

DevTest 9.x – Best Practice Architecture

#3 – Multiple Teams – multiple DevTest server installations with multiple Registries

Introduction

Customers often ask “Where do we Start” when setting up and configuring their DevTest architecture. Of course, this really depends on how they plan to use DevTest - whether using Virtual Services or running tests or maybe a combination of both. So based on potential usage of DevTest, we started building a Best Practice Architecture Guide based on real world deployments. As a part of this initiative, we will build out several architectures that customers or CA Services have implemented, or that are new/tested architectures that fit a specific need.

Our intention is to provide options for customers, partners and CA Services on how to implement DevTest within their environment to meet the specific use cases.

This document will contain guidance on implementing DevTest on premise, in private or public clouds or hybrid environments. It will describe the specific DevTest Capabilities.

This third use case walks through setting up DevTest 9.1 in a Microsoft® Windows Azure (Azure) environment for multiple teams that do not work on a common project, but need segregated and separated resources to create, test and operate automated tests and virtual services. The Source Control Management (SCM) is located on premise (on physical systems or in private cloud) for access by the clients.

Document Changes

Version	Date	Primary Author	Description
1.0	04/07/2016	Ulrich Vogt, Koustubh Warty	Initial creation

Contents

Introduction	1
Document Changes	1
Business Case	3
Architectural Considerations	4
Architecture Diagram	4
DevTest Implementation Details	5
System Setup	7
Virtual Network and subnets	7
Network Security Group	7
Windows Firewalls	8
Firewall Policies	8
Enable ICMP Echo (ping)	9
Ports exposed for DevTest Server Components	10
Inbound Access	10
Outbound Access	11
Virtual Machines	12
DNS Service Registration	13
Use Case #3 Deployment in Azure	13
Architecture Diagram	13
Virtual Network and subnets	14
DevTest-Net-EDS	16
devtest-uc0-eds	17
DevTest-Net-UC31	18
devtest-u3a-reg	19
devtest-u3a-vse	20
devtest-u3a-sim	21
Devtest-u3a-db	22
DevTest-Net-UC32	23
DevTest-Net-UC33	23
DevTest configuration	23
Verification of DevTest Environment Setup	24
DevTest Server Name Mapping (NAT)	24

DevTest Web UI Access Connectivity.....	25
Enterprise Dashboard Web UI Access from customer network	25
DevTest Server Console Access from client network.....	26
DevTest Portal Access from client network	27
DevTest Portal Connectivity Tests from client network	27
Running Tests from DevTest Portal	28
Deploy Virtual Service to VSE from the DevTest Portal	29
DevTest Workstation Connectivity Tests from client network	30
Connect local Workstation to DevTest Registry in cloud.....	30
Connect local client to Application in cloud.....	31
Connect local client to virtual service in cloud	32
Stage test from local DevTest workstation to cloud	33
Deploy Virtual Service from local DevTest workstation to cloud	35
Appendix	38
VM customization	38
OS Configuration	38
Installed Software	38
Service configuration	38
SCM Details	38

Business Case

This architecture sample covers the business case of a customer who wants to deploy a DevTest configuration in a public cloud for internal or external teams to access. Different teams, who do not need or do not want to share projects and do not want to share test resources, will have their separate DevTest server environments. Each DevTest environment connects to a single DevTest Enterprise Dashboard to report DevTest resource usage and to query licensed features. The customer queries this information from the Enterprise Dashboard Web UI.

In this example, each DevTest server deployment consists of installing and configuring the DevTest server components in Microsoft® Azure environment on individual VMs as explained in the document. We believe that a single VSE or Simulator instance per node is the simplest option for administering the Dev Test environment in the cloud.

The SCM will be on the premise on the client network. Every time a DevTest user works on any DevTest asset, namely Test Cases, Virtual Services, mar files, etc., they will checkout from the SCM server locally, and after modification of those assets, will check back the modified asset(s) to the SCM server.

Architectural Considerations

Following restrictions apply to a DevTest architecture and are recommended to follow:

1. To warrant performance Registry database must be located electronically close to DevTest server components. Please see [Database Requirements](#) for details.
2. Enterprise Dashboard database contains audit data. Resources related to audit data should persist and must not be removed.
3. The SCM used here is the free standard edition of the Visual SVN server from <https://www.visualsvn.com/server/>
4. Every user machine will have a client that can talk to the SVN server. In this case, we have used the free version of Tortoise SVN Client from <https://tortoisesvn.net/downloads.html>

Note: This document does not go into the details on installing and configuring the SVN server and the client.

This architecture design applies to Service Virtualization and Application Test.

Note: In this architecture, the DB server is part of the cloud installation, and therefore, by default, supposed to be available for a limited amount of time only. Decommissioning the Virtual Machines in the cloud as used in this DevTest architecture will also delete all DevTest reports stored in database. If the customer wants DevTest reports to be available after destroying the DevTest environment, s/he needs to consider a DB backup strategy.

Architecture Diagram

The Architecture diagram shows seven different DevTest deployments:

- A 'Customer' environment hosting the Enterprise Dashboard service only for monitoring, reporting and auditing the usage of DevTest services.
- Three 'Client' environments, representing teams that are internal or external to the 'Customer' environments. A single workstation represents each team, which can be comprise of multiple users on multiple workstations accessing the same DevTest server deployment in Microsoft® Azure.
- Three 'Azure' environments, representing DevTest server deployments. Each Azure environment is assigned to a 'Client'.

The Architecture diagram shows ports that need to be open for external DevTest service access. For the sake of simplicity, it does not show ports that needs to be open for other services, such as Remote Desktop Protocol or Windows PowerShell.

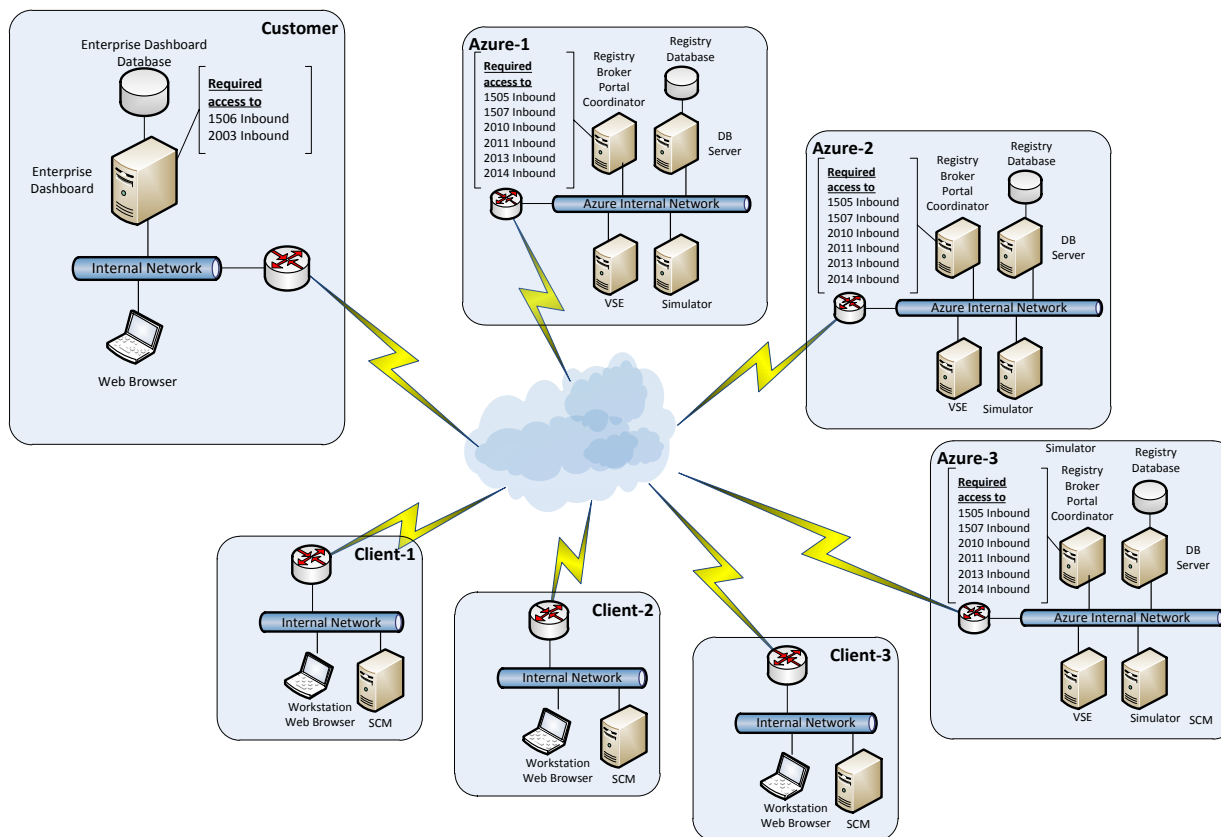


Figure 1 - Architecture Diagram

DevTest Implementation Details

Installation of some DevTest components depends on business case requirement. Most components are required regardless of use case to provide the DevTest infrastructure.

In order to run Service Virtualization or Application Test in cloud environments DevTest VSE and DevTest Simulator services must be provided. These components require additional DevTest services installed and running:

DevTest service	Requires
Simulator	Coordinator, Registry
VSE	Registry
<i>Shared among DevTest components</i>	Portal

- Enterprise Dashboard

The Enterprise Dashboard service and its database are installed on premise or in a private or public cloud. For the sake of simplicity, the database is installed locally on the same system as the Enterprise Dashboard service, but a dedicated database server can also be used.

Installation of the Enterprise Dashboard service is mandatory.

- Registry

Registry service and its database are installed on an Azure VM in the cloud. The Registry database is installed on a dedicated database server to better visualize the need for a database resource. The Registry Service requires access to the Enterprise Dashboard service for license information and for providing usage data.

Installation of a Registry service is required for every DevTest use case.

- Broker

Broker Service is installed on the same system as the Registry Service. Broker Service is required for communication with the DevTest agent, especially when using Continuous Application Insight.

- Portal

Portal Service is installed on the same system as the Registry Service. Portal Service is required for Web access to DevTest server components.

- Coordinator

Coordinator Service is installed on the same system as the Registry Service. Coordinator Service is required to stage test cases to Simulator Services. A Coordinator Service can manage multiple Simulator services.

Coordinator services are required for Application Test use cases only.

- Simulator

Simulator service is installed on a separate Azure VM. If multiple Simulator services are required for scalability reasons, they are assumed to be installed on different Azure VMs.

Simulator services are required for Application Test use cases only. Therefore, VMs for Simulator services can be provided on demand.

- Virtual Service Environment (VSE)

VSE service is installed on a separate Azure VM. If multiple VSE services are required for scalability reasons, they are assumed to be installed on different Azure VMs.

VSE services are required for Service Virtualization only. Therefore, VMs for VSE services can be provided on demand.

System Setup

DevTest is deployed in a distributed configuration. It is distributed in separate environments, on premise and in Microsoft® Azure. The different client teams run the local DevTest Workstations on premise, DevTest Server run in separate subnets of a Microsoft® Azure Virtual Network.

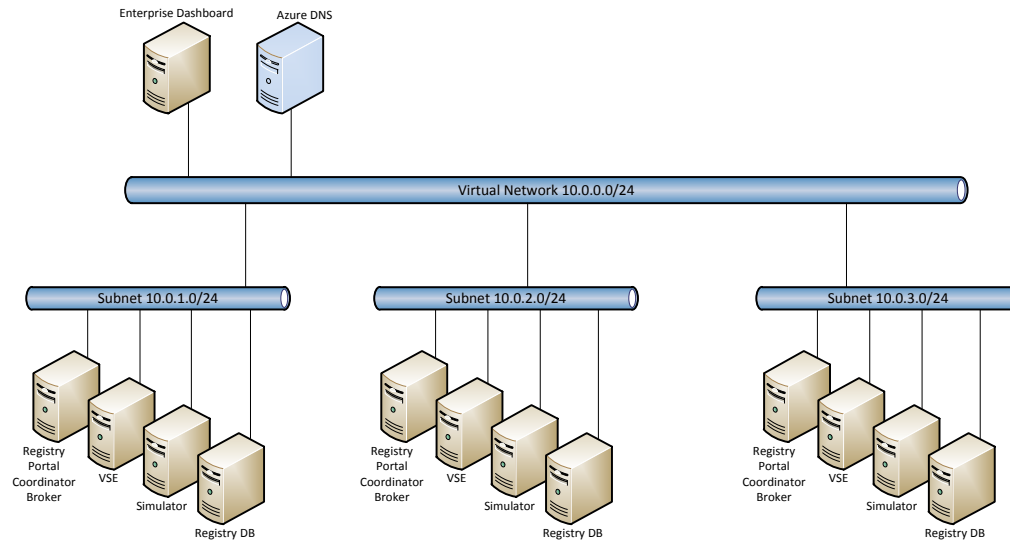


Figure 2 - Microsoft(R) Azure Virtual Network diagram for DevTest Servers

The order to create the environment for DevTest server deployments in Microsoft® Azure is as follows

1. Create Azure Virtual Network for DevTest Enterprise Dashboard
2. Create subnets in Azure Virtual Network per DevTest Server deployment
3. Create DevTest Network Security Groups for subnets and for DevTest servers Registry, database, VSE and Simulator
4. Assign Network Security Group to subnets
5. Create DevTest server VMs for each DevTest server deployment, connect them to the according subnet and assign the appropriate DevTest server specific Network Security Group.

Microsoft® offers two types of portals to manage a Microsoft® Azure environment. For setting up this architecture, we used the new portal (<https://portal.azure.com>).

Virtual Network and subnets

In Microsoft® Azure, the user can create Virtual Networks. Each Virtual Network can host several subnets. There are several locations available to run the Virtual Network. In our sample, this Virtual Network is located in the Central US region of Azure.

A Virtual Network can also host an Azure DNS server or a custom DNS server to resolve the server names of all the VMs in this Virtual Network.

Network Security Group

In Microsoft® Azure a Network Security Group secures access to both subnets and virtual machines in Microsoft® Azure. By default, Network Security Groups block any inbound network traffic that is not required to access the provided services, but allow any outbound traffic. Therefore, the user must

configure inbound rules for Network Security Groups on both subnet and on VM level to allow access to DevTest services.

Microsoft® Azure creates the RDP specific inbound port rule as part of the VM provisioning process automatically if a new Network Security Group is created automatically during this process. Otherwise, the inbound RDP port rule must be added manually to a manually created Network Security Group.

Other than for Microsoft® Azure Cloud Service no inbound port rule is created for Powershell access to the Windows 2012 R2 VMs.

For other services that users need remote access to from the Internet, such as some of the DevTest services, the DevTest administrator of the Azure environment has to create additional inbound port rules.

Important: Definition of the Microsoft® Azure Network Security Group for a VM does not replace firewall settings within the VM. A VM's firewall can still block custom ports that pass the Network Security Groups in Microsoft® Azure.

Windows Firewalls

In Microsoft® Azure VMs running Windows 2012 or Windows 2012 R2, the default firewall settings block any inbound traffic unless matched by a rule, and allow any outgoing traffic unless matched by a rule. Provisioning Windows 2012 or Windows 2012 R2 Microsoft® Azure creates firewall rules automatically for ports required by Remote Desktop Protocol (RDP) or PowerShell.

Firewall Policies

All (Windows) virtual machines of the DevTest cloud in Azure are configured with identical firewall settings:

- Same firewall settings for Domain Profile, Private Profile and Public Profile
 - Firewall is activated (on) for Domain and Public Profiles, but off for Private Profile
- By default, Windows 2012 server and Windows 2012 R2 server templates in Microsoft® Azure block any inbound traffic unless matched by a rule and allow any outbound traffic unless matched by a rule.
- If the firewall is not configured accordingly then there are no Inbound and Outbound connections available between the VMs and the outside world

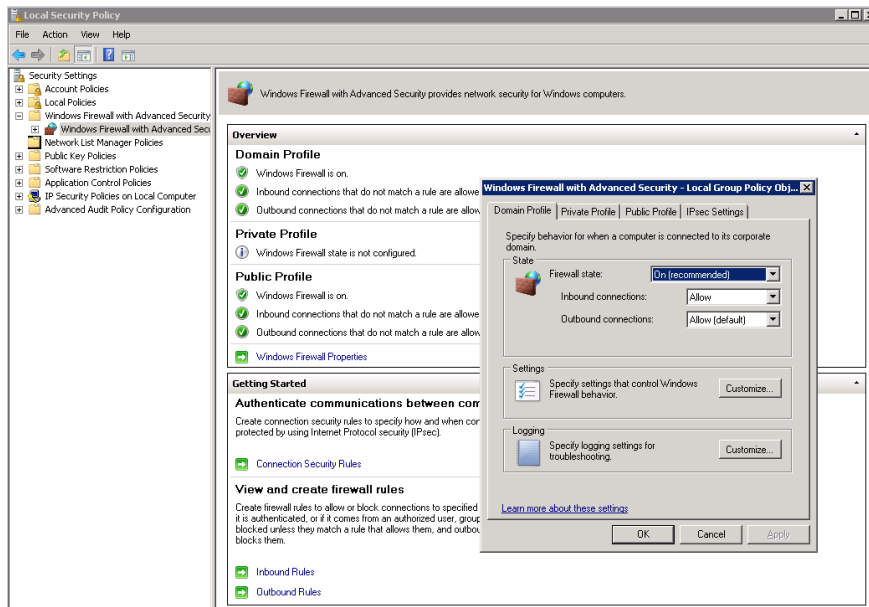


Figure 3 - Firewall settings for Domain Profile

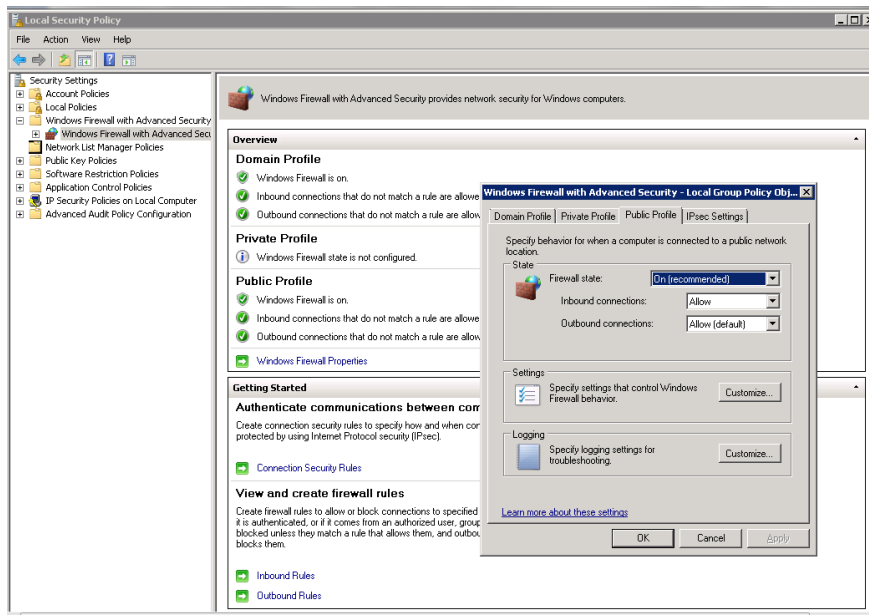


Figure 4 - Firewall setting Public profile

Enable ICMP Echo (ping)

Ping is often used to check reachability of systems. By default, *ping* is blocked by Windows firewalls and Azure Network Security Groups.

In order to enable ICMP echo protocol in Windows Server 2012 R2 to response to *ping* requests on IPv4 the following inbound rule must be enabled for all profiles:

Inbound Rules											
Name	Group	Profile	Enabled	Action	Override	Program	Local Address	Remote Address	Protocol	Local Port	
File and Printer Sharing (Echo Request - ICMPv4-In)	File and Printer Sharing	All	Yes	Allow	No	Any	Any	Any	ICMPv4	Any	

There is an equivalent role to enable ICMP echo on IPv6.

In Azure Network Security Groups a rule allowing all inbound traffic also lets ICMP Echo protocol elements pass.

The screenshot shows the configuration for an 'All-In' Network Security Group rule. The rule name is 'All-In'. The priority is set to 100. The source is set to 'Any' (with options for CIDR block and Tag). The protocol is set to 'Any' (with options for TCP and UDP). The source port range is set to '*'. The destination is set to 'Any' (with options for CIDR block and Tag). The destination port range is set to '*'. The action is set to 'Allow' (with an option for Deny).

Figure 5 - Allow all protocols rule

Ports exposed for DevTest Server Components

Network Security Groups control access to services available on the virtual machines in Microsoft® Azure. Network Security Groups are associated to subnets and VMs. Network Security Groups can be assigned, created, edited during VM provisioning process or afterwards.

Inbound Access

The following services on the virtual machines running Microsoft® Windows Server need inbound access for remote users:

- Windows Services on all virtual machines running Windows Operating System

Service name	Default Port no.
Remote Desktop	3389
PowerShell	5986

- DevTest Services
 - Enterprise Dashboard

Service name	Default Port no.
Enterprise Dashboard	1506

Enterprise Dashboard CIC	2003
--------------------------	------

- Registry Server

Service name	Default Port no.
Registry	2010
Coordinator	2011
Server Console Web UI	1505
Portal UI	1507
LISA Bank	8080
Forward Cars Web UI	3434

- VSE Servers need inbound access configured for any deployed virtual service. Following port assignments are examples:

Virtual Service name	Example
LISA Bank	8001
Forward Cars Web UI VS	3434
Forward Cars Inventory VS	3500

Outbound Access

When installing DevTest components on VMs in the cloud, DevTest Registry services access the DevTest Enterprise Dashboard service. Therefore, outbound access from Microsoft® Azure VMs to Enterprise Dashboard service on customer premise on default port 1506 (DevTest 9.1) and port 2003 (DevTest < 9.1) must be available.

Following services on these virtual machines running Microsoft® Windows Server may need outbound access:

- DevTest Services

Service name	Default Port no.
Enterprise Dashboard	2003

The following screenshots show the Network Security Groups and mappings to private ports on the different VMs

Virtual Machines

The DevTest Server components run in Microsoft® Azure virtual machines. These are configured with following resources:

Host name	Private IP	Size	#NICs
devtest-uc0-edb	<dynamic>	Basic_A2	1 NIC
devtest-u3a-reg	<dynamic>	Basic_A2	1 NIC
devtest-u3a-vse	<dynamic>	Basic_A2	1 NIC
devtest-u3a-sim	<dynamic>	Basic_A2	1 NIC
devtest-u3a-db	<dynamic>	Basic_A2	1 NIC
devtest-u3b-reg	<dynamic>	Basic_A2	1 NIC
devtest-u3b-vse	<dynamic>	Basic_A2	1 NIC
devtest-u3b-sim	<dynamic>	Basic_A2	1 NIC
devtest-u3b-db	<dynamic>	Basic_A2	1 NIC

Microsoft® Azure assigns public IP addresses dynamically upon VM startup. Microsoft® Azure Portal displays a VM's public IP address:

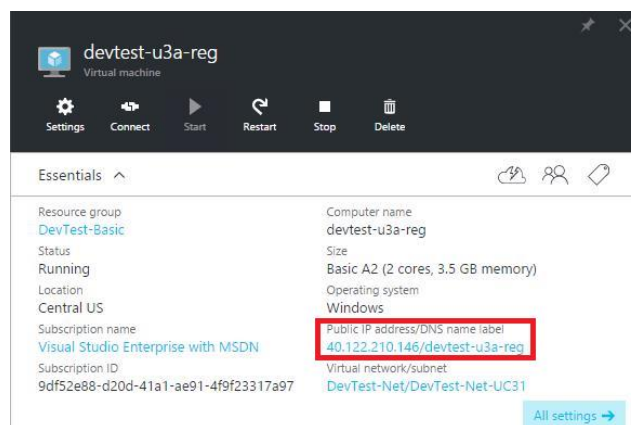


Figure 6 - VM private IP address

In current Microsoft® Azure version, the VM sizes represent following resources:

Size Type	Cores	RAM	#Disk	Disk size
Basic_A3	4	7 GB	2	130 GB / 120 GB
Basic_A2	2	3.5 GB	2	130 GB / 60 GB

DNS Service Registration

After provisioning, a VM must register to Azure DNS service, by assigning a DNS name.

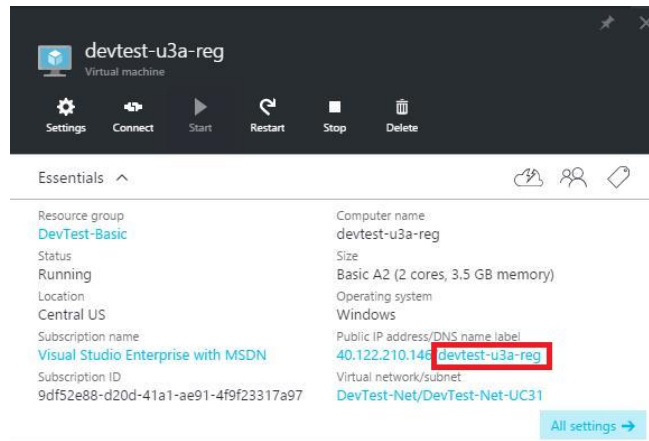


Figure 7 - DNS name

DNS Service in Microsoft® Azure is able to resolve the VM name within the virtual network. On the client network, the Microsoft® Azure DNS server has to be integrated with the DNS server on the client network. See [Virtual Networks Overview](#) for details on virtual networks in Azure.

All Virtual Machines running DevTest components use Microsoft® Windows 2012 Server or Windows 2012 R2. Both are 64-Bit systems.

Use Case #3 Deployment in Azure

Architecture Diagram

The Architecture diagram shows seven different DevTest deployments:

- A 'Customer' environment hosting the Enterprise Dashboard service only for monitoring, reporting and auditing the usage of DevTest services.
- Three 'Client' environments, representing teams that are internal or external to the 'Customer'. A single workstation represents each team, which can be comprised of multiple users on multiple workstations accessing the same DevTest server deployment in Microsoft® Azure.

- Three 'Azure' environments, representing DevTest server deployments. Each Azure environment is assigned to a 'Client'.

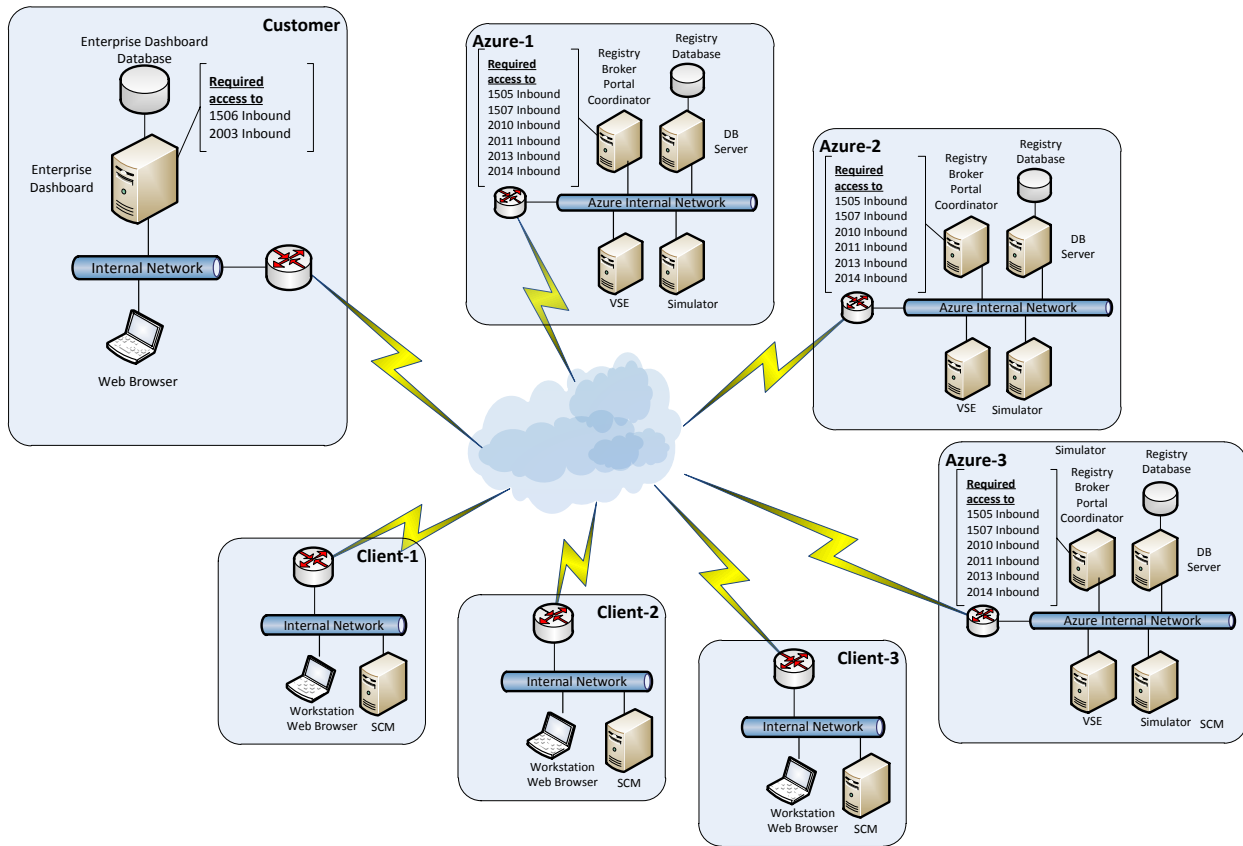


Figure 8 - Architecture Diagram Use Case #3

Virtual Network and subnets

In Microsoft® Azure, we create a Virtual Network called *DevTest-Net* to host the entire DevTest deployment. We assign it to Microsoft® Azure Resource Group *DevTest-EDS*.

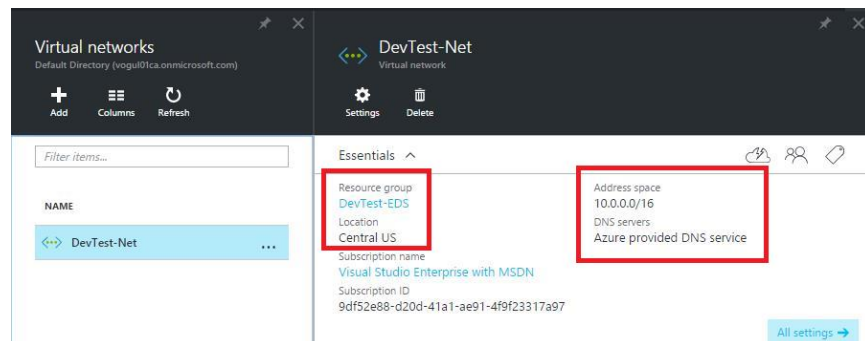


Figure 9 - Azure Virtual Network

There are several locations available to run the Virtual Network. In our sample, this Virtual Network is located in the Central US region of Azure. It got address space 10.0.0.0/16 assigned, which can have 4096 addresses.

It also got an Azure DNS server to resolve the server names of all the VMs in this Virtual Network.

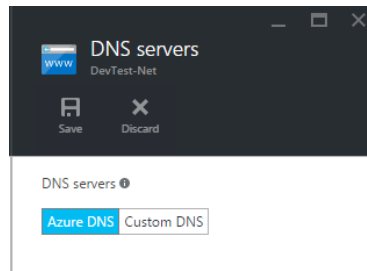


Figure 10 - Azure DNS Server Definition

We separate the four DevTest Server environments for the customer and for the three different client teams by subnets in this Microsoft® Azure Virtual Network:

Name	Subnet name	Subnet	DevTest version
Customer	DevTest-Net-EDS	10.0.0.0/24	9.1
Azure-1	DevTest-Net-UC31	10.0.1.0/24	9.1
Azure-2	DevTest-Net-UC32	10.0.2.0/24	9.0
Azure-3	DevTest-Net-UC33	10.0.3.0/24	-

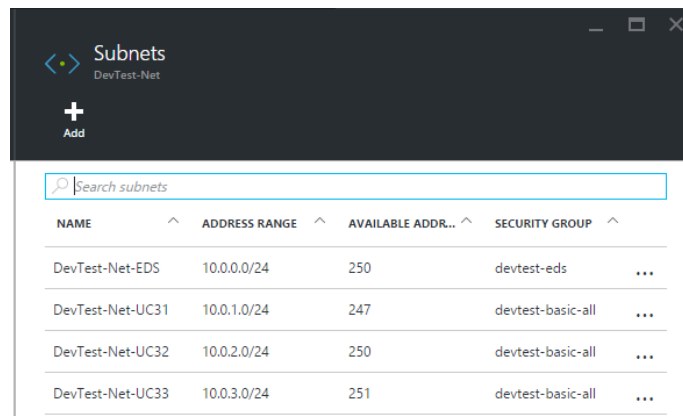


Figure 11 - Subnets in Azure Virtual Network

Microsoft® Azure subnet 'DevTest-Net-EDS' represents the customer environment, hosting the Enterprise Dashboard Service that monitors the resource usage by all the other DevTest server environments. Microsoft® Azure subnet 'DevTest-Net-UC31', 'DevTest-Net-UC32' and 'DevTest-Net-UC33' host the client specific DevTest server deployments. We omit deployment of 'Azure-3', as it would be a copy of 'Azure-1'.

We recommend to use different Microsoft® Azure subnets for each DevTest Server deployment in order to separate and segregate client specific DevTest server deployments.

DevTest-Net-EDS

Microsoft® Azure subnet ‘DevTest-Net-EDS’ hosts the single VM ‘devtest-uc0-eds’ that runs the customer’s Enterprise Dashboard

Network Security Group *devtest-eds* controls access to subnet ‘DevTest-Net-EDS’

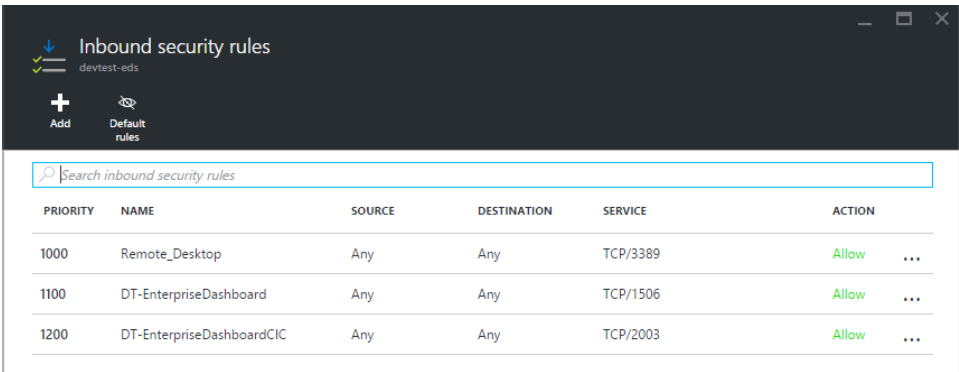


Figure 12 - Network Security Group for Enterprise Dashboard subnet

The Enterprise Dashboard server ‘devtest-uc0-eds’ is configured with following resources:

Host name	Private IP	Size	#NICs
devtest-uc0-edb	<dynamic>	Basic_A2	1 NIC

VM Host name	Public address for RDP
devtest-uc0-edb	devtest-uc0-edb.centralus.cloudapp.azure.com:3389

devtest-uc0-eds

'devtest-uc0-eds' hosts the DevTest Enterprise Dashboard service.

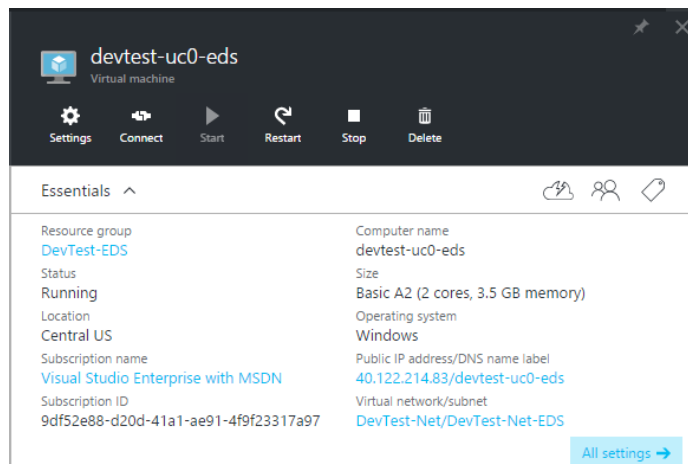


Figure 13 - VM devtest-uc0-eds

Assuming that all outbound traffic is allowed the DevTest administrator has to allow inbound traffic to access Enterprise Dashboard services in the firewall on domain, public and private networks:

Service Name	Protocol	Port	Profile	Required by
DevTest Enterprise Dashboard	TCP	1506	External	DevTest Registry 9.1 services in Cloud Services
DevTest Enterprise Dashboard CIC (Legacy Registry earlier than 9.1)	TCP	2003	External	DevTest Registry 9.0 (or earlier) services in Cloud Services

Network Security Group *devtest-eds* makes these port numbers public.

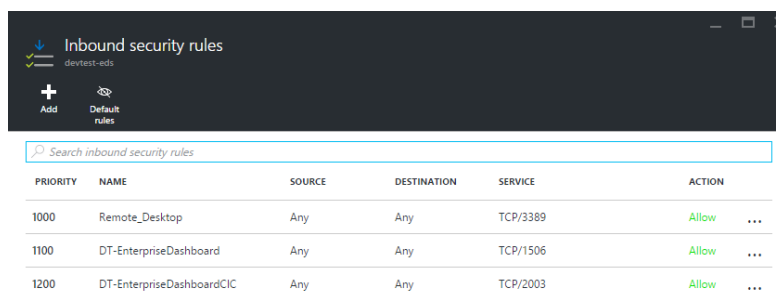
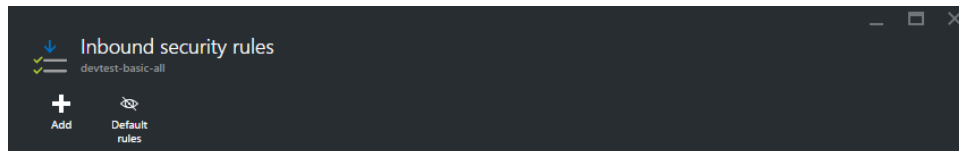


Figure 14 - Network Security Group for subnet DevTest-Net-EDS and for VM devtest-uc0-eds

Network Security Group *devtest-eds* controls access to subnet *devtest-net-eds* and to network connection of *devtest-uc0-eds*.

DevTest-Net-UC31

Network Security Group *devtest-basic-all* secures access to each of the DevTest server subnets *DevTest-Net-UC31*, *DevTest-Net-UC32* and *DevTest-Net-UC33*:



PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
1000	Remote_Desktop	Any	Any	TCP/3389	Allow ...
1100	DT-ServerConsole	Any	Any	TCP/1505	Allow ...
1200	DT-Portal	Any	Any	TCP/1507	Allow ...
1300	DT-Broker	Any	Any	TCP/2009	Allow ...
1400	DT-Registry	Any	Any	TCP/2010	Allow ...
1500	DT-Coordinator	Any	Any	TCP/2011	Allow ...
1600	DT-VSE	Any	Any	TCP/2013	Allow ...
1700	DT-Simulator	Any	Any	TCP/2014	Allow ...
1800	DT-LisaBank	Any	Any	TCP/8080	Allow ...
1900	DT-LisaBankVS	Any	Any	TCP/8001	Allow ...
2000	DT-FwdCars	Any	Any	TCP/3434	Allow ...
2100	DT-FwdCarsVS	Any	Any	TCP/3434	Allow ...
2200	DT-FwdCarsInvVS	Any	Any	TCP/3500	Allow ...

Figure 15 - Network Security Group for Registry subnets

Microsoft® Azure subnet 'DevTest-Net-UC31' hosts the DevTest 9.1 server environment for 'client-1'

Host name	Private IP	Size	#NICs
devtest-u3a-reg	<dynamic>	Basic_A2	1 NIC
devtest-u3a-vse	<dynamic>	Basic_A2	1 NIC
devtest-u3a-sim	<dynamic>	Basic_A2	1 NIC
devtest-u3a-db	<dynamic>	Basic_A2	1 NIC

VM Host name	Public address for RDP
devtest-u3a-reg	devtest-u3a-reg.centralus.cloudapp.azure.com:3389
devtest-u3a-db	devtest-u3a-db.centralus.cloudapp.azure.com:3389
devtest-u3a-sim	devtest-u3a-sim.centralus.cloudapp.azure.com:3389
devtest-u3a-vse	devtest-u3a-vse.centralus.cloudapp.azure.com:3389

devtest-u3a-reg

'devtest-u3a-reg' hosts central DevTest services, such as registry, broker, portal, and coordinator.

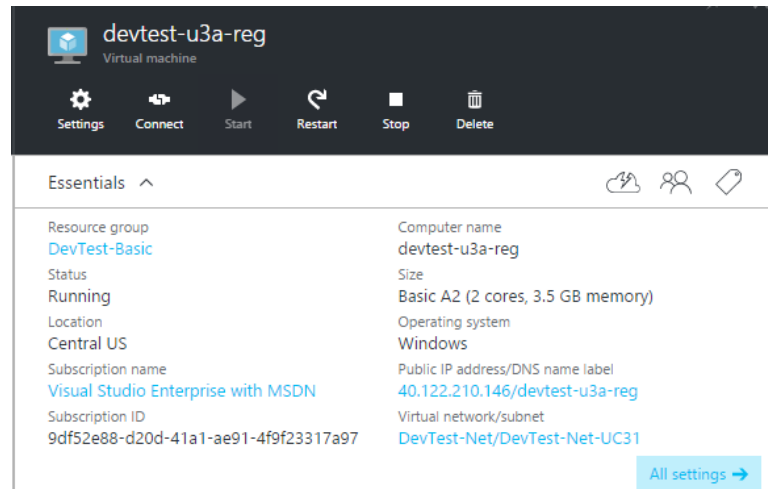
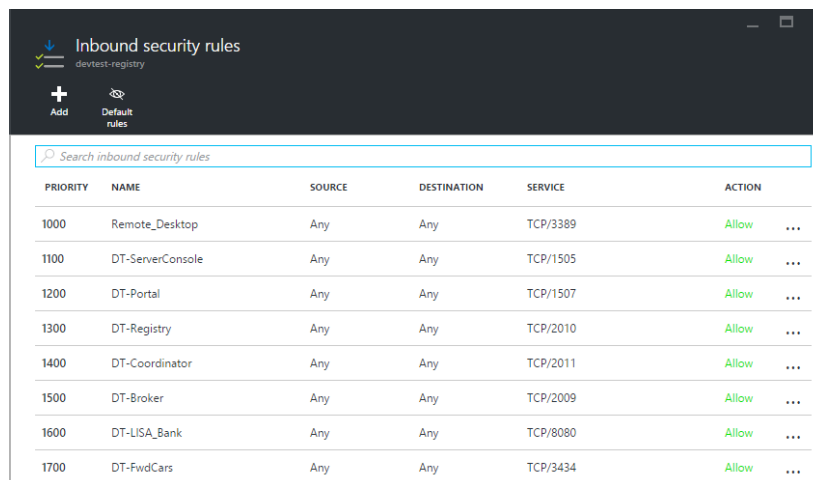


Figure 16 – VM devtest-u3a-reg

Assuming that all outbound traffic is allowed the DevTest administrator has to allow following inbound traffic in the firewall on domain, public and private networks to access following DevTest services:

Service Name	Protocol	Port	Profile	Required by
DevTest Server Console	TCP	1505	External	Server Console Web UI
DevTest Portal	TCP	1507	External	Portal Web UI
DevTest Registry	TCP	2010	External	Workstation registration
DevTest Coordinator	TCP	2011	External	Staging tests from Workstation
DevTest Broker	TCP	2009	External	DevTest Java Agent outside of Cloud Service
LISA Bank	TCP	8080	External	LISA Bank kiosk access from client network

Network Security Group *devtest-registry* allows inbound traffic for these ports of services provided by the VM.



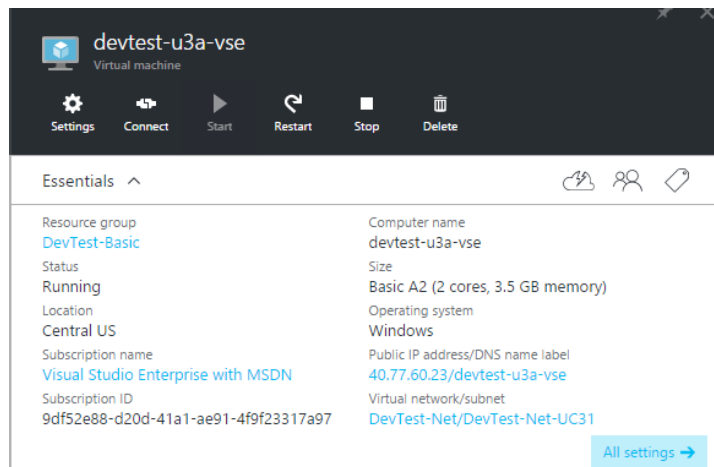
PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
1000	Remote_Desktop	Any	Any	TCP/3389	Allow ...
1100	DT-ServerConsole	Any	Any	TCP/1505	Allow ...
1200	DT-Portal	Any	Any	TCP/1507	Allow ...
1300	DT-Registry	Any	Any	TCP/2010	Allow ...
1400	DT-Coordinator	Any	Any	TCP/2011	Allow ...
1500	DT-Broker	Any	Any	TCP/2009	Allow ...
1600	DT-LISA_Bank	Any	Any	TCP/8080	Allow ...
1700	DT-FwdCars	Any	Any	TCP/3434	Allow ...

Figure 17 - Network Security Group for DevTest Registry services

External Access to the DevTest broker service (default port 2009) is necessary in case DevTest agents need to connect to the DevTest broker service. The application running the DevTest agents has to be placed electronically close to the broker service.

devtest-u3a-vse

'devtest-u3a-vse' hosts the DevTest Virtual Service Environment running virtual services.



devtest-u3a-vse Virtual machine	
Settings	Connect
Start	Restart
Stop	Delete
Essentials	
Resource group	Computer name
DevTest-Basic	devtest-u3a-vse
Status	Size
Running	Basic A2 (2 cores, 3.5 GB memory)
Location	Operating system
Central US	Windows
Subscription name	Public IP address/DNS name label
Visual Studio Enterprise with MSDN	40.77.60.23/devtest-u3a-vse
Subscription ID	Virtual network/subnet
9df52e88-d20d-41a1-ae91-4f9f23317a97	DevTest-Net/DevTest-Net-UC31
All settings	

Figure 18 - VM devtest-u3a-vse

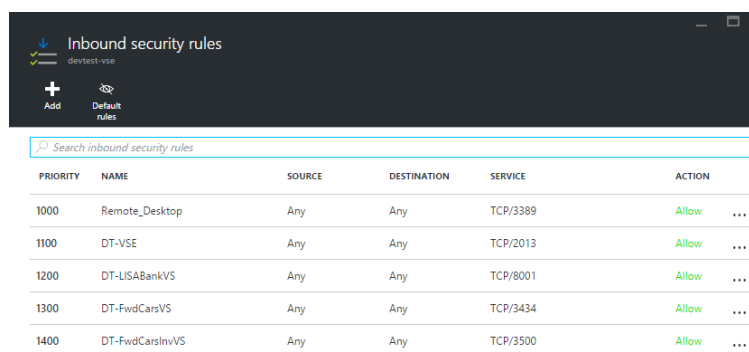
Assuming that all outbound traffic is allowed, the DevTest administrator has to allow the following inbound traffic in the firewall on domain, public and private networks to access the following DevTest services:

Service Name	Protocol	Port	Profile	Required by
DevTest VSE	TCP	2013	External	Deploy a VS from Workstation

DevTest LISABank VS	TCP	8001	External	Access to LISABank VS from client network
DevTest Forward Cars VS	TCP	3434	External	Access to Forward Cars VS from client network
DevTest Forward Cars Inv VS	TCP	3500	External	Access to Forward Cars Inventory VS from client network

External access to the VSE service is required to query and deploy virtual services from local workstations. External access to deployed virtual services and their URLs is required for client communication.

Network Security Group *devtest-vse* allows inbound traffic for these ports of services provided by the VM:



PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
1000	Remote_Desktop	Any	Any	TCP/3389	Allow ...
1100	DT-VSE	Any	Any	TCP/2013	Allow ...
1200	DT-LISABankVS	Any	Any	TCP/8001	Allow ...
1300	DT-FwdCarsVS	Any	Any	TCP/3434	Allow ...
1400	DT-FwdCarsInvVS	Any	Any	TCP/3500	Allow ...

Figure 19 - Network Security Group for VSE

External access to port numbers of virtual services must be enabled. In this example, port numbers to virtual services for LISA Bank, Forward Cars Web Service and Forward Cars Inventory Services are enabled for external access.

devtest-u3a-sim

'devtest-u3a-sim' hosts the DevTest Simulator service required for running automated tests managed by the DevTest Coordinator Service.

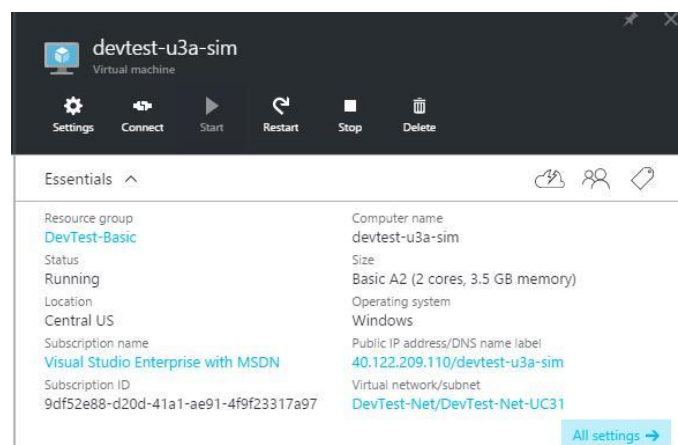


Figure 20 - VM devtest-u3a-sim

Assuming that all outbound traffic is allowed, the DevTest administrator has to allow the following inbound traffic in the firewall on domain, public and private networks to access the following DevTest services:

Service Name	Protocol	Port	Profile	Required by
DevTest Simulator	TCP	2014	External	Workstation to retrieve test events of a completed staged test case

Network Security Group *devtest-sim* allows inbound traffic for these ports of services provided by the VM:

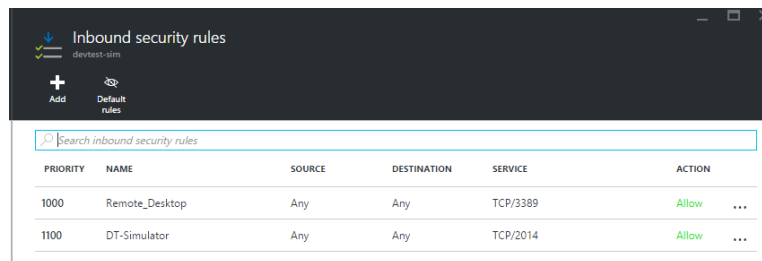


Figure 21 - Network Security Group for DevTest Simulator

Devtest-u3a-db

'devtest-u3a-db' hosts the DevTest database system created and managed by the registry. The DevTest workstation requires external access to the database.

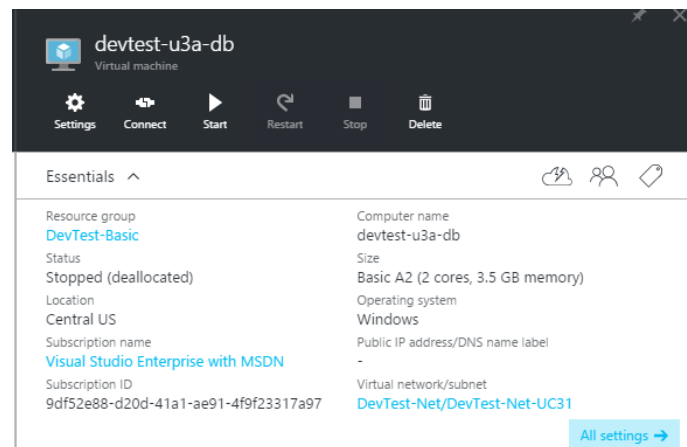
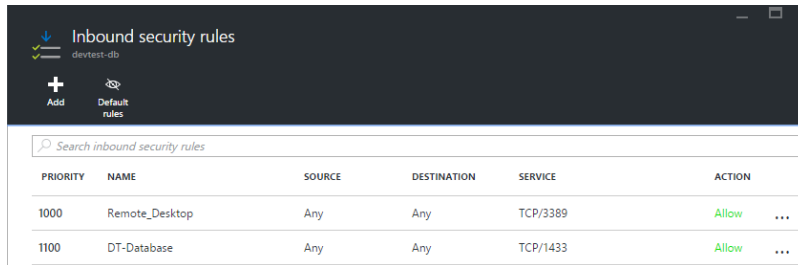


Figure 22 - VM devtest-u3a-db

Service Name	Protocol	Port	Profile	Required by
MS SQL Server	TCP	1433	External	DB for DevTest Registry

The default port to access MS SQL server is 1433 and is enabled by Network Security Group *devtest-db*.



PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
1000	Remote_Desktop	Any	Any	TCP/3389	Allow
1100	DT-Database	Any	Any	TCP/1433	Allow

Figure 23 - Network Security Group for DevTest Registry database server

DevTest-Net-UC32

This Microsoft® Azure subnet was created but no VMs provided to it. Microsoft® Azure Cloud Service ‘DevTest-Net-UC32’ hosts the DevTest 9.0 server environment for ‘client-2’. Settings of the cloud services and the VMs in this Cloud Service are equivalent to setting in subnet ‘DevTest-Net-UC31’.

VM Host name Public address for RDP

devtest-u3b-reg	devtest-u3b-reg.centralus.cloudapp.azure.com:3389
devtest-u3b-sim	devtest-u3b-sim.centralus.cloudapp.azure.com:3389
devtest-u3b-vse	devtest-u3b-vse.centralus.cloudapp.azure.com:3389
devtest-u3b-db	devtest-u3b-db.centralus.cloudapp.azure.com:3389

DevTest-Net-UC33

This Microsoft® Azure subnet was created but no VMs provided to it. Microsoft® Azure subnet ‘DevTest-Net-UC33’ hosts a DevTest 9.1 server environment for ‘client-3’. Settings of the cloud services and the VMs in this Cloud Service are equivalent to setting in subnet ‘DevTest-Net-UC31’.

DevTest configuration

DevTest 9.0 and 9.1 are installed with default settings and default ports.

Installing the distributed service components VSE and Simulator as Windows services on dedicated VMs, DevTest server components were configured not to start automatically on system start.

Before reconfiguring the Registry, VSE and Simulator services to start automatically, modify the local.properties file on the dedicated VM’s.

1. The VSE and Simulator service, respectively, need information where to find the Registry service to connect. In subnet ‘DevTest-Net-UC31’, for instance, we extend file **local.properties** by

`lisa.registry.url=tcp://devtest-u3a-reg:2010/Registry`

Note: We do not use the IP address to access the Registry service, but the Registry VM’s hostname. This saves us updating local.properties each time after restart. Microsoft® Azure takes care of name resolution.

2. Because DevTest portal displays VSEs and Simulators by name, these names need to be distinct. By default, all VSE services are called ‘VSE’, and all Simulator services go by ‘Simulator’. We use an identifying substring of the system’s hostname to identify the VSE or Simulator service, respectively, in the various UIs.

- On server *devtest-u3a-vse* we add the distinct VSE name to **local.properties**
`lisa.vseName=U3A-VSE`
- On server *devtest-u3a-sim* we add the distinct Simulator name to **local.properties**
`lisa.simulatorName=U3A-Sim`

Verification of DevTest Environment Setup

This section covers steps to verify that the Microsoft® Azure setup is working correctly for users at the customer's and the client team's side. This verification includes steps to verify that

- Enterprise Dashboard UI is available on customer network
- DevTest Server Console is available on client network for all DevTest Registries
- DevTest Portal is available on client network for all DevTest Registries and is able to
 - launch test cases
 - Monitor tests
 - Deploy virtual services
 - Display virtual services running on VSE server system
- Local Workstation on client network connects to the Registry service in Microsoft® Azure and to check if
 - Local test client (LISA Bank kiosk) connects to application in Microsoft® Azure
 - Local test client (LISA Bank kiosk) connects to virtual service in Microsoft® Azure
 - A local test case can be staged to DevTest Coordinator Service in Microsoft® Azure from Workstation
 - A virtual services can be deployed to VSE in in Microsoft® Azure from Workstation

These verification steps are executed on a system connected to client or customer network, respectively.

DevTest Server Name Resolution

The Microsoft® Azure DNS service publishes the FQDN of a VM, but does not publish the host name of the VM. Therefore, the FQDN of an Azure VM is resolved by Azure DNS, but VM host name is not.

An external DevTest user accesses DevTest services, which are deployed in a Microsoft® Azure subnet, by the FQDN name of the VM and the public service port number. Host names of the Microsoft® Azure VMs running the DevTest service are hidden and unknown to external DevTest users.

This is not an issue when accessing DevTest services directly via Web UI, such as the DevTest Portal, the DevTest Server Console or the DevTest Enterprise Dashboard, for instance, because the user addresses these DevTest services by the DNS name of Microsoft® Azure VM and the service ports.

Hiding of Microsoft® Azure VM host names, however, is a problem, if an external DevTest user uses DevTest workstation, and selects specific tasks that are executed on DevTest servers: staging a test case to a Coordinator or deploying a virtual service to a VSE, for instance. When executing such a task, the DevTest workstation queries DevTest Registry for the DevTest resource (i.e. host name and port) connected to the Registry and attempts to access the qualified system directly from the client network. The DevTest Registry, however, returns host name and port number only of the registered DevTest

service VM. Because Microsoft® Azure VM host name cannot be resolved on the client network, the intended operation will fail.

The following procedure avoids this failure: On the client network

- The client's DNS service must add a DNS Forwarder rule to the Microsoft® Azure DNS server for unresolved names, or,
- The Microsoft® Azure VM host name must be mapped to the public FQDN of the Microsoft® Azure Cloud Service. DNS supports name mapping by a CNAME record.

The regular way to map a network name to another one is to add a CNAME record to the DNS service. Because we usually do not have access to the DNS service, and as a temporary workaround, we add an entry to the local *etc/hosts* file, which maps the IP address of

- *devtest-u3a-reg.centralus.cloudapp.azure.com* to *devtest-u3a-reg* (hosting the Coordinator service),
- *devtest-u3a-sim.centralus.cloudapp.azure.com* to *devtest-u3a-sim* (hosting the Simulator Service), and
- *devtest-u3a-vse.C* to *devtest-u3a-vse*, running the VSE.

The IP addresses of VMs in subnets *DevTest-Net-UC32* and *DevTest-Net-UC33* are mapped accordingly to their host names.

```
52.165.36.94      devtest-uc0-eds
52.165.46.237    devtest-u3a-reg
13.67.190.199    devtest-u3a-db
13.89.45.219     devtest-u3a-vse
40.86.94.179     devtest-u3a-sim
13.89.40.101     devtest-u3b-reg
```

Note: This workaround will fail when the Microsoft® Azure VMs are recycled, because they will then get new public IP addresses. Then, *etc/hosts* file needs an update with the new IP addresses.

DevTest Web UI Access Connectivity


DevTest supports three Web User Interfaces at the following URLs and default ports:

- DevTest Enterprise Dashboard UI – <http://<EnterpriseDashboardServer>:1506>
- DevTest Portal - <http://<PortalServer>:1507/devtest>
- DevTest Server Console - <http://<RegistryServer>:1505>

Enterprise Dashboard Web UI Access from customer network

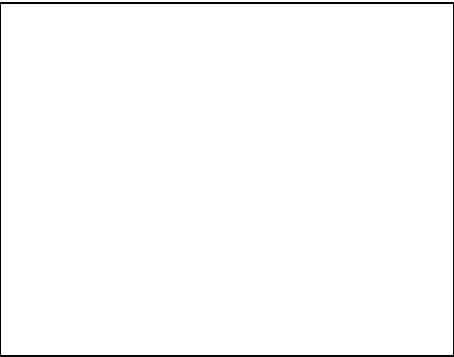
The user launches DevTest Enterprise Dashboard UI from

<http://devtest-uc0-eds.centralus.cloudapp.azure.com:1506>

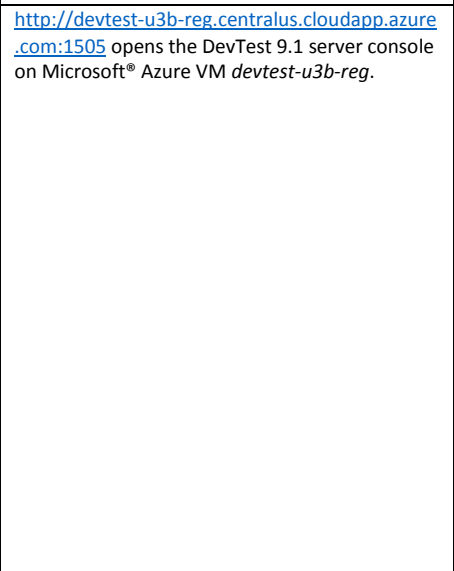


Registry Name	Name	Last Update	Version	Coordinators	Simulators	VSEs	Workstations	Agents	Labels	DevTest Console URL
reg-uc0-eds	reg-uc0-eds	7/4/2019 12:00:00 PM	3.1.0 (3.1.0.201)	1	1	1	0	0	1	http://devtest-uc0-eds:1506/index.html
reg-uc0-sim	reg-uc0-sim	7/4/2019 12:00:00 PM	3.1.0 (3.1.0.201)	1	1	1	0	0	1	http://devtest-uc0-sim:1506/index.html

Enterprise Dashboard shows both the registry servers devtest-u3a-reg and devtest-u3b-reg.

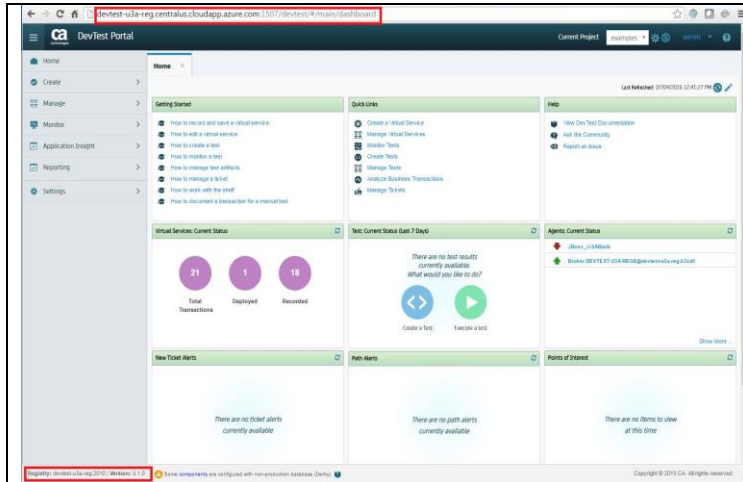


centralus.cloudapp.azure.com:1505
centralus.cloudapp.azure.com:1505 for 'Azure-2'.

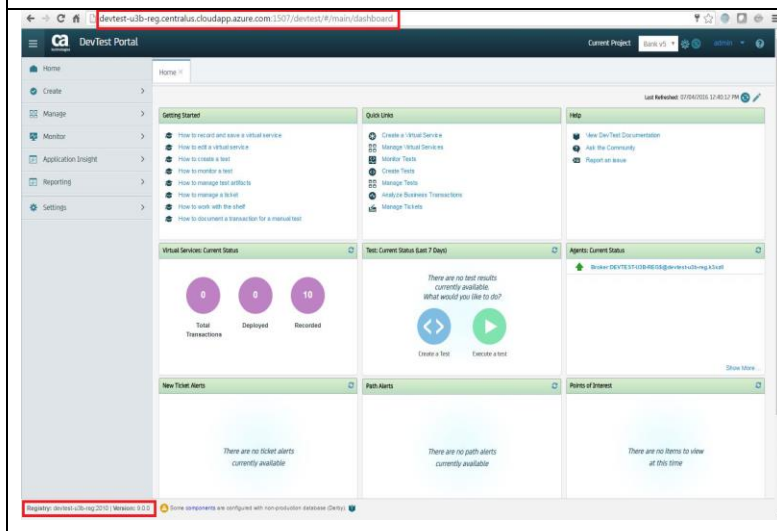


DevTest Portal Access from client network

DevTest Portal is the Web UI frontend to access DevTest Server components. The DevTest Portal console is launched on <http://devtest-u3a-reg.centralus.cloudapp.azure.com:1507/devtest> for 'Azure-1' and on <http://devtest-u3b-reg.centralus.cloudapp.azure.com:1507/devtest> for 'Azure-2'.



<http://devtest-u3a-reg.centralus.cloudapp.azure.com:1507/devtest> opens the DevTest 9.1 portal on Microsoft® Azure VM *devtest-u3a-reg*.



<http://devtest-u3b-reg.centralus.cloudapp.azure.com:1507/devtest> opens the DevTest 9.0 portal on Microsoft® Azure VM *devtest-u3b-reg*.

DevTest Portal Connectivity Tests from client network

DevTest Portal is the Web UI frontend to access DevTest Server components and to take actions on DevTest services, resources and assets.

- Connectivity to required DevTest Server components such as Simulator, Coordinator and VSEs
- Staging a test to validate access to Coordinator and Simulator
- Deployment of a virtual service to verify access to VSE

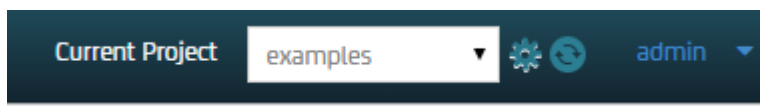
For the sake of simplicity, we describe the verification tests executed on Microsoft® Azure subnet *DevTest-Net-UC31* only.

The Portal of 'devtest-u3a-reg' displays the VSE 'UC3A-VSE' that was installed on the dedicated Microsoft® Azure VM 'devtest-u3a-vse' (see above).

Running Tests from DevTest Portal

In the DevTest Portal, select the 'examples' folder.

Note: In order to view the 'examples' folder in the DevTest Portal, you would need to copy that folder (<DevTest home>\examples) to the Projects folder <DevTest home>\projects.



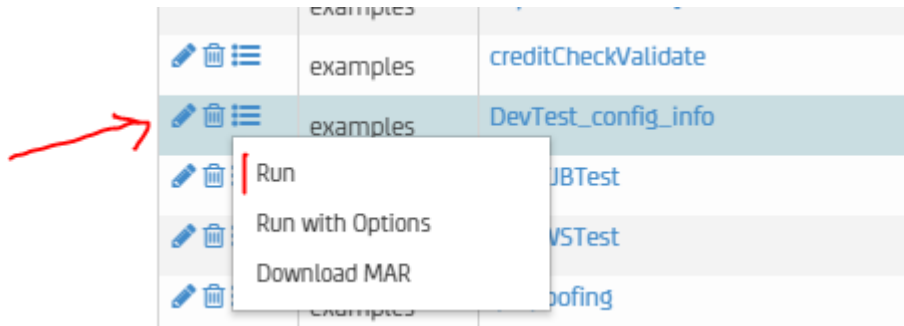
In the left menu – select Manage → Tests

Manage

All Resources
API Tests
Tests
Test Suites
Virtual Services
Copybook Bundles
Data Sets
Monitor
Application Insight
Reporting
Settings
Agents

Actions	Project	Name
	examples	
	examples	AccountControlIMDB
	examples	async-consumer-jms
	examples	creditCheckValidate
	examples	DevTest_config_info
	examples	ejb3EJBTest
	examples	ejb3WSTest
	examples	ip-spoofing
	examples	JMS
	examples	load data from a csv file
	examples	log-watcher
	examples	main_all_should_fail
	examples	main_all_should_pass
	examples	multi-tier-combo
	examples	multi-tier-combo-selenium
	examples	rawSoap

You will now locate and run the *DevTest_config_info* test case. To simplify the search sort the name 'ascending'.



As soon as you click Run, a new tab called 'Monitoring Tests' is opened. The test should run to completion.

Summary (Last 7 Days) Last Refreshed: 07/04/2016 1:01:08 PM

2	0	0	0	0	0
Passed	Failed	Aborted	Errors	Warnings	Running

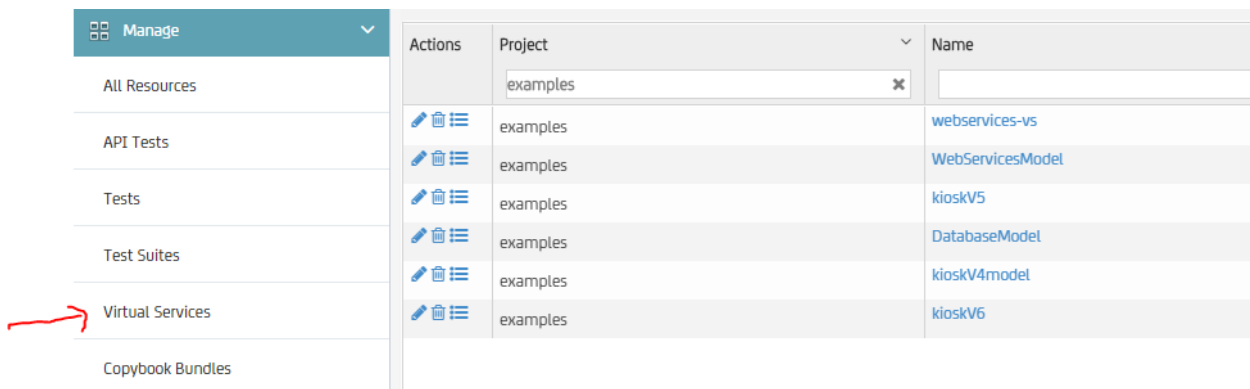
Suites/Tests Filter by 'Name' | Filter by 'Executed By' | ☒ Tests ☒ Suites | ☒ Passed ☒ Failed ☒ Aborted ☒ Running

Name	Status	Messages	Start Time	Duration	Executed By	Description
DevTest_config_info	✓		07/04/2016 1:00:28 PM	10.1s	admin	This test case fetches diag...

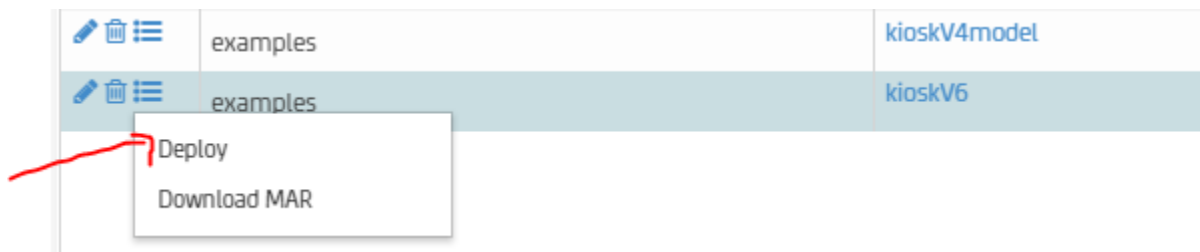
You can now click on the completed test case name to drill down into the details of the execution steps.

Deploy Virtual Service to VSE from the DevTest Portal

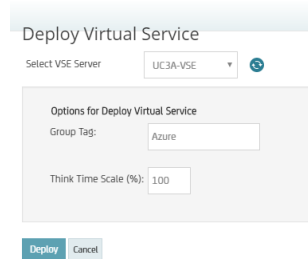
In the left menu, click on Manage → Virtual Services to open a list of available virtual services in the examples folder.



Deploy the virtual service called *kioskV6* by clicking on the Options action and selecting Deploy



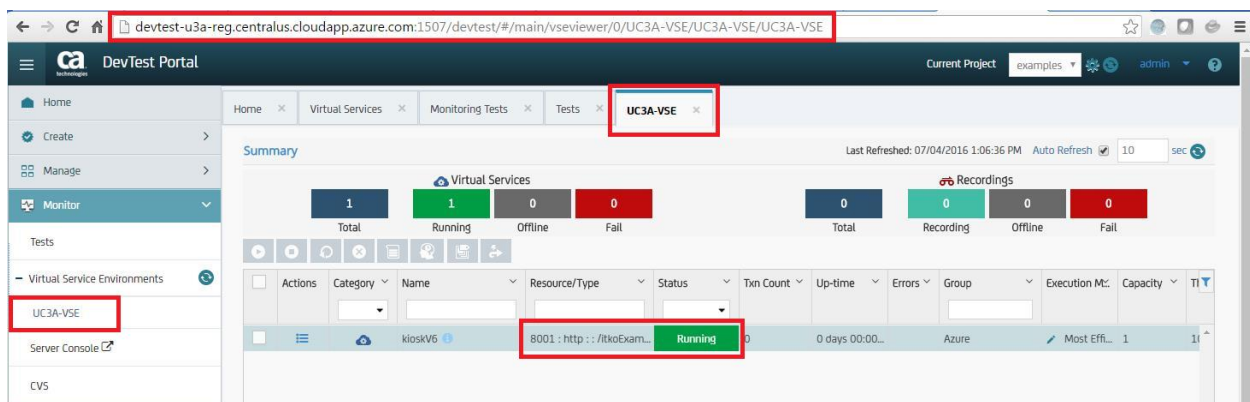
Pick VSE server *UC3A-VSE* and provide an optional Group tag. You should see a message about successful deployment. We pick *UC3A-VSE*, because we added a public port to the cloud service to access this VSE on port 8001.



The image shows a 'Deploy Virtual Service' dialog box. It has a 'Select VSE Server' dropdown menu with 'UC3A-VSE' selected. Below it, there are 'Options for Deploy Virtual Service' including a 'Group Tag' field with 'Azure' and a 'Think Time Scale (%)' field with '100'. At the bottom are 'Deploy' and 'Cancel' buttons.

Click on Deploy to submit the virtual service to the selected VSE.

Now click on *Monitor* → *Virtual Service Environment* → *UC3A-VSE*. This will open a new tab called *UC3A-VSE* and you should see the *kioskV6* VS deployed and running successfully on port 8001.



The image is a screenshot of the DevTest Portal. The browser address bar shows a URL ending in 'UC3A-VSE'. The portal has a sidebar with 'Monitor' selected, and 'UC3A-VSE' is highlighted under 'Virtual Service Environments'. The main area shows a 'Summary' for 'Virtual Services' with a table of status counts: Total (1), Running (1), Offline (0), Fail (0). Below this is a table of virtual services. The first row shows 'kioskV6' with a status of 'Running' and a URL '8001 : http : //tkoExam...'. The 'Running' status and the URL are highlighted with red boxes.

Actions	Category	Name	Resource/Type	Status	Txn Count	Up-time	Errors	Group	Execution M:	Capacity
		kioskV6		Running	0	0 days 00:00...		Azure	Most Eff...	11

DevTest Workstation Connectivity Tests from client network

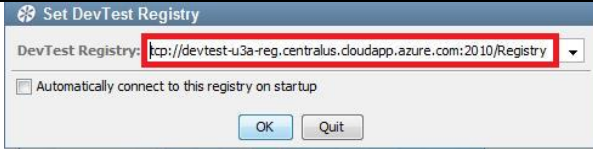
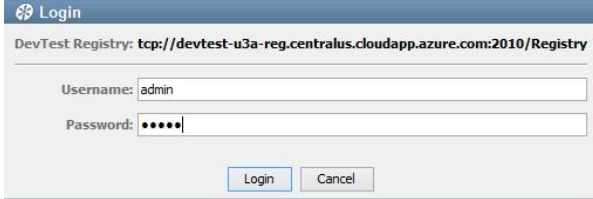
To verify that a client's workstation connects to and works correctly with the DevTest server installation in Microsoft® Azure we run following tests:

1. Connect DevTest Workstation on local network to DevTest Registry service in Microsoft® Azure
2. Connect a local Web client test application to a Web service in Microsoft® Azure
3. Connect a local Web client test application to a virtual Web service on DevTest VSE server in Microsoft® Azure
4. Stage a test case from local Workstation to a DevTest Coordinator and DevTest Simulator running in Microsoft® Azure
5. Deploy a virtual service from local DevTest Workstation to a DevTest VSE server in Microsoft® Azure

For the sake of simplicity, we describe the verification tests executed with Microsoft® Azure subnet *Azure-1* only.

Connect local Workstation to DevTest Registry in cloud



First we verify that the local workstation on the client network is able to access the DevTest Registry service in the Microsoft® Azure environment.

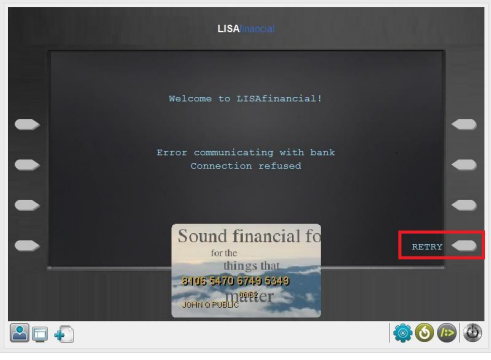
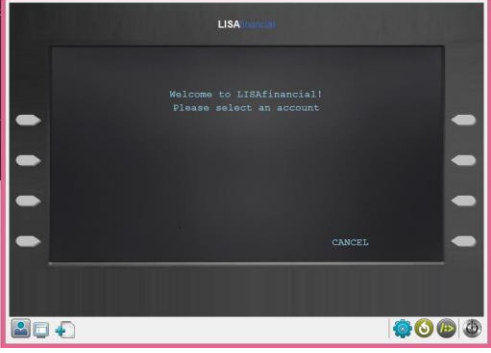
 	<p>Connect local workstation on company network to registry on Microsoft® Azure VM <i>devtest-u3a-reg.centralus.cloudapp.azure.com:2010</i>.</p>
--	--

Connect local client to Application in cloud

Now we want to verify that a local test client can connect to an application in the Microsoft® Azure environment. As an example, we connect the local kiosk application to LISA Bank application running on the DevTest Registry server in Microsoft® Azure.

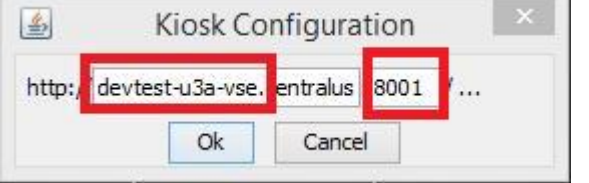
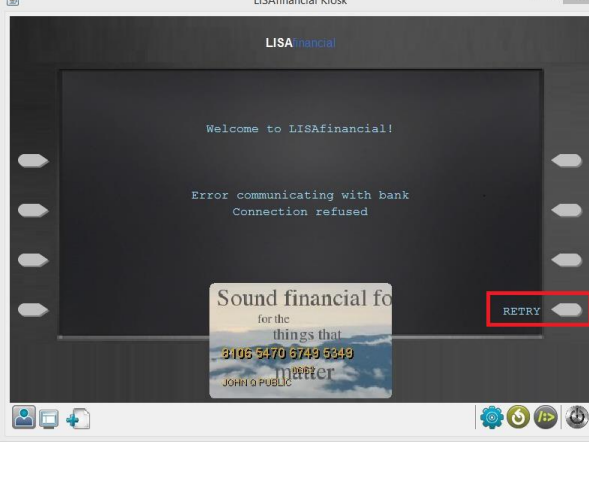
Start LISA Bank demo on *devtest-u3a-reg.centralus.cloudapp.azure.com*. We can access the live LISA Bank service on *devtest-u3a-reg.centralus.cloudapp.azure.com:8080*.

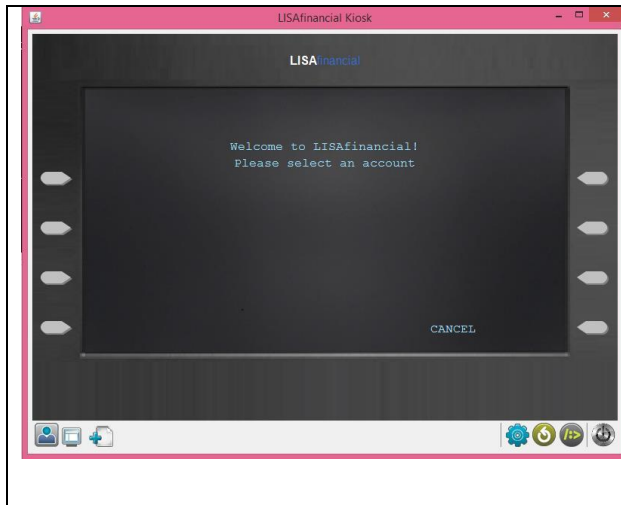
	<p>Start Kiosk from local Workstation (on Company network).</p> <p>Note the failed connection to the bank.</p>
	<p>Edit the URL to access LISA bank to <i>devtest-u3a-reg.centralus.cloudapp.azure.com:8080</i></p>

	<p>Click on the Retry button to apply the changed URL ...</p>
	<p>... and the kiosk on local workstation is now connected to the LISA Bank application on <i>devtest-u3a-reg</i> in Microsoft® Azure.</p> <p>Click on the credit card to select a user, try Lisa Simpson with password golisa, which will log you in.</p> <p>The screenshot on the left shows a successful log in of Lisa Simpson.</p>

Connect local client to virtual service in cloud

Now we want to test whether or not the kiosk client can connect to virtual service *kioskv6* that we deployed in a previous step. This virtual service listens on port *devtest-u3a-vse.centralus.cloudapp.azure.com:8001*.

	<p>Edit the URL to access LISA bank to <i>devtest-u3a-vse.centralus.cloudapp.azure.com:8001</i></p>
	<p>Click on the Retry button to apply the changed URL ...</p>



... and the kiosk on local workstation is now connected to the virtual LISA Bank application on *devtest-u3a-vse.centralus.cloudapp.azure.com:8001* in Microsoft® Azure.

Click on the credit card to select a user, try Lisa Simpson with password golisa, which will log you in.

The screenshot on the left shows a successful log in of Lisa Simpson to the virtual service kioskv6.

The following screenshot of the DevTest Portal proves that the kiosk queried the Kiosk Virtual Service

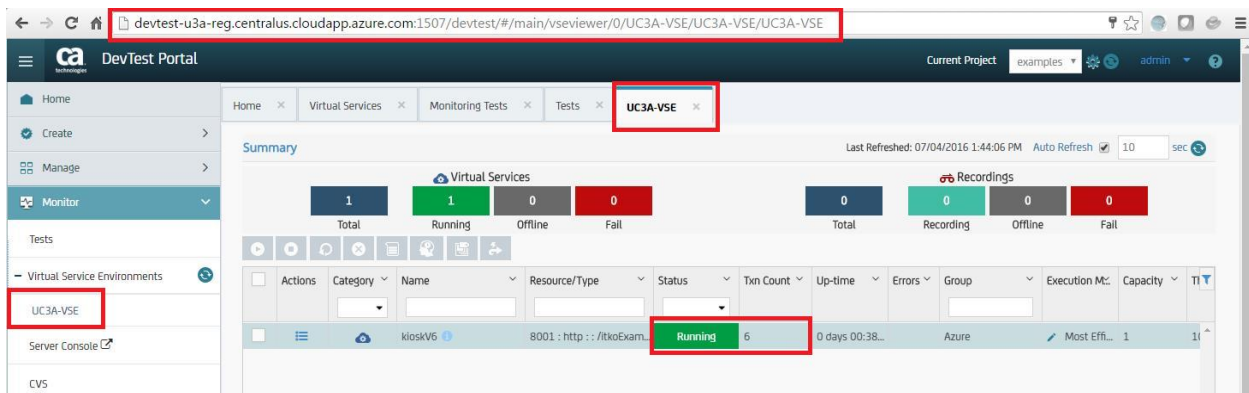
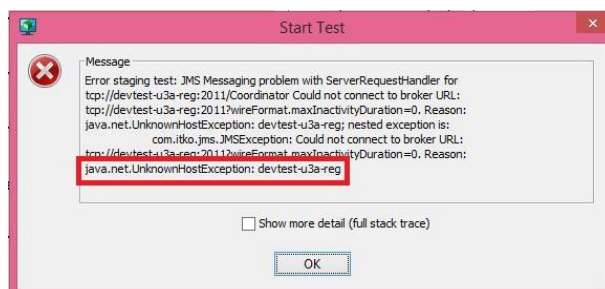


Figure 24 - DevTest Portal VSE Transaction Counter

Stage test from local DevTest workstation to cloud

Staging a test case from local workstation to DevTest in Microsoft® Azure requires a name mapping from *devtest-u3a-reg* and *devtest-uc3-sim* to *devtest-u3a-reg.centralus.cloudapp.azure.com*. See [DevTest Server Name Resolution](#).

DevTest workstation retrieves the Coordinator address from Registry service. The Registry service, however, knows the hostname and port number of the Coordinator service. DNS, however, does not resolve the hostname. Unless we configure a name mapping from *devtest-u3a-reg* to *devtest-u3a-reg.centralus.cloudapp.azure.com*, we will not be able to access the Coordinator service running on *devtest-u3a-reg*.



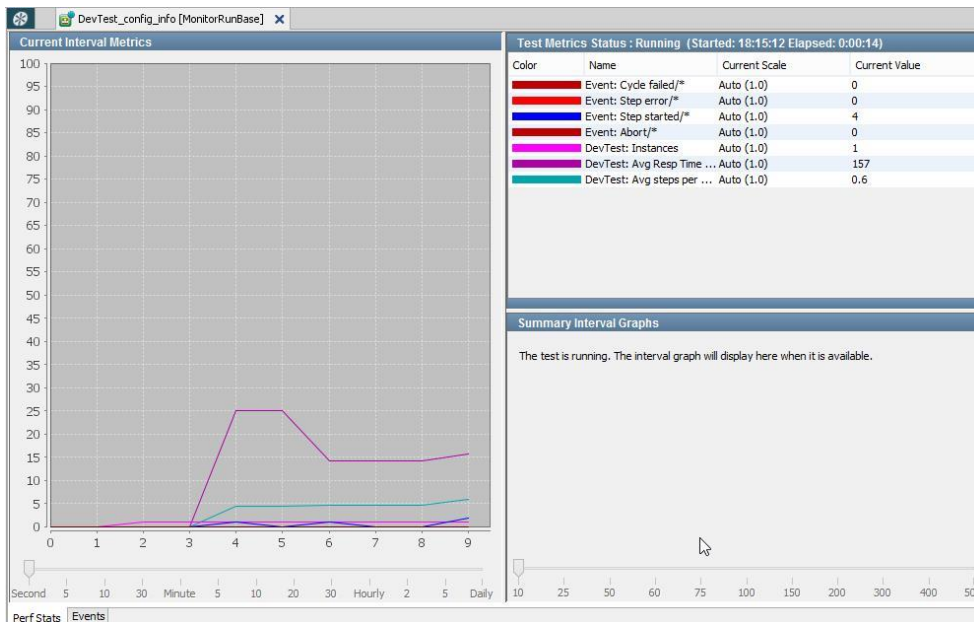
The regular way to map one name to another one is to add a CNAME record to the DNS service. Because we do not have access to the DNS service and as a temporary workaround, we add an entry to the local hosts file, which maps the IP address of *devtest-u3a-reg.centralus.cloudapp.azure.com* to *devtest-u3a-reg*.

```
52.165.36.94    devtest-uc0-eds
52.165.46.237   devtest-u3a-reg
13.67.190.199   devtest-u3a-db
13.89.45.219    devtest-u3a-vse
40.86.94.179    devtest-u3a-sim
```

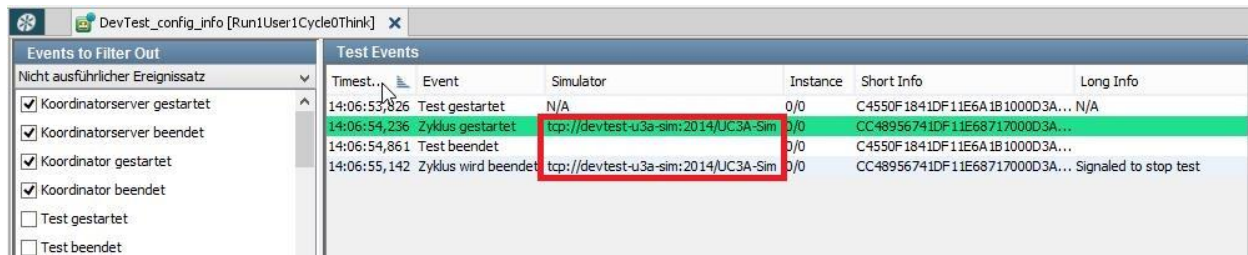
The same also applies to the DevTest Simulator Service, which the Workstation accesses to retrieve the list of test events.

This workaround – redefining the IP address of the DevTest Coordinator and the DevTest Simulator Service – will fail when the Microsoft® Azure VM is recycled, because it will then get a new public IP address assigned.

With this workaround in place, we can now stage a test from local workstation to the Coordinator in Microsoft® Azure, which the DevTest Simulator in Azure executes. For the screenshots below, we staged the same *DevTest_config_info* test that we used to test the portal using Staging Doc *Run1User1Cycle0Think*.



Watching the test events prove that the test was executed on a Simulator instance on the DevTest Simulator server VM the Microsoft® Azure environment that we installed and configured.



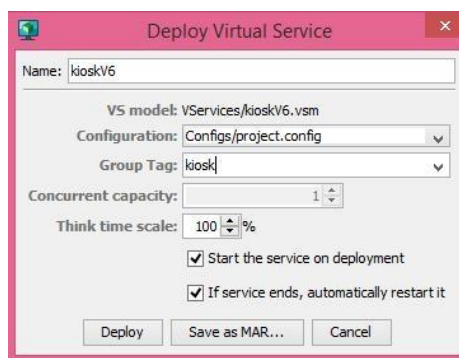
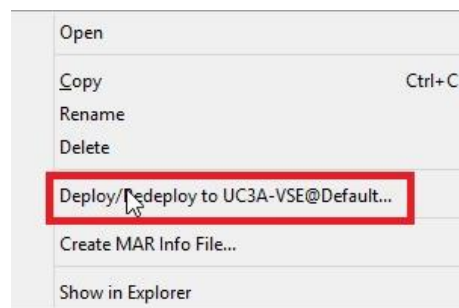
Timest...	Event	Simulator	Instance	Short Info	Long Info
14:06:53,826	Test gestartet	N/A	0/0	C4550F1841DF11E6A1B1000D3A...	N/A
14:06:54,236	Zyklus gestartet	http://devtest-u3a-sim:2014/UC3A-Sim	0/0	CC48956741DF11E68717000D3A...	
14:06:54,861	Test beendet		0/0	C4550F1841DF11E6A1B1000D3A...	
14:06:55,142	Zyklus wird beendet	http://devtest-u3a-sim:2014/UC3A-Sim	0/0	CC48956741DF11E68717000D3A...	Signaled to stop test

Deploy Virtual Service from local DevTest workstation to cloud

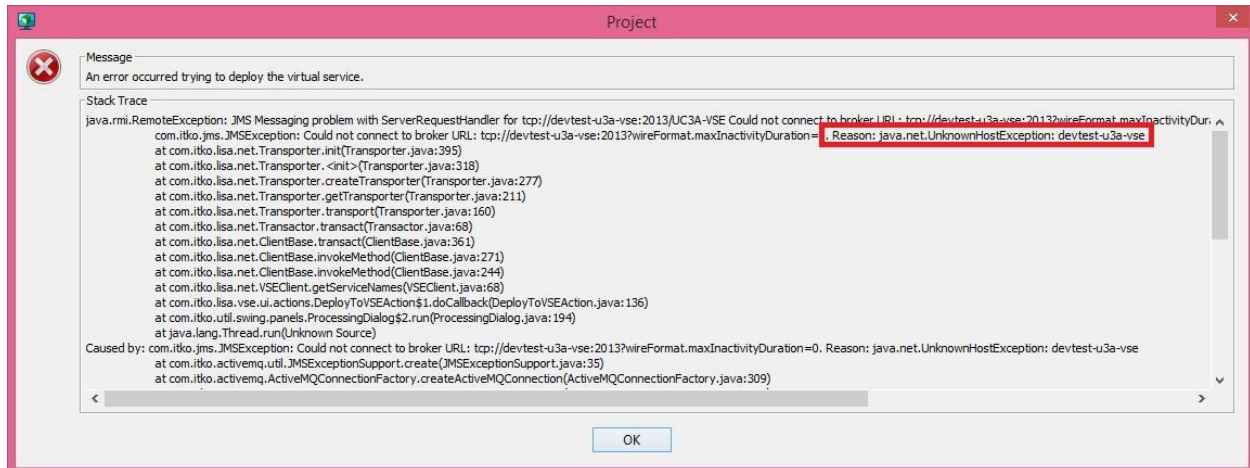
To test the deployment of a virtual service from a local workstation connected to the client's network to the DevTest VSE server in Microsoft® Azure, we deploy the same sample *kioskv6* example that we used testing the DevTest Portal.

Note: Make sure to remove any *kioskv6* virtual service running on port 8001 from VSE first.

We select *kioskv6.vsm* from the *VServices* folder in DevTest Workstation and select *Deploy/Redeploy to U3A-VSE@Default...* in order to deploy the *Kioskv6* Virtual Service to the VSE server VM that we created in Microsoft® Azure. DevTest Workstation has retrieved, again, the available VSE servers from DevTest Registry on *devtest-u3a-reg*, and got a list a VM host names and port numbers in return.



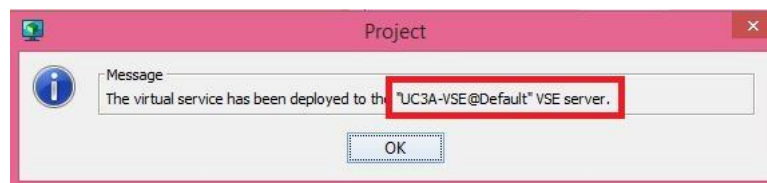
DevTest Workstation now attempts to deploy the virtual service directly to the VM host and port number, but fails, because the Microsoft® Azure VM host name is unknown on the client network.



Applying our workaround procedure, we add another entry to the local *etc/hosts* file of the client workstation for *devtest-u3a-vse* and give it the same IP address of the FQDN of Microsoft® Azure VM *devtest-u3a-vse.centralus.cloudapp.azure.com*.

```
52.165.36.94    devtest-uc0-eds
52.165.46.237   devtest-u3a-reg
13.67.190.199   devtest-u3a-db
13.89.45.219    devtest-u3a-vse
40.86.94.179    devtest-u3a-sim
13.89.40.101    devtest-u3b-reg
```

With this new setting workstation is now able to deploy the virtual service to the VSE server, and we get a success message.



Finally, we can verify if the virtual service is actually running on the VSE server in Microsoft® Azure as expected.

The screenshot displays the DevTest Portal interface. The left sidebar shows the navigation menu with 'Monitor' selected. The main content area shows the 'UC3A-VSE' virtual service environment. The status is 'Running'.

Summary

Last Refreshed: 07/04/2016 2:27:36 PM Auto Refresh 10 sec

Virtual Services				Recordings			
Total	Running	Offline	Fail	Total	Recording	Offline	Fail
1	1	0	0	0	0	0	0

Actions	Category	Name	Resource/Type	Status	Txn Count	Up-time	Errors	Group	Execution M...	Capacity	T1
		kioskV6	8001 : http :: /tkoExam...	Running	0	0 days 00:01...		kiosk	Most Eff...	1	11

Appendix

VM customization

OS Configuration

- Added inbound firewall rules to allow incoming traffic for DevTest services.
- Disable IE Enhanced Security for both administrators and users
- Add cmd.exe to taskbar, Start menu and Desktop
- Add permanent share to [\\devtest-u3a-reg\c\\$](#) for sharing installers.

Installed Software

- Chrome Browser
- Notepad++
- DevTest Server
 - Pointing to Enterprise Dashboard service on *devtest-uc0-eds.centralus.cloudapp.azure.com:1506*, or *devtest-uc0-eds.centralus.cloudapp.azure.com:2003*, respectively.
 - Installing DevTest services as Windows services without automatic start at start up
 - Demo servers installed

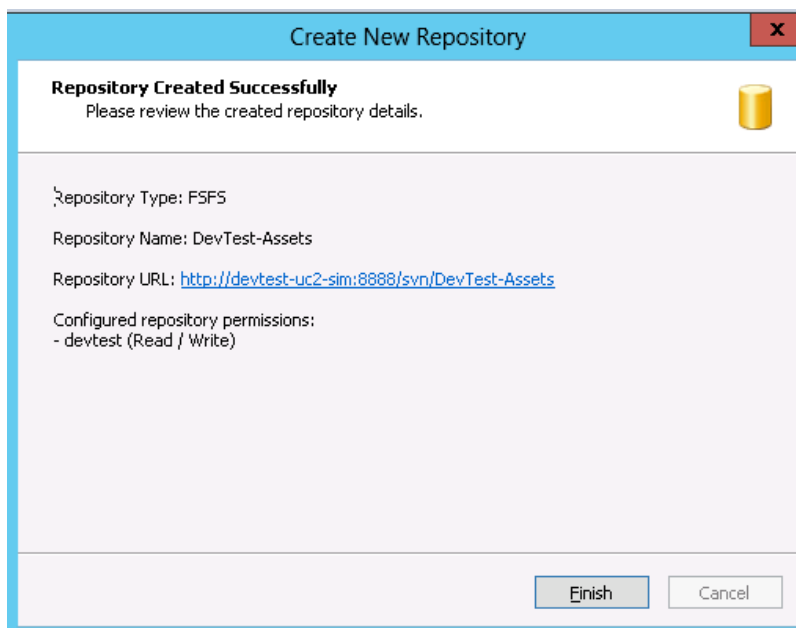
Service configuration

- Updated local.properties, site.properties and dradis.properties according to document
- Reconfigured Windows services VSE and Simulator for automatic start at system start
- Started VSE and Simulator Windows services

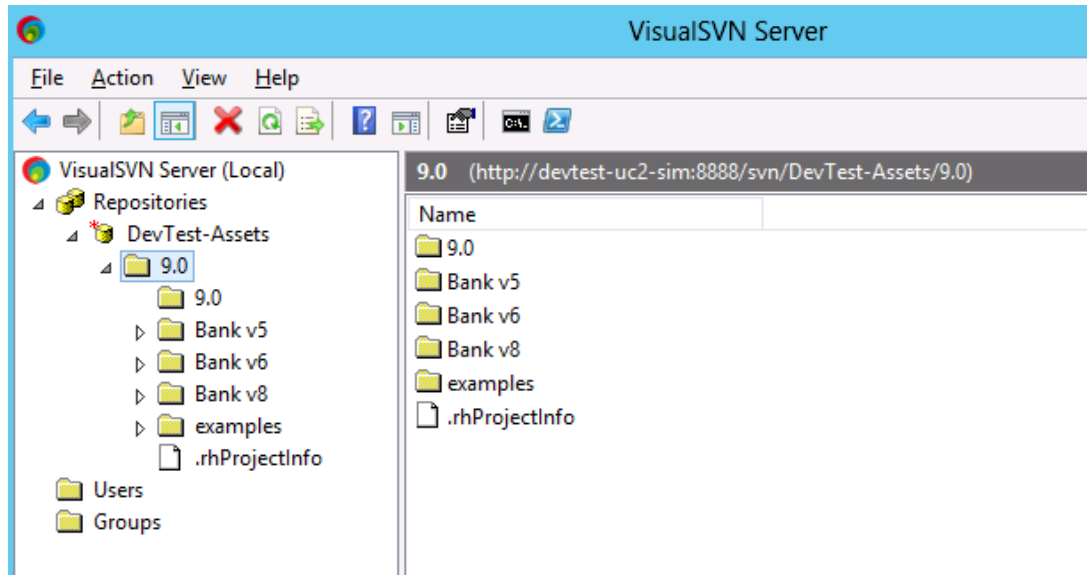
SCM Details

This section shows how SCM is used –

- A new repository was created in Visual SVN Server called DevTest-Assets

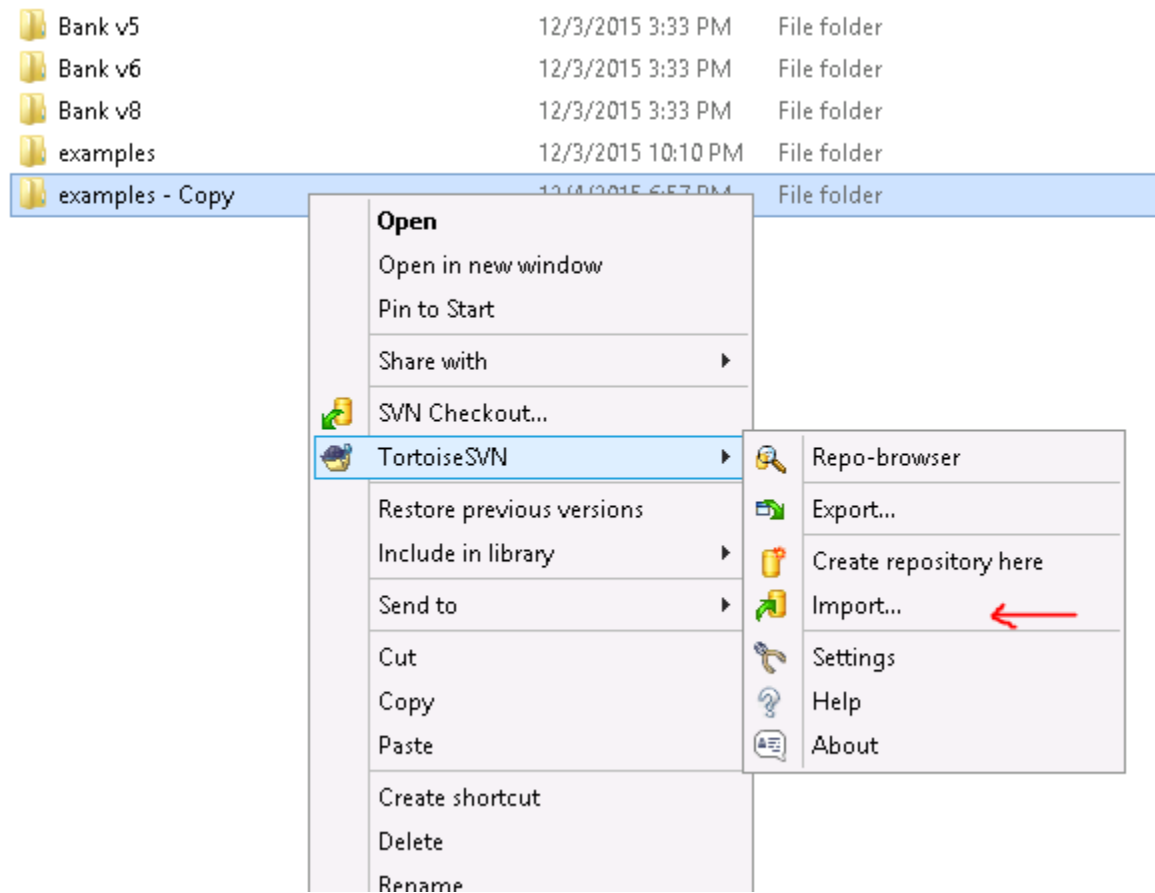


- From the location of the DevTest “Projects” folder, the Tortoise SVN client was used to import all the Project assets into SVN’s DevTest-Assets repository

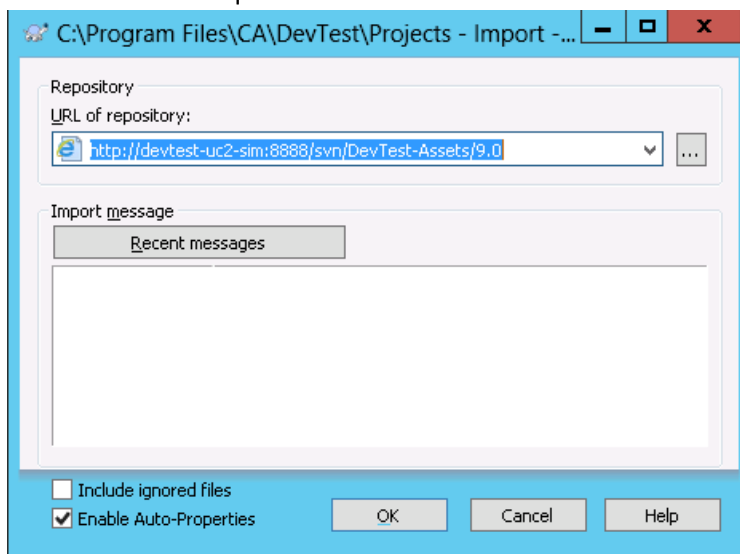


- At this point, the assumption is that the DevTest users have a mapped drive to the DevTest Projects folder. Once they create/update a new DevTest asset, they can simply commit those changes to the SVN using the Tortoise client.
- In this example, this end user would like to back up the 'examples' folder as 'examples - copy'. After copy/paste, a new folder called 'examples – copy' is created.

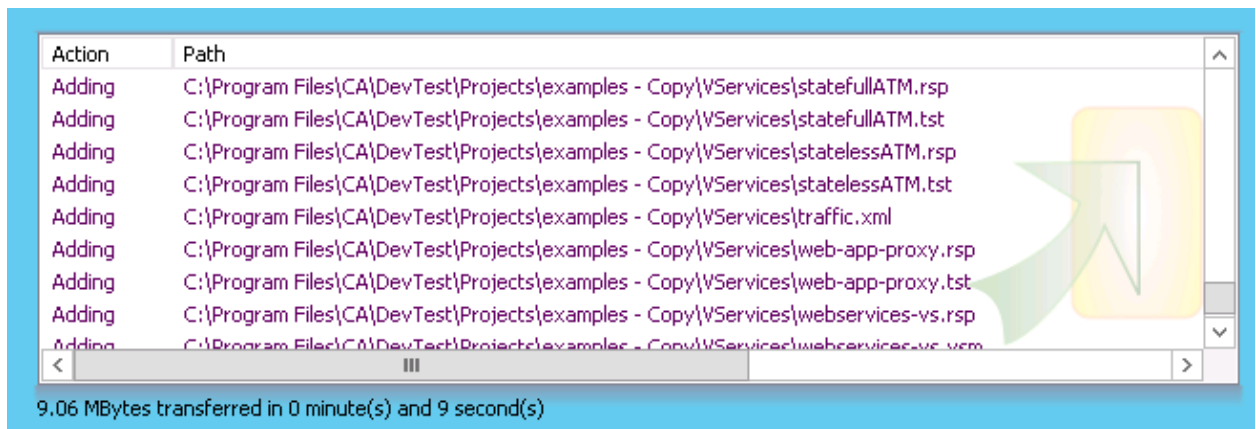
- Then the user right clicks and proceeds to check in the new folder to SVN as shown below



Authenticate as required –



Completed check in to SVN server shown



A new folder is created in the SVN server for 'examples – copy'.

In this way, the same technique can be applied to any DevTest assets that are created/modified.

If you were using the DevTest Workstation and the assets were stored locally then you would check in and check out assets from the SVN server without the need to map to the centrally located DevTest Projects folder.