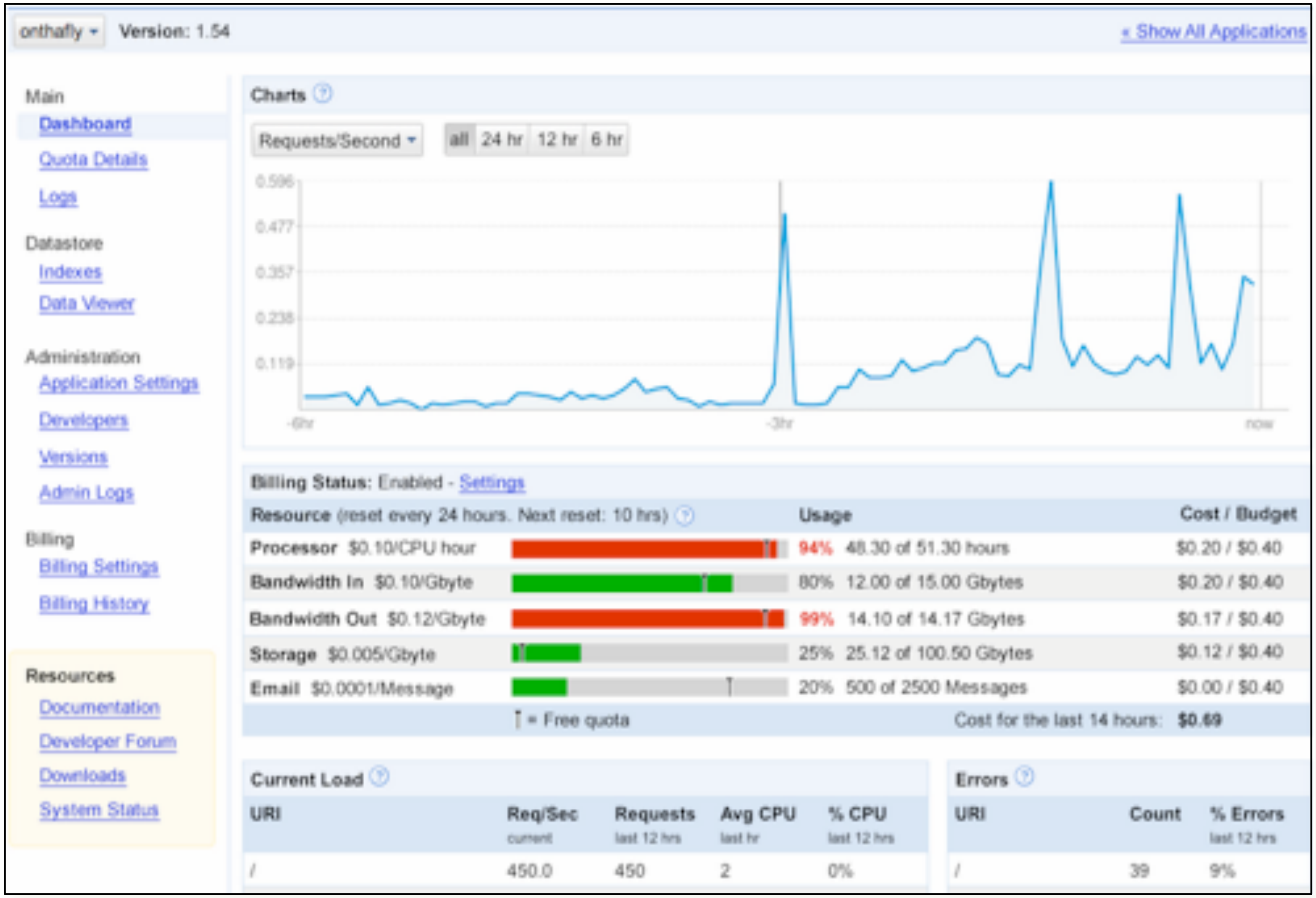
Write code

Test locally

Push to Google servers

Administer via web console

Admin Console



Billing Status: Enabled - Settings

Resource (reset every 24 hours. Next reset: 10 hrs)		Usage		Cost / Budget
Processor $0.10/CPU hour		94%	48.30 of 51.30 hours	$0.20 / $0.40
Bandwidth In $0.10/Gbyte		80%	12.00 of 15.00 Gbytes	$0.20 / $0.40
Bandwidth Out $0.12/Gbyte		99%	14.10 of 14.17 Gbytes	$0.17 / $0.40
Storage $0.005/Gbyte		25%	25.12 of 100.50 Gbytes	$0.12 / $0.40
Email $0.0001/Message		20%	500 of 2500 Messages	$0.00 / $0.40
	Ｉ = Free quota		Cost for the last 14 hours:	**$0.69**

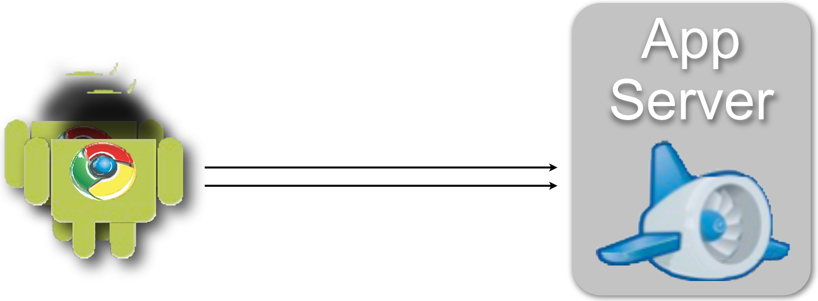
Current Load

URI	Req/Sec current	Requests last 12 hrs	Avg CPU last hr	% CPU last 12 hrs
/	450.0	450	2	0%

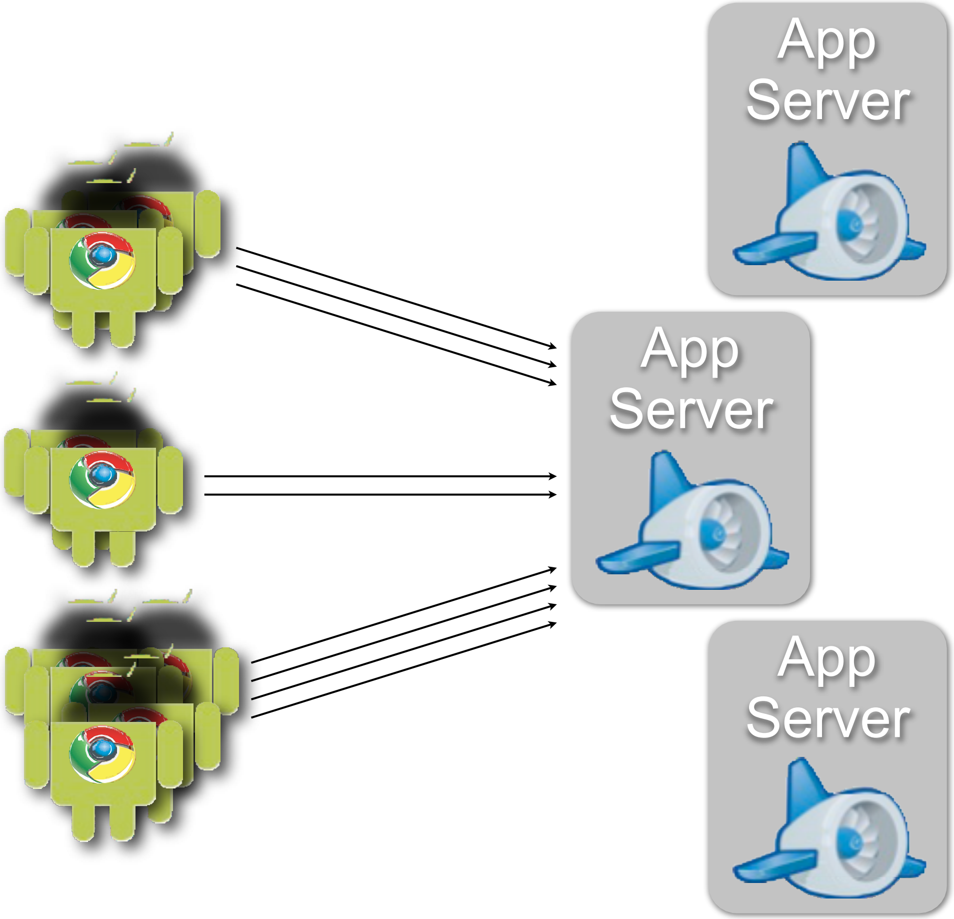
Errors

URI	Count	% Errors last 12 hrs
/	39	9%

Scales dynamically



Scales dynamically

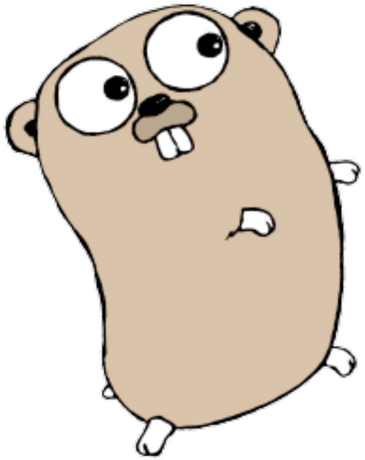


Google App Engine

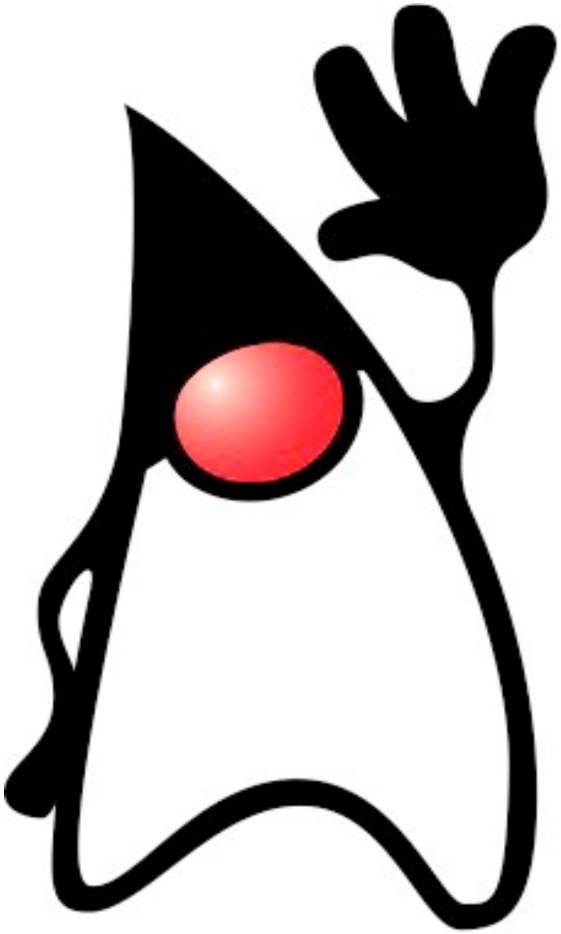
Easy to **build**
Easy to **manage**
Easy to **scale**

Google App Engine

“We wear pagers so
you don’t have to”



Go Gopher

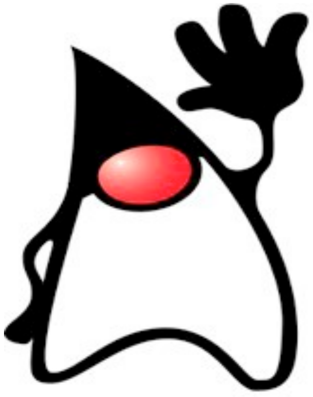


Duke, the Java mascot

Copyright © Sun Microsystems Inc., all rights reserved.

Extended Language support through JVM

- Java
- Scala
- JRuby (Ruby)
- Groovy
- Quercus (PHP)
- Rhino (JavaScript)
- Jython (Python)
- Clojure



Duke, the Java mascot

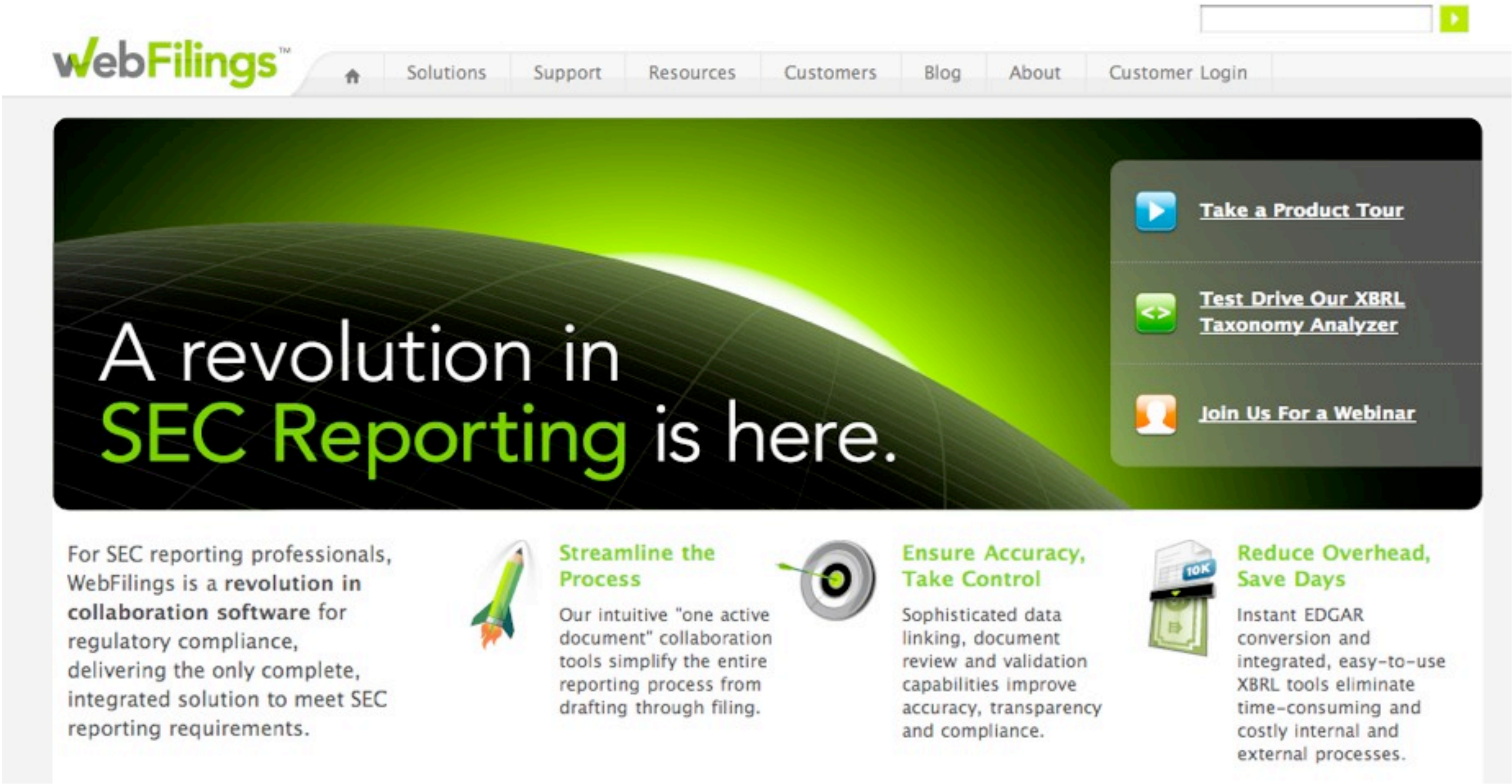
Copyright © Sun Microsystems Inc., all rights reserved.

>100K Developers

>200K Apps

>1.5B daily pageviews

Customer: WebFilings



A revolution in SEC Reporting is here.

For SEC reporting professionals, WebFilings is a **revolution in collaboration software** for regulatory compliance, delivering the only complete, integrated solution to meet SEC reporting requirements.

Streamline the Process

Our intuitive "one active document" collaboration tools simplify the entire reporting process from drafting through filing.

Ensure Accuracy, Take Control

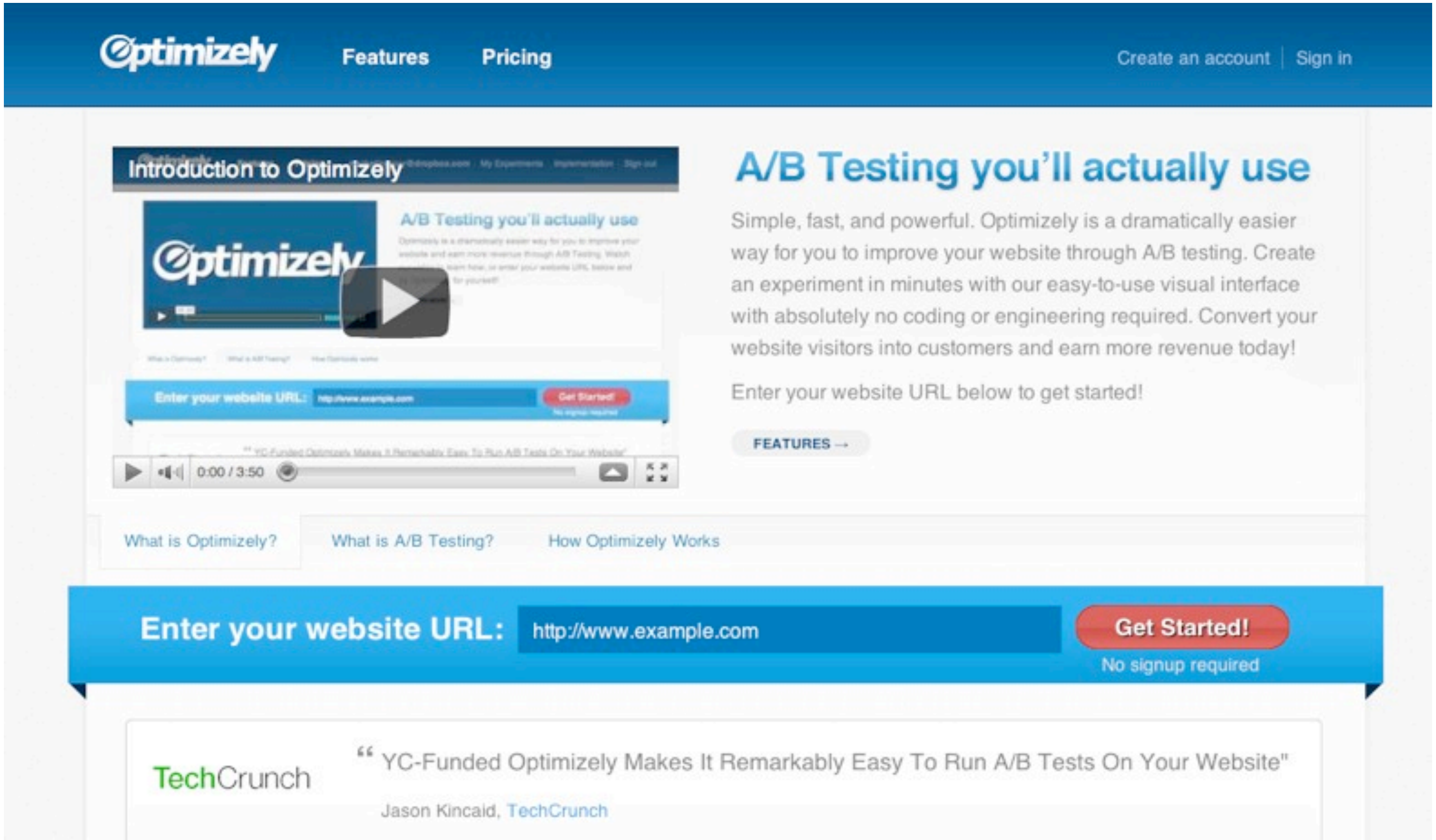
Sophisticated data linking, document review and validation capabilities improve accuracy, transparency and compliance.

Reduce Overhead, Save Days

Instant EDGAR conversion and integrated, easy-to-use XBRL tools eliminate time-consuming and costly internal and external processes.

Disruptive multi-tenant App Engine application adopted by Fortune 500 companies.

Customer: Optimizely



A/B Testing you'll actually use

Simple, fast, and powerful. Optimizely is a dramatically easier way for you to improve your website through A/B testing. Create an experiment in minutes with our easy-to-use visual interface with absolutely no coding or engineering required. Convert your website visitors into customers and earn more revenue today!

Enter your website URL below to get started!

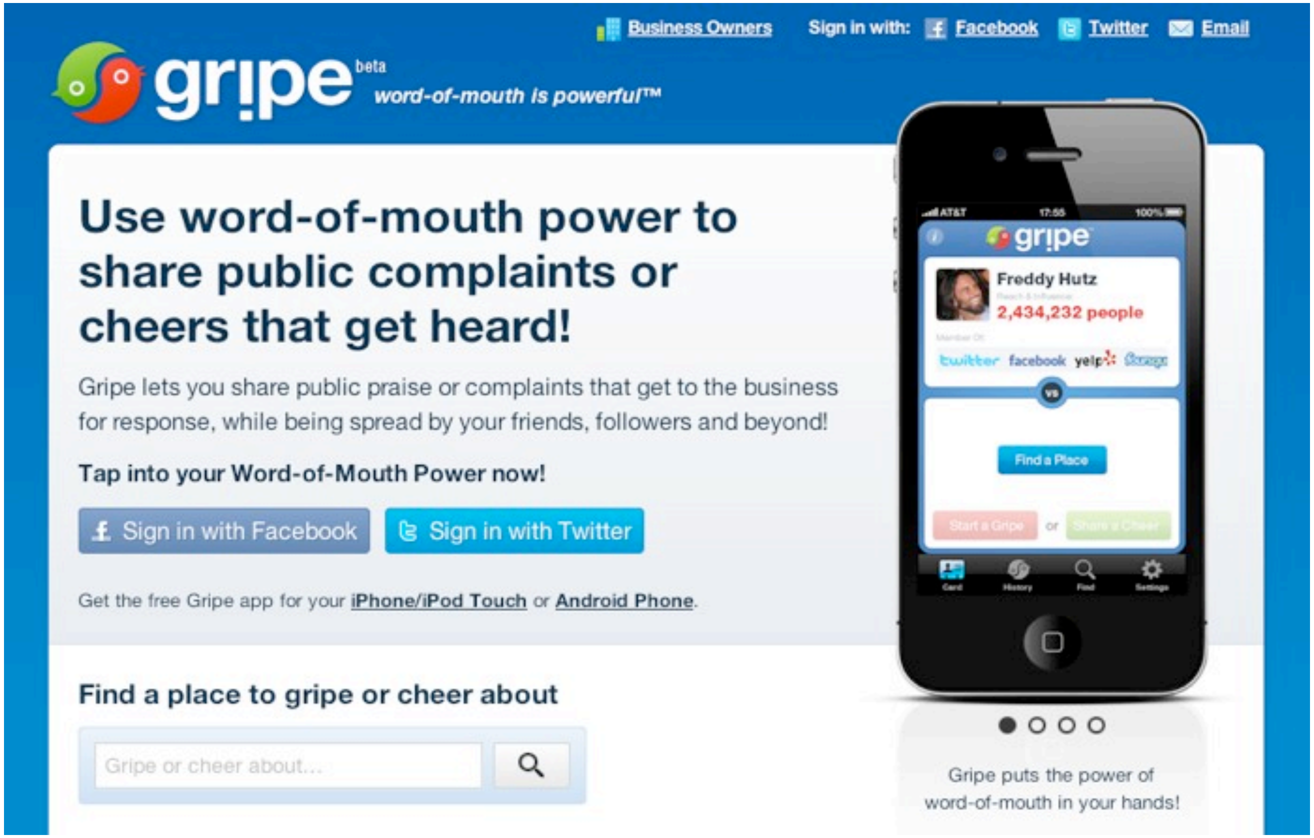
Enter your website URL: http://www.example.com

" YC-Funded Optimizely Makes It Remarkably Easy To Run A/B Tests On Your Website"

Jason Kincaid, TechCrunch

Y-Combinator funded startup now processing 250M events through App Engine

Customer: Gri!pe



Business Owners Sign in with: Facebook Twitter Email

gripe beta word-of-mouth is powerful™

Use word-of-mouth power to share public complaints or cheers that get heard!

Gripe lets you share public praise or complaints that get to the business for response, while being spread by your friends, followers and beyond!

Tap into your Word-of-Mouth Power now!

Sign in with Facebook | Sign in with Twitter

Get the free Gripe app for your iPhone/iPod Touch or Android Phone.

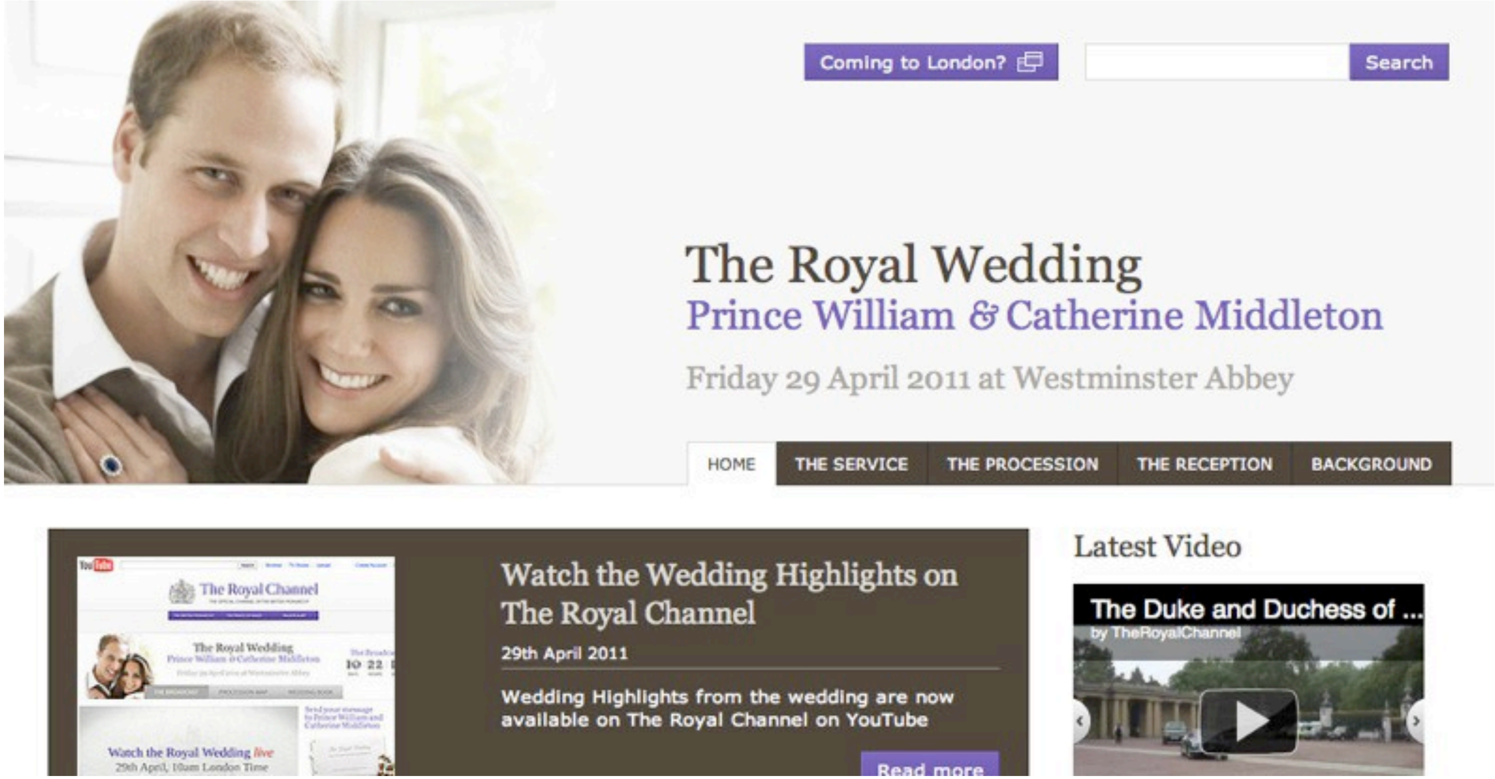
Find a place to gripe or cheer about

Gripe or cheer about...

Gripe puts the power of word-of-mouth in your hands!

Mobile app backend - Used App Engine to rapidly scale to serve traffic spikes from industry press and appearance on The View

Customer: The Royal Wedding



The Royal Wedding
Prince William & Catherine Middleton
Friday 29 April 2011 at Westminster Abbey

Watch the Wedding Highlights on The Royal Channel
29th April 2011
Wedding Highlights from the wedding are now available on The Royal Channel on YouTube

The Royal Channel
The Royal Wedding
Watch the Royal Wedding *live*
29th April, 10am London Time

Latest Video
The Duke and Duchess of ...
by TheRoyalChannel

Peaked at 32,000 requests per second with no disruption!

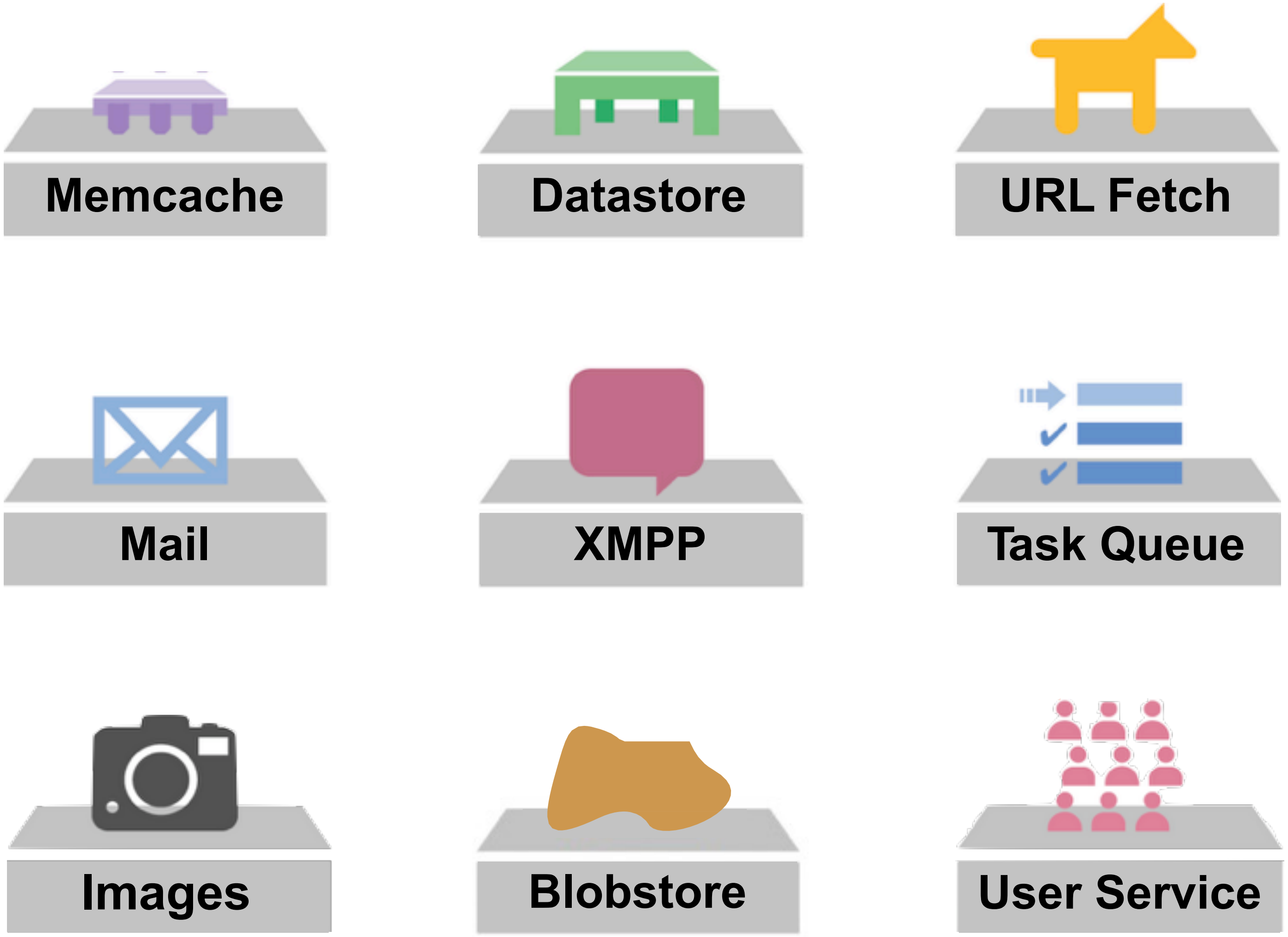
Anatomy of an App Engine application

Bloggart

Open Source Python Blog that runs on App Engine

https://github.com/Arachnid/bloggart

Core APIs



Memcache

Datastore

URL Fetch

Mail

XMPP

Task Queue

Images

Blobstore

User Service

Bloggart

Datastore

Blog post storage

Memcache

General caching Long

running

Task Queue

background jobs

User Service

Admin authentication

URL Fetch

Comments via Disqus API

App Engine Datastore

Schemaless, non-relational datastore built on top of Google's Bigtable technology

Enables **rapid development** and **scalability**

High Replication

- strongly consistent
- multi datacenter
- **High reliability**
- consistent performance
- **no data loss**

More features

- Prospective Search API http://code.google.com/appengine/docs/python/prospectivesearch/
- High Performance Image API
- App Engine Mapreduce
- Pipeline API

Summary

- LetApp Engine drive for you - worry about where you are going
- App Engine enables **rapid development cycles** without sacrificing **scalability** or **reliability**

Eclipse WTP
Web Service Tools

Kathy Chan

IBM Rational Software

WTP Committer

kathy@ca.ibm.com

Contents

- Overview of WTP
- Web Services tools in WTP
- What’s new in Web services tools in WTP 1.5
- Demo
- The future
- References

WTP Project Overview

- WTP provides tools for Java Web application development
  - Tools for application developers
  - Platform for tool developers
- Subprojects focus on industry standards
  - Web Standard Tools – IETF, W3C, OASIS, WS-I, ANSI, etc
  - J2EE Standard Tools – JCP

WTP WST - Web Standard Tools

- Web Projects
- Web server control
- Structured Source Editing Framework
- HTML
- JavaScript
- CSS
- XML
- DTD
- XSD
- Web services (WSDL, WS-I)
- Data access

WTP JST - J2EE Standard Tools

- J2EE Projects
- J2EE server control
- Servlets
- JSP^TM
- EJB^TM
- Java Web services (JAX-RPC)

Web Services Tools Overview

- WST Internet Proxy Preferences
- WST Environment / Command Frameworks
- WST Web Services
  - Web Services Explorer
  - Web Services Scenario and Finder Framework
  - Web Services Ant tasks
- JST Web Services
  - Extensible Web Services Wizardry
  - Bottom-up, Top-down and Client scenarios
  - End-to-end orchestration, driving J2EE and Server tools
  - Extensions for Apache Axis

WST Web Services

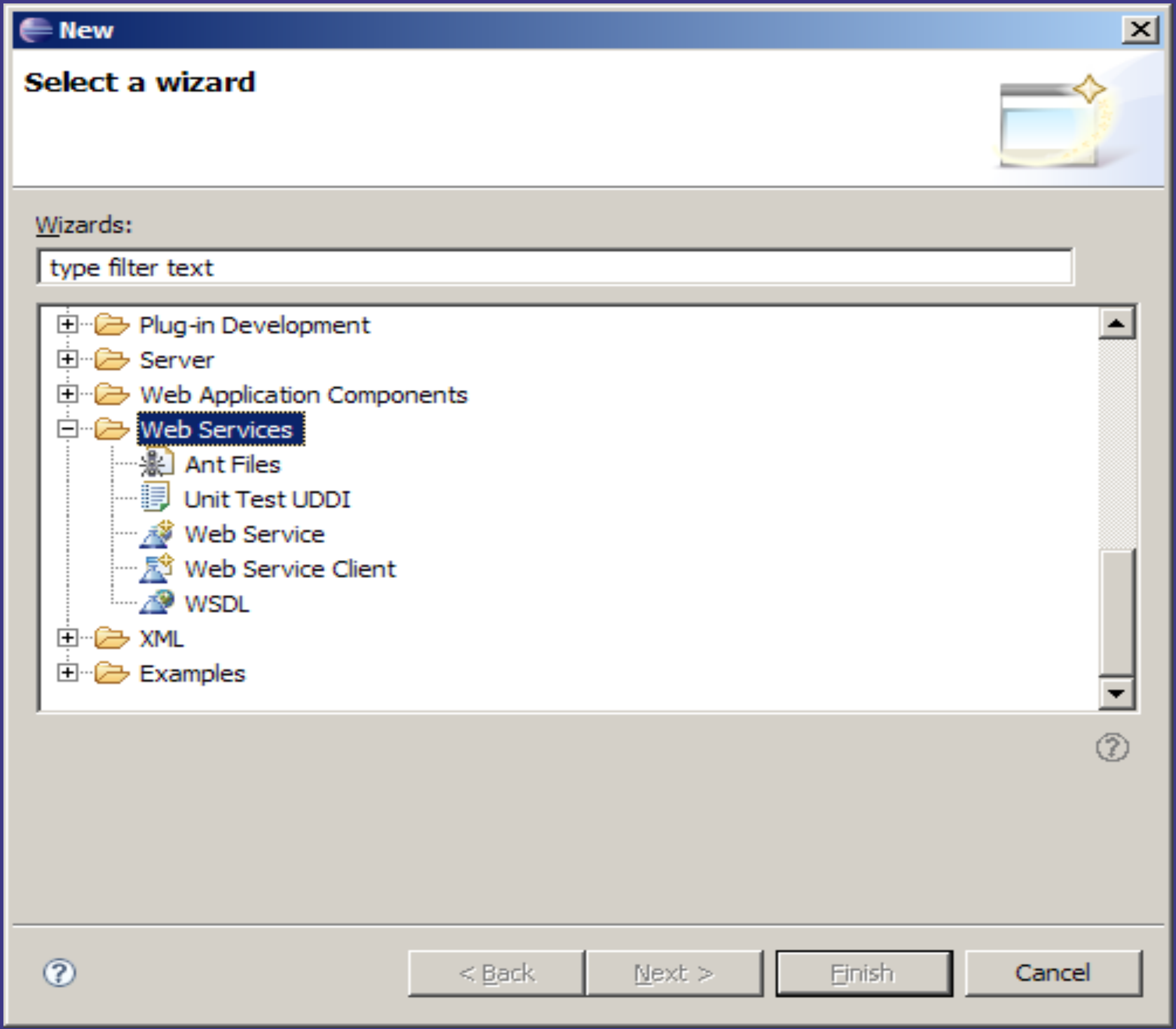
- Web Services Preferences
- Web Services Explorer
  - Integrated Web application
  - Publication, discovery and WSDL / XSD native testing
- Web Services extension points and provisional API for
  - Discover
  - Develop / Assemble / Deploy / Install / Run
  - Test
  - Publish
  - Web Services Finder
- Web Services Ant tasks

JST Web Services

Web Services for Java (or other languages)

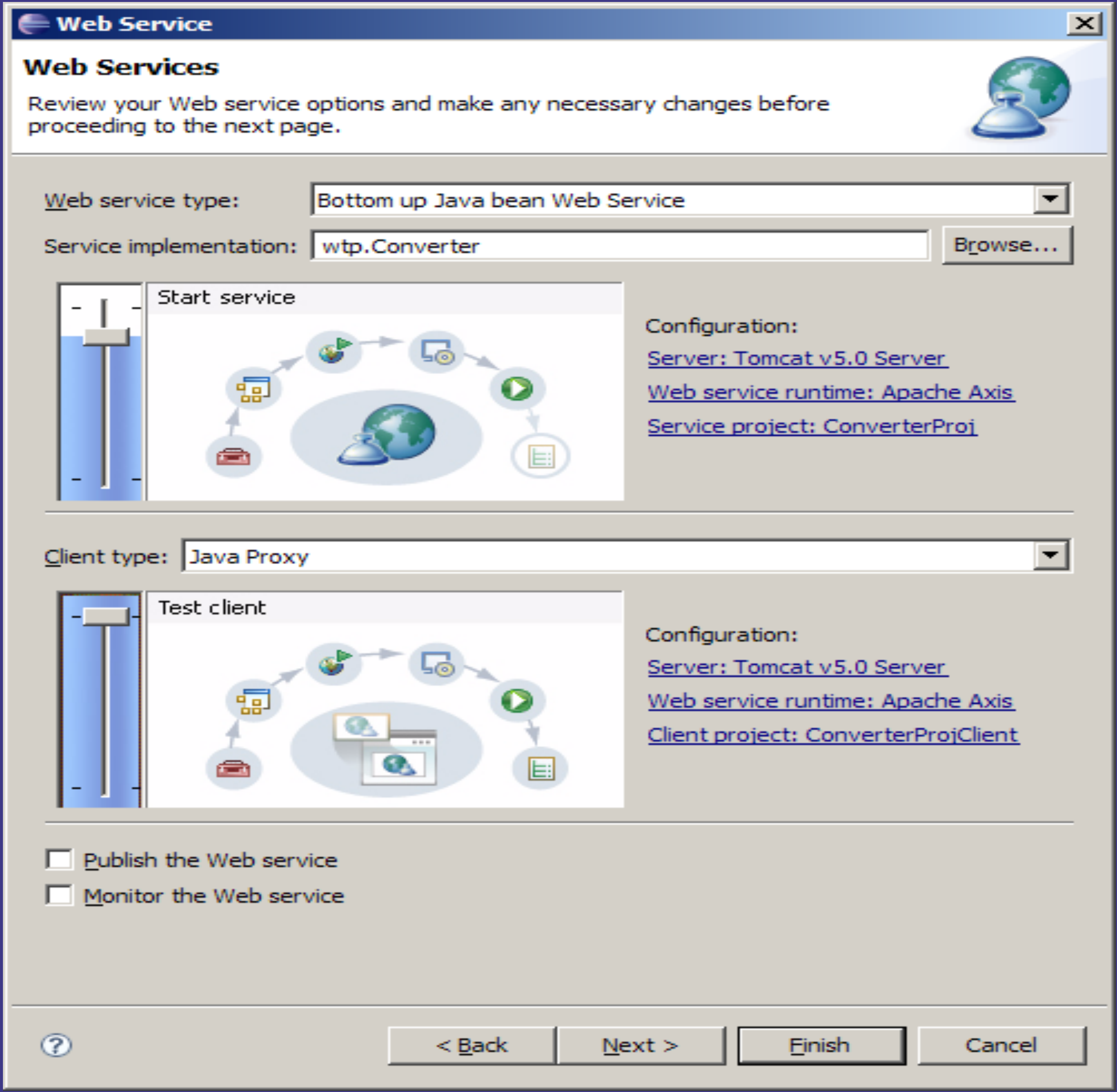
- Preferences
- Web service client wizardry
- Web service creation wizardry
- Extensions for Apache Axis on various servers
- JSR-109 handler configuration wizardry
- Java Web service finders
- Web services import / export

Wizardry



New
Select a wizard
Wizards:
type filter text
Plug-in Development
Server
Web Application Components
Web Services
Ant Files
Unit Test UDDI
Web Service
Web Service Client
WSDL
XML
Examples
< Back
Next >
Finish
Cancel

Wizardry



Web Services

Review your Web service options and make any necessary changes before proceeding to the next page.

Web service type: Bottom up Java bean Web Service

Service implementation: wtp.Converter

Start service

Configuration:

Server: Tomcat v5.0 Server

Web service runtime: Apache Axis

Service project: ConverterProj

Client type: Java Proxy

Test client

Configuration:

Server: Tomcat v5.0 Server

Web service runtime: Apache Axis

Client project: ConverterProjClient

☐ Publish the Web service

☐ Monitor the Web service

Using Java to implement SOAP web Services: JAX-WS

Web Technology
2II25

Dr. Katrien Verbert
Dr. ir. Natasha Stash
Dr. George Fletcher

TU/e Technische Universiteit **Eindhoven** University of Technology

Where innovation starts

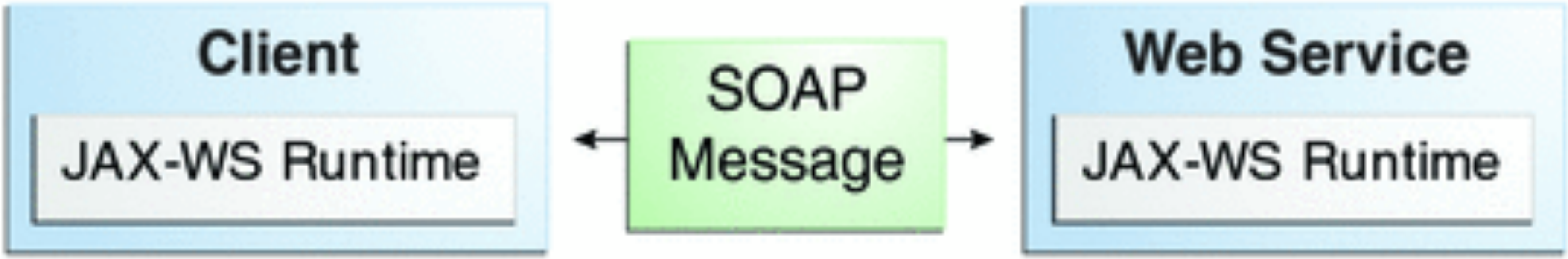
JAX-WS 2.0

- Part of Java EE
- New in Java SE 6
- API stack for web services.
- New API's:
  - JAX-WS, SAAJ, Web Service metadata
- New packages:
  - javax.xml.ws, javax.xml.soap, javax.jws

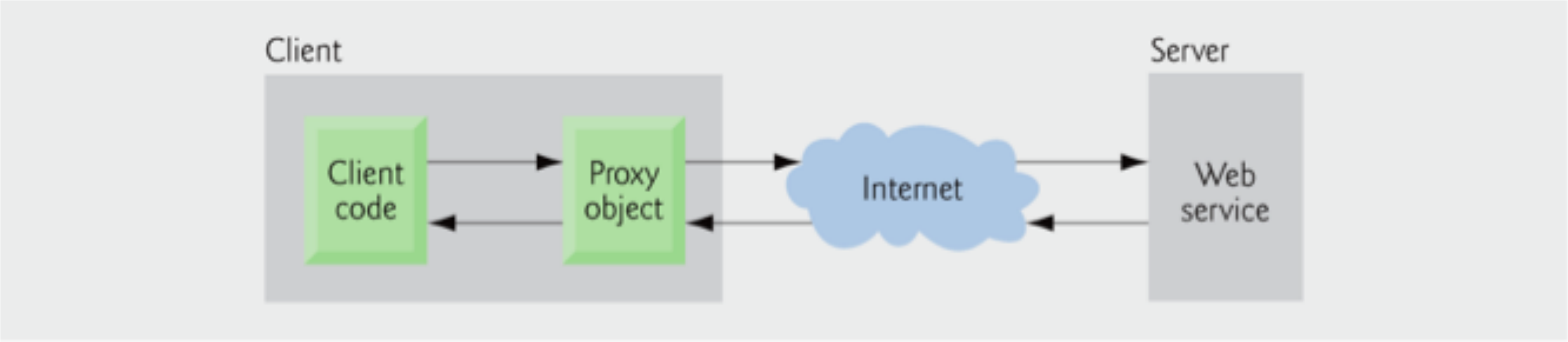
Java Web Services Basics

- Remote machine or server
  - The computer on which a web service resides
- A client application that accesses a web service sends a method call over a network to the remote machine, which processes the call and returns a response over the network to the application
- In Java, a web service is implemented as a class that resides on a server
- Publishing a web service
  - Making a web service available to receive client requests
- Consuming a web service
  - Using a web service from a client application

Communication between JAX-WS Web Service and Client



Java Web Services Basics



Writing a webservice

```
package loanservice;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.xml.ws.Endpoint;

@WebService
public class LoanApprover {

    @WebMethod
    public boolean approve(String name) {
        return name.equals("Mike");
     }
}
```

Annotations

Annotations are a new feature of JDK 1.5 and later.

- Essentially they are markers in the Java source code
- That can be used by external tools to generate code

Format looks like

```
@ThisIsAnAnnotation(foo=“bar”)
```

Annotations can occur only in specific places in the code

- before a class definition,
- before a method declaration, ...

Requirements of a JAX-WS Endpoint

- The implementing class must be annotated with the @WebService or @WebServiceProvider annotation
- The business methods of the implementing class must be public.
- The business methods must not be declared static or final.
- Business methods that are exposed to web service clients must be annotated with @WebMethod.
- Business methods that are exposed to web service clients must have JAXB-compatible parameters and return types.
    - See the list of JAXB default data type bindings at
    - http://docs.oracle.com/javaee/5/tutorial/doc/bnazq.html#bnazs.

@WebService annotation

- Indicates that a class represents a web service
- Optional element name
  - specifies the name of the proxy class that will be generated for the client
- Optional element serviceName
  - specifies the name of the class to obtain a proxy object.

Creating, Publishing, Testing and Describing a Web Service

Calculator web service

- Provides method that takes two integers
- Can determine their sum

CalculatorWS example

```
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

@WebService(serviceName = "CalculatorWS")
public class CalculatorWS {

    @WebMethod
    public int add (@WebParam (name= "value1") int value1,
      @WebParam( name="value2" ) int value2){
        return value1 + value2;
    }
}
```

Declare that method add is a WebMethod

Specify parameter names

Coding the Service Endpoint Implementation Class

- @WebService annotation at the beginning of each new web service class you create
- @WebMethod annotation at the beginning of each method that is exposed as a WSDL operation
  - Methods that are tagged with the @WebMethod annotation can be called remotely
  - Methods that are not tagged with @WebMethod are not accessible to clients that consume the web service
- @WebParam annotation is used here to control the name of a parameter in the WSDL
  - Without this annotation the parameter name = arg0

- @WebMethod annotation
  - Optional operationName element to specify the method name that is exposed to the web service’s client
- Parameters of web methods are annotated with the @WebParam annotation
  - Optional element name indicates the parameter name that is exposed to the web service’s clients

Building, Packaging, and Deploying the Service

Java IDEs

- Netbeans
  - download: http://netbeans.org/
  - tutorial: http://netbeans.org/kb/docs/websvc/jax-ws.html?print=yes
- Eclipse
  - download: http://www.eclipse.org/
  - tutorial: http://rantincsharp.wordpress.com/2008/10/14/a-simple-soap-web-service-example-in-eclipse-ganymede/
- IntelliJ IDEA
  - download: http://www.jetbrains.com/idea/
  - tutorial: http://wiki.jetbrains.net/intellij/Web_Services_with_IntelliJ_IDEA#JAX_WS

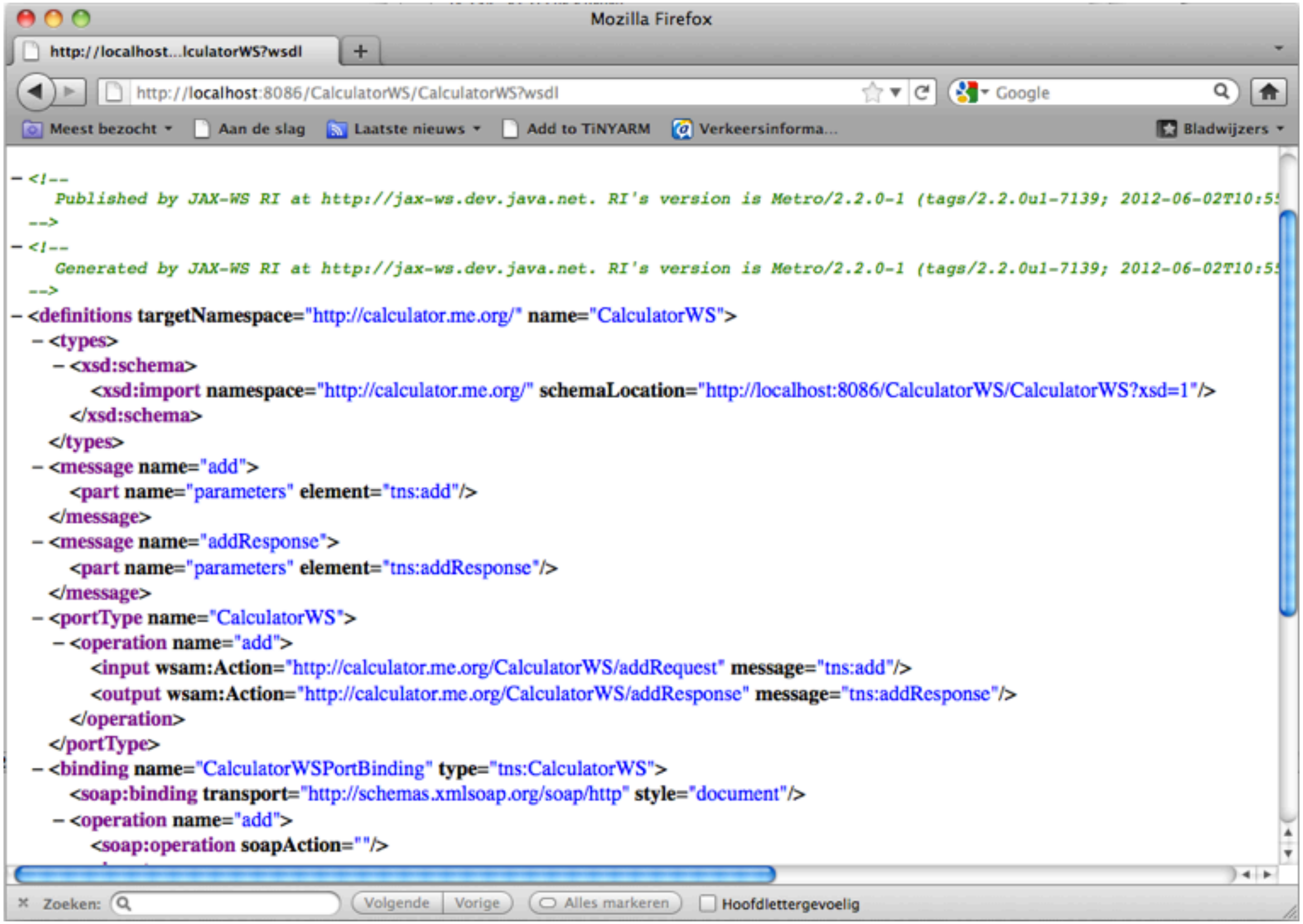
Using Ant

http://docs.oracle.com/javaee/6/tutorial/doc/bnayn.html

Describing a Web Service with the Web Service Description Language (WSDL)

- To consume a web service
  - Must know where to find the web service
  - Must be provided with the web service’s description
- Web Service Description Language (WSDL)
  - Describe web services in a platform-independent manner
  - The server generates a WSDL dynamically for you
  - Client tools parse the WSDL to create the client-side proxy class that accesses the web service
- To view the WSDL for a web service
  - Type URL in the browser’s address field followed by ?WSDL or
  - Click the WSDL File link in the Sun Java System Application Server’s Tester web page

Example WSDL



```
- <!--
    Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is Metro/2.2.0-1 (tags/2.2.0u1-7139; 2012-06-02T10:5
  -->
- <!--
    Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is Metro/2.2.0-1 (tags/2.2.0u1-7139; 2012-06-02T10:5
  -->
- <definitions targetNamespace="http://calculator.me.org/" name="CalculatorWS">
  - <types>
    - <xsd:schema>
        <xsd:import namespace="http://calculator.me.org/" schemaLocation="http://localhost:8086/CalculatorWS/CalculatorWS?xsd=1"/>
      </xsd:schema>
    </types>
  - <message name="add">
      <part name="parameters" element="tns:add"/>
    </message>
  - <message name="addResponse">
      <part name="parameters" element="tns:addResponse"/>
    </message>
  - <portType name="CalculatorWS">
    - <operation name="add">
        <input wsam:Action="http://calculator.me.org/CalculatorWS/addRequest" message="tns:add"/>
        <output wsam:Action="http://calculator.me.org/CalculatorWS/addResponse" message="tns:addResponse"/>
      </operation>
    </portType>
  - <binding name="CalculatorWSPortBinding" type="tns:CalculatorWS">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    - <operation name="add">
        <soap:operation soapAction=""/>
```

Creating a Client in Netbeans to Consume a Web Service

- When you add a web service reference
  - IDE creates and compiles the client-side artifacts
  - the framework of Java code that supports the client-side proxy class
- Client calls methods on a proxy object
  - Proxy uses client-side artifacts to interact with the web service
- To add a web service reference
  - Right click the client project name in the Netbeans Projects tab
  - Select New > Web Service Client…
  - Specify the URL of the web service’s WSDL in the dialog’s WSDL URL field

Calculator client

```
import calculator.*;

public class CalculatorClient {
    public static void main(String[] args) {
        CalculatorWS_Service service=new CalculatorWS_Service();
        CalculatorWS port= service.getCalculatorWSPort();
        int result = port.add(2, 3);
        System.out.println(result);
    }
}
```

Relevant links

- Netbeans tutorial for developing a SOAP-based web services:

  http://netbeans.org/kb/docs/websvc/jax-ws.html

- Building SOAP-based web services with JAX-WS:

  http://docs.oracle.com/javaee/6/tutorial/ doc/ bnayl.html

SOAP and XML processing

Web Technology
2II25

Dr. Katrien Verbert
Dr. ir. Natasha Stash
Dr. George Fletcher

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

XML document

```
<?xml version="1.0"?>
   <Order>
            <Date>2003/07/04</Date>
            <CustomerId>123</CustomerId>
            <CustomerName>Acme Alpha</CustomerName>
            <Item>
                  <ItemId> 987</ItemId>
                  <ItemName>Coupler</ItemName>
                  <Quantity>5</Quantity>
            </Item>
           <Item>
                  <ItemId>654</ItemId>
                  <ItemName>Connector</ItemName>
                  <Quantity unit="12">3</Quantity>
            </Item>
    </Order>
```

Parsing XML

Goal

Read XML files into data structures in programming languages

Possible strategies

- Parse into generic tree structure (DOM)
- Parse as sequence of events (SAX)
- Automatically parse to language-specific objects (JAXB)

JAXB

JAXB: Java API for XML Bindings

Defines an API for automatically representing XML schema as collections of Java classes.

Most convenient for application programming

Annotations markup

@XmlAttribute to designate a field as an attribute

@XmlRootElement to designate the document root element.

@XmlElement to designate a field as a node element

@XmlElementWrapper to specify the element that encloses a repeating series of elements

Note that you should specify only the getter method as @XmlAttribute or @XmlElement.

Jaxb oddly treats both the field and the getter method as independent entities

Order example

```
import javax.xml.bind.annotation.*;

@XmlRootElement
public class Item {

    @XmlElement
    private String itemId;
    @XmlElement
    private String ItemName;
    @XmlElement
    private int quantity;

    public Item() {
    }
}
```

Order example

```
import javax.xml.bind.annotation.*;
import java.util.*;

@XmlRootElement
public class Order {

    @XmlElement
    private String date;
    @XmlElement
    private String customerId;
    @XmlElement
    private String customerName;
    @XmlElement
    private List<Item> items;

    public Order() {
        this.items=new ArrayList<Item>();
    }
```

Marshalling

marshalling

the process of producing an XML document from Java objects

unmarshalling

the process of producing a content tree from an XML document

JAXB only allows you to unmarshal valid XML documents

JAXB only allows you to marshal valid content trees into XML

Marshalling example

```
public String toXmlString(){
        try{
            JAXBContext context=JAXBContext.newInstance(Order.class);
            Marshaller m = context.createMarshaller();
            m.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, Boolean.TRUE);
            ByteArrayOutputStream b=new ByteArrayOutputStream();
            m.marshal(this,b);
            return b.toString();
        }catch (Exception e){
            e.printStackTrace();
            return null;
        }
}
```

Unmarshalling example

```
public Order fromXmlString(String s){
    try{
        JAXBContext jaxbContext = JAXBContext.newInstance(Order.class);
        Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller()
        Order order = (Order) jaxbUnmarshaller.unmarshal(new StreamSource( new
         StringReader(s)));
        return order;
    }catch (Exception e){
        e.printStackTrace();
        return null;
    }
}
```

Test transformation

```
public static void main(String args[]){
        Order o=new Order("1 March 2013", "123", "Katrien");
        o.getItems().add(new Item("1", "iPhone 5", 2));
        o.getItems().add(new Item("2", "Nokia Lumia 800", 2));
        System.out.println(o.toXmlString());

    }
```

Output

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<order>
    <customerId>123</customerId>
    <customerName>Katrien Verbert</customerName>
    <date>12 February 2013</date>
    <items>
        <itemId>id1</itemId>
        <ItemName>Iphone 5</ItemName>
        <quantity>2</quantity>
    </items>
    <items>
        <itemId>id2</itemId>
        <ItemName>Nokia Lumia 800</ItemName>
        <quantity>1</quantity>
    </items>
</order>
```

Using Java to implement REST Web Services: JAX-RS

Web Technology
2II25

Dr. Katrien Verbert
Dr. ir. Natasha Stash
Dr. George Fletcher

Technische Universiteit **Eindhoven** University of Technology

Where innovation starts

Restful Web Services Frameworks and APIs

- JAX-RS - The Java API for RESTful Web Services
- uses annotations to make plain old Java objects (POJOs) and resources available through HTTP
- Sun Reference Project: Jersey
- Other Vendors: CXF (Apache), RESTEasy(JBoss) and Restlet

- JAX-RS tutorial:

  http://docs.oracle.com/javaee/6/tutorial/doc/gilik.html

JAX-RS HTTP

- Templated mapping and subresources

  @Path

- MIME handling

  @Provides, @Consumes

- HTTP methods

  @GET, @POST, @UPDATE, @DELETE, @HEAD, @OPTIONS, @HttpMethod

- Caching

  evaluatePreconditions

Example

```
package com.sun.jersey.samples.helloworld.resources;
import javax.ws.rs.GET;
import javax.ws.rs.Produces;
import javax.ws.rs.Path;

// The Java class will be hosted at the URI path "/helloworld"
@Path("/helloworld")
public class HelloWorldResource {

    // The Java method will process HTTP GET requests
    @GET
    // The Java method will produce content identified by the MIME Media
    // type "text/plain"
    @Produces("text/plain")
    public String getClichedMessage() {
        // Return some cliched textual content
        return "Hello World";
    }
}
```

@Path Annotation and URI Path Templates

@Path annotation

- identifies the URI path template to which the resource responds
- is specified at the class or method level of a resource

URI path templates are URIs with variables embedded within the URI syntax

- these variables are substituted at runtime
- variables are denoted by braces ({ and })

  @Path("/users/{username}")

- example request

  http://example.com/users/Galileo

@PathParam annotation

To obtain the value of the user name, the @PathParam annotation may be used on the method parameter of a request method

```
@Path("/users/{username}")
public class UserResource {

   @GET
   @Produces("text/xml")
   public String getUser(@PathParam("username") String userName) {
      ...
   }
}
```

Examples of URI Path Templates

URI Path Template	URI After Substitution
http://example.com/{name1}/{name2}/	http://example.com/james/gatz/
http://example.com/{question}/ {question}/{question}/	http://example.com/why/why/why/
http://example.com/maps/{location}	http://example.com/maps/Main%20Street

@Produces Annotation

@Produces annotation is used to specify the MIME media types or representations a resource can produce and send back to the client

- applied at the class level: default for all methods
- applied at the method level overrides any @Produces annotations applied at the class level

@produces annotation

```
@Path("/myResource")
@Produces("text/plain")
public class SomeResource {
    @GET
    public String doGetAsPlainText() {
        ...
    }

    @GET
    @Produces("text/html")
    public String doGetAsHtml() {
        ...
    }
}
```

@Consumes Annotation

@Consumes annotation is used to specify which MIME media types of representations a resource can accept

```
@POST
@Consumes("text/plain")
public void postClichedMessage(String message) {
    // Store the message
}
```

JAX RS data handling

- URI templates
  - {name}
  - {regex}
- Parameters
  - @PathParam
  - @QueryParam
  - @FormParam
  - @MatrixParam
  - @HeaderParam
  - @SessionParam
  - @UriInfo
  - @Encoded
  - @CoockieParam
  - @DefaultValue
  - No annotation (entity)

- Return valuesResponseGenericEntityNullStatus 204Entity (marshalled)ExceptionsWebApplicationExceptionUncheckedFilters, error pagesCheckedServletException, WebServiceException@Provider extends ExceptionMapper