Web Services and SOA

Rob Richards

5/22/2007

http://xri.net/=rob.richards

What is a Web Service?

"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network."

W3C Web Services Architecture Working Group,
W3C Working Group Note 11 February 2004

Types of Web Services

- RPC Services
  - More tightly coupled
  - Deal with operations
- Document based services
  - More loosely coupled
  - Deal with messages
- RESTful
  - Use common technologies and operations
  - Deal with state

Example Order Class

```
class Order {
    public $customerID;

    /* Create New or Retrieve Order */
    public function __constructor($orderID=0) {}

    /* Add Product to Order object */
    public function addProduct($sku, $quantity) {}

    /* Save Order  – returns orderID */
    public function saveOrder() {}

    /* retrieve orders by Customer ID */
    static function getOrders($customerID) {}
}
```

Example Order

```
/* Instantiate New Order Object */
$oOrder = new Order();
$oOrder->customerID = 12345;

/* Add Some Products */
$oOrder->addProduct(12345, 1);
$oOrder->addProduct(67890, 2);

/* Place the Order */
$orderID = $oOrder->saveOrder();
```

Order Processor Interface

orderType getOrder(int $orderID)

int createOrder(orderType $order)

orderListType getOrders(int $clientID)

OrderType Definition

```
<complexType name="OrderType">
  <sequence>
    <element name="orderId" type="ID" nillable="true"/>
    <element name="customerId" type="integer"/>
    <element maxOccurs="unbounded" ref="tns2:item"/>
  </sequence>
</complexType>
```

Item Definition

```
<element name="item">
  <complexType>
    <sequence>
    <element name="itemId" type="integer"/>
    <element name="quantity" type="integer"/>
    </sequence>
  </complexType>
</element>
```

Messages

```
<element name="createOrder">
  <complexType>
    <sequence>
      <element name="order" type="tns2:OrderType"/>
    </sequence>
  </complexType>
</element>
<element name="createOrderResponse" type="integer"/>
```

```
<message name="createOrderRequest">
  <part name="createOrderRequest" element="tns2:createOrder"/>
</message>
<message name="createOrderResponse">
  <part name="return" element="tns2:createOrderResponse"/>
</message>
```

PHP Type Classes

```
class cItem {
    public $itemId;
    public $quantity;

    public function __construct($itemId, $quantity) {
        $this->itemId = $itemId;
        $this->quantity = $quantity;
    }
}

class cOrder {
    public $orderId = NULL;
    public $customerId;
    public $item = array();
}
```

SOAP Server

```
function createOrder($message) {
    /* logic for placing order */
    /* return newly created Order ID
    return $orderID;
}

$server = new SoapServer("order.wsdl");
$server->addFunction("createOrder");
$server->handle();
```

SOAP Client

```
$sClient = new SoapClient('order.wsdl');

/* Instantiate new order object and set customer ID*/
$order = new cOrder();
$order->customerId = 1111;

/* Add some products */
$order->item[] = new cItem('12345', 1);
$order->item[] = new cItem('67890', 2);

$orderID = $sClient->createOrder(array('order'=>$order));

print $orderID;
```

Service Oriented Architecture
SOA

The SOA Hype

- Increased Return on Investment (ROI)
- Customer Retention
- Faster time to market
- Seamless Interoperability
- Decrease development time
- Simplify System maintenance
- Business Agility
- Re-usability

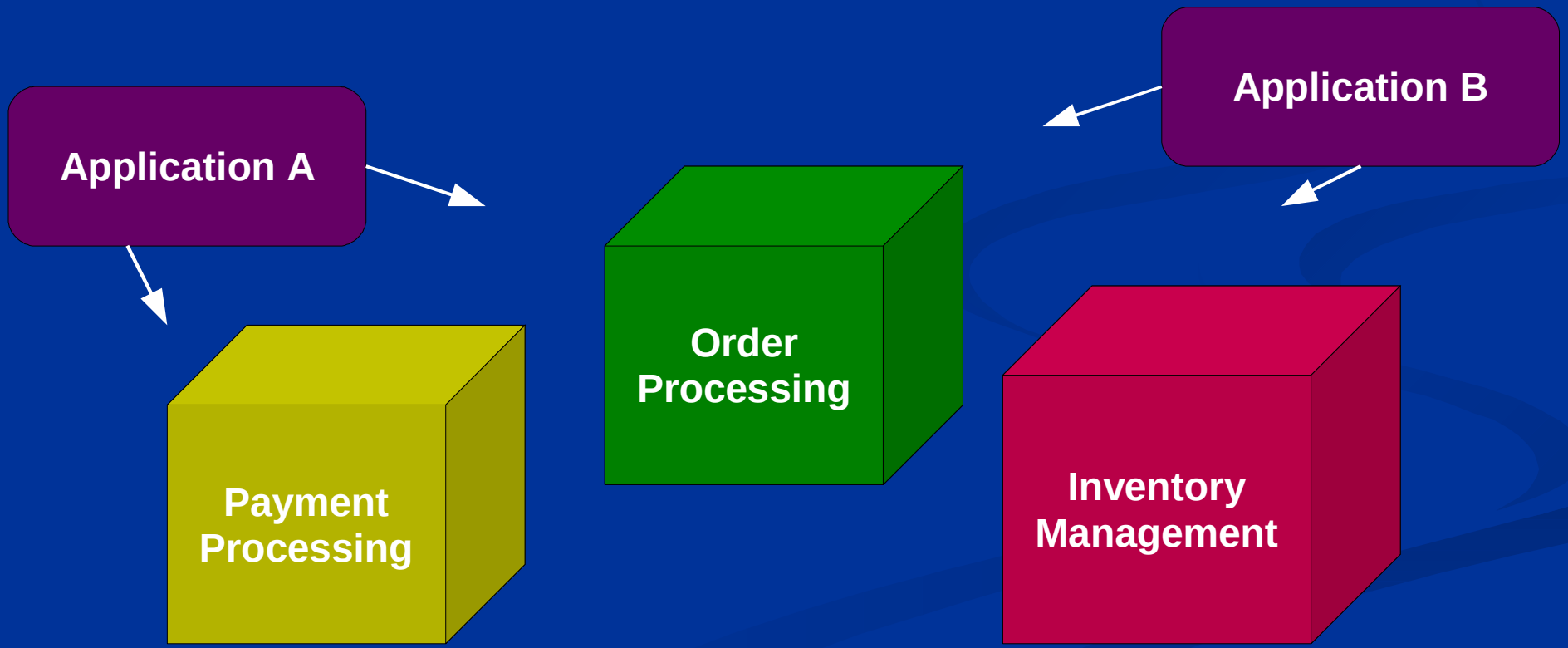
SOA Facts
www.soafacts.com

- Guns don't kill people, the SOA WS-* stack kills people.
- SOA is being used in the developing world to solve hunger. Entire populations will be fed on future business value.
- Not content to just best sliced bread, SOA is actually the best thing since beer, wine, coffee, ice cream, chocolate... oh, and sliced bread.
- Dante has a special level in hell for consultants whose resumes do not say SOA.
- SOA - building contractor jobs, one Visio slide at a time.
- The Answer to the Ultimate Question about Life, the Universe, and Everything is ..... SOA

What is SOA?

Simplified and Generalized Answer

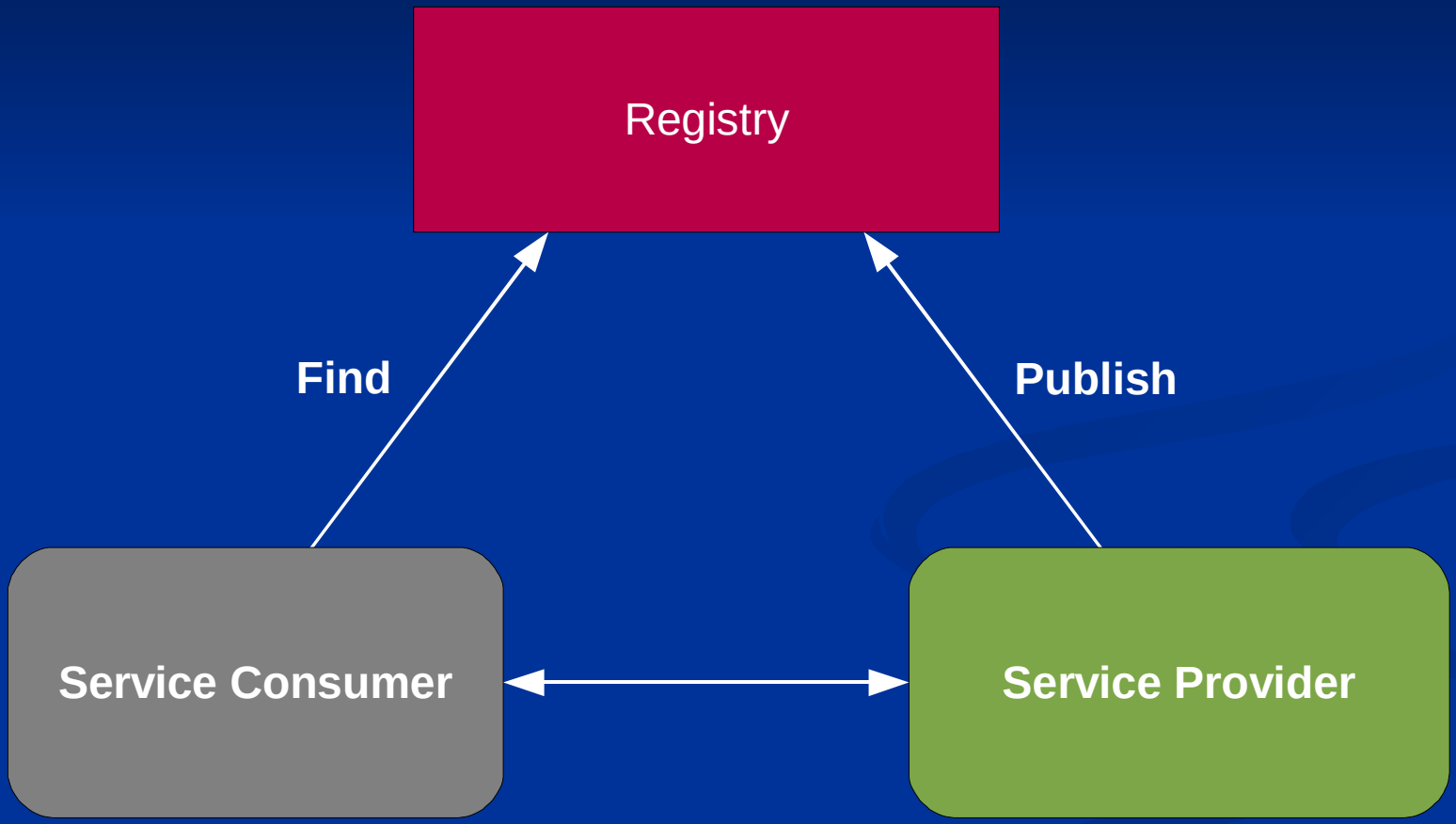
Service-oriented architecture is a style of building applications based on independent and re-usable building blocks that provide some specific functionality.



Building Blocks are Services

- A collection of related endpoints
- Loosely Coupled
- Well Defined Interfaces
- Interface Granularity
- Reusable
- Discoverable

SOA Model



Common Technologies

- SOAP
- Web Service Definition Language (WSDL)
- Universal Description Discovery and Integration (UDDI)
- WS-?
  - WS-Security
  - WS-Address
  - WS-xxxxxx

May often hear this referred to a WS-*

SOA and REST

SOA

Order Processing

getOrder(o_id)
getOrders(c_id)
addOrder(c_id, Order)
updateOrder(Order)

Customer Management

getCustomer(c_id)
getCustomers()
newCustomer(C)
updateCustomer(C)

REST

/orders

/orders/{o_id}

/customers

/customers/{c_id}

/customers/{c_id}/orders

GET
PUT
POST
DELETE

SOA: More than WS-*

- Many definitions of SOA do not preclude REST
- Different jobs require different tools
- Common Goals
  - Distributed
  - Interoperability
  - Reusability
  - Loosely coupled

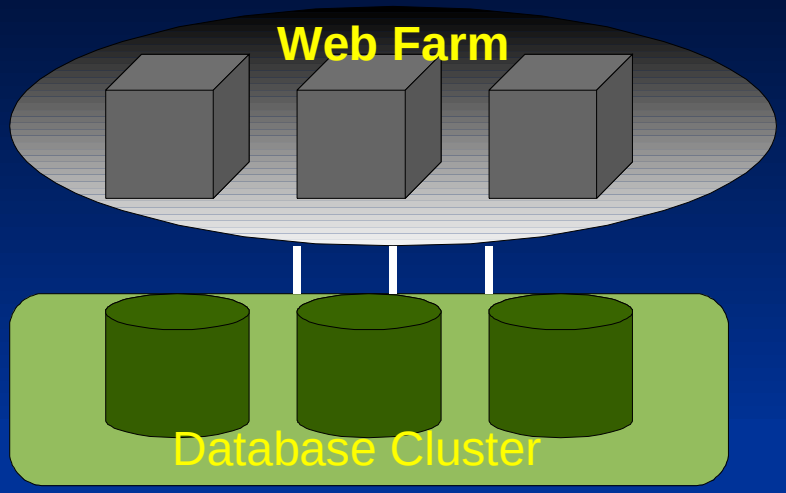
Arguments for WS-* Stack

- Existing architecture
- Legacy System Integration
- Granular Security Requirements
- Two-phase commits
- Asynchronous Messaging
- Target Consumers

Implementing SOA
A Continuous Cycle

- Clearly define why you are going to implement SOA
  - Everyone else is doing it so why shouldn't we?
  - Tangible reasons that would be beneficial to the company
- All levels of the organization must see the benefits
  - Impacts many areas of a company from finance to operations
  - Unless everyone buys into the idea it will fail from the start
- Assess and evaluate the current infrastructure
  - Visualize the infrastructure by areas of functionality
  - Are there areas that will benefit the most?
- Make a Plan
  - Clearly identify what and how you plan on implementing
  - DO NOT START OFF TO BIG!

Initial Architecture



Hosting Facility

DSL

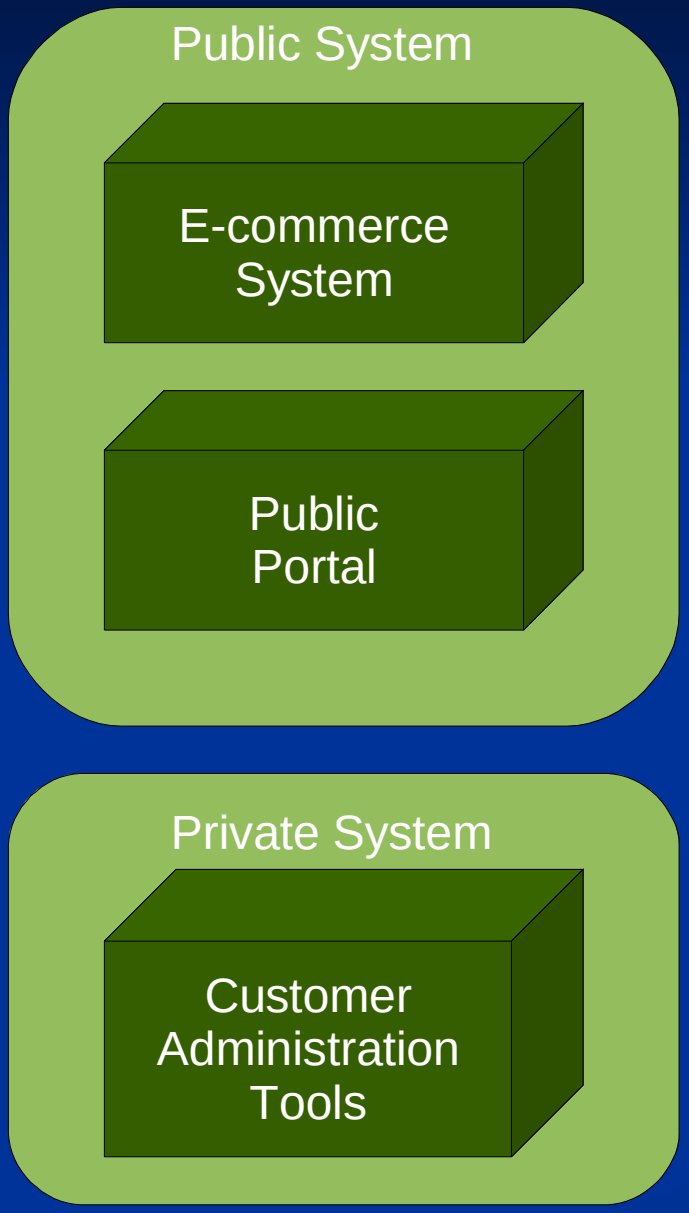
Local Office

Financial Application

Customer Relationship Application

Salesperson Management Application

Initial Software Architecture

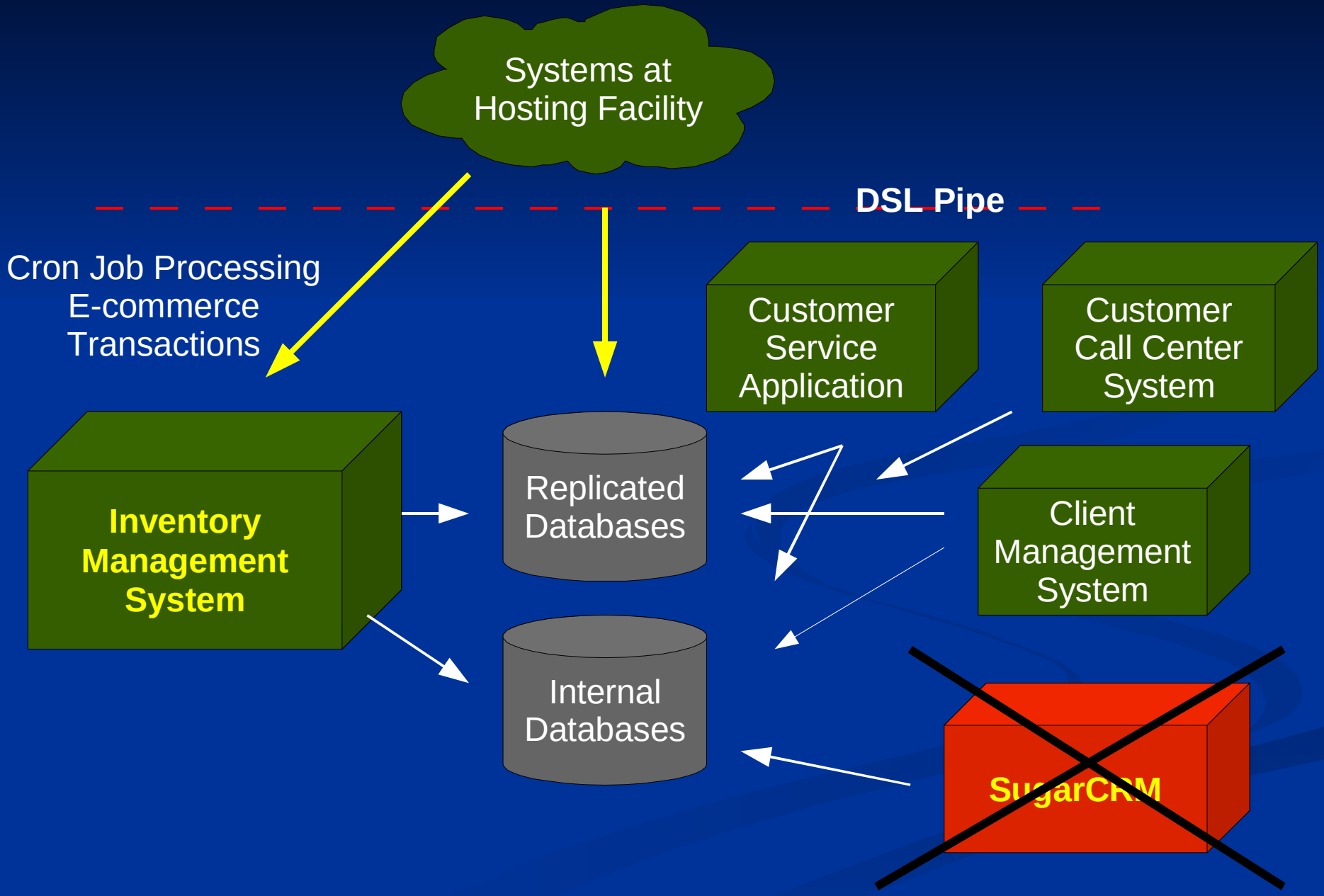


- Tightly coupled
- Re-Usability
  - Object Oriented coding
  - Organized procedural code
- Agility based on how fast we can code and get things tested
- Local office used third party stand alone applications

The Growing Pains

- Customer base is growing
- Organization is growing
- Existing infrastructure no longer meets company needs
  - Need more robust Customer Management system
  - Need more robust SalesPerson Management system
  - Business requires tighter integration into public web applications
- Keep expenses down
- **Time to market is critical!**

Git 'Er Done!


Systems at
Hosting Facility
DSL Pipe
Cron Job Processing
E-commerce
Transactions
Customer
Service
Application
Customer
Call Center
System
Inventory
Management
System
Replicated
Databases
Client
Management
System
Internal
Databases
SugarCRM

So Where's the SOA?

- Potential Disadvantages
  - We now have duplicate code
  - Multiple systems performing similar functionality
  - Potential higher maintenance costs
  - More systems to administer
- Potential Advantages
  - Systems were rolled out quick
  - Minimal cost to develop applications
  - No impact on rest of organization

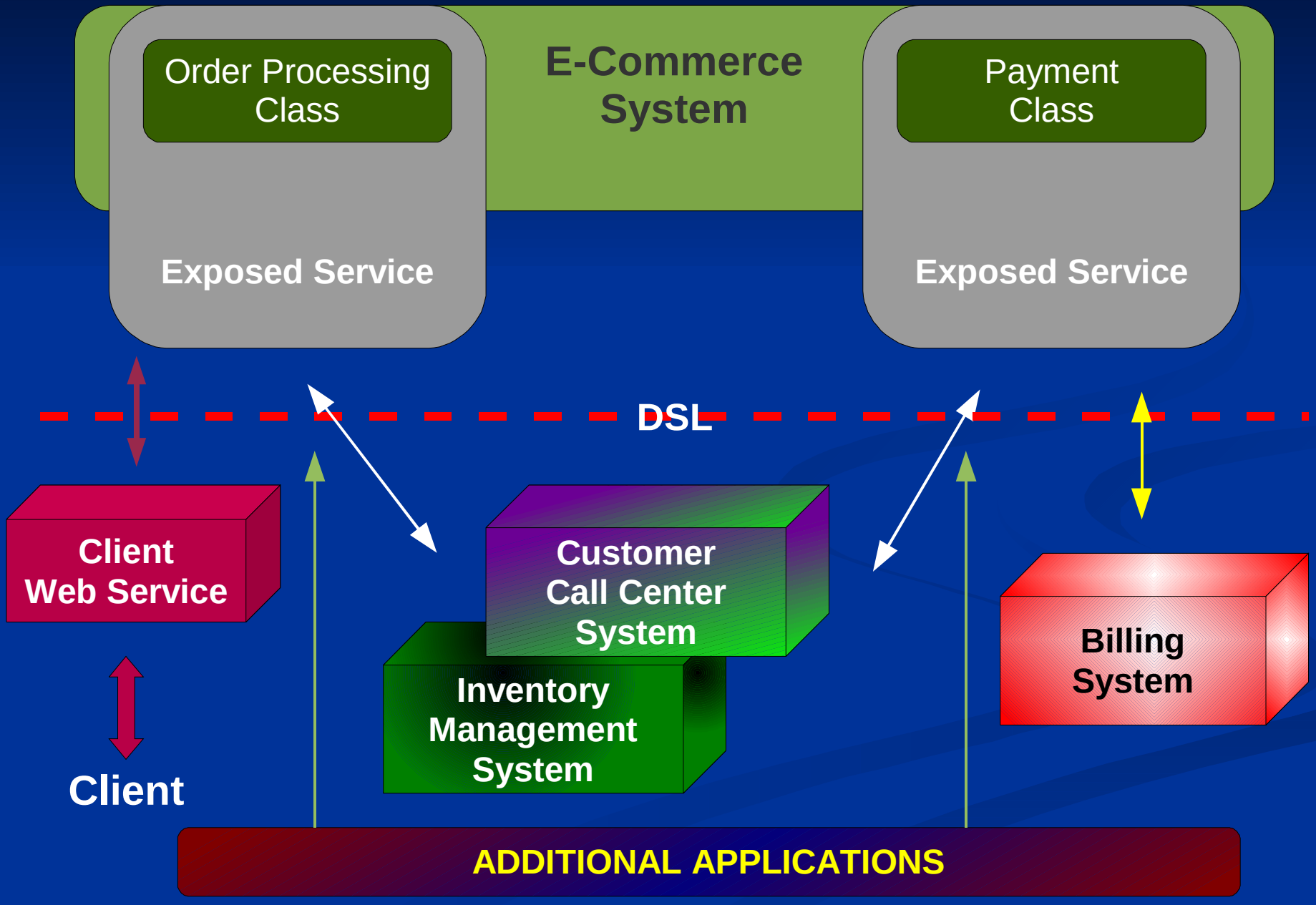
Things to Consider

- Justification or Mandate?
- Existing problems or challenges?
- Is it cost effective?
- It is time effective?
- Impact on current projects and applications
- Plan before you execute
  - What do you expect to achieve?
  - Not everything should be a service

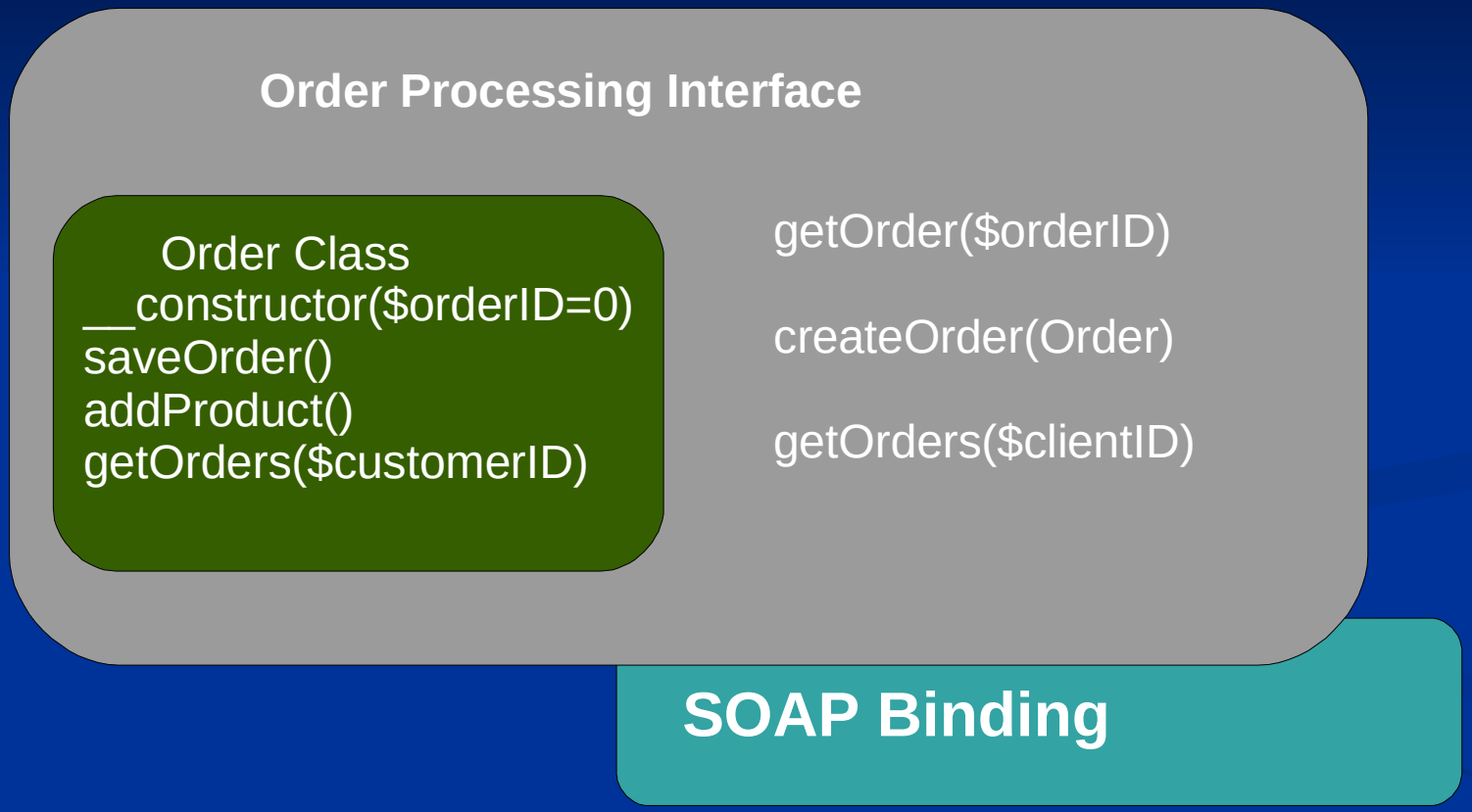
Finally implementing SOA

- Order Processing
  - Order information must be exposed to our clients
  - Orders must be accessible from a number of different systems
  - 100% uptime is critical for public E-commerce system
- Payment Processing
  - Multiple applications process payments
  - Need to be able to globally swap out underlying processor being used
  - 100% uptime is critical for public E-commerce system

The Architecture



Order Processing Service



Look before you Leap!

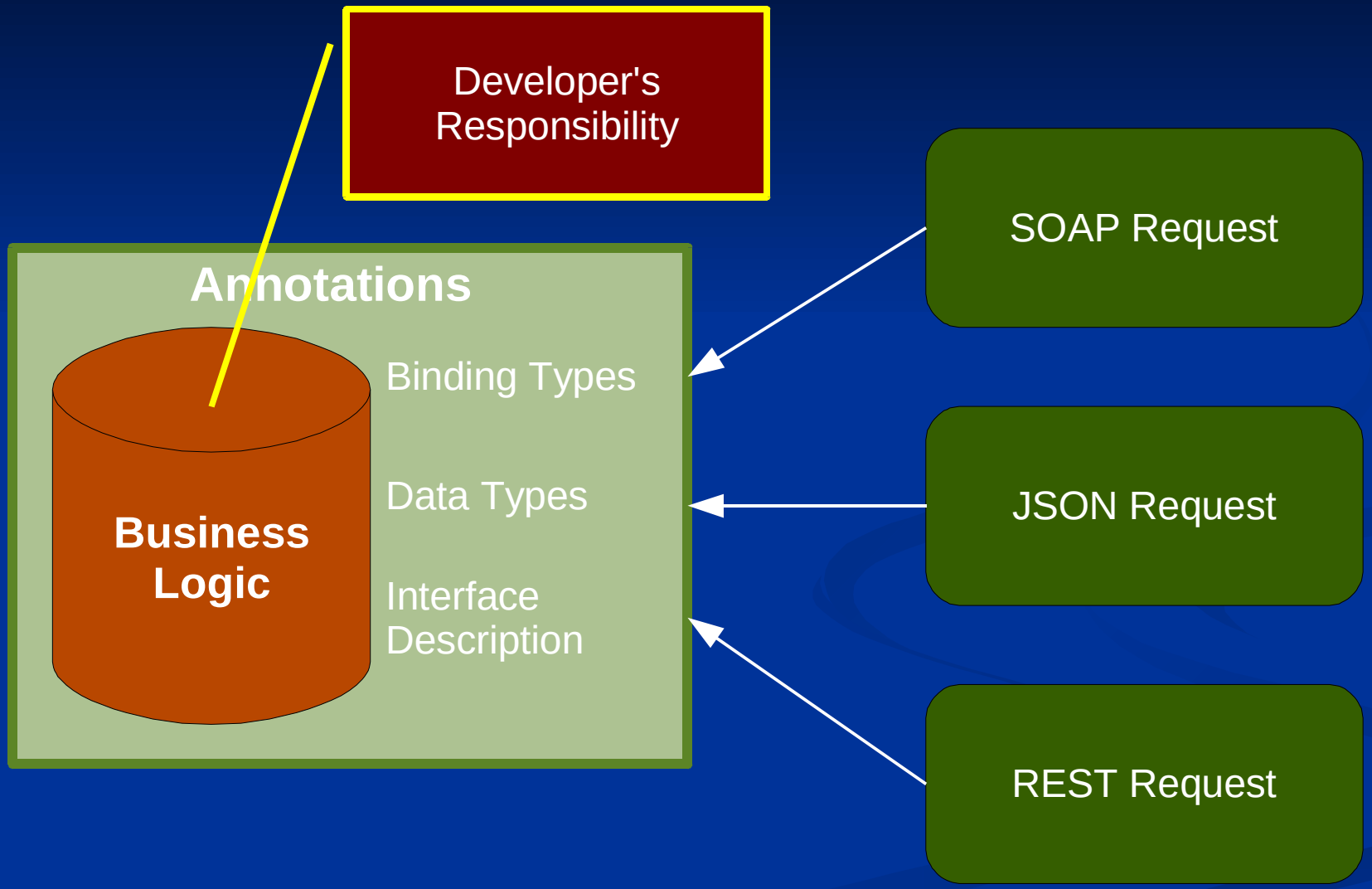
- Is the service critical to the business?
  - What happens in the event of network problems?
  - What happens in the event of hardware problems?
- Are there any security issues?
  - Is HTTPS sufficient?
  - Does WS-Security need to be leveraged?
  - Are there any other authentication factors to consider?
- How do you keep track of all the services?
  - Are they tracked manually?
  - How do you know which ones are available?
  - Do you implement a UDDI Registry?

Service Component Architecture
SCA

Service Component Architecture (SCA)

- Project by the Open Service Oriented Architecture (OSOA) collaboration - http://www.osoa.org/display/Main/Home
- Allows the developer to concentrate on the business logic rather than how it is all connected together
- http://www.osoa.org/display/PHP/SOA+PHP+Homepage
- Combined with the SDO extension
- Pecl repository: http://pecl.php.net/package/SCA_SDO
- Still in development

SCA


Developer's Responsibility
SOAP Request
Annotations
Binding Types
Data Types
Business Logic
Interface Description
JSON Request
REST Request

SCA Component

```
include "SCA/SCA.php";

/**
 * @service
 * @binding.ws
 * @binding.jsonrpc
 * @binding.rest.rpc
 */

class RobsService {

    /** Find out who I am.
     * @return string Who I am.
     */
    public function whoAmI()
    {
        return "I am Rob Richards";
    }
}
```

Calling the Component

```
include "SCA/SCA.php";
$url = 'http://localhost/robsService.php';

/* Use SOAP Binding */
$robservice = SCA::getService($url.'?wsdl');
$whoami = $robservice->whoAmI();

/* Use JSON Binding */
$robservice = SCA::getService($url.'?smd');
$whoami = $robservice->whoAmI();

/* Use SOAP Binding */
$client = new SoapClient($url.'?wsdl');
$whoami = $client->whoAmI();

/* Use REST Binding */
$whoami = file_get_contents($url.'/whoAmI');

/* Use Local Binding */
$robservice = SCA::getService('/home/rrichards/robweb/robsService.php');
$whoami = $robservice->whoAmI();
```

Implementing SOA
A Continuous Cycle (DON'T FORGET)

- Clearly define why you are going to implement SOA
  - Everyone else is doing it so why shouldn't we?
  - Tangible reasons that would be beneficial to the company
- All levels of the organization must see the benefits
  - Impacts many areas of a company from finance to operations
  - Unless everyone buys into the idea it will fail from the start
- Assess and evaluate the current infrastructure
  - Visualize the infrastructure by areas of functionality
  - Are there areas that will benefit the most?
- Make a Plan
  - Clearly identify what and how you plan on implementing
  - DO NOT START OFF TO BIG!

QUESTIONS?

Web Services and SOA

Rob Richards
http://xri.net/=rob.richards