Welcome to the

2017 Global Azure BOOTCAMP

Israel

Azure Service Fabric

Eran Stiller & Tamir Dresher

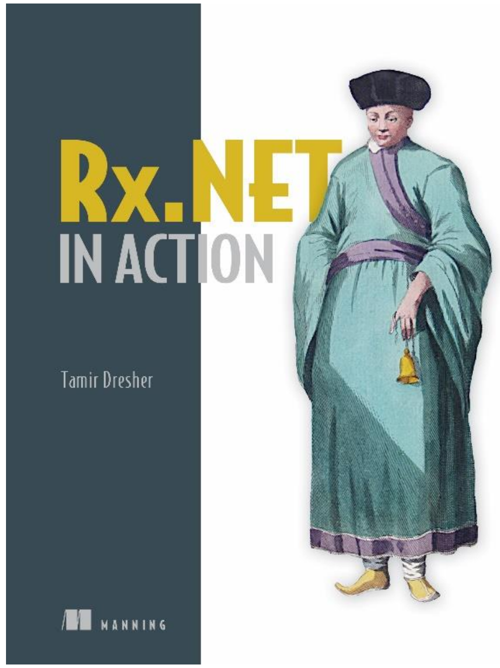
CodeValue

About Tamir

@tamir_dresher
tamirdr@codevalue.net
http://www.TamirDresher.com.

OzCode Your Road to Magical Debugging

- Author of *Rx.NET in Action* (manning publications)
- Software architect, consultant and instructor
- Software Engineering Lecturer @ Ruppin Academic Center
- OzCode (www.oz-code.com) Evangelist



About Eran

@eranstiller
erans@codevalue.net
http://stiller.blog

- Cloud Division Leader & Co-Founder at CodeValue
- Software architect, consultant and instructor
- Microsoft Azure MVP
- Expert in large-scale, server-side, highly-concurrent systems
- Active member of Microsoft Azure Advisors group
- Co-Founder of Azure Israel Meetup

Sponsors & Prices

2017
Global Azure
BOOTCAMP

A BIG thank you to the 2017 Global Sponsors!

myget

opsgility

SERVICEBUS 360

JET BRAINS

Cloudmonix

Microsoft

SentryOne™

For providing the "Stuff We All Get"!

2017 Global Azure BOOTCAMP

"Stuff We All Get"

Sponsor	Offering
Cloudmonix https://cloudmonix.com/	Cloudmonix offers 1 full month of unlimited monitoring under the Ultimate plan.
SentryOne http://www.sentryone.com/	SQL Sentry offers an extended evaluation for every attendee.
ServiceBus360 http://www.servicebus360.com/	Servicebus360 offers an extended license until September 30 for all attendees, and this for 3 namespaces, 3 alarms per namespace, 3 resources per namespace to monitor and 2 users.
OpsGility http://www.Opsgility.com/	Opsgility offers a 30 day subscription free trial for all attendees.

2017 Global Azure BOOTCAMP

Raffle Prizes

1 Winner: One license chosen from ReSharper, dotTrace Memory, dotTrace Performance, dotCover, dotPeek, PhpStorm, PyCharm, IntelliJ IDEA, AppCode, WebStorm, RubyMine. You'll receive a voucher with a code from the organizers.

1 Winner: a 1 year license of DBSentry (formerly Performance Advisor for Azure SQL Database) for raffle per location. Give your Full name and email to the Organizers.

MyGet is offering a free Starter subscription for 1 year for 1 attendee per location. Give your Full name and email to the Organizers.

They offer 3 12-month subscriptions per GAB Location.
Give your Full name, email phone and company to the Organizers.

2017 Global Azure BOOTCAMP

Time to get going!
Have a GREAT Azure day!

2017
Global Azure
BOOTCAMP

Topics

Introducing Microservices & Azure Service Fabric

Developing Service Fabric Applications

Service Fabric Concepts

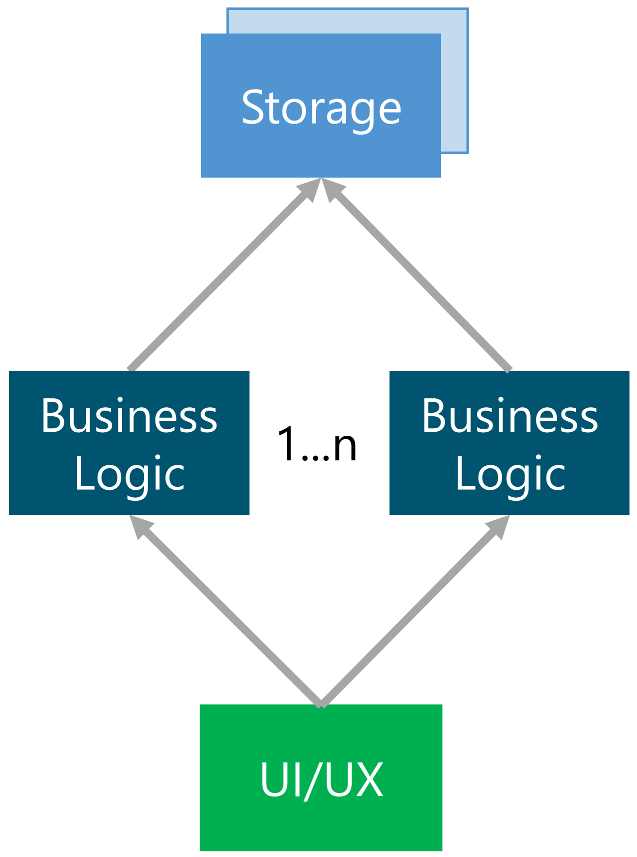
Deployments & Upgrades

Introducing Microservices & Azure Service Fabric

The Monolith

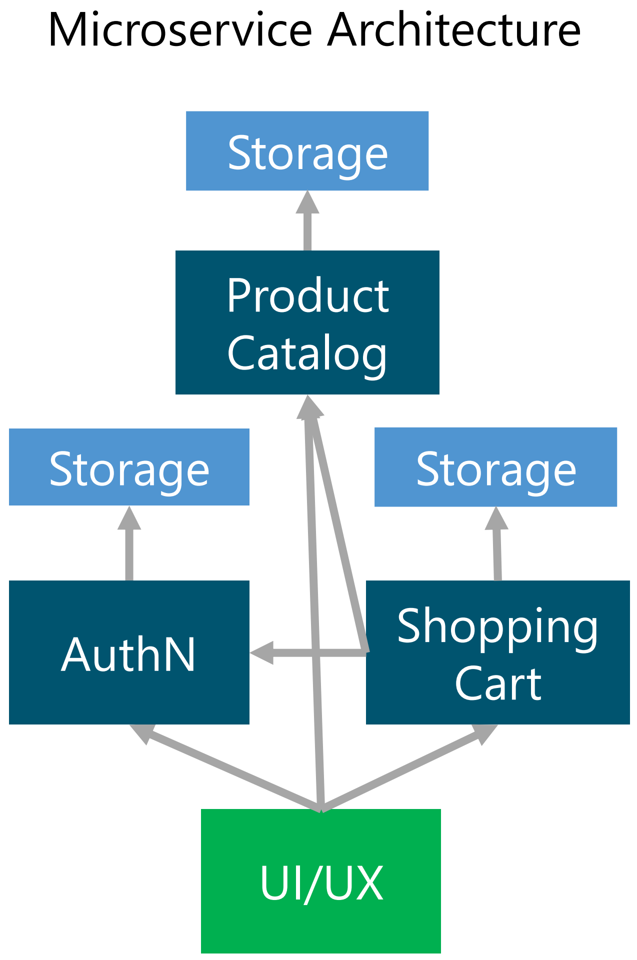
- Large self-contained application
- Deployed and scaled as a group
- Storage if often a single store
- Address the entire set of application needs

Monolithic Architecture



Microservices Architecture

- An evolution of Service Oriented Architecture
  - Create a system as a suite of small, independently deployable services
  - Services communicate across lightweight protocols
- Considerations
  - Services can be scaled to meet their individual needs
  - Supports use of different technology stacks for different system parts
  - Process automation is important (DevOps)



Microservices dilemmas

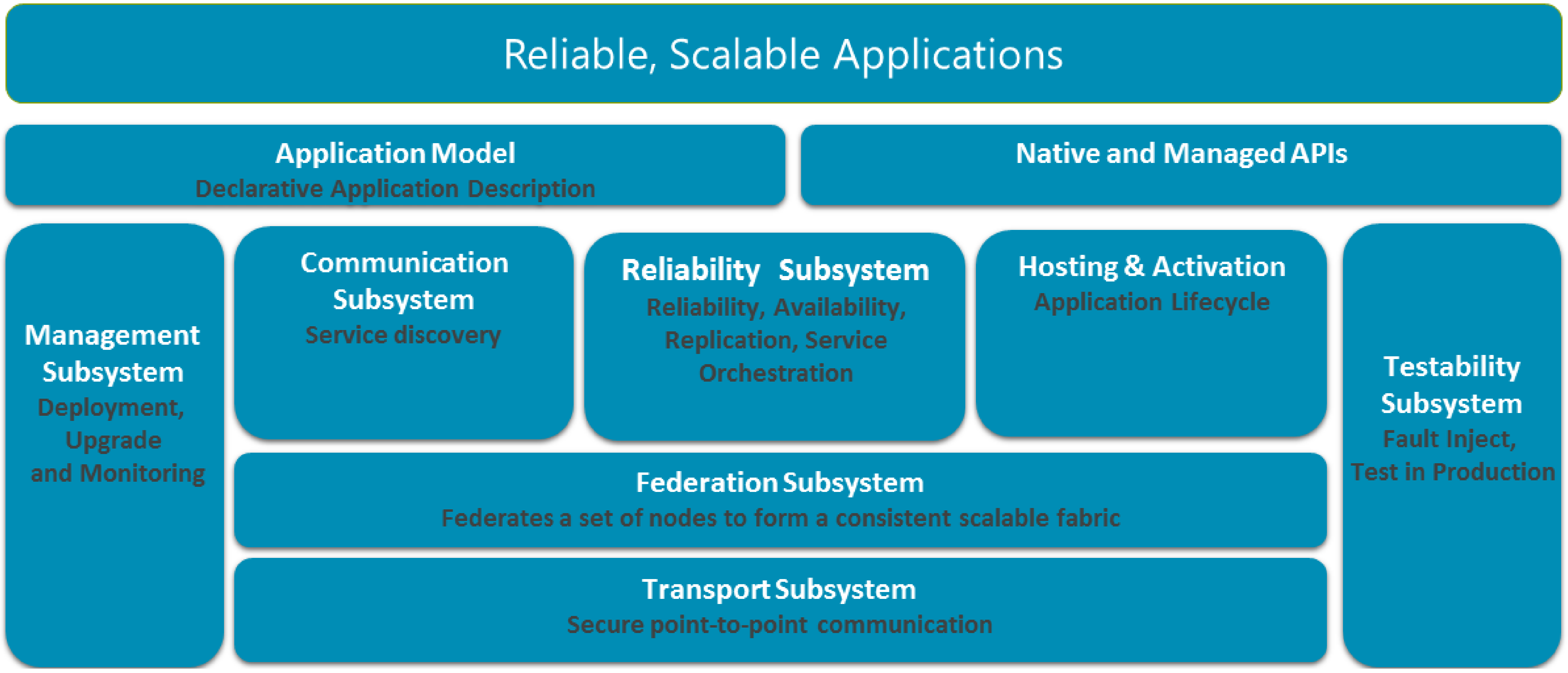
- Discovery
- Deployment
- Scaling
- Health monitoring
- Resource utilization
- State management

Orchestration

Azure Service Fabric

- Can be thought of as an orchestrator of services across a managed cluster of machines
- Good for microservices, but not only
- Can be run in Azure, on-premises, or in other clouds
- Proven platform used by several Microsoft Cloud services, including:
  - Azure SQL Database
  - Document DB
  - Cortana
  - Microsoft Power BI
  - Microsoft Intune
  - Azure Event Hubs
  - Azure IoT Hub
  - Skype for Business

Azure Service Fabric - Architecture


Reliable, Scalable Applications
Application Model
Declarative Application Description
Native and Managed APIs
Management Subsystem
Deployment, Upgrade and Monitoring
Communication Subsystem
Service discovery
Reliability Subsystem
Reliability, Availability, Replication, Service Orchestration
Hosting & Activation
Application Lifecycle
Testability Subsystem
Fault Inject, Test in Production
Federation Subsystem
Federates a set of nodes to form a consistent scalable fabric
Transport Subsystem
Secure point-to-point communication

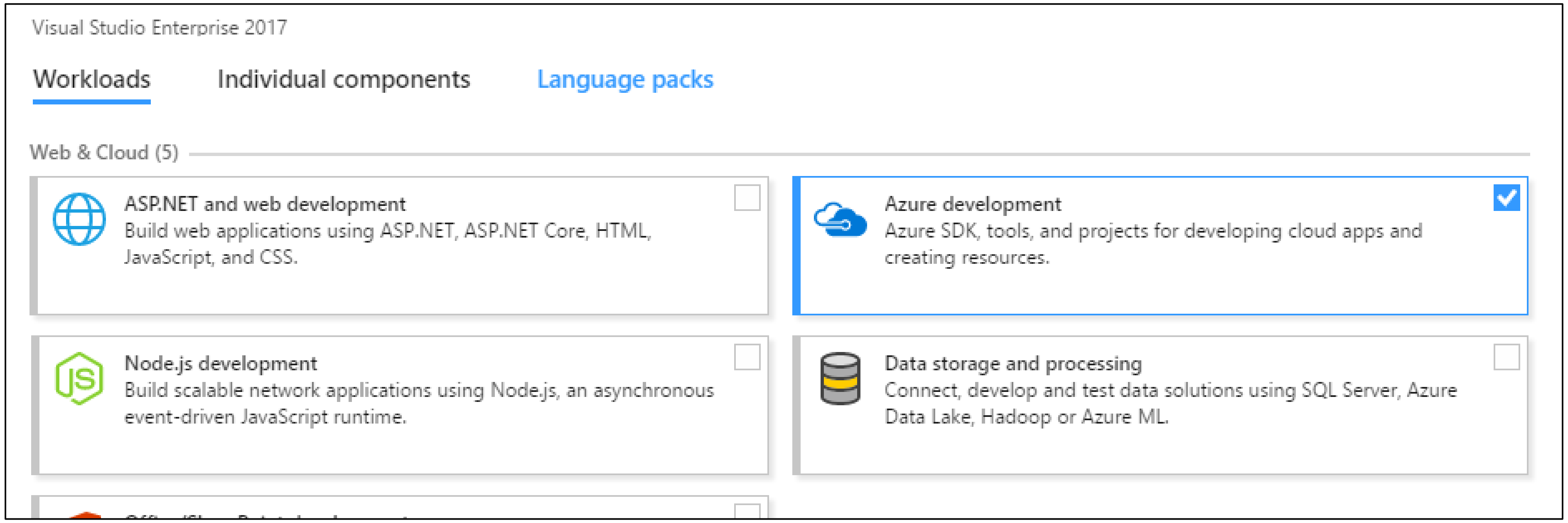
https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-architecture

Developing Service Fabric Applications

Windows Setup

Installation

The Microsoft Azure Service Fabric SDK is included in the “Azure development” Workload in the Visual Studio 2017 installer



Visual Studio Enterprise 2017

Workloads | Individual components | Language packs

Web & Cloud (5)

ASP.NET and web development [ ]
Build web applications using ASP.NET, ASP.NET Core, HTML, JavaScript, and CSS.

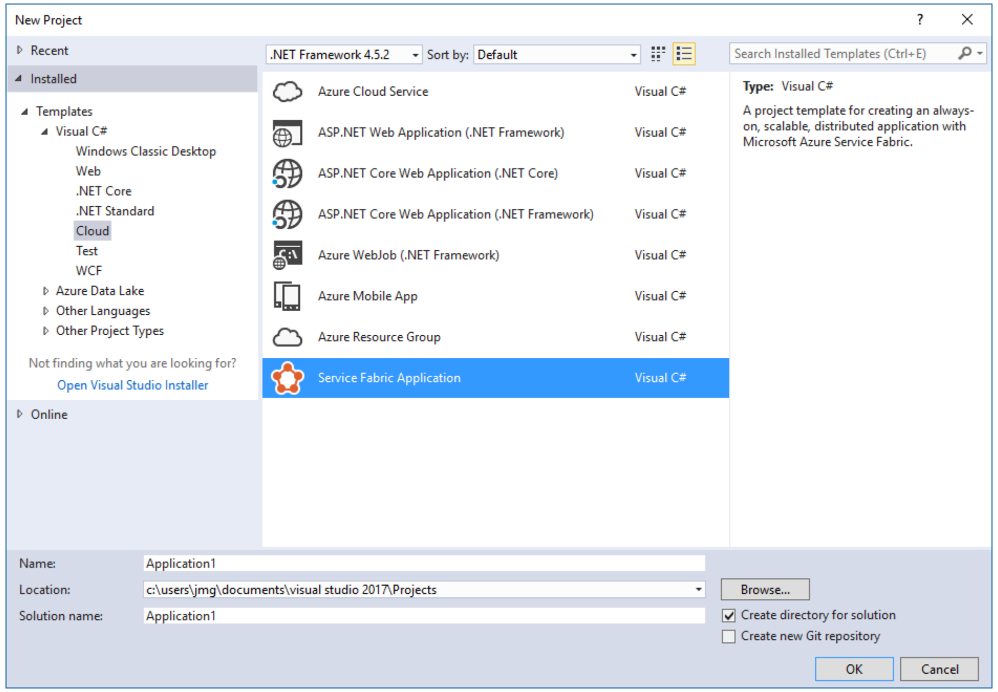
Azure development [✓]
Azure SDK, tools, and projects for developing cloud apps and creating resources.

Node.js development [ ]
Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Data storage and processing [ ]
Connect, develop and test data solutions using SQL Server, Azure Data Lake, Hadoop or Azure ML.

Creating a Project

- Create a new project by selecting the “Service Fabric Application” template from either the Cloud project node.
- Choose your desired service type template



New Project

- Recent
- Installed
  - Templates
    - Visual C#
      - Windows Classic Desktop
      - Web
      - .NET Core
      - .NET Standard
      - Cloud
      - Test
      - WCF
    - Azure Data Lake
    - Other Languages
    - Other Project Types

Not finding what you are looking for?
Open Visual Studio Installer

- Online

.NET Framework 4.5.2 | Sort by: Default | Search Installed Templates (Ctrl+E)

Azure Cloud Service	Visual C#
ASP.NET Web Application (.NET Framework)	Visual C#
ASP.NET Core Web Application (.NET Core)	Visual C#
ASP.NET Core Web Application (.NET Framework)	Visual C#
Azure WebJob (.NET Framework)	Visual C#
Azure Mobile App	Visual C#
Azure Resource Group	Visual C#
Service Fabric Application	Visual C#

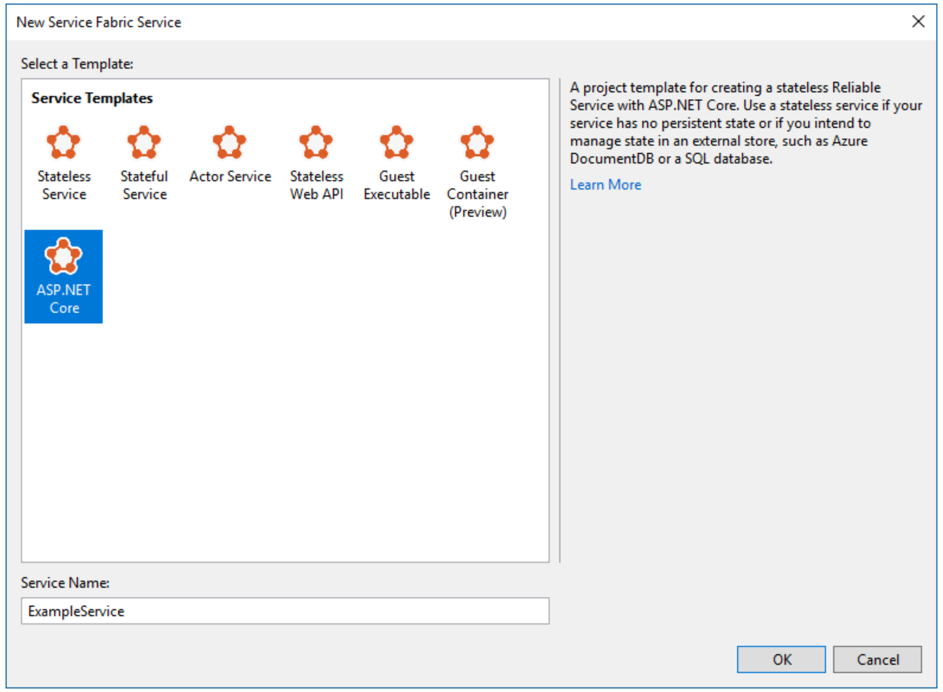
Type: Visual C#

A project template for creating an always-on, scalable, distributed application with Microsoft Azure Service Fabric.

Name: Application1
Location: c:\users\jmg\documents\visual studio 2017\Projects
Solution name: Application1

Browse...
[✓] Create directory for solution
[ ] Create new Git repository

OK | Cancel



New Service Fabric Service

Select a Template:

Service Templates

Stateless Service | Stateful Service | Actor Service | Stateless Web API | Guest Executable | Guest Container (Preview) | ASP.NET Core

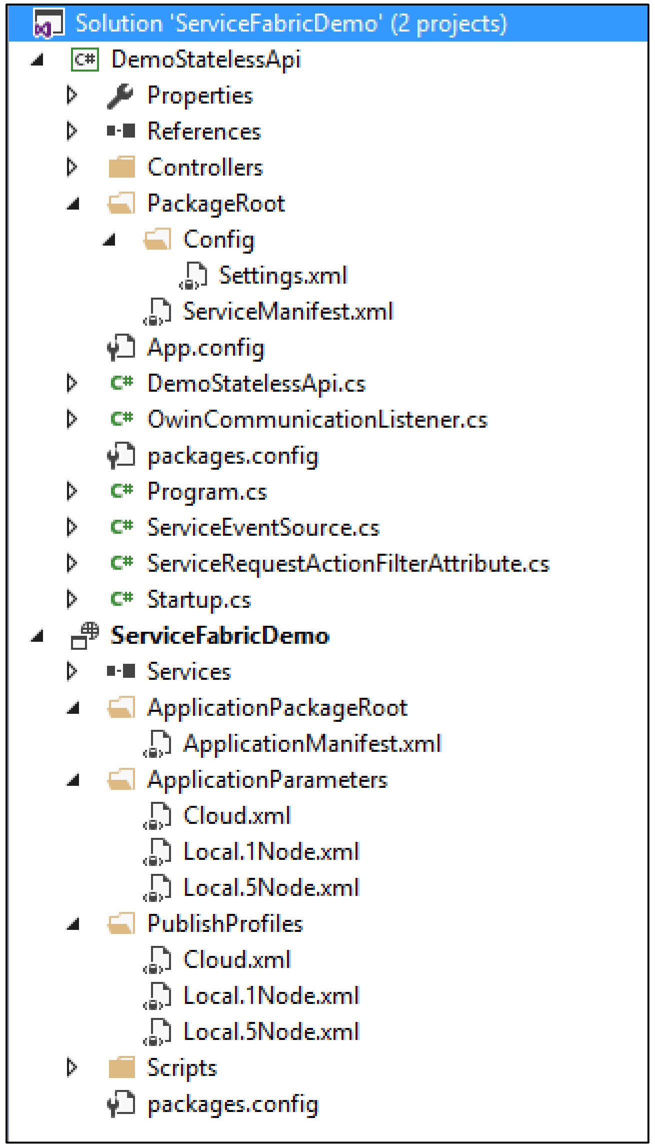
A project template for creating a stateless Reliable Service with ASP.NET Core. Use a stateless service if your service has no persistent state or if you intend to manage state in an external store, such as Azure DocumentDB or a SQL database.
Learn More

Service Name: ExampleService

OK | Cancel

Solution Structure

- Application Project
  - The Services node lists the service projects that make up the Application
  - Includes an Application Manifest that describes the services that make up the Application
  - Includes application publishing profiles, and per-profile settings.
- Service Project(s)
  - Includes a Service Manifest that describes the Service characteristics
  - Includes the Service code and configuration



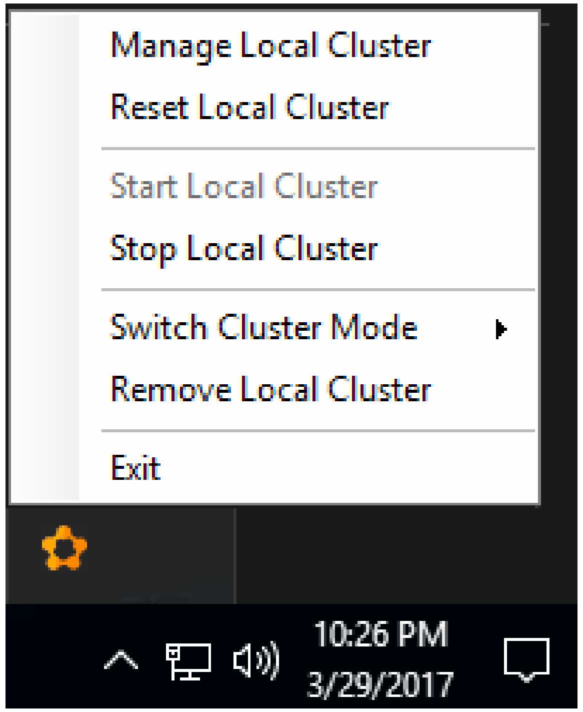
```
Solution 'ServiceFabricDemo' (2 projects)
  DemoStatelessApi
    Properties
    References
    Controllers
    PackageRoot
      Config
        Settings.xml
      ServiceManifest.xml
    App.config
    DemoStatelessApi.cs
    OwinCommunicationListener.cs
    packages.config
    Program.cs
    ServiceEventSource.cs
    ServiceRequestActionFilterAttribute.cs
    Startup.cs
  ServiceFabricDemo
    Services
    ApplicationPackageRoot
      ApplicationManifest.xml
    ApplicationParameters
      Cloud.xml
      Local.1Node.xml
      Local.5Node.xml
    PublishProfiles
      Cloud.xml
      Local.1Node.xml
      Local.5Node.xml
    Scripts
    packages.config
```

Demo

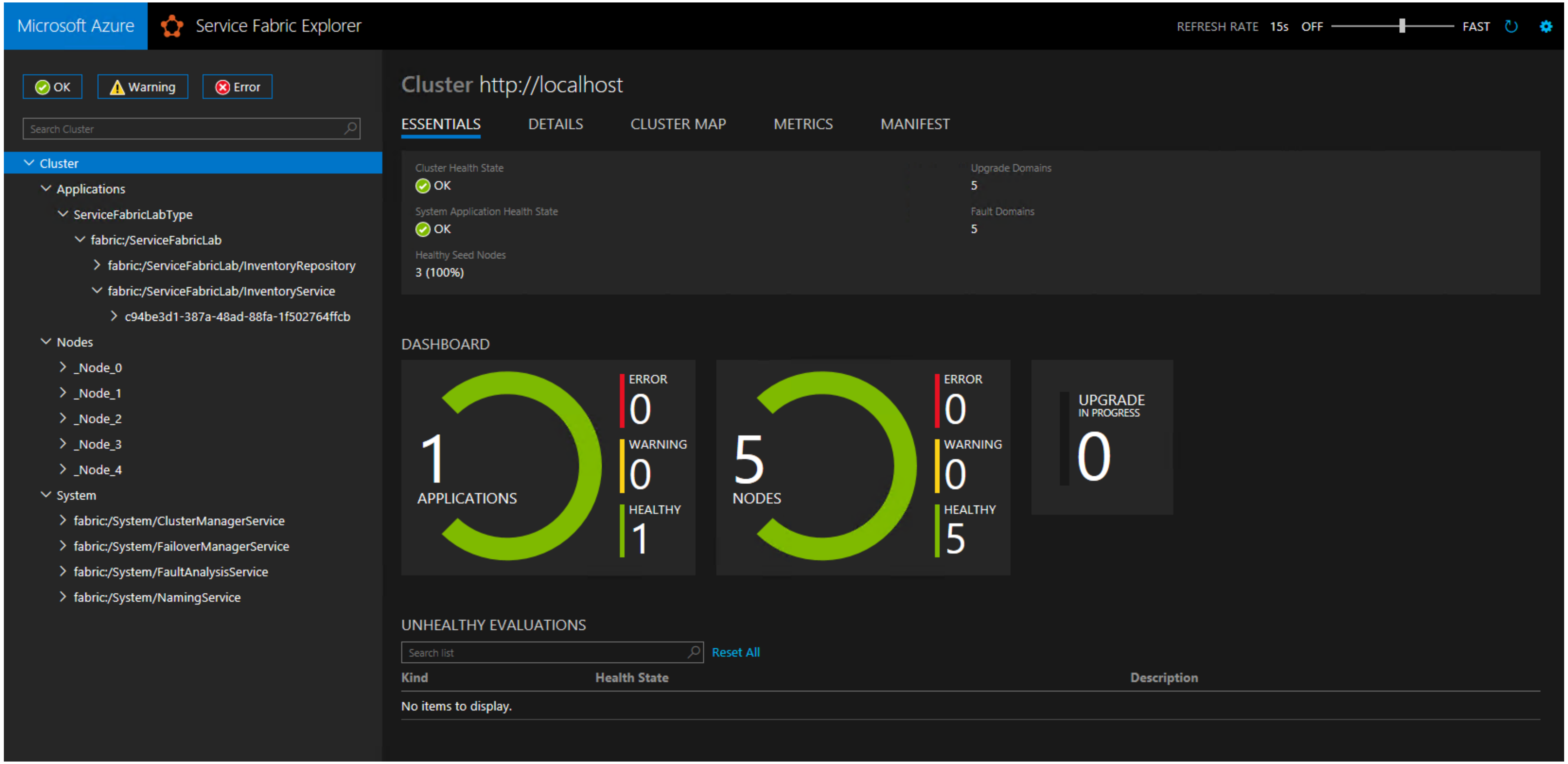
Guest Executables

Service Fabric Local Cluster

- Installed as part of the Service Fabric SDK
- Provides a local debug & test environment
- Can provision a 1-node or 5-node cluster
- Not an emulator or a simulator
- Can be managed via the Windows System Tray application or via PowerShell scripts.
- Visual Studio is configured to work with the Local Cluster



Service Fabric Explorer



Cluster
- Applications
  - ServiceFabricLabType
    - fabric:/ServiceFabricLab
      - fabric:/ServiceFabricLab/InventoryRepository
      - fabric:/ServiceFabricLab/InventoryService
        - c94be3d1-387a-48ad-88fa-1f502764ffcb
- Nodes
  - _Node_0
  - _Node_1
  - _Node_2
  - _Node_3
  - _Node_4
- System
  - fabric:/System/ClusterManagerService
  - fabric:/System/FailoverManagerService
  - fabric:/System/FaultAnalysisService
  - fabric:/System/NamingService

Cluster http://localhost

Cluster Health State
OK

System Application Health State
OK

Healthy Seed Nodes
3 (100%)

Upgrade Domains
5

Fault Domains
5

DASHBOARD

1 APPLICATIONS — ERROR 0, WARNING 0, HEALTHY 1

5 NODES — ERROR 0, WARNING 0, HEALTHY 5

UPGRADE IN PROGRESS 0

UNHEALTHY EVALUATIONS

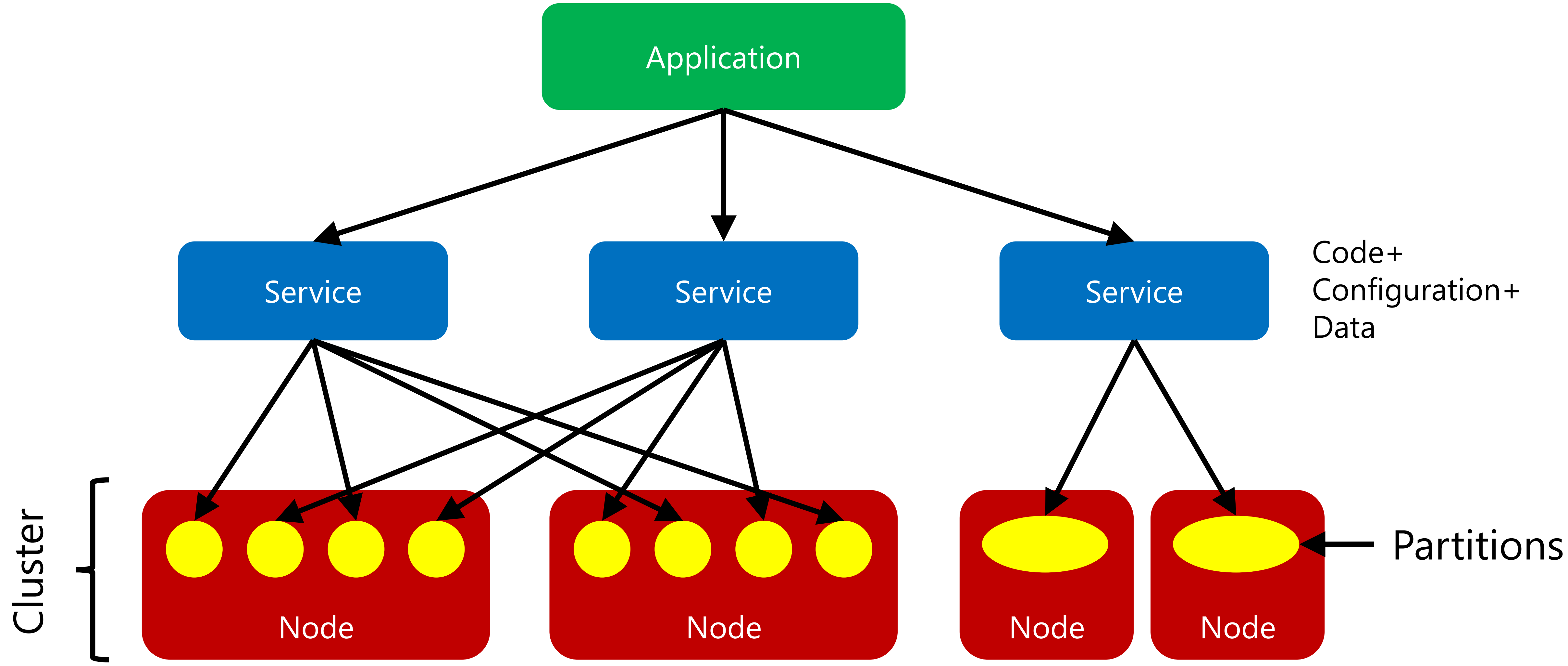
Kind	Health State	Description
No items to display.		

Demo

Monitoring Service Fabric

Service Fabric Concepts

Service Fabric Concepts



Service Models

Reliable Service & Reliable Actor Services

- Leverage Service Fabric's own programming models
- Networking Naming Service support
- Integrate with code, configuration, and data upgrades
- Instances are not process isolated for each instance, but instead are created as objects (higher density on host.)

Guest Executable Services

- Arbitrary executable packaged in a Service Fabric service
- Can be written in any language
- Service Fabric manages orchestration and execution management of the executable
- Limited integration with the Service Fabric APIs

Container Services

- Containers can either be Reliable Services or Guest Containers
- Supports Docker Containers (Linux) and Windows Server Containers
- Currently in Preview

Reliable Services

Stateless Service

- Typically used for Web API front-ends or background workers that look at external queues
- No state is maintained within the service, though external state storage can certainly be used

Stateful Service

- Used when state must be consistent and present for the service to function
- Leverages the Reliable Collections
  - ReliableQueue, ReliableDictionary
  - State is kept locally, but replicated for high-availability, backed with disk storage for durability, and transactional.

Demo

Stateless vs. Stateful

Reliable Services communication

- Service Fabric is agnostic about services communication.
  All protocols and stacks are acceptable (UDP, HTTP, WebSockets…)
- The Reliable Services API uses a simple interface for service communication

```
public interface ICommunicationListener
{
  Task<string> OpenAsync(CancellationToken cancellationToken);

  Task CloseAsync(CancellationToken cancellationToken);

  void Abort();
}
```

Reliable Services Listeners

```
class MyStatelessService : StatelessService
{
    protected override IEnumerable<ServiceInstanceListener> CreateServiceInstanceListeners()
    {
        ...
    }
    ...
}
```

```
class MyStatefullService : StatefullService
{
    protected override IEnumerable<ServiceReplicaListener> CreateServiceReplicaListeners()
    {
        ...
    }
    ...
}
```

Reliable Services Listeners

```
protected override IEnumerable<ServiceReplicaListener> CreateServiceReplicaListeners()
{
    return new[]
    {
        new ServiceReplicaListener(context =>
            new MyCustomHttpListener(context),
            "HTTPReadonlyEndpoint",
            true),

        new ServiceReplicaListener(context =>
            this.CreateServiceRemotingListener(context),
            "rpcPrimaryEndpoint",
            false)
    };
}
```

Reliable Services Listeners - endpoints

- The exposed endpoints are described in the ServiceManifest.xml

```xml
<Resources>
  <Endpoints>
    <Endpoint Name="ServiceEndpoint" Protocol="http" Port="80" />
    <Endpoint Name=" OtherServiceEndpoint" Protocol="tcp" Port="8585" />
  </Endpoints>
</Resources>
```

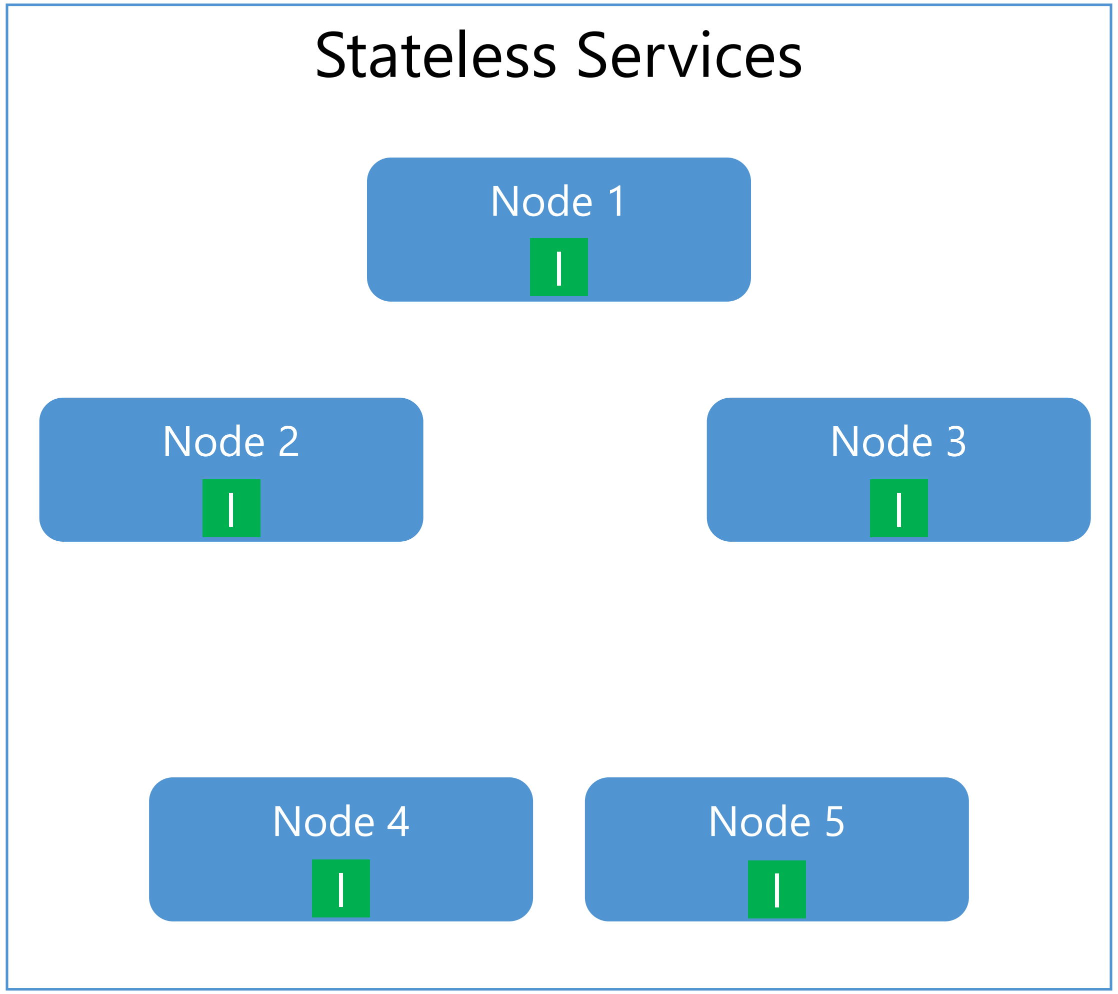
- The endpoint resources are accessible from code

```csharp
var codePackageActivationContext = serviceContext.CodePackageActivationContext;
var port = codePackageActivationContext.GetEndpoint("ServiceEndpoint").Port;
```

Demo

Stateless WebAPI

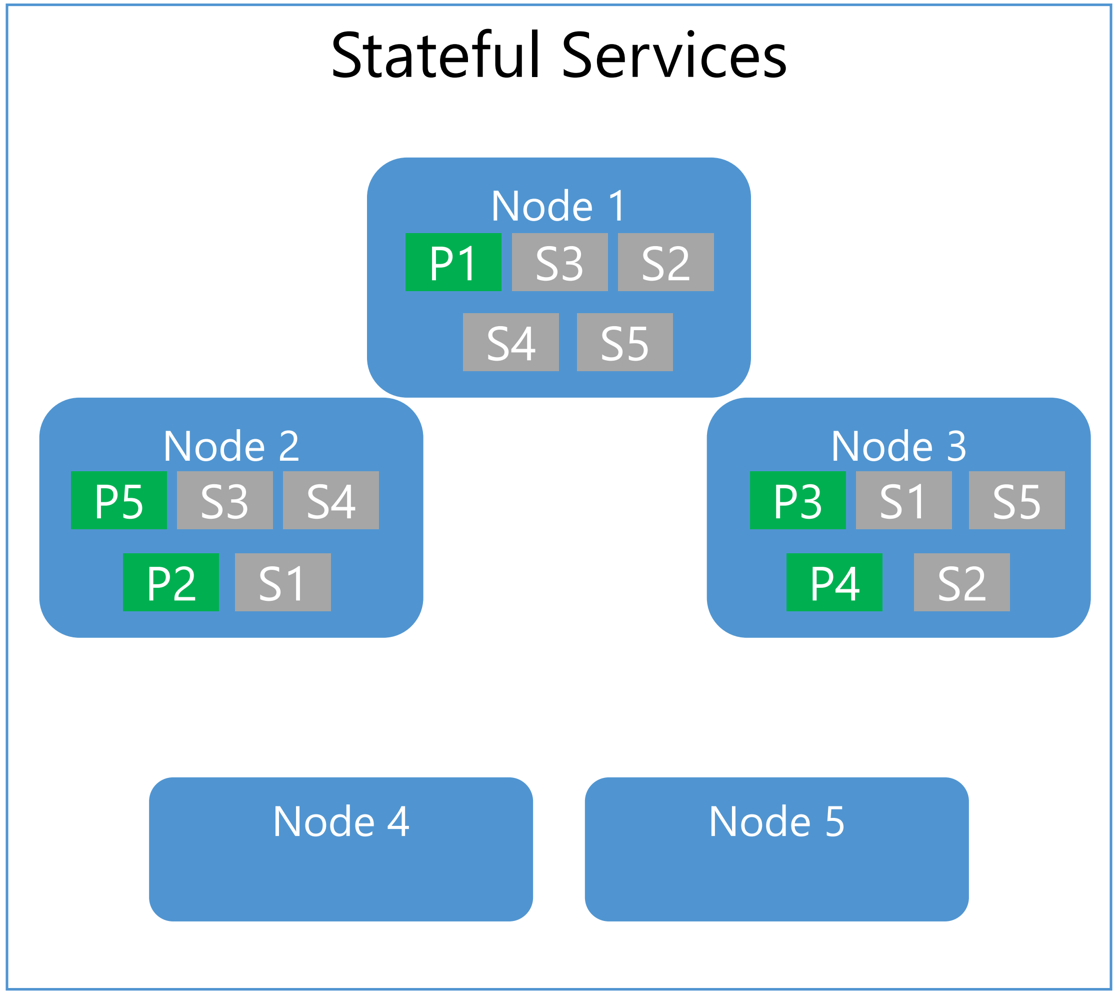
Scalability & Partitioning – Stateless Services



- In a stateless environment, scalability and availability generally achieved by adding instances
- Place an instance of the service in each Node
- Add Nodes to scale out
- Partitioning is often not needed in stateless services

Scalability & Partitioning – Stateful Services

- For stateful services
  - Services are Distributed for scalability
  - Service state is Replicated for availability
- Distribute primary replicas across the nodes in the cluster and also place secondary replicas
- Add Nodes, then rebalance to scale out



Communicating with a statefull service

- The first step is to resolve an endpoint address of the partition you want to talk to
  - *ServicePartitionResolver* does the runtime service endpoint resolution
- *ServicePartitionClient* provides the fault-handling and service partition address resolution loop

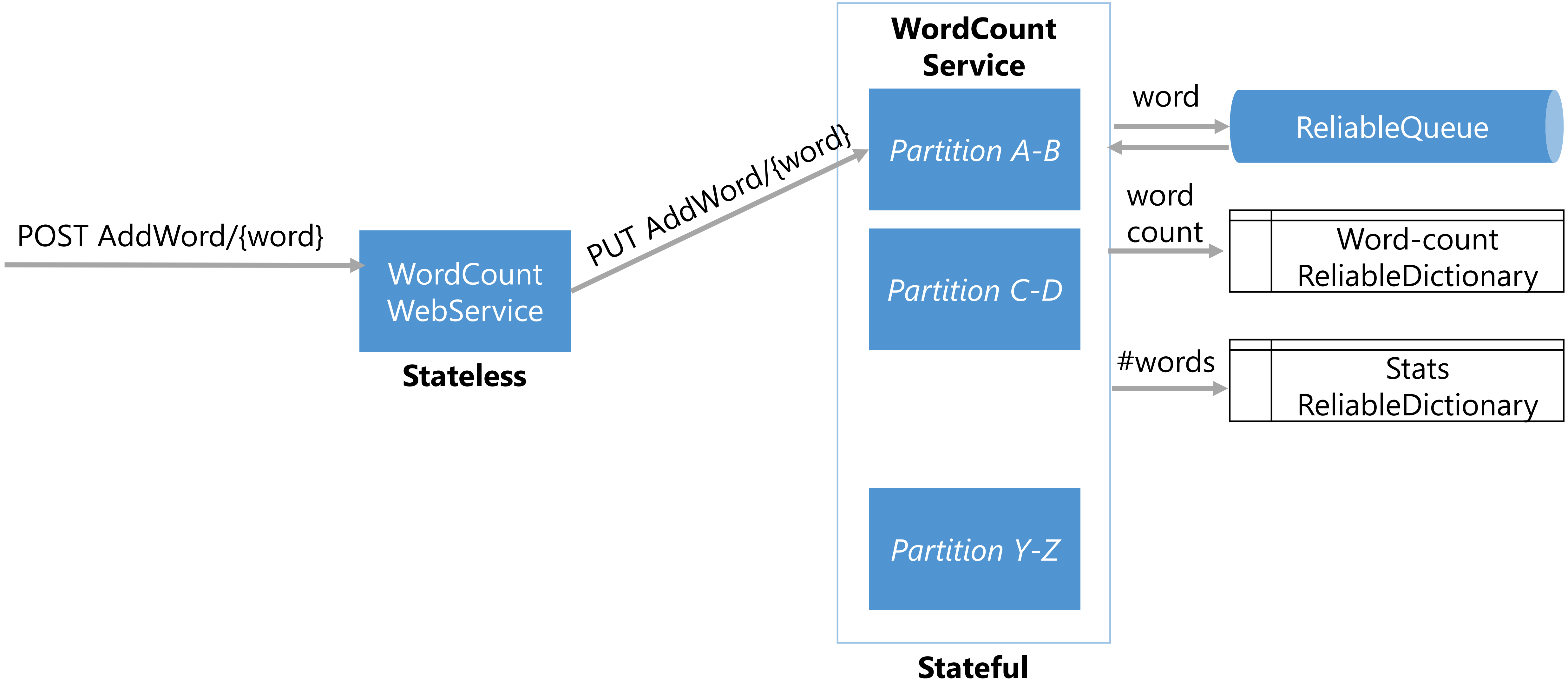
```
Var partitionClient = new ServicePartitionClient<HttpCommunicationClient>(
        communicationFactory,
        serviceUri,
        new ServicePartitionKey(partitionKey));

await partitionClient.InvokeWithRetryAsync(...);
```

Demo

WordCount – Partitioning the ABC

WordCount Architecture



Reliable Actors

- Implementation of the Virtual Actor pattern
  - Ideal when the application involves many (thousands) of small isolated units of logic & state
- Built on top of Stateful Reliable Services
- Actors are instantiated upon reference and collected upon inactivity
  - State Management can be used during reference to “restore” an actor
- Actor actions are turn-based (single-threaded) per instance.
  - Supports Timers and Reminders
  - Also supports Events

Deployments & Upgrades

Deploying in Azure

- Service Fabric Applications are deployed in Azure into a Virtual Machine Cluster in a Virtual Machine Scale Set (VMSS)
- Clusters can be created on VMs running Windows Server 2012 R2, Windows Server 2016, or Linux Ubuntu 16.04 (in preview)
- Clusters networking can be configured via Azure VNets, Subnets, Public IP Addresses, and internal and/or external load balancing
- Can be managed with Azure ARM templates,
- Auto-scaling support is provided by Azure VMSS
- Management is via a combination of Azure tools and Service Fabric tools

Demo

Deploying to Azure

Application Deployments

- Two types of deployments in Service Fabric – Full and Upgrade
- In a Full Deployment, the entire application is torn down (if already present) and then a new application instance is deployed.
- Upgrade Deployments
  - Service Fabric supports rolling upgrades to maintain uptime
  - Service Fabric monitors application health during the upgrade. If the system violates the application health policy, the upgrade is rolled back.
  - Several modes govern the automation and health-check behavior during an upgrade.

Summary

- Monolith vs. Microservices
- Service Fabric Orchestration
- Stateless vs. Stateful services
  - Scalability and Partitioning
  - Reliable services Programming model
- Deploying to Local Cluster
- Deploying to Azure
- Upgrades

Resources

- Service Fabric Documentation
  https://docs.microsoft.com/en-us/azure/service-fabric/
- Samples
  - https://github.com/Azure-Samples/service-fabric-dotnet-getting-started
  - https://github.com/Azure-Samples/service-fabric-dotnet-containers

Workshop

Get Your Hands Dirty!

- https://github.com/CodeValue/AzureBootcamp-ServiceFabric