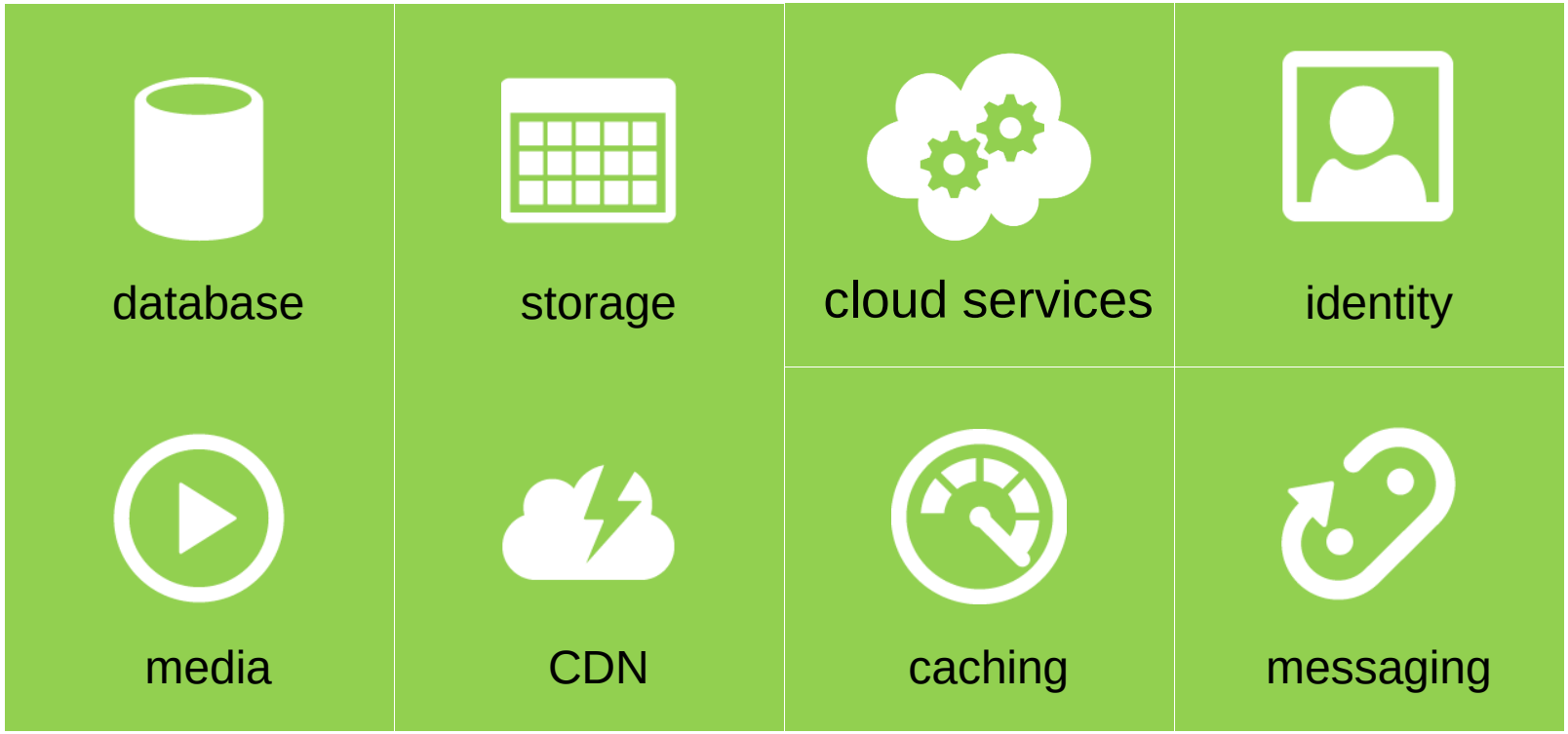
Azure Application Building Blocks



Commonly used components inside the building blocks

1. Cloud Services – Azure WebAPI, Azure WebJob
2. Caching – Redis Cache
3. Messaging – Azure Service Bus On Premise & In the Cloud
4. Storage – Blob storage
5. Database – Azure SQL Server

Azure Application Building Blocks - Database

Use Case 1 – Azure Cloud Table

What topics were covered by Microsoft in the POC:

- Use the existing AIG Resource Group to create an Azure Database.
- Create a simple table inside Azure. [Not Covered]
- Connect to an on premise database. [Not Covered]
- Set Firewall rules for outbound / inbound traffic.
- Copy the connection string from the Azure Portal.
- Use SQL Management Studio to connect to Azure database.

Refer to this help document for more information:

- https://docs.microsoft.com/en-us/azure/sql-database/sql-database-get-started
- https://docs.microsoft.com/en-us/azure/app-service-web/web-sites-hybrid-connection-connect-on-premises-sql-server

Azure Application Building Blocks - Storage

Use Case 2 – Azure Blob Storage

What topics were covered by Microsoft in the POC:

- Create an Azure Blob Container
- Upload a file as a blob to the container
- Download the uploaded file as blob from the container
- List Files in a container using code. [Not Covered]
- Maintain meta data for the blobs using an oracle database [Not Covered]
- Meta data would include file name, path, Order ID etc. [Not Covered]

Refer to this help document for more information:

- https://docs.microsoft.com/en-us/azure/storage/storage-dotnet-how-to-use-blobs

Azure Application Building Blocks - Database

Use Case 3 – Azure WebApi

What topics were covered by Microsoft in the POC:

- Create a Web API project and deploy it into the Azure Portal
- Create a Loan Calculator Controller that does a simple calculation
- Deploy Web API to cloud.
- Create a controller that retrieves data from Oracle. [Not Covered]

Refer to sample project provided by Microsoft:

- LoanCalculatorAPI

Azure Application Building Blocks - Messaging

Use Case 4 – Azure Service Bus

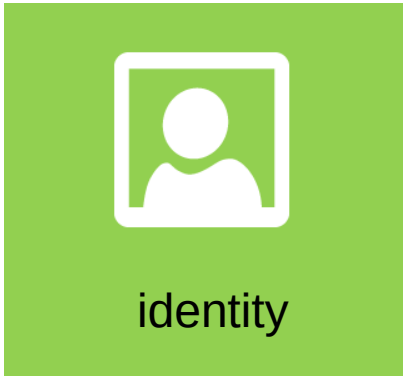
What topics were covered by Microsoft in the POC:

- Create a Service Bus using the Azure Portal
- Create a queue on the Bus
- Send a message to the bus queue
- Received a message from the bus
- Register a service bus in Visual Studio 2015 [Not Covered]
- Build a message relay agent [Not Covered]

Refer to sample project provided by Microsoft:

- LoanCalculatorWebJob

Azure Application Building Blocks - Messaging



Use Case 5 – Security

What topics were covered by Microsoft in the POC:

- Configure account level security / roles in Azure Portal
- Setup security certificates for applications
- Deploy applications

Azure Application Building Blocks - Caching

Use Case 6 – Caching, Session State.

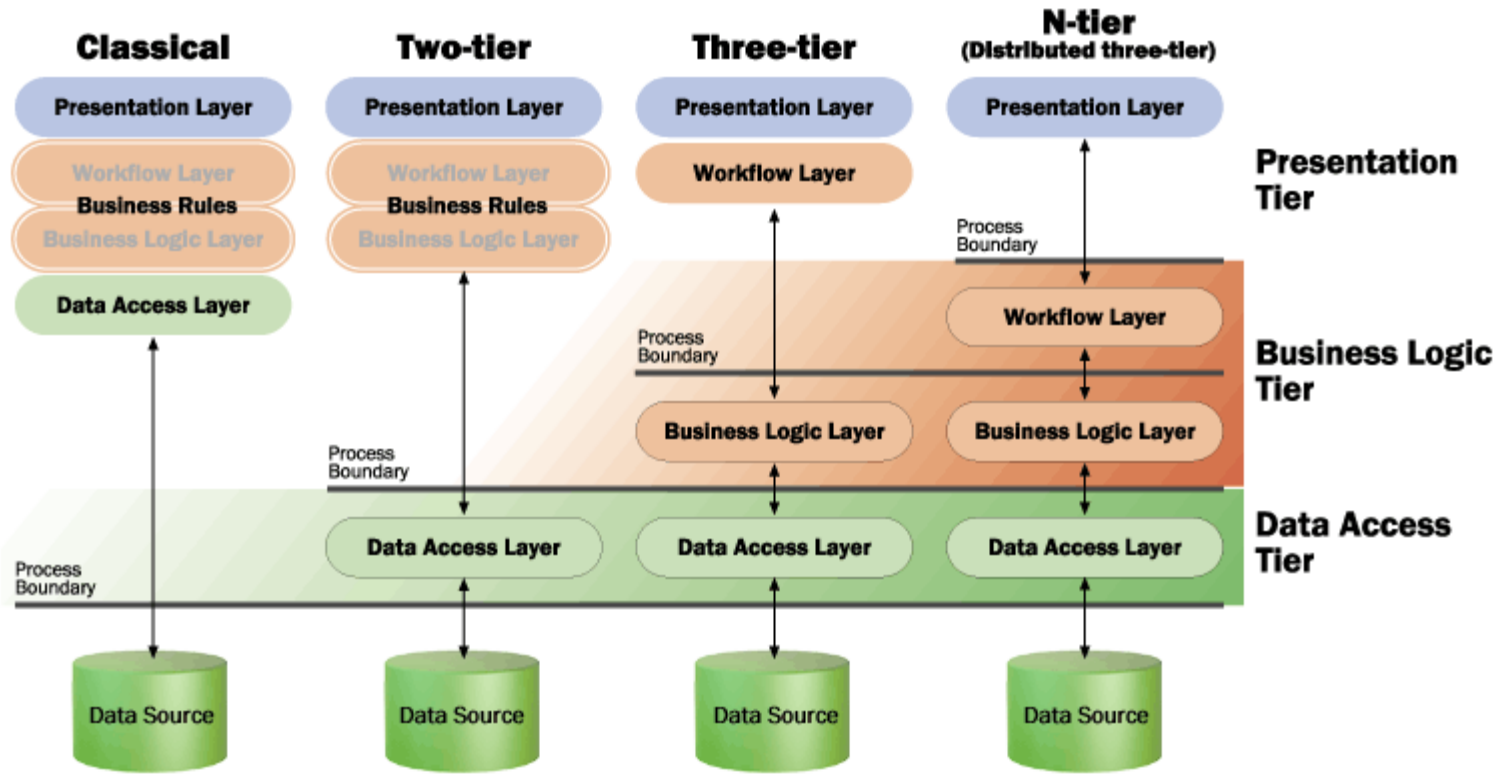
What topics were covered by Microsoft in the POC:

- Create a REDIS Cache
- Use REDIS Dictionary Object to store session state

Refer to REDIS Tutorial to see how it works:

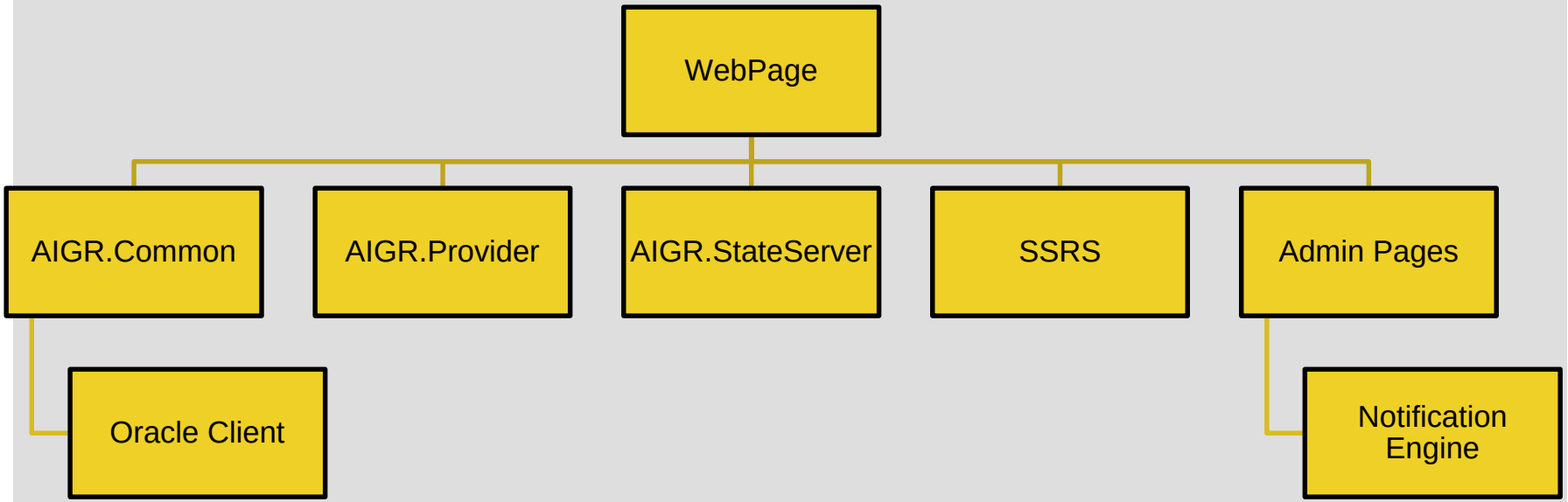
- http://try.redis.io/
- https://redislabs.com/ebook/redis-in-action/appendix-a/a3-installing-on-windows/a3-2-installing-redis-on-window
- https://servicestack.net/redis

N-Tier Architecture

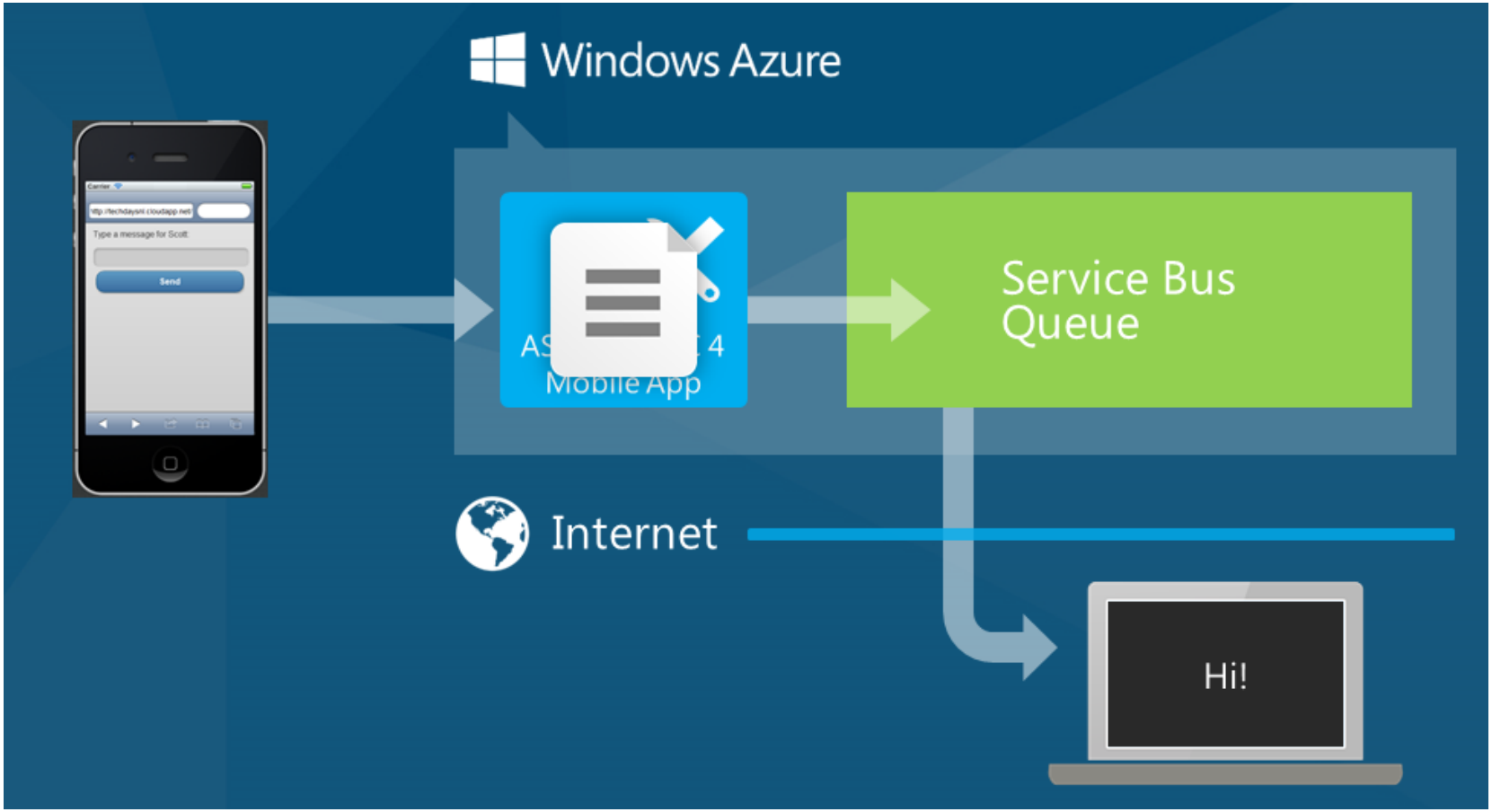


AGILENet - Find the domain

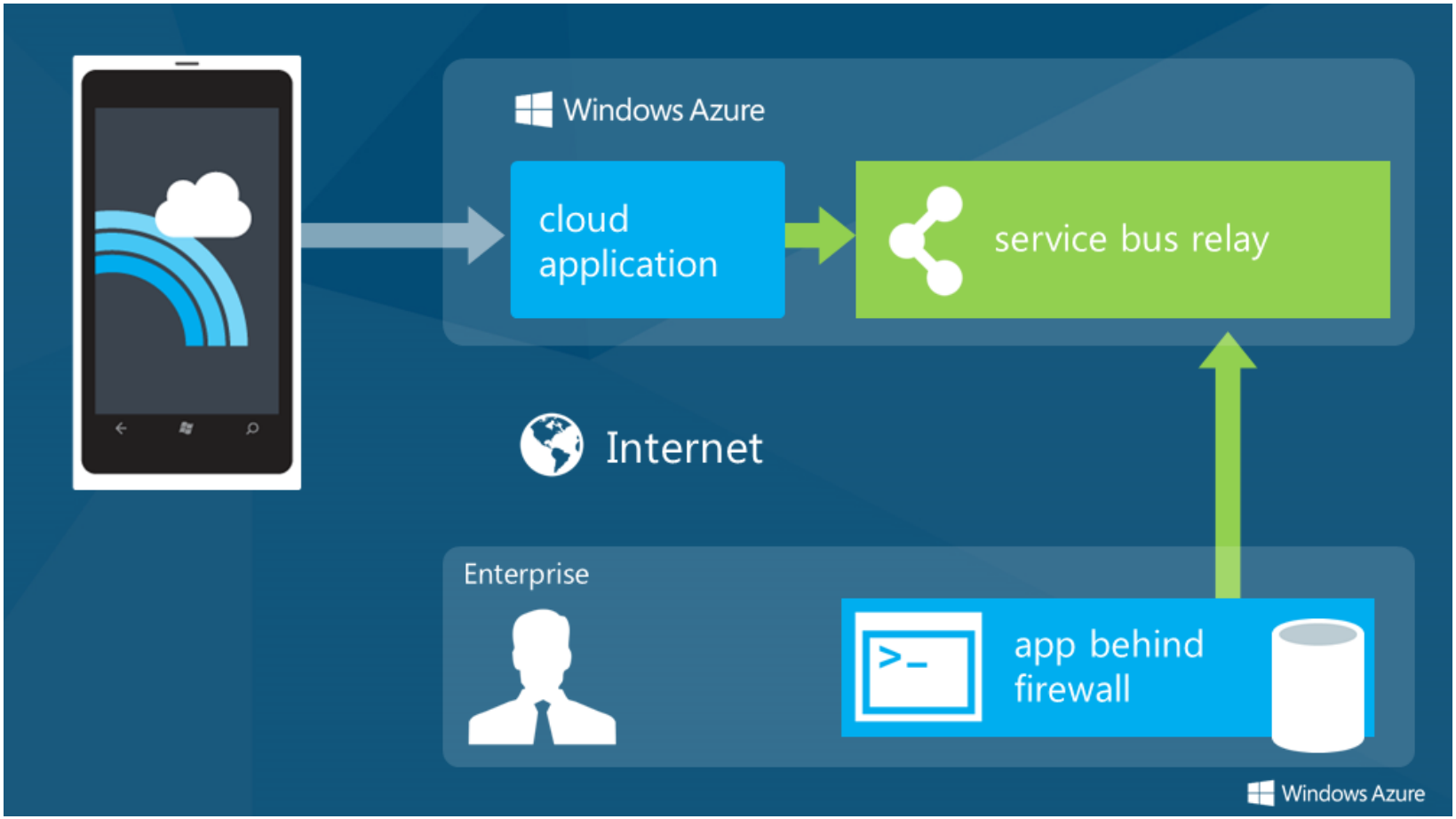
- Two Tier Architecture
- WebPage uses SQLHelper to call a stored procedure on Oracle DB.
- DataSet is bound to user controls on web page.
- Identify the domain for each object.



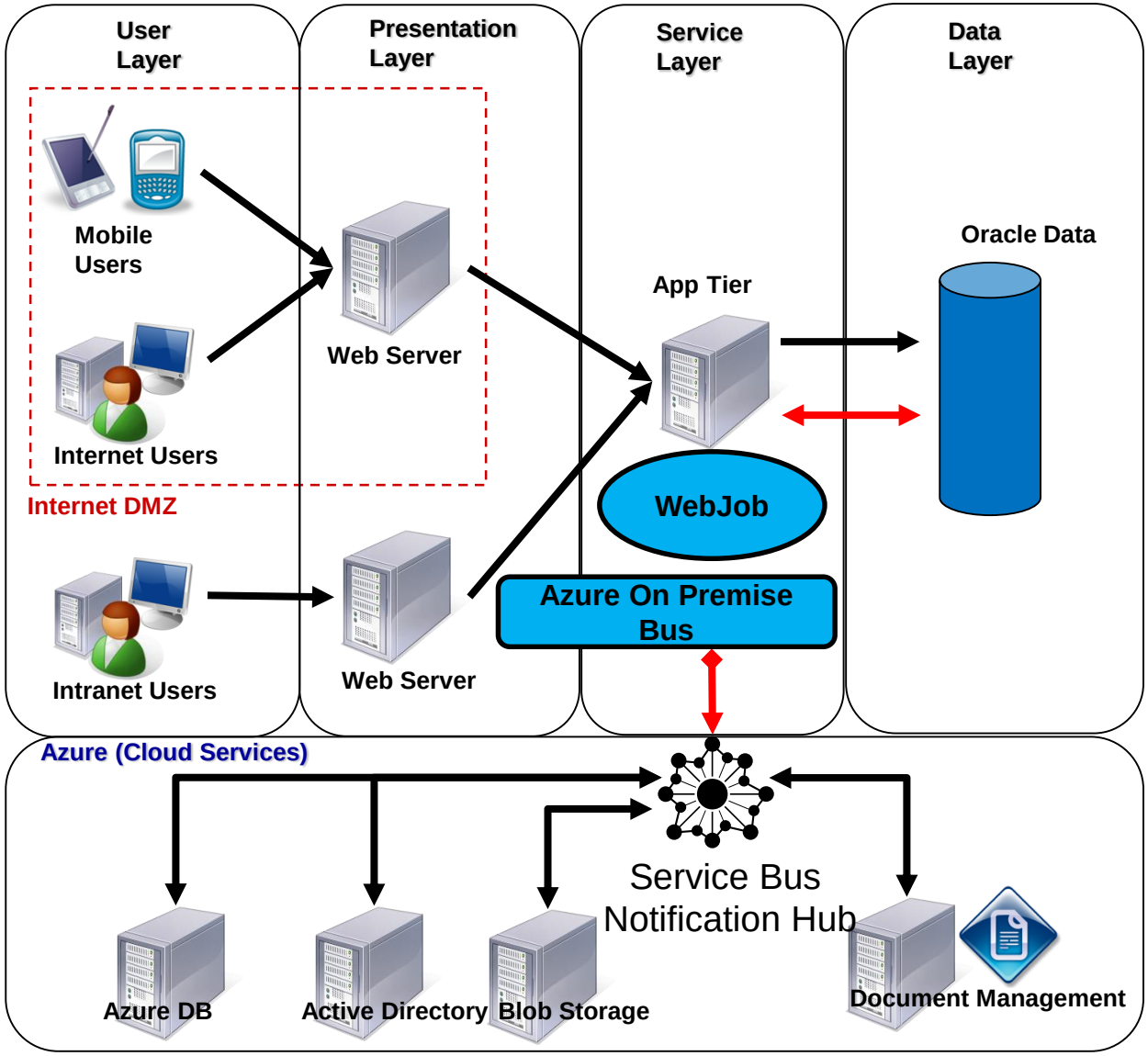
Data synchronization via Bus Scenario inbound



Data synchronization scenario outbound



Infrastructure



Chain of Responsibility Design Pattern

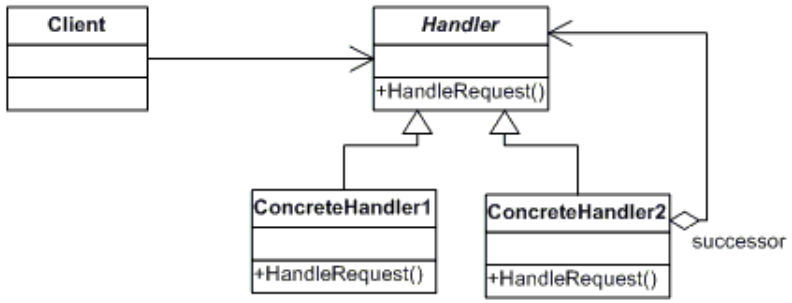
- definition
- UML diagram

---

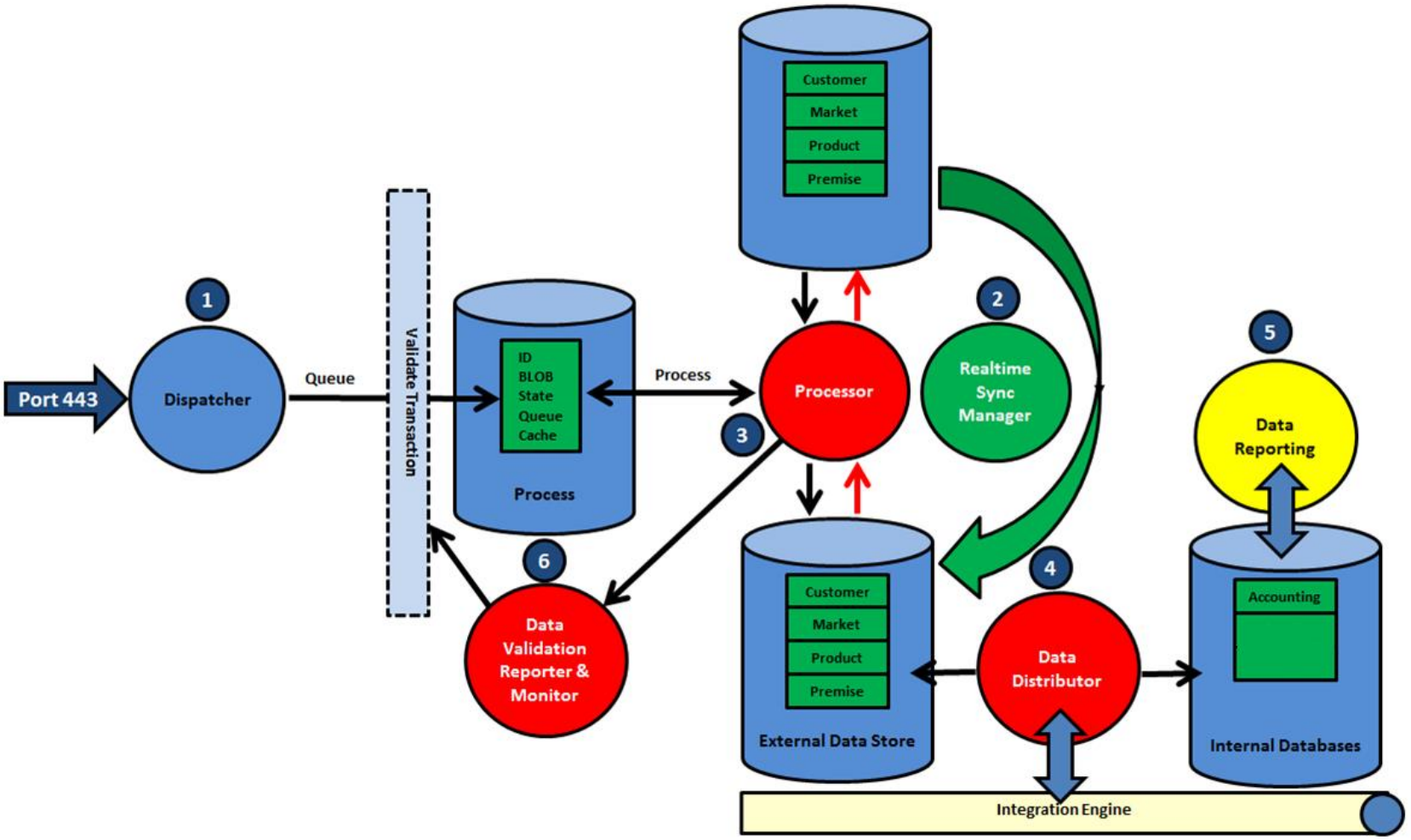
definition

The **Chain** of **Responsibility Pattern** describes how we handle a single request by a **chain** of multiple handler objects. The request has to be processed by only one handler object from this **chain**. However, the determination of processing the request is decided by the current handler. If the current handler object is able to process the request, then the request will be processed in the current handler object; otherwise, the current handler object needs to shirk **responsibility** and push the request to the next **chain** handler object. And so on and so forth until the request is processed.

UML class diagram



Integration Workflow



Questions?