

Ubiquitous Communication with the Windows Azure AppFabric Service Bus

Christian Weyer, thinktexture

christian.weyer@thinktexture.com



Christian Weyer and thinktecture

- Solution architect and principal consultant at thinktecture
- Focus on
 - distributed applications
 - service orientation, workflows
 - cloud computing
 - interoperability
 - end-to-end solutions
 - Windows Server, WCF, WF, MSMQ, Windows Azure platform
- Microsoft MVP for Windows Azure (Architecture)
- Independent Microsoft Regional Director for Germany
- <http://blogs.thinktecture.com/cweyer>
- christian.weyer@thinktecture.com

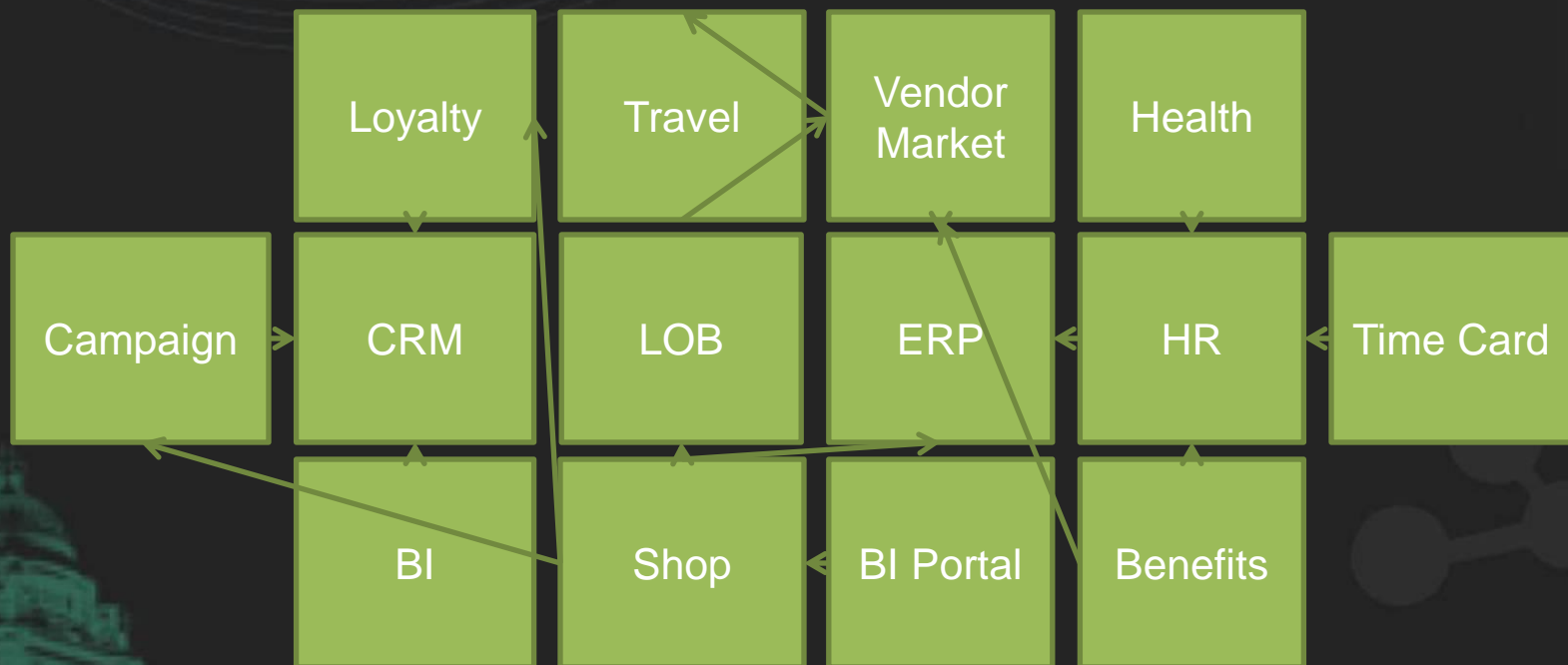


Let's Ride the Bus

- Motivation
- Scenarios & Challenges
- Service Bus Relay & Messaging
- WCF Bindings
- Message Buffers
- Misc



Integrated Enterprise



Very Few Go 'All In' On The Cloud (aka Hybrid)

The diagram illustrates a hybrid cloud architecture. A horizontal line separates the internal 'Security Realm' from the external 'Sourced Vendor Market'. Inside the Security Realm, various business systems are interconnected: Campaign, CRM, LOB, ERP, HR, Time Card, BI, Shop, BI Portal, and Benefits. These systems are connected by a network of green arrows. Outside the Security Realm, the 'Sourced Vendor Market' is represented by a red box. Yellow arrows with question marks point from the Travel and ERP systems inside the Security Realm to the Sourced Vendor Market, indicating potential integration or data exchange.

Firewall

NAT

Security Realm

Sourced Vendor Market

Loyalty

Travel

Health

Campaign

CRM

LOB

ERP

HR

Time Card

BI

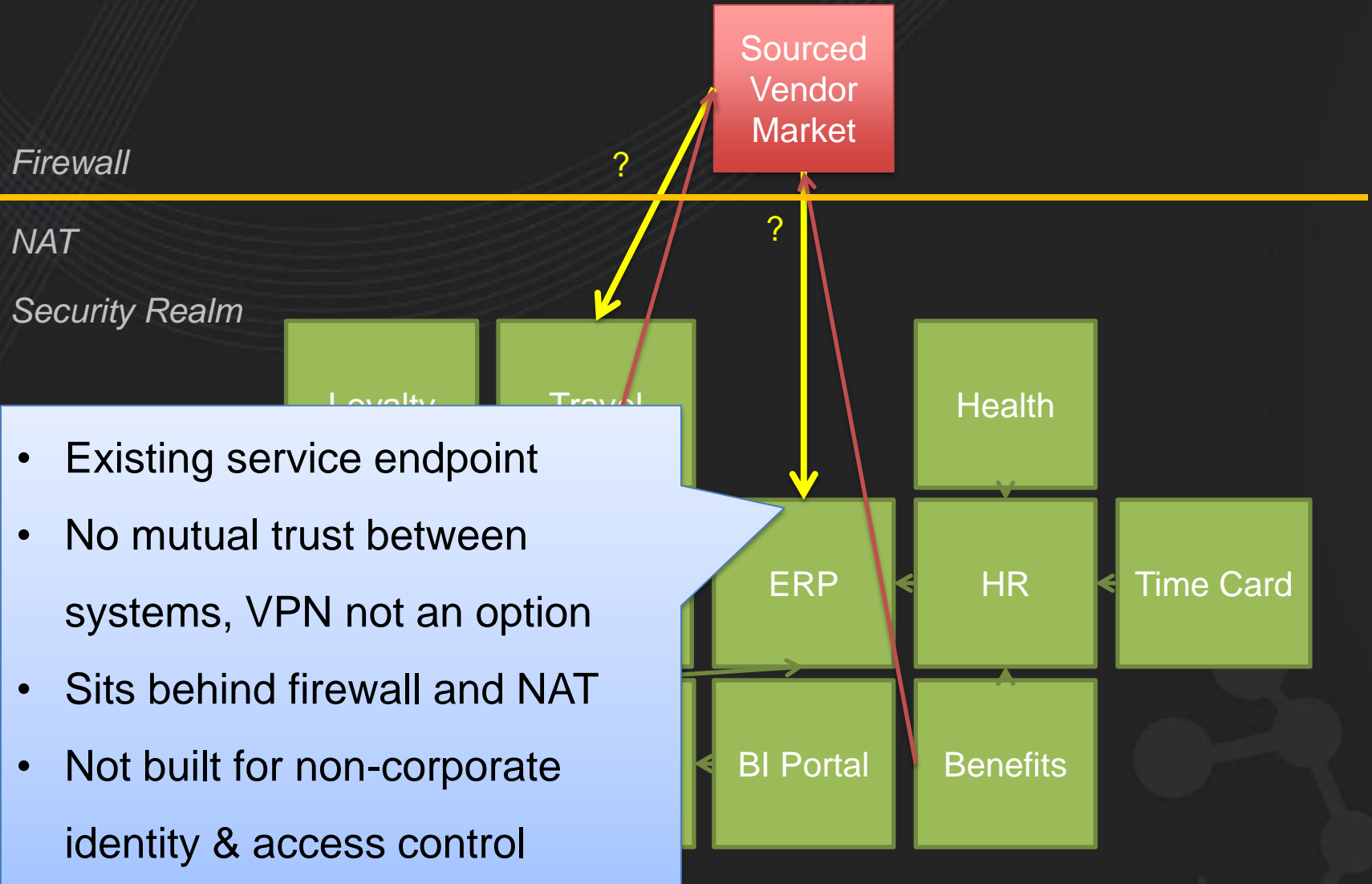
Shop

BI Portal

Benefits



How Can This Work?



Wanted Patterns & Usage Scenarios

- Service remototing
 - syndication
 - peer-to-peer
- Multi-casting, Pub/Sub
- Callbacks, notifications
- Light-weight orchestration
- Remote control
- Tunneling



Communication & Connectivity Challenges

- Not every application is a server application
 - a number of different app styles need to communicate
 - proper DMZ is seldom in place
- Advanced communication patterns
 - peer-to-peer
 - publish/subscribe
 - enabled for cross-platform and -technology
- IPv4 address shortage
 - dynamic IP address allocation
 - Network Address Translation (NAT)



Sender



Dynamic IP

Network Address Translation

Network Firewall

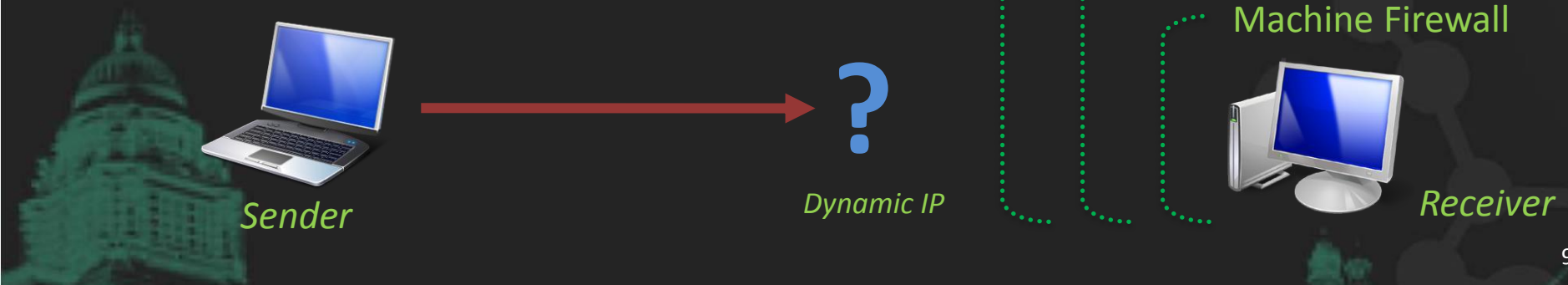
Machine Firewall



Receiver

Dealing with Connectivity Challenges

- Dynamic DNS
 - NAT port mappings / UPnP
 - Open inbound firewall ports
 - Virtual private networks (VPN)
-
- Problem
 - brittle, difficult, insecure, impractical
 - recurring patterns of workarounds



Communication Swiss Army Knife



Windows Azure AppFabric

- Developer infrastructure services in the Cloud



Access Control



Caching



Service Bus



Composite App

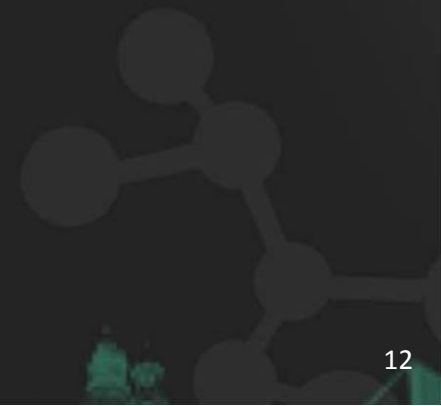


Integration

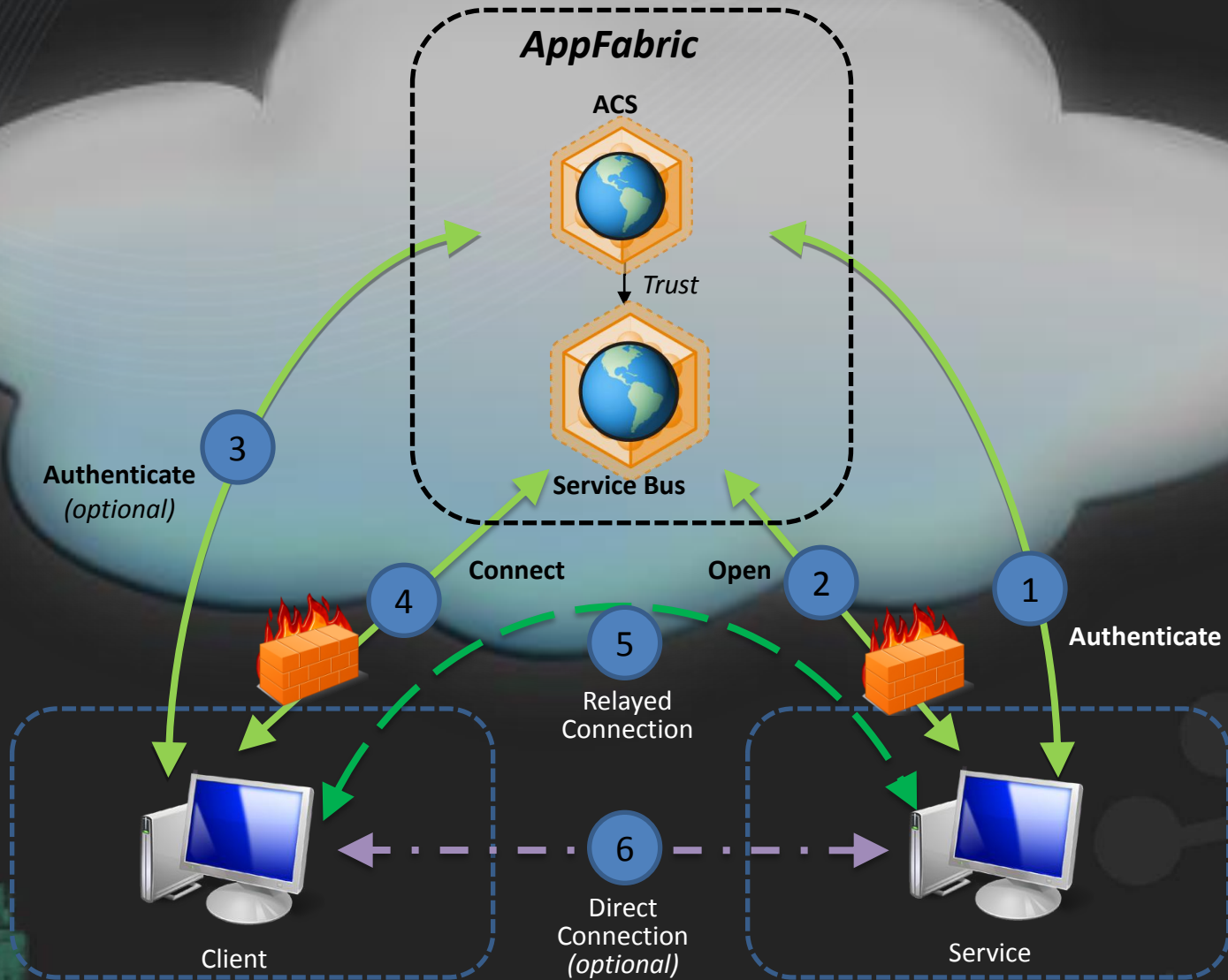
Windows Azure AppFabric Service Bus

- Pervasive, secure connectivity for services
 - no matter where they are hosted, where they are consumed
- Runs on Windows Azure Compute and Storage
- Available in data centers all over the world

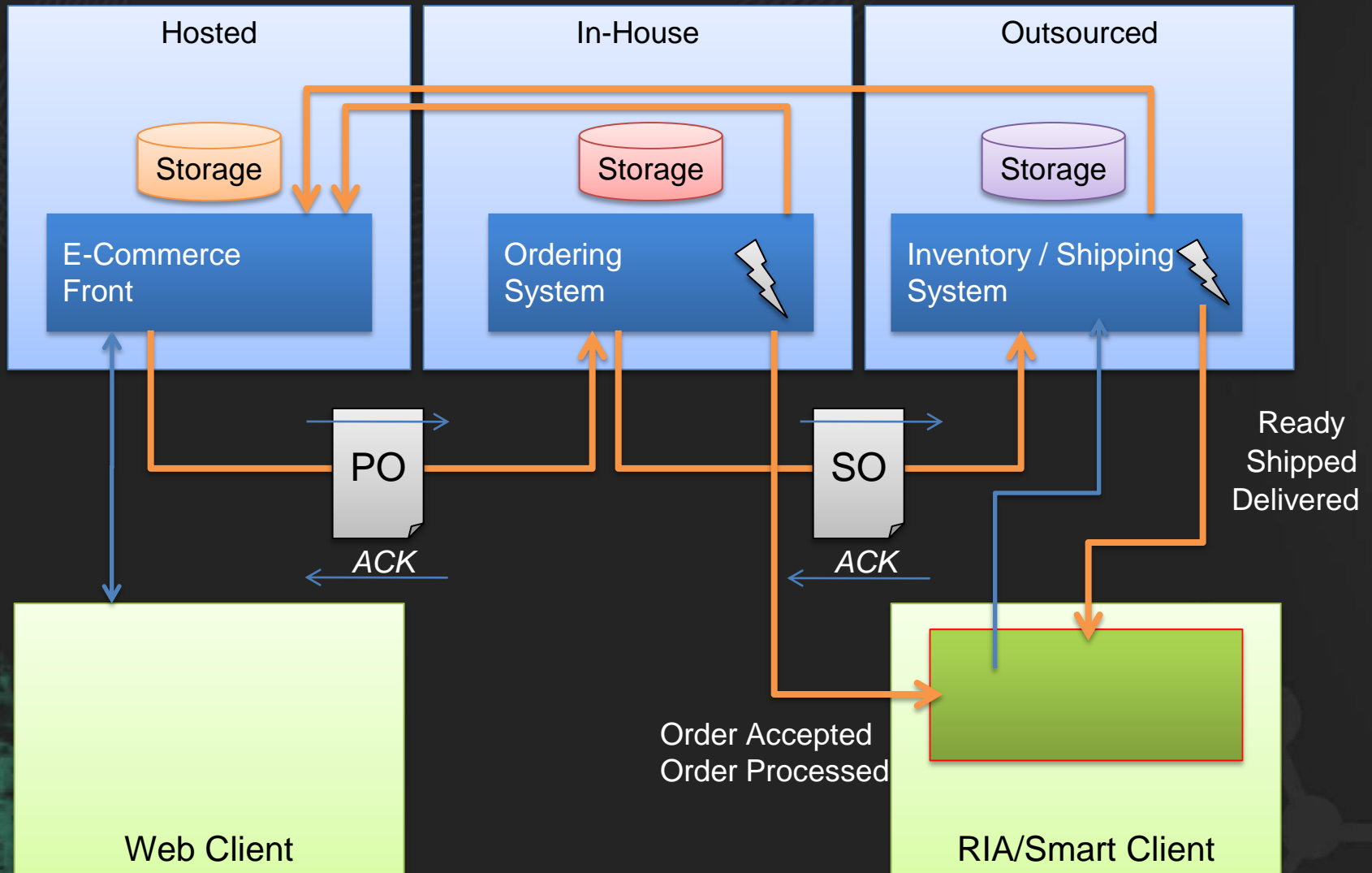
- Today, three main concepts
 - naming system
 - service registry
 - secure connectivity & messaging



Connectivity Through the Cloud



Service Remoting



Talking to the Service Bus

- Two kinds of APIs for consuming the Service Bus
 - WCF-enabled clients use the Service Bus SDK
 - All others can leverage REST programming features through message buffers



WCF Bindings

- Primary programming model: WCF
- Family of bindings for the Service Bus

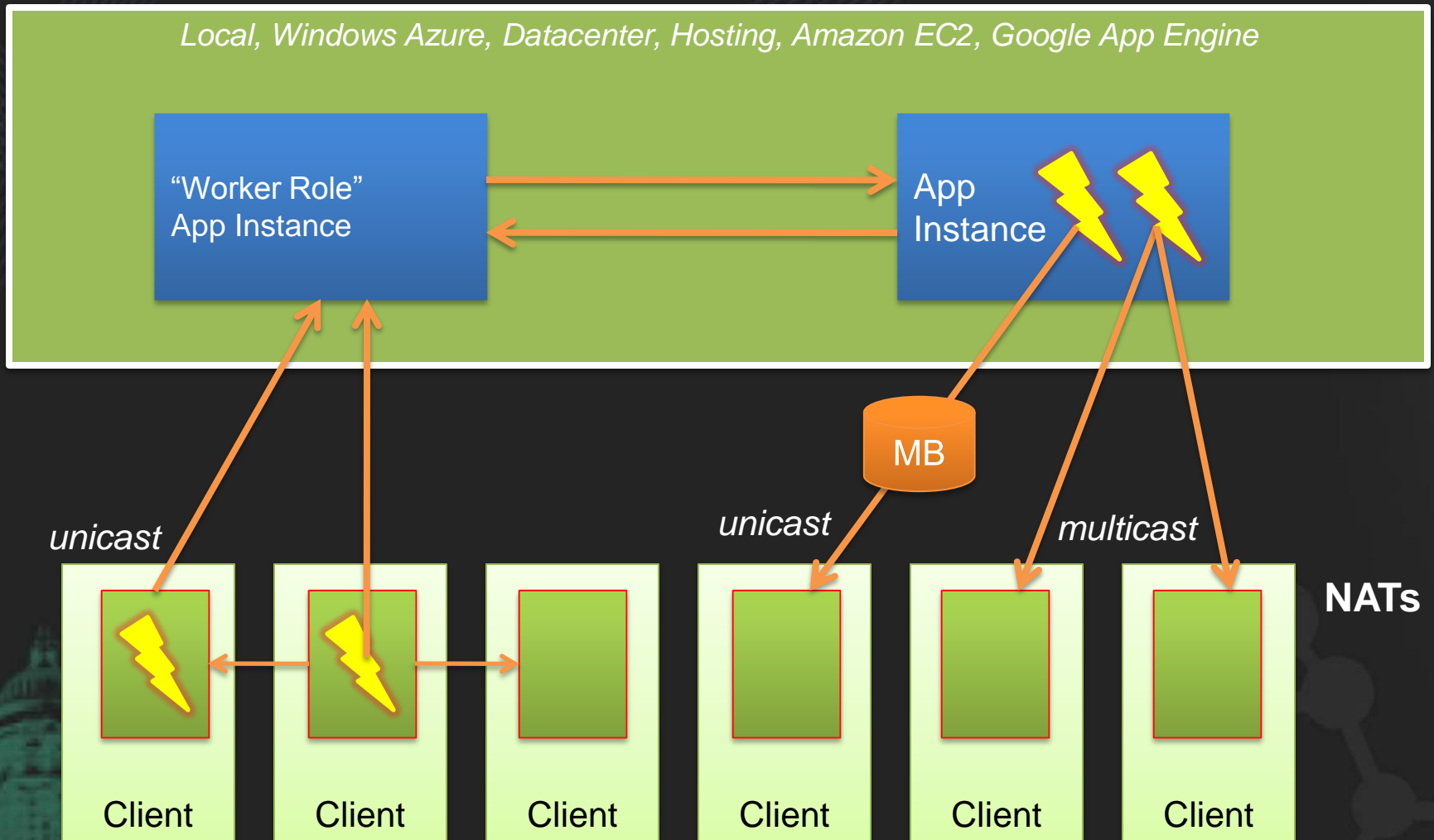
Corresponding WCF Binding	Service Bus Relay Binding
BasicHttpBinding	BasicHttpRelayBinding
WebHttpBinding	WebHttpRelayBinding
WS2007HttpBinding	WS2007HttpRelayBinding
NetTcpBinding	NetTcpRelayBinding
n/a	NetOnewayRelayBinding
n/a	NetEventRelayBinding

WCF Binding Primitives

- All bindings are based on a set of transport primitives
- Build your own custom binding

Service Bus Relay Binding	Transport Binding Element
BasicHttpRelayBinding	Http(s)RelayTransportBindingElement
WebHttpRelayBinding	Http(s)RelayTransportBindingElement
WS2007HttpRelayBinding	Http(s)RelayTransportBindingElement
NetTcpRelayBinding	TcpRelayTransportBindingElement
NetOnewayRelayBinding	OnewayRelayTransportBindingElement
NetEventRelayBinding	OnewayRelayTransportBindingElement

Eventing



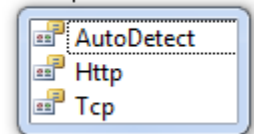
Port Configuration

- Corporate firewalls often block outgoing traffic
 - except port 80 and 443
- Service Bus bindings need open outgoing ports in default mode
- Port range is specified at 9350-9353
 - NetTcpRelayBinding uses 9352 and 9353
 - all others use 9350 and 9351
- Even when ports are open some obstacles may still be in your way
 - HTTP 1.0-only proxies
 - 'intelligently' filtering proxies
 - need for WinHTTP proxy settings

Connectivity Modes

- System connectivity mode indicates whether to use TCP or HTTP to negotiate connections to the Service Bus
 - if only port 80 and/or 443 are available you can force the Service Bus to use HTTP always
- HTTP mode uses web streams
 - pair of HTTP requests bundled into fast bi-directional binary channel

```
ServiceBusEnvironment.SystemConnectivity.Mode = ConnectivityMode.
```



Connection Modes

- Service Bus bindings can try to establish direct connections through firewalls
 - hybrid connectivity mode (relayed is default)
 - event on channel property available to determine connection change

```
<bindings>
  <netTcpRelayBinding>
    <binding name="hybrid" connectionMode="Hybrid">
    </binding>
  </netTcpRelayBinding>
</bindings>
```

Message Buffers

- Small, temporary caches where messages can be held for a short time until they are retrieved
 - accessible to applications using HTTP
 - long poll for messages
- Message buffers are governed by policies
- REST protocol to expose various operations on the message buffer
 - creating a message buffer
 - sending a message to the message buffer
 - retrieving a message from the message buffer
 - deleting a message buffer
- .NET API available

Message Buffer API

// Create a message buffer

```
var client = new WebClient();
client.BaseAddress = string.Format("https://{0}.servicebus.windows.net/{1}/",
    serviceNameSpace, bufferName);
client.Headers[HttpRequestHeader.ContentType] =
    "application/atom+xml;type=entry;charset=utf-8";
client.Headers[HttpRequestHeader.Authorization] = authHeaderValue;
client.UploadData(String.Empty, "PUT", Encoding.UTF8.GetBytes(policy));
```

// Send a message to the message buffer

```
client.Headers[HttpRequestHeader.ContentType] = "text/xml";
client.Headers[HttpRequestHeader.Authorization] = authHeaderValue;
client.UploadData("messages?timeout=20", "POST", Encoding.UTF8.GetBytes(
    "<msg1>This is message #1</msg1>"));
```

// Retrieve message

```
client.Headers[HttpRequestHeader.Authorization] = authHeaderValue;
string payload = Encoding.UTF8.GetString(client.UploadData(
    "messages/head?timeout=20", "DELETE", new byte[0]));
```

// Delete the message buffer

```
client.Headers[HttpRequestHeader.Authorization] = authHeaderValue;
client.UploadData(String.Empty, "DELETE", new byte[0]);
```

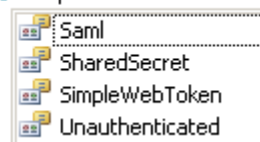
Hosting

- WCF Services with Service Bus-based bindings can only be self-hosted in a reliable way
- IIS/WAS hosting possible but no immediate integration
 - can listen from an active IIS hosted AppDomain
 - Windows Server AppFabric Auto-Start can help
- Key reason
 - no extensibility mechanism to get between IIS and HTTP.SYS for HTTP traffic
 - need to be able to replace HTTP.SYS for Service Bus to work with IIS
 - activation for other bindings not yet implemented

Security

- Service Bus mostly transparent for WCF security
 - use WS-Security for end-to-end protection
 - SSL endpoints on the Service Bus
- Service Bus tightly integrated with ACS
 - ACS emits Send/Listen/Manage claims
- Listen claim in security token needed for opening endpoints
 - baked-in issuer name / issuer key pair
 - can federate with any STS through ACS
- Several supported token formats
 - shared secret, SWT, SAML

```
tokenProvider.CredentialType = TransportClientCredentialType.Saml;  
tokenProvider.Credentials.Saml.SamlToken = samlToken;
```



Prizing

- ACS pricing per transaction
 - 1.99 \$ per 100K transactions
- Service Bus pricing per ,connection'
 - \$3.99 per connection-month
 - data transfer costs extra
- You can buy packages
- How does it work?
 - max. amount of concurrent connections per day (monitored in a 5 minutes interval)
 - invoiced based on average of this connection number

Recommendations

- Design explicitly for the Service Bus
- Understand design of Service Bus namespaces, ACS scopes and rules
- Service Bus usage currently priced based on number of simultaneous connections
 - may have implications on communication design



Summary

- Pervasive, secure connectivity for services
 - “Communication Swiss army knife”
 - secure NAT traversal
 - integrated with Access Control Service
- Enables new interesting applications
- WCF-based programming model
 - enables new communication patterns, e.g. pub/sub
- SOAP and HTTP protocol options
 - not only for .NET clients
- The Service Bus keeps evolving – new features ahead
 - powerful queues

Resources

- christian.weyer@thinktecture.com
- <http://weblogs.thinktecture.com/cweyer>
- Windows Azure blog category
 - <http://weblogs.thinktecture.com/cweyer/azure/>



Christian Weyer and thinktecture

- In-depth consulting and training for software architects & developers
- Focus on
 - distributed applications
 - service orientation, workflows
 - cloud computing
 - interoperability
 - end-to-end solutions
 - Windows Server, WCF, WF, MSMQ, Windows Azure platform
- Contact: christian.weyer@thinktecture.com

Get Started with Windows Azure For Free Today!

- **MSDN Subscriber**

- Activate Your Free Included MSDN Benefits via <http://tinyurl.com/activatemsdnazurebenefits>

- **Individual:**

- Get a Free Azure Introductory via <http://tinyurl.com/freeintroazureoffer>
 - Free Computation hours and Storage
- Get 30 Days Free Windows Azure via <http://www.windowsazurepass.com>
 - Select Belgium and enter Promo code: AZP001

- **Partner**

- Get free monthly access to Azure with Partner Cloud Essentials via <http://www.microsoftcloudpartner.com/>

Start Developing on the Windows Azure Platform



1. Activate your Benefits (see previous slide)
2. Get the Tools via
<http://tinyurl.com/toolsforazure>
3. First learn how to create an application via
<http://tinyurl.com/deployazureapplication>

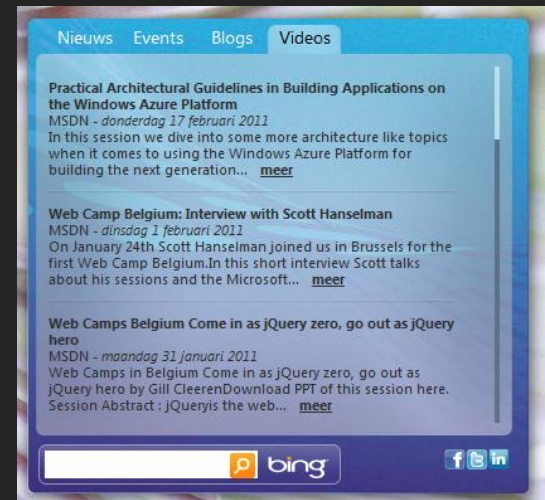
Stay up to date with MSDN Belux

- Register for our newsletters and stay up to date:
<http://www.msdn-newsletters.be>
 - Technical updates
 - Event announcements and registration
 - Top downloads
- Follow our blog
<http://blogs.msdn.com/belux>
- Join us on Facebook
<http://www.facebook.com/msdnbe>
<http://www.facebook.com/msdnbelux>
- LinkedIn: <http://linkd.in/msdnbelux/>
- Twitter: [@msdnbelux](https://twitter.com/msdnbelux)

Download

MSDN/TechNet Desktop Gadget

<http://bit.ly/msdntngadget>



TechDays 2011 On-Demand

- **Watch** this session on-demand via Channel9
<http://channel9.msdn.com/belux>
- Download to your favorite MP3 or video player
- Get access to slides and recommended resources by the speakers



THANK YOU

