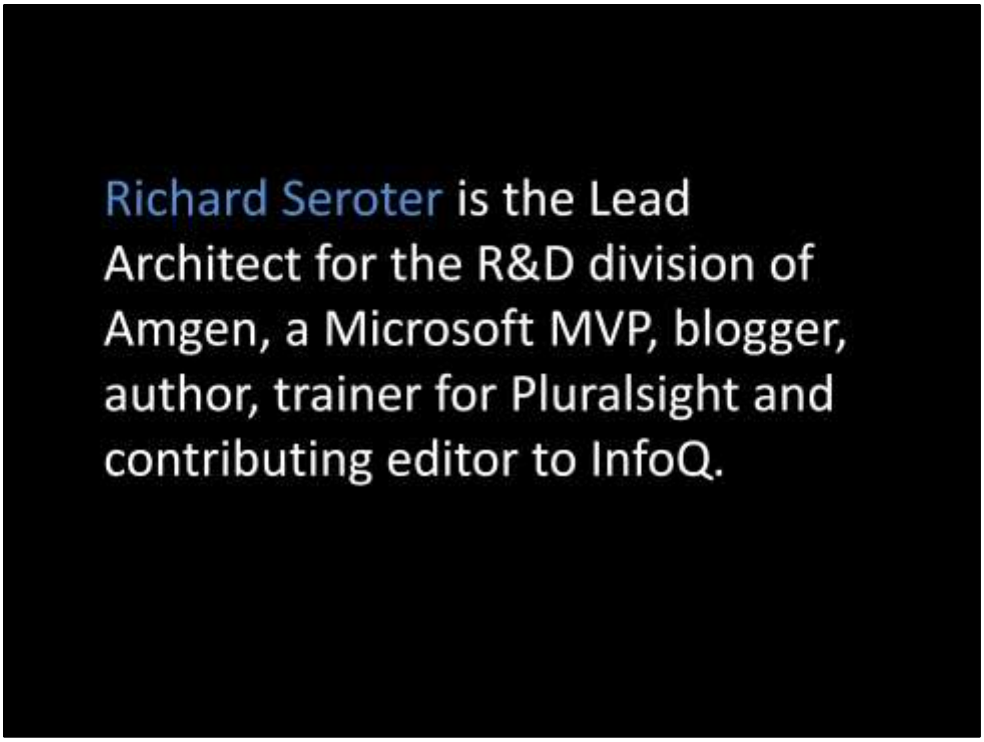
Integrating the Cloud: Patterns for Success

Richard Seroter
Principal Architect, Amgen
Twitter: @rseroter
Blog: seroter.wordpress.com

Thanks for having me …

For me … some interesting cloud integration suites to investigate further …
http://www.snaplogic.com/what-we-do/integration-platform/snapcenter.php
http://integration.pervasive.com/UsageScenarios/ApplicationIntegration/CloudIntegration.aspx
http://www.boomi.com/resources/whitepapers



Richard Seroter is the Lead Architect for the R&D division of Amgen, a Microsoft MVP, blogger, author, trainer for Pluralsight and contributing editor to InfoQ.

Who I am …

Cloud computing is exploding as a software delivery style.

[Where are we]

- Increased popularity towards using packaged software as a service or building apps in cloud platforms
- More than 95 percent of organizations expect to maintain or increase their investments in software as a service (SaaS) and more than one-third have migration projects under way from on-premises to SaaS, according to a survey by Gartner, Inc.
    - Currently, communications (52 percent), utilities (51 percent), and banking and securities (49 percent) industries rank highest with respect to SaaS deployed across the horizontal and vertical-specific categories sampled. In 2012, those industries ranking highest with respect to their plans to use SaaS include federal government (33 percent), banking and securities (22 percent) and wholesale trade (20 percent). Beyond 2012, top industries considering SaaS are manufacturing and natural resources (37 percent), wholesale trade and retail (each 29 percent).

There is the risk of creating new application silos.

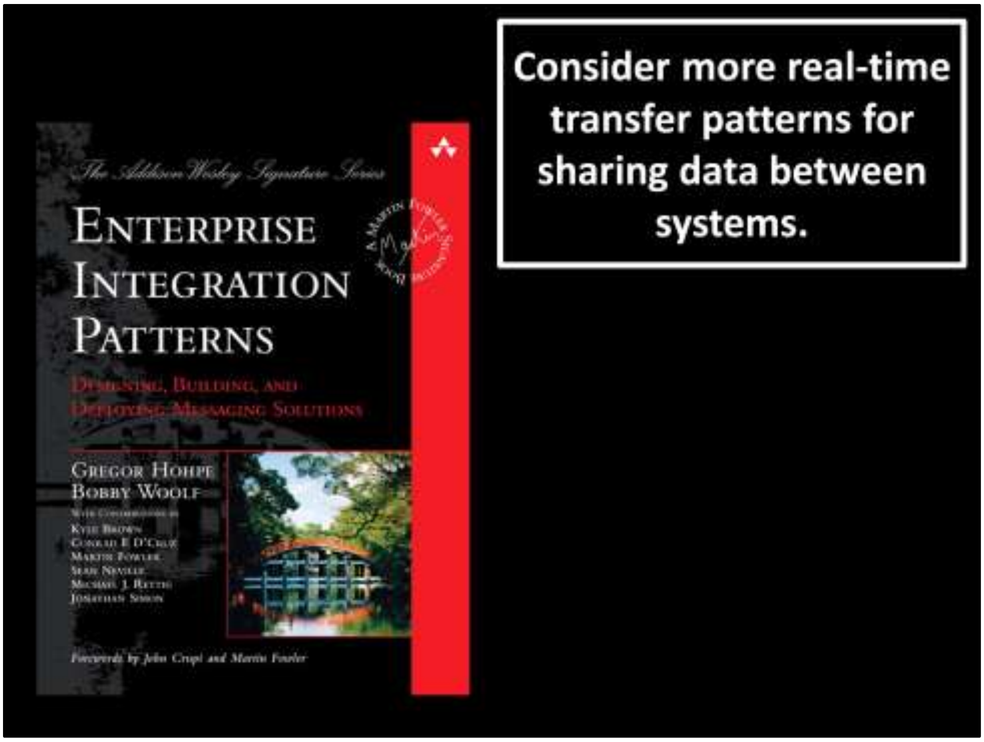
[What’s the problem?]

- SaaS not really built with data exchange in mind
  - Information Week survey (http://www.informationweek.com/news/cloud-computing/software/231002362?queryText=SaaS+Integration) of SaaS satisfaction areas showed “ease of deployment” #1 and integration last
- Easy to see these as “off site” and either do limited integration, or, irregular batch loads
- Three scenarios: on-premise to cloud, cloud to on-premise, and cloud to cloud
  - Each have unique challenges, but each are valuable and necessary scenarios

We want to be able to apply proven integration patterns to cloud scenarios.

[where do we want to be?]

- There are tried and true integration styles for sharing data/processes between systems
- We’re taking that reliable chassis and applying it to new technologies


The Addison-Wesley Signature Series
Enterprise Integration Patterns
Designing, Building, and Deploying Messaging Solutions
Gregor Hohpe
Bobby Woolf
With Contributions by
Kyle Brown
Conrad F. D'Cruz
Martin Fowler
Sean Neville
Michael J. Rettig
Jonathan Simon
Forewords by John Crupi and Martin Fowler
A Martin Fowler Signature Book

Consider more real-time transfer patterns for sharing data between systems.

[how do we get there?]
A seminal book on the topic is the EIP book by Hohpe and Woolf
Discusses a handful of communication styles and integration patterns. We'll focus on three that are called out there:

- Shared database
- Remote procedure invocation
- Messaging / async

For each pattern …

- Define it
- When it makes sense
- Constraints
- Cloud considerations
- Cloud technologies
- Demo

Use a Shared Database when data needs to be available instantly and consistently.

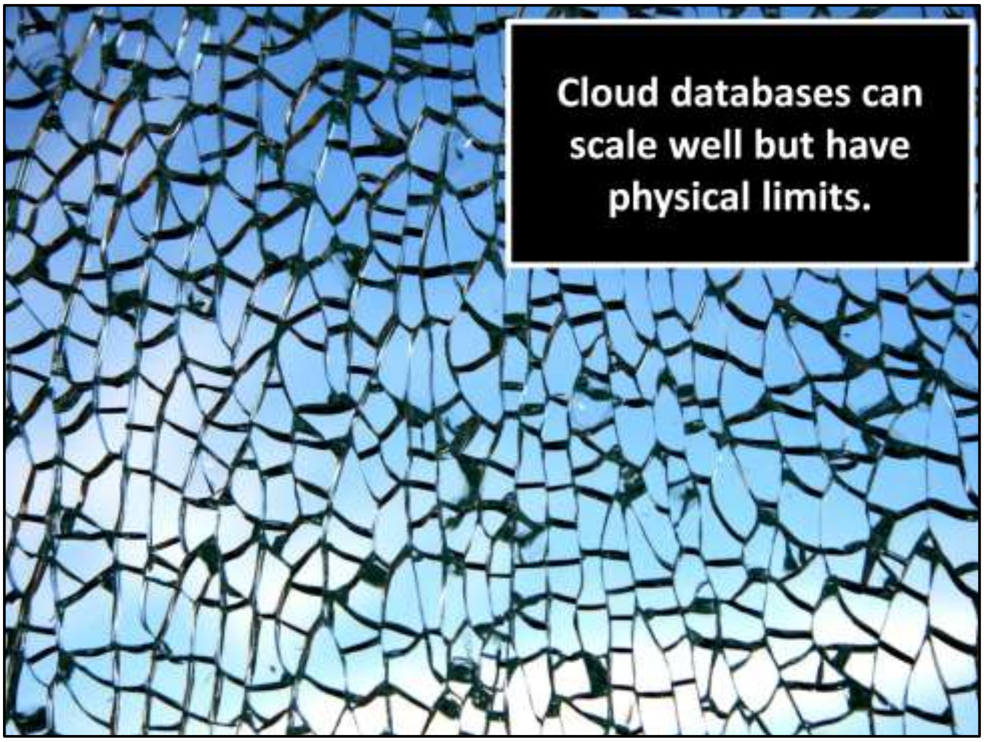
When it makes sense …

- Sharing via ETL or file isn't timely
- What if you need the latest data?
- Could be for reporting, reference data or transactions
- Use when you have common data (and/or structure) but different CONSUMING interfaces
  - Think of multi-tenant apps where you share a DB, but partition by user
  - Could be an ODS or data mart where you want common schema used by others
- Get a single view of the data (no need to force each end to define a data format)
- All dependent systems are consistent at the same time
- Single data access strategy (SQL syntax)

We can expect challenges with data consistency and schema agreement.

Challenges

- ** Design **
- Tough to agree on formats (for ODS, mart, transactional records)
  - Unified schema that satisfies everyone? Commodity only, or, strong leadership to form a standard (or extensibility)
- ** Contention / Performance **
- Less likely to use when have multiple apps manipulating same transactional data
  - Could have data records defined where you could change parts of one record while someone changes parts of others
- May get fewer inconsistencies, but still have issues of simultaneous updates
  - Can try to handle with transactions, but transactions are often the enemy of scalability
  - http://www.ics.uci.edu/~cs223/papers/cidr07p15.pdf
- Try to avoid updates / conflicts between applications sharing the data
  - New rows only, or read only (reporting scenario, or reference data)
  - Avoid deadlocks
- Could get poor performance if apps are distributed and all accessing over WAN
- ** COTS support **
- Packaged apps rarely accept an external database as its source
- Opposite of "Shared nothing" where nodes are self sufficient http://en.wikipedia.org/wiki/Shared_nothing_architecture
  - May mean sharding http://www.codefutures.com/database-sharding/

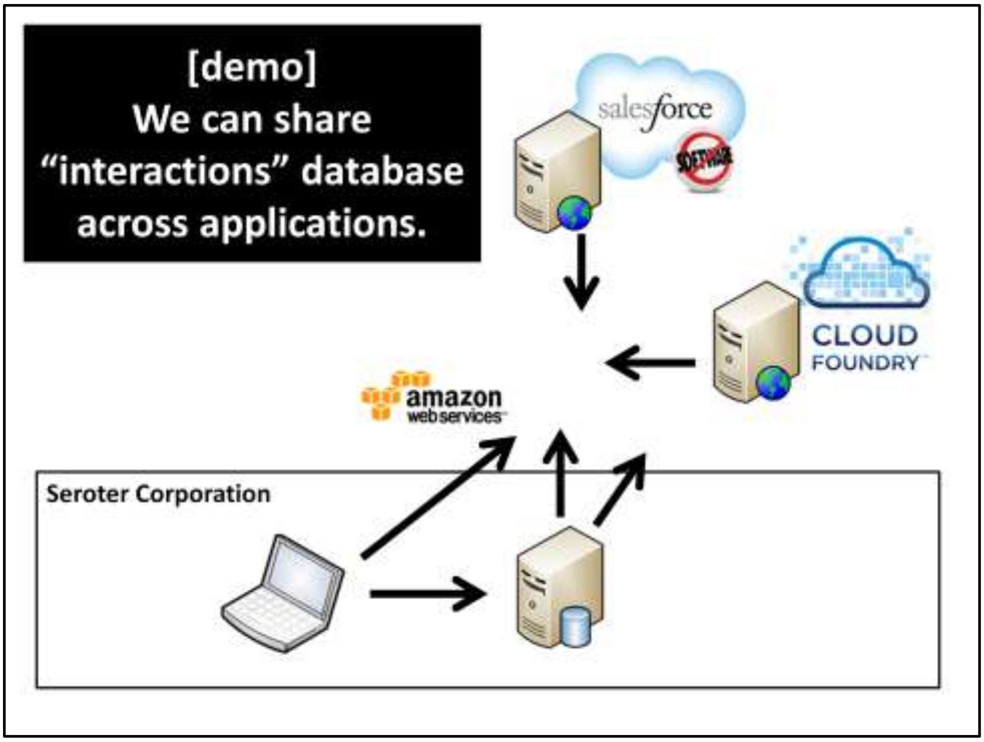


Cloud databases can scale well but have physical limits.

Cloud Considerations

- ** Access protocols **
- Cloud provides either DB or web protocol access
- Have choices as to how to access
  - If remediating existing apps, may want the option of using “standard” ODBC APIs and not have to change much code
  - For newer (or rebuilt) apps, many services provide a RESTful API for data access
- ** Identity **
- Identity providers
- If shared, how do you apply granular access?
- ** Performance **
- May have good performance
  - Sharding built in to most of these (split based on domains/groups)
  - In some cases have eventual consistent reads, but SimpleDB supports option to have consistent reads http://aws.amazon.com/articles/3572?_encoding=UTF8&jiveRedirect=1
  - Conditional PUT and DELETE with expected values (to prevent overwriting fresher data)
    - Can still do eventually consistent read and trust the Conditionals to enforce consistency
    - Could use version or timestamp value as part of Conditional

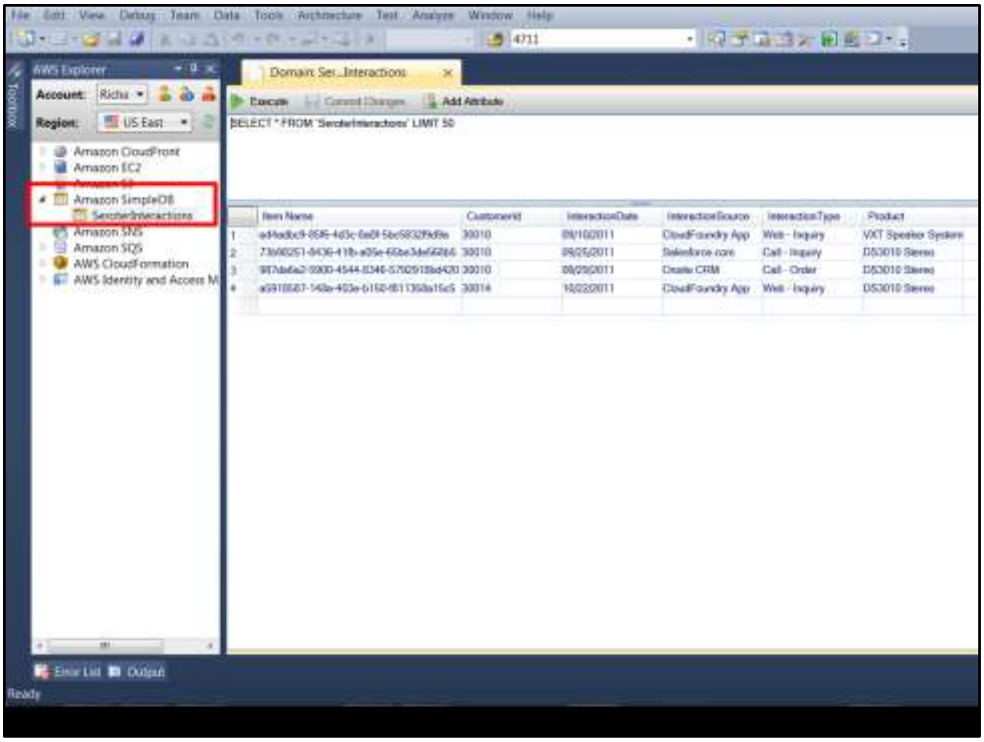
- ** Different DB storage options **
- Could use cloud RDMS if you don’t have massive scale needs
- Can use a schema-less product like SimpleDB
- ** Provider limits **
- Writes are throttled to SimpleDB, so can use sharded domains if expect more than XYZ puts/second (do batch)
- http://practicalcloudcomputing.com/post/712653349/simpledb-essentials-for-high-performance-users-part-1
- May have limited transactions
    - Across tables/entities for schema-less like Azure tables
- ** Options **
- DBs
    - Structured RDMS
        - Amazon RDS
        - SQL Azure
        - Database.com
        - Database hosted on IaaS platform like AWS
    - Schema-less
        - SimpleDB
        - Azure Tables
        - Google AppEngine Data Store http://code.google.com/appengine/docs/python/datastore/overview.html
            - Optimistic concurrency
            - Distributes data when necessary
            - Limits in number of calls per minute
    - Blobs
        - S3
        - Azure Blob Storage



[demo]
We can share "interactions" database across applications.

Demo

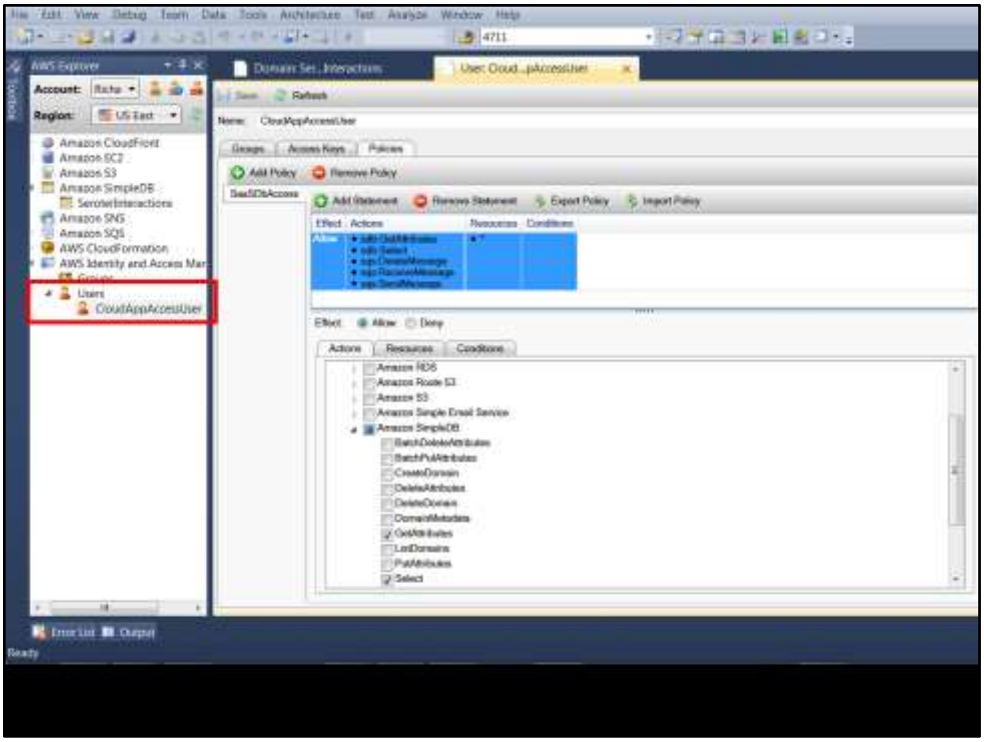
- Slide shows "before and after"
- Demo start …
  - Show db values from VS 2010; Show IAM and credentials specific to a user account
- .NET app
  - Retrieve items; Open app and don't put customer ID in; then put just the 30010 account in
- Cloud Foundry
  - Ruby app where I query the database via REST after building signed string
  - http://seroter-cloudintegration.cloudfoundry.com/lookup/30010
  - http://seroter-cloudintegration.cloudfoundry.com/lookup/30014
  - Show Ruby class; VMC to show running app instances
- SFDC
  - Custom code that looks at shared DB
  - https://c.na11.visual.force.com/apex/InteractionHistory?id=003A0000001YNPe (30010)
  - https://c.na11.visual.force.com/apex/InteractionHistory?id=003A0000001YycD (30014)
  - Show SFDC code
- .NET app
  - Add new item; Refresh SDFC and Cloud Foundry apps



```
SELECT * FROM `SeroterInteractions` LIMIT 50
```

	Item Name	Customerid	InteractionDate	InteractionSource	InteractionType	Product
1	ed4adbc9-8686-4d3c-8a0f-5bc5832f9d8a	30010	09/10/2011	CloudFoundry App	Web - Inquiry	VXT Speaker System
2	73b00251-8436-41fb-a05e-65be3de562b6	30010	09/25/2011	Salesforce.com	Call - Inquiry	DS3010 Stereo
3	987da6a2-0900-4544-8346-5792918bd420	30010	09/29/2011	Create CRM	Call - Order	DS3010 Stereo
4	a5910587-140a-403e-b150-f611358a15c5	30014	10/22/2011	CloudFoundry App	Web - Inquiry	DS3010 Stereo

Can view my Amazon Web Services (AWS) SimpleDB content via Visual Studio 2010 plugin. Notice that I have four rows of data for two different customers.



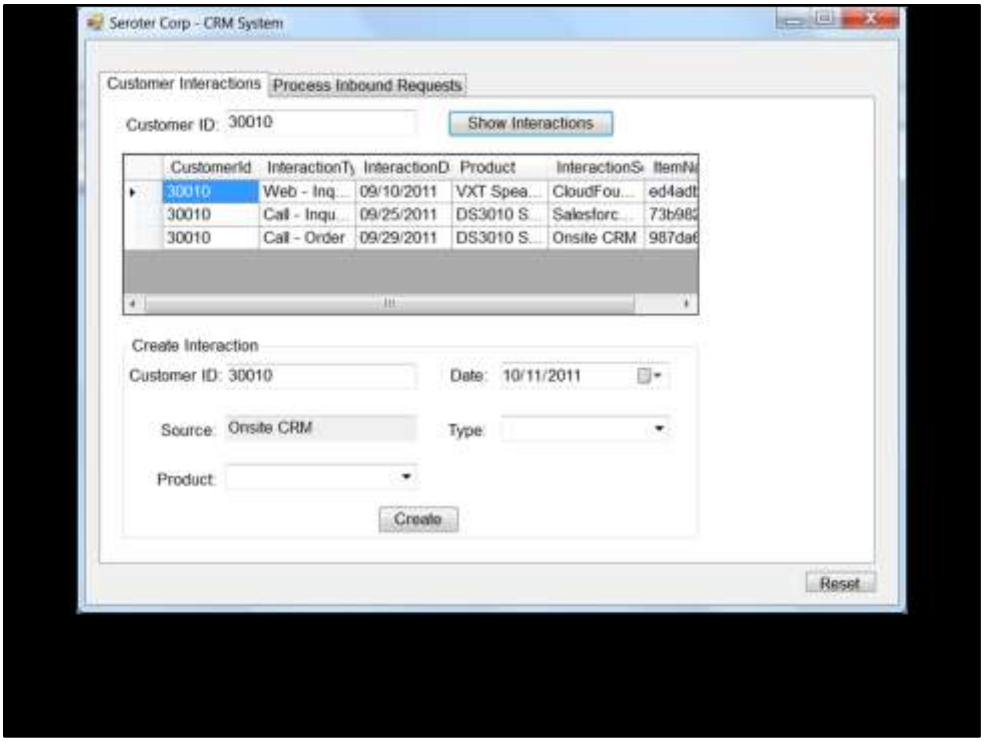
Name: CloudAppAccessUser

Effect	Actions	Resources	Conditions
Allow	• sdb:GetAttributes • sdb:Select • sqs:DeleteMessage • sqs:ReceiveMessage • sqs:SendMessage	• *	

Effect ● Allow ○ Deny

- □ Amazon RDS
- □ Amazon Route 53
- □ Amazon S3
- □ Amazon Simple Email Service
- ■ Amazon SimpleDB
  - □ BatchDeleteAttributes
  - □ BatchPutAttributes
  - □ CreateDomain
  - □ DeleteAttributes
  - □ DeleteDomain
  - □ DomainMetadata
  - ☑ GetAttributes
  - □ ListDomains
  - □ PutAttributes
  - ☑ Select

I can also do granular role-based permissions in AWS and restrict who can create/update/read data



Seroter Corp - CRM System

Customer ID: 30010

	CustomerId	InteractionTy	InteractionD	Product	InteractionS	ItemN
▸	30010	Web - Inq...	09/10/2011	VXT Spea...	CloudFou...	ed4adt
	30010	Call - Inqu...	09/25/2011	DS3010 S...	Salesforc...	73b98
	30010	Call - Order	09/29/2011	DS3010 S...	Onsite CRM	987da

Create Interaction

Customer ID: 30010

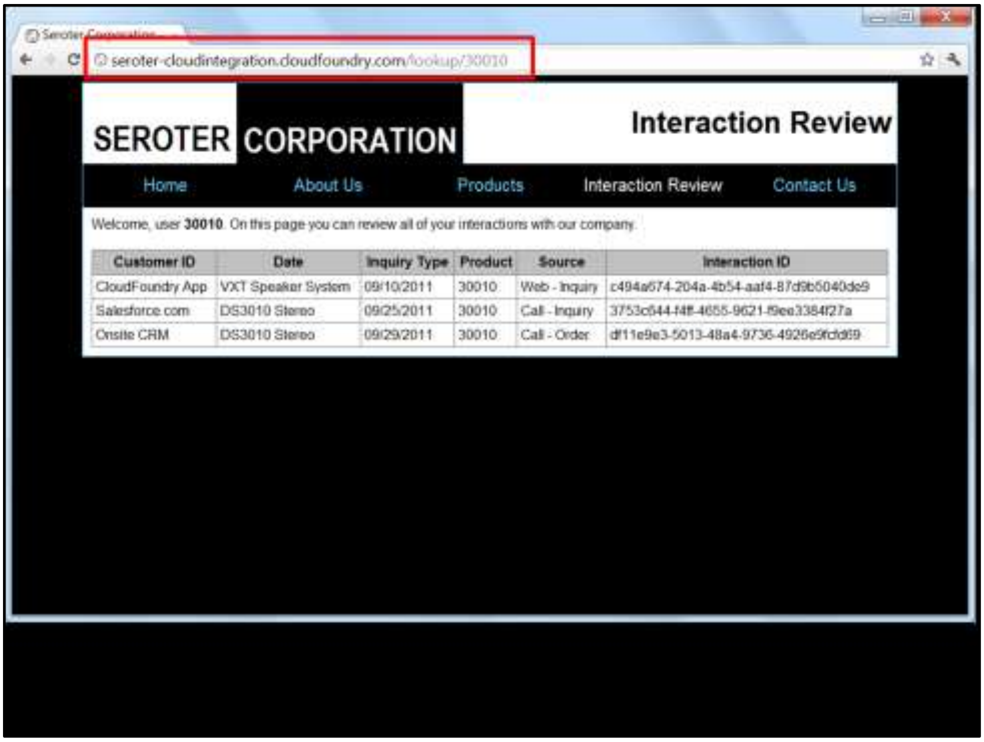
Date: 10/11/2011

Source: Onsite CRM

Type:

Product:

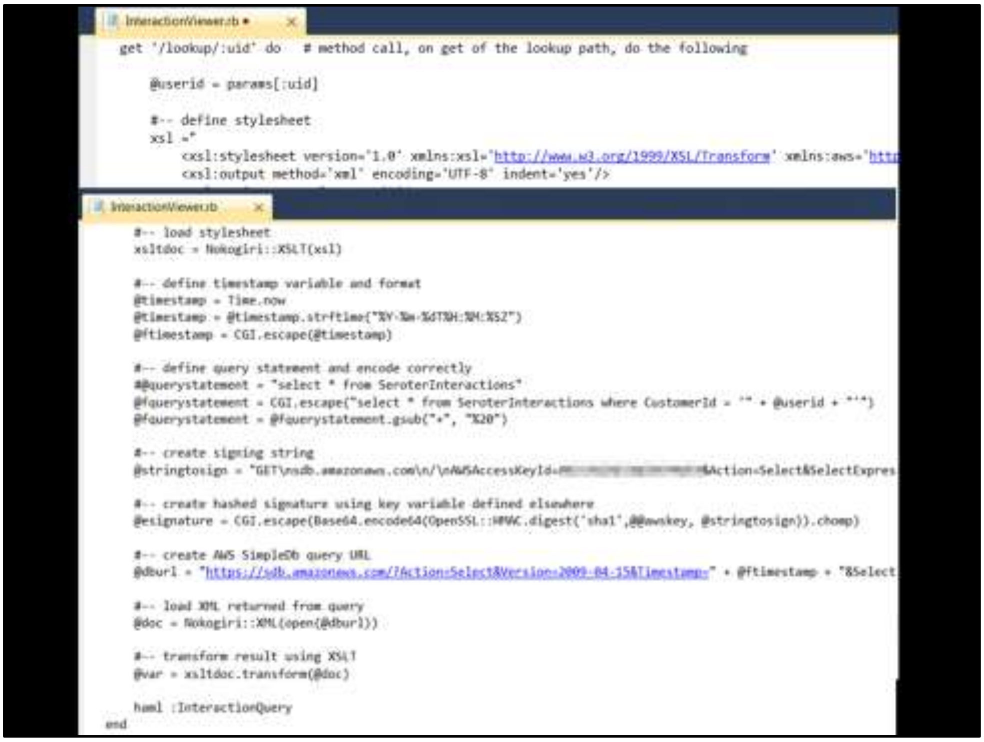
I built a client application that pulls all the interactions from AWS SimpleDB for a given user



Welcome, user **30010**. On this page you can review all of your interactions with our company.

Customer ID	Date	Inquiry Type	Product	Source	Interaction ID
CloudFoundry App	VXT Speaker System	09/10/2011	30010	Web - Inquiry	c494a674-204a-4b54-aaf4-87d9b5040de9
Salesforce.com	DS3010 Stereo	09/25/2011	30010	Call - Inquiry	3753c644-f4ff-4655-9621-f9ee3384f27a
Onsite CRM	DS3010 Stereo	09/29/2011	30010	Call - Order	df11e9e3-5013-48a4-9736-4926e9fcfd69

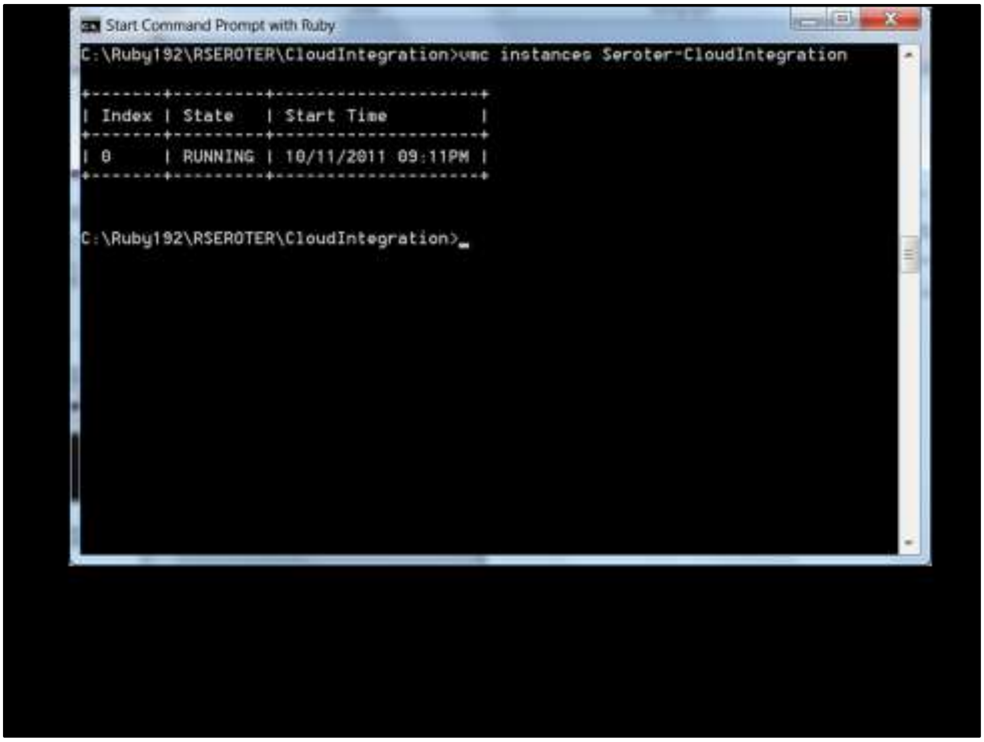
I’ve also built a Ruby web application hosted in VMWare’s public/private cloud called Cloud Foundry



```
get '/lookup/:uid' do   # method call, on get of the lookup path, do the following

    @userid = params[:uid]

    #-- define stylesheet
    xsl ="
        <xsl:stylesheet version='1.0' xmlns:xsl='http://www.w3.org/1999/XSL/Transform' xmlns:aws='http
        <xsl:output method='xml' encoding='UTF-8' indent='yes'/>
```

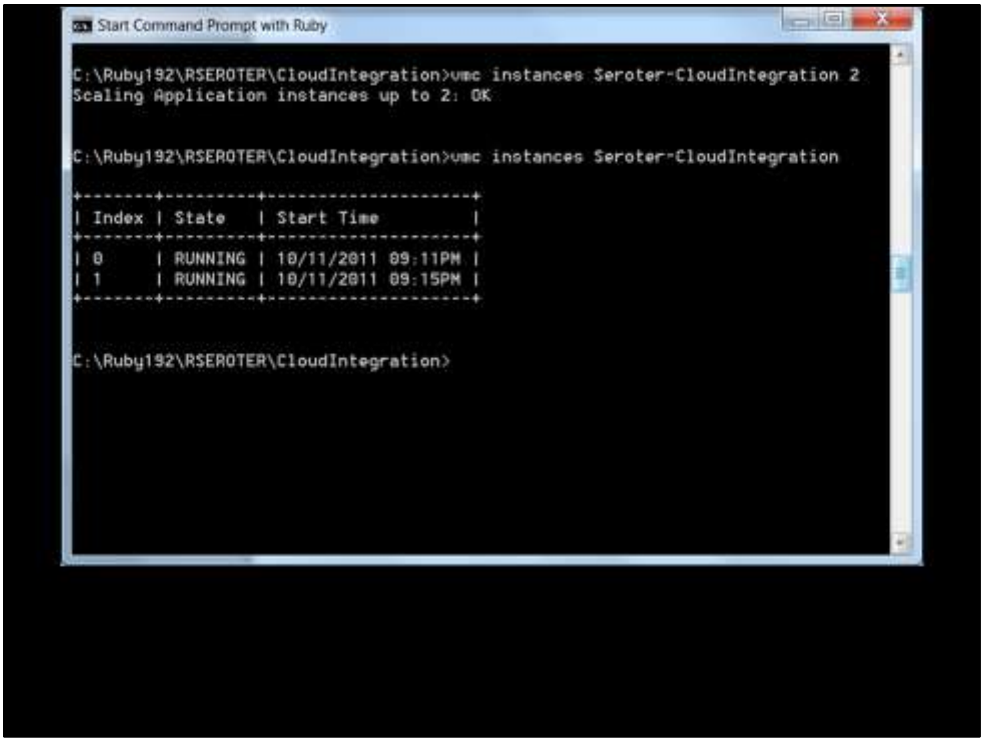
```
    #-- load stylesheet
    xsltdoc = Nokogiri::XSLT(xsl)

    #-- define timestamp variable and format
    @timestamp = Time.now
    @timestamp = @timestamp.strftime("%Y-%m-%dT%H:%M:%SZ")
    @ftimestamp = CGI.escape(@timestamp)

    #-- define query statement and encode correctly
    #@querystatement = "select * from SeroterInteractions"
    @fquerystatement = CGI.escape("select * from SeroterInteractions where CustomerId = '" + @userid + "'")
    @fquerystatement = @fquerystatement.gsub("+", "%20")

    #-- create signing string
    @stringtosign = "GET\nsdb.amazonaws.com\n/\nAWSAccessKeyId=[redacted]&Action=Select&SelectExpres

    #-- create hashed signature using key variable defined elsewhere
    @esignature = CGI.escape(Base64.encode64(OpenSSL::HMAC.digest('sha1',@@awskey, @stringtosign)).chomp)

    #-- create AWS SimpleDb query URL
    @dburl = "https://sdb.amazonaws.com/?Action=Select&Version=2009-04-15&Timestamp=" + @ftimestamp + "&Select

    #-- load XML returned from query
    @doc = Nokogiri::XML(open(@dburl))

    #-- transform result using XSLT
    @var = xsltdoc.transform(@doc)

    haml :InteractionQuery
end
```

My Cloud Foundry Ruby code consumes the AWS SimpleDB data via web services



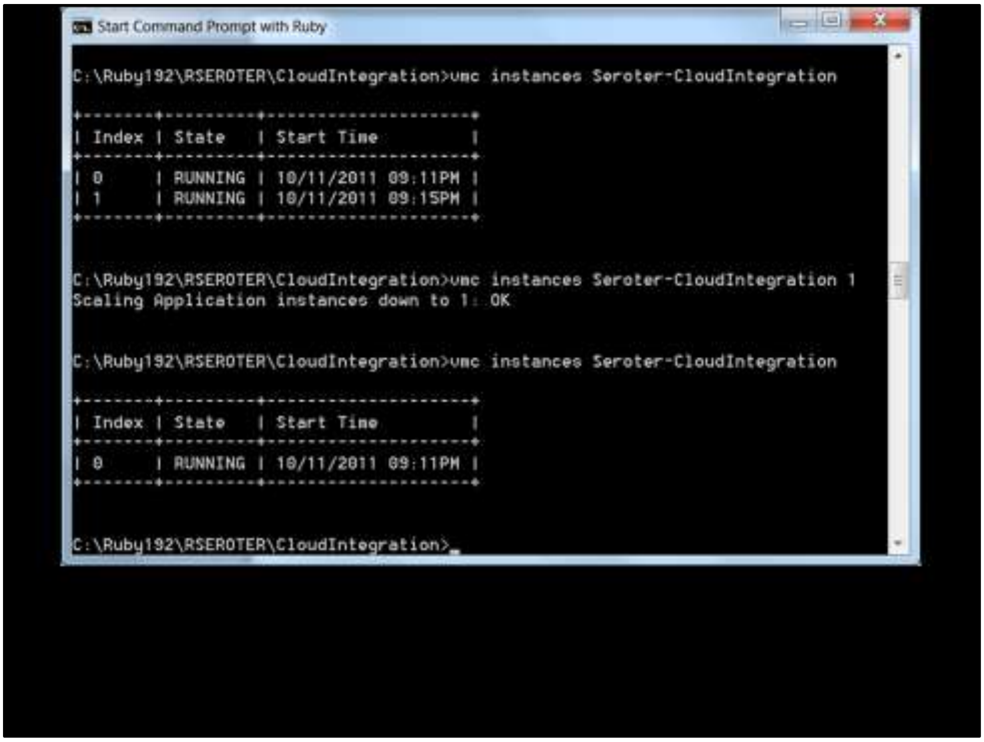
```
C:\Ruby192\RSEROTER\CloudIntegration>vmc instances Seroter-CloudIntegration

+-------+---------+--------------------+
| Index | State   | Start Time         |
+-------+---------+--------------------+
| 0     | RUNNING | 10/11/2011 09:11PM |
+-------+---------+--------------------+


C:\Ruby192\RSEROTER\CloudIntegration>_
```

You can see how many instances of my web app are deployed in the VMWare cloud



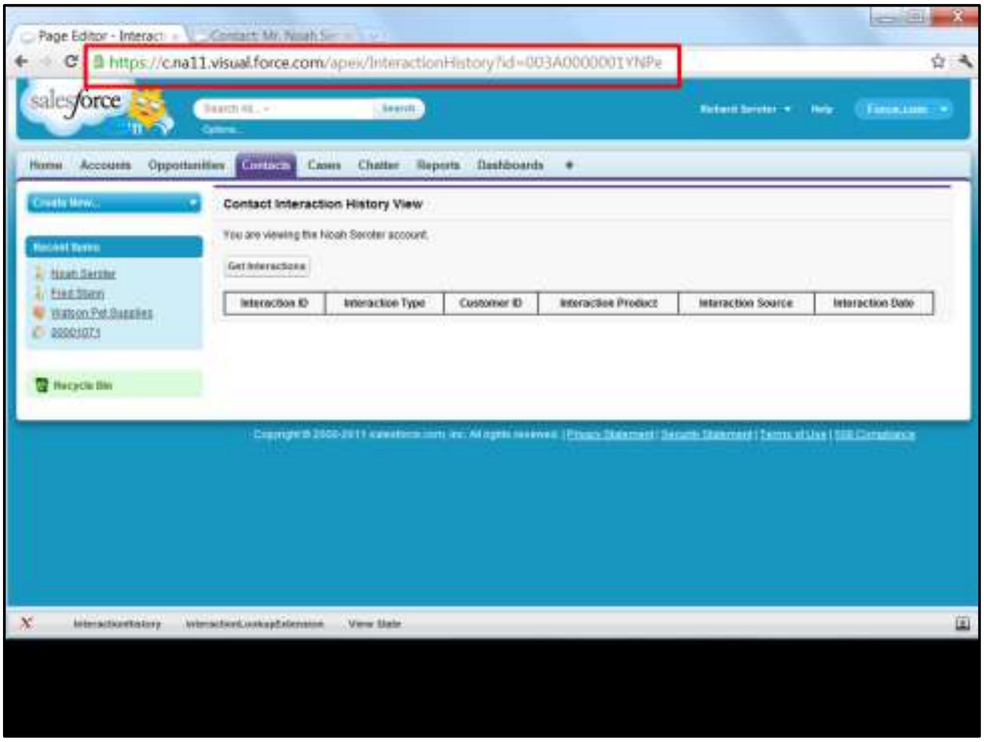
```
C:\Ruby192\RSEROTER\CloudIntegration>vmc instances Seroter-CloudIntegration 2
Scaling Application instances up to 2: OK


C:\Ruby192\RSEROTER\CloudIntegration>vmc instances Seroter-CloudIntegration

+-------+---------+--------------------+
| Index | State   | Start Time         |
+-------+---------+--------------------+
| 0     | RUNNING | 10/11/2011 09:11PM |
| 1     | RUNNING | 10/11/2011 09:15PM |
+-------+---------+--------------------+


C:\Ruby192\RSEROTER\CloudIntegration>
```

With one command, I *instantly* jump to two instances. Immediately load balanced.



```
C:\Ruby192\RSEROTER\CloudIntegration>vmc instances Seroter-CloudIntegration

+-------+---------+--------------------+
| Index | State   | Start Time         |
+-------+---------+--------------------+
| 0     | RUNNING | 10/11/2011 09:11PM |
| 1     | RUNNING | 10/11/2011 09:15PM |
+-------+---------+--------------------+


C:\Ruby192\RSEROTER\CloudIntegration>vmc instances Seroter-CloudIntegration 1
Scaling Application instances down to 1: OK

C:\Ruby192\RSEROTER\CloudIntegration>vmc instances Seroter-CloudIntegration

+-------+---------+--------------------+
| Index | State   | Start Time         |
+-------+---------+--------------------+
| 0     | RUNNING | 10/11/2011 09:11PM |
+-------+---------+--------------------+


C:\Ruby192\RSEROTER\CloudIntegration>_
```

I can then *instantly* move back to a single instance.



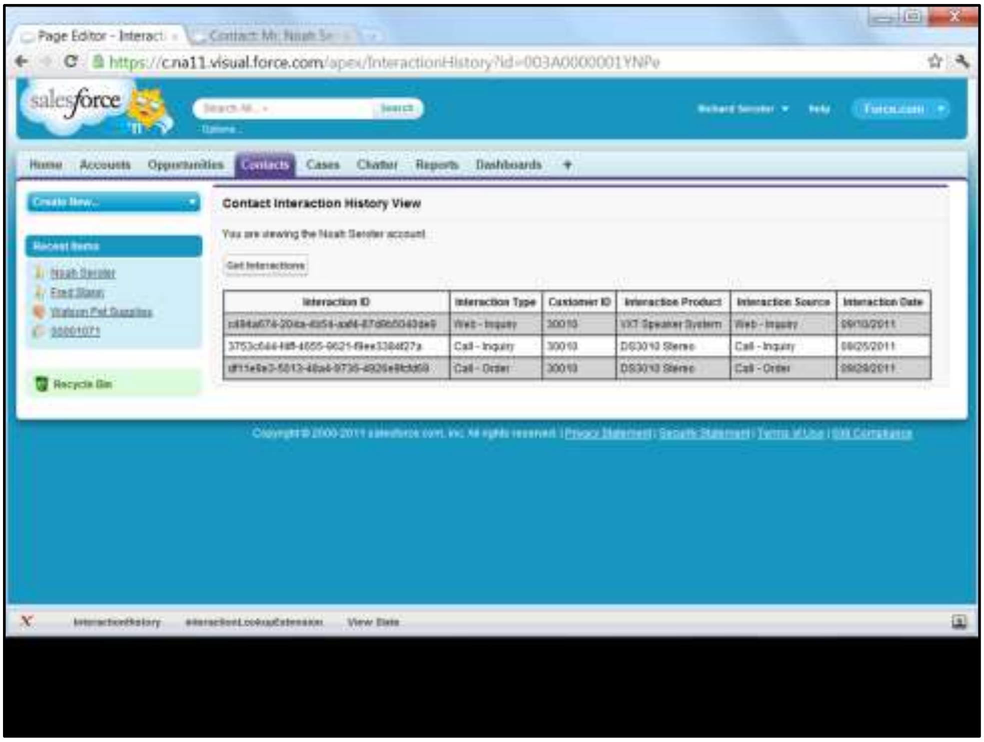
https://c.na11.visual.force.com/apex/InteractionHistory?id=003A0000001YNPe

Contact Interaction History View

You are viewing the Noah Serster account.

Interaction ID	Interaction Type	Customer ID	Interaction Product	Interaction Source	Interaction Date

I’m now in my Salesforce.com account where I built a custom page to pull SimpleDB data



Contact Interaction History View

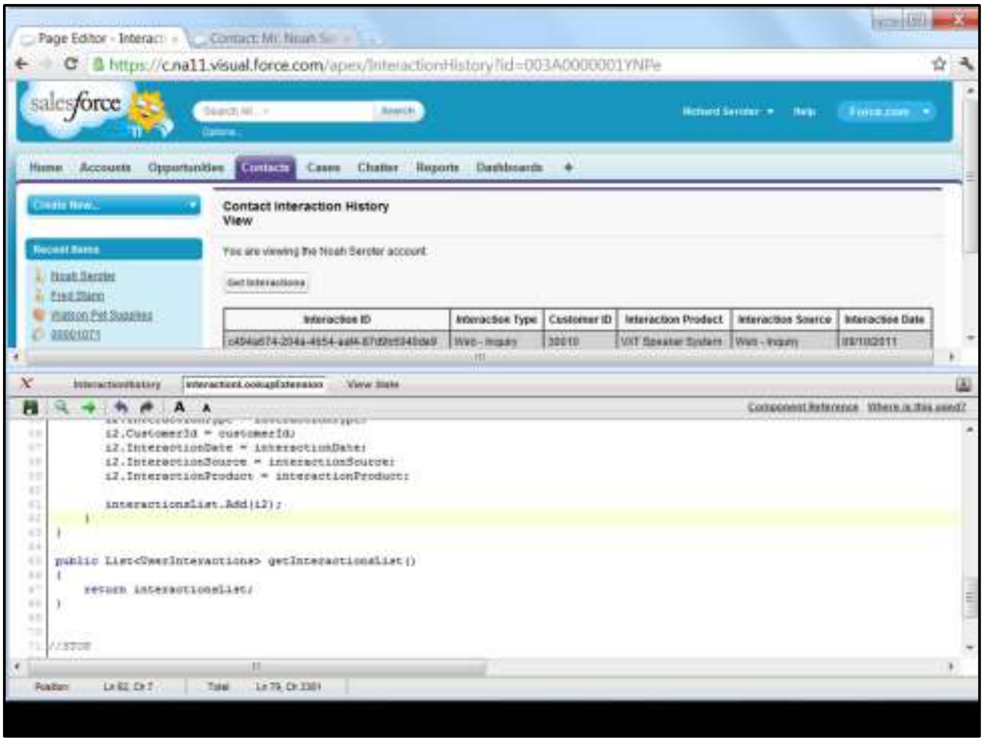
You are viewing the Noah Seroter account.

Get Interactions

Interaction ID	Interaction Type	Customer ID	Interaction Product	Interaction Source	Interaction Date
c484a674-204a-4b54-aa84-87d8650403de9	Web - Inquiry	30010	VXT Speaker System	Web - Inquiry	09/10/2011
3753c644-f4f8-4655-9621-f9ee3384f27a	Call - Inquiry	30010	DS3010 Stereo	Call - Inquiry	09/25/2011
df11e8e3-5013-48a4-9736-4926e8fcfd68	Call - Order	30010	DS3010 Stereo	Call - Order	09/28/2011

Copyright © 2000-2011 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance
InteractionHistory
InteractionLookupExtension
View State

By clicking the button on my page, I retrieved all the interactions for this Salesforce.com customer



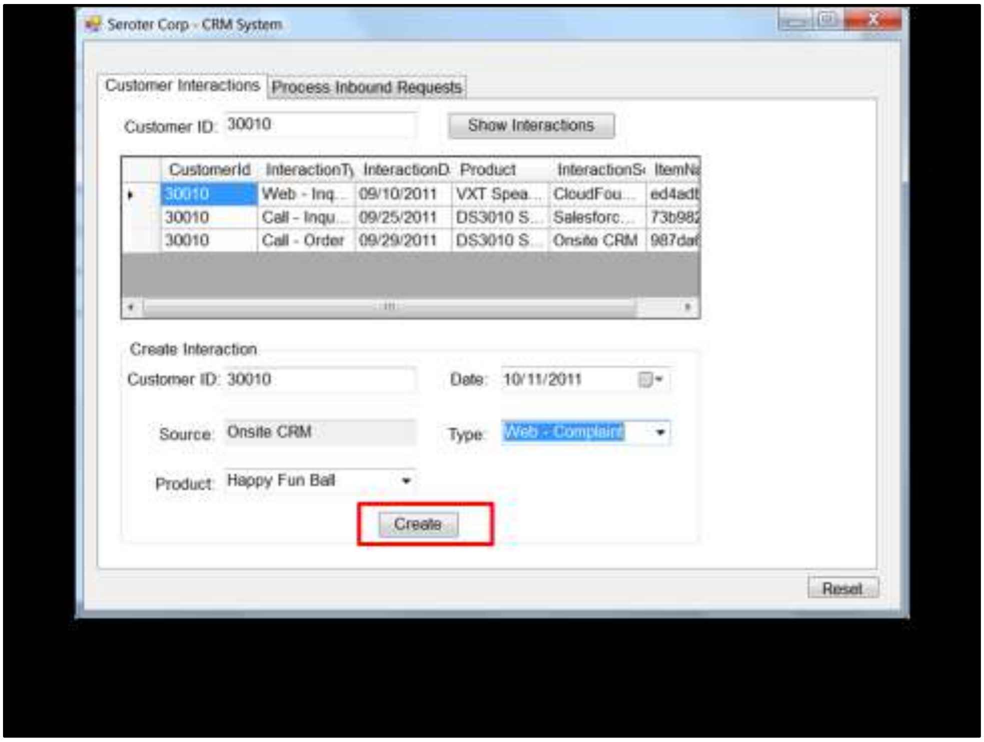
Contact Interaction History View

You are viewing the Noah Seroter account

Interaction ID	Interaction Type	Customer ID	Interaction Product	Interaction Source	Interaction Date
c494a674-204a-4654-aa14-87d2c5340da9	Web - Inquiry	30010	VXT Speaker System	Web - Inquiry	08/10/2011

```
        i2.CustomerId = customerId;
        i2.InteractionDate = interactionDate;
        i2.InteractionSource = interactionSource;
        i2.InteractionProduct = interactionProduct;

        interactionsList.Add(i2);
    }
}

public List<UserInteractions> getInteractionsList()
{
    return interactionsList;
}


//STUB
```

My code show that I consume the AWS SimpleDB service in a very similar way to Cloud Foundry app



Customer ID: 30010

CustomerId	InteractionTy	InteractionD	Product	InteractionS	ItemNa
30010	Web - Inq...	09/10/2011	VXT Spea...	CloudFou...	ed4adt
30010	Call - Inqu...	09/25/2011	DS3010 S...	Salesforc...	73b982
30010	Call - Order	09/29/2011	DS3010 S...	Onsite CRM	987dat

Customer ID: 30010

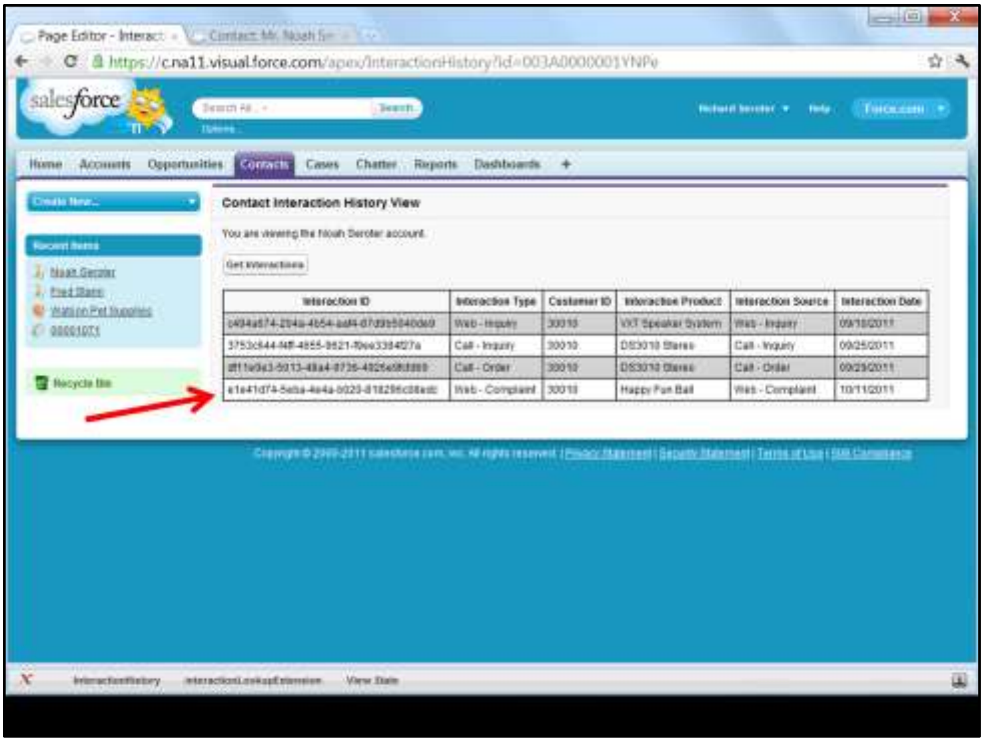
Date: 10/11/2011

Source: Onsite CRM

Type: Web - Complaint

Product: Happy Fun Ball

Within my on-premises app, I added a new record to my SimpleDB database



Contact Interaction History View

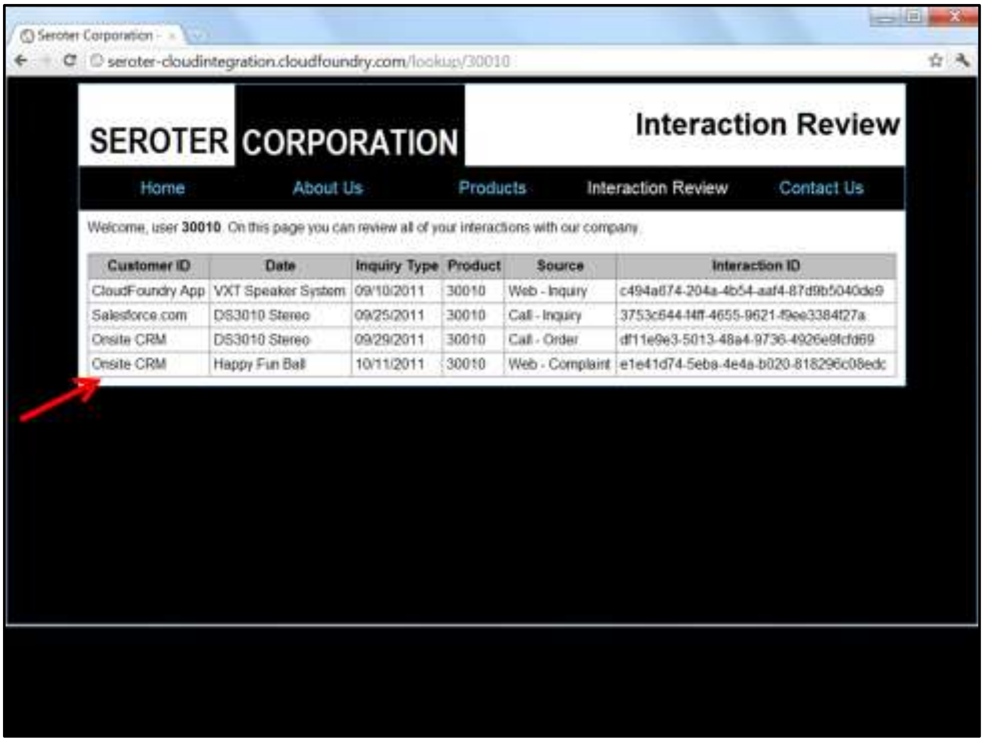
You are viewing the Noah Serotter account.

Get Interactions

Interaction ID	Interaction Type	Customer ID	Interaction Product	Interaction Source	Interaction Date
c494a574-204a-4654-aaf4-87d9b5840de9	Web - Inquiry	30010	VXT Speaker System	Web - Inquiry	09/10/2011
3753c644-f4ff-4855-8621-f0ee3384f27a	Call - Inquiry	30010	DS3010 Stereo	Call - Inquiry	09/25/2011
aff1a9a3-5013-48a4-8736-4826a6fd888	Call - Order	30010	DS3010 Stereo	Call - Order	09/29/2011
e1e41d74-5eba-4e4a-b023-818296c08adc	Web - Complaint	30010	Happy Fun Ball	Web - Complaint	10/11/2011

Copyright © 2000-2011 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance

That row is now immediately visible in Salesforce.com



Welcome, user **30010**. On this page you can review all of your interactions with our company.

Customer ID	Date	Inquiry Type	Product	Source	Interaction ID
CloudFoundry App	VXT Speaker System	09/10/2011	30010	Web - Inquiry	c494a674-204a-4b54-aaf4-87d9b5040de9
Salesforce.com	DS3010 Stereo	09/25/2011	30010	Call - Inquiry	3753c644-f4ff-4655-9621-f9ee3384f27a
Onsite CRM	DS3010 Stereo	09/29/2011	30010	Call - Order	df11e9e3-5013-48a4-9736-4926e9fcfd69
Onsite CRM	Happy Fun Ball	10/11/2011	30010	Web - Complaint	e1e41d74-5eba-4e4a-b020-818296c08edc

… and within my Cloud Foundry app!

Use queries to execute operations in other systems.

When it makes sense …

- Have multiple apps, built independently and data/processes need to be shared
- Just sharing data not always enough; processes, workflows too
- Data stays with it’s source
- SOA – business services that initiate action based on invocation
    - Encapsulation that hides internals
        - Hide underlying changes
        - Can be responsive to necessary changes vs. getting everyone on board with (shared) database changes
    - Abstraction gives you coarse functions instead of granular function calls
    - Interoperability
        - Rely on HTTP and XML/JSON
    - Reusability
- Very familiar pattern for developers (request/response)
- Good for fine grained functions and mashup services
    - Want data before moving to next step
- Often don’t need guaranteed delivery or a broker since you can just retry the request

Request/reply operations are typically blocking calls.

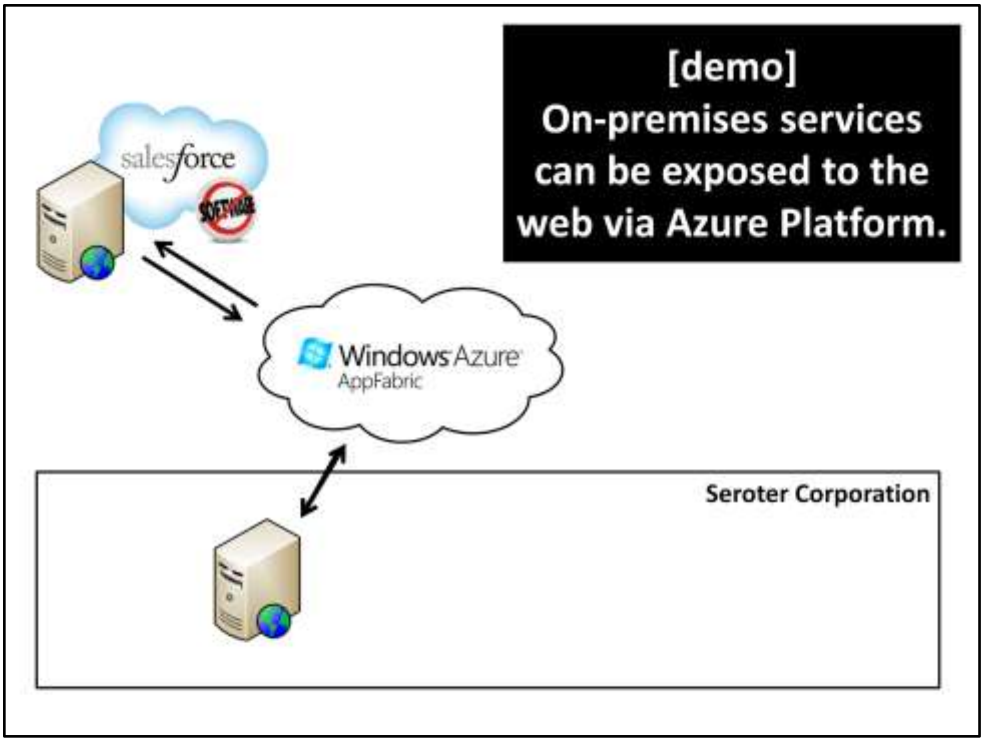
Challenges

- ** Coupling **
- Not as bad as using a shared database
- Still have relatively tight coupling
    - Hard to change one piece
- ** Security / Capability gaps **
- Have to support a wider variety of capabilities by service providers
- Lack of uniform security strategies, inconsistent support for transactions
- Depends on protocols that can be consumed
- ** Locking **
- Blocking call for the sender
    - Receiver processes exceptions (pro and con)
- ** May be transient data **
- If a straight lookup without persistence … not available for reports, workflows etc
- ** COTS support **
- Many COTS systems don’t natively expose their capabilities as services
- Some have gotten better, but you still see hyper-granular or abstract services exposed by leading vendors

Target system must be exposed to internet in some way.

Cloud Considerations

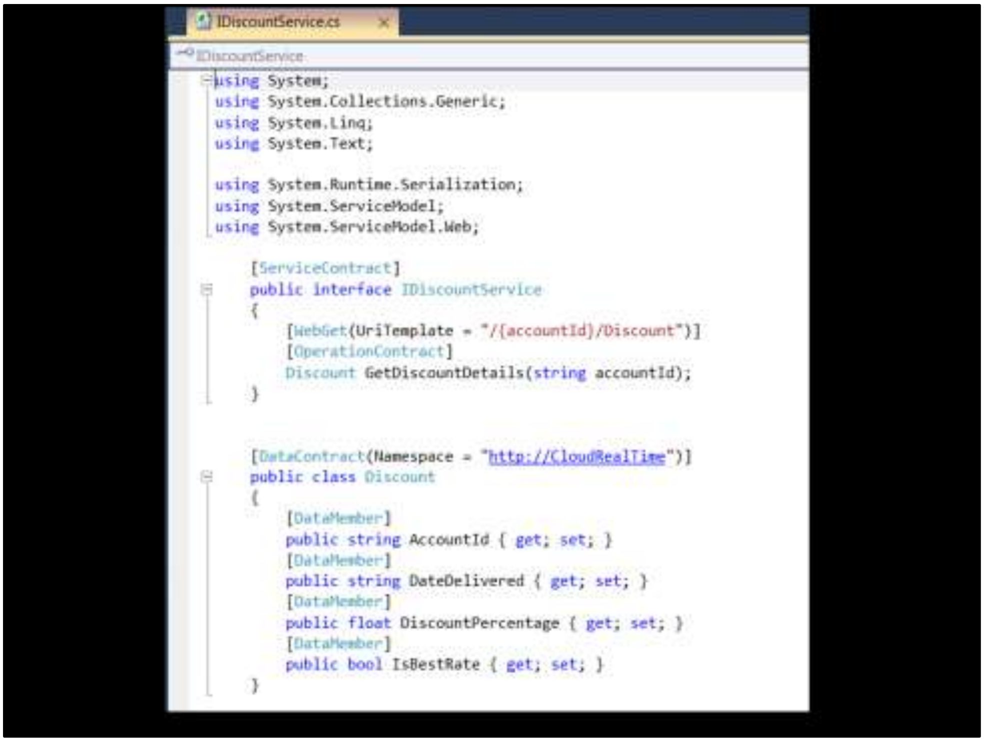
- ** Security **
  - How do you consume onsite services?
  - How federate when doing cloud-to-cloud
- ** Latency **
  - May require double hop if we first get a token and then make actual request
  - Given that this is a blocking call, may need to use AJAX design
- ** Access **
  - May require specific protocols (HTTP)
  - Not going to see broad support for DB-specific protocols
- ** Option **
  - Cloud to on-premises
    - Could use internet facing proxy service that forwards request to back-end system/DB
    - Could use VPN between caller and target system
    - Relies on cloud app that supports VPN
      - Amazon VPC, Google Data Connector, Azure Connect
    - Could use Windows Azure AppFabric for cloud based relay service (not durable)
  - Cloud to cloud
    - Leverage web services on both sides; Uwith cloud integration provider



[demo]
On-premises services can be exposed to the web via Azure Platform.

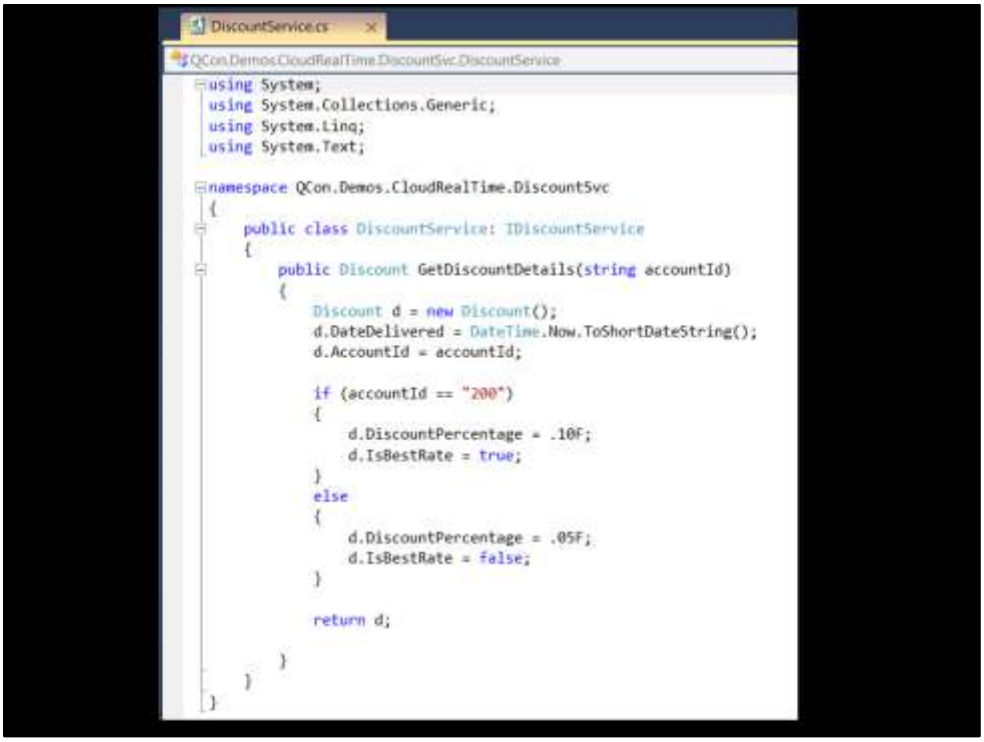
Demo

- Show on premise service
  - REST contract
  - Implementation
- Show what makes it “cloudy”
  - Cloud bindings
- Start service
- Show in registry
  - http://richardseroter.servicebus.windows.net
- https://c.na11.visual.force.com/apex/DiscountLookup?id=001A000000YBX3C
- Show SFDC call (with token for security)
- Call service
- Call service from account with different ID



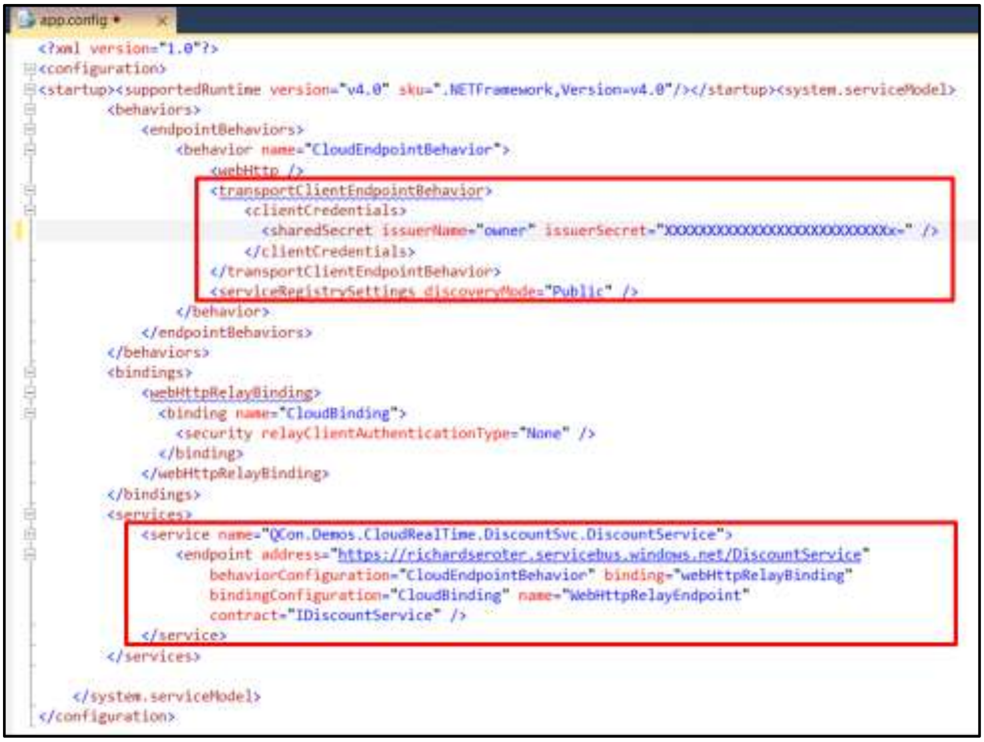
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;

    [ServiceContract]
    public interface IDiscountService
    {
        [WebGet(UriTemplate = "/{accountId}/Discount")]
        [OperationContract]
        Discount GetDiscountDetails(string accountId);
    }


    [DataContract(Namespace = "http://CloudRealTime")]
    public class Discount
    {
        [DataMember]
        public string AccountId { get; set; }
        [DataMember]
        public string DateDelivered { get; set; }
        [DataMember]
        public float DiscountPercentage { get; set; }
        [DataMember]
        public bool IsBestRate { get; set; }
    }
```

I’ve built a custom WCF service that uses a RESTful (vs. SOAP) web service strategy



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace QCon.Demos.CloudRealTime.DiscountSvc
{
    public class DiscountService: IDiscountService
    {
        public Discount GetDiscountDetails(string accountId)
        {
            Discount d = new Discount();
            d.DateDelivered = DateTime.Now.ToShortDateString();
            d.AccountId = accountId;

            if (accountId == "200")
            {
                d.DiscountPercentage = .10F;
                d.IsBestRate = true;
            }
            else
            {
                d.DiscountPercentage = .05F;
                d.IsBestRate = false;
            }

            return d;

        }
    }
}
```

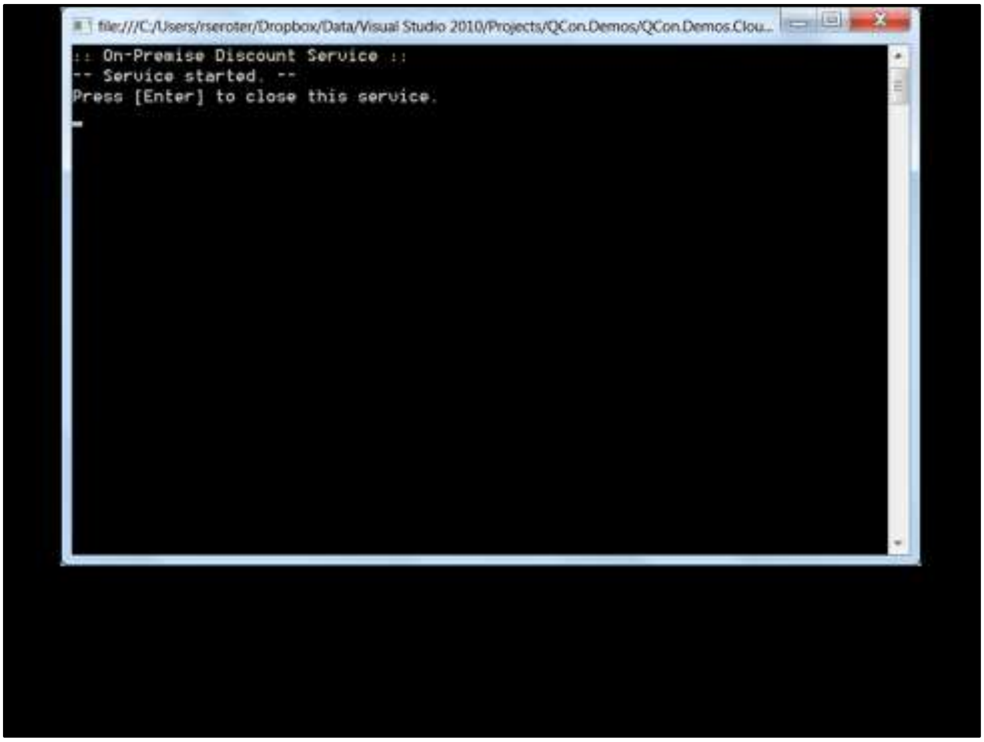
The implementation of this service simply says if the user ID is 200, the discount is 10%. Otherwise, 5%.



```
<?xml version="1.0"?>
<configuration>
<startup><supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/></startup><system.serviceModel>
        <behaviors>
            <endpointBehaviors>
                <behavior name="CloudEndpointBehavior">
                    <webHttp />
                    <transportClientEndpointBehavior>
                        <clientCredentials>
                          <sharedSecret issuerName="owner" issuerSecret="XXXXXXXXXXXXXXXXXXXXXXXXXXXx=" />
                        </clientCredentials>
                    </transportClientEndpointBehavior>
                    <serviceRegistrySettings discoveryMode="Public" />
                </behavior>
            </endpointBehaviors>
        </behaviors>
        <bindings>
            <webHttpRelayBinding>
              <binding name="CloudBinding">
                <security relayClientAuthenticationType="None" />
              </binding>
            </webHttpRelayBinding>
        </bindings>
        <services>
            <service name="QCon.Demos.CloudRealTime.DiscountSvc.DiscountService">
                <endpoint address="https://richardseroter.servicebus.windows.net/DiscountService"
                    behaviorConfiguration="CloudEndpointBehavior" binding="webHttpRelayBinding"
                    bindingConfiguration="CloudBinding" name="WebHttpRelayEndpoint"
                    contract="IDiscountService" />
            </service>
        </services>

    </system.serviceModel>
</configuration>
```

This service connects to Windows Azure AppFabric via this configuration. What this does is create a secure, two-way tunnel to the Microsoft cloud.

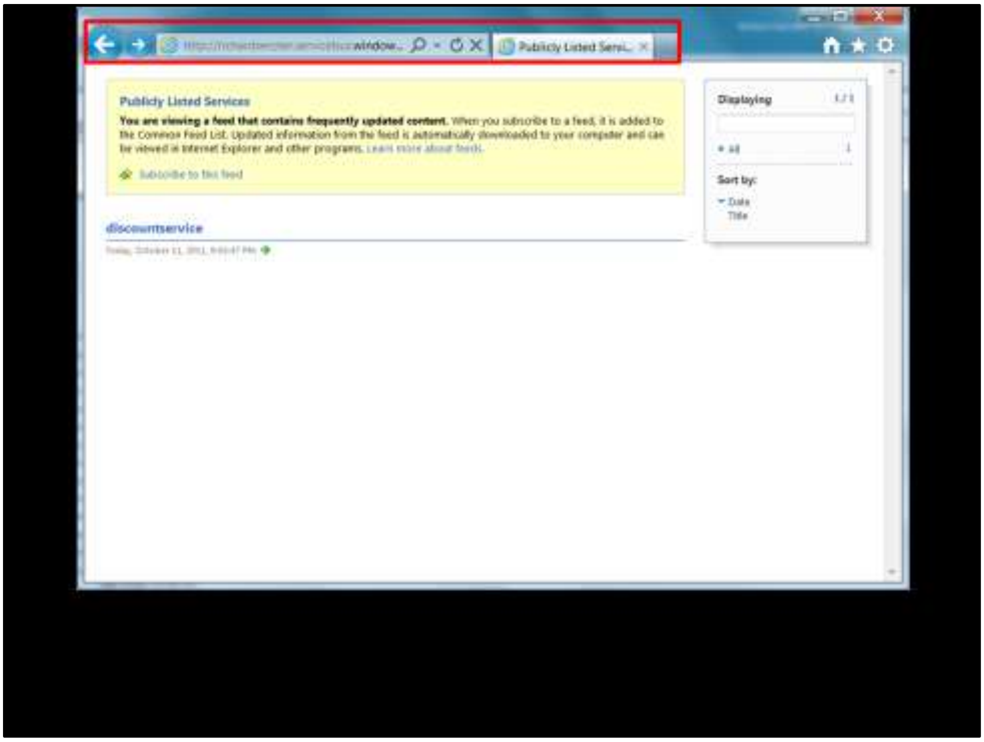
Messages sent to http://richardseroter.servicebus.windows.net/DiscountService will get “relayed” to my on-premises web service!



file:///C:/Users/rseroter/Dropbox/Data/Visual Studio 2010/Projects/QCon.Demos/QCon.Demos.Clou...

```
:: On-Premise Discount Service ::
-- Service started. --
Press [Enter] to close this service.
```

I start up my on-premises service, which initiates the binding to the cloud



Publicly Listed Services

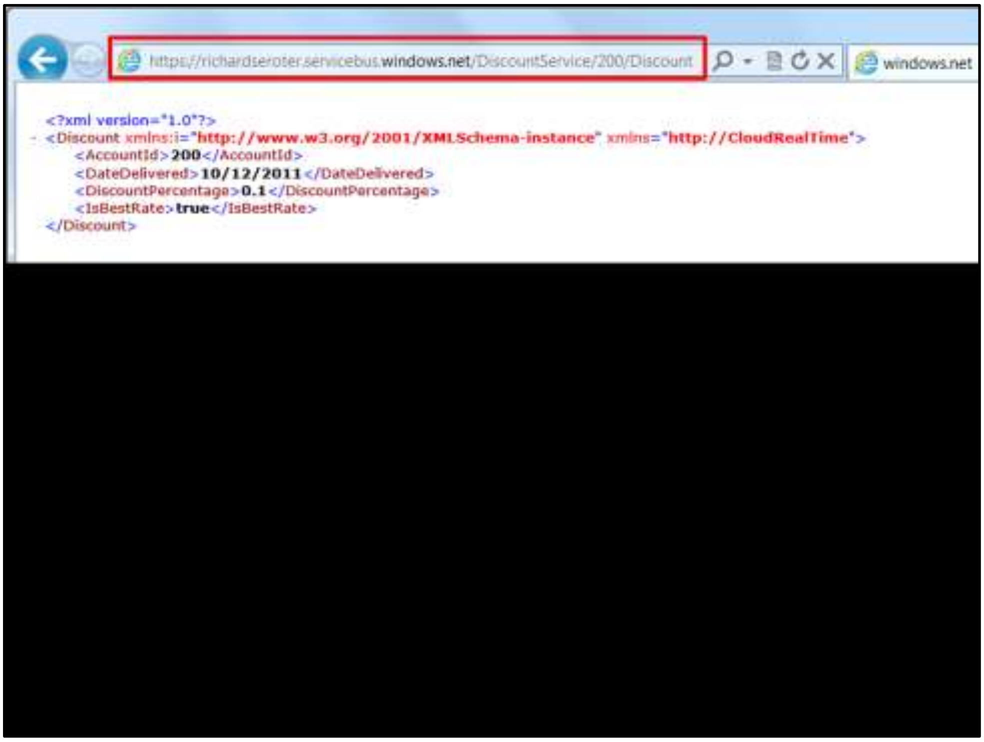
You are viewing a feed that contains frequently updated content. When you subscribe to a feed, it is added to the Common Feed List. Updated information from the feed is automatically downloaded to your computer and can be viewed in Internet Explorer and other programs. Learn more about feeds.

Subscribe to this feed

discountservice

Today, October 11, 2011, 9:03:47 PM

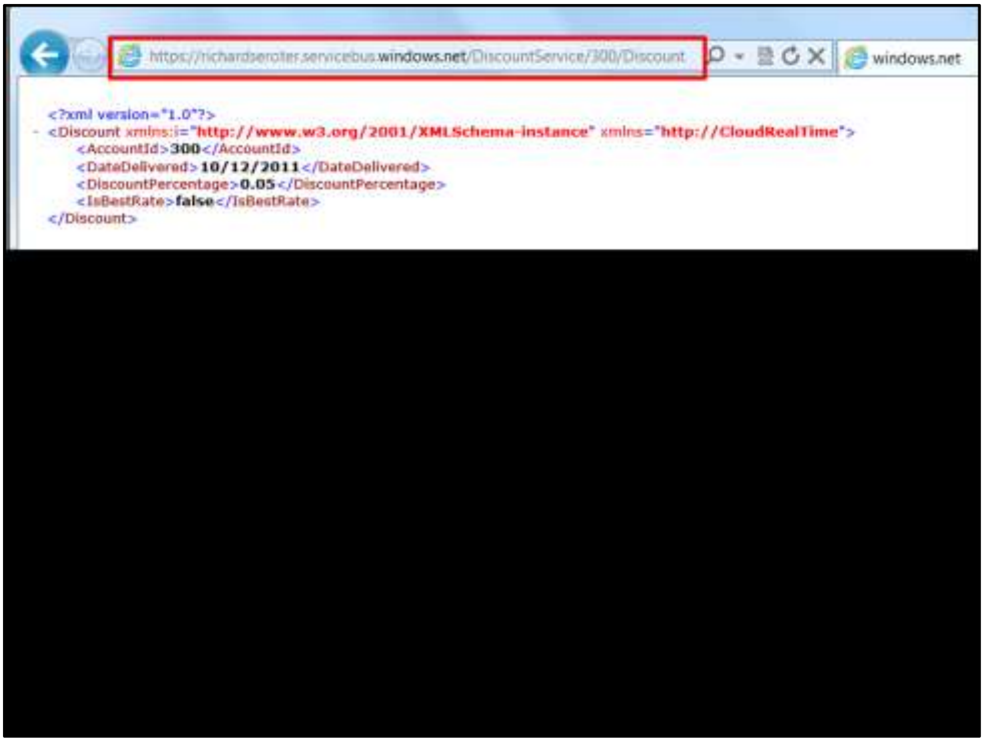
I can see my on-premises service exposed in my cloud registry



https://richardseroter.servicebus.windows.net/DiscountService/200/Discount

```
  <?xml version="1.0"?>
- <Discount xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://CloudRealTime">
      <AccountId>200</AccountId>
      <DateDelivered>10/12/2011</DateDelivered>
      <DiscountPercentage>0.1</DiscountPercentage>
      <IsBestRate>true</IsBestRate>
  </Discount>
```

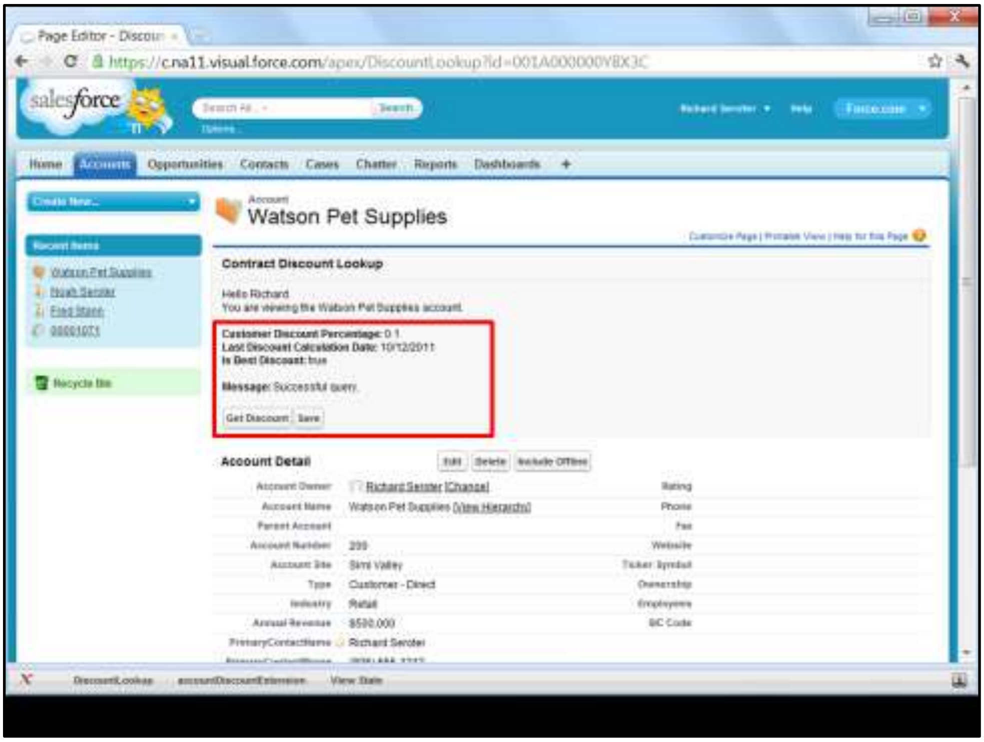
Because I have turned off caller security, I can test my service using any browser.



https://richardseroter.servicebus.windows.net/DiscountService/300/Discount

```
<?xml version="1.0"?>
- <Discount xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://CloudRealTime">
     <AccountId>300</AccountId>
     <DateDelivered>10/12/2011</DateDelivered>
     <DiscountPercentage>0.05</DiscountPercentage>
     <IsBestRate>false</IsBestRate>
  </Discount>
```

Note that I get a different value when I use a customer ID besides 200



Account

Watson Pet Supplies

Contract Discount Lookup

Hello Richard
You are viewing the Watson Pet Supplies account.

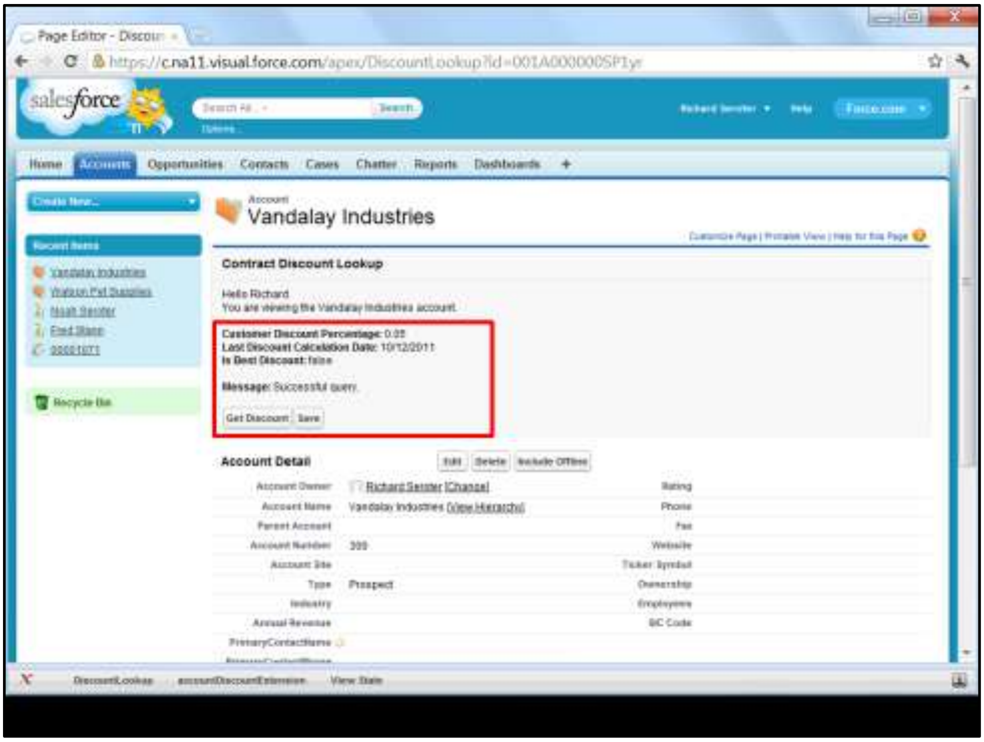
Customer Discount Percentage: 0.1
Last Discount Calculation Date: 10/12/2011
Is Best Discount: true

Message: Successful query.

Account Detail

Account Owner	Richard Seroter [Change]	Rating	
Account Name	Watson Pet Supplies [View Hierarchy]	Phone	
Parent Account		Fax	
Account Number	209	Website	
Account Site	Simi Valley	Ticker Symbol	
Type	Customer - Direct	Ownership	
Industry	Retail	Employees	
Annual Revenue	$500,000	SIC Code	
PrimaryContactName	Richard Seroter		
PrimaryContactPhone	(909) 444-1212		

Within Salesforce.com, I have a custom page which consumes the cloud relay service and executes my on-premise business logic in real time!



Vandalay Industries

Contract Discount Lookup

Hello Richard
You are viewing the Vandalay Industries account.

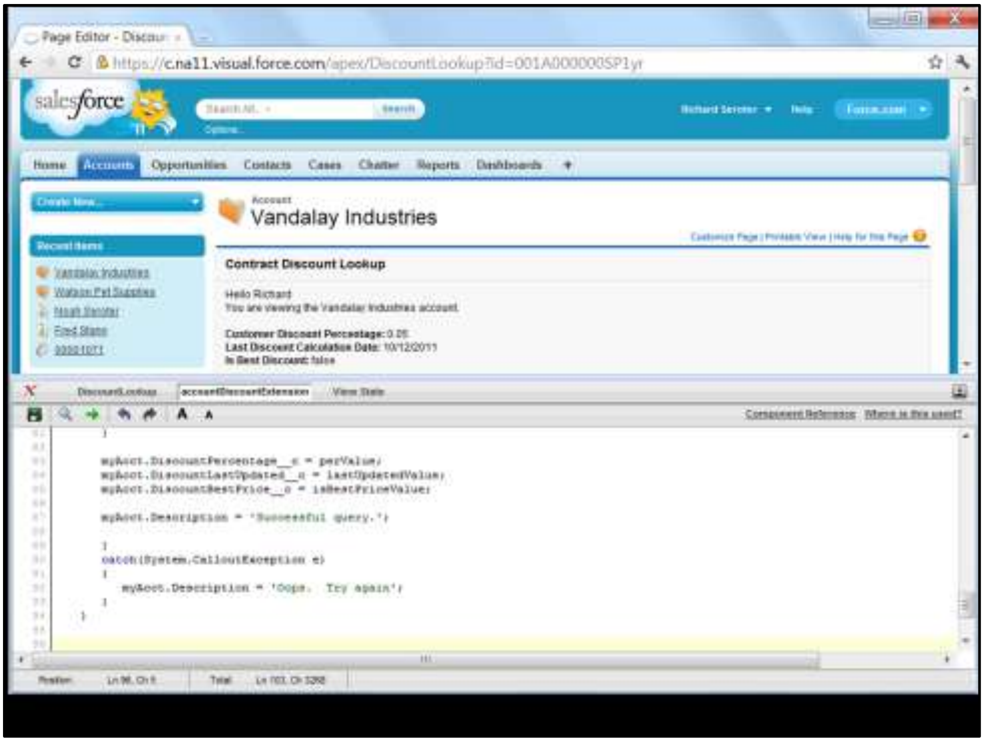
Customer Discount Percentage: 0.05
Last Discount Calculation Date: 10/12/2011
Is Best Discount: false

Message: Successful query.

Account Detail

Account Owner	Richard Sender [Change]	Rating	
Account Name	Vandalay Industries [View Hierarchy]	Phone	
Parent Account		Fax	
Account Number	300	Website	
Account Site		Ticker Symbol	
Type	Prospect	Ownership	
Industry		Employees	
Annual Revenue		SIC Code	
PrimaryContactName			

For a different account ID (besides 200), a different discount rate is returned



Account

Vandalay Industries

Contract Discount Lookup

Hello Richard
You are viewing the Vandalay Industries account.

Customer Discount Percentage: 0.05
Last Discount Calculation Date: 10/12/2011
Is Best Discount: false

```
        }

        myAcct.DiscountPercentage__c = perValue;
        myAcct.DiscountLastUpdated__c = lastUpdatedValue;
        myAcct.DiscountBestPrice__c = isBestPriceValue;

        myAcct.Description = 'Successful query.';

        }
        catch(System.CalloutException e)
        {
           myAcct.Description = 'Oops.  Try again';
        }
    }
```

Behind the scenes, this code connects to the relay service and optionally passes in a security token.

Asynchronous messaging works well for distributed workloads.

When it makes sense …

- Share data between systems in a responsive and scalable way
    - Caller doesn’t have to wait
    - Think of sending an email; it doesn’t block Outlook until your recipient gets it
- Want to do broadcast (pub/sub) or multicast (defined recipients)
- Caller doesn’t need to care where it goes
- Supports disconnected applications (not online at the same time)
- Have multiple replicable units (bus scenario)

Data shared asynchronously will eventually be consistent.

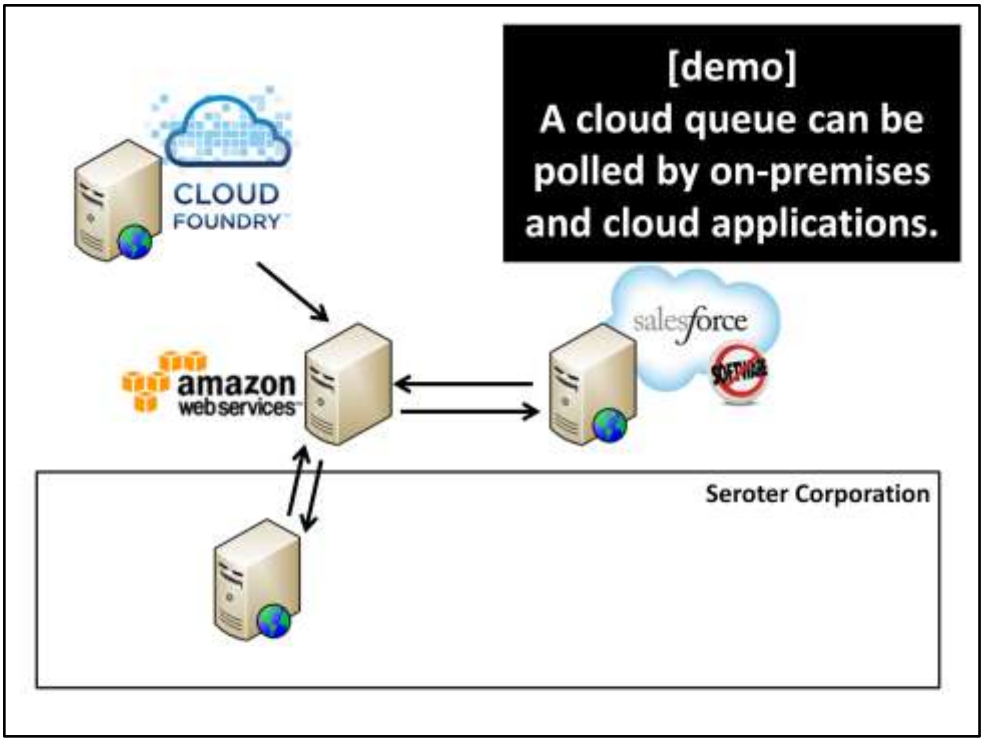
Challenges

- ** Not real time synchronization **
- Inherently includes eventual consistency
    - No simultaneous or instant updates
- ** May need durability / reliability **
- Without durability in the middle, stand the chance to lose data
- A router can be used if you want to direct the messages to places unknown to the caller
    - Router can also do activities like data transformation, protocol bridging, or workflow
- Idempotence needed in many cases
    - Data shared my tell receiving system to go get data (handle dupes ok)
- ** App support for receiving or sending async messages **
- Few applications natively share data asynchronously

To perform a reliable exchange, may use a cloud broker.

Cloud Considerations

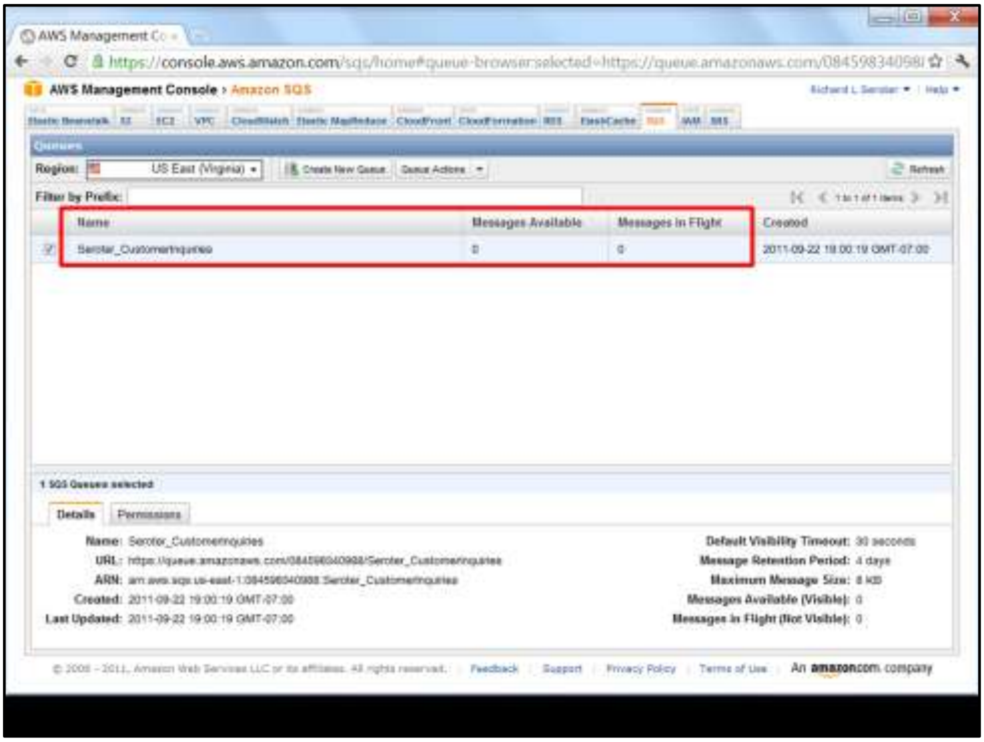
- ** App support **
- Seems even fewer cloud apps share or receive async
- ** Security **
- May still need to poke a hole in the firewall
- ** Provider limits **
- Cloud vendor limits on polling (e.g. SFDC)
    - May want to put lots into queue and “peek” to retrieve/delete only certain ones
- Distributed nature of cloud arch could result in unexpected (lack) of data
    - Distributed queues may not return all items on each poll (machine sample via Amazon)
- **Options **
- Some SaaS platforms bake in async push (e.g. salesforce.com)
- Could leverage a cloud queue that everyone talks to
    - Or other cloud broker (DB, message router)
    - Could work for both cloud-to-cloud or between cloud and on-premises (both ways)



[demo]
A cloud queue can be polled by on-premises and cloud applications.

Demo

- Show queue on AWS console
- Show CF call that puts stuff into queue
- Show queue in AWS console
- Call Pull from on-premises client
  - Do retrieve plus delete
- Now empty if you look at AWS console
- Go to SFDC
  - Have code that can call “poller” every hour and update SFDC by creating a “case”
  - Show poller interface
  - https://c.na11.visual.force.com/apex/CaseView
  - Can also trigger manually and check queue and create “case”
  - See AWS queue is now empty



https://console.aws.amazon.com/sqs/home#queue-browser:selected=https://queue.amazonaws.com/08459834098

Region: US East (Virginia)

Filter by Prefix:

	Name	Messages Available	Messages in Flight	Created
☑	Seroter_CustomerInquiries	0	0	2011-09-22 19:00:19 GMT-07:00

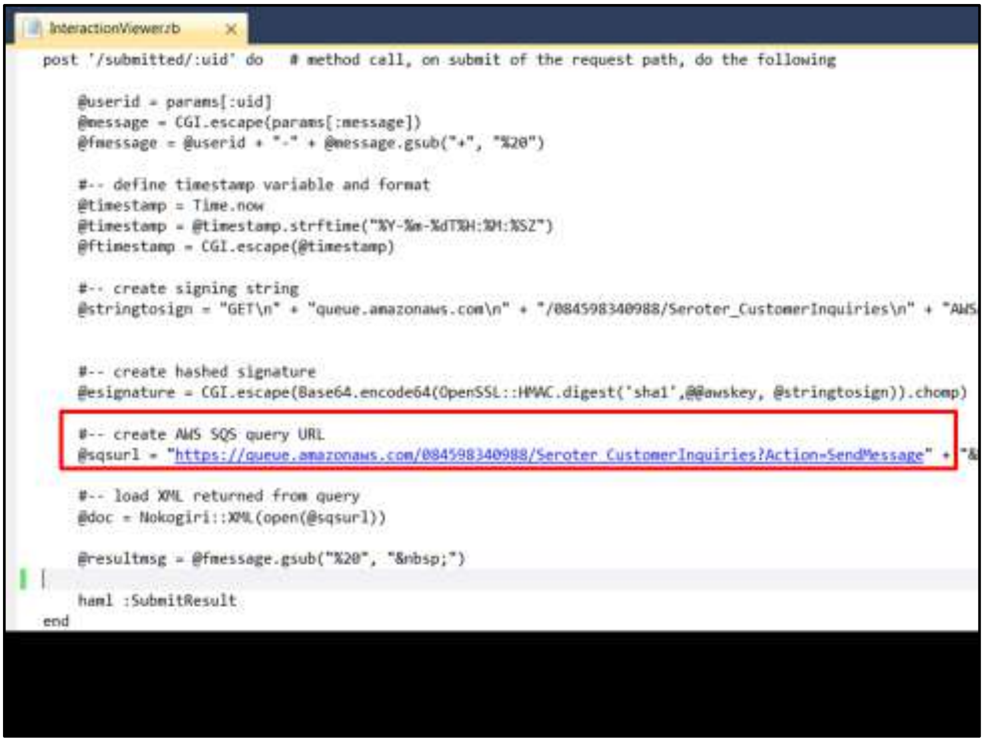
1 SQS Queues selected

Name: Seroter_CustomerInquiries
URL: https://queue.amazonaws.com/084598040988/Seroter_CustomerInquiries
ARN: arn:aws:sqs:us-east-1:084598040988:Seroter_CustomerInquiries
Created: 2011-09-22 19:00:19 GMT-07:00
Last Updated: 2011-09-22 19:00:19 GMT-07:00

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 8 KB
Messages Available (Visible): 0
Messages in Flight (Not Visible): 0

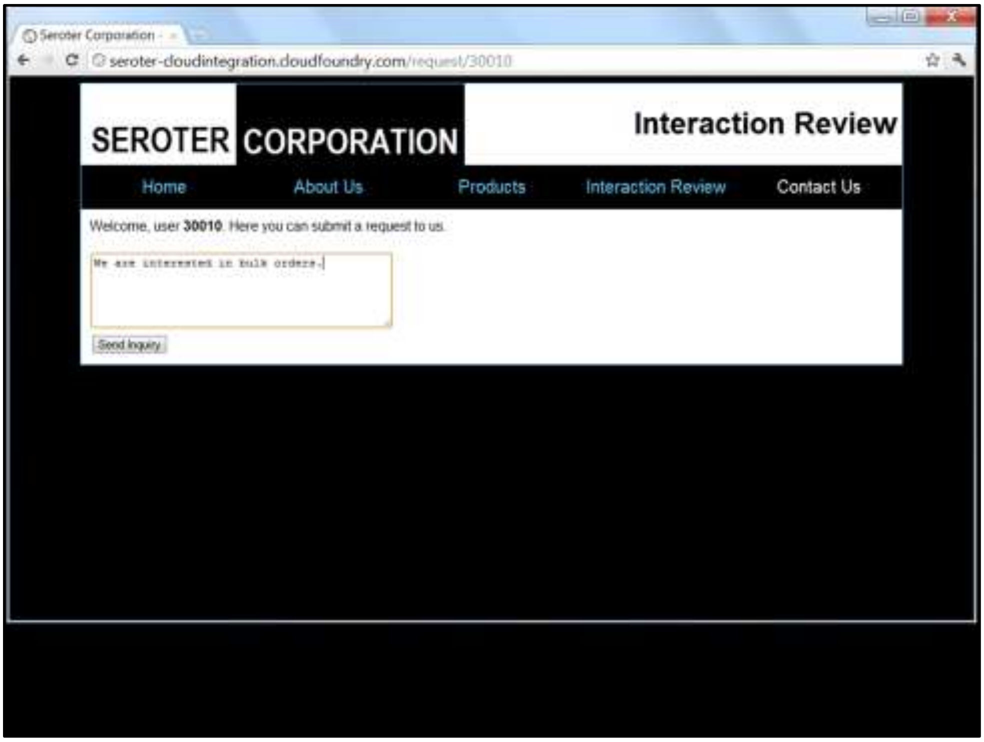
© 2008 - 2011, Amazon Web Services LLC or its affiliates. All rights reserved.

I have an AWS Simple Queue Service (SQS) queue created



```
post '/submitted/:uid' do   # method call, on submit of the request path, do the following

    @userid = params[:uid]
    @message = CGI.escape(params[:message])
    @fmessage = @userid + "-" + @message.gsub("+", "%20")

    #-- define timestamp variable and format
    @timestamp = Time.now
    @timestamp = @timestamp.strftime("%Y-%m-%dT%H:%M:%SZ")
    @ftimestamp = CGI.escape(@timestamp)

    #-- create signing string
    @stringtosign = "GET\n" + "queue.amazonaws.com\n" + "/084598340988/Seroter_CustomerInquiries\n" + "AWS


    #-- create hashed signature
    @esignature = CGI.escape(Base64.encode64(OpenSSL::HMAC.digest('sha1',@@awskey, @stringtosign)).chomp)

    #-- create AWS SQS query URL
    @sqsurl = "https://queue.amazonaws.com/084598340988/Seroter_CustomerInquiries?Action=SendMessage" + "&

    #-- load XML returned from query
    @doc = Nokogiri::XML(open(@sqsurl))

    @resultmsg = @fmessage.gsub("%20", "&nbsp;")

    haml :SubmitResult
end
```

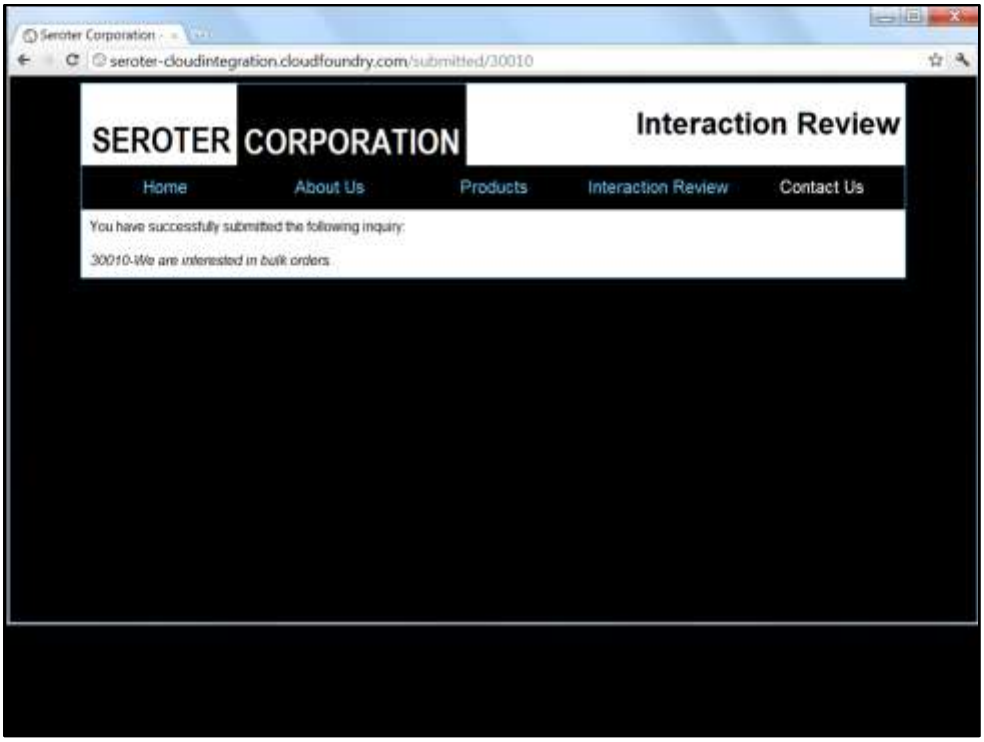
Within my Cloud Foundry app, I build up my request to SQS and add a message to the queue



Welcome, user **30010**. Here you can submit a request to us.

We are interested in bulk orders.

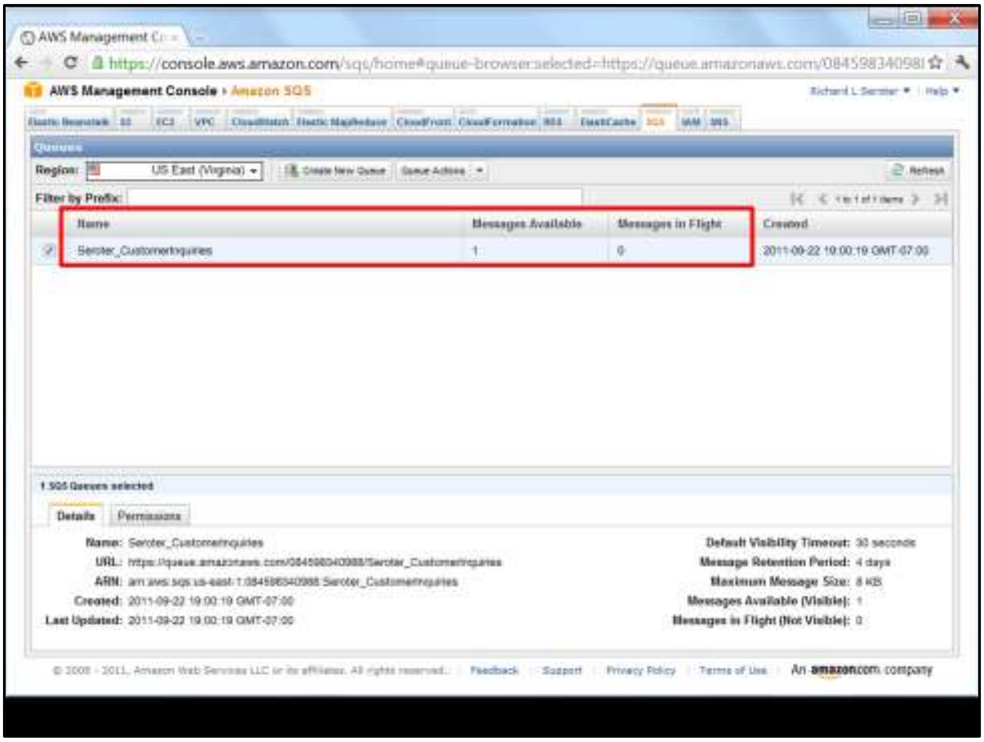
In this Cloud Foundry app, I send “feedback” messages from websites to a queue where either an on-premises application, or other cloud app, can read them.



You have successfully submitted the following inquiry:

30010-We are interested in bulk orders

A message has been added to the queue.



Region: US East (Virginia)

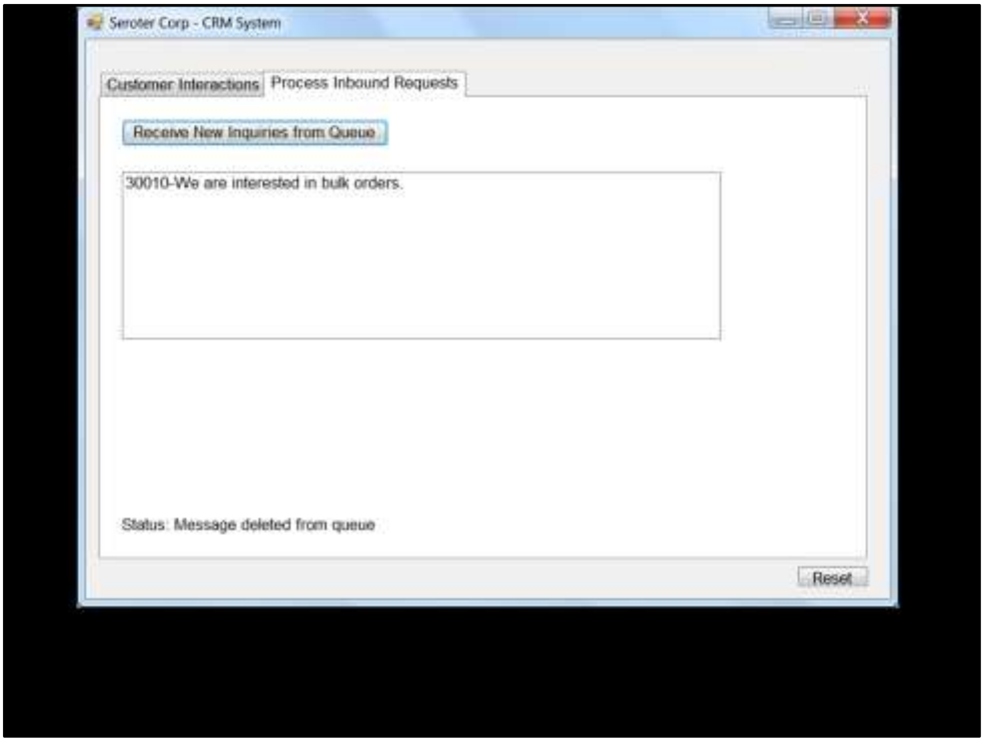
Filter by Prefix:

	Name	Messages Available	Messages in Flight	Created
☑	Seroter_CustomerInquiries	1	0	2011-09-22 19:00:19 GMT-07:00

Name: Seroter_CustomerInquiries
URL: https://queue.amazonaws.com/084598340988/Seroter_CustomerInquiries
ARN: arn:aws:sqs:us-east-1:084598340988:Seroter_CustomerInquiries
Created: 2011-09-22 19:00:19 GMT-07:00
Last Updated: 2011-09-22 19:00:19 GMT-07:00

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 8 KB
Messages Available (Visible): 1
Messages in Flight (Not Visible): 0

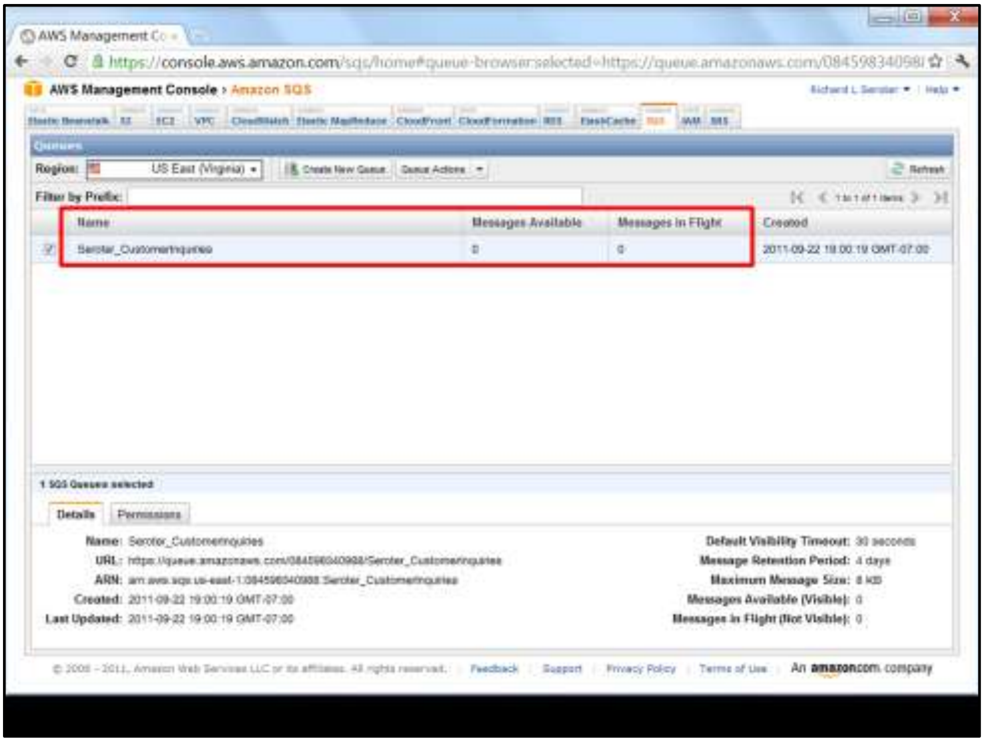
I can see that my queue has a message in it.



30010-We are interested in bulk orders.

Status: Message deleted from queue

Within my on-premises app, I can retrieve and then delete that message (so that other queue readers don’t get it).



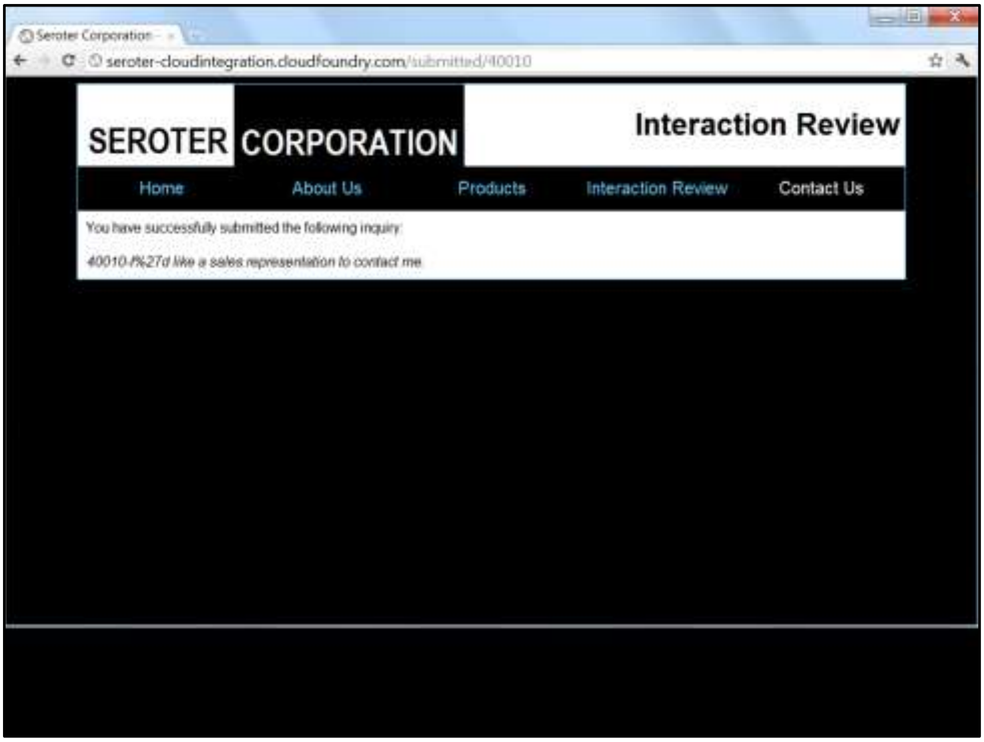
AWS Management Co
https://console.aws.amazon.com/sqs/home#queue-browser:selected=https://queue.amazonaws.com/084598340988
AWS Management Console › Amazon SQS
Richard L Seroter ▾ | Help ▾
Elastic Beanstalk S3 EC2 VPC CloudWatch Elastic MapReduce CloudFront CloudFormation RDS ElastiCache SQS IAM SNS
Queues
Region: US East (Virginia) ▾
Create New Queue
Queue Actions ▾
Refresh
Filter by Prefix:
1 to 1 of 1 items
1 SQS Queues selected
Details
Permissions
© 2008 - 2011, Amazon Web Services LLC or its affiliates. All rights reserved.
Feedback
Support
Privacy Policy
Terms of Use
An amazon.com company

	Name	Messages Available	Messages in Flight	Created
☑	Seroter_CustomerInquiries	0	0	2011-09-22 19:00:19 GMT-07:00

Name: Seroter_CustomerInquiries
URL: https://queue.amazonaws.com/084598340988/Seroter_CustomerInquiries
ARN: arn:aws:sqs:us-east-1:084598340988:Seroter_CustomerInquiries
Created: 2011-09-22 19:00:19 GMT-07:00
Last Updated: 2011-09-22 19:00:19 GMT-07:00

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 8 KB
Messages Available (Visible): 0
Messages in Flight (Not Visible): 0

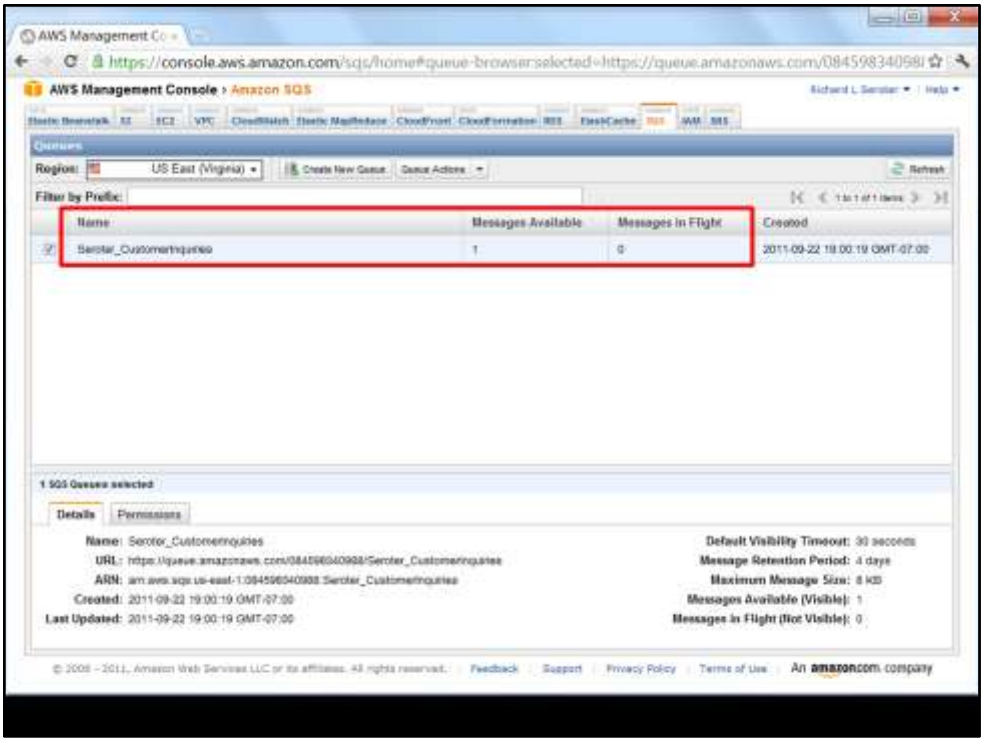
My queue is now empty.



You have successfully submitted the following inquiry:

40010-I%27d like a sales representation to contact me

I sent another message for a different customer.



Region: US East (Virginia)

	Name	Messages Available	Messages in Flight	Created
☑	Seroter_CustomerInquiries	1	0	2011-09-22 19:00:19 GMT-07:00

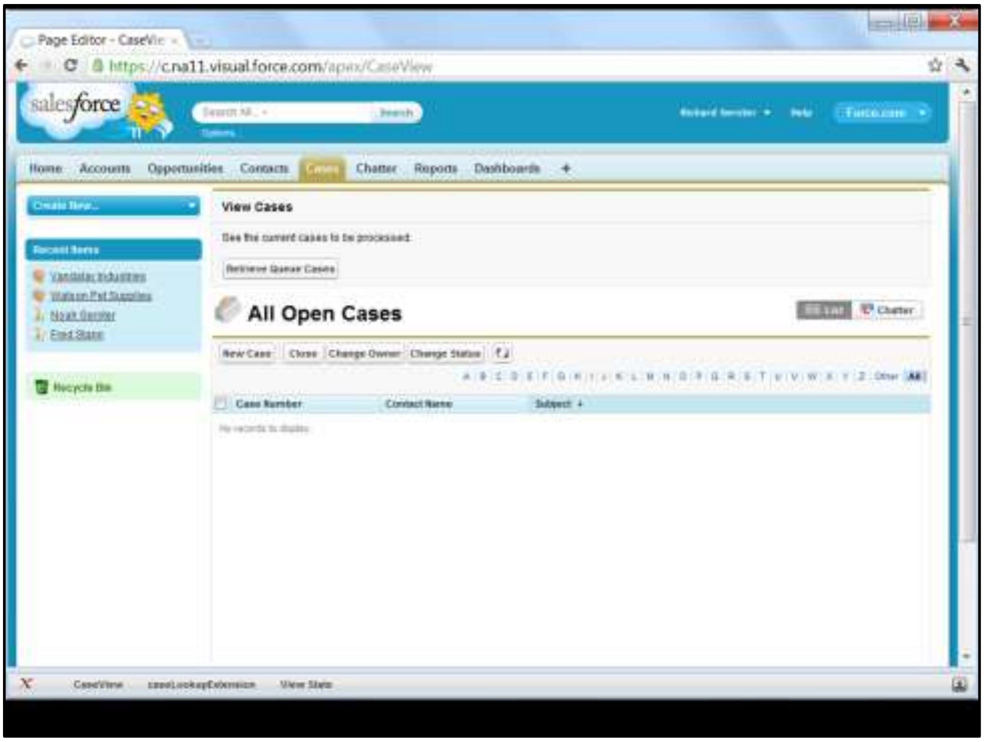
1 SQS Queues selected

Name: Seroter_CustomerInquiries
URL: https://queue.amazonaws.com/084598340988/Seroter_CustomerInquiries
ARN: arn:aws:sqs:us-east-1:084598340988:Seroter_CustomerInquiries
Created: 2011-09-22 19:00:19 GMT-07:00
Last Updated: 2011-09-22 19:00:19 GMT-07:00

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 8 KB
Messages Available (Visible): 1
Messages in Flight (Not Visible): 0

© 2008 - 2011, Amazon Web Services LLC or its affiliates. All rights reserved.

My queue now has one message



View Cases

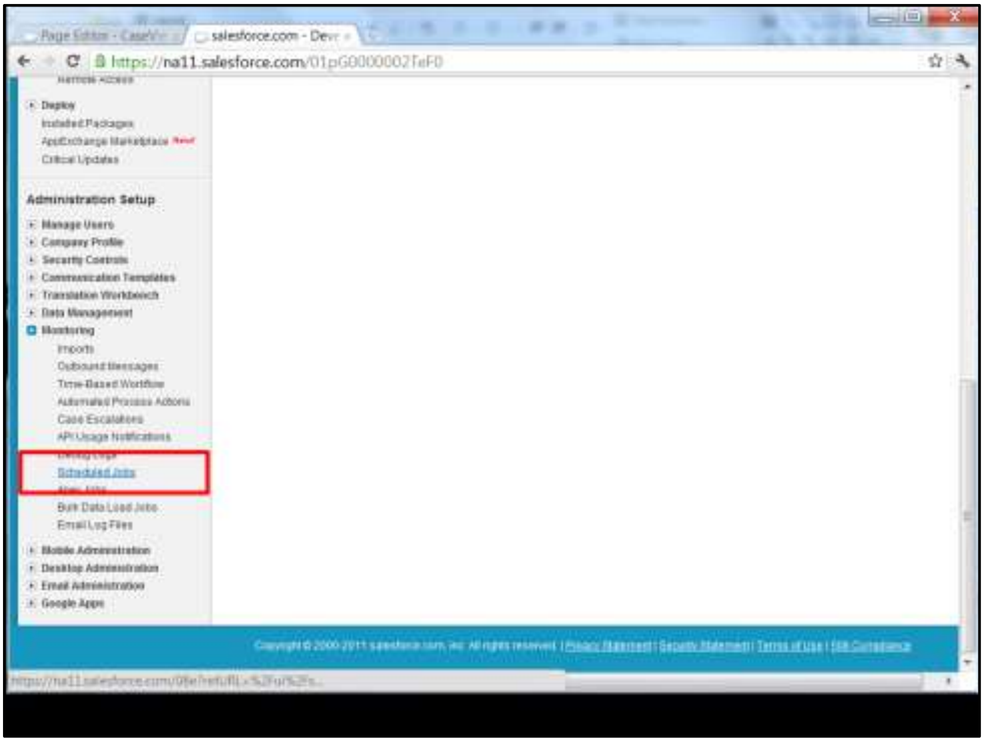
See the current cases to be processed.

All Open Cases

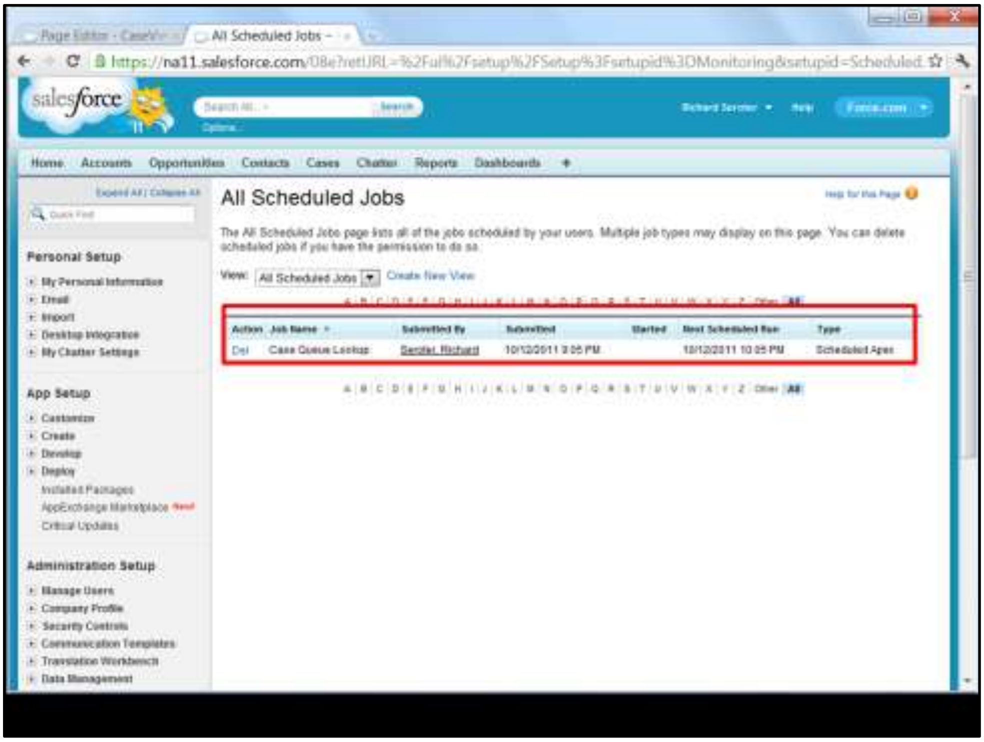
□ Case Number	Contact Name	Subject ↑

No records to display.

Within Salesforce.com, I have a custom page which queries the queue, and if it finds anything, it creates a new “case” in the system.



I can also schedule this “queue query” to run on scheduled intervals for a more real-time data exchange. Instead of doing bulk integration every day, we could send messages to the queue and have them loaded into Salesforce.com every hour or so.



https://na11.salesforce.com/08e?retURL=%2Fui%2Fsetup%2FSetup%3Fsetupid%3DMonitoring&setupid=Scheduled

All Scheduled Jobs

The All Scheduled Jobs page lists all of the jobs scheduled by your users. Multiple job types may display on this page. You can delete scheduled jobs if you have the permission to do so.

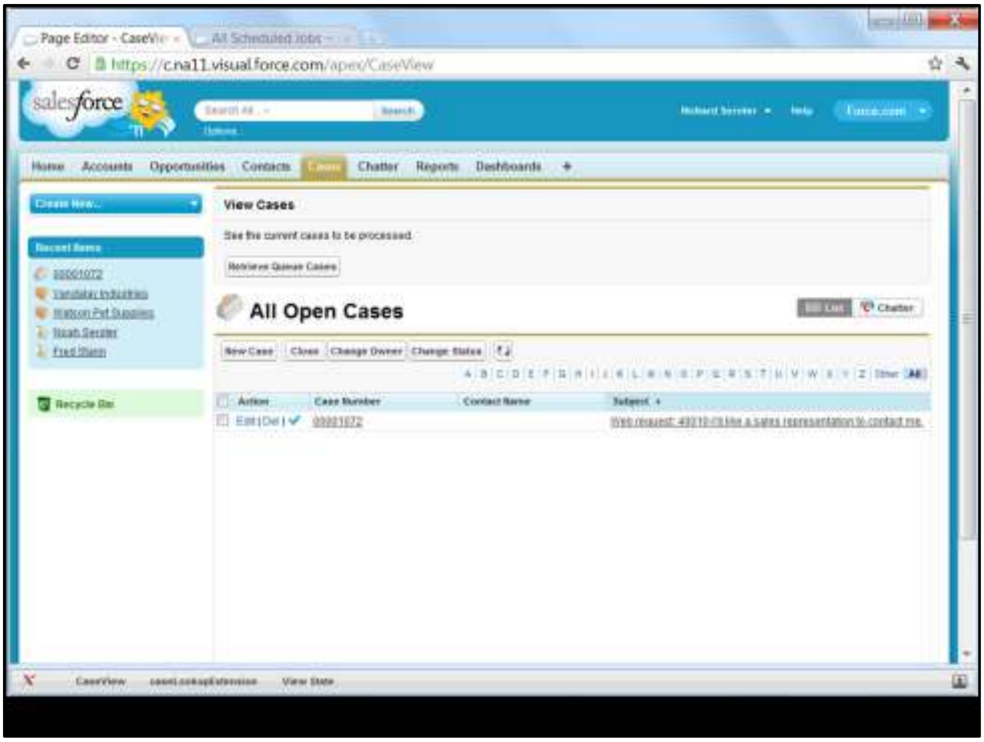
View: All Scheduled Jobs ▾ Create New View

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | All

Action	Job Name	Submitted By	Submitted	Started	Next Scheduled Run	Type
Del	Case Queue Lookup	Sertler, Richard	10/12/2011 9:05 PM		10/12/2011 10:05 PM	Scheduled Apex

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | All

Here is my job scheduled to run every hour.



View Cases

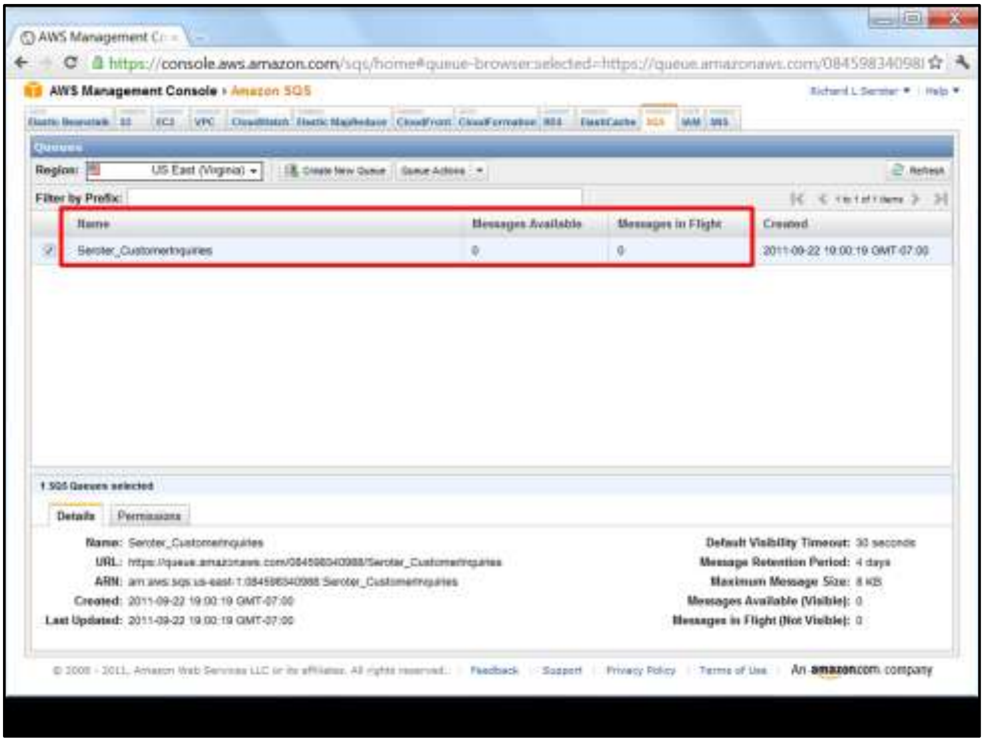
See the current cases to be processed.

Retrieve Queue Cases

All Open Cases

□	Action	Case Number	Contact Name	Subject
□	Edit | Del | ✓	00001072		Web request: 49210-I'd like a sales representation to contact me.

Since I'm impatient, I can also click the button the page to look for queue messages. Here, the message we sent from Cloud Foundry shows up as a case in Salesforce.com.



Region: US East (Virginia)

Filter by Prefix:

	Name	Messages Available	Messages in Flight	Created
☑	Seroter_CustomerInquiries	0	0	2011-09-22 19:00:19 GMT-07:00

1 SQS Queues selected

Name: Seroter_CustomerInquiries
URL: https://queue.amazonaws.com/084598340988/Seroter_CustomerInquiries
ARN: arn:aws:sqs:us-east-1:084598340988:Seroter_CustomerInquiries
Created: 2011-09-22 19:00:19 GMT-07:00
Last Updated: 2011-09-22 19:00:19 GMT-07:00

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 8 KB
Messages Available (Visible): 0
Messages in Flight (Not Visible): 0

© 2008 - 2011, Amazon Web Services LLC or its affiliates. All rights reserved.

The queue is now empty.

Apply proven patterns to cloud integration and look for more than just bulk transfer scenarios.

Questions?

@rseroter | seroter.wordpress.com

QCon

北京站 · 2012年4月18~20日

www.qconbeijing.com（11月启动）

QCon杭州站官网和资料

www.qconhangzhou.com

全球企业开发大会

INTERNATIONAL
SOFTWARE DEVELOPMENT
CONFERENCE