BUILDING STABLE AGILE CLOUD APPLICATIONS

Dan Piessens

@dpiessens

centare™

inrule TECHNOLOGY®

API HEALTHCARE

WordPress.com

PayPal + Braintree Developer

SKYLINE TECHNOLOGIES

SafeNet

DevExpress

omni resources

corvisa cloud

TechSmith®

ABOUT ME

- Senior Agile Consultant
- 13 Years Experience as Developer, IT Consultant, Architect, Trainer, Coach
- 2008 – 2013 MS Patterns & Practices Champion

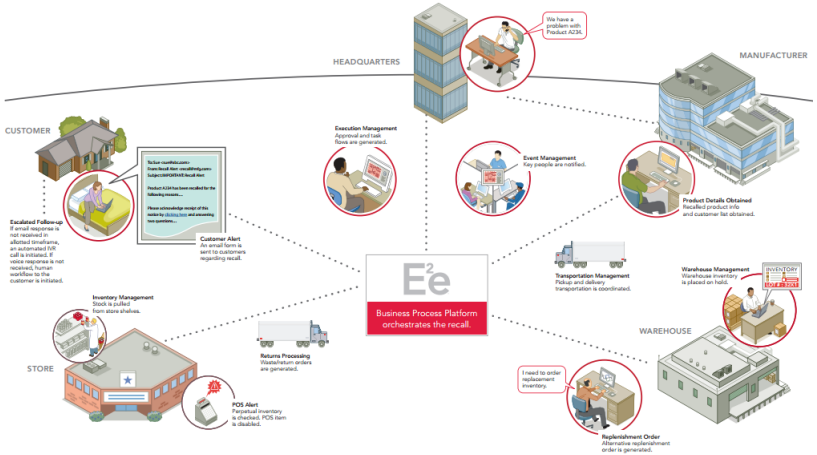
TALES FROM THE TRENCHES

"It was a good idea at the time. . ."

THE SUPPLY CHAIN COMPANY

- Large software vendor for supply chain solutions
- Software was designed for on premise installation
- Until one day . . .

They tried to go to the cloud!



THE INSURANCE COMPANY

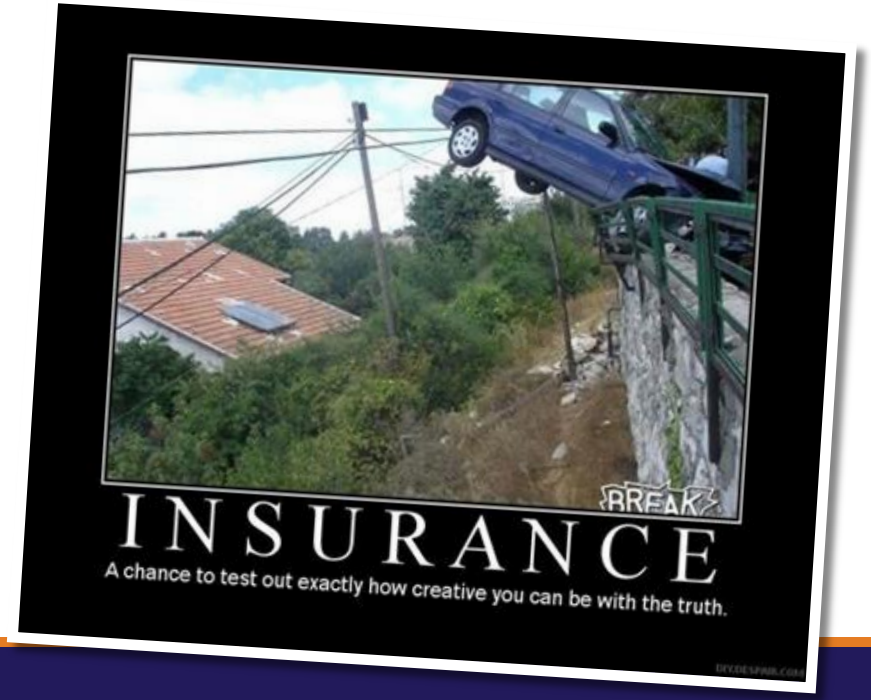
- SaaS software for insurance brokers
- Large stable private cloud
- Until one day . . .

Open Enrollment Started

ANOTHER INSURANCE COMPANY

- Green field consumer insurance software
- Rapid development schedule
- Heavily used 3rd party backend
- Until one day . . .

They fired the 3rd party vendor!



WHY DO WE CARE?

"It works fine most of the time"

IT'S ALL ABOUT PERCEPTION

- Response times affect perception
  - 0.1s - Users feeling that they are directly manipulating the UI
  - 1.0s - Users feeling that they are freely navigating the UI
  - 2.0s – User feel a noticeable delay in the UI
  - 10s - Users feeling that their experience is impaired
- This was from a paper in 1968!

http://www.nngroup.com/articles/response-times-3-important-limits/

CURRENT HARDWARE IS STALE

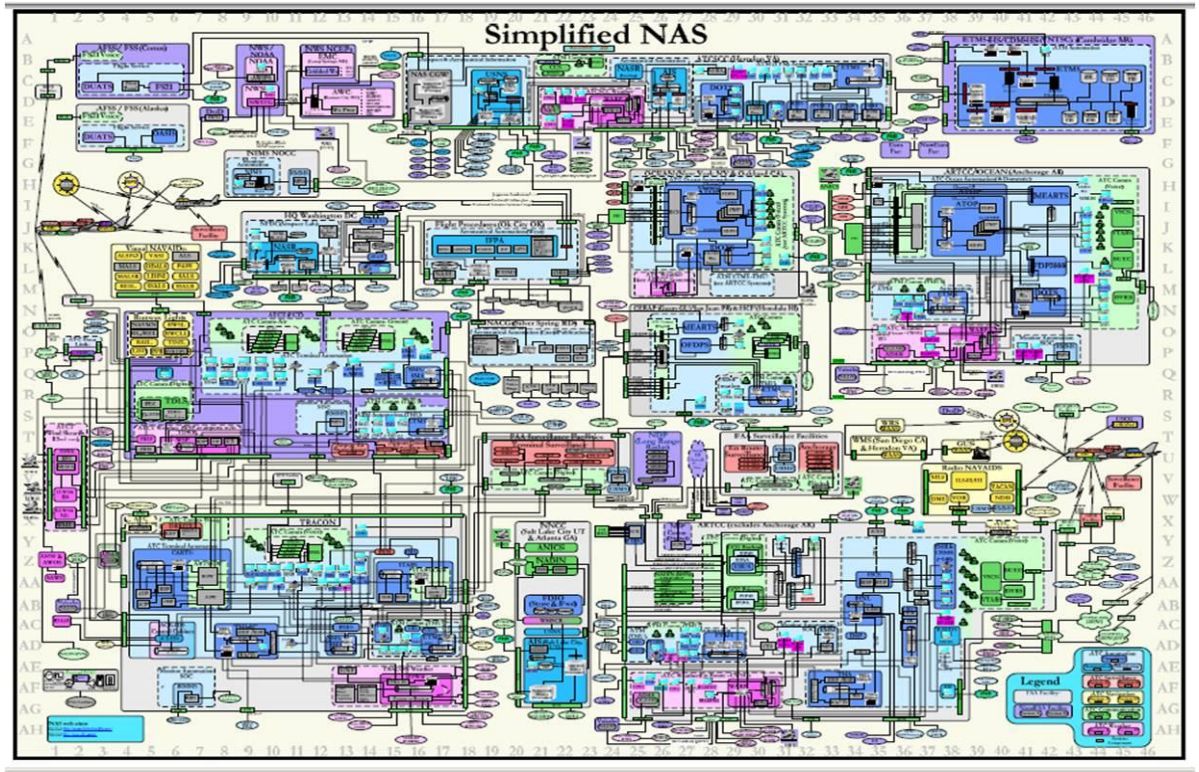
TO THE CLOUD!

CLOUD ADVANTAGES

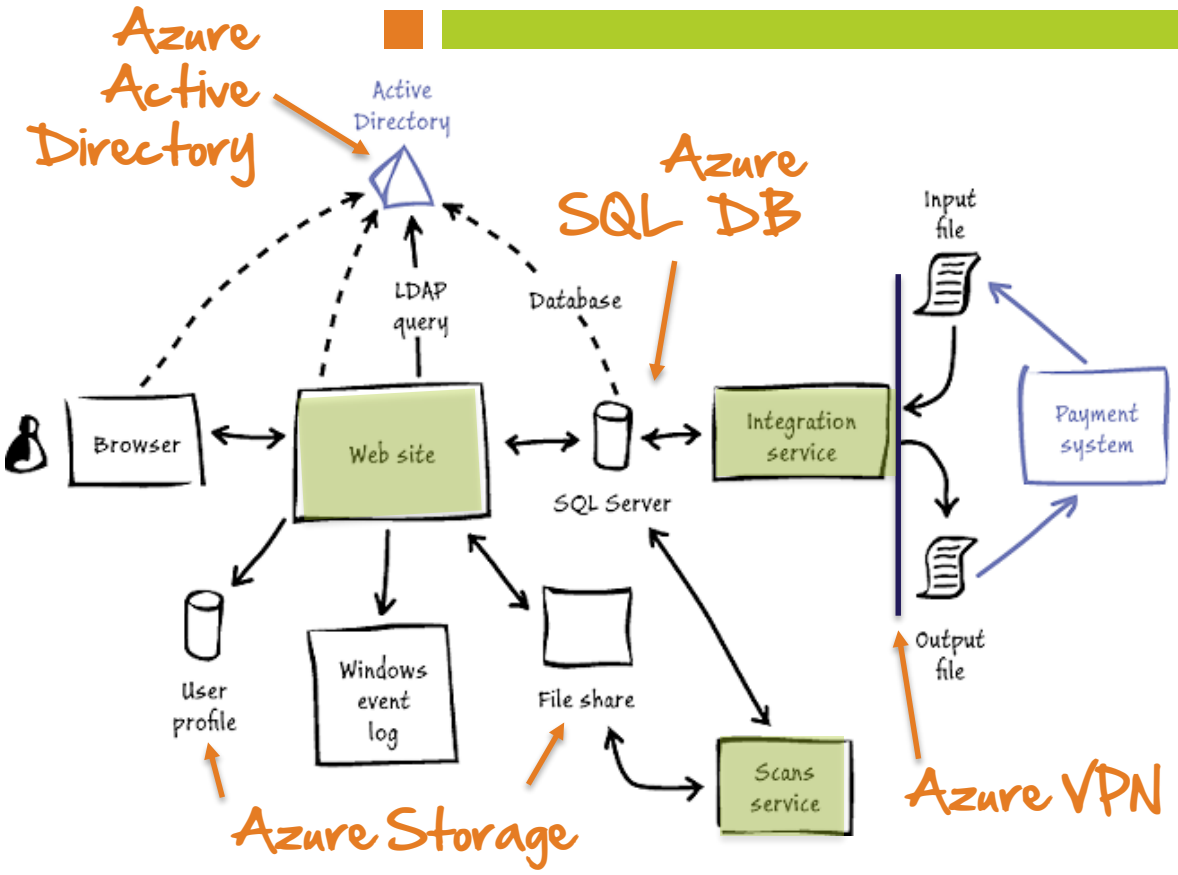
- Agility
- Secure*
- Scalable
- Cost Effective
- Full Automation Available

* Easy to make insecure

BUT YOUR APP LOOKS LIKE THIS!



OR MAYBE MORE LIKE THIS



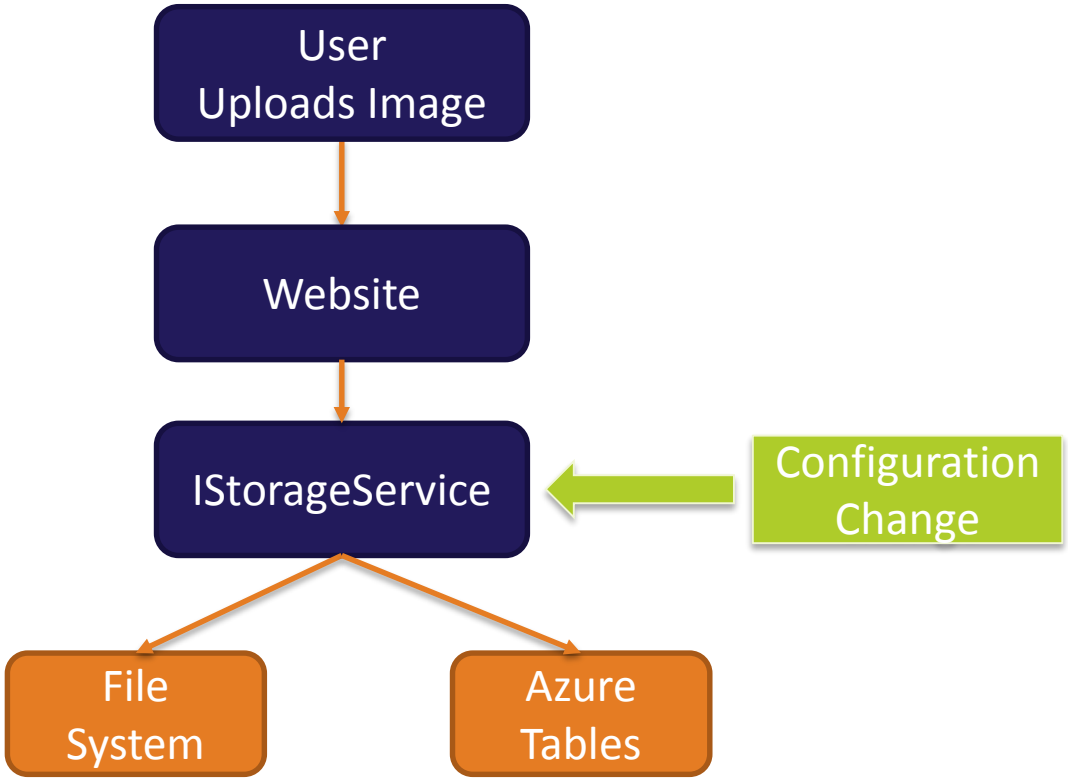
AVOIDING THE "BIG BANG"

- You don't need to migrate all at once
- Start with what comes "out of the box"
  - Cloud Databases (SQL Azure)
  - Caching Providers (SQL Cache, Redis Cache)
  - Virtual Machines
- Setup a VPN
  - Extremely easy, script is done for you
  - Azure has ExpressRoute (L2 connection)

INVERSION OF CONTROL

- Allows your architecture to “emerge”
- Consolidates use of SDKs to support the cloud
- Allows for hybrid support of cloud and on-premise
- Separates code responsibility
- Testability

EXAMPLE: STORAGE



ADDING RESILIENCY

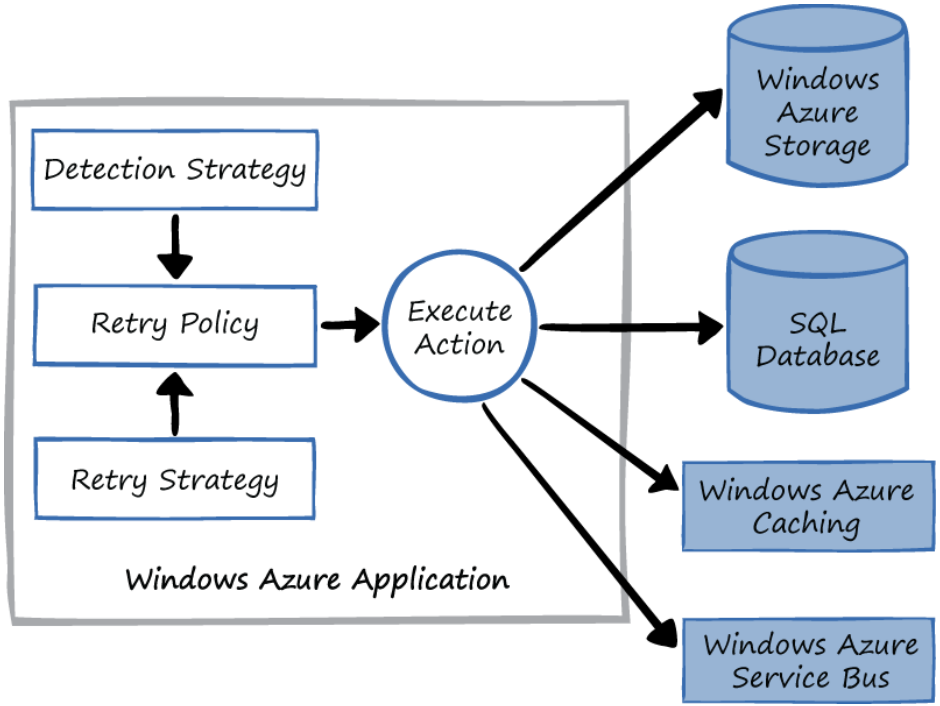
"I'm not quite dead yet...it's just a flesh wound!"

-Monty Python

TRANSIENT ERRORS

- No dependency is available 100% of the time
- Need to separate application failures from transient failures
- Retrying on transient errors produces less error logs
  - BUT it increases wait time! More to come on that…

HOW IT WORKS


Detection Strategy
Retry Policy
Retry Strategy
Execute Action
Windows Azure Application
Windows Azure Storage
SQL Database
Windows Azure Caching
Windows Azure Service Bus

TYPES OF RETRY POLICIES

Retry strategy	Example (intervals between retries in seconds)
Fixed interval	2,2,2,2,2,2
Incremental intervals	2,4,6,8,10,12
Random exponential back-off intervals	2, 3.755, 9.176, 14.306, 31.895

WHAT MAKES THIS WORK

- Asynchronous methods
- Isolated operations
- Known transient failures
- Recording when you “give up”
- Make retry strategies global (use IoC)

CODE EXAMPLE

```
public async Task<ActionResult> Index()
{
    // Step 1 - Setup retry
    var retryStrategy = new ExponentialBackoff(10, TimeSpan.FromSeconds(2), TimeSpan.FromSeconds(20), TimeSpan.FromMilliseconds

    // Step 2  - Create a retry policy
    var retryPolicy = new RetryPolicy<CustomTransientErrorDetectionStrategy>(retryStrategy);

    StockQuote quote;
    try
    {
        // Step 3 - Attempt the action
        quote = await retryPolicy.ExecuteAsync(() => this._stockService.GetQuote("MSFT"));
    }
    catch (Exception ex)
    {
        // Log error here
        Debug.WriteLine("The call failed!, Details: {0}", ex);
        throw;
    }

    return View(quote);
}
```

IT GETS SIMPLER

- Detecting errors can be difficult
  - Dig into errors, check status codes, details in messages, etc.
- Extensions exist to help
  - Caching
  - Database
  - Storage
  - Service Bus

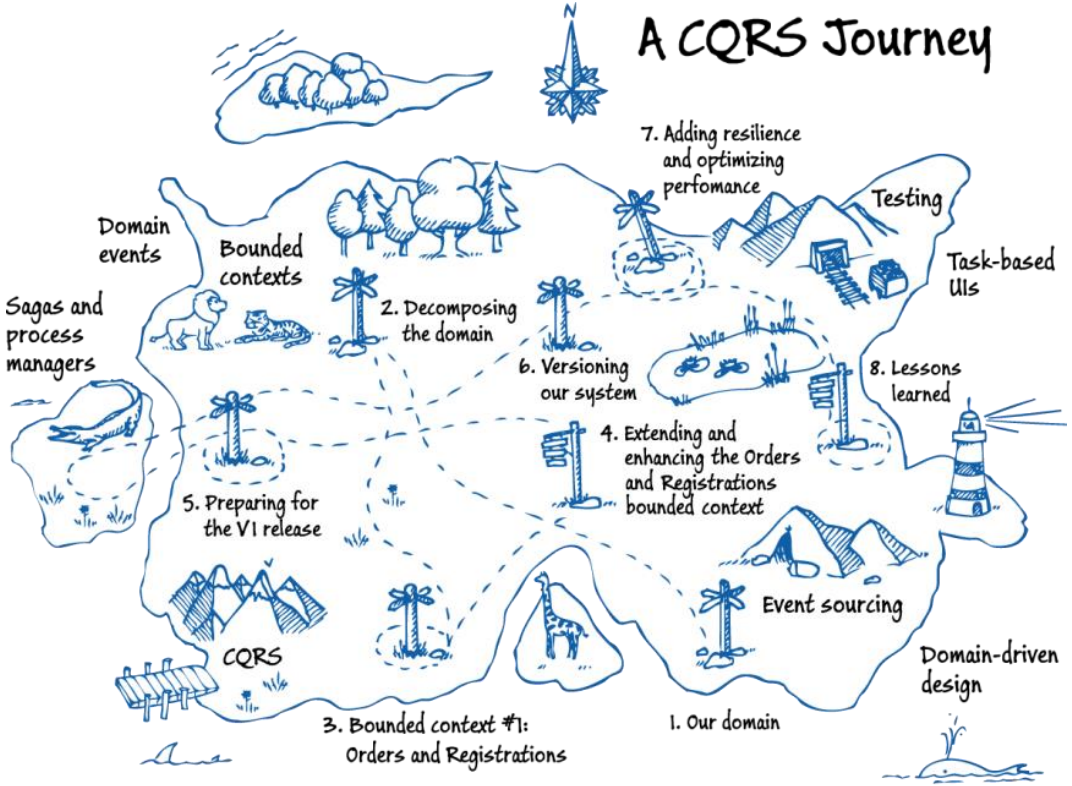
What about authorization?

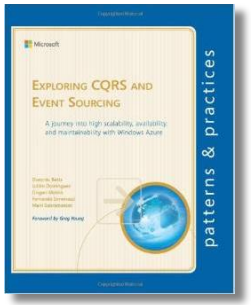
```
public class CustomTransientErrorDetectionStrategy : ITransientErrorDetectionStrategy
{
    public bool IsTransient(Exception ex)
    {
        return (ex is SocketException);
    }
}
```

WHAT ABOUT RESPONSE TIME?

- Retry is good for availability but can make the user wait
- "What happens if we don't want it to fail?"
- "I don't want the site to tip over if things get busy"
- "Auto-scale doesn't work for my application"

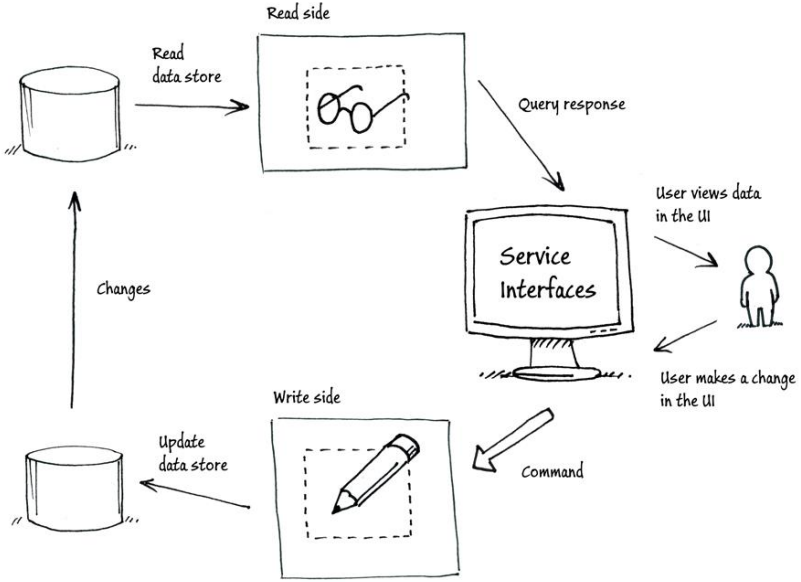
COMMAND & QUERY RESPONSIBILITY SEGREGATION





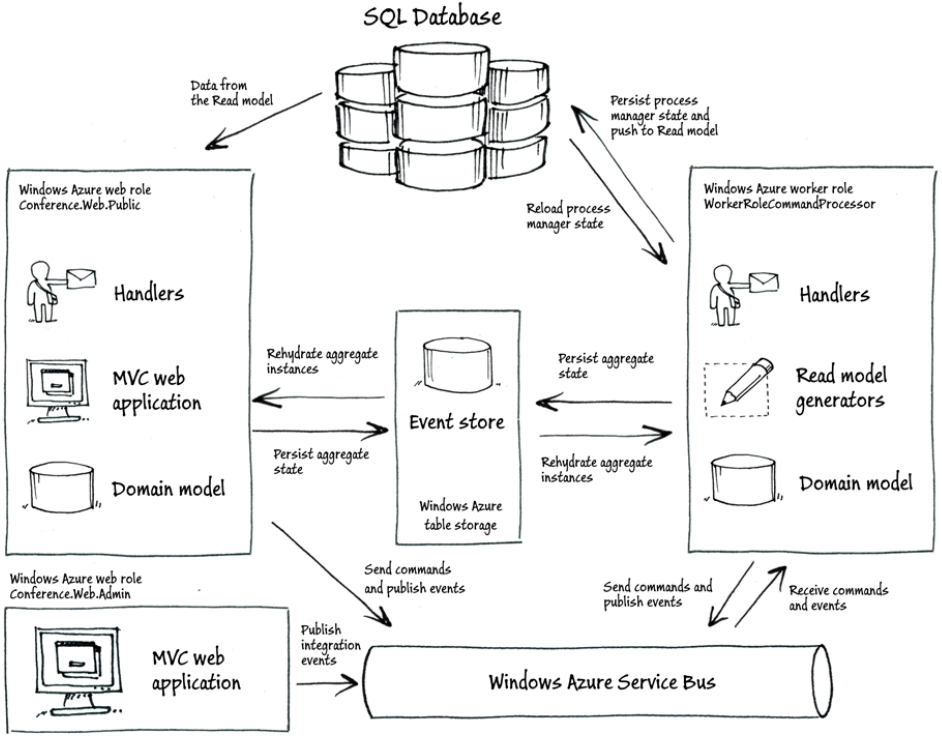
http://msdn.microsoft.com/en-us/library/jj554200.aspx

HOW IT WORKS



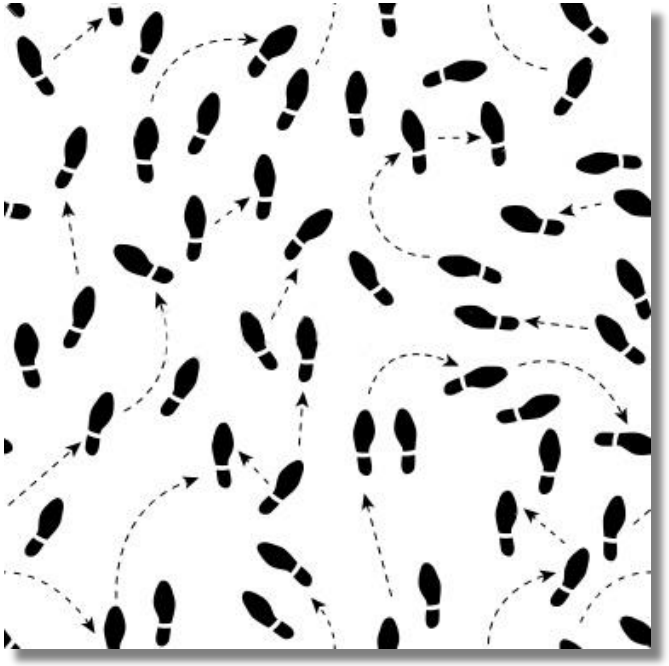
- Updates write to an event store
- Workers process events
- Aggregate state is persisted
- System reads aggregate state

THE CONFERENCE APPLICATION



WOW THAT'S COMPLEX!

- DON'T use this for every system
- Do research and read
- Build the pattern incrementally
  - Pro Tip: Try a small application
- Think about deployment



OTHER PERFORMANCE TIPS

- Caching
  - Static Content
  - Slow-Moving Results
- Compression and Optimization
- Call Separation

DEPLOYMENT AND THE CLOUD

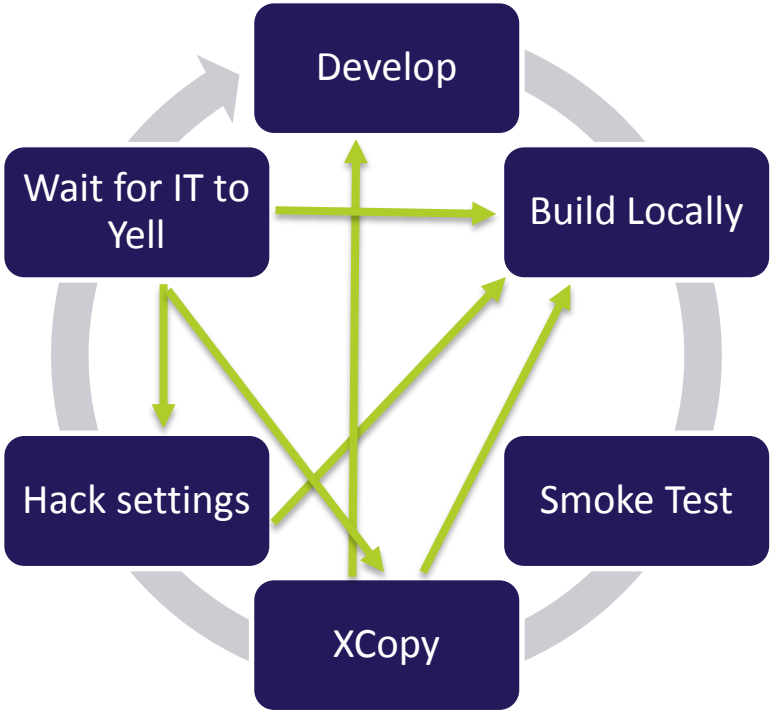
THE CONTINUOUS CONTINUUM

Continuous…	What it Does	Started By
Integration	Builds and Asserts Code Quality	No-One (Automated)
Deployment	Manages Application Releases to an Environment	Anyone (Dev, Qa, Business)
Delivery	Releases New Functionality	Business

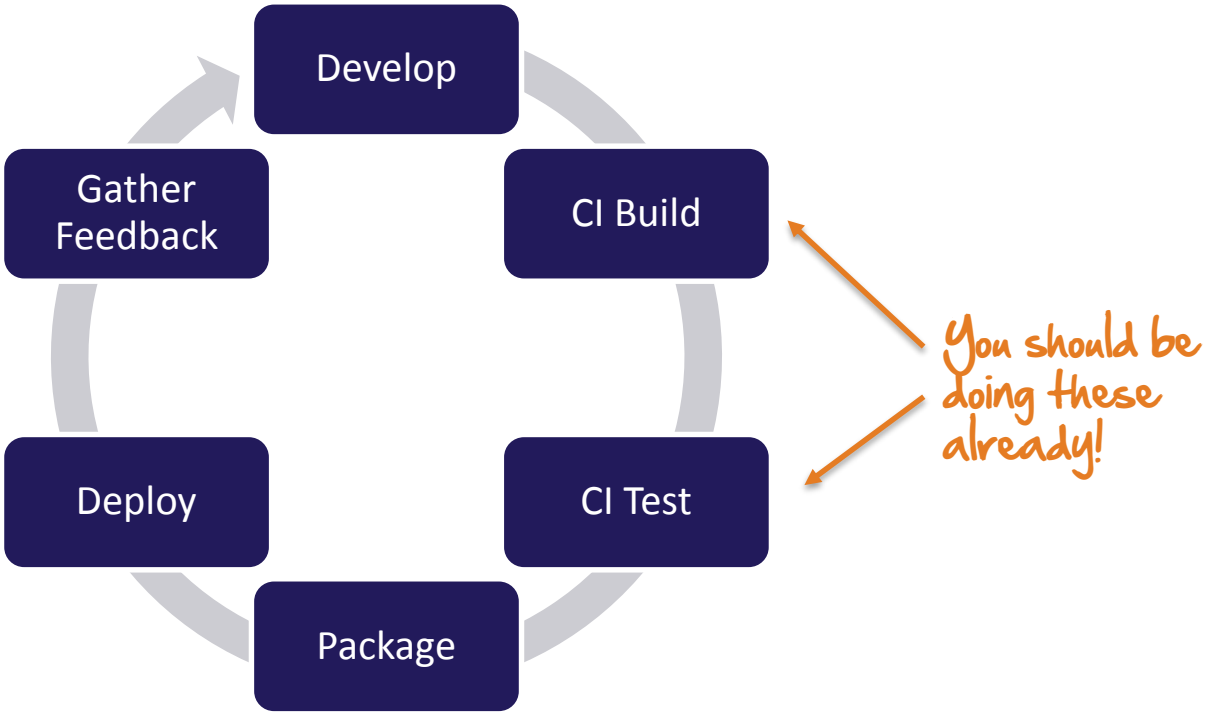
WHY DO DEPLOYMENTS FAIL?

- Large Work Batches
- Large Batches != Deployment Size
- How often do you deploy your software?

MOST DEPLOYMENTS TODAY



THE IDEAL DEPLOYMENT CYCLE

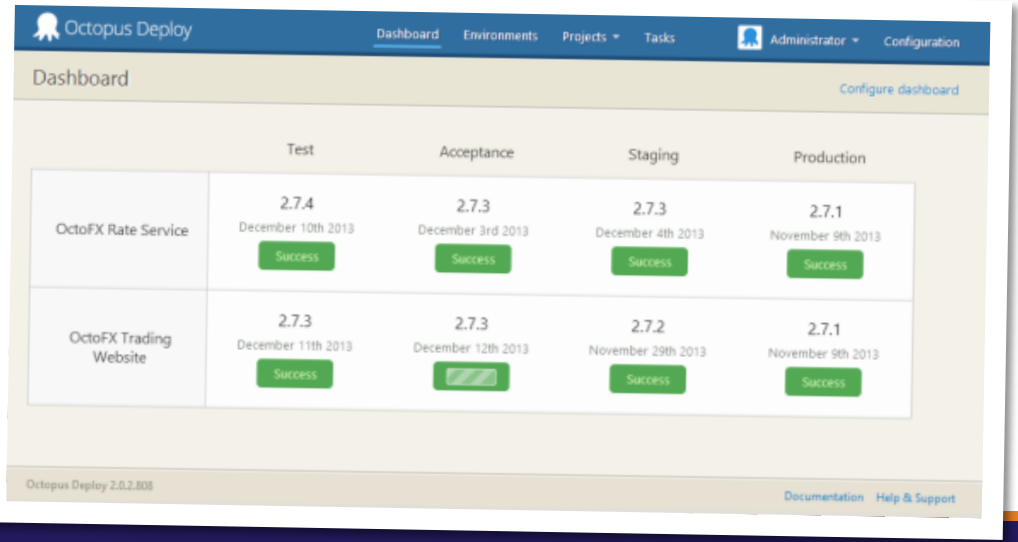


WHAT MAKES THIS WORK?

- Small Work Batches
- Automated Quality Gates
  - Unit Tests, Code Coverage, Quality Checkers (FxCop, etc.)
- Repeatable Process
  - Goal: No Manual Steps During Deployment

DEPLOYMENT: OCTOPUS DEPLOY

- Flexible Deployment Tool
- Deploys both On-Premise and to the Cloud
- Focused on Windows Apps
  - Web Applications
  - Windows Services
  - Click Once Apps
  - Databases
  - PowerShell



WHY CARE ABOUT DEPLOYMENT?

- Deployment to the cloud can be complicated
- Think about deploying everything each time
  - Application
  - Database
  - Service Bus Topics / Queues
  - Storage Container
- Treat your application settings like your code

IT'S NOT JUST ABOUT SOFTWARE

- Automate Your Infrastructure
  - New Development Environments
  - Automated Testing
  - Disaster Recovery / Scalability
- Tooling
  - Puppet
  - Chef
  - PowerShell Desired State

FLEXIBLE RELEASES

FEATURE TOGGLES

- A mechanism to switch between features at runtime
- Separates delivery from deployment
- Typically done at the UI / Service layer
- Scary?

You already have this in your application… User Login and Authorization!

MAKING TOGGLES EXPLICIT

Brittle!

```
HomePageModel homePageModel;

if (ConfigurationManager.AppSettings["ToggleMyFeature"] == "true")
{
    homePageModel = new HomePageModel
    {
        WelcomeMessage = "Well this is different...",
        SubMessage = "Something changed, not sure what",
        Title = "Base Page"
    };
}
else
{
    homePageModel = new HomePageModel
    {
        WelcomeMessage = "Welcome to my page!",
        SubMessage = "Now this is cool :)",
        Title = "Home Page"
    };
}
```

MAKING TOGGLES EXPLICIT

Easier to Refactor

```
HomePageModel homePageModel;

if (ToggleManager.IsEnabled<ToggleMyFeature>())
{
    homePageModel = new HomePageModel
    {
        WelcomeMessage = "Well this is different...",
        SubMessage = "Something changed, not sure what",
        Title = "Base Page"
    };
}
else
{
    homePageModel = new HomePageModel
    {
        WelcomeMessage = "Welcome to my page!",
        SubMessage = "Now this is cool :)",
        Title = "Home Page"
    };
}
```

It's a Class!

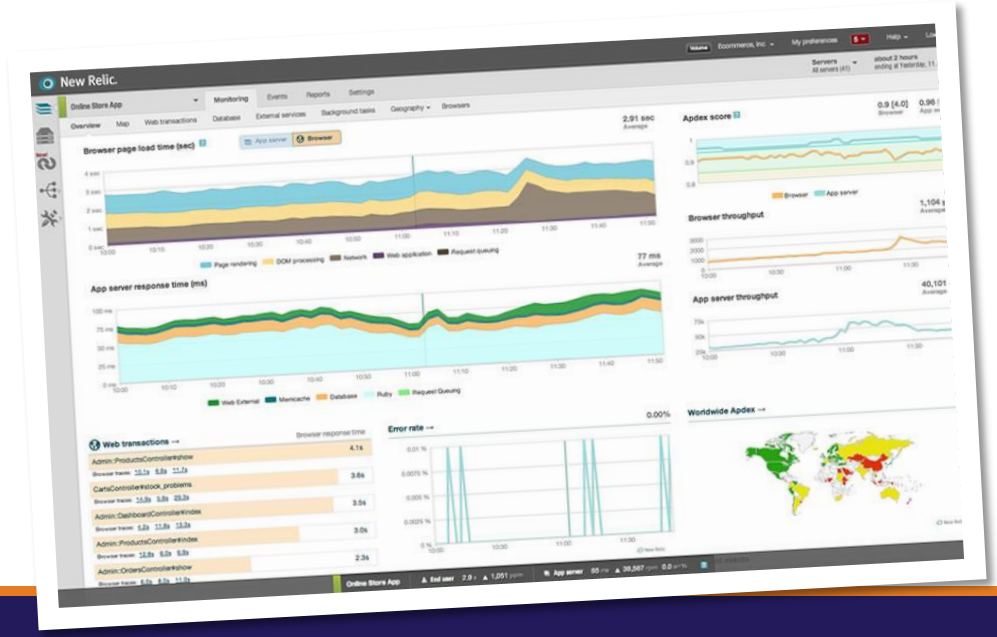
TOGGLES AND DEPLOYMENT

- Toggles do stuff! = Has a performance impact
- Must correlate changes to runtime feedback
- Flip toggles via deployments
- Nuget package: FeatureSwitch

GETTING FEEDBACK

RUNTIME FEEDBACK

- Instrument your applications at runtime!
- Many tools available
  - New Relic
  - Application Insights
  - Splunk
  - Raygun.io
- Include User Analytics
  - Google
  - All above tools

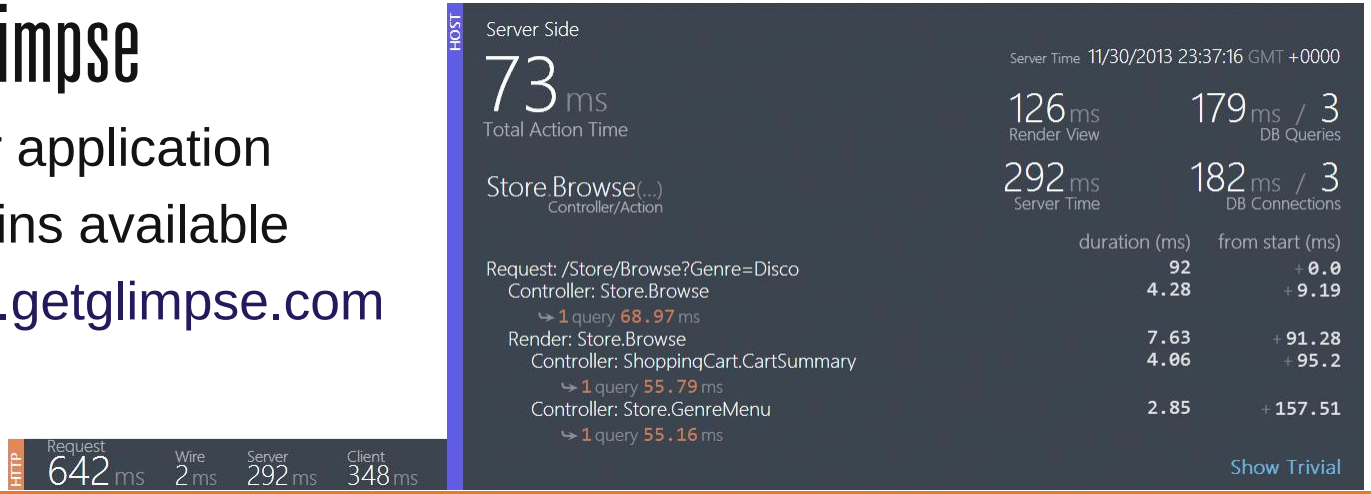


DIG DEEPER FOR DATA

- Monitoring tools have APIs for tracing / logging
- Instrument key transactions in your system
- Track performance end-over-end for deployments
- Choose a logging framework and leverage it!
  - Log to a common location

ANALYZE EARLIER

- Find errors during development
- Make performance reviews part of your DoD
- Example: Glimpse
  - Trace your application
  - Many plugins available
  - http://www.getglimpse.com



TAKE AWAY CONCEPTS

- Scaling enhancements can always be done incrementally
- Design your system around performance
- Provide fast feedback
- Treat deployment settings like code
- Separate deployments from releases

"Deployments are like exercise, the more you do them the less it hurts"

-Dan Piessens

RESOURCES

- Moving Applications to the Cloud 3rd Edition
  - http://msdn.microsoft.com/en-us/library/ff728592.aspx
- Building Hybrid Applications in the Cloud on Microsoft Azure
  - http://msdn.microsoft.com/en-us/library/hh871440.aspx
- CQRS Journey
  - http://msdn.microsoft.com/en-us/library/jj554200.aspx
- Transient Fault Handling Core
  - http://msdn.microsoft.com/en-us/library/hh675232.aspx
- Octopus Deploy
  - http://www.octopusdeploy.com


8/10/15 - 8/12/15

QUESTIONS?

Thank You!

Email: dan.piessens@centare.com Blog: www.danpiessens.com Twitter: @dpiessens