Integration Patterns

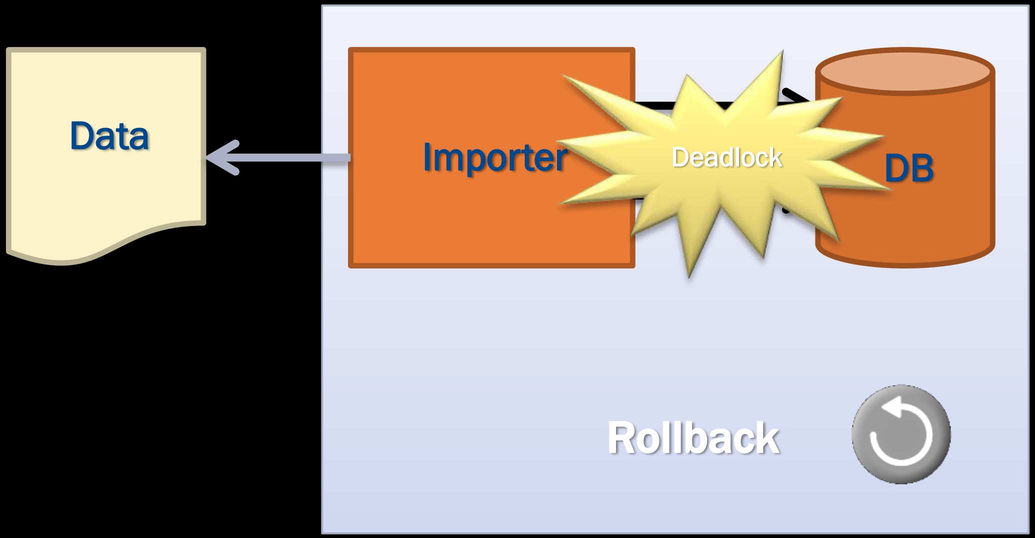
Processing bulk data

THE DAILY AS/400 DUMP

Characteristics

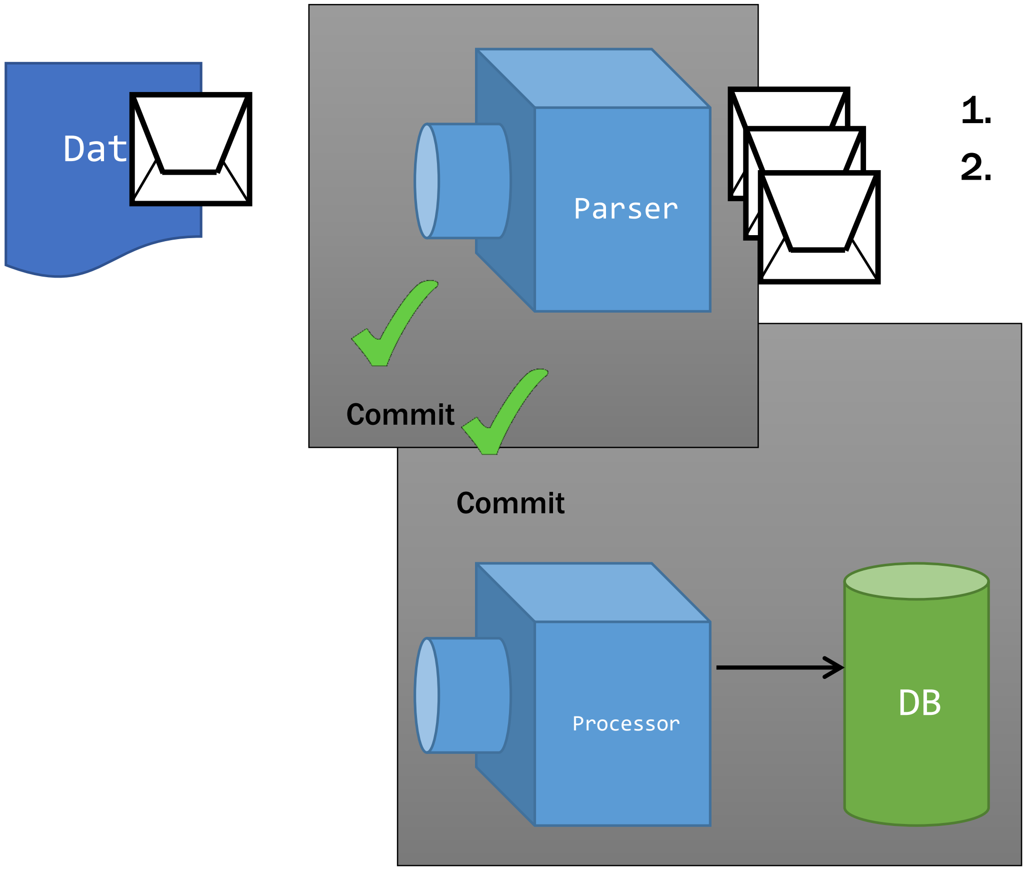
- 1/day dump of entire SKU catalog
- Flat file
- 1.4M rows daily and growing
- Fill in details for support website

Consuming Files



What made it through?

Consuming Files

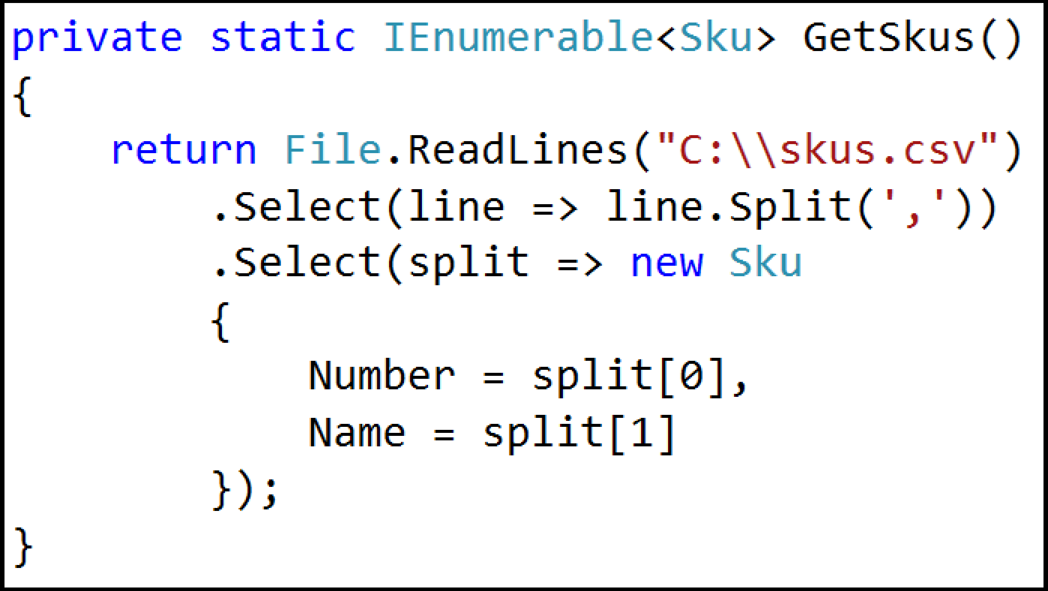


1. Process File
2. Process Line

```csharp
var fileContents = File.ReadAllLines(path);

var productsPurchased = fileContents
    .Select(line => line.Split(','))
    .Select(split => new RecordProductPurchased
    {
        TransactionId = split[0],
        Quantity = Convert.ToInt32(split[1]),
        Price = Convert.ToDecimal(split[2]),
        Sku = split[3]
    })
    .ToArray();

foreach (var item in productsPurchased)
{
    Bus.Send(item);
}
```

```
public void Handle(RecordProductPurchased message)
{
    using (var conn = new SqlConnection(_connectionString))
    {
        conn.Open();

        using (var tx = conn.BeginTransaction())
        using (var cmd = new SqlCommand(
            "INSERT ProductPurchased" +
            "(TransactionId, Quantity, Price, Sku)" +
            " VALUES (@TransactionId, @Quantity, @Price, @Sku)", conn, tx))
        {
            cmd.Parameters.AddWithValue("TransactionId", message.TransactionId);
            cmd.Parameters.AddWithValue("Quantity", message.Quantity);
            cmd.Parameters.AddWithValue("Price", message.Price);
            cmd.Parameters.AddWithValue("Sku", message.Sku);
            cmd.ExecuteNonQuery();

            tx.Commit();
        }
    }
```

Streaming Bulk Copy



```csharp
private static IEnumerable<Sku> GetSkus()
{
    return File.ReadLines("C:\\skus.csv")
        .Select(line => line.Split(','))
        .Select(split => new Sku
        {
            Number = split[0],
            Name = split[1]
        });
}
```

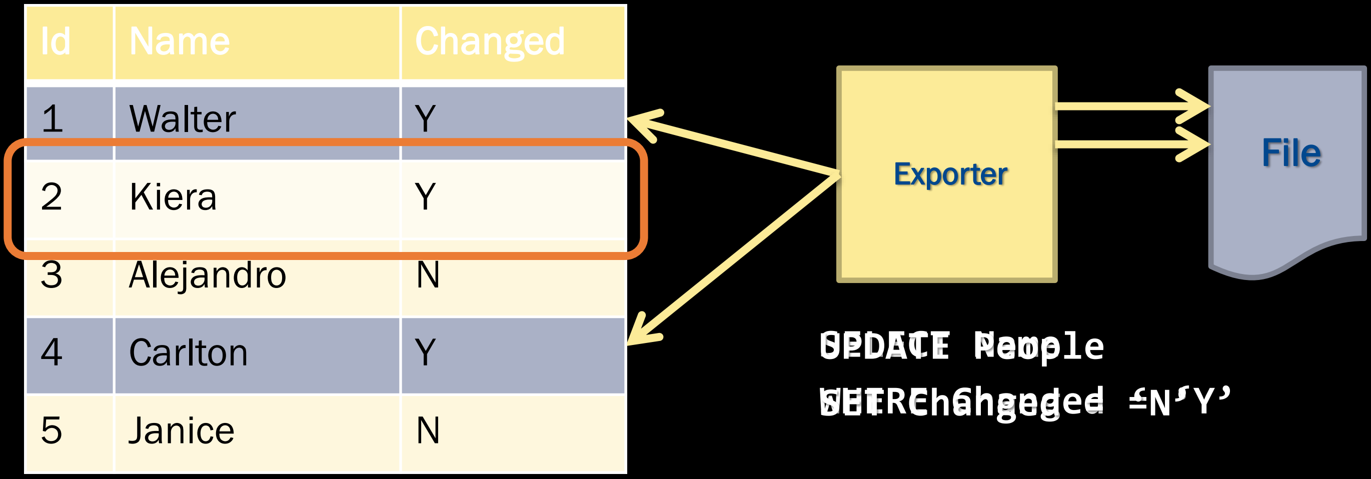
Streaming Bulk Copy

```
var mapping = MapBuilder
    .MapAllProperties<Sku>()
    .DestinationTable("Skus")
    .Build();

using (var bulkWriter = mapping.CreateBulkWriter("ConnectionString"))
{
    var skus = GetSkus();
    bulkWriter.WriteToDatabase(skus);
}
```

Exporting Bulk Data

DATA WAREHOUSE EXPORTS

Producing Files



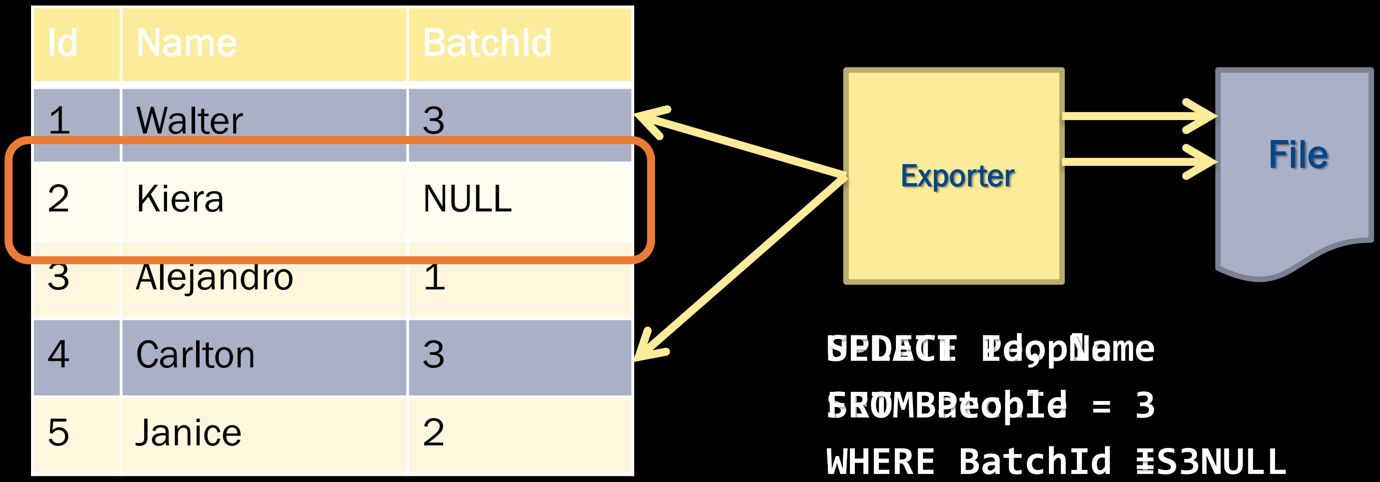
Id	Name	Changed
1	Walter	Y
2	Kiera	Y
3	Alejandro	N
4	Carlton	Y
5	Janice	N

```
SELECT Name
WHERE Changed = 'Y'
```

```
UPDATE People
SET Changed = 'N'
```

We're missing a change

Producing Files



Id	Name	BatchId
1	Walter	3
2	Kiera	NULL
3	Alejandro	1
4	Carlton	3
5	Janice	2

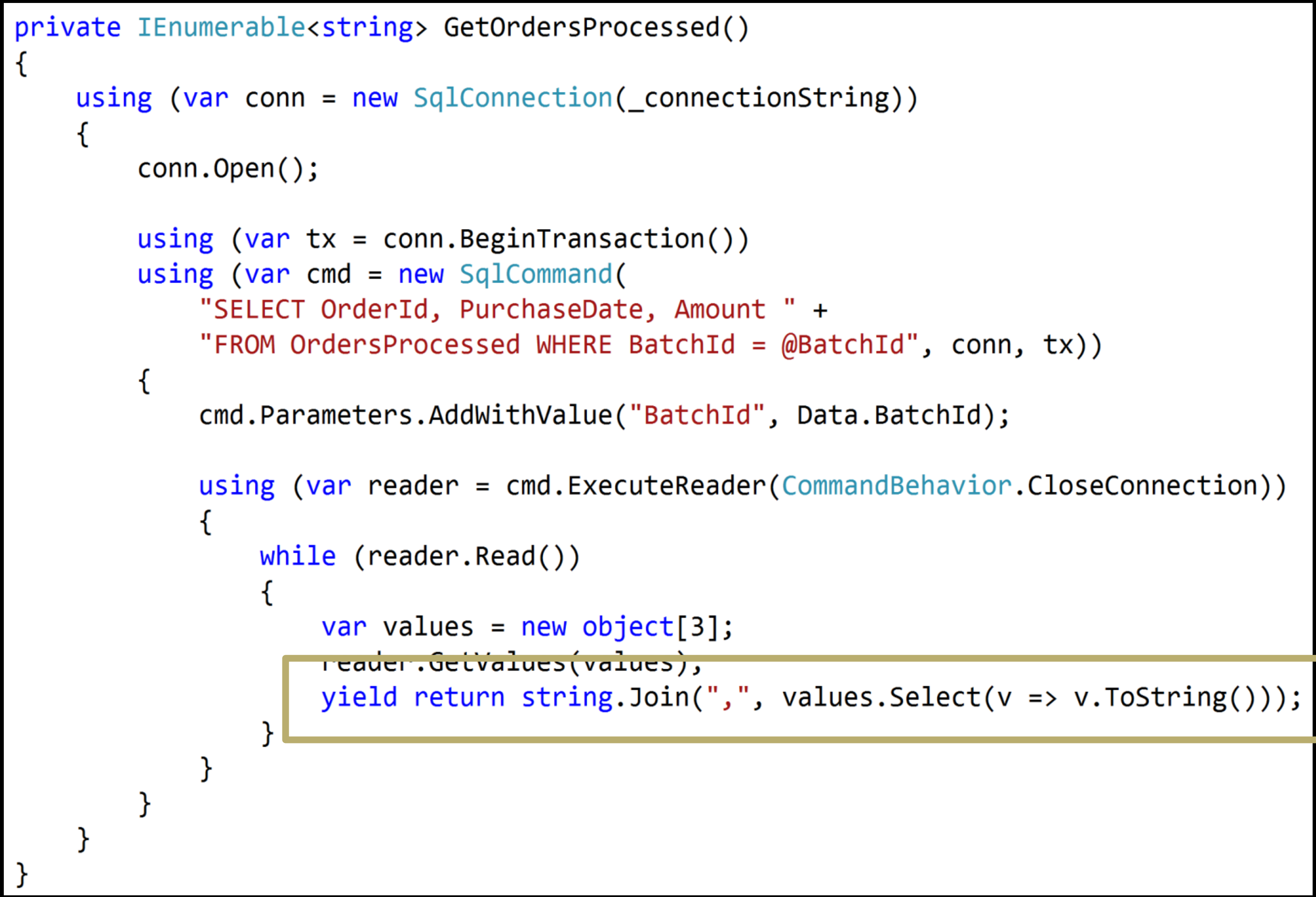
```
SELECT Id, Name
FROM People
WHERE BatchId = 3
```

```
UPDATE People
SET BatchId = 3
WHERE BatchId IS NULL
```

1. **Mark records to export**
2. **Export marked records**

```
public void Handle(ExportFile message)
{
    Data.BatchId = message.BatchId;

    MarkOrdersToExport();

    Bus.SendLocal(new ExportFileContents {BatchId = Data.BatchId});
}

private void MarkOrdersToExport()
{
    using (var conn = new SqlConnection(_connectionString))
    {
        conn.Open();

        using (var tx = conn.BeginTransaction())
        using (var cmd = new SqlCommand(
            "UPDATE OrdersProcessed " +
            "SET BatchId = @BatchId", conn, tx))
        {
            cmd.Parameters.AddWithValue("BatchId", Data.BatchId);
            cmd.ExecuteNonQuery();

            tx.Commit();
        }
    }
}
```

```
public void Handle(ExportFileContents message)
{
    var contents = GetOrdersProcessed();

    WriteContentsToFile(contents);

    MarkAsComplete();
}
```

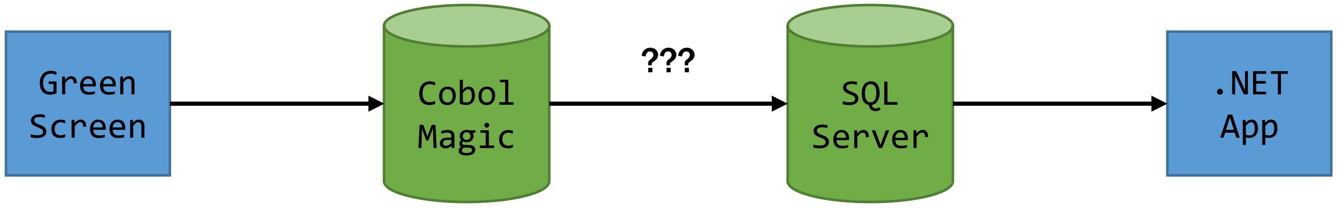


```
private IEnumerable<string> GetOrdersProcessed()
{
    using (var conn = new SqlConnection(_connectionString))
    {
        conn.Open();

        using (var tx = conn.BeginTransaction())
        using (var cmd = new SqlCommand(
            "SELECT OrderId, PurchaseDate, Amount " +
            "FROM OrdersProcessed WHERE BatchId = @BatchId", conn, tx))
        {
            cmd.Parameters.AddWithValue("BatchId", Data.BatchId);

            using (var reader = cmd.ExecuteReader(CommandBehavior.CloseConnection))
            {
                while (reader.Read())
                {
                    var values = new object[3];
                    reader.GetValues(values);
                    yield return string.Join(",", values.Select(v => v.ToString()));
                }
            }
        }
    }
}
```

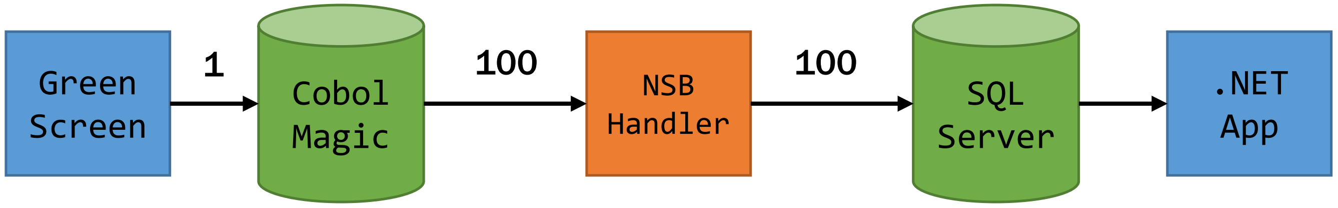
More Mainframe Madness

Stored Procedure Explosions

App Modernization Transition

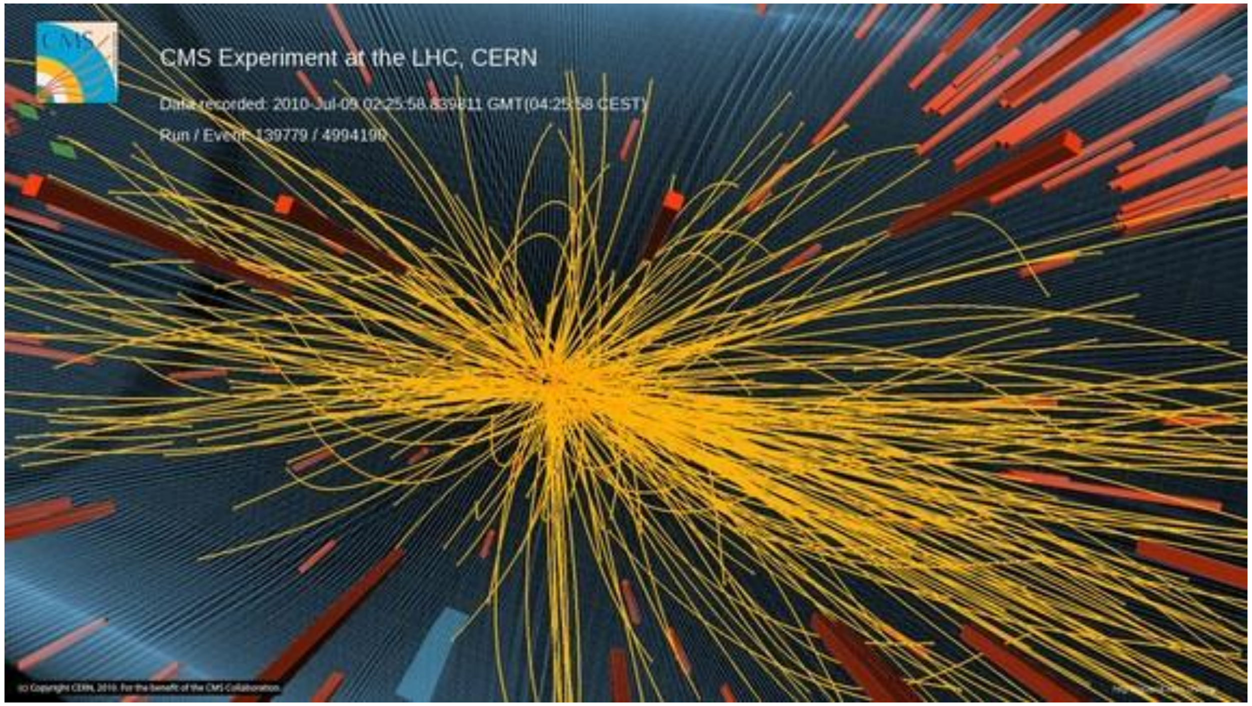


Trigger Happy

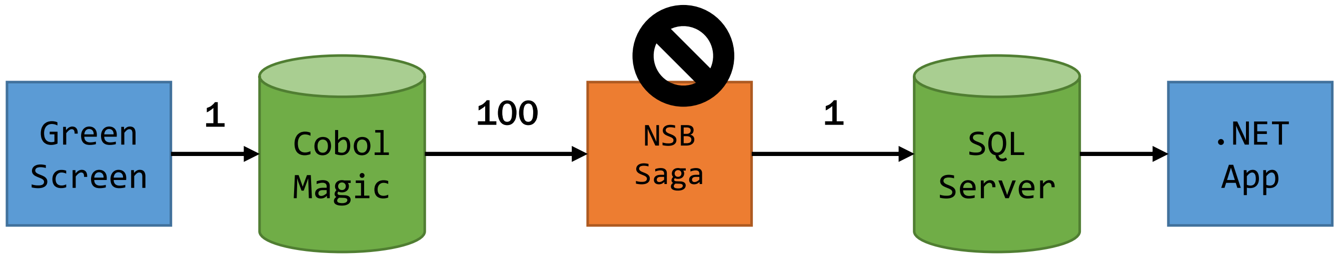


⚠ Sloooooow

Particle Explosions


CMS Experiment at the LHC, CERN
Data recorded: 2010-Jul-09 02:25:58.839811 GMT(04:25:58 CEST)
Run / Event: 139779 / 4994190
(c) Copyright CERN, 2010. For the benefit of the CMS Collaboration.

Slow Your Roll



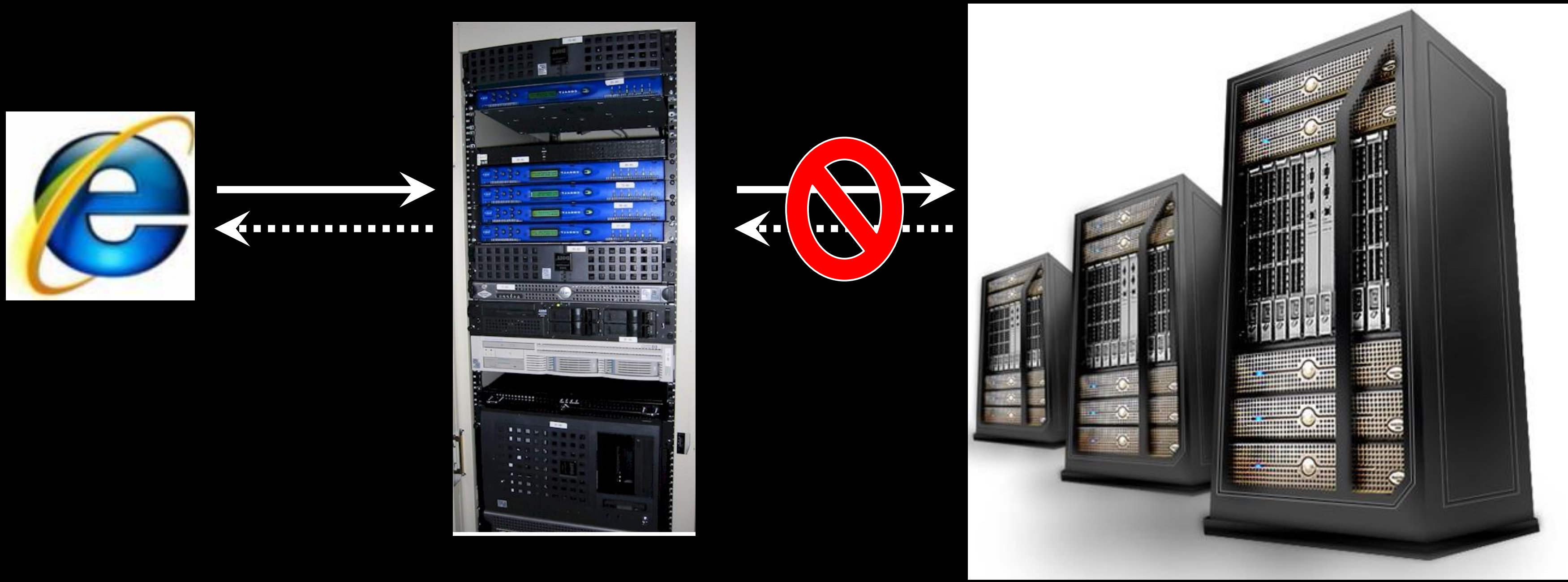
```
public class LegacyProductUpdatedSaga
    : Saga<LegacyProductUpdatedData>,
    IAmStartedByMessages<LegacyProductUpdated>,
    IHandleTimeouts<UpdateProductTimeout>
{
    protected override void ConfigureHowToFindSaga(
        SagaPropertyMapper<LegacyProductUpdatedData> mapper)
    {
        mapper.ConfigureMapping<LegacyProductUpdated>(m => m.ProductId)
            .ToSaga(s => s.ProductId);
    }
```

```
public void Handle(LegacyProductUpdated message)
{
    Data.ProductId = message.ProductId;

    if (Data.RequestedTimeout)
        return;

    Data.RequestedTimeout = true;
    RequestTimeout<UpdateProductTimeout>(TimeSpan.FromSeconds(5));
}

public void Timeout(UpdateProductTimeout state)
{
    Bus.Send(new UpdateNextGenProductData {ProductId = Data.ProductId});
    MarkAsComplete();
}
```

Web Services & Integration

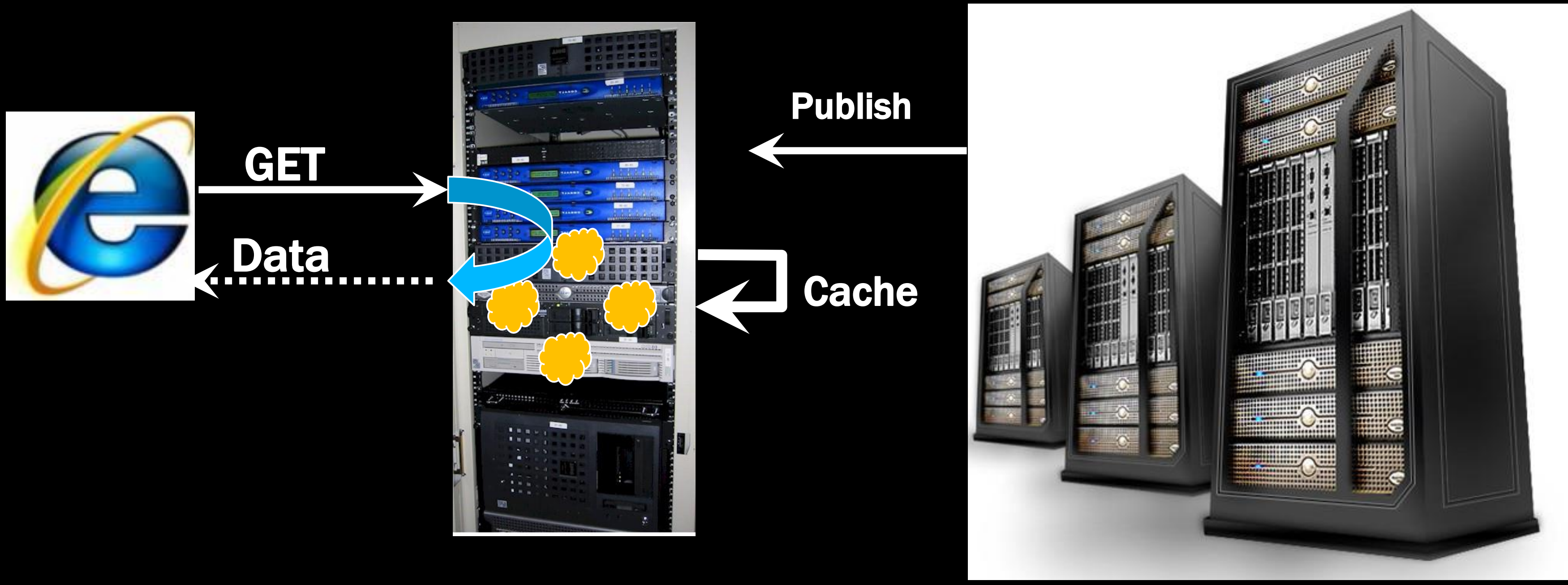
Web Apps

- HTTP request/response with browsers
- Should avoid request/response with backend
  - Easier said than done



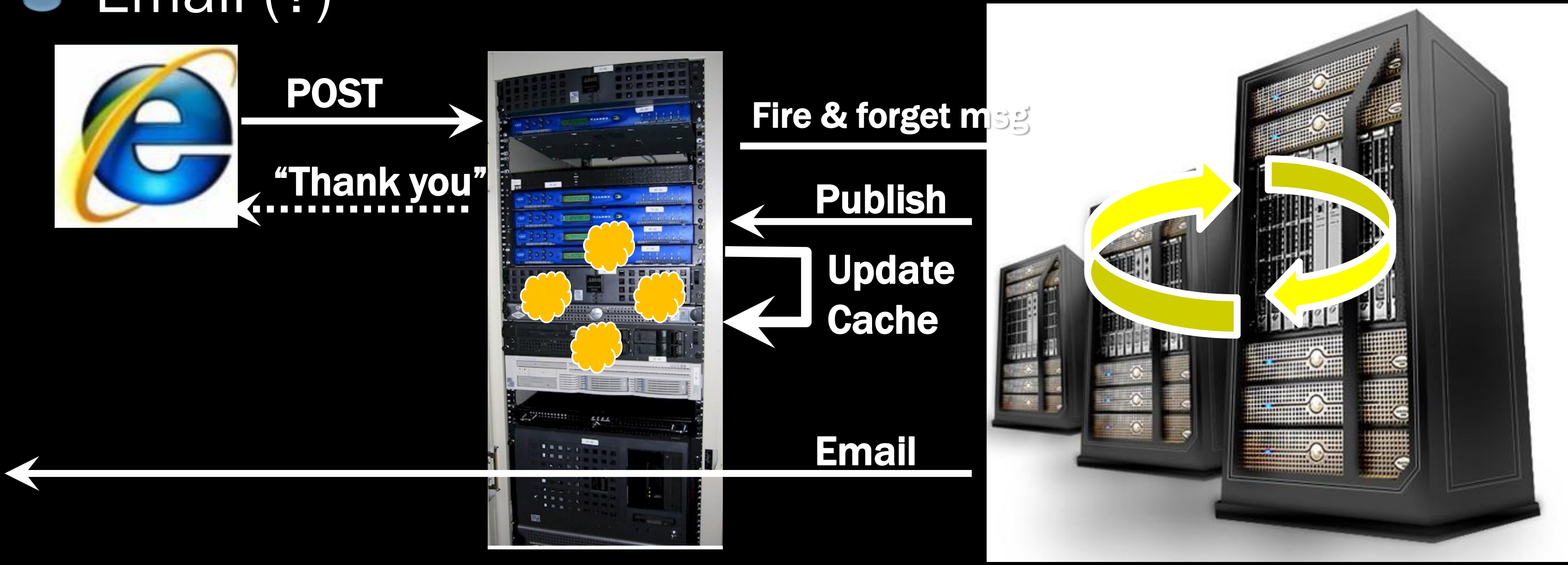
Web Apps, Pub/Sub, and Caching

- Web app front end subscribes to events
  - Caches data from events in the web tier
  - Serves data from the cache to browsers



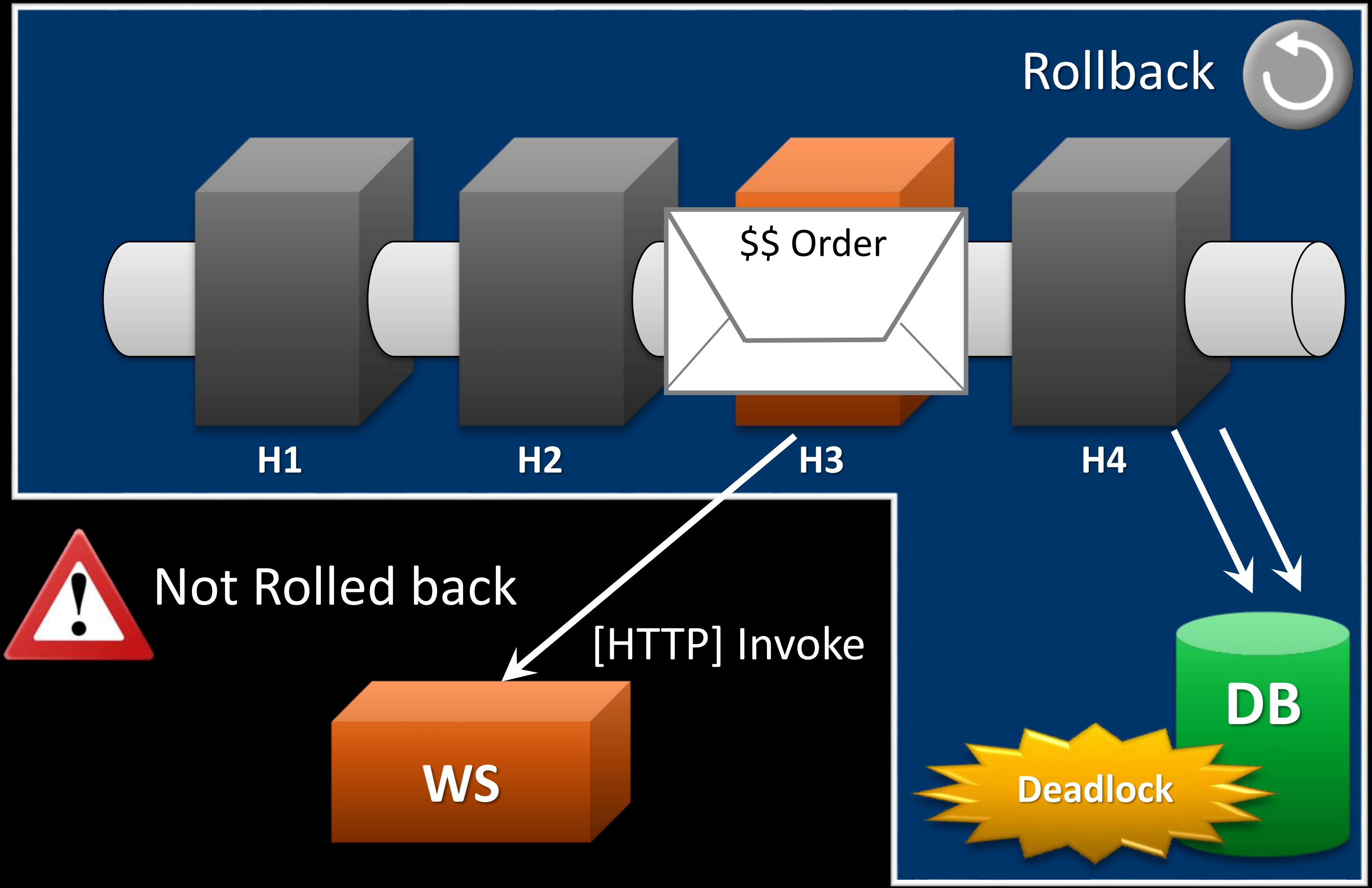
Changing data

- User submits changes to data
  - Send message and show user “thank you” page
  - Successful processing causes publish/cache update
  - Email (?)



Exceptions, Consistency, Integration

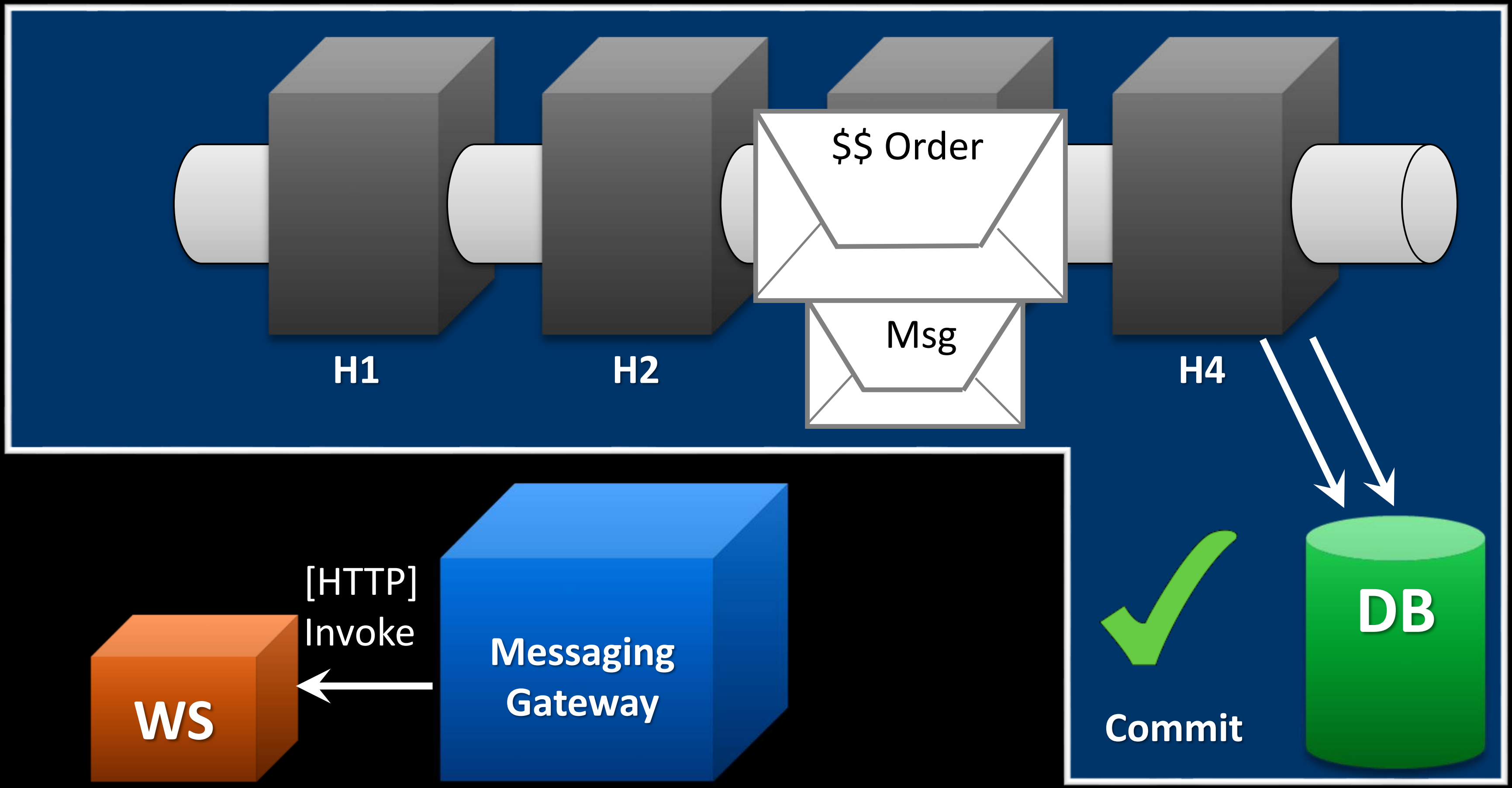
Invoking web services from handlers



Integrating messaging & WS

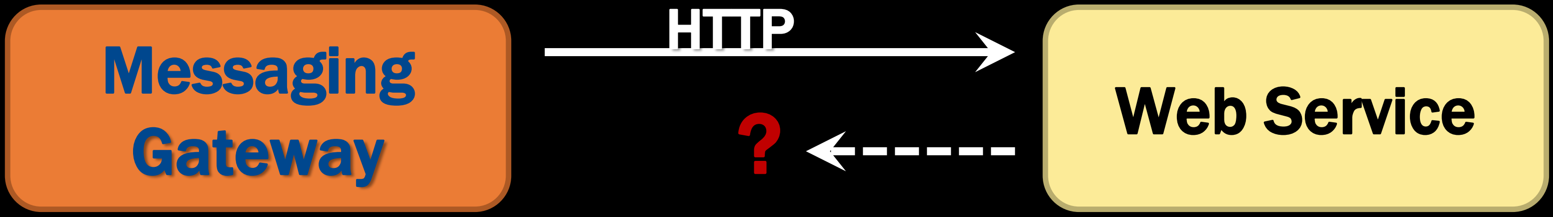
The right way

The message won’t be sent if there’s a failure



Messaging to WS Integration

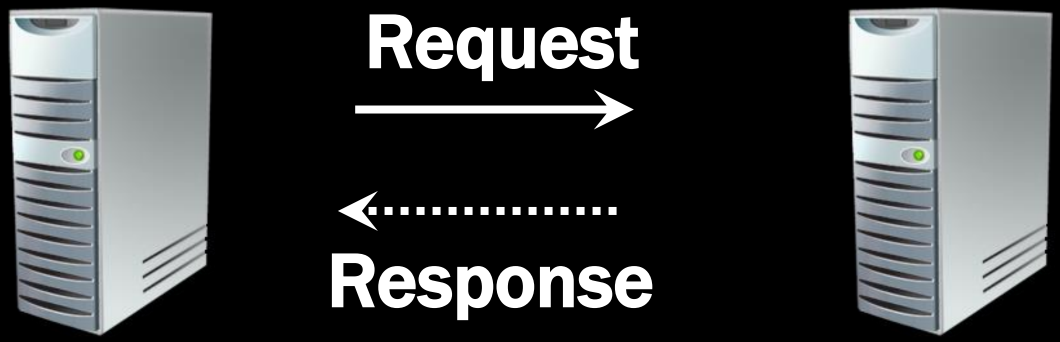
- When calling external web services from a messaging endpoint, if they're down, regular retry logic kicks in.



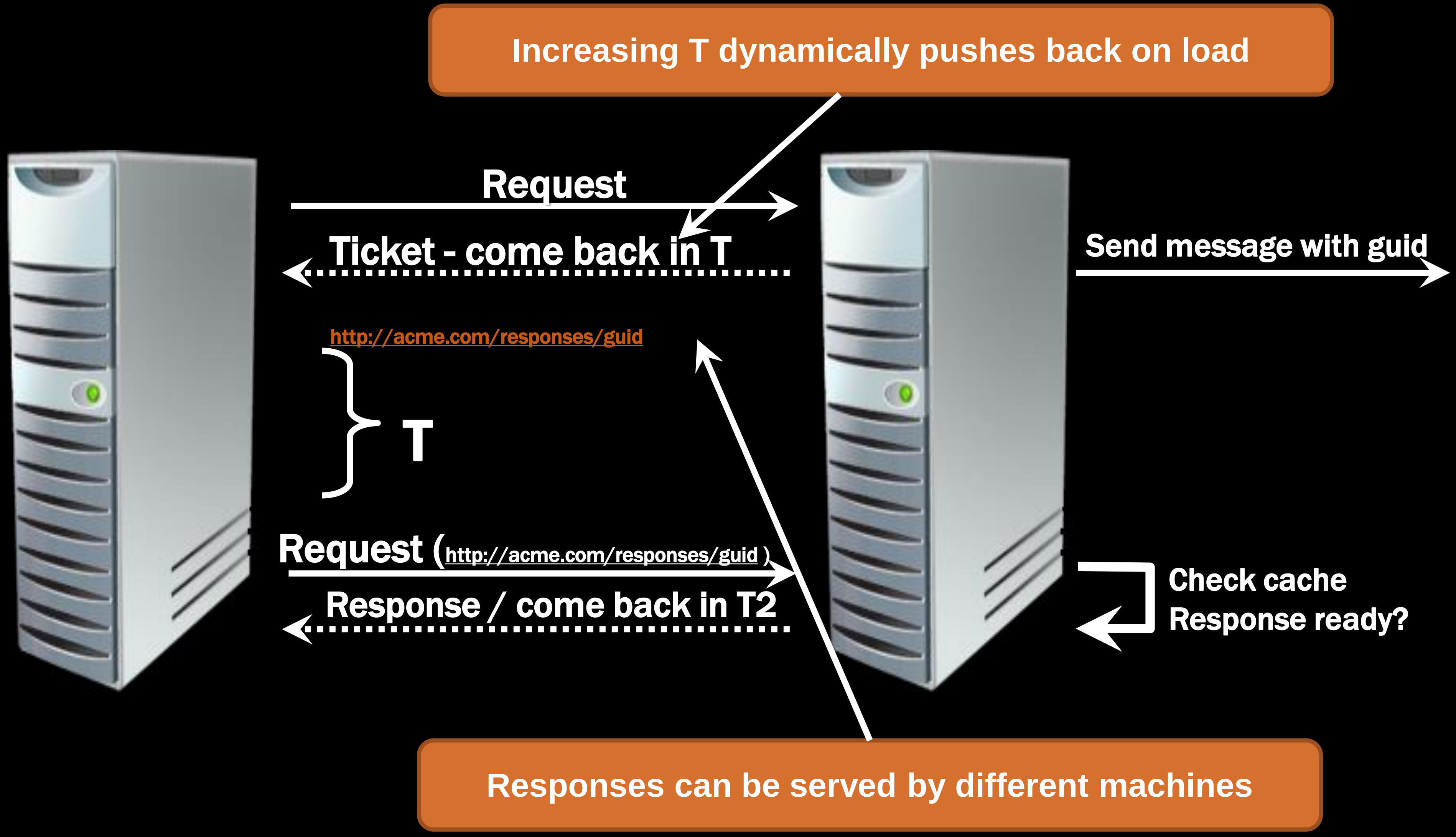
HTTP Timeout Exception & Retry

Web Services & Integration

- Request/response synchronous web service most broadly known
  - Makes sense to expose this kind of endpoint
  - Throttle it – not a scalable solution
  - Direct users to a different integration *protocol*



Web Integration Protocols


Increasing T dynamically pushes back on load
Request
Ticket - come back in T
Send message with guid
http://acme.com/responses/guid
T
Request (http://acme.com/responses/guid)
Response / come back in T2
Check cache
Response ready?
Responses can be served by different machines

Exceptions & Request / Response

- As opposed to RPC, an exception doesn’t return to the endpoint which sent the request
- If there’s a problem with the system, users don’t need to call the helpdesk – administrators can see that there’s a problem from the error queue.
- Consider responding to the user via email

Questions?

Thank you

Jimmy Bogard

Chief Architect, Headspring

jimmybogard.lostechies.com

jimmy@headspring.com

@jbogard

Particular
Software

Open

Q & A

Particular
Software