#01

súbete a la nube de Microsoft

Introducción a Windows Azure

Ibón Landa Martín
Unai Zorrilla Castro

campus MVP .com

Plain Concepts
The Microsoft Technologies Company

Súbete a la nube de Microsoft

Parte 1: Introducción a Windows Azure

Ibón Landa Martín
Unai Zorrilla Castro

SÚBETE A LA NUBE DE MICROSOFT PARTE I: INTRODUCCIÓN A WINDOWS AZURE

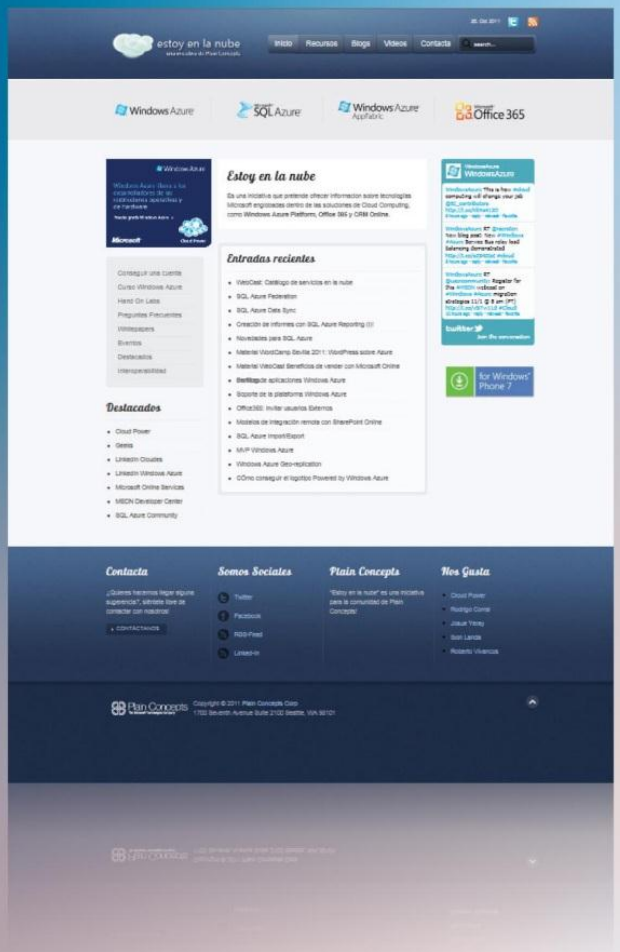
Noviembre de 2011

Contenido

estoy en la nube

INICIATIVA DE PLAIN CONCEPTS

www.estoyenlanube.com



Windows Azure

www.plainconcepts.com

Plain Concepts is a company specialized in Microsoft technologies, agile methodologies, Application Lifecycle Management, performance tuning, advanced debugging, software architecture and User Experience.

Plain Concepts focuses on delivering high quality consulting, mentoring and training as well as in being an effective and reliable team resolving all type of software development issues.

Plain Concepts

¿Aún quieres más?

campus MVP

Formación online especializada en tecnologías Microsoft.

krasis PRESS

Los libros que lo saben todo sobre tecnologías Microsoft.

Síguenos y descubrirás los mejores trucos y recursos:

facebook.com/campusmvp twitter.com/campusmvp

feed your brain®

- Sin tener que desplazarse
- Sin romper el ritmo de trabajo
- Preguntándole a los que más saben

infórmate ya:
902 876 475
www.campusmvp.com

http://www.krasis.com

krasis

Microsoft Partner
Silver Learning
Silver Software Development

Introducción

Bienvenido amigo lector a esta obra, la cual esperamos sea de su agrado y que le sirva profesionalmente para aprender o profundizar sus conocimientos sobre **cloud computing** en general y sobre **Windows Azure** en particular.

Los autores hemos ido recopilando, a lo largo de bastante tiempo, la información más valiosa sobre la plataforma y actualizándola a medida que desde el equipo de producto de Microsoft liberaban nuevas versiones del SDK (Software Develpment Kit) y de la propia plataforma. El objetivo es presentarte de la mejor forma posible toda la información necesaria para que aprendas o mejores sus conocimientos sobre Windows Azure.

El contenido de este libro digital se divide en 4 partes fundamentales:

- **La primera parte** es básicamente una introducción a los conceptos **de cloud computing** y **Windows Azure**. En esta introducción haremos un repaso a la arquitectura de Windows Azure, destacando los diferentes conceptos que la forman y las ventajas de cada uno de ellos. Lógicamente también veremos cómo empezar a trabajar con la plataforma, desde el proceso de alta de una cuenta hasta la configuración de los distintos elementos de trabajo.

- La **segunda parte** de esta obra desgranará una de las partes fundamentales de la plataforma: el almacenamiento. En concreto se verá con cierta profundidad **Sql Azure**, para aprender a manejarnos con nuestras bases de datos en la nube, los elementos soportados y no soportados y sobre todo, cómo trabajar de una forma correcta con nuestras bases de datos de Sql Azure. Además veremos el almacenamiento no relacional, Windows Azure Storage, desde la configuración hasta el trabajo y las posibilidades de escalado que nos ofrece.

- **Windows Azure App Fabric** y la parte dedicada a **Windows Identity Foundation** conforman la **cuarta entrega** de estos contenidos, en la que podremos ver desde el uso de bus de servicios de Azure hasta la cache de Windows Azure.

- **Para terminar** se presentarán una serie de apéndices con consideraciones de diseño de aplicaciones en la nube así como los principales manuales de las herramientas de trabajo habitual en la plataforma.

Esperamos que todo el trabajo y cariño que hemos puesto los autores y la editorial (Krasis Press) en hacer accesible esta obra de manera gratuita, hayan servido para que aprendas y mejores en tu carrera profesional.

CAPÍTULO

1

Windows Azure Platform

1.-LOS RETOS DE LA ESCALABILIDAD Y DISPONIBILIDAD DE APLICACIONES

Uno de los mayores problemas a los que se enfrentan muchos proyectos de software en Internet es asegurar su capacidad para escalar. Asegurar que el sistema puede asumir cargas crecientes o incluso manejar picos de carga con naturalidad no es un problema menor.

Estas necesidades de escalabilidad pueden aparecer paulatinamente en el tiempo, de forma que pueda haber tiempo para digerir esta nueva situación; o es posible que de manera eventual aparezcan cargas enormes, en respuesta a algo que ocurre puntualmente o incluso a menudo sin previo aviso.

Existen muchas situaciones de este estilo: de repente la web recibe enlaces de otra web muy popular atrayendo mucho tráfico, o mejora drásticamente su posición en los buscadores y las visitas se multiplican. O la pasarela de SMS, fruto de un jugoso contrato tiene que doblar su capacidad de la noche a la mañana. Provisionar de manera ágil el hardware necesario para hacer frente a este tipo de situaciones no es sencillo.

Otra situación que ocurre a menudo es que los sistemas soportan picos temporales de carga durante determinadas épocas del año o ante eventos concretos. Por ejemplo, una web especializada en la venta de cava, en Navidad multiplica por cien las visitas. O pensemos en el sistema de matriculación de una Universidad que, típicamente, sólo trabaja a pleno rendimiento en el mes de Septiembre. Sin embargo debe estar dimensionado para esos picos de demanda, lo que tiene como consecuencia que durante once meses al año nuestra carísima infraestructura está ociosa y desaprovechada.

Un buen ejemplo de esta estacionalidad, concreto y real, se evidencia claramente al observar el gráfico de carga de la página web del campeonato de Wimbledon. La solicitación sobre sus servidores es casi nula todo al año, pero alcanza picos elevadísimos durante la celebración del torneo:

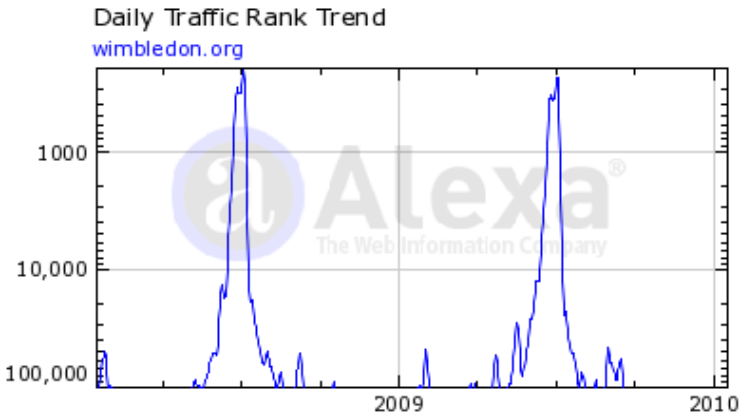


Figura 1.1.- Carga de la página web del campeonato de Wimbledon

Evidentemente resultaría inaceptable que una empresa no pudiese vender en la época más rentable del año, o que la Universidad no pudiese matricular a sus alumnos en Septiembre. También resulta inaceptable para los usuarios que las aplicaciones no estén disponibles cuando las necesitan. Así aparece un nuevo factor en la ecuación: la alta disponibilidad.

Los típicos patrones de carga ante las que el concepto "Nube" puede ayudar son los siguientes:

- **Aplicaciones con "picos" predecibles:** como el ejemplo anterior de Wimbledon, en el que se sabe de antemano que la demanda va a multiplicarse enormemente en determinadas épocas o ante eventos concretos. Con una implementación tradicional se desperdiciaría capacidad y generaría gran complejidad para el departamento de TI. En la nube se pueden contratar los recursos necesarios exactamente el tiempo que se necesiten.
- **Aplicaciones con "picos" impredecibles:** no es posible determinar cuándo van a producirse ni de qué dimensión serán, por lo que dimensionar la infraestructura apropiada no es posible con el esquema tradicional. Además estos picos impactan en el rendimiento y por tanto en el negocio, que puede verse incluso interrumpido. Los servicios Cloud pueden escalarse de inmediato ante una demanda inesperada.
- **Aplicaciones de crecimiento rápido:** aquellas que crecen mucho en demanda en virtud de un gran éxito entre los usuarios. Escalar y crecer es un gran reto tanto de desarrollo como para el equipo de TI de las empresas. Por ejemplo, la verdadera dificultad de una aplicación como Twitter, que es funcionalmente muy sencilla, es el poder crecer y llegar a gestionar millones de usuarios simultáneos. Albergada en un sistema en la nube puede aumentar su capacidad ilimitadamente a medida que el número de usuarios crece.
- **Aplicaciones On-Off:** aplicaciones que trabajan y se paran de manera predecible, alternando periodos de inactividad con periodos de trabajo intenso. En el esquema tradicional se desaprovechan mucho las capacidades del sistema, que estarán sobredimensionadas. Con el esquema de la nube se pueden gestionar cambios para aumentar y reducir las capacidades según cada fase, disminuyendo los costes a cero en los periodos sin demanda.

En definitiva, son innumerables las situaciones en las que es vital contar con capacidades de proceso, almacenamiento y ancho de banda que se puedan utilizar según las necesidades de cada momento y que sean, a efectos prácticos, ilimitadas.

1.1.- Infraestructura, configuración y mantenimiento

Desde el punto de vista de los desarrolladores, el mantenimiento de los "elementos base" que facilitan el despliegue de sus aplicaciones es algo de lo que no deberían tener que preocuparse. Así, la única preocupación de un desarrollador debería tener es la crear las aplicaciones, y que por tanto, el despliegue fuera igual de fácil tanto para un proyecto pequeño como para uno masivo.

Sin embargo existen cuestiones típicas de infraestructura de las que deben estar pendientes las empresas que mantienen sus propios servidores y centros de datos, como por ejemplo:

- Gestión del Sistema Operativo sobre el que se ejecutan las aplicaciones y servicios.
- Configuración de servicios "base": S.O., servidor de datos, Servidor de aplicaciones...
- Gestión de actualizaciones y parches de los servicios "base"
- Diagnóstico de fallos de servicios "base"
- Respuesta a fallos de hardware
- Disponibilidad de capacidad de almacenamiento adecuada, lo que puede llegar a ser muy complejo.

- Monitorización
- Respuesta ante desastres
- Configurar balanceado de carga
- Gestionar incrementos súbitos de tráfico y, por ende, potencia de proceso.
- Etc...

Y esto multiplicado por el número de servidores o máquinas virtuales que tengan disponibles, lo cual es exponencialmente complejo. Todo esto impacta directamente sobre otro importante factor a considerar: los costes de operación.

Finalmente, otra cuestión que preocupa a todos los responsables de sistemas informáticos es la capacidad para recuperarse ante desastres. No es raro ver empresas que ven peligrar la continuidad de su negocio cuando su "cuarto de servidores" se inunda accidentalmente.

Desplegando las aplicaciones en infraestructuras en la nube es posible despreocuparse de todas estas complejidades, centrándose específicamente en las necesidades funcionales de las aplicaciones, y no en los detalles necesarios para hacerlas funcionar.

1.2.- En definitiva…

¿Por qué mantener y pagar sistemas que sólo trabajan a pleno rendimiento en ciertos momentos? ¿Por qué tener que pagar por infraestructura que se infrautiliza? ¿Por qué no dejar en manos de expertos la seguridad de los datos, la disponibilidad de las aplicaciones? ¿Cómo asegurar la redundancia de nuestros datos sin carísimas inversiones?

Escalabilidad, alta disponibilidad y reducción de costes de operación son las principales promesas de la nube.

> **Nota**: Microsoft dispone de una herramienta gratuita para determinar los costes totales de propiedad (TCO) así como el retorno de la inversión (ROI) de una aplicación, que compara la opción de gestionarla directamente en una empresa con la de albergarla en la plataforma Windows Azure. Es interesante utilizarla para analizar costes y averiguar lo que más puede interesar según el caso...

2.-¿QUÉ ES CLOUD COMPUTING?

La Nube es la respuesta a las necesidades de los departamentos de tecnología de la información de las empresas en la búsqueda de mayor escalabilidad, alta disponibilidad y reducción de los costes operativos.

La idea fundamental es simple: dejemos en manos de empresas especializadas la gestión de la infraestructura en la que corren las aplicaciones. Los desarrolladores controlan las aplicaciones y los datos pero no se preocupan de la infraestructura sobre la que corren éstas. Es una segunda empresa la que se encarga de proporcionar esta infraestructura y no solo a una empresa en concreto, sino a cientos de miles.

La ventaja es clara, se trata de economía de escala: considerando los costes por unidad de tiempo de ejecución, es muchísimo más económico establecer para miles de aplicaciones una infraestructura altamente escalable, con muy elevada disponibilidad y con ancho de banda cuasi-ilimitado, que hacerlo para una decena de ellas.

Las características fundamentales que debe proporcionar un servicio para poder ser considerado "Cloud Computing" son:

- Auto-servicio por parte de los clientes
- Capacidad de medir el servicio
- Pago por uso

- Rápida elasticidad
- Distribución geográfica

Dentro de los servicios en la nube se pueden distinguir tres categorías fundamentales:

1. **SaaS (Software como Servicio)**: se trata del cambio a versiones en la nube de aplicaciones tradicionalmente instaladas en una infraestructura propia. Por ejemplo Exchange Online es una versión SaaS del servidor de correo Exchange Server de Microsoft, que se ofrece hospedada en los Data Centers de Microsoft o sus partners. Una empresa puede contratarlo y empezar a usarlo de inmediato sin tener que adquirir licencias ni hardware específico, sin tener que mantenerlo y pagando sólo por lo que utiliza. Actualmente Microsoft ofrece una amplia selección de aplicaciones en modo SaaS a través de sus Online Services.
2. **IaaS (Infraestructura como Servicio)**: es el hospedaje de aplicaciones existentes dentro de máquinas virtuales en la nube. Siguen siendo aplicaciones normales, ejecutadas en un sistema operativo común, sólo que se ejecutan dentro de máquinas virtuales en la nube. Permite despreocuparse de la gestión de los servidores físicos, de modo que evitamos problemas de fallos de hardware o desastres locales, y suelen ofrecer capacidad para crecer en el almacenamiento. No obstante sigue siendo necesario ocuparse de muchas otras cuestiones de infraestructura como la alta disponibilidad, el balanceo de carga, gestión del S.O y el software base, etc... Microsoft permite gestionar esto de manera privada a empresas a través de System Center y sus productos de virtualización. Los ejemplos de carácter público más conocidos de este tipo de servicio son Amazon EC2 y RackSpace.
3. **PaaS (Plataforma como Servicio):** se trata de los servicios en la nube que permiten crear aplicaciones específicamente desarrolladas para trabajar en entornos "Cloud" con las ventajas ya señaladas. Toda la infraestructura de base, tanto el software como el hardware, son transparentes para nosotros. Se crea la aplicación y se despliega obteniendo alta disponibilidad, alta escalabilidad, y nulos costes de operación. La desventaja es que cada plataforma PaaS tiene sus propias API de desarrollo por lo que es complicado mover una aplicación de una a otra "nube". Es en esta categoría en donde se encaja Windows Azure Platform. Otros ejemplos serían App Engine de Google o Force.com de Salesforce.

Microsoft dispone de amplia experiencia de ofrecer servicios en la nube, desde hace al menos 11 años, tanto en el ámbito del consumidor final como en el mundo empresarial. Para hacerse una idea sólo algunas cifras al respecto:

- Alberga más de 500 millones de cuentas de correo de Hotmail/LiveMail en el mundo
- Gestiona más de 620.000 cuentas de Exchange sólo en España
- Soporta más de 30.000 millones de autenticaciones con LiveID al mes
- El portal MSN sirve más de 10.000 millones de páginas al mes.
- En promedio, cada 30 días se mueven a través del servicio Messenger más de 240.000 millones de mensajes.
- Se realizan 2.000 millones de búsquedas al mes en el buscador Bing.

Unas cifras realmente impresionantes.

Además de los centros de datos actuales, Microsoft está construyendo enormes Data Center nuevos en todo el mundo con el propósito de proporcionar servicio adicional en el futuro a los clientes de Windows Azure Platform.

Microsoft también ofrece Windows Azure Platform en formato appliance, lo que posibilita que los clientes o partners puedan llegar a montar sus propios Data Centers disponiendo de toda la funcionalidad que Windows Azure Platform ofrece.

Los appliance están orientados a proveedores de servicios, grandes empresas y gobiernos que por diversos motivos puedan necesitar disponer de una infraestructura propia, sin que ello suponga perder otros beneficios de Windows Azure Platform.

Otra cuestión importantísima de Windows Azure Platform frente a servicios de otros fabricantes es que incluye en el contrato un acuerdo de nivel de servicio (SLA). Es decir, al contrario que en otros servicios PaaS, Microsoft se compromete a garantizar elevados niveles de disponibilidad, capacidad, conectividad así como monitorización y supervisión, lo que es una garantía adicional para aplicaciones críticas empresariales. Las distintas SLA de la plataforma Windows Azure se pueden consultar aquí.

3.-APLICACIONES PRÁCTICAS DE LA NUBE

El concepto de Nube puede encajar a la perfección en algún de los patrones típicos de carga que se pueden dar en las aplicaciones de software:

- Aplicaciones con "picos" predecibles: como el ejemplo anterior de Wimbledon, en el que se sabe de antemano que la demanda va a multiplicarse enormemente en determinadas épocas o ante eventos concretos. Con una implementación tradicional se desperdiciaría capacidad y generaría gran complejidad para el departamento de TI. En la nube se pueden contratar los recursos necesarios exactamente el tiempo que se necesiten.

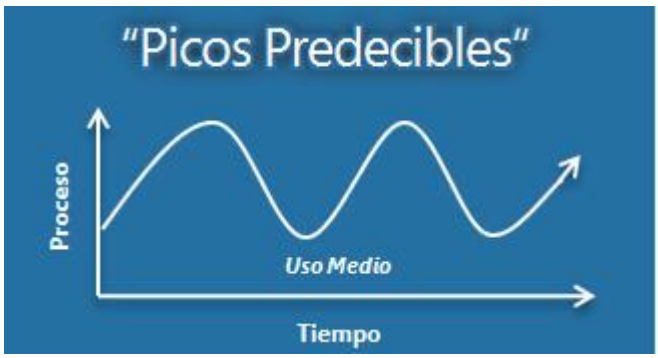


Figura 1.2.- Proceso de las aplicaciones con picos predecibles

- Aplicaciones con "picos" impredecibles: no es posible determinar cuándo van a producirse ni de qué dimensión serán, por lo que dimensionar la infraestructura apropiada no es posible con el esquema tradicional. Además estos picos impactan en el rendimiento y por tanto en el negocio, que puede verse incluso interrumpido. Los servicios Cloud pueden escalarse de inmediato ante una demanda inesperada.

Figura 1.3- Proceso de las aplicaciones con picos impredecibles

- Aplicaciones de crecimiento rápido: aquellas que crecen mucho en demanda en virtud de un gran éxito entre los usuarios. Escalar y crecer es un gran reto tanto de desarrollo como para el equipo de TI de las empresas. Por ejemplo, la verdadera dificultad de una aplicación como Twitter, que es funcionalmente muy sencilla, es el poder crecer y llegar a gestionar millones de usuarios simultáneos. Albergada en un sistema en la nube puede aumentar su capacidad ilimitadamente a medida que el número de usuarios crece.

Figura 1.4.- Proceso de las aplicaciones con crecimiento rápido

- Aplicaciones On-Off: aplicaciones que trabajan y se paran de manera predecible, alternando periodos de inactividad con periodos de trabajo intenso. En el esquema tradicional se desaprovechan mucho las capacidades del sistema, que estarán sobredimensionadas. Con el esquema de la nube se pueden gestionar cambios para aumentar y reducir las capacidades según cada fase, disminuyendo los costes a cero en los periodos sin demanda.

Figura 1.5.- Proceso de las aplicaciones "on-off"

4.-¿QUÉ ES WINDOWS AZURE PLATFORM?

Windows Azure Platform es un servicio PaaS que forma parte de la oferta de servicios online de Microsoft. Proporciona un entorno familiar y flexible para desarrollar aplicaciones y servicios en la nube con todas las ventajas que esto supone.

Con Windows Azure Platform una empresa puede reducir el tiempo de lanzamiento de los productos y adaptarse fácilmente a medida que la demanda de éstos crezca.

Figura 1.6.- Componentes de Windows Azure

Windows Azure es una plataforma interoperable, que permite desarrollar en diversos lenguajes así como la comunicación con cualquier entorno externo. Además, si bien Visual Studio es la herramienta más productiva a la hora de trabajar con la plataforma, se dispone de herramientas y SDKs para otros sistemas y entornos.

La plataforma de Windows Azure es un conjunto de servicios base en la nube, que pueden usarse conjuntamente o de manera independiente, el cual permite que:

- Los desarrolladores utilicen sus habilidades actuales y sus herramientas conocidas para desarrollar aplicaciones cloud
- Los ISVs y los integradores de sistemas ingresen al mercado con rapidez y paguen a medida que usted ingrese
- Los administradores IT obtengan acceso a un nuevo conjunto de recursos sin agregar complejidad
- Empresas de todos los tamaños respondan con rapidez a medida que cambian las necesidades comerciales

Widows Azure Platform está compuesta por los siguientes servicios:

- Windows Azure: ofrece a los desarrolladores servicios de ejecución y almacenamiento bajo demanda. Dada la abstracción de la que provee a las aplicaciones se suele definir como el sistema operativo de la nube, lo cual es una forma más sencilla de entenderlo. Utilizando este servicio, los desarrolladores podrán desplegar y gestionar sus aplicaciones en los centros de datos de Microsoft. Además ofrece mecanismos simples de almacenamiento (tablas y blobs) y de comunicaciones basadas en colas.
- Microsoft SQL Azure: es una base de datos relacional en la nube que permite tener acceso a datos relacionales desde cualquier lugar en cualquier momento. Se puede considerar como un servidor de datos SQL Server convencional pero adaptado para funcionar en la nube, en donde la alta disponibilidad es una de las prioridades. Hay que destacar que SQL Azure es el primer gestor para la nube realmente relacional, que puede ejecutar y entender consultas SQL.
- Windows Azure Platform AppFabric: es la parte de Windows Azure Platform que proporciona autorización, autenticación y mensajería de manera que permite la comunicación segura entre las aplicaciones desplegadas en una organización y las aplicaciones desplegadas en Windows Azure.
- AppFabric Service Bus ofrece a los desarrolladores la flexibilidad para escoger cómo se comunican sus aplicaciones, solucionando retos impuestos por cortafuegos, NAT, IP dinámicas, etc.
- AppFabric Access Control posibilita una autorización simple y segura para servicios web REST, que además puede federarse con un gran abanico de proveedores de identidades.
- MarketPlace: Se trata de un servicio que permite a los desarrolladores e 'Information Workers' encontrar, adquirir y gestionar suscripciones a datos en la plataforma Windows Azure.

Dallas es un marketplace de información donde podremos encontrar datos de diferente naturaleza y diversos proveedores en un único punto. A través de las APIs de Dallas se puede consumir ese contenido desde prácticamente cualquier plataforma, aplicación o flujo de negocio.

El siguiente gráfico se muestra los servicios que esta plataforma proporciona y como se relacionan entre sí:

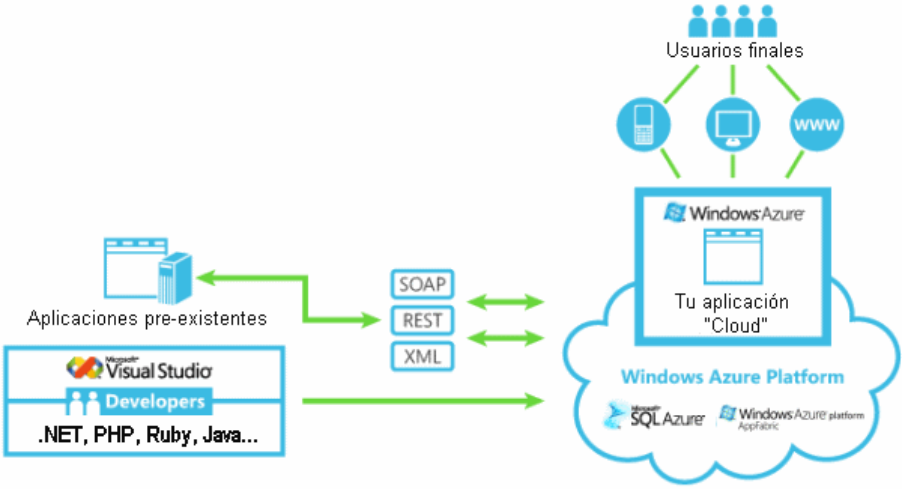


Figura 1.7.- Servicios de Windows Azure Platform

Se pueden ver en el gráfico como Windows Azure Platform proporciona servicios que se pueden utilizar desde cualquier plataforma gracias al uso de estándares universalmente aceptados como SOAP, REST y XML. Está soportado el desarrollo directo sobre la plataforma con .NET pero también con PHP, Java, Ruby on Rails... y desde diferentes entornos aparte de Visual Studio, como por ejemplo Eclipse.

5.-¿QUÉ ES WINDOWS AZURE?

Windows Azure es el sistema operativo en la nube de Microsoft. Proporciona un entorno gestionado para la ejecución y el despliegue de aplicaciones y servicios en la nube. Windows Azure proporciona a los desarrolladores un entorno de computación bajo demanda y almacenamiento alojado en los centros de datos de Microsoft para aplicaciones en la web.

Aunque Windows Azure permite reutilizar todos los conocimientos de .NET es también una plataforma abierta a otros lenguajes y plataformas mediante el uso de estándares y el soporte para lenguajes ajenos a la plataforma .Net como, por ejemplo, PHP, lenguajes no manejados como C/C++ nativo, así como soporte para el protocolo FastCGI.

Entre las ventajas de Windows Azure se encuentran la reducción de costes de operación y aprovisionamiento de las aplicaciones, la respuesta rápida a cambios en las necesidades de los clientes y el negocio, la capacidad para escalar según las necesidades de la aplicación, etc...

Desde el punto de vista del desarrollo con .NET, Windows Azure permite ejecutar aplicaciones ASP.NET y código .NET en la nube, proporcionando una plataforma de ejecución basada, a día de hoy, en el framework de .NET 4.0 e IIS 7, complementado con un API de almacenamiento y de "tracing" propios de la plataforma Azure. Ofrece un portal que permite gestionar las aplicaciones Azure de una manera cómoda y natural.

El entorno de ejecución de Windows Azure es 'full trust' lo que permite ejecutar servicios de Windows Communication Foundation e incluso llamar a código nativo desde las aplicaciones Azure basadas en .NET.

Además Windows Azure va a proporcionar servicios de almacenamiento no relacional y colas con acceso autenticado, alta redundancia (triple) y accesible mediante una interfaz REST que se puede utilizar desde cualquier lenguaje que tenga la capacidad de realizar peticiones HTTP, que es tanto como decir cualquier lenguaje moderno.

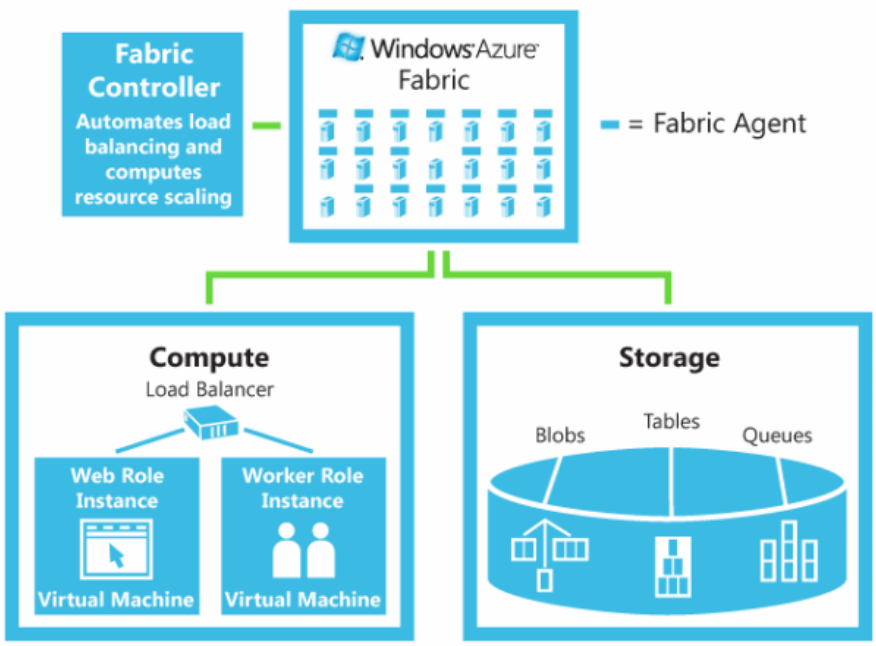


Figura 1.8.- Arquitectura de la infraestructura de Windows Azure

En el gráfico anterior se muestra como elemento a destacar el Windows Azure Fabric, los cimientos sobre los que se levanta la plataforma Azure.

"El Fabric", como se conoce familiarmente, es el componente de la arquitectura que se encarga de proporcionar los servicios base de Windows Azure de manera transparente respecto a la infraestructura IT subyacente.

El desarrollador no sabrá nunca en qué máquina concreta del centro de datos de Microsoft se está ejecutando la aplicación Azure. El Fabric se encarga de asegurar que la aplicación recibe tiempo de ejecución, ancho de banda y recursos en general para su ejecución, balanceando la carga a las máquinas virtuales que considere necesario de manera trasparente para la aplicación.

Además orquesta para la aplicación el acceso a los recursos de almacenamiento y colas de la plataforma Windows Azure, siendo todo ello algo de lo que no se debe preocupar el desarrollador.

5.1.- Beneficios de Windows Azure

- Ejecutar procesos genéricos en la nube
- Crear, modificar y distribuir aplicaciones escalables con un mínimo de recursos internos
- Realizar almacenamiento de alto volumen, procesamiento de lotes y cómputos intensos o de alto volumen
- Crear, evaluar, depurar y distribuir servicios web con rapidez y de forma accesible
- Llevar sus ideas al mercado con mayor rapidez y paga cuando lo obtiene
- Reduce costes de generación y extensión de recursos internos
- Reduce el esfuerzo y los costes de administración de TI Responde con rapidez a los cambios de las necesidades de su empresa y sus clientes
- Amplía y reduce sus recursos de TI en función de sus necesidades
- Consume recursos de informática SOLO cuando surgen la necesidad
- Se enfoca menos en administrar restricciones y recursos operativos
- Elimina la necesidad de administrar hardware
- Utiliza sus actuales habilidades de desarrollo para crear aplicaciones en la nube

6.-¿QUÉ ES SQL AZURE?

SQL Azure es una base de datos relacional en la nube construida sobre la tecnología de SQL Server. Proporciona servicios de bases de datos altamente escalables y con altísima disponibilidad alojados por Microsoft en la nube. Estos servicios facilitan enormemente el despliegue de bases de datos.

Una ventaja añadida es que los desarrolladores y el personal de IT no necesitan instalar, actualizar y gestionar la infraestructura de bases de datos. La alta disponibilidad, aspecto siempre complejo, es gestionado de manera transparente.

La gran ventaja de utilizar SQL Azure frente a otros sistemas de almacenamiento en la nube es que todos los conocimientos sobre bases de datos relacionales y el lenguaje de consulta SQL siguen siendo válidos. No es necesario adaptar los conocimientos a nuevos paradigmas de almacenamiento, como pasa con otros sistemas de almacenamiento en la nube no basados en bases de datos relacionales ni SQL. "Si sabes utilizar SQL Server, todos tus conocimientos te valen para SQL Azure".

SQL Azure permite incluso migrar backends de datos a la nube si tener que tocar ni una sola línea de código de las aplicaciones en un gran número de escenarios. Es cierto que hay ciertas características de SQL Server que SQL Azure no soporta, pero si soporta todas las más usadas:

- Tablas, tablas temporales, vistas, índices, roles, procedimientos almacenados y funciones.
- Consultas complejas y 'joins' entre múltiples tablas.
- Insert, update y delete.
- Restricciones
- Transacciones

Entre las características no soportadas cabe destacar:

- Transacciones distribuidas
- El broker de mensajes de SQL Server
- Consultas a servidores remotos
- Acceso desde tecnologías antiguas, ya obsoletas, en concreto OleDb.

A la hora de conectarse desde las aplicaciones clientes, se pueden elegir varios tipos de conexión:

- ADO.NET, incluido Entity Framework.
- Acceso ODBC nativo.
- Soporte para PHP.

SQL Azure es parte de la plataforma .NET, pero tiene un coste adicional al coste de Windows Azure. Windows Azure proporciona servicios de almacenamiento no relacionales.

7.-¿QUÉ ES WINDOWS AZURE APPFABRIC?

Windows Azure AppFabric

Windows Azure Platform AppFabric proporciona un bus de servicios empresarial y un servicio de control de acceso que permite integrar servicios y aplicaciones que se ejecutan en la nube, en proveedores de alojamiento tradicionales y en la propia empresa basándose en estándares de interoperabilidad.

7.1.- AppFabric Service Bus

Un bus de servicios empresarial (AppFabric Service Bus) permite orquestar la conectividad segura entre diferentes servicios y aplicaciones a través de cortafuegos y redes utilizando numerosos patrones de comunicación.

Los diferentes servicios se registran en el bus de servicios de manera que pueden ser fácilmente accedidos a través de las más variadas tipologías de red. Si una aplicación tiene que consumir e interactuar con una gran cantidad de servicios, algunos de ellos controlados por terceros, utilizar un bus de servicios permite "olvidarse" de detalles como la autenticación y autorización, los protocolos de comunicación, los cortafuegos y otras cuestiones técnicas, delegándolos en el bus de servicios. De esta manera, los desarrolladores pueden centrarse en solucionar escenarios de negocio y no perderse en los detalles de implementación de los servicios.

El bus de servicios de la plataforma Azure facilita la labor de conectar aplicaciones que se ejecutan sobre Windows Azure o contra SQL Azure con aplicaciones que corren en una infraestructura propia y contra servidores de bases de datos convencionales.

Otro escenario en el que el bus de servicios ayuda enormemente es en la creación de aplicaciones compuestas mediante la integración de diferentes servicios ya existentes y nuevos servicios que se ejecutan en la plataforma Azure.

A continuación puede verse el esquema de funcionamiento del bus de servicios de la plataforma Azure.

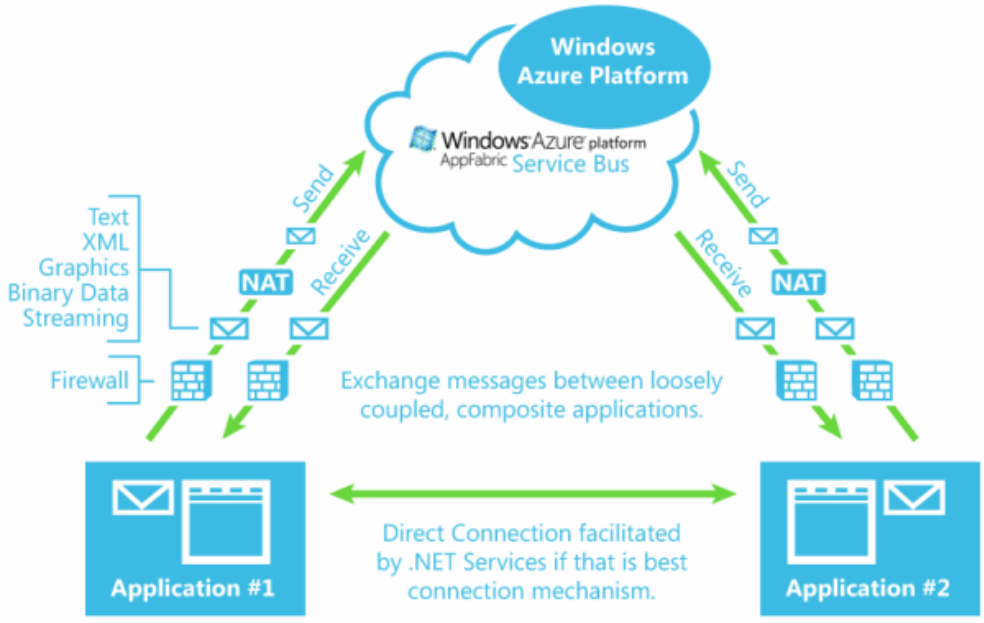


Figura 1.9.- AppFabric Service Bus

Del esquema anterior los puntos más destacables son:

- Que el mecanismo preferido de comunicación entre aplicaciones en la nube es utilizar la infraestructura que el framework de .NET proporciona para tal fin, que es Window Communication Foundation.
- Que cuando la comunicación directa no es posible por cuestiones relacionada con la topología de la red, o por el hecho de que no se controlen todos los servicios y que se necesite integración con otros servicios de terceros, el bus de servicios de .NET puede ahorrar mucho trabajo proporcionando: mecanismos de comunicación amigables con los cortafuegos, comunicaciones basadas en mensajes entre servicios, direccionamiento entre redes heterogéneas (NAT) y servicios de orquestación.

7.2.- AppFabric Access Control

El servicio de Control de acceso (AppFabric Access Control) permite generar una autorización federada entre aplicaciones y servicios, sin la programación complicada que, por lo general, se requiere para proteger aplicaciones que atraviesan los límites de la organización.

Al admitir un sencillo modelo declarativo de reglas y claims, las reglas del Control de acceso pueden configurarse con facilidad y flexibilidad para cubrir varias necesidades de seguridad y distintas infraestructuras de administración de identidades.

A continuación puede verse el esquema de funcionamiento del Control de acceso de la plataforma Azure.

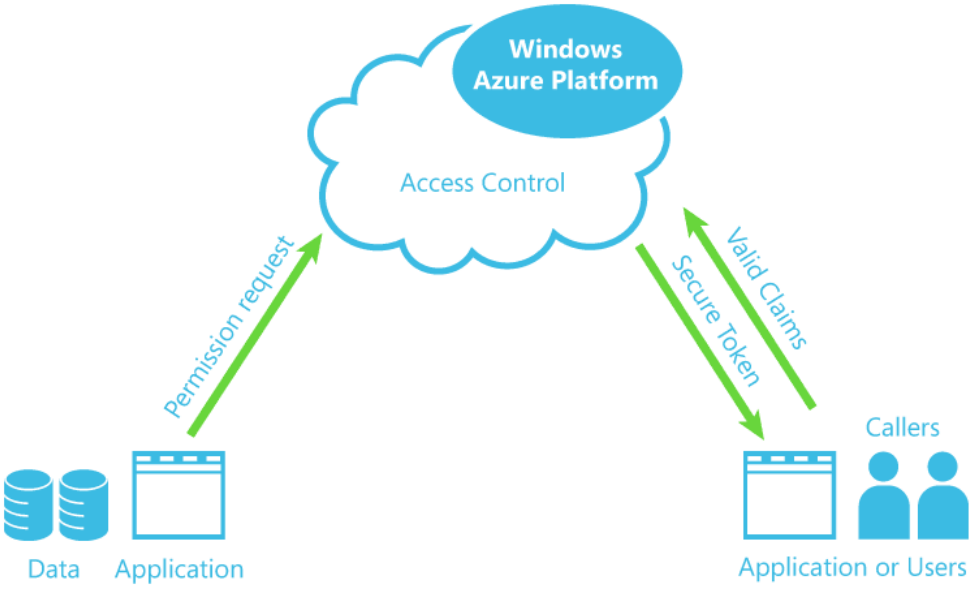


Figura 1.10.- AppFabric Access Control

No se puede decir que este componente de la plataforma Azure sea de uso común en la gran mayoría de las aplicaciones Azure, pero sí que es de gran utilidad en las ocasiones en las que orquestar servicios y comunicarlos entre sí es la principal labor que tiene que realizar la aplicación.

7.3.- AppFabric Cache

Windows Azure AppFabric Caching es un sistema de caché distribuida, en memoria, que se ofrece como un servicio en la nube.

Un servicio similar ya existía para soluciones on-premise, integrado dentro de Windows Server AppFabric. Ahora tenemos las mismas capacidades y características para aplicaciones que estén en la nube. Se ofrece como un servicio en la nube, por lo que como veremos nos vamos a tener que hacer nada relacionado con las tareas de instalación, configuración o administración…simplemente hacer uso del servicio.

CAPÍTULO 2

Windows Azure

Windows Azure es el sistema operativo en la nube de Microsoft. Proporciona un entorno gestionado para la ejecución y el despliegue de aplicaciones y servicios en la nube. Windows Azure proporciona a los desarrolladores un entorno de computación bajo demanda y almacenamiento alojado en los centros de datos de Microsoft para aplicaciones en la web.

Aunque Windows Azure permite reutilizar todos los conocimientos de .NET es también una plataforma abierta a otros lenguajes y plataformas mediante el uso de estándares y el soporte para lenguajes ajenos a la plataforma .Net como, por ejemplo, PHP, lenguajes no manejados como C/C++ nativo, así como soporte para el protocolo FastCGI.

Entre las ventajas de Windows Azure se encuentran la reducción de costes de operación y aprovisionamiento de las aplicaciones, la respuesta rápida a cambios en las necesidades de los clientes y el negocio, la capacidad para escalar según las necesidades de la aplicación, etc...

Desde el punto de vista del desarrollo con .NET, Windows Azure permite ejecutar aplicaciones ASP.NET y código .NET en la nube, proporcionando una plataforma de ejecución basada, a día de hoy, en el framework de .NET 4.0 e IIS 7, complementado con un API de almacenamiento y de "tracing" propios de la plataforma Azure. Ofrece un portal que permite gestionar las aplicaciones Azure de una manera cómoda y natural.

El entorno de ejecución de Windows Azure es 'full trust' lo que permite ejecutar servicios de Windows Communication Foundation e incluso llamar a código nativo desde las aplicaciones Azure basadas en .NET.

Además Windows Azure va a proporcionar servicios de almacenamiento no relacional y colas con acceso autenticado, alta redundancia (triple) y accesible mediante una interfaz REST que se puede utilizar desde cualquier lenguaje que tenga la capacidad de realizar peticiones HTTP, que es tanto como decir cualquier lenguaje moderno.

1.-¿CÓMO CREAR UNA CUENTA DE WINDOWS AZURE?

En la página oficial del producto, se puede ver la referencia de precios y promociones relacionados con la plataforma Windows Azure http://www.microsoft.com/windowsazure/pricing/.

Los tipos de suscripciones y promociones se pueden agrupar 3 grandes grupos: Promociones con recursos gratuitos, promociones de bolsas de recursos y la suscripción de producción.

> **Nota**: Es recomendable consultar la página oficial para tener la relación actualizada de las promociones existentes y de los recursos disponibles en cada promoción.

Una vez decidido el tipo o tipos de suscripción a contratar, todas las altas se realizan desde el portal de cliente de Microsoft Online Services.

1.1.- Promociones con recursos gratuitos

Dentro de esta categoría se encuentran dos promociones diferentes. La promoción para suscriptores MSDN y una promoción abierta a todo el mundo. Ambas tienen un punto en común y es que disponen de una bolsa de recursos gratuita. Esto significa que habrá una serie de horas de computación, espacio de almacenamiento, bases de datos y transacciones gratis. El uso que sobrepase esa bolsa de horas se cargará a los precios determinados por tipo de recurso.

Para pruebas, aprendizaje de la plataforma e incluso proyectos piloto, son las opciones más adecuadas.

Recuerde que en la página oficial se indica que se cobra por despliegue en el caso de Windows Azure y por almacenamiento en el caso del storage. De modo que si se usan las cuentas gratuitas para hacer pruebas, hay que recordar borrar los despliegues y los proyectos de storage entre pruebas.

1.2.- Ofertas de bolsas de horas para desarrolladores

En el caso de que las promociones/suscripciones anteriores se queden cortas, existe la posibilidad de contratar unas bolsas de recursos extendidas: La development accelerator core y la development accelerator extended.

La diferencia entre ambas radica en el número y tipo de recursos de cada suscripción, siendo la extended una versión más completa que la core.

Se pueden ver los detalles actualizados de la diferencia de recursos entre las cuentas en la página oficial.

1.3.- Entorno de producción

Por último, si lo que se busca es una suscripción para utilizar en producción, entonces la opción será la suscripción Pay per use / Pay as you Grow.

Con esta suscripción sólo se paga por los recursos que se utilicen.

2.-PROMOCIÓN ESPECIAL DE INTRODUCCIÓN A WINDOWS AZURE

La promoción especial de introducción a la Plataforma Windows Azure está disponible para cualquier persona que quiera probar la plataforma, no hay requisitos de acceso.

> **Nota**: Es recomendable consultar la página oficial para tener la relación actualizada de las promociones existentes y de los recursos disponibles en cada promoción.

Antes de empezar, recalcar que los recursos que se utilizan están limitados en número. Al sobrepasar este número de recursos se cargará en cuenta al precio establecido. El alta de todas las promociones se realizan desde el portal de cliente de Microsoft Online Services.

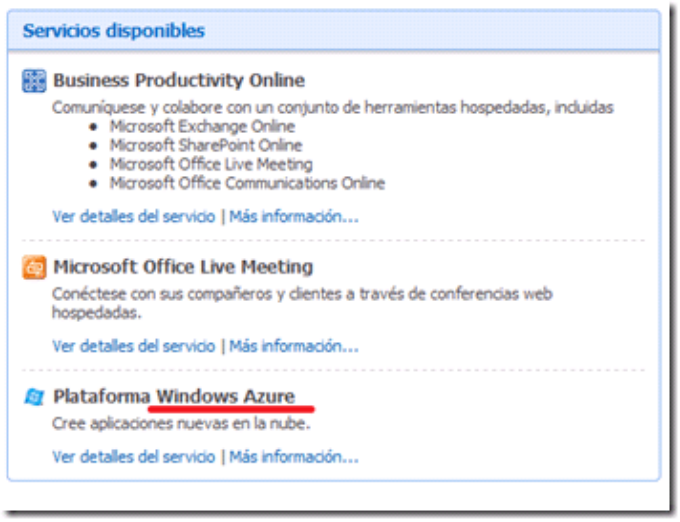


Figura 2.1.- Enlace a oferta de Windows Azure

Para acceder a la oferta gratuita, es necesario darse de alta en el portal de MS Online Services con una cuenta Live ID válida. Una vez registrado, es necesario hacer clic en la pestaña de servicios.

Figura 2.2.- Enlace a la sección de servicios

Y aquí se ve que hay diferentes promociones, pero sólo la primera es la promoción de acceso gratuito, de modo que se debe seleccionar esa. De todos modos, en el plan de tarifas están todos los detalles de facturación y ahí se verá que por el consumo de los recursos asignados el precio es de 0€.

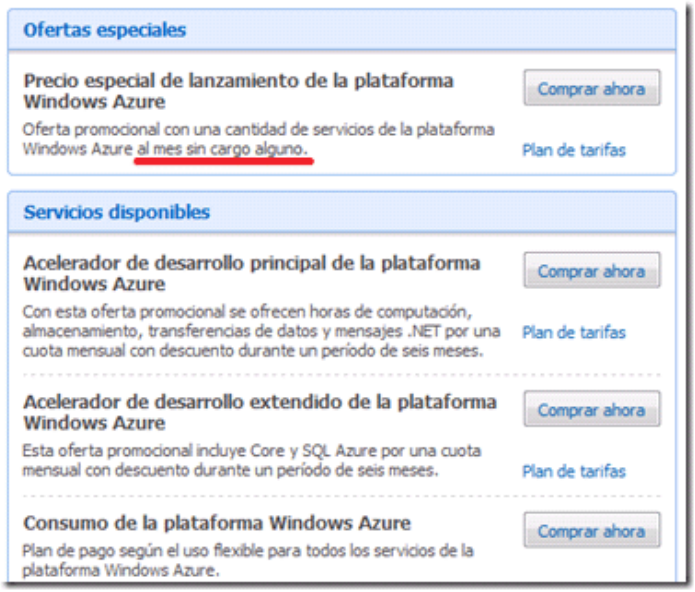


Figura 2.3.- Enlace a la oferta especial

El proceso de alta pedirá una tarjeta de crédito. Como se ha comentado anteriormente, esta promoción da una serie de recursos gratuitos que pueden verse en la oferta, en el caso de que se exceda esa cuota, se cargará el uso de los recursos en la cuenta.

Tras dar la tarjeta y confirmar el pedido, se podrá pasar a activar la suscripción, asignar administrador, darle un nombre a la suscripción...

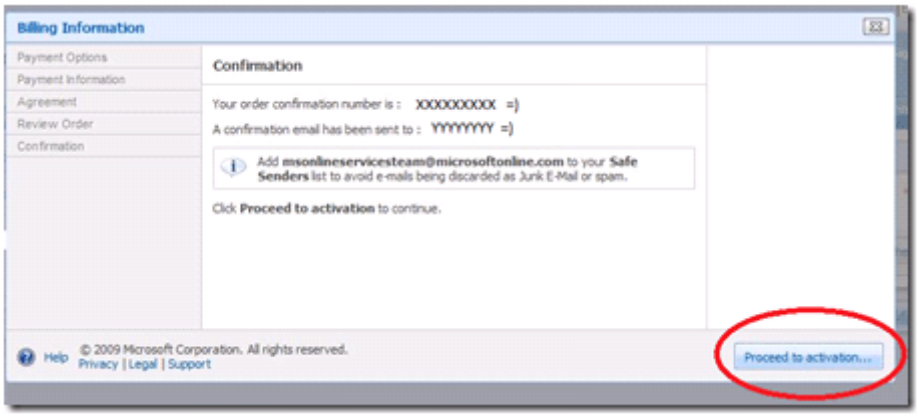


Figura 2.4.- Enlace para activar la subscripción

Una vez activado, llegará un correo notificando indicando que la cuenta está disponible para el uso y con los enlaces al portal de desarrollo.

Al entrar, se verá la suscripción recién creada disponible para trabajar.

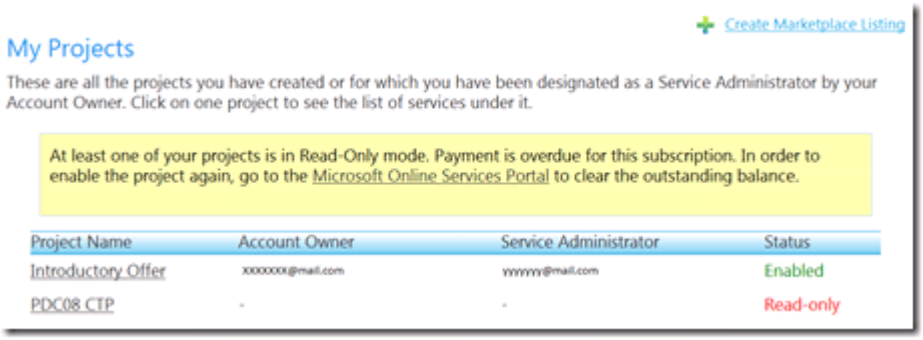


Figura 2.5.- Portal de Windows Azure Platform

3.-PROMOCIÓN ESPECIAL PARA SUBSCRIPTORES DE MSDN

La promoción de beneficios para suscriptores MSDN está restringida a personas/organizaciones que dispongan de una suscripción MSDN.

Antes de empezar, recalcar que los recursos que se utilizan están limitados en número. Al sobrepasar este número de recursos se cargará en cuenta al precio establecido.

El alta de todas las promociones se realizan desde el portal de cliente de Microsoft Online Services.

Los suscriptores MSDN disponen de una bolsa de recursos superior a la que puede acceder cualquier usuario con la promoción introductoria de la Plataforma Windows Azure.

Para activar estos beneficios para suscriptores, se debe partir de la página oficial de ofertas de la Plataforma Windows Azure . En la zona inferior izquierda, aparecerá el siguiente logo:

Figura 2.6.- Enlace a la oferta para subscriptores de MSDN

Al hacer clic llevará a la página de inicio de la activación, donde se indican los países en los que está disponible la promoción. En la parte alta y a la derecha de la pantalla, es posible registrarse haciendo click en "Sign Up Now"

Figura 2.7.- Enlace iniciar proceso de autenticación

Esta acción irá al portal de nuestra suscripción MSDN, donde aparecerá un apartado de Plataforma Windows Azure.

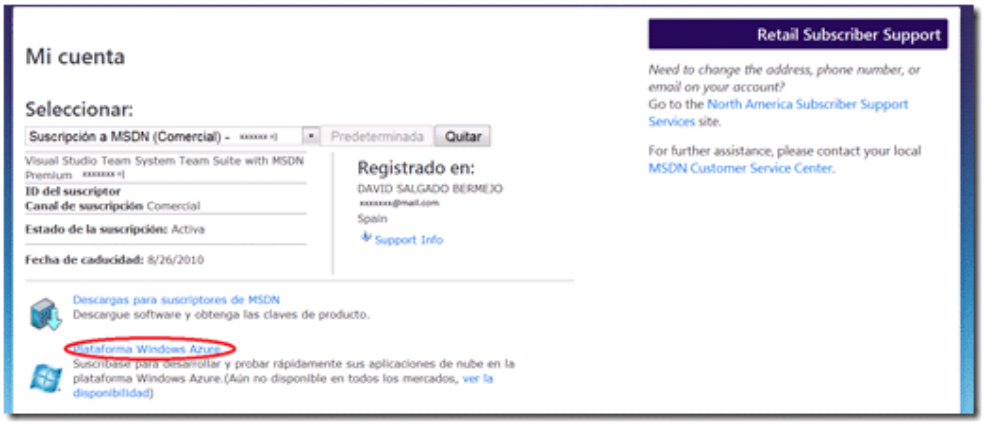


Figura 2.8.- Enlace a la oferta de Windows Azure

Lo que lleva directamente al portal de Microsoft Online Services., donde habrá que hacer login con una cuenta Live ID válida.

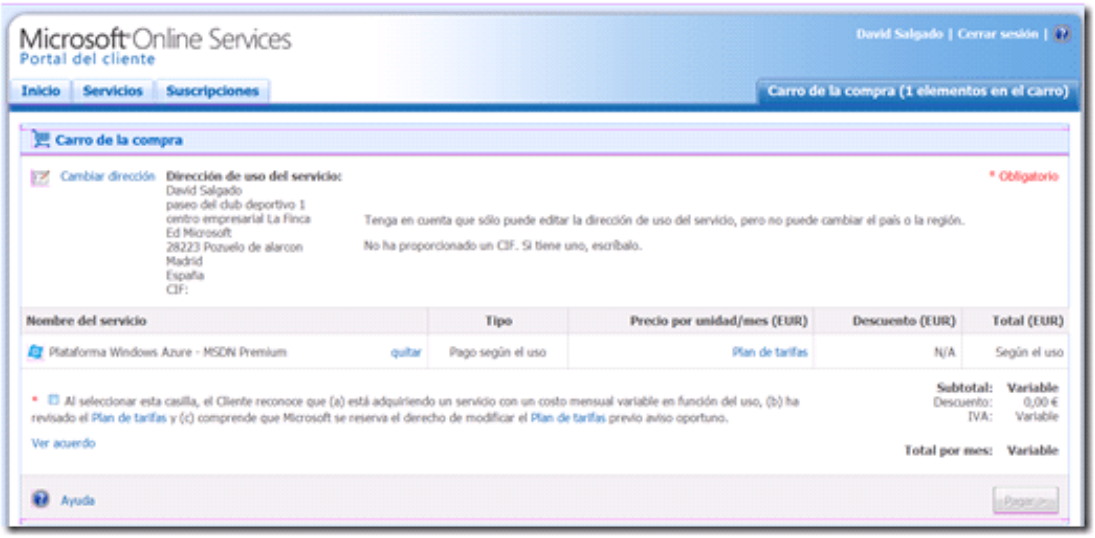


Figura 2.9.- Portal de Microsoft Online Services

Una vez en el portal de Online Services hay que realizar la "compra" del servicio. Si bien la promoción para suscriptores da un número de recursos muy elevado, si lo se supera o si en el futuro (tras los 8 meses de promoción) se quiere renovar, hace falta una tarjeta de crédito.

Una vez en el portal de online services, el proceso es similar al de dar de alta las cuentas de la promoción gratuita.

4.-EL ENTORNO DE EJECUCIÓN DE WINDOWS AZURE

Windows Azure proporciona un entorno de ejecución, un sistema operativo, en la nube. Este entorno de ejecución está basado en Microsoft Windows, Internet Information Server y tecnologías de virtualización. La infraestructura subyacente es trasparente para las aplicaciones. Nunca puede saberse si la aplicación está corriendo en una determinada máquina u otra, o incluso en varias.

Un sistema operativo tradicionalmente proporciona los siguientes servicios:

- Un entorno de ejecución de aplicaciones que actúa como una capa de abstracción sobre el hardware subyacente.
- Un sistema de archivos compartido con control de acceso.
- Un orquestador de recursos compartidos de computación.

Cuando se habla de un sistema operativo en la nube los problemas a resolver son similares, pero la respuesta es ligeramente diferente.

Veamos lo que proporciona Windows Azure en relación a lo que proporciona un sistema operativo tradicional:

- Un entorno de ejecución de aplicaciones que permite abstraerse de la infraestructura subyacente. Cuando se necesita una escalabilidad y una disponibilidad casi ilimitadas, se necesita una capa de abstracción que permita abstraerse de la arquitectura distribuida en infinidad (o no) de máquinas físicas. Si una máquina en concreto de uno de los centros de datos de Microsoft necesita ser sustituida, estas situación, gracias a Windows Azure es totalmente trasparente para la aplicación.
- Una aplicación en la nube, lógicamente, necesita almacenar información. Necesita algo semejante a un sistema de archivos. Lógicamente, una vez más, un sistema de archivos tradicional está demasiado acoplado con máquinas concretas. Si se necesita asegurar la escalabilidad, la redundancia, y la disponibilidad del sistema de archivos en la nube no se puede atar a máquinas concretas. Window Azure se encarga de abstraer los detalles de almacenamiento.
- Evidentemente las aplicaciones en la nube van a compartir infraestructura con cientos de miles o quizás millones de otras aplicaciones. El entorno de ejecución de Windows Azure debe orquestar los recursos que esas aplicaciones usan, dotándolas del tiempo de ejecución necesario, memoria y ancho de banda y repartiéndolo adecuadamente. Lógicamente para una aplicación que se ejecuta en Windows Azure debe ser totalmente transparente el origen de ese tiempo de ejecución, de esa memoria o del ancho de banda de red.

Windows Azure resuelve el mismo problema que un sistema operativo tradicional a la hora de abstraer la aplicación de ciertos detalles. Un sistema operativo nos abstrae del hardware y un sistema operativo en la nube nos abstrae del centro de datos subyacente.

El ya mencionado Fabric Controller es el componente de la arquitectura de Windows Azure que orquesta y abstrae a las aplicaciones de la infraestructura, jugando el papel que en un sistema operativo tradicional jugaría la capa de abstracción de hardware HAL.

Puede encontrarse más información sobre cómo funciona la infraestructura subyacente de un centro de datos de Microsoft en este interesante video sobre el tema en Channel 9 o este otro video que da una pequeña visita por uno de estos centros de datos, ambos están en inglés.

5.-ARQUITECTURA DE UNA APLICACIÓN WINDOWS AZURE

La arquitectura de un servicio alojado en Windows Azure se basa en componentes auto-contenidos desarrollados típicamente con código .NET. Estos componentes son conocidos en Windows Azure como roles.

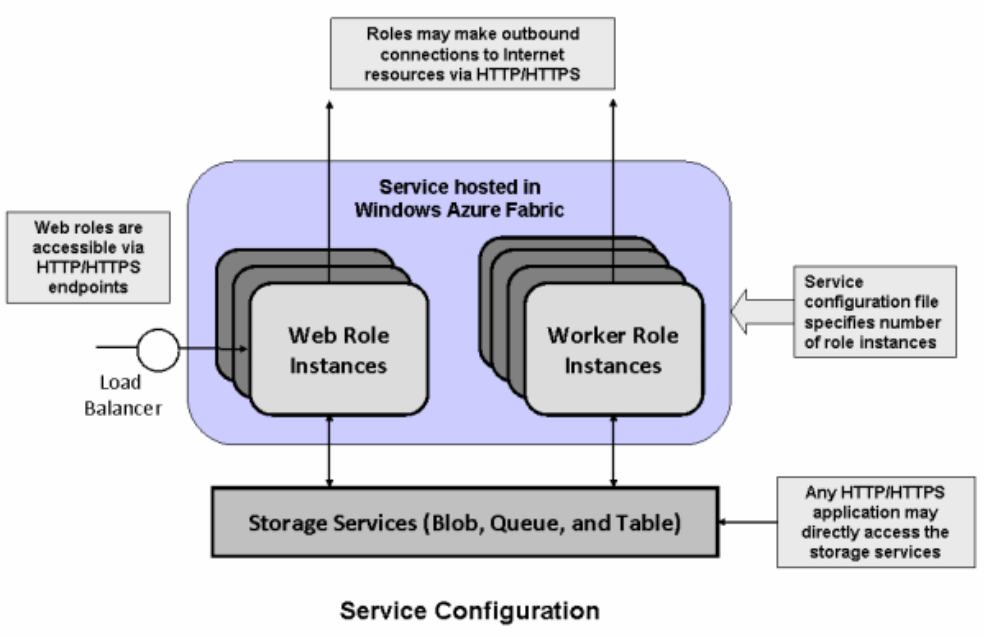


Figura 2.10.- Arquitectura de una aplicación Azure

Una aplicación alojada en Windows Azure se implementa como la composición de uno o más roles. Estas aplicaciones pueden ejecutar una o más instancias de cada uno de estos roles. Este detalle se define mediante simples archivos de configuración.

En función del número de instancias de un determinado rol, este rol recibirá más o menos capacidad de proceso. Cada una de las instancias es replicada en múltiples máquinas físicas. Cada instancia recibe aproximadamente el equivalente a una máquina con las siguientes características: un núcleo de aproximadamente 1.7 Mhz y 2 Gb de RAM.

Existen tres tipos de roles en Windows Azure:

- Web rol: Un 'web rol' es una aplicación basada en web accesible mediante HTTP o HTTPS. Un web rol es alojado en un entorno de ejecución que soporta un subconjunto bastante amplio de ASP.NET y Windows Comunication Foundation.
- Worker rol: Un 'worker role' es un proceso que corre en segundo plano. Sería el equivalente a un servicio de Windows en la plaforma Windows Azure. Un worker rol se puede comunicar con los servicios de almacenamiento y de colas de Windows Azure, incluso puede comunicarse directamente con otros roles.
- Virtual Machine Role: Es un rol que permite al usuario desplegar una imagen (VHD) personaliza de un Windows Server 2008 R2. El usuario dispone de mayor libertad para poder configurar el rol, pero a su vez tiene mayores responsabilidades sobre el estado de la máquina.

Windows Azure impone ciertas restricciones en tiempo de ejecución a lo que un rol puede hacer. Para ello utiliza la combinación de políticas de acceso a código (CAS) de .NET y políticas de seguridad de Windows.

Todos los tipos de roles puede establecer conexiones de salida hacia recursos en Internet usando HTTP o HTTPS y usando TCP/IP sockets, y atender peticiones entrantes sobre HTTP o HTTPS.

Todos los tipos de roles tienen acceso a ciertos servicios que la plataforma de ejecución de Windows Azure expone mediante las librerías del SDK de Windows Azure:

- Acceso al almacenamiento privado del rol. ¡Atención!, no se debe confundir con los servicios de almacenamiento de Windows Azure. Se trata de almacenamiento local que se utiliza típicamente como cache. No se puede confiar en que este tipo de almacenamiento sea persistente en el tiempo y está bastante limitado en capacidad.
- Los servicios para traceo (tracing) y diagnóstico de Windows Azure.

- Servicios que permiten informar al Fabric Controler del estado de la aplicación.

6.-CREAR UN PROYECTO CON VISUAL STUDIO

Las Visual Studio Tools for Windows Azure y el SDK de Windows Azure son elementos imprescindibles para desarrollar aplicaciones con Visual Studio para Windows Azure. Una vez instalados estos componentes en la máquina de desarrollo se podrá comenzar a crear proyectos Windows Azure, tal y como puede verse en la siguiente captura de pantalla:

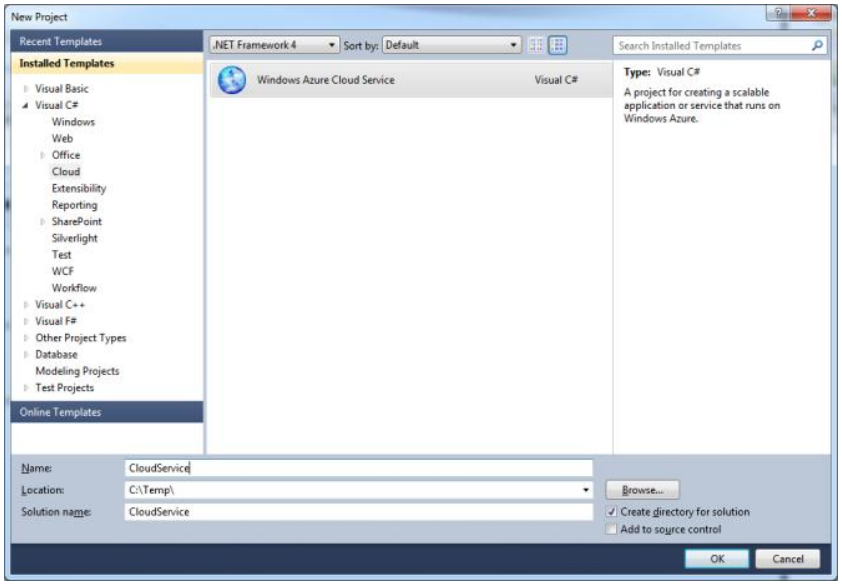


Figura 2.11.- Crear un nuevo proyecto de Azure

En la captura se encuentra seleccionado un proyecto de tipo Cloud Service de Visual C# pero nada impide crear un proyecto de este tipo en Visual Basic.NET. Es posible crear soluciones de Windows Azure que combinen roles desarrollados en diferentes lenguajes.

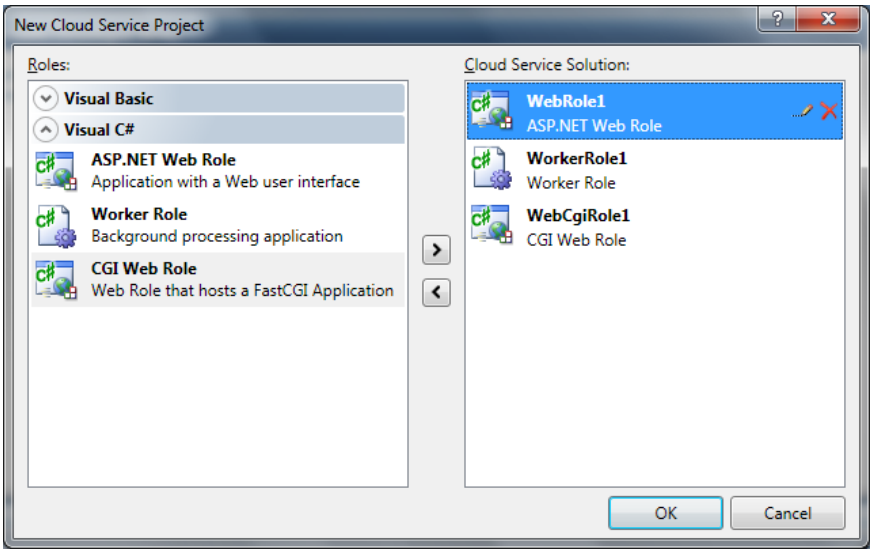


Figura 2.12.- Pantalla de selección de roles del proyecto

Una vez confirmada la creación del proyecto, aparecerá un asistente que permite añadir roles al proyecto de Windows Azure. Pueden añadirse tantos roles de tipo ASP.NET Worker Role o Worker Role como necesite la aplicación a desarrollar, tal y como se puede ver en la siguiente figura.

Es posible editar el nombre de los roles que se añaden poniendo el ratón sobre ellos en el panel de la derecha y pulsando sobre el icono editar. Si se necesitan añadir más roles tras haber creado el proyecto es posible hacerlo sin problemas posteriormente.

Una vez aceptados los pasos dados en este asistente, se creará el proyecto.

7.-EL ENTORNO DE SIMULACIÓN

Resulta evidente que por una simple cuestión de productividad no es viable tener que desplegar las aplicaciones en la nube de Windows Azure cada vez que se quiera depurarlas. Además aunque esto no importase, Windows Azure no permite depurar con Visual Studio las aplicaciones desplegadas.

Otro escenario es que no siempre puede ser posible estar conectados a la nube, puede que la máquina de desarrollo no tenga acceso a Internet en un momento dado.

Es por esto necesario simular de algún modo el entorno de ejecución de Windows Azure, el Fabric, en la máquina de desarrollo. Esta es la misión que lleva a cabo el Compute Emulator.

Como parte del SDK de Windows Azure se instala este componente, Compute Emulator. Puede ejecutarlo manualmente desde el menú de inicio pero lo habitual será que sean las propias Visual Studio Tools for Windows Azure las que se encarguen de levantarlo de manera automática cuando se inicie la depuración o la ejecución de un proyecto de Windows Azure.

Cuando el Compute Emulator se encuentra en ejecución, se podrá ver un icono similar a un engranaje en el área de notificación del sistema operativo Windows, tal y como puede verse en la siguiente imagen:

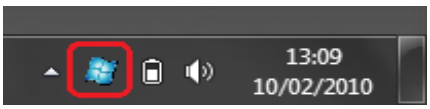


Figura 2.13.-Icono del Compute Emulator

Haciendo doble-clic en ese icono -o desde su menú contextual- se puede acceder a la interfaz del Compute Emulator.

El Compute Emulator permite alojar e interactuar con los servicios de Windows Azure que se están desarrollando sin tener que desplegarlos en el Fabric real de Windows Azure. Este es el aspecto que tiene el Compute Emulator alojando servicios en una máquina de desarrollo:

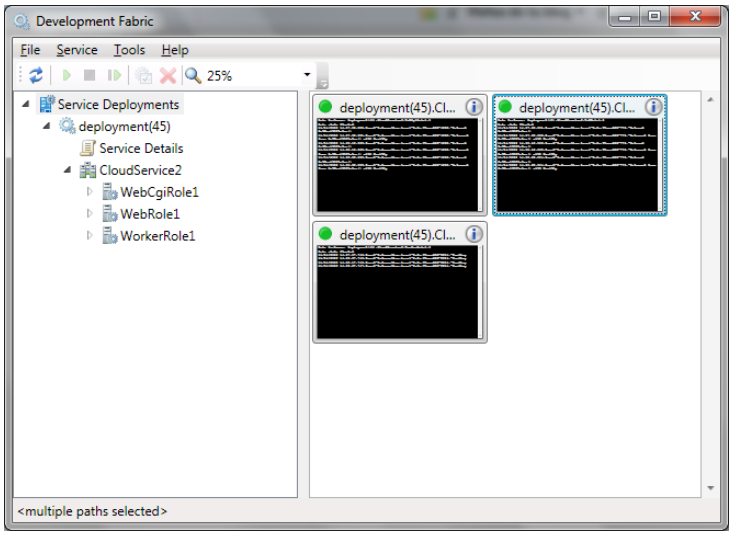


Figura 2.14.- Compute Emulator

El Compute Emulator permite llevar a cabo, entre otras, las siguientes labores:

- Monitorizar el estado de nuestros servicios.
- Establecer el nivel de trazas y verlas.
- Limpiar el log de cada instancia o servicios.
- Adjuntar el depurador de Visual Studio a los servicios.
- Inspeccionar el almacenamiento local de los servicios.

8.-DESPLEGAR UN SERVICIO EN WINDOWS AZURE

A lo largo de este capítulo se mostrará cómo es posible realizar un despliegue de una aplicación en Windows Azure; se mostrará cómo crear un servicio de Windows Azure y cómo desplegar sobre éste una nueva aplicación.

Si se accede a **https://windows.azure.com/** con una cuenta válida se puede ver el portal de administración de Windows Azure, el cual utiliza Silverlight como tecnología.

En la parte superior de la herramienta siempre se encuentra disponible una barra de herramientas que muestra las diferentes acciones que pueden realizarse, en función de lo que se esté mostrando en ese momento en la herramienta.

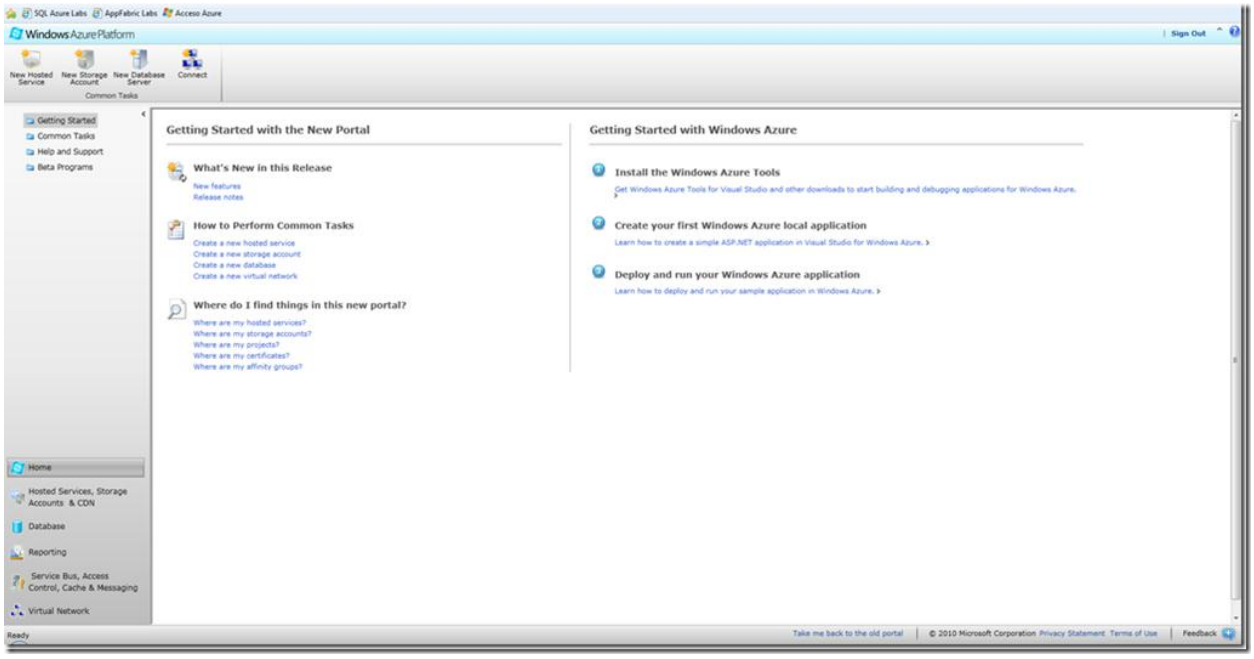


Figura 2.15.- Ventana principal del portal

Si se selecciona la segunda opción del menú vertical inferior, se pueden realizar las diferentes labores administrativas asociadas al despliegue de servicios, ya sea servicios "hosteados" o servicios de almacenamiento.

Si se quiere desplegar una aplicación, será necesario seleccionar "New Hosted Service", para crear el servicio y realizar un despliegue nuevo.

Figura 2.16.- Ventana principal del portal

Se debe indicar el nombre del servicio, URL, región, dónde lo queremos desplegar etc…

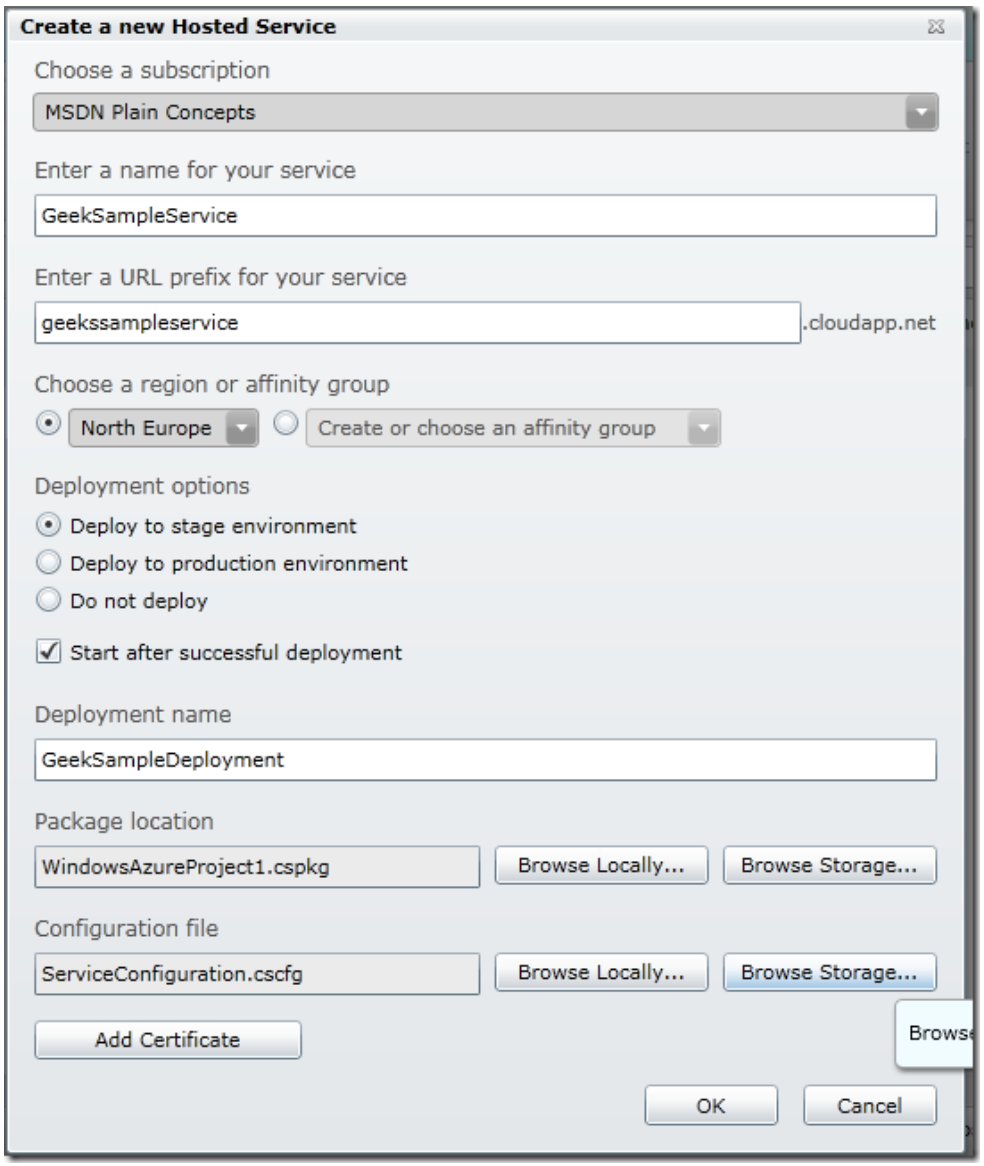


Figura 2.17.- Ventana principal del portal

Al desplegar se puede producir el siguiente warning que avisa que sería recomendable tener más de una instancia por rol para asegurar la disponibilidad de la aplicación.

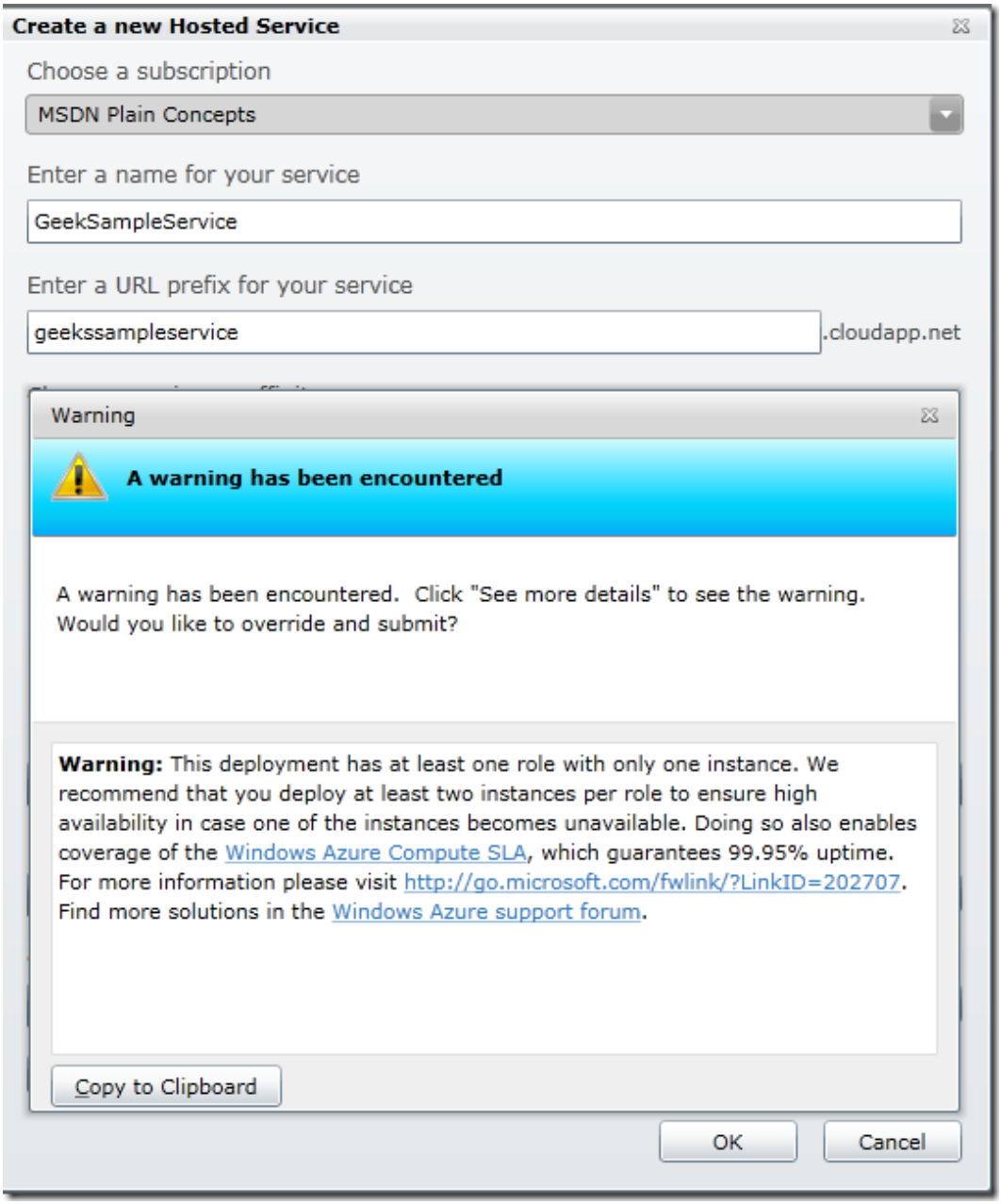


Figura 2.18.- Ventana principal del portal

Una vez creado el servicio, se podrán ver desde la vista principal todos los servicios que se encuentran desplegados, las instancias, el estado de las instancias, la información de los diferentes elementos etc…

En la parte superior siempre aparece una barra de herramientas con las acciones posibles.

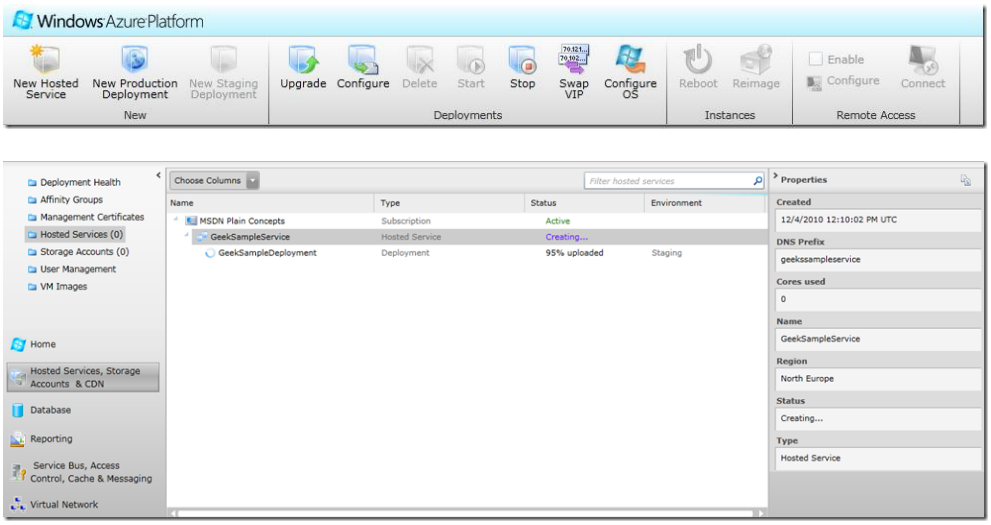


Figura 2.19.- Ventana principal del portal

Se puede configurar qué información debe mostrar, para que cada administrador elija la información que considere relevante:

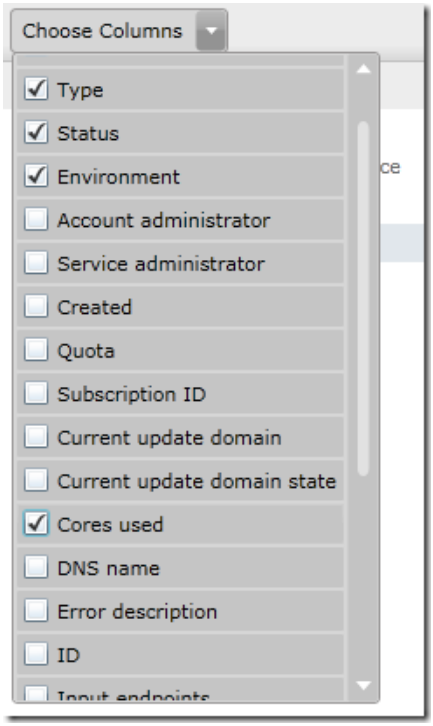


Figura 2.20.- Ventana principal del portal

Crear un servicio de almacenamiento será igual de sencillo.

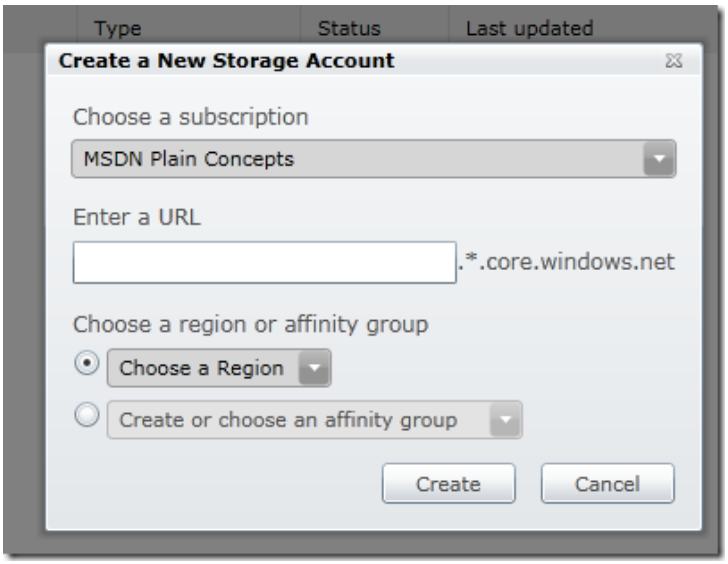


Figura 2.21.- Ventana principal del portal

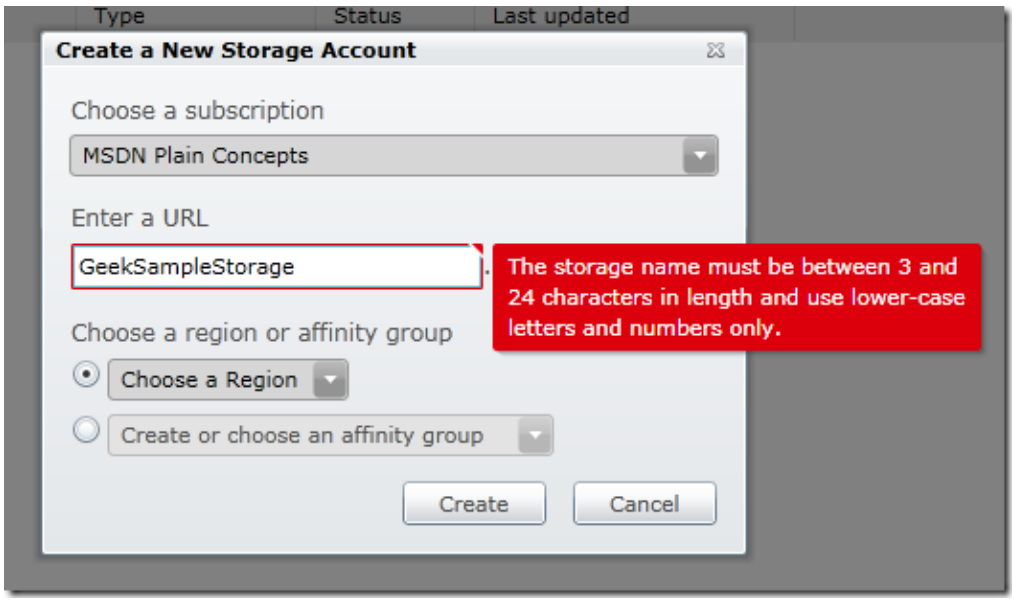


Figura 2.22.- Ventana principal del portal

Y del mismo modo que en el caso anterior, existirá una vista principal dónde pueden verse todo los servicios de almacenamiento.

En la parte superior siempre se aparece una barra de herramientas con las acciones posibles.

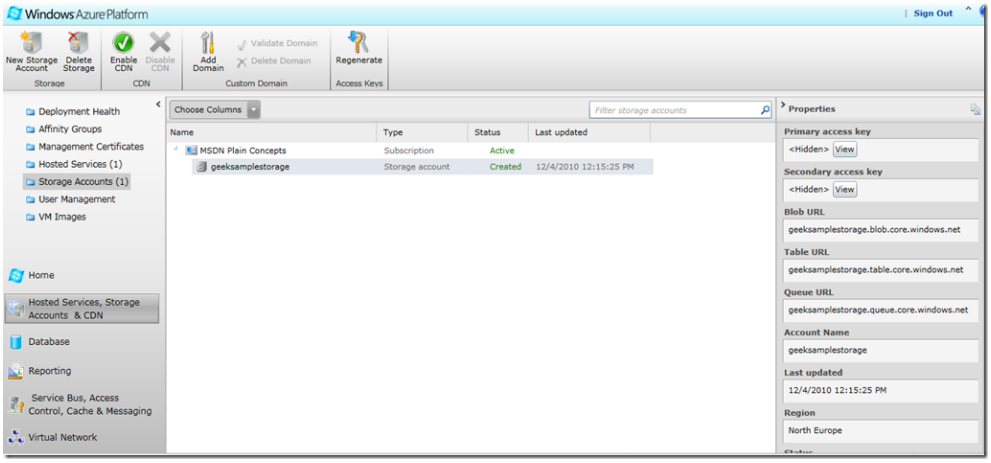


Figura 2.23.- Ventana principal del portal

Una vez desplegado el servicio se puede ver toda la información del mismo.

Una cosa interesante es este cuadro de mando, dónde puede obtener información resumida sobre el "estado de salud" de los servicios desplegados:

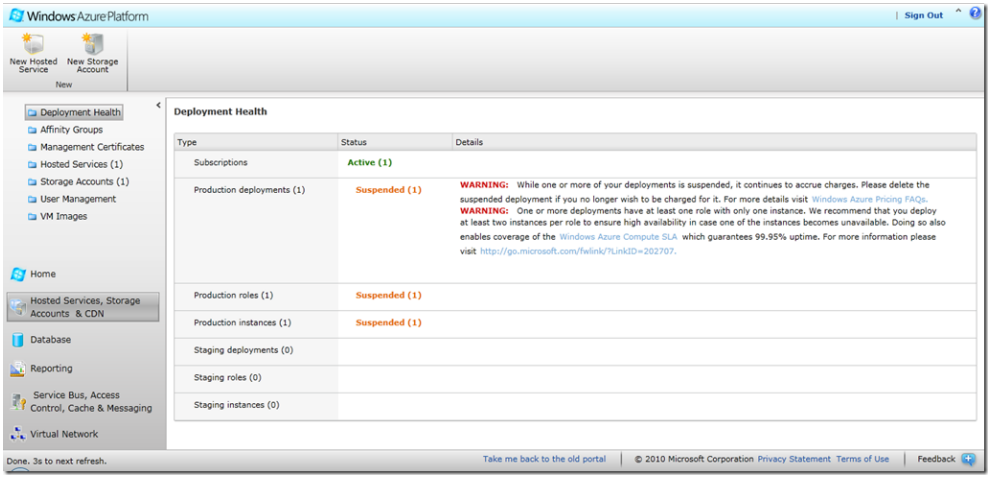


Figura 2.24.- Ventana principal del portal

En la vista de documentos siempre se encuentra disponible una barra de herramientas con las acciones posibles a realizar sobre un servicio:

Figura 2.25.- Ventana principal del portal

Por ejemplo, se puede realizar la opción "swap VIPs", que hace un paso de preproducción (staging) a producción:

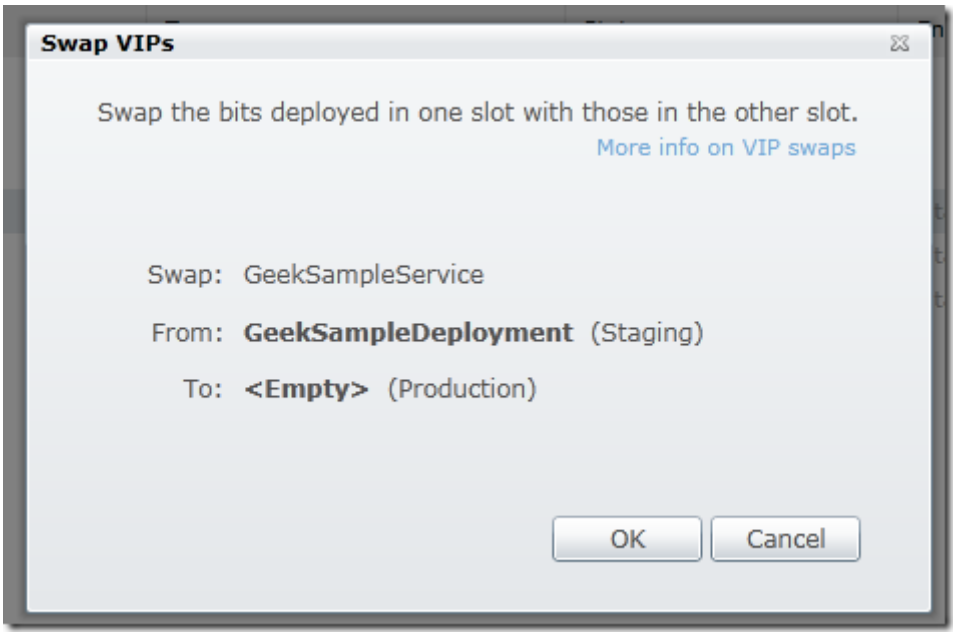


Figura 2.26.- Ventana principal del portal

O cambiar la configuración del servicio desplegado:

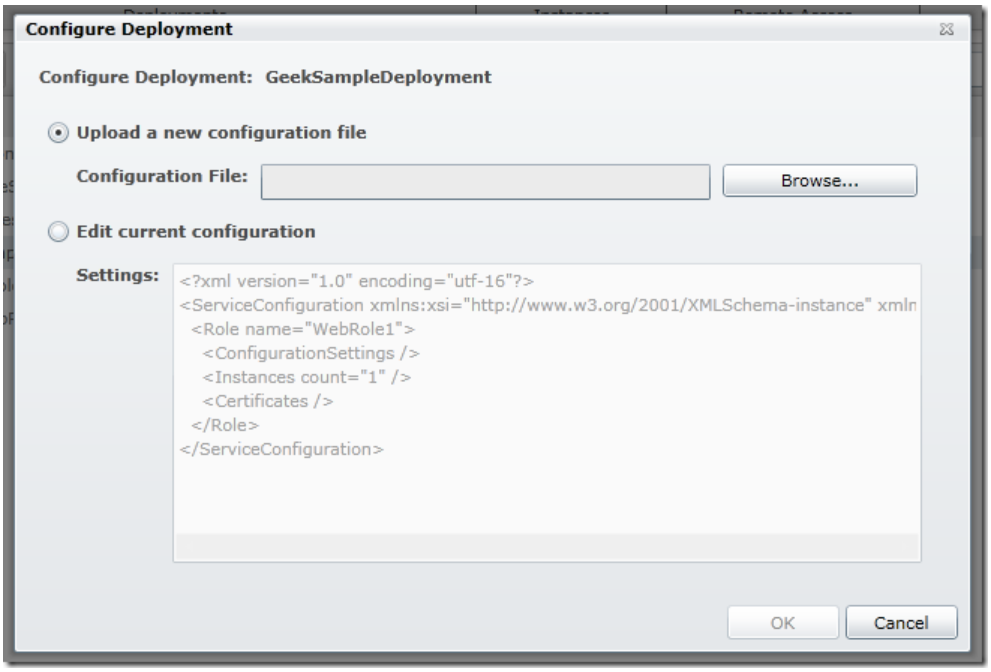


Figura 2.27.- Ventana principal del portal

Cambiar la versión del sistema operativo:

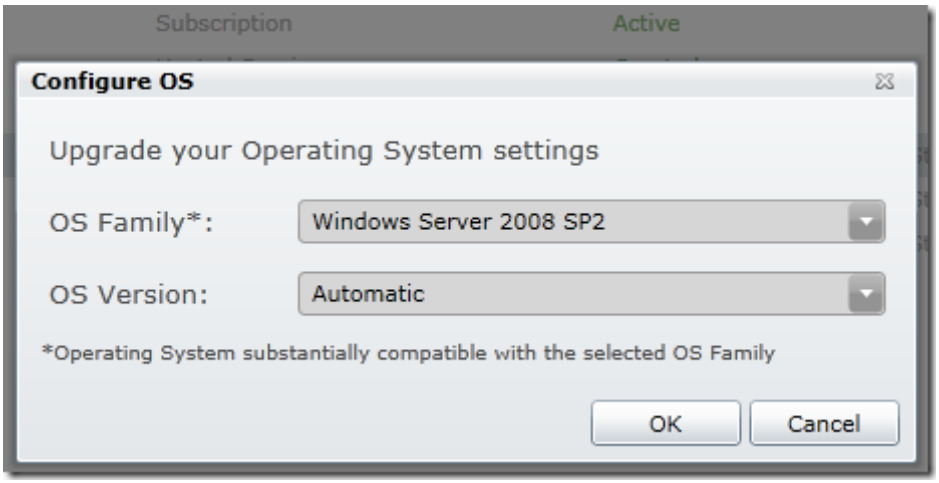


Figura 2.28.- Ventana principal del portal

9.-CONEXIÓN REMOTA A INSTANCIAS DE WINDOWS AZURE

Una de las opciones que nos ofrece la plataforma Windows Azure es que es posible conectarse de forma remota a las instancias desplegadas en Windows Azure. Para poder realizar dicha operación, sólo es necesario configurar el servicio que a desplegar para que permita conexiones remotas.

A parte de establecer el usuario y contraseña de conexión, la única peculiaridad es que necesario indicar un certificado. El objetivo del certificado es similar a cómo se usan los certificados para hacer labores de administración. A la hora de activar la funcionalidad será necesario subir junto con el servicio un certificado. El asistente de publicación permite la creación de un certificado de ejemplo para poder desplegarlo junto con el servicio.

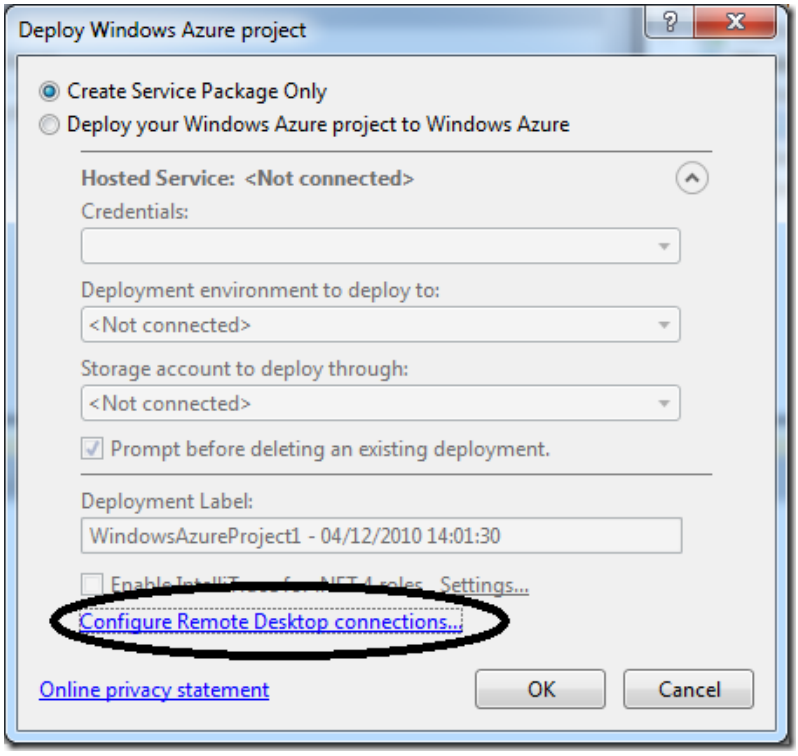


Figura 2.29.- Ventana principal del portal

Una vez realizada dicha configuración, sólo es necesario desplegar el servicio tal y como se ha visto anteriormente.

Lógicamente, lo único que hay que recordar es que el junto con el servicio a desplegar hay que subir el certificado, el fichero pfx. El asistente de publicación mostrará un aviso si previamente a realizar el despliegue no se ha subido el certificado.

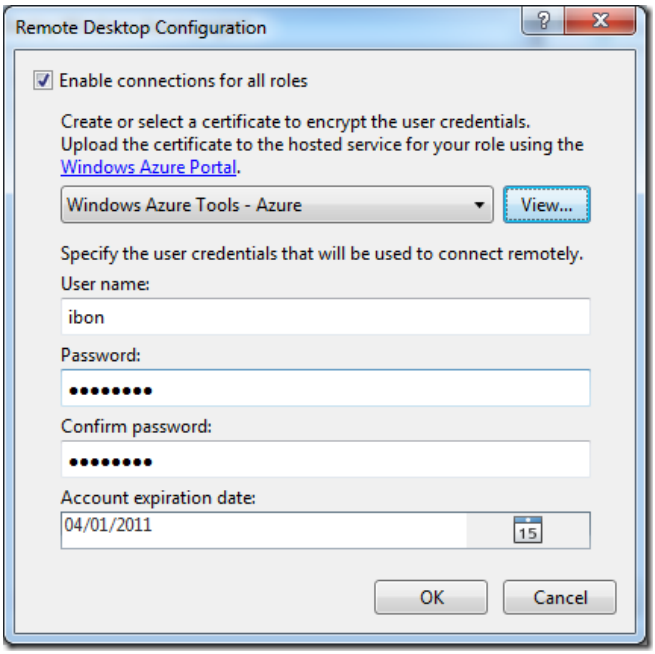


Figura 2.30.- Ventana principal del portal

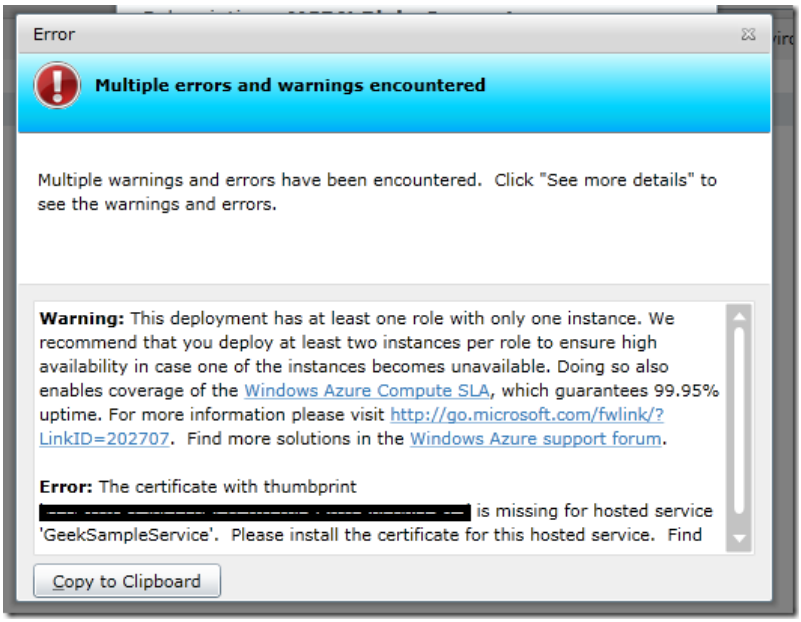


Figura 2.31.- Ventana principal del portal

Para conseguir el pfx sólo será necesario exportarlo desde el certificado. Esta operación de forma sencilla se puede realizar seleccionando la opción "view" de la ventana de configuración de la administración remota que se mostraba anteriormente.

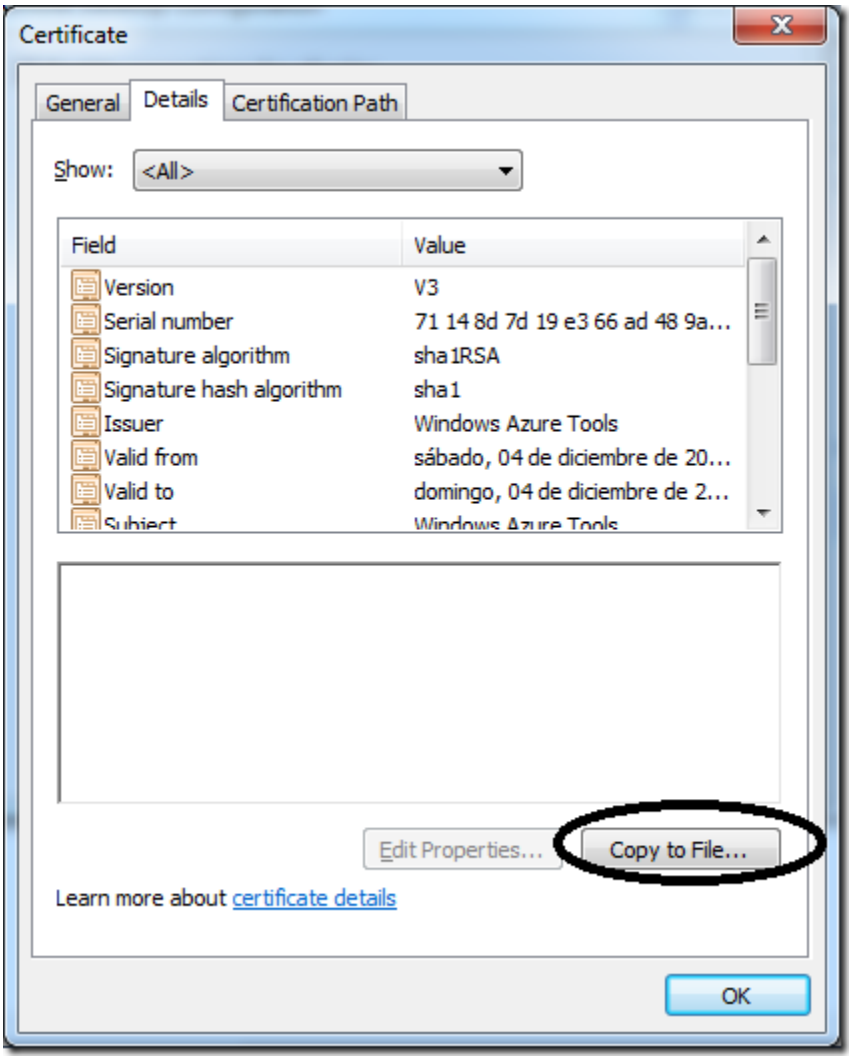


Figura 2.32.- Ventana principal del portal

Una vez se encuentre disponible el fichero pfx del certificado exportado en el equipo, sólo será necesario asociarlo al servicio dónde se vaya a desplegar la aplicación.

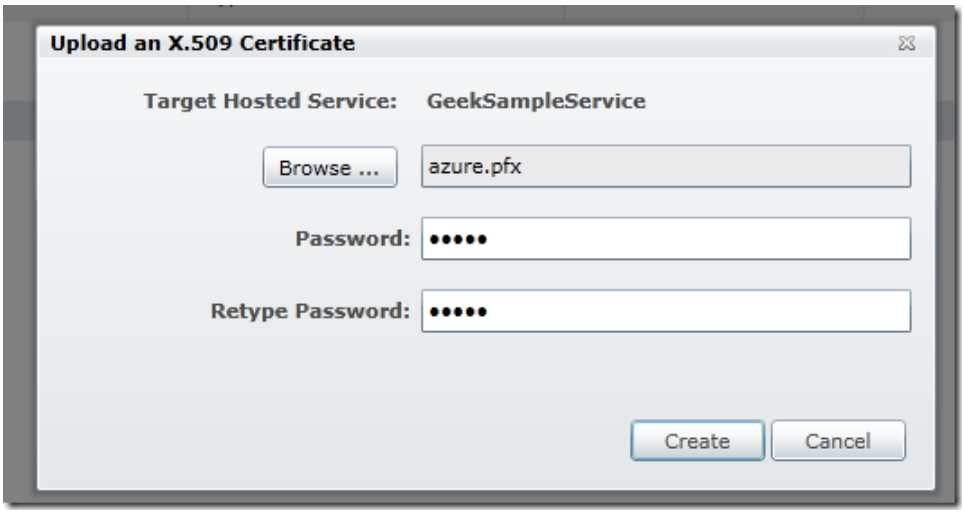


Figura 2.33.- Ventana principal del portal

Una vez desplegada la aplicación estará disponible la opción “Connect”, que permitirá conectarse de forma remota a la instancia que se elija, del mismo modo que se podría realizar la conexión a un servidor on-premise.

Figura 2.34.- Ventana principal del portal

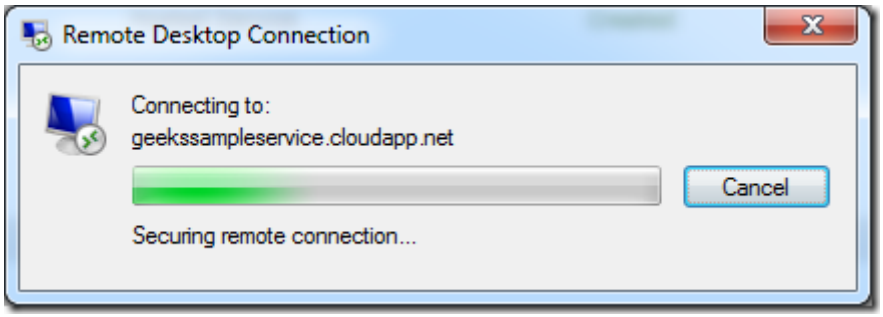


Figura 2.35.- Ventana principal del portal

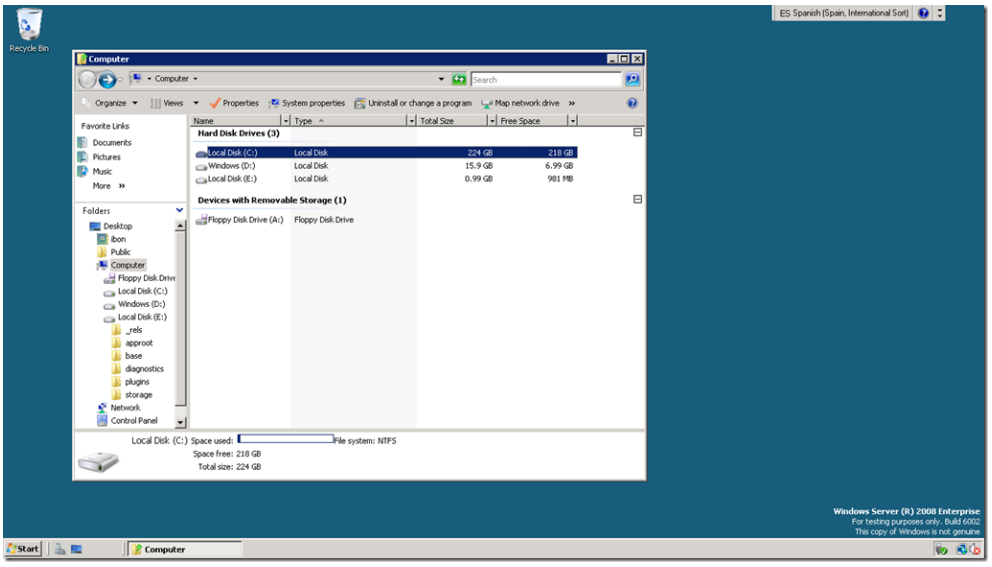


Figura 2.36.- Ventana principal del portal

10.- WORKER ROLES

La arquitectura de un servicio alojado en Windows Azure se basa en componentes auto-contenidos desarrollados típicamente con código .NET. Estos componentes son conocidos en Windows Azure como roles.

Una aplicación alojada en Windows Azure se implementa como la composición de uno o más roles. Estas aplicaciones pueden ejecutar una o más instancias de cada uno de estos roles. Este detalle se define mediante simples archivos de configuración.

Existen dos tipos de roles en Windows Azure:

- Web rol: Un 'web rol' es una aplicación basada en web accesible mediante HTTP o HTTPS. Un web rol es alojado en un entorno de ejecución que soporta un subconjunto bastante amplio de ASP.NET y Windows Comunication Foundation.
- Worker rol: Un 'worker role' es un proceso que corre en segundo plano. Sería el equivalente a un servicio de Windows en la plaforma Windows Azure. Un worker rol se puede comunicar con los servicios de almacenamiento y de colas de Windows Azure, incluso puede comunicarse directamente con otros roles.

Windows Azure impone ciertas restricciones en tiempo de ejecución a lo que un rol puede hacer. Para ello utiliza la combinación de políticas de acceso a código (CAS) de .NET y políticas de seguridad de Windows.

Todos los tipos de roles puede establecer conexiones de salida hacia recursos en Internet usando HTTP o HTTPS y usando TCP/IP sockets, y atender peticiones entrantes y solamente sobre HTTP o HTTPS.

Todos los tipos de roles tienen acceso a ciertos servicios que la plataforma de ejecución de Windows Azure expone mediante las librerías del SDK de Windows Azure:

- Acceso al almacenamiento privado del rol. ¡Atención!, no se debe confundir con los servicios de almacenamiento de Windows Azure. Se trata de almacenamiento local que se utiliza típicamente como cache. No se puede confiar en que este tipo de almacenamiento sea persistente en el tiempo y está bastante limitado en capacidad.
- Los servicios para traceo (tracing) y diagnóstico de Windows Azure.
- Servicios que permiten informar al Fabric Controler del estado de la aplicación.

10.1.- Cómo crear un Worker Role

Al crear un nuevo proyecto aparecerá un asistente que permite añadir roles al proyecto de Windows Azure. Pueden añadirse tantos roles de tipo ASP.NET Worker Role o Worker Role como necesite la aplicación a desarrollar, tal y como se puede ver en la siguiente figura.

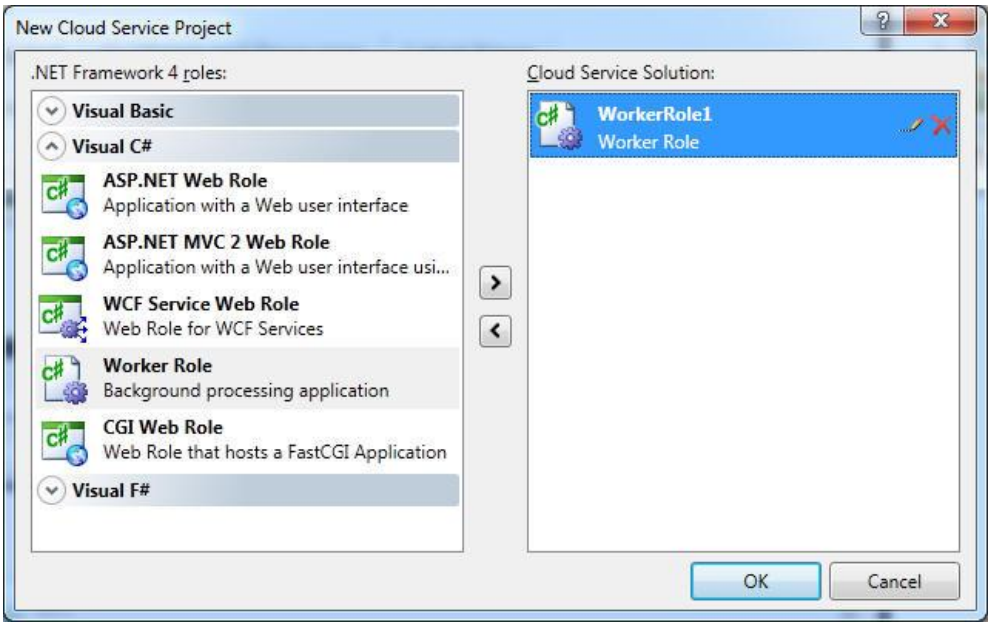


Figura 2.37.- Crear nuevo proyecto

Una vez creado el proyecto, podremos ver el código fuente generado y ver el comportamiento básico es un Worker Role. Dentro del bucle while se incluiría la lógica necesaria para realizar las tareas que se requieran realizar de forma periódica.

```
public override void Run()
        {
            // This is a sample worker implementation. Replace with your logic.
            Trace.WriteLine("WorkerRole1 entry point called", "Information");

            while (true)
            {
                Thread.Sleep(10000);
                Trace.WriteLine("Working", "Information");
            }
        }
```

10.2.- Endpoints

Los worker roles, a diferencia de los web roles, permiten tener tantos "endpoints" como necesite y atender peticiones entrantes y salientes tanto en el protocolo HTTP, HTTPS como TCP/IP.

En la configuración del rol se pueden establecer los protocolos y puertos dónde el rol podrá atender peticiones, indicando también el tipo de endpoint; de entrada o interno.

Un endpoint de tipo "entrada" permite recibir peticiones desde clientes que están fuera de Windows Azure, mientras que los endpoints internos sólo pueden emplearse para comunicaciones dentro de Windows Azure. Este segundo caso se utiliza típicamente para poder comunicar roles entre sí.

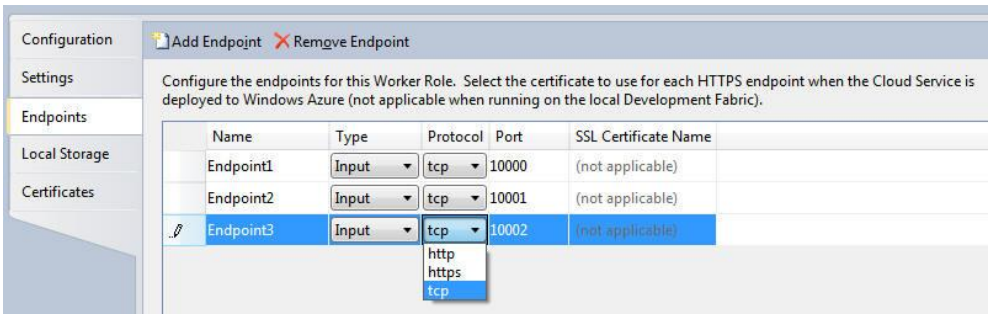


Figura 2.38.- Configuración Worker Role

10.3.- Comunicación entre Web y Worker roles

Una aplicación de Windows Azure puede estar compuesta por uno o más roles. Los roles pueden funcionar de manera independiente pero lo más habituales es que es mayor o menor medida sea necesario comunicar los roles que conforman una aplicación entre sí.

- Web rol: Un 'web rol' es una aplicación basada en web accesible mediante HTTP o HTTPS. Un web rol es alojado en un entorno de ejecución que soporta un subconjunto bastante amplio de ASP.NET y Windows Comunication Foundation.
- Worker rol: Un 'worker role' es un proceso que corre en segundo plano. Sería el equivalente a un servicio de Windows en la plaforma Windows Azure. Un worker rol se puede comunicar con los servicios de almacenamiento y de colas de Windows Azure, incluso puede comunicarse directamente con otros roles.

10.4.- Tipos de conexiones

Los Web Roles únicamente pueden atender peticiones a través del protocolo HTTP o HTTPS.

Un Web Role sólo puede disponer de un único endpoint HTTP interno que puede emplearse para comunicaciones dentro de Windows Azure y de dos endpoints para recibir conexiones entradas (uno HTTP y otro HTTPS), conexiones desde cliente que residen fuera de Windows Azure.

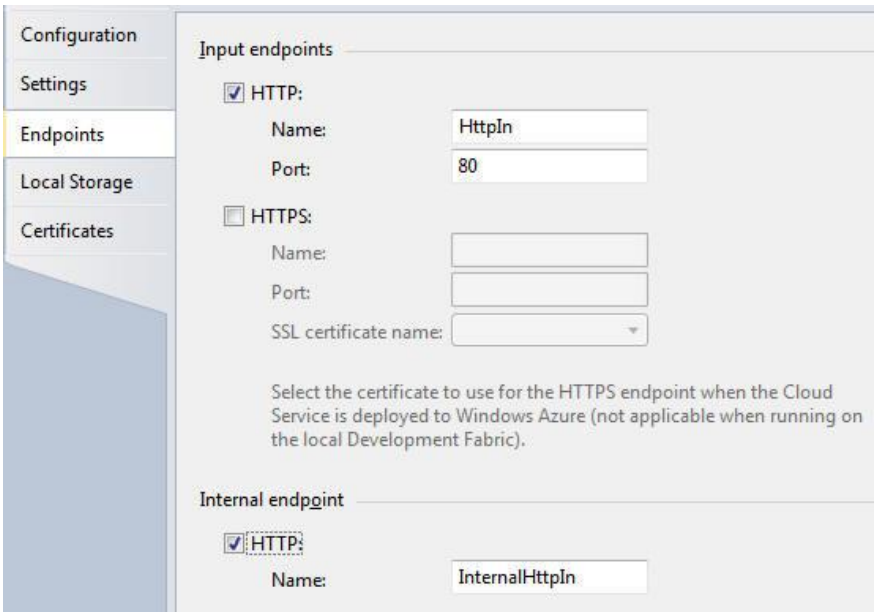


Figura 2.39.- Configuración de un Web Role

Los Worker Roles, a diferencia de los web roles, permiten tener tantos "endpoints" como necesite y atender peticiones entrantes y salientes tanto en el protocolo HTTP, HTTPS como TCP/IP.

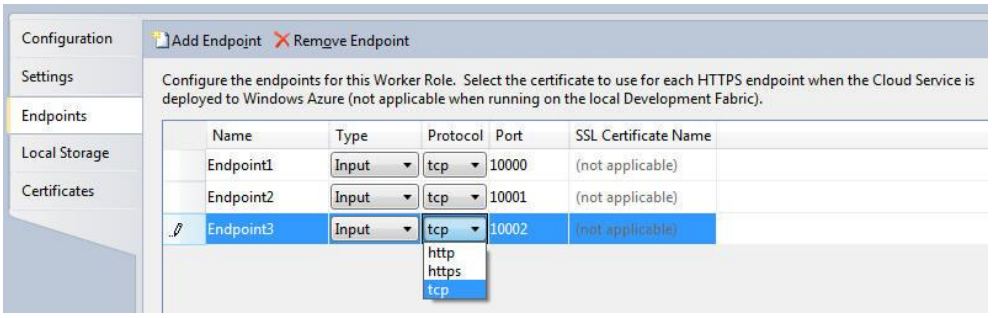


Figura 2.40.- Configuración de un Worker Role Role

10.5.- Comunicación asíncrona

Una primera alternativa a la hora de comunicar dos roles entre sí, independientemente del tipo de rol, es la comunicación asíncrona. La comunicación asíncrona es un sistema comúnmente utilizado en muchas aplicaciones hoy en día, por ejemplo, para cubrir escenarios dónde la carga de proceso puede ser muy grande y poder ofrecer un alto grado de escalabilidad.

El servicio de colas de Windows Azure proporciona un mecanismo fiable y persistente para la comunicación asíncrona entre aplicaciones de Windows Azure. El API de colas de Windows Azure, construido también sobre REST, está basado en dos tipos de abstracciones: colas y mensajes.

Un escenario habitual podría ser disponer de un Web Role que sea capaz de recibir peticiones entrantes y que tenga un requisito muy alto en cuanto a la escalabilidad. En este caso, puede interesar que el Web Role reciba las peticiones y tan pronto las reciba la incluya en una cola de Windows Azure Storage.

```
//Añadimos el mensaje a la cola

_Queue.AddMessage(new CloudQueueMessage(GetNumberOfPhotos()));
```

Un Worker Role podría encargarse de leer de forma periódica la cola, leer los mensajes que en ella encuentre y realizar la lógica necesaria con la información recibida.

```
//Vemos si hay un mensaje en la cola
//Si lo hay leemos el número de fotos
CloudQueueMessage message =  Queue.GetMessage();

if (message != null)

{
     int numberOfPhotos = int.Parse(message.AsString);

     //Procesamos el mensaje

     ...

     //Borramos el mensaje

      Queue.DeleteMessage(message);
}
```

10.6.- Comunicación directa

La comunicación asíncrona es una primera alternativa para comunicar roles, aunque no la única. Como se ha visto anteriormente los roles puede exponer endpoints internos que puede ser utilizados para realizar comunicaciones entre roles de la aplicación que residen en Windows Azure.

Si se emplean los endpoints internos, la comunicación puede hacerse de manera directa.

A través del API que proporciona el SDK, cualquier roles es capaz de descubrir de forma dinámica la información sobre el resto de roles que componen la aplicación, descubrir sus endpoints y se capaz de comunicarse con ellos una vez tiene la URI dónde se encuentran a la escucha. Lógicamente, tanto los roles como los endpoints disponen de un nombre, nombres con los cuáles un determinado rol puede descubrir la URI del rol con el que desea comunicarse.

```
   foreach (var roles in RoleEnvironment.Roles)
                    {
                        var instances =
RoleEnvironment.Roles[roles.Key].Instances;

                        foreach (var instance in instances)
                        {
                            foreach (var endpoint in
instance.InstanceEndpoints)
                                Console.Write(endpoint.Value.IPEndpoint);
                        }
                    }
```

11.- CONFIGURACIÓN DE APLICACIONES WINDOWS AZURE

Todo desarrollador de .NET sabe que el lugar en donde debe ponerse la configuración de las aplicaciones es en un archivo app.config o web.config. Este mecanismo funciona perfectamente en Windows Azure siempre y cuando no se necesite cambiar los valores de los parámetros de configuración.

El motivo es que Windows Azure no permite cambiar la configuración almacenada en un archivo .config una vez lo se ha desplegado a la nube. La única opción para reconfigurar la aplicación sería volverla a desplegar. Evidentemente esto no es operativo y hace que este mecanismo de configuración tradicional no sea todo lo útil que debiera en aplicaciones para Azure.

Así, Windows Azure define un mecanismo propio de configuración basado en dos tipos de archivos:

11.1.- Archivo de definición del servicio

El archivo de definición del servicio (.csdef), especifica los 'endpoints' del servicio, y establece la declaración de los parámetros de configuración del mismo. Este archivo no puede ser cambiado una vez el servicio ha sido desplegado. Este es el aspecto típico de este archivo:

```
   <?xml version="1.0" encoding="utf-8"?>
   <ServiceDefinition name="CloudService"
   xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDef
inition">
       <WorkerRole name="WorkerRole" enableNativeCodeExecution="false">
       <ConfigurationSettings>
              <Setting name="message"/>
       </ConfigurationSettings>
       </WorkerRole>
   </ServiceDefinition>
```

11.2.- Archivo de configuración del servicio

Del ejemplo anterior se deduce que el archivo de definición anterior declara un ajuste o parámetro cuyo nombre es message. Se puede establecer el valor de este parámetro en el archivo de configuración del servicio (.cscfg).

El archivo de configuración del servicio, como se puede deducir de su nombre, permite especificar las opciones de configuración para uno o más roles. Este archivo puede ser dinámicamente modificado una vez la aplicación está desplegada en Windows Azure. A continuación se puede ver el aspecto de un archivo .cscfg:

```
   <?xml version="1.0"?>
   <ServiceConfiguration serviceName="CloudService"
   xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceCon
figuration">
       <Role name="WorkerRole1">
              <Instances count="1" />
              <ConfigurationSettings>
                     <Setting name="message" value="Hola, Windows
Azure" />
              </ConfigurationSettings>
       </Role>
   </ServiceConfiguration>
```

En este archivo .cscfg, se puede ver el valor de configuración que se ha dado para el parámetro del servicio message que se ha declarado en el archivo de definición del servicio.

Una vez está establecida la configuración del servicio el siguiente paso es leerla. Para ello, el SDK de Windows Azure cuenta con la clase RoleEnviroment. Esta clase nos proporcionar un método, GetConfigurationSettingValue que nos permite pasarle el nombre del parámetro de configuración a leer (message en el ejemplo) y devuelve su valor como una cadena.

```
   if (RoleEnviroment.IsAvilable)
   {
       string message =
RoleEnvironment.GetConfigurationSettingValue("message");
   }
```

Nota: Debemos asegurarnos de que la aplicación está ejecutándose dentro del Fabric para poder invocar a los métodos de RoleEnviroment. Sino recibiríamos una excepción.

Para más comodidad también es posible, desde las propiedades de rol, usar el editor de propiedades que proporcionan la extensiones de Azure para Visual Studio.

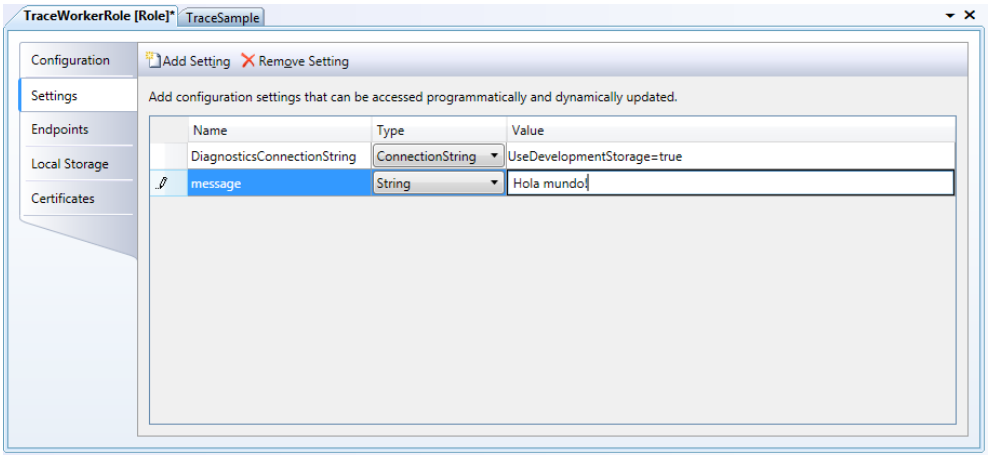


Figura 2.41.- Settings de un rol

12.- VERSIONES DE APLICACIONES WINDOWS AZURE

En toda aplicación uno de los puntos claves a tener en cuenta es todo lo relacionado con el despliegue de la aplicación, acción que no resulta complicada con Windows Azure, pero incluso más importante que el despliegue es todo lo relacionado con el proceso de actualización y versiones de aplicaciones.

12.1.- Entornos de producción y staging

Al realizar el despliegue de una aplicación en servicio de Windows Azure se puede elegir si se desea publicar la aplicación en preproducción o producción.

Ambos entornos son exactamente iguales siendo la única diferencia que si el despliegue se realizar en el entorno de preproducción Windows Azure proporcionará una URL temporal para probar la aplicación. Si en entorno se realiza en el entorno de producción, la URL que se asignará será la establecida al crear el servicio de Azure.

No es necesario desplegar primero la aplicación en el entorno de preproducción, se puede desplegar directamente en el entorno de producción. Tener ambos entornos posibilita poder tener dos versiones diferentes de la aplicación publicada de forma simultánea, cada una publicada en una URI diferente.

12.2.- Actualización de preproducción a producción (VIP-Swap)

Si el despliegue se realiza sobre el entorno de preproducción pasarlo a entorno de producción es una tarea que puede realizarse de forma inmediata, únicamente pulsando sobre el botón que se encuentra entre los dos entornos.

Figura 2.42.- Barra de herramientas

Este tipo de actualización es inmediata, ya que Windows Azure realmente no migra la aplicación de entorno, simplemente cambia la redirección de las URI actualizando el DNS, para que la URI de entorno de producción apunta a las máquinas dónde está el despliegue de preproducción. La URI de entorno de preproducción dejará de existir.

Este tipo de actualización permite que la aplicación esté siempre disponible y que no haya ningún momento en el cuál no ofrece servicio.

Como punto negativo de este tipo de actualización, es que no se puede cambiar la configuración entre el entorno de preproducción y producción. Por ejemplo, no puede cambiarse el número de instancias desplegadas. Si en el entorno de producción se tiene una instancia de la aplicación, al desplegar a producción sólo se tendrá esa única instancia.

12.3.- Actualización de un entorno de forma directa (Upgrade)

La segunda opción a la hora de actualizar un despliegue es la actualización directa de entorno. A través de la opción "Upgrade" es posible actualizar el despliegue. Permite actualizarse tanto el entorno de preproducción como en entorno de producción.

En este tipo de actualización es importante conocer el concepto de "upgrade domains". Los upgrade domains son agrupaciones lógicas que se emplean en el proceso de actualización. Las actualizaciones se realizan teniendo en cuenta estas agrupaciones; primero se actualiza un upgrade domain, luego otro, luego etc...y no se pasa de uno a otro hasta que el anterior se ha actualizado correctamente.

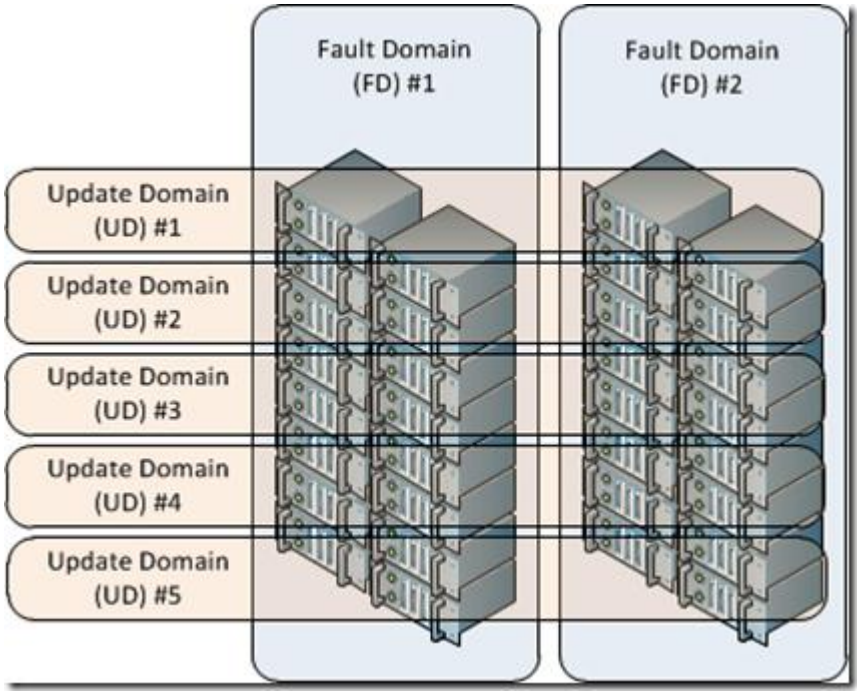


Figura 2.43.- Arquitectura de servidores

Los "upgrade domains" evitan paradas de servicio siempre y cuando el servicio se encuentre desplegado en más de un dominio y también sirven como medida de seguridad ante un error en el proceso de actualización; Si falla la actualización de un dominio no se pasa al siguiente.

Por defecto su valor es 5, valor que se puede modificar en el fichero de configuración CSDEF.

Entre las opciones de actualización se permite definir si el proceso de actualización será automático o manual. Si el proceso es manual deberá ser el usuario el que decida ir actualizando dominio a dominio. Si el proceso es automático la actualización se realizará de forma secuencia.

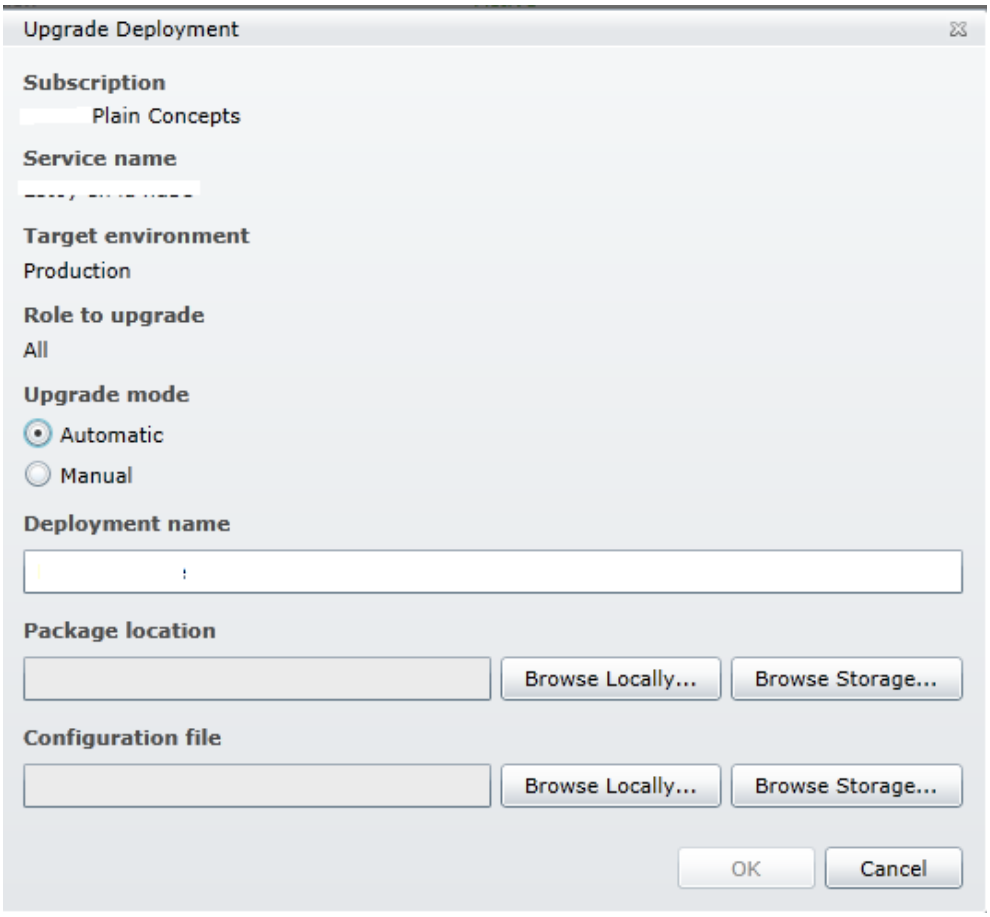


Figura 2.44.- Opción de actualización

Este tipo de actualización permite actualizar todos los roles de aplicación o solamente aquellos que interesen al usuario.

12.4.- Actualización de la configuración (Update)

Otro tipo de actualización posible es la actualización de la configuración del servicio. A través de la opción "Configure" puede cambiarse los parámetros de configuración del fichero cscfg de la aplicación.

Por ejemplo, cambiando este fichero es posible establecer el número de instancias del servicio que se quiere desplegar.

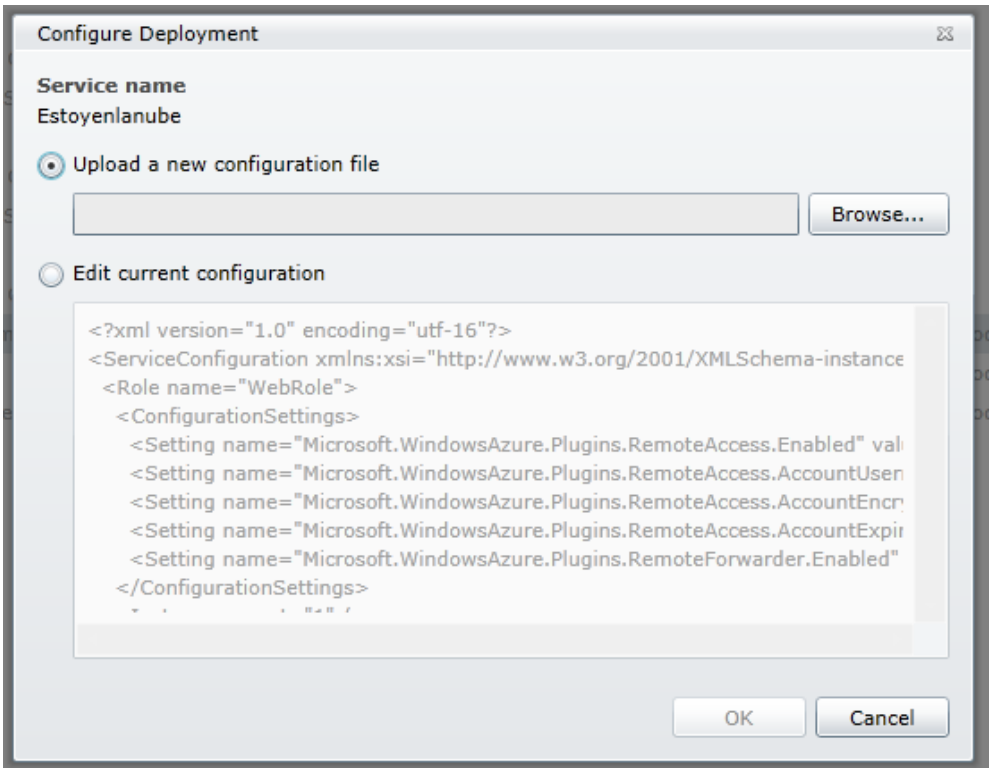


Figura 2.45.- Actualización de la configuración

En este tipo de actualización también es necesario tener en cuenta el concepto de "upgrade domain".

13.-TIP: DISPONIBILIDAD DE APLICACIONES EN EL PROCESO DE ACTUALIZACIÓN

Disponibilidad, escalabilidad, tolerancia a fallos son algunos de los requisitos con los que se encuentran habitualmente muchas de las aplicaciones que se desarrollan y como seguro que ya es todos conocido, Windows Azure puede ser un gran aliado a la hora de conseguirlos.

A continuación se incluye un pequeño "tip" relativo a la disponibilidad de las aplicaciones.

Cuando se despliega una aplicación en Windows Azure se puede elegir la versión del sistema operativo que se desea emplear. Generalmente, la alternativa más habitual es desplegar sobre la última versión disponible en ese momento…

Pero ¿Qué tipo de actualización es la adecuada? ¿Automática o Manual? ¿Tiene alguna implicación para la aplicación?

Si se selecciona la opción de actualización automática, cada vez que haya una nueva versión del sistema operativo de Azure, las instancias dónde residen la aplicación se actualizarán de forma automática. Si se actualiza, significa que durante un determinado período de tiempo, en tiempo de actualización, la aplicación no estará disponible.

El tiempo de actualización no tiene por qué ser largo pero es un período de inactividad de la aplicación que no se puede controlar y no se sabe cuándo va a ocurrir.

En las aplicaciones que por carga se necesite más de una instancia no existirá este problema porque si se actualiza una instancia la otra atenderás todas las peticiones de manera puntual.

Pero si sólo es necesario una instancia, por el tipo de aplicación que sea, tener que poner dos instancias sólo para protegerme de las actualizaciones del sistema operativo no sería práctico y además, sería el doble de caro.

La solución pasa por seleccionar el modo manual en el despliegue y elegir la versión del sistema operativo.

De esta manera, Windows Azure no realizará el proceso automático y el usuario tendrá todo el control sobre el proceso de actualización. El usuario controla el proceso y el momento de inactividad de la aplicación.

14.- DIAGNÓSTICO Y TRAZAS

En Windows Azure no se dispone de acceso a recursos locales del sistema de archivos ni al log de eventos de Windows.

Esta situación plantea la necesidad de tener un mecanismo diferente a los tradicionales para escribir las trazas de la aplicación. Algo que permita detectar y diagnosticar comportamientos anómalos de la aplicación cuando ésta se encuentra desplegada en la plataforma Windows Azure.

El SDK de Windows Azure proporcionar un 'trace listener' especialmente diseñado para ser utilizado en aplicaciones en la nube. Este listener está implementado en la clase DiagnosticsMonitorTraceListener del namespace Microsoft.WindowsAzure.Diagnostics y deriva de la clase TraceListener del framework de .Net. Trabajar con este listener es, por tanto, idéntico a trabajar con cualquier otro trace listener de .Net.

Si quisiéramos escribir trazas y diagnosticar el estado de un servicio, lo primero sería asegurarse de que en el archivo de configuración (.config) de la aplicación está configurado correctamente el trace listener de Azure:

```
<system.diagnostics>
<trace>
    <listeners>
            <add
type="Microsoft.WindowsAzure.Diagnostics.DiagnosticMonitorTraceListener,
    Microsoft.WindowsAzure.Diagnostics, Version=1.0.0.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35"
            name="AzureDiagnostics">
            <filter type="" />
            </add>
    </listeners>
</trace>
</system.diagnostics>
```

Una vez configurado correctamente el trace listener de Azure, se pueden lanzar trazas desde la aplicación usando las clases Debug y Trace del framework de .Net.

```
Trace.TraceInformation("La aplicación se está ejecutando");
...
Trace.TraceWarning("¡Se produjo un warning!");
...
Trace.TraceError("¡Se produjo un error critico!");
...
```

Como parámetro al método Trace correspondiente al nivel del error se debe pasar el mensaje que se desea que aparezca en el log. Después se podrá establecer el nivel de detalles del log en un momento dado, tanto en el Development Fabric como en el Fabric real de Windows Azure.

Nota: Hay que destacar que todas aquellos mensajes de traza que se lancen con el nivel Critical, serán enviados también como alertas y solo deberían ser utilizados en situaciones que exijan una intervención urgente por parte de un operador de la aplicación.

El resultado de las trazas se puede ver en el Development Fabric:

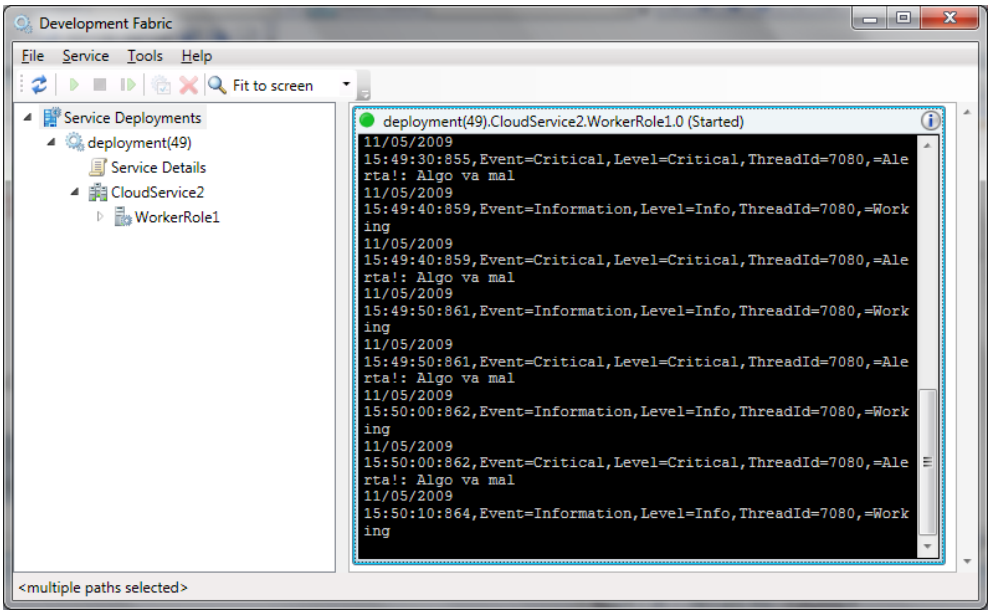


Figura 2.46.- Compute Emulator

Es posible cambiar el nivel de log en del Development Fabric desde el menú contextual del role correspondiente:

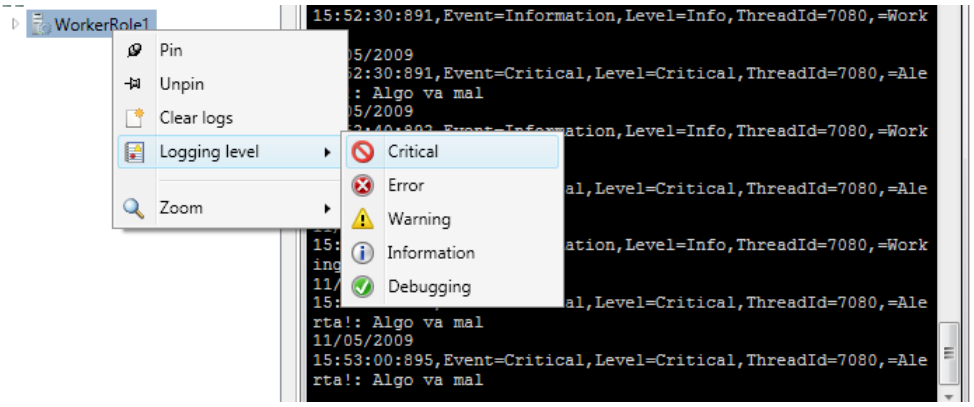


Figura 2.47.- Configuración del nivel de trazas

Una vez establecido el nivel de log a Critical, solo las trazas de este nivel aparecen, como se puede observar en la captura.

Evidentemente ver las trazas de la aplicación en el Development Fabric es muy útil durante el desarrollo y depuración de la misma, pero también se necesitará ver como se está comportando la aplicación una vez desplegada en la nube.

Pare ello existe la posibilidad de almacenar las trazas de la aplicación en el almacenamiento de Window Azure. Para ello basta con utilizar el método Start de la clase Diagnostics monitor para configurar una cadena de conexión al almacenamiento de Azure.

Además de inicializar el sistema de trazas e indicarle la cadena de conexión al Windows Azure Storage, en el siguiente ejemplo se configura la periodicidad que debe tenerse en cuenta para enviar la información al almacenamiento. En el ejemplo se configura para que las trazas se envíen cada 1 minutos.

```
      //Get the default initial configuration for Windows Azure Diagnostics
                  DiagnosticMonitorConfiguration diagConfig =
   DiagnosticMonitor.GetDefaultInitialConfiguration();

                  //Specify the scheduled transfer
```

```
                diagConfig.Logs.ScheduledTransferPeriod =
System.TimeSpan.FromMinutes(1.0);

                //Start the diagnostic monitor with this updated
configuration.
                DiagnosticMonitor.Start("DiagnosticsConnectionString",
diagConfig);
```

14.1.- Tipos de trazas

Windows Azure, además de las trazas generadas por la aplicación, es capaz de recoger múltiple información que puede servir para diagnosticar diferentes situaciones que pueden darse dentro de la plataforma y la aplicación.
Es capaz de recoger la siguiente información:

- Logs de Windows Azure.
- Logs de la aplicación.
- Logs de IIS 7.0.
- Logs generados por la infraestructura de Windows Azure.
- Logs de peticiones fallidas.
- Logs generados en el log de eventos de windows.
- Contadores de rendimiento.
- Volcados de memoria.

En el siguiente ejemplo se muestra como configurar la aplicación para recoger información sobre los contadores de rendimiento.

```
   // Get default initial configuration.
       var config = DiagnosticMonitor.GetDefaultInitialConfiguration();

       // Adding performance counters to the default diagnostic
configuration
       config.PerformanceCounters.DataSources.Add(
           new PerformanceCounterConfiguration()
           {
               CounterSpecifier = @"\Processor( Total)\% Processor
Time",
               SampleRate = TimeSpan.FromSeconds(5)
           });


config.DiagnosticInfrastructureLogs.ScheduledTransferLogLevelFilter =
LogLevel.Error;
       config.DiagnosticInfrastructureLogs.ScheduledTransferPeriod =
TimeSpan.FromMinutes(5);

       // Start the diagnostic monitor with the modified configuration.
       DiagnosticMonitor.Start("DiagnosticsConnectionString", config);
```

Aunque la configuración sobre la información a recoger puede establecer dentro de la propia aplicación, éste también puede establecer una vez desplegada la aplicación a través del API de administración de Windows Azure.

15.- CONFIGURACIÓN DE LA INFORMACIÓN DE DIAGNÓSTICO

Desde el Sdk 1.3 de Windows Azure es posible activar las trazas y la ubicación de las mismas desde la configuración del rol al que está asociada la aplicación.

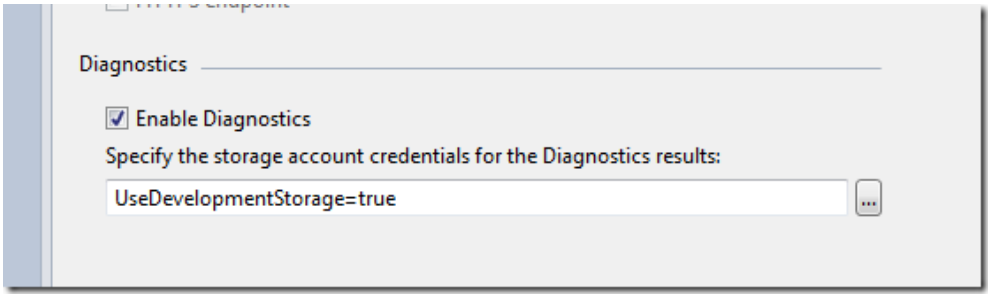


Figura 2.48.- Configuración del almacenamiento

Realmente no es algo que no se pudiese hacer anteriormente, pero lo que hace es simplificar esta acción y sobre todo, que sea fácil seguir la buena práctica de escribir las trazas en Windows Azure Storage.

Activando esta opción se crea una entrada en la sección settings llamada Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString que apunta al Storage dónde queremos que se guarden las trazas.

Antes de la salida del Sdk 1.3 la activación del sistema de trazas se hacía habitualmente en el evento "OnStart" del RoleEntryPoint, haciendo la llamada a DiagnosticMonitor.Start().

A través de las sobrecargas del método Start se podía configurar dónde debían generarse las trazas, por ejemplo al Storage, y toda la configuración que se necesite; qué información se quiere obtener (trazas, eventos, contadores…), cada cuánto tiempo se debe enviar la información al storage etc…

Desde el Sdk 1.3 la llamada al evento Start ya no es necesaria, pero aun así hay que configurar otros aspectos cómo qué trazas coger y con qué periodicidad.

A continuación se muestra el código que habría que incluir en el evento "OnStart" del RoleEntryPoint para indicar que se deben coger las trazas de la aplicación y que se deben enviar al storage cada 1 minuto.

```
 1:          private void ConfigureDiagnostics()
 2:          {
 3:              String myConnectionString =
"Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString";
 4:
 5:              CloudStorageAccount cloudStorageAccount =
CloudStorageAccount.Parse(RoleEnvironment.GetConfigurationSettingValue(m
yConnectionString));
 6:
 7:              DeploymentDiagnosticManager diagnosticManager =
 8:                                      new
DeploymentDiagnosticManager(cloudStorageAccount,
RoleEnvironment.DeploymentId);
 9:
10:
11:              RoleInstanceDiagnosticManager
roleInstanceDiagnosticManager =
12:
cloudStorageAccount.CreateRoleInstanceDiagnosticManager(
13:
RoleEnvironment.DeploymentId,
14:
RoleEnvironment.CurrentRoleInstance.Role.Name,
15:
RoleEnvironment.CurrentRoleInstance.Id);
16:
17:              DiagnosticMonitorConfiguration
diagnosticMonitorConfiguration =
roleInstanceDiagnosticManager.GetCurrentConfiguration();
18:
diagnosticMonitorConfiguration.Logs.ScheduledTransferPeriod =
TimeSpan.FromMinutes(1.0d);
19:
20:
roleInstanceDiagnosticManager.SetCurrentConfiguration
(diagnosticMonitorConfiguration);
21:          }
```

16.- DEPURANDO LA NUBE

Como se ha comentado anteriormente, una aplicación desplegada no puede ser depurada desde Visual Studio. En la mayoría de los casos será suficiente con el Compute Emulator para poder depurar la aplicación desde Visual Studio y encontrar los posibles errores que pudiera tener la aplicación.

Aunque no existe la opción de depurar de forma directa desde Visual Studio, existe la posibilidad de utilizar la característica de Visual Studio "Intellitrace" para poder realizar una depuración desde Visual Studio.

Esta características es una de las novedades de Visual Studio 2010.Se trata de una herramienta que ofrece la posibilidad de realizar depuración histórica y es una parte clave para tratar los escenarios típicos dónde es muy difícil reproducir un determinado escenario. Desde el punto de vista de Windows Azure la integración de IntelliTrace permite depurar problemas que ocurren en la nube pero que no son reproducibles en el entorno de desarrollo.

Al hacer el despliegue desde Visual Studio se puede activar la característica para aquellos roles que empleen el framework 4.0.

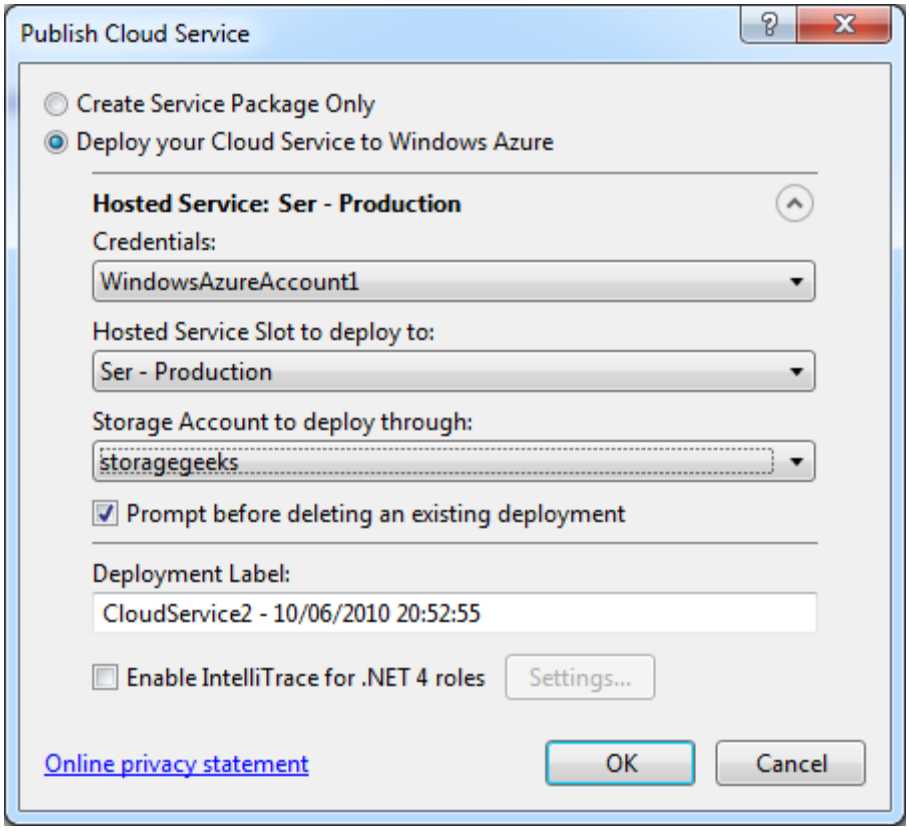


Figura 2.49.- Pantalla de publicación

Pudiendo configurar la información que se desea recoger.

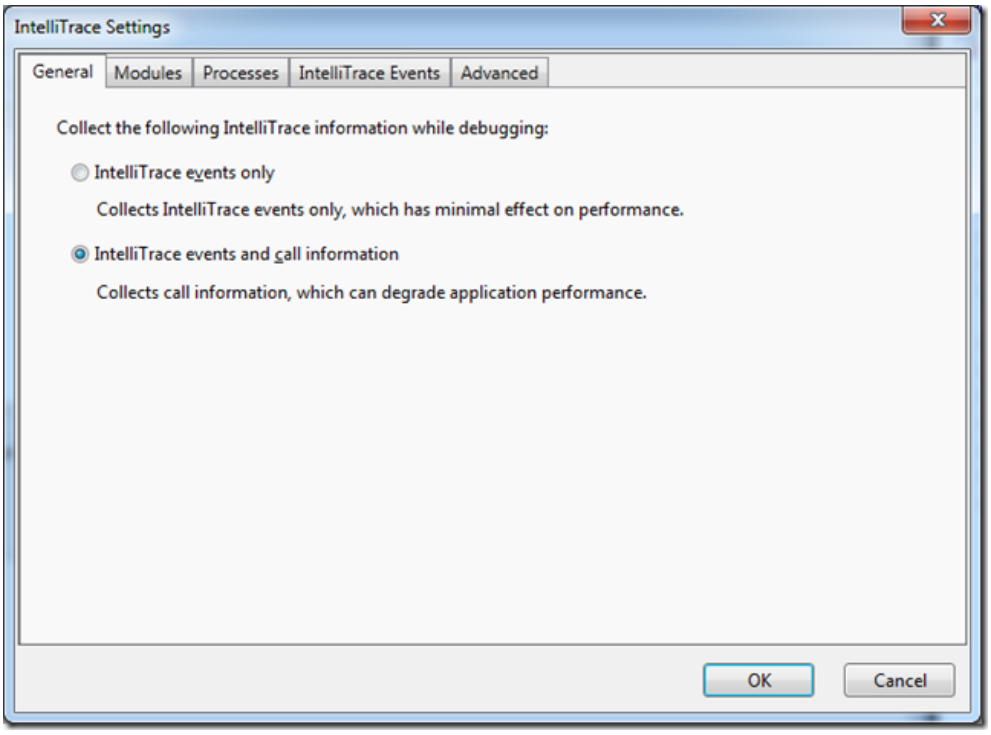


Figura 2.50.- Configuración del Intellitrace

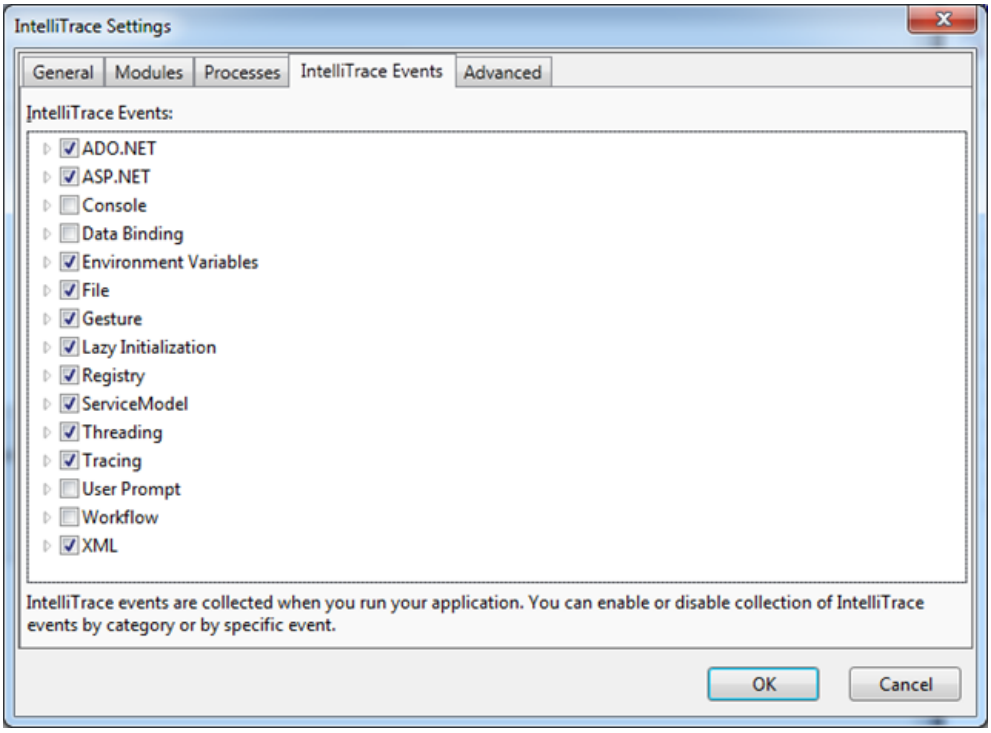


Figura 2.51.- Configuración del Intellitrace

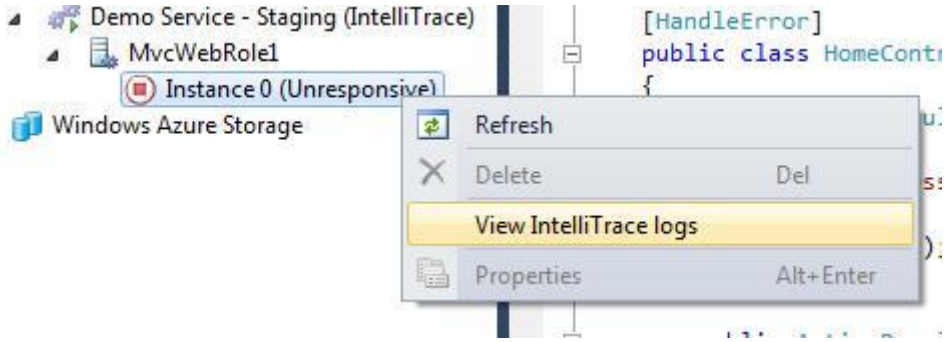


Figura 2.52.- Ver los log generados por la aplicación

17.- FULL IIS EN WINDOWS AZURE

Windows Azure ofrece el soporte completo a todas las características de Internet Information Server, característica conocida como Full IIS.

Antes de disponer de esta característica Windows Azure ya disponía de WebRoles que podían exponer endpoints HTTP.

Windows Azure para realizar esta labor no empleaba un IIS, por lo que no se podía hacer uso de todas las características que este producto ofrece; directorios virtuales, múltiples sites, exponer múltiples endpoints, endpoints TCP etc…

Windows Azure hacía uso del componente Hosted Web Core (HWC), que como su nombre indica, es en núcleo del IIS, por lo que ofrece las características más importantes de IIS, pero no todas. Seguramente en muchos escenarios con la funcionalidad ofrecida por HWC sería más que suficiente.

Por ejemplo, usando la versión completa del IIS se pueden tener varios sites dentro del mismo webrole, se pueden tener múltiples endpoint (HTTP/HTTP/TCP) sobre el mismo WebRoles, directorios y aplicaciones virtuales etc…

Desde el punto de vista de la arquitectura este diagrama muestra cómo es el comportamiento de la aplicación cuando se hace uso del HWC y cuando se hace uso de la versión completa del IIS.

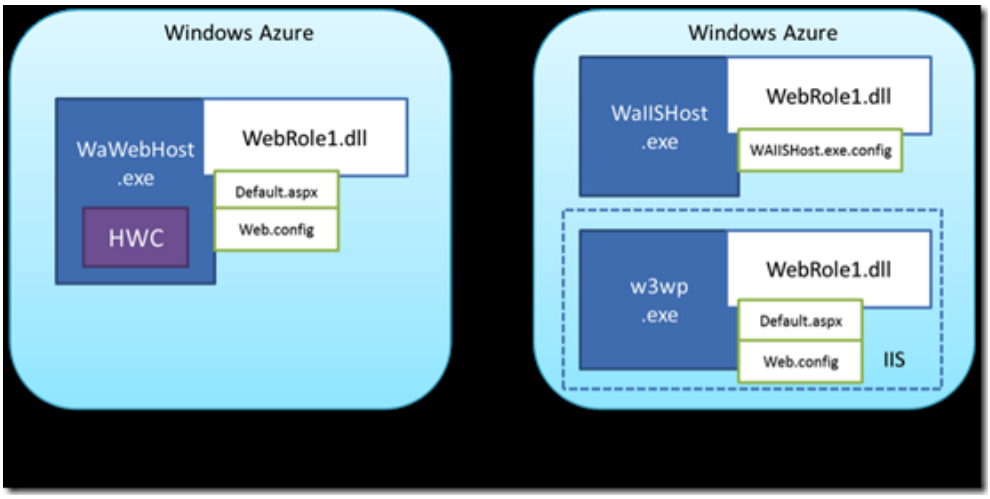


Figura 2.53.- Arquitectura WebRole

A nivel funcional, sí que habrá pequeñas diferencias, como por ejemplo, que desde el WebEntryPoint no se tendrá acceso al fichero de configuración de la aplicación (web.config) si se está en el modo Full IIS.

Por defecto, cuando se crea un nuevo WebRole se empleará la versión completa del IIS, aunque si se desea se podría configurar para que haga uso únicamente del HWC. De este modo el comportamiento sería el mismo que tenía un WebRole con la versión anterior del Sdk.

Si se crea una aplicación nueva y se añade un WebRole, una de las primeras cosas que se podrá ver es que en la configuración de los endpoint pueden configurarse tantos endpoints como se quiera, pudiendo elegir tres tipos de protocolos; HTTP, HTTPS o TCP. Con la versión anterior, un WebRole sólo podía tener dos endpoints como mucho, uno HTTP y otro HTTPS.

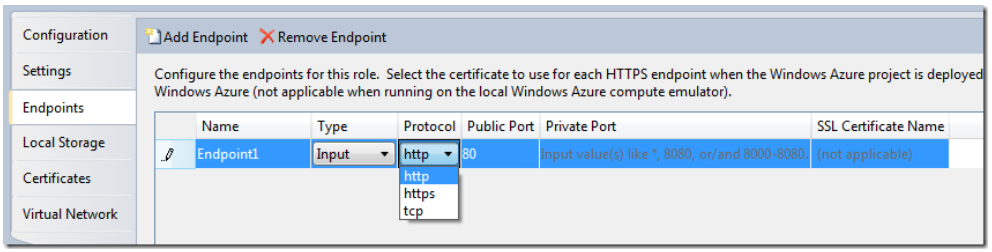


Figura 2.54.- Configuración endpoints

La siguiente peculiaridad la encontraremos en el fichero ServiceDefinition.csdef, que contendrá lo siguiente:

```
    <?xml version="1.0" encoding="utf-8"?>
    <ServiceDefinition name="SampleWebRole"
xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDefini
tion">
      <WebRole name="WebRole">
        <Sites>
          <Site name="Web">
            <Bindings>
              <Binding name="Endpoint1" endpointName="Endpoint1" />
            </Bindings>
          </Site>
        </Sites>
        <Endpoints>
          <InputEndpoint name="Endpoint1" protocol="http" port="80" />
        </Endpoints>
        <Imports>
          <Import moduleName="Diagnostics" />
        </Imports>
      </WebRole>
    </ServiceDefinition>
```

En el XML contiene la definición de los sites que contiene el WebRole y los endpoints asociados. En este caso existe un único site asociado un endpoint http que emplea el puerto 80.

El site “Web” se considera que el site por defecto y es el único en el que no debe indicarse la propiedad “physicalDirectory”. Si por ejemplo se cambia el nombre al site se tendrá que añadir dicha propiedad, para indicar la ruta física dónde se encuentra.

Las rutas son path relativos al fichero ServiceDefinition.csdef. (WebRole es el nombre que le he dado al role)

```
    <?xml version="1.0" encoding="utf-8"?>
    <ServiceDefinition name="SampleWebRole"

xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDefini
tion">
      <WebRole name="WebRole">
        <Sites>
          <Site name="MyCustomWebSite" physicalDirectory="..\WebRole">
            <Bindings>
              <Binding name="Endpoint1" endpointName="Endpoint1" />
            </Bindings>
          </Site>
        </Sites>
        <Endpoints>
          <InputEndpoint name="Endpoint1" protocol="http" port="80" />
        </Endpoints>
        <Imports>
          <Import moduleName="Diagnostics" />
        </Imports>
      </WebRole>
    </ServiceDefinition>
```

Sobre este fichero se podrán hacer las modificaciones que se consideren para añadir más sites, directorios virtuales o lo que se necesite. Los conceptos son los mismos que se manejan al trabajar con IIS.

Por ejemplo, en el XML siguiente se configura un WebRole con dos sites. Los dos sites se encuentras asociados al mismo endpoint.

Se hace uso de la propiedad "hostHeader" para determinar qué site debe responder en cada momento. En función de la URL de la que venga la petición, contestará un site u otro. Lógicamente, en este caso es el mismo.

```
<?xml version="1.0" encoding="utf-8"?>
<ServiceDefinition name="SampleWebRole"

xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDefini
tion">
  <WebRole name="WebRole">
    <Sites>
      <Site name="MyFirstCustomWebSite"
physicalDirectory="..\WebRole">
        <Bindings>
          <Binding name="Endpoint1" endpointName="Endpoint1"
hostHeader="www.myfirstsite.com"/>
        </Bindings>
      </Site>
      <Site name="MySecondCustomWebSite"
physicalDirectory="..\WebRole">
        <Bindings>
          <Binding name="Endpoint1" endpointName="Endpoint1"
hostHeader="www.mysecondsite.com"  />
        </Bindings>
      </Site>
    </Sites>
    <Endpoints>
      <InputEndpoint name="Endpoint1" protocol="http" port="80" />
    </Endpoints>
    <Imports>
      <Import moduleName="Diagnostics" />
    </Imports>
  </WebRole>
</ServiceDefinition>
```

Para poder probar en local este escenario es necesario modificar el fichero "host" del equipo donde se esté desarrollando, para que las URLs configuradas en el hostHeader apunten a la dirección 127.0.0.1.

Una vez configurado este escenario y si se pulsa F5 en Visual Studio, se mostrará el siguiente error: "HTTP 400 Bad Request".

El problema es que al haber cambiado la configuración por defecto, Visual Studio y el Compute Emulator no son capaces de encontrar el site por defecto que debe resolver la petición. Es por este motivo por el que toca realizar algún ajuste adicional.

En la herramienta del configuración del IIS (con el modo depuración de Visual Studio arrancado) se puede ver que hay 2 sites temporales como los siguientes: (uno por cada site de la aplicación)

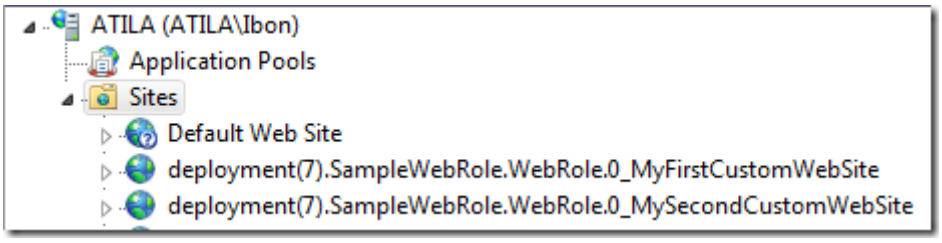


Figura 2.55.- Internet Information Server

Es necesario cambiar el binding de estos sites para que apunten a un puerto válido. Si se establece www.myfirstsite.com:8081 en el navegador ya podrá visualizarse la aplicación contestando peticiones.

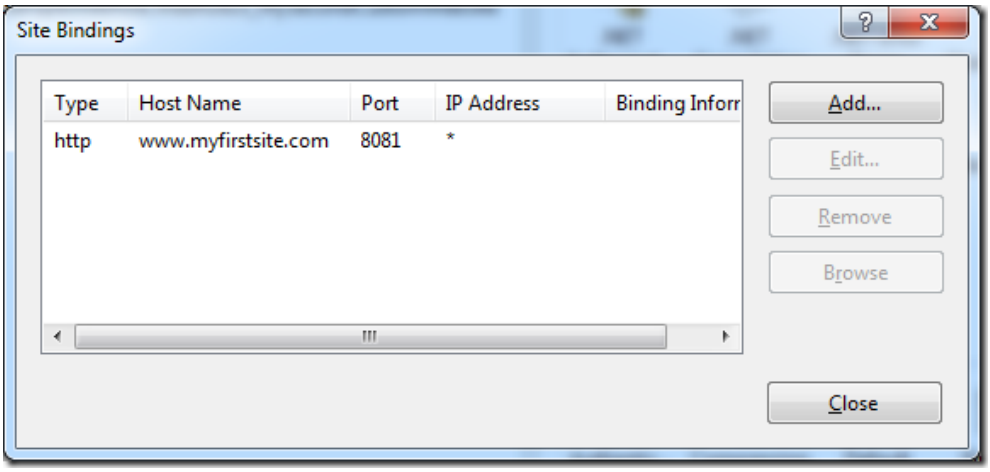


Figura 2.56.- Configuración de bindings

Adicionalmente podemos añadir un nuevo proyecto Web a la aplicación, un nuevo proyecto a la solución de tipo WebApplication. No es un nuevo WebRole, es simplemente un nuevo proyecto web llamado SecondWebApp.

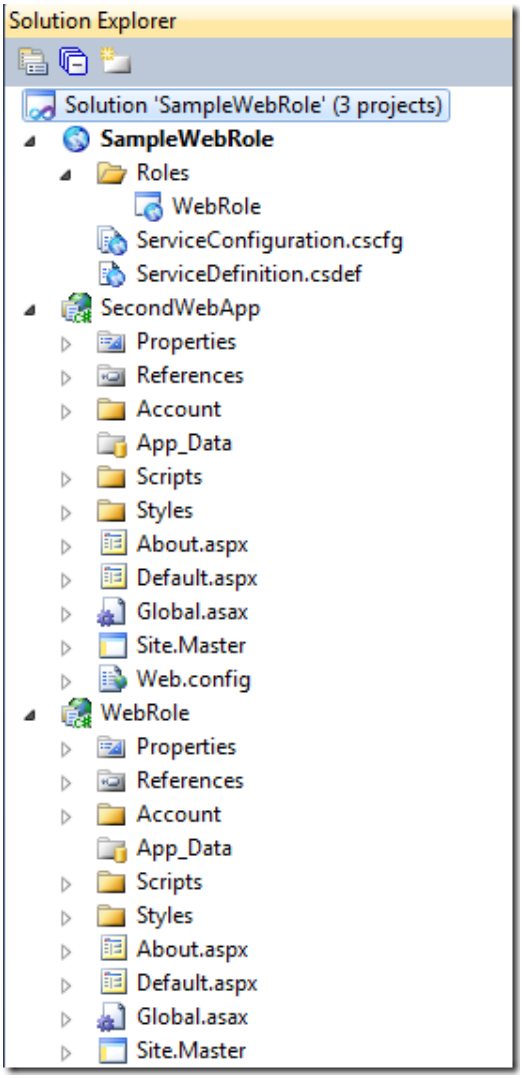


Figura 2.57.- Vista del contenido del proyecto

Se puede configurar cada site para que apunte a un proyecto diferente:

```
    <?xml version="1.0" encoding="utf-8"?>
    <ServiceDefinition name="SampleWebRole"

xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDefini
tion">
      <WebRole name="WebRole">
        <Sites>
          <Site name="MyFirstCustomWebSite"
physicalDirectory="..\SecondWebApp">
            <Bindings>
              <Binding name="Endpoint1" endpointName="Endpoint1"
hostHeader="www.myfirstsite.com"/>
            </Bindings>
          </Site>
          <Site name="MySecondCustomWebSite"
physicalDirectory="..\WebRole">
            <Bindings>
              <Binding name="Endpoint1" endpointName="Endpoint1"
hostHeader="www.mysecondsite.com"  />
            </Bindings>
          </Site>
        </Sites>
        <Endpoints>
          <InputEndpoint name="Endpoint1" protocol="http" port="80" />
        </Endpoints>
        <Imports>
          <Import moduleName="Diagnostics" />
        </Imports>
      </WebRole>
    </ServiceDefinition>
```

Y si se arranca la aplicación (F5), se puede ver que en función de la URL que se establezca en el navegador contestará una aplicación u otra.

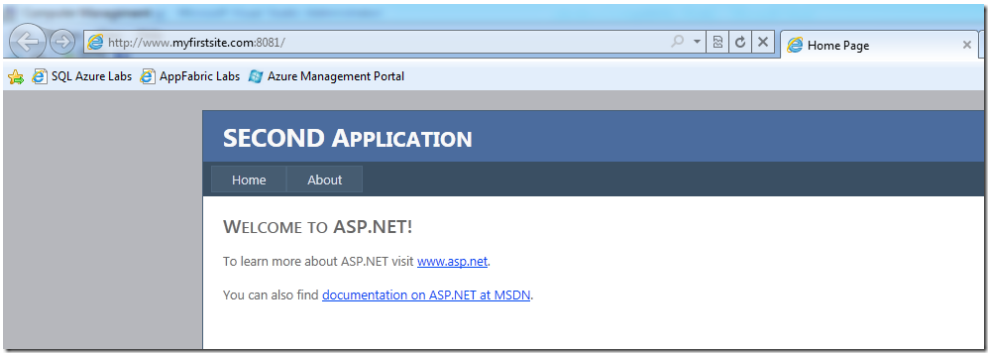


Figura 2.58.- Vista de la aplicación

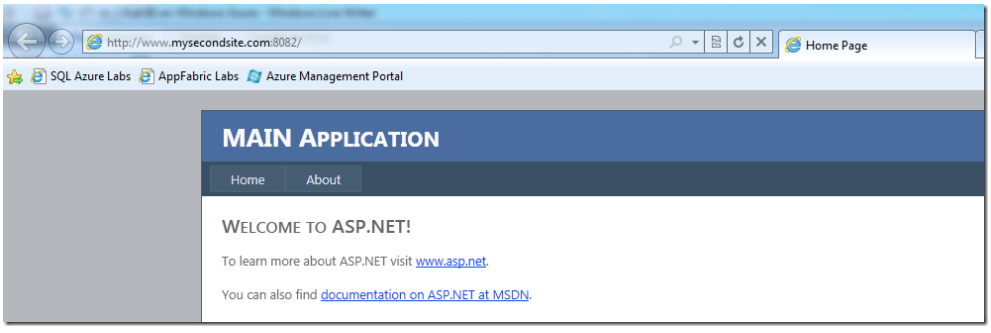


Figura 2.59.- Vista de la aplicación

Otra posibilidad es la de crear directorios o aplicaciones virtuales. En este ejemplo se ve cómo crear una aplicación virtual dentro de un site.

```
<?xml version="1.0" encoding="utf-8"?>
<ServiceDefinition name="SampleWebRole"

xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceDefini
tion">
     <WebRole name="WebRole">
       <Sites>
         <Site name="Web" physicalDirectory="..\WebRole">
           <VirtualApplication name="SecondApp"
physicalDirectory="..\SecondWebApp"></VirtualApplication>
           <Bindings>
             <Binding name="Endpoint1" endpointName="Endpoint1"/>
           </Bindings>
         </Site>
       </Sites>
       <Endpoints>
         <InputEndpoint name="Endpoint1" protocol="http" port="80" />
       </Endpoints>
       <Imports>
         <Import moduleName="Diagnostics" />
       </Imports>
     </WebRole>
</ServiceDefinition>
```

Poniendo la URL base se mostrará la aplicación principal y si se pone la URL de la aplicación virtual responderá la segunda aplicación.

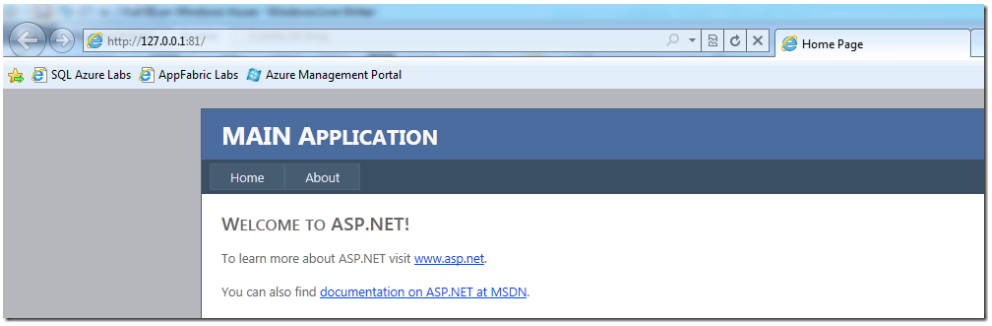


Figura 2.60.- Vista de la aplicación

En la URL se añade el nombre de la aplicación virtual.

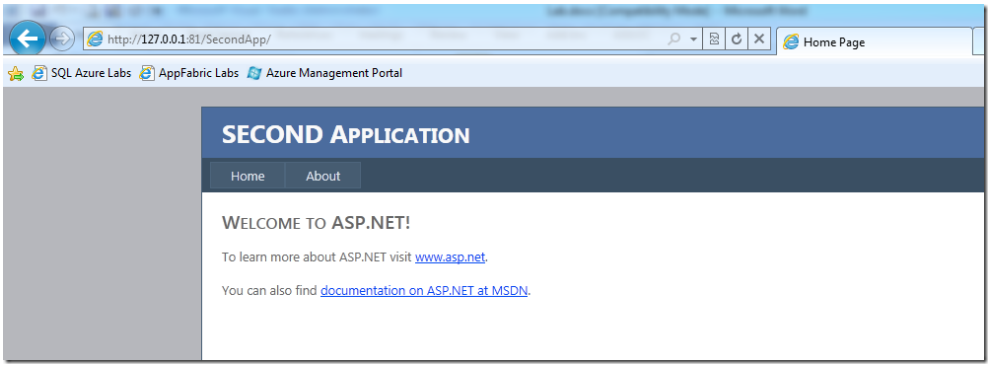


Figura 2.61.- Vista de la aplicación

18.- WINDOWS AZURE CONNECT

Windows Azure Connect permite conectar de forma sencilla aplicaciones on-premise que residen en una organización con aplicaciones que estén en Windows Azure, todo a través de una red virtual privada que permite que todas las comunicaciones entre dichos elementos se realicen de forma segura.

En algunos escenarios la nube ofrece ventajas indiscutibles sobre soluciones on-premise, pero en escenarios empresariales no siempre es posible realizar una migración completa de una aplicación, ya que éste puede tener relaciones con otras aplicaciones o componentes empresariales que no pueden o no quieren ser migrados a la nube, como puede ser un servidor de aplicaciones LOB, ERPs, bases de datos en servidores locales, etc. o el propio Active Directory corporativo.

Es aquí donde adquiere especial importancia el concepto de nube híbrida y la necesidad de disponer de una tecnología que permita la implementación de las mismas de forma segura.

Una cosa que hay que tener en cuenta, es que Azure Connect simplifica el escenario de conectar servicios de Windows Azure con recursos on-premise (servidores de ficheros, Active Directory, aplicaciones…), pero no entra en la comunicación entre servicios de Windows Azure, ya que la plataforma ya tiene mecanismos propios para comunicar dos servicios Azure, ya sea de manera síncrona o asíncrona.

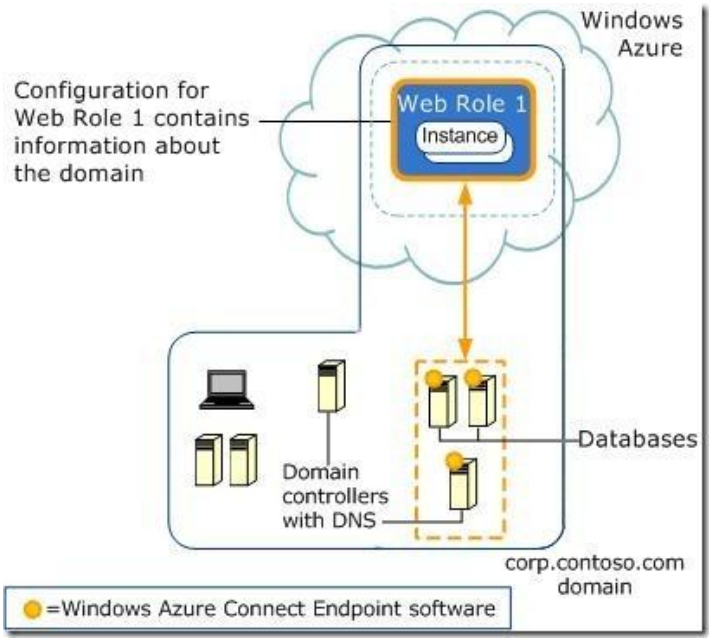


Figura 2.62.- Arquitectura

La creación y configuración de estas redes virtuales se realiza de forma muy sencilla. En cada equipo que se quiere que forme parte de la red virtual es necesario instalar el agente de connect (endpoint).

Figura 2.63.- Opción Virtual Network

Si se quiere instalar el agente de connect en un servicio que vaya a ser desplegado en Windows Azure (Web o Worker Role), únicamente es necesario obtener un token de activación desde el portal de Windows Azure e incluirlo dentro de las propiedades del rol de Windows Azure. Con estos sencillos pasos todas las instancias de los roles desplegados en Windows Azure dispondrán del agente de connect.

Figura 2.64.- Obtener token de activación

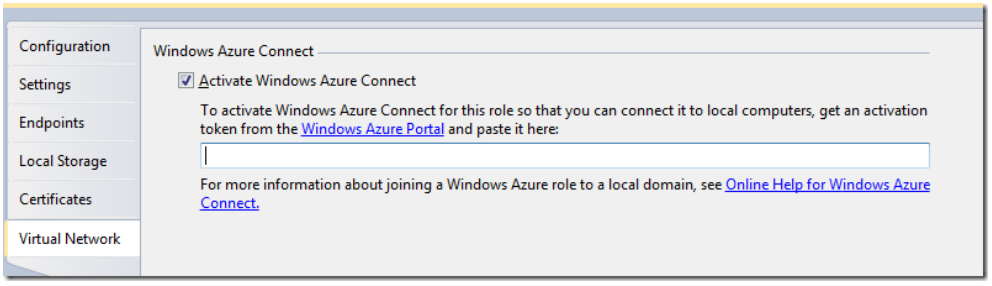


Figura 2.65.- Añadir token en las propiedades del proyecto

Si se quiere añadir un servidor on-premise o un VMRole (cómo instalar en un VMRole) se debe instalar el agente de connect manualmente.

Desde el portal de Windows Azure se puede indicar que se quiere instalar el agente de connect. Esta acción dará una URL que se tendrá que poner en el navegador para que se inicie el proceso de instalación del agente de connect.

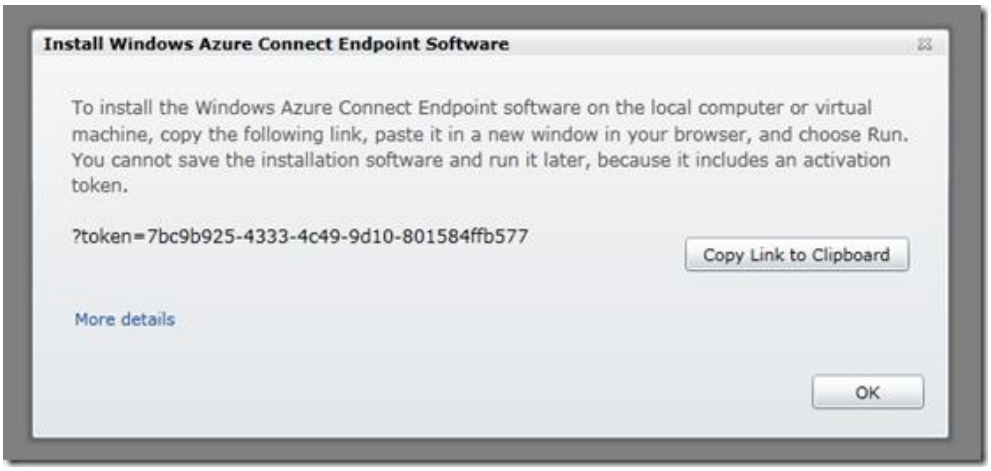


Figura 2.66.- Instalación del agente

La URL que genera es una URL única, que no puede ser usada para instalar más de un agente; Una URL, un agente.

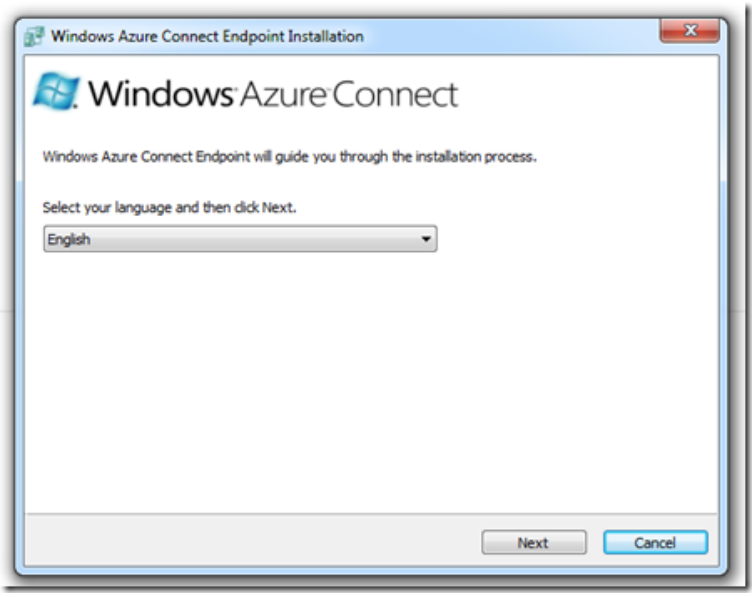


Figura 2.67.- Ventana principal del instalador del agente

El agente de connect además de la instalación por URL permitirá disponer de un instalador standalone para poder hacer distribuciones a través de mecanismos y herramientas de distribución de software. Por ejemplo, desde las políticas de Active Directory.

El agente de connect está disponible para Windows Vista, Windows 7 y Windows 2008, aunque en futuras versiones se espera disponer de mecanismos para que cualquier tipo de sistema operativo pueda formar parte de la red virtual de connect, incluido sistemas operativos no-Microsoft.

Una vez instalados los agentes, todos se verán dentro de la sección de "Virtual Network".

Figura 2.68.- Virtual Network

Desde aquí se podrán crear grupos de equipos. Se puede decir que cada grupo es como una red. Si se quiere crear una red con 3 equipos, se deberá crear un nuevo grupo y añadir dichos equipos a dicho grupo.

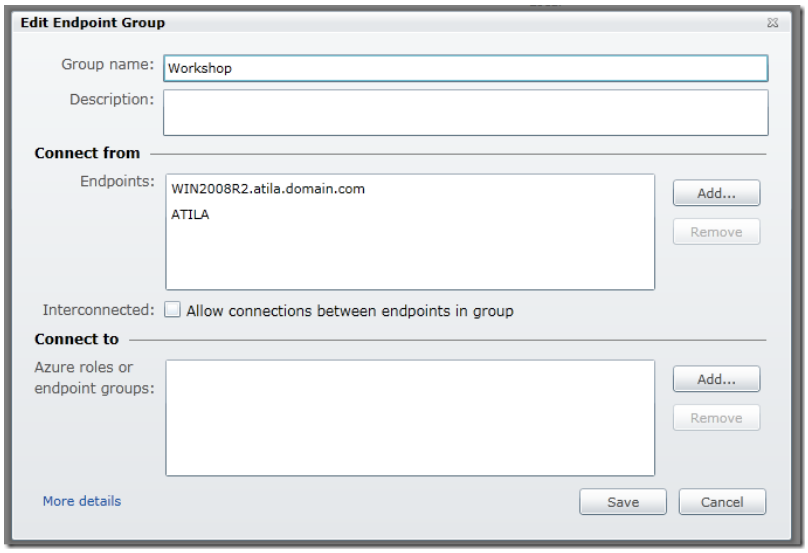


Figura 2.69.- Configuración de los grupos

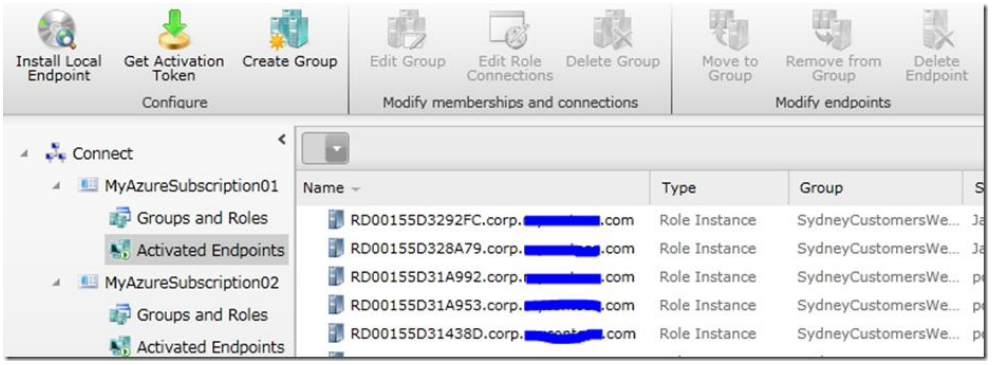


Figura 2.70.- Vista de los endpoints disponibles

Una vez creados los grupos y añadidos los equipos a los grupos, éstos formarán parte de la misma red virtual y podrán comunicarse de forma segura a través del servicio de relay.

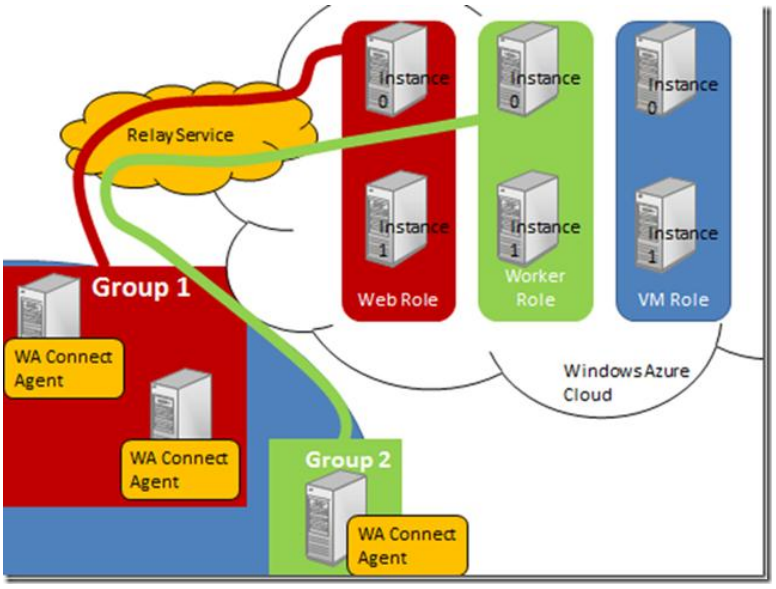


Figura 2.71.- Arquitectura de alto nivel

19.- TRAFFIC MANAGER

Traffic Manager es una herramienta que permite distribuir el tráfico que recibe la aplicación desde Internet entre dos o más servicios hosteados en Windows Azure, incluso de data centers diferentes, lógicamente, todo accesible desde la misma URL.

Con este sistema se puede tener una aplicación desplegada en dos datacenters diferentes, uno en USA y otro en Europa, y hacer que los dos sean accesibles desde la misma URL, pudiendo configurar en qué caso debe contestar uno y en qué caso tiene que contestar otro.

En esencia, la funcionalidad es un servicio de DNS, el cual el usuario podrá configurar:

- Failover: Todo el tráfico se mapea a un servicio. Si éste falla, el tráfico se mapea al servicio que hace de backup.
- Performance: El tráfico se mapea al servicio más cercano, desde el punto de vista de "saltos entre los routers".
- Round-Robin: Distribuye las peticiones entre servicios de Azure definidos en las políticas.

Para tener acceso a esta funcionalidad es necesario que solicitar la activación, acción que se puede realizar desde el portal de Windows Azure:

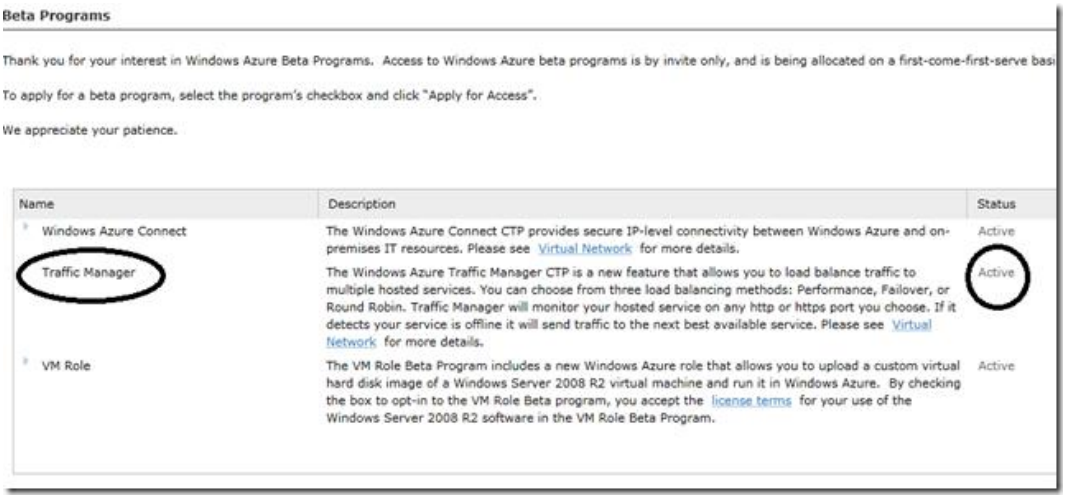


Figura 2.72.- Dar de alta Traffic Manager

Una vez activada la funcionalidad, desde el "Virtual Network" se puede acceder a la nueva funcionalidad que ofrece Traffic Manager para poder crear y configurar las políticas; elegir el tipo de balanceo, los servicios implicados etc…

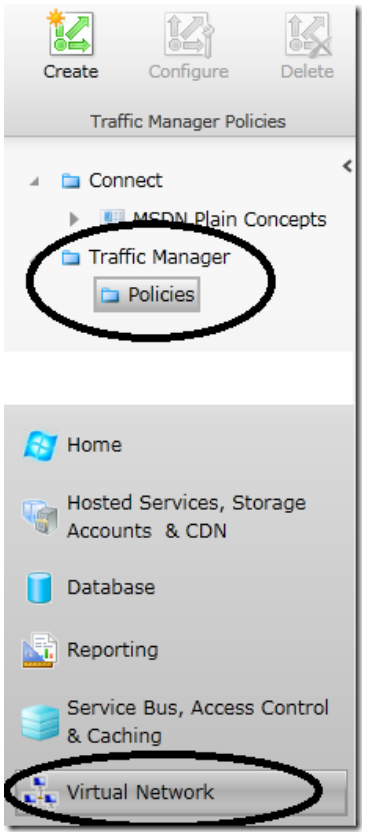


Figura 2.73.- Acceso a traffic manager

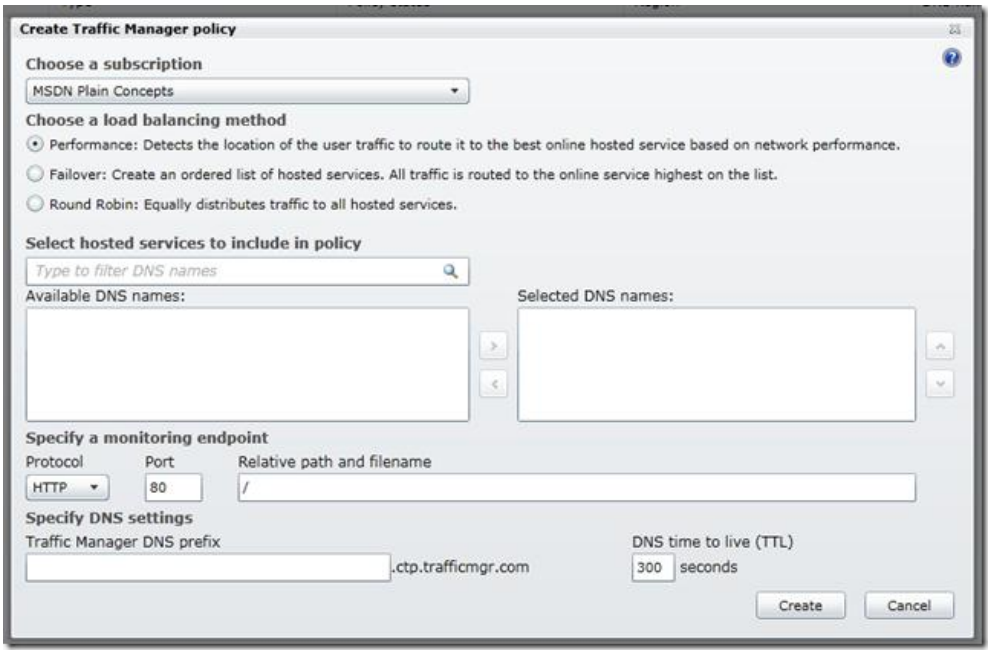


Figura 2.74.- Configuración

20.- TIP: USAR WINDOWS 2008 R2

Windows Azure da la posibilidad de desplegar las aplicaciones en roles de Azure que tenga Windows 2008 Server R2.

Por defecto los roles se despliegan en instancias de Windows 2008 Server, lo que hace que no la aplicación no pueda beneficiarse de las mejoras que ofrece la versión R2, como puede ser la posibilidad de realizar administración remota con powershell, IIS 7.5 o AppLocker.

Para que las instancias empleen Windows 2008 Server R2 sólo es necesario establecerlo en el fichero de configuración; ServiceConfiguration.cscfg

Dentro de este fichero se puede establecer el atributo osFamily a 2, forzando de esta manera a que el despliegue se haga usando la versión R2.

```
      <ServiceConfiguration serviceName="DynamicWeb"
      xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceCon
   figuration"
      osFamily="2" osVersion="*">
```

21.- EJECUTAR TAREAS ELEVADAS DURANTE EL CICLO DE VIDA DE UN ROL AZURE

Cuando se desarrolla para Windows Azure es habitual encontrarse con distintos escenarios que van desde aplicaciones completamente .NET y aplicaciones que son migraciones de aplicaciones existentes. En ese sentido uno de los dolores de cabeza a la hora de trabajar con Azure son los registros de componentes COM durante el arranque del rol de Azure. Este tipo de problema se soluciona normalmente creando una tarea en el startup del rol que desea consumir ese tipo de componentes COM.

Si por ejemplo durante el ciclo de ejecución de un rol se quiere ejecutar un proceso con elevación, es decir con permisos completos de administrador no es posible hacerlo porque el proceso que hostea la web y el worker role no está elevado, y aunque se le indique a la hora de ejecutar el proceso eso no va a funcionar.

Es por eso que es posible hacer un pequeño truco para que se puedan ejecutar proceso elevados durante nuestro ciclo de ejecución del rol, es decir en cualquier momento, así es posible registrar un componentes COM, o llamar a ejecutables del SO de manera mucho más cómoda.

Para poder llegar a esa aproximación es necesario buscar un entorno donde se puedan ejecutar las aplicaciones de manera elevada, y ese entorno es el entorno de startup del rol, así que de alguna manera lo que hay que tener es un proceso sentinel que se arranque en el startup del rol y que acepte peticiones para ejecutar procesos de manera elevada.

Pues justamente eso es lo que se muestra a continuación, utilizando WCF para abrir un pipe de comunicación entre los procesos se va a crear un servicio que escuche peticiones de otro proceso a través de un pipe para enviar un mensaje que representa una invocación de un proceso.

21.1.- Definición del servicio

Como lo que se quiere hacer es exponer un servicio de WCF a través de pipes de Windows, es necesario definir la interfaz del contrato de operaciones:

```
    [ServiceContract(Namespace =
"http://azure.plainconcepts.com/schemas/04/2011/azure/executionhost")]
    public interface IExecutionHost
    {
        [OperationContract]
        void ExecuteTask(ProcessTask host);
    }
```

Una vez que se tiene definido el contrato servicio es necesario hacer dos cosas, primero hacer la implementación del servicio, es decir el proceso sentinel que escuchará las peticiones recibidas y hará el trabajo de ejecutar esos procesos.

```
    [ServiceBehavior(InstanceContextMode = InstanceContextMode.Single)]
    public class ExecutionHostService : IExecutionHost
    {
        public void ExecuteTask(ProcessTask host)
        {
            Process process = new Process();
            process.StartInfo = host.StartInfo;
            process.Start();
        }
    }
```

Otra cosa que es necesario hacer en el proceso sentinel es hostear el servicio y ponerlo a escuchar peticiones a través del binding que se seleccione, en este caso NetNamedPipeBinding:

```
    public class ExecutionHostServiceManager
    {
        public ExecutionHostServiceManager()
        {
            service = new ExecutionHostService();
            ServiceHost host = new ServiceHost(service);

            string address =
"net.pipe://PlainConcepts/Azure/ExecutionHost";
            NetNamedPipeBinding binding = new
NetNamedPipeBinding(NetNamedPipeSecurityMode.None);
            host.AddServiceEndpoint(typeof(IExecutionHost), binding,
address);

            // Add a mex endpoint
            long maxBufferPoolSize = binding.MaxBufferPoolSize;

            int maxBufferSize = binding.MaxBufferSize;

            int maxConnections = binding.MaxConnections;

            long maxReceivedMessageSize =
                binding.MaxReceivedMessageSize;

            NetNamedPipeSecurity security = binding.Security;

            string scheme = binding.Scheme;
```

```
            XmlDictionaryReaderQuotas readerQuotas =
                binding.ReaderQuotas;

            BindingElementCollection bCollection =
binding.CreateBindingElements();

            HostNameComparisonMode hostNameComparisonMode =
                binding.HostNameComparisonMode;

            bool TransactionFlow = binding.TransactionFlow;

            TransactionProtocol transactionProtocol =
                binding.TransactionProtocol;

            EnvelopeVersion envelopeVersion =
                binding.EnvelopeVersion;

            TransferMode transferMode =
                binding.TransferMode;
            host.Open();
        }

        private ExecutionHostService service;
    }
```

Todo ello hay que ponerlo en un pequeño programa de consola que será el proceso en sí que hosteará el pipe de Windows que aceptará peticiones a través de WCF:

```
    class Program
    {
        static void Main(string[] args)
        {
            new ExecutionHostServiceManager();
            Thread.Sleep(Timeout.Infinite);
        }
    }
```

Al final de la ejecución de la clase hay un Thread.Sleep(Timeout.Infinite) que permite esperar eternamente en el proceso para que así el proceso esté disponible durante todo el ciclo de vida del rol, permitiendo ejecutar un proceso elevado en cualquier momento.

21.2.- Haciendo llamadas al servicio

Como bien es sabido para poder hacer llamadas a un servicio de WCF lo primero que es necesario hacer es generar un proxy en el cliente para hacer esas llamadas.

Lo que vamos a hacer es una clase que herede de ClientBase<T> siendo T la interfaz del contrato de operaciones del servicio.

```
    public class ExecutionHostClient : ClientBase<IExecutionHost>
    {
        static ExecutionHostClient()
        {
            string address =
"net.pipe://PlainConcepts/Azure/ExecutionHost";
            NetNamedPipeBinding binding = new
NetNamedPipeBinding(NetNamedPipeSecurityMode.None);
            binding.CloseTimeout = TimeSpan.MaxValue;
            binding.ReceiveTimeout = TimeSpan.MaxValue;
            binding.SendTimeout = TimeSpan.MaxValue;
            EndpointAddress endpoint = new EndpointAddress(address);
            client = new ExecutionHostClient(binding, endpoint);
        }

        public ExecutionHostClient(Binding binding, EndpointAddress
remoteAddress) :
            base(binding, remoteAddress)
        {
```

```
        }

        public void ExecuteTask(ProcessTask task)
        {
            Channel.ExecuteTask(task);
        }

        public static void ExecuteRemoteTask(ProcessTask task)
        {
            client.ExecuteTask(task);
        }

        private static ExecutionHostClient client;
    }
```

Es importante que el proxy se inicialice con el mismo binding que el de servidor para que las invocaciones funcionen. En este ejemplo para simplificar se tendrá una referencia estática del proxy y solamente se expondrá a través de un método estático.

21.3.- Invocando servicios

Para el ejemplo actual se puede registrar los componentes COM de una carpeta que se tenga en el worker role:

```
    public class RegisterComHelper
        {
            public RegisterComHelper()
            {

            }

            public void Register()
            {
                // hay que buscar la localizacion en el servidor de azure
de donde estan los ensamblados
                // como no sabemos dónde estan los ficheros tenemos que
buscar el modulo
                // Habitania.RegisterCom.dll que es específico para este
ejemplo
                // así nos aseguramos que estamos buscando la dll
correcta
                Process current = Process.GetCurrentProcess();
                var found = (from p in
current.Modules.Cast<ProcessModule>().ToList()
                             where p.ModuleName ==
"PlainConcepts.Azure.WorkerRoleDemo.dll"
                             select p).FirstOrDefault();

                if (found != null)
                {
                    // a partir de la locacion del modulo cargada por el
proceso
                    // somos capaces de encontrar la informacion del
directorio y buscar
                    // la carpeta dlls que contiene la lista de dlls que
queremos registar
                    string directoryLocation =
Path.GetDirectoryName(found.FileName);

                    string dllPath = Path.Combine(directoryLocation,
"V3COM30");

                    string[] files = Directory.GetFiles(dllPath);

                    foreach (var item in files)
                    {
                        if (item.EndsWith(".dll"))
                            RegisterComObject(item);
                    }

                    dllPath = Path.Combine(directoryLocation, "V3COM");
```

```
                    files = Directory.GetFiles(dllPath);

                    foreach (var item in files)
                    {
                        if (item.EndsWith(".dll"))
                            RegisterComObject(item);
                    }
                }
            }

            private void RegisterComObject(string filePath)
            {
                ProcessStartInfo info = new ProcessStartInfo();
                info.FileName = Path.Combine(

Environment.GetFolderPath(Environment.SpecialFolder.System),
                    "regsvr32.exe");
                info.Arguments = string.Format("/i {0}", filePath);
                info.UseShellExecute = false;

                ExecutionHostClient.ExecuteRemoteTask(new ProcessTask()
                {
                    StartInfo = info
                });
            }
        }
```

Este método para trabajar con Windows Azure puede ser un poco complicado de montar, pero una vez hecho se dispone de un mecanismo muy sencillo para hacer cosas más complicadas como por ejemplo ejecutar otro tipo de tareas de mantenimiento directamente desde ahí.

22.- TAREAS ADMINISTRATIVAS

A través de portal de Azure es posible hacer todas las operaciones que la plataforma permite, como crear servicios, storage, desplegarlos, configurarlos, configurar la seguridad etc…pero trabajar siempre usando el portal de Azure puede ser un trabajo pesado y poco productivo.

Por este motivo la plataforma Windows Azure ofrece un API de administración expuesto por REST que permite realizar todas las operaciones que se pueden hacern desde el portal.

Mediante este API REST podremos realizar todas las labores de administración que necesitemos, sin necesidad de tener que acceder al portal web.

Para simplificar el uso y la creación de llamadas REST al API de administración existen varias alternativas:

- Hacerte una nueva aplicación, atacando directamente al API REST.
- Usar los CmdLets de PowerShell para Azure.
- Usar la aplicación de línea de comandos csmanage.exe. Disponible con código fuente.
- Usar otras aplicaciones de terceros, comerciales o no, que ofrezcan labores administrativas.

Sea como sea, si se desean hacer labores administrativas sin usar el portal web, es necesario que se haga de forma segura, que todo el mundo no lo pueda realizar.

Las labores de administración requieren de un certificado x.509, certificado que hay que subir previamente a la cuenta de Azure, a través del portal Web. (Certificados de ejemplo para pruebas los podrías crear desde IIS).

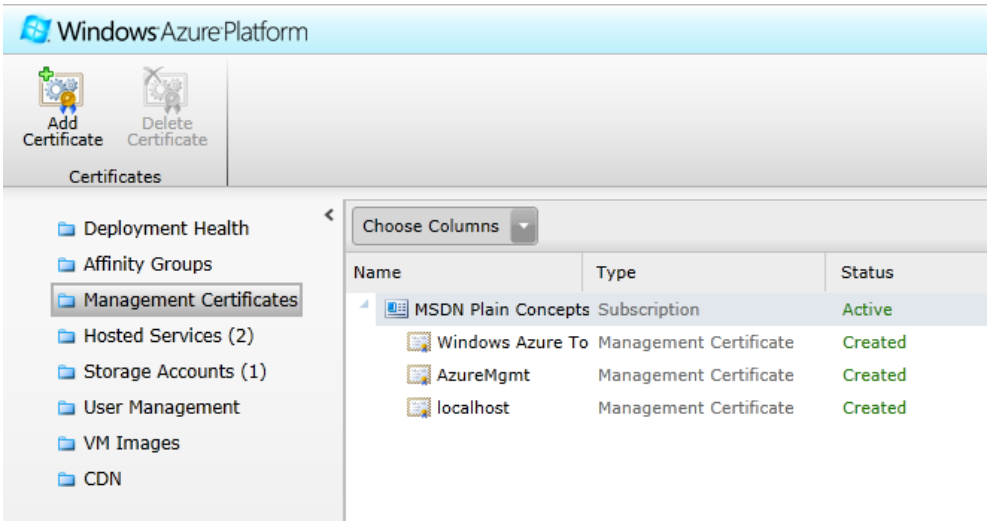


Figura 2.75.- Certificados de administración

Una vez subido el certificado, el dato importante a tener en cuenta es la huella digital, el Thumbprint. Este valor es un valor que vamos a necesitar en las aplicaciones cliente desde las cuáles queremos realizar labores administrativas.

Otro valores que también necesitaremos, es el identificador de nuestra subscripción, que también lo podemos obtener del portal.

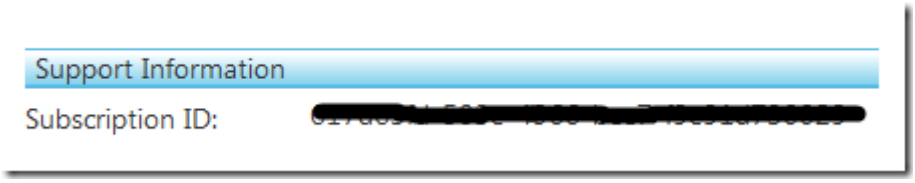


Figura 2.76.- SubscriptionID

Y una vez hecho esto, ya podemos hacer cualquier de las opciones que hemos visto antes para administrar Azure.

Por ejemplo, una cosa que podemos hacer es desplegar en Azure directamente desde Visual Studio, como veíamos en un post anterior. El desplegar directamente requería instalar el certificado x.509.

Recordad que para configurar la cuenta necesitábamos el certificado y el subcription id.

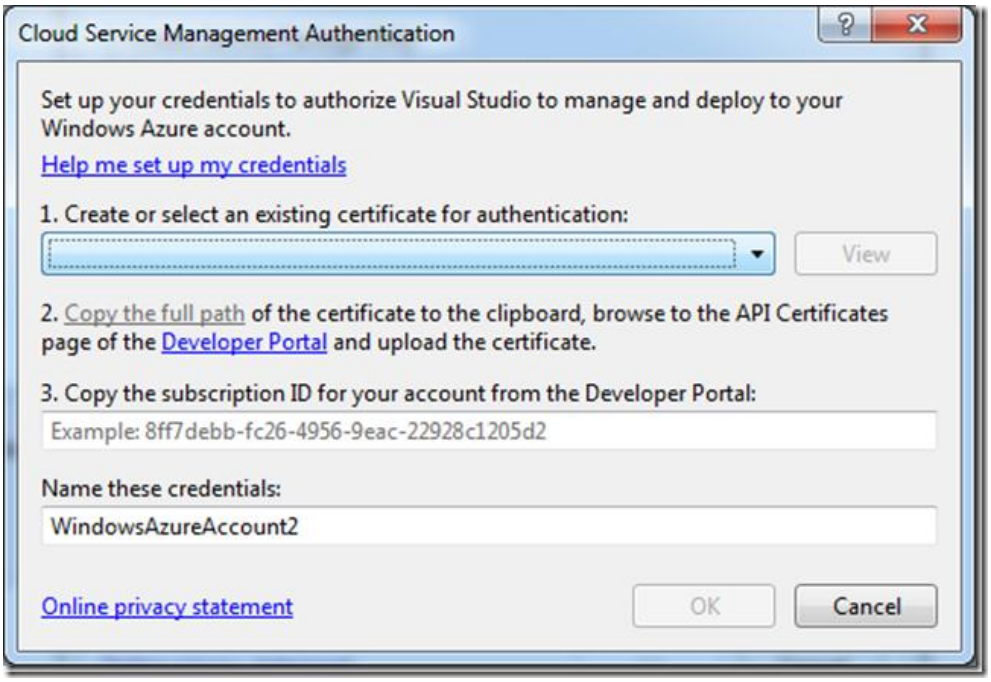


Figura 2.77.- Configuración de certificados en Visual Studio

Por ejemplo, si usáramos csmanage, la configuración la tendríamos que incluir en el fichero de configuración de csmanage.

```
    <appSettings>
        <add key="CheckServerCertificate"
             value="true"/>
        <!-- Insert your subscriptionId as shown by the Windows Azure
developer portal -->
        <add key="SubscriptionId"
             value="<SubscriptionId>"/>
        <!-- Insert your certificate thumbprint without spaces -->
        <add key="CertificateThumbprint"
             value="<Thumbprint>"/>
      </appSettings>
```

Y después simplemente utilizarlo

```
    csmanage /list-hosted-services

    csmanage /slot:production /delete-deployement /name:<name>
```

23.- POWERSHELL CMDLETS PARA WINDOWS AZURE

Como ya comentábamos en un punto anterior, todo lo que se puede hacer desde el portal de administración de Azure se puede hacer sin necesidad de usar este portal, a través de un API REST de administración.

Pero para simplificar el trabajo existen otras opciones, que no pasan por tener que explicar las peticiones REST manualmente.

Una opción muy interesante es usar comandos de powershell para realizar las labores de administración sobre Azure, para los cuál es necesario descargarse e instalar el siguiente componente.

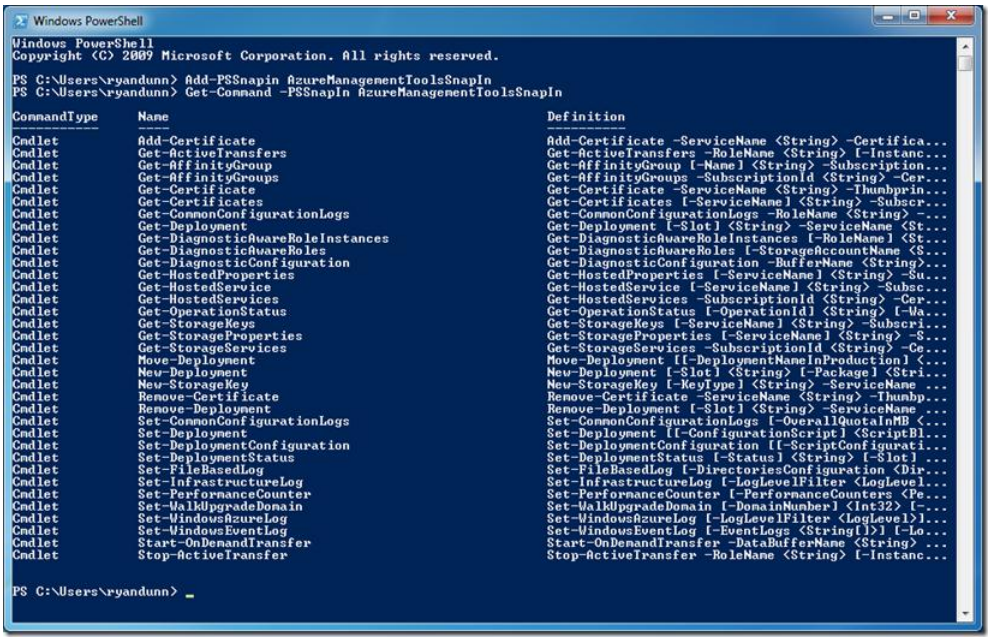


Figura 2.78.- CmdLets de PowerShell

¿Cómo se puede indicar a un servicio desplegado en Windows Azure que aumente una instancia?

```
    $cert = Get-Item cert:\CurrentUser\My\<Certificate ThumbPrint>
    $sub = <Azure Subcription Id>
    $servicename = <Service Name>

    Get-HostedService $servicename -Certificate $cert -SubscriptionId
$sub |
    Get-Deployment -Slot Production |
    Set-DeploymentConfiguration
{$_.RolesConfiguration["WebSample"].InstanceCount += 1}
```

Este ejemplo coge la configuración del servicio, lo modifica para aumentar en 1 el número de instancias configurado y establece la nueva configuración en el servicio de Azure.

24.- WINDOWS AZURE POWERSHELL 2.0

Trabajando con aplicaciones reales en Windows Azure te das cuenta rápidamente de que necesitas automatizar ciertas tareas, ya que hacerlas de manera manual puede resultar muy pesado y poco productivo; despliegues, actualizaciones, monitorizaciones etc…

Para esta fin los CmdLets de PowerShell para Windows Azure ha sido desde su aparición un gran aliado, aliado que ahora se actualiza a la versión 2.0 que incluye nueva funcionalidad gracias a que Windows Azure ofrece cada vez más APIs de Administración que los desarrolladores pueden utilizar.

Tanto la descarga como la documentación sobre su utilización se pueden descargar desde CodePlex.

Las novedades más importantes son los comandos para trabajar con SQL Azure, aquellos que exportan todas las métricas del Storage y los que permiten tratar la información de diagnóstico de las aplicaciones Azure.

Por ejemplo, algunos comandos nuevos son:

Tabla 2.1.- Windows Azure Storage Analytics

Comando	Descripción
Get-StorageAnalyticsLogs	Downloads the analytics logs for the specified service.
Get-StorageAnalyticsMetrics	Downloads the Windows Azure Storage Analytics metrics for the specified service
Get-StoragePropertiesForAnalytics	Analytics properties for a storage account.
Set-StoragePropertiesForAnalytics	Sets Windows Azure Storage Analytics properties for a storage account.

Tabla 2.2.- SQL Azure Servers

Comando	Descripción
Get-SqlAzureServer	Enumerates SQL Azure servers that are provisioned for a subscription
New-SqlAzureServer	Adds a new SQL Azure server to a subscription
Remove-SqlAzureServer	Deletes a SQL Azure server from a subscription
Set-SqlAzurePassword	Sets the administrative password of a SQL Azure server.

Tabla 2.3.- SQL Azure Firewall

Comando	Descripción
Get-SqlAzureFirewallRules	Retrieves a list of all the firewall rules for a SQL Azure server
New-SqlAzureFirewallRule	Updates an existing firewall rule or adds a new firewall rule for a SQL Azure server
Remove-SqlAzureFirewallRule	Deletes a firewall rule from a SQL Azure server.

25.- PROFILING

Desde la aparición del Sdk 1.4 existe la posibilidad de usar un profiler junto con los despliegues de Windows Azure.

En funcionamiento y configuración es exactamente igual que el profiler que se podría emplear en aplicaciones on-premise, lo único que cambia es se debe indicar que se quiere hacer profiling de la aplicación que se están desplegando.

En el menú de publicación de Visual Studio se puede ver cómo está disponible dicha opción. Una vez seleccionada también se debe elegir entre los tipos de profiling que existen.

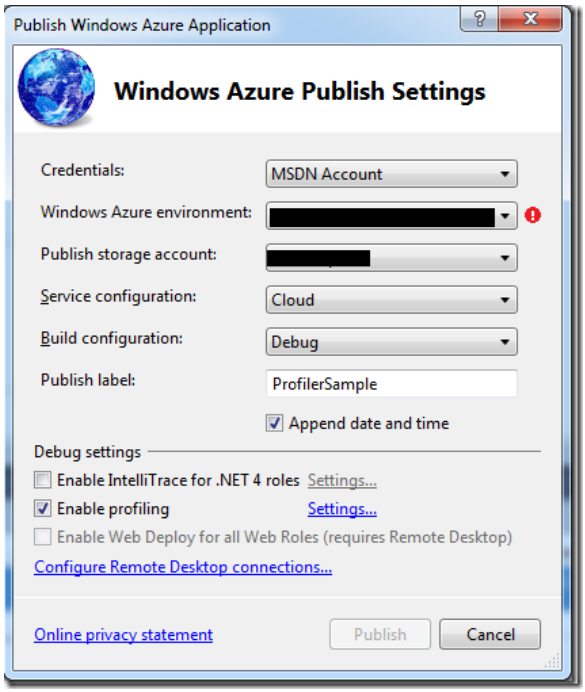


Figura 2.79.- Opciones de publicación

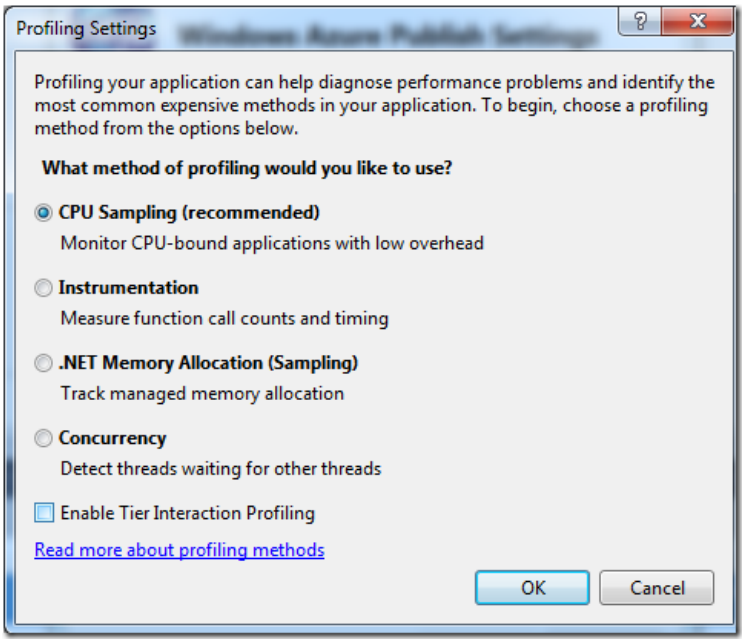


Figura 2.80.- Configuración de profiling

Antes de obtener la información generada, se debe indicar que se quieren usar los símbolos de depuración.

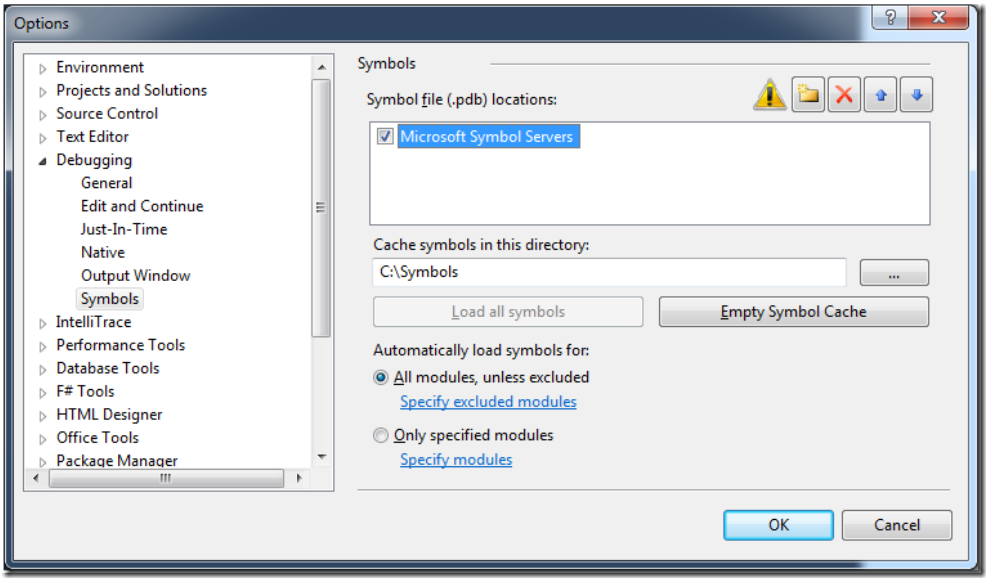


Figura 2.81.- Configuración de los símbolos

Una vez desplegada la aplicación se puede acceder a la información a través del "server explorer" de Visual Studio. Es en este momento dónde se solicita la información al agente de profiling instalando en la instancia desplegada para que éste deje la información en el storage, para poder descargarla desde ahí.

Y una vez se dispone de la información, sólo queda analizarla y buscar los puntos de mejora, tal y como se haría con cualquier otra aplicación no desplegada en Windows Azure.

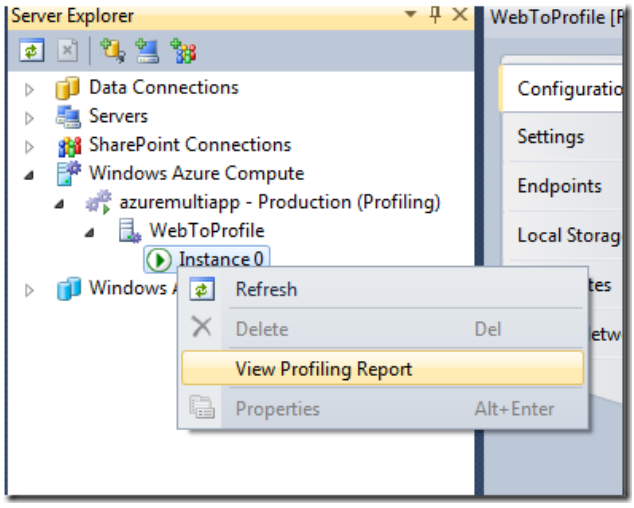


Figura 2.82.- Obtener información de profiler

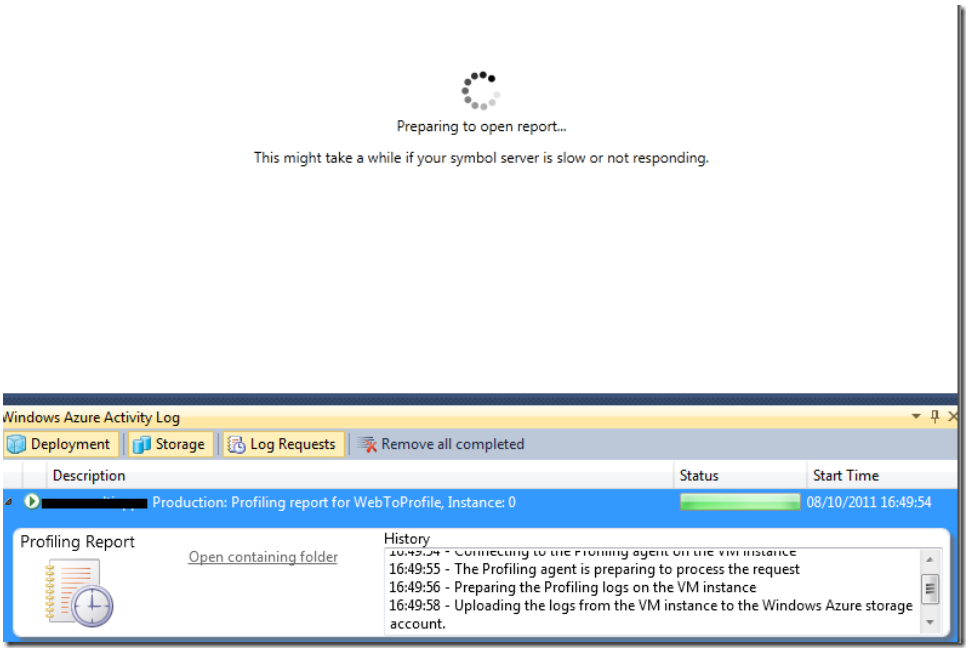


Figura 2.83.- Obtener la información de profiling

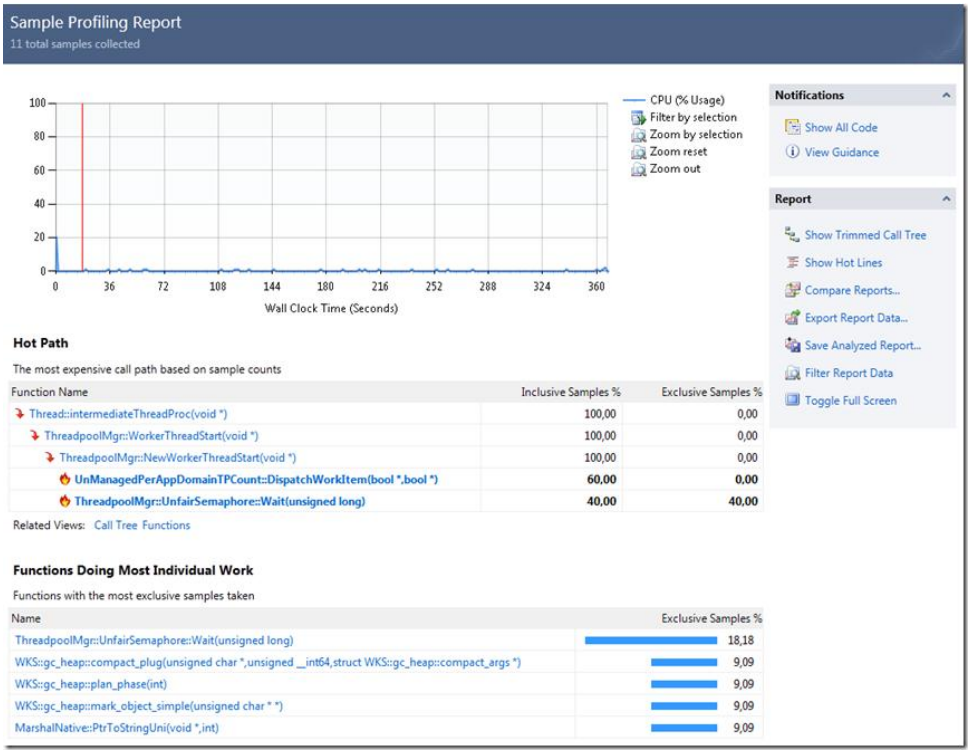


Figura 2.84.-Información de profiling

26.-VIRTUAL MACHINE ROLE

Un VMRole es un que permite al usuario desplegar una imagen (VHD) personalizada un Windows Server 2008 R2. El usuario dispone de mayor control sobre el contenido de la máquina, ya que es éste quién genera la imagen y quién instala todo aquello que considere necesario.

A su vez, el usuario que utilice este tipo de rol tiene mayores responsabilidades, como encargarse de las actualizaciones.

En la mayoría de ocasiones la opción más adecuada sin lugar a dudas será usar un Web o Worker Role, utilizando las "startup tasks" para instalar todo el software que se necesite. Aun así, existen escenarios dónde usar un VMRole es la única alternativa.

Un VMRole no es más que un fichero VHD que contiene el sistema operativo y opcionalmente uno o más VHD diferenciales.

El modelo y proceso de programación y despliegue es similar al que se realiza para los otros dos roles, salvo que en este caso se despliega un fichero VHD que contiene la imagen de un servidor.

Por ejemplo, aunque se despliegue una imagen personalizada, el Fabric de Windows Azure se encargará de monitorizar el estado de la instancia y si el funcionamiento no es el adecuado se encargaría de regenerar la imagen de la instancia (reimage).

Por este motivo, al igual que ocurre con los otros roles, las aplicaciones desplegadas en este rol no puede mantener el estado en la instancia, ya que este se puede perder; no un rol adecuado para instalar aplicaciones como SQL Server, SharePoint o cualquier otra que requiera estado.

Para algunas aplicaciones personalizadas sí existe la posibilidad de implementar adaptadores personalizados que guarden la información que se quiera persistir al Windows Azure Storage y así no perder el estado de la máquina aunque éste se regenere.

Trabajando con VMRole existen algunas restricciones que deben tenerse en cuenta:

- El sistema operativo de la imagen puede ser Windows 2008 Server R2 Enterprise y Standard, siempre en inglés.
- No está soportado en el entorno de desarrollo.
- No soporta protocolo UDP.

El primer paso si se quiere trabajar con un VMRole será crear la imagen usando Hyper-V Manager.

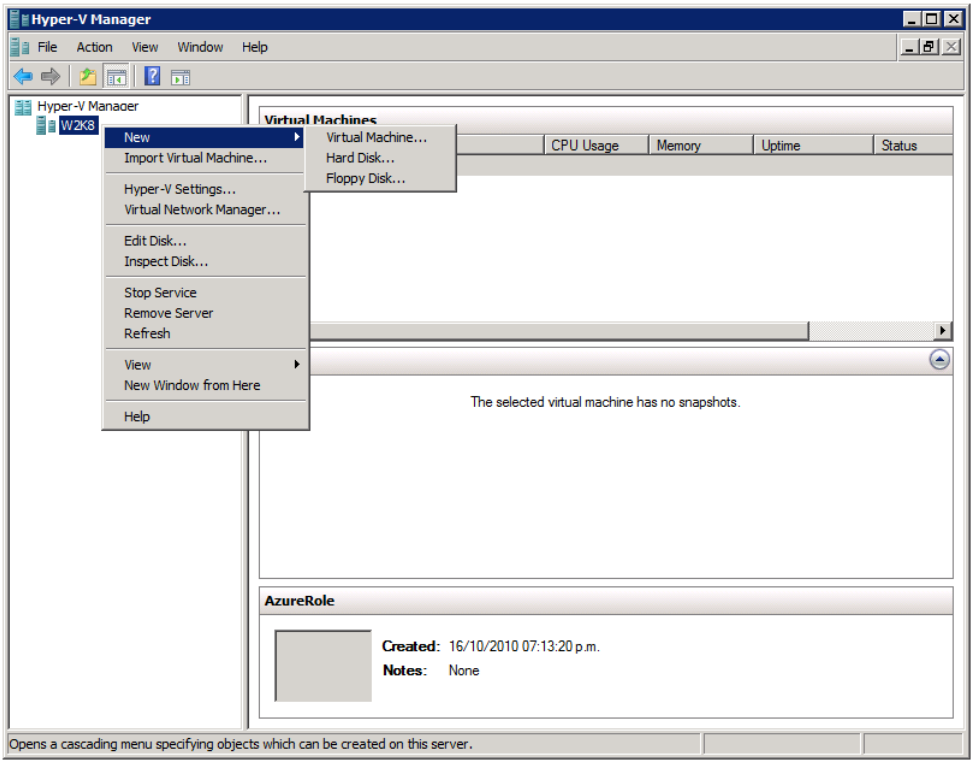


Figura 2.85.-Hyper-V Manager

En la documentación y training kit de Windows Azure existe una documentación detallada paso a paso sobre cómo crear la imagen y todos aquellos aspectos a tener en cuenta.

Una vez creada, el siguiente paso es subir la imagen, para lo cual se puede usar la aplicación csupload disponible con el Sdk de Windows Azure. Esta aplicación hace uso del API de administración, por lo que requerirá haber subido previamente un certificado de administración al portal.

```
csupload Add-VMImage -Connection "SubscriptionId=<YOUR-SUBSCRIPTION-
ID>; CertificateThumbprint=<YOUR-CERTIFICATE-THUMBPRINT>" -Description
"Base image Windows Server 2008 R2" -LiteralPath "<PATH-TO-VHD-FILE>" -
Name baseimage.vhd -Location <HOSTED-SERVICE-LOCATION>
```

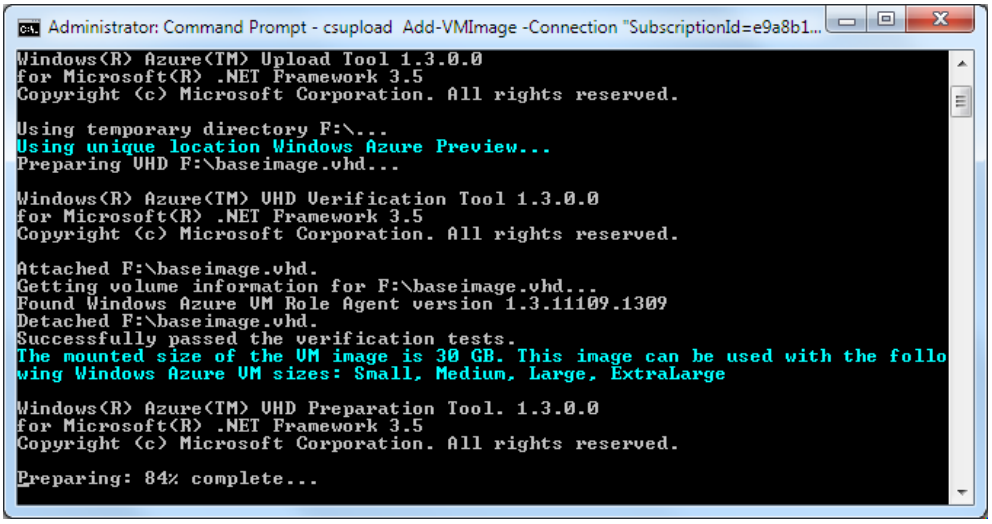


Figura 2.86.- CSUpload

Una vez subida la imagen se podrá ver que está disponible en el portal de Windows Azure, dentro de la sección VM Images.

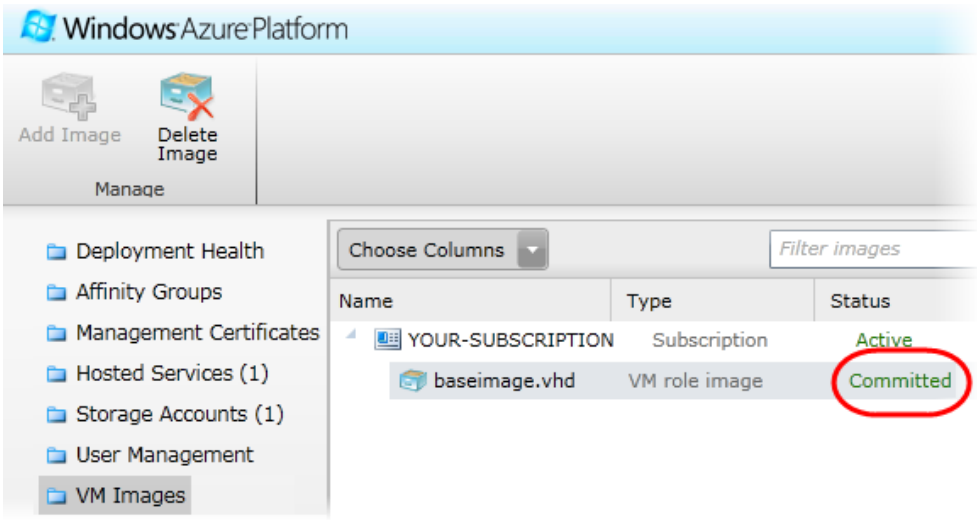


Figura 2.87.-VM Images

Una vez la imagen se encuentra subida en Windows Azure, el siguiente paso es crear un servicio (Hosted Services) y hacer un despliegue de un rol que haga uso de la imagen.

Para ello es necesario crear un proyecto de Visual Studio sin roles. Una vez creado desde el menú contextual aparecerá la opción de añadir un nuevo rol de tipo Virtual Machine Role.

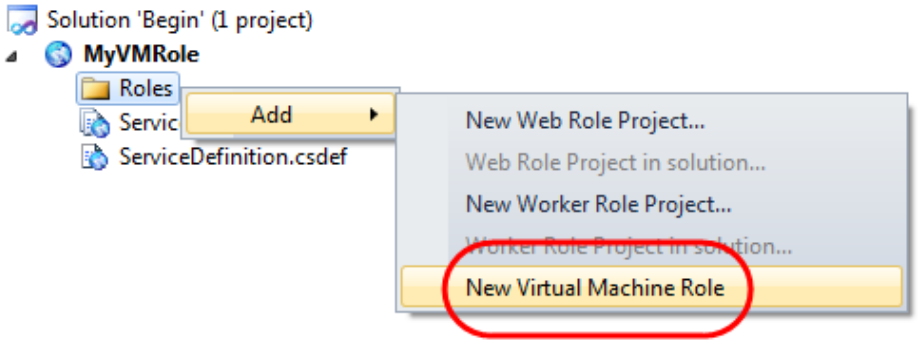


Figura 2.88.- Añadir VMRole

Una vez creado, se debe establecer la configuración como se hace en el resto de roles; endpoints dónde debe escuchar, certificados, número de instancias…la única diferencia, es que en este caso no se despliega una aplicación, sino que se despliega una imagen que previamente se ha subido.

Para que aparezca la lista de imágenes disponibles en el servidor es necesario haber configurado Visual Studio para que éste pueda tener acceso a la información de la subscripción Windows Azure.

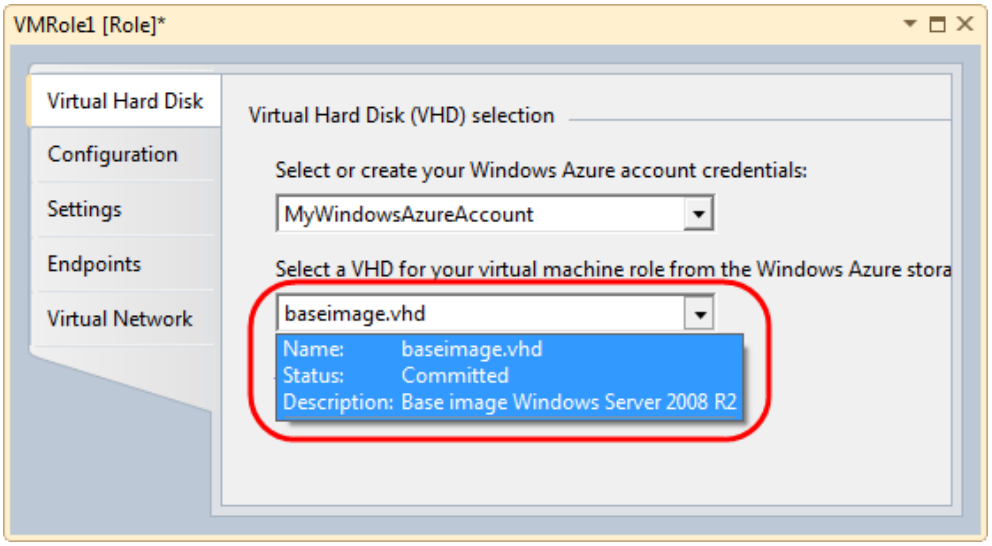


Figura 2.89.- Seleccionar imagen

Una vez hecho esto, el proyecto puede ser desplegado directamente o de forma manual, creando un nuevo despliegue y subiendo los ficheros de configuración y despliegue generados por Visual Studio al seleccionar el despliegue manual.

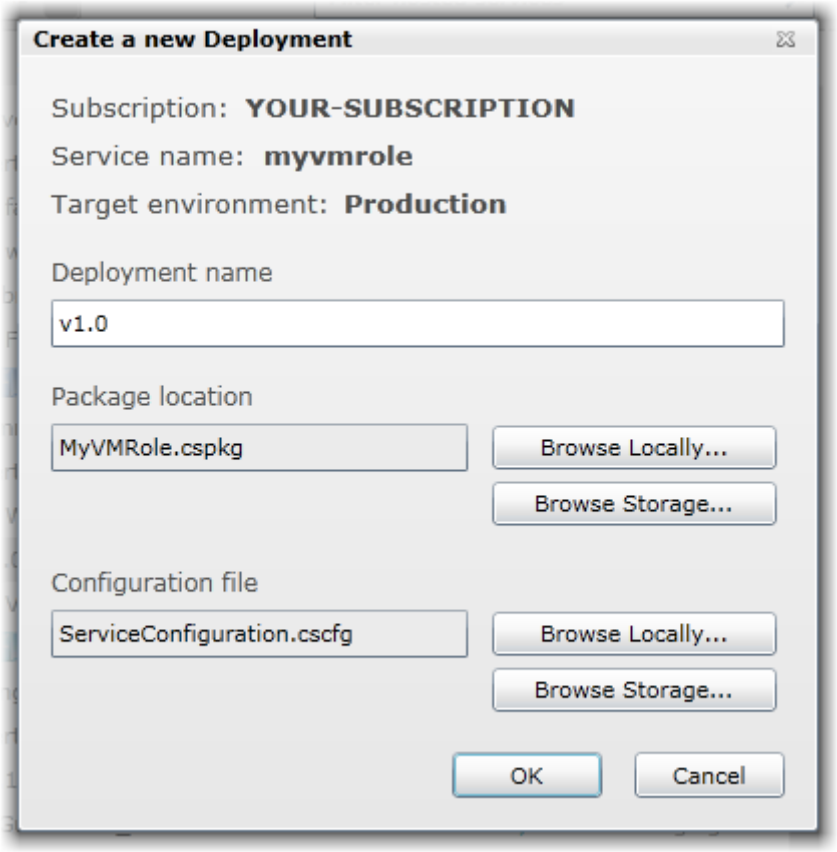
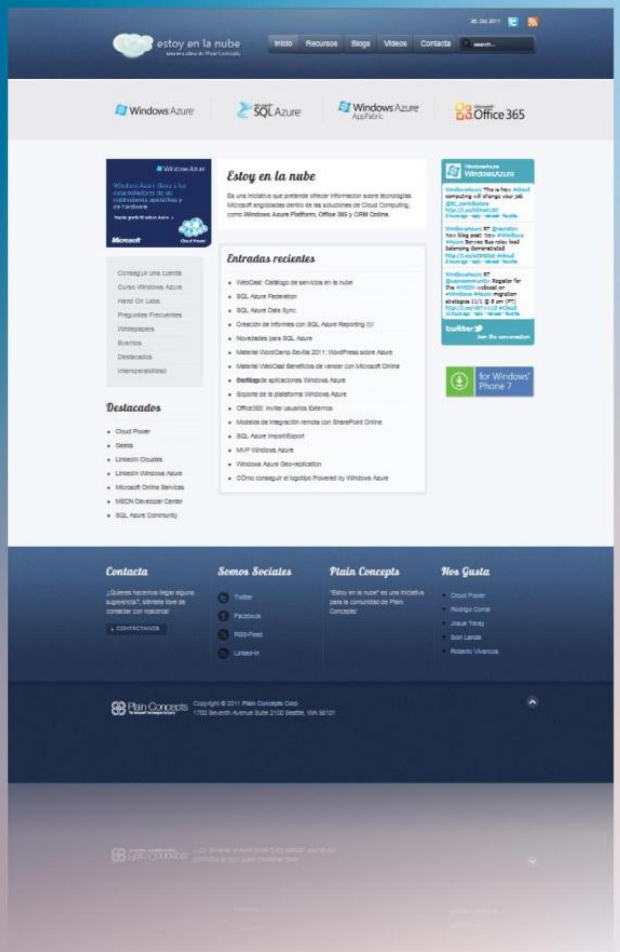


Figura 2.90.- Crear despliegue

estoy en la nube

INICIATIVA DE PLAIN CONCEPTS

www.estoyenlanube.com



Windows Azure

www.plainconcepts.com

Plain Concepts is a company specialized in Microsoft technologies, agile methodologies, Application Lifecycle Management, performance tuning, advanced debugging, software architecture and User Experience.

Plain Concepts focuses on delivering high quality consulting, mentoring and training as well as in being an effective and reliable team resolving all type of software development issues.

Plain Concepts

¿Aún quieres más?

campus MVP

Formación online especializada en tecnologías Microsoft.

krasis PRESS

Los libros que lo saben todo sobre tecnologías Microsoft.

Síguenos y descubrirás los mejores trucos y recursos:

facebook.com/campusmvp twitter.com/campusmvp

feed your brain®

- Sin tener que desplazarse
- Sin romper el ritmo de trabajo
- Preguntándole a los que más saben

infórmate ya:
902 876 475
www.campusmvp.com

http://www.krasis.com

krasis

Microsoft Partner
Silver Learning
Silver Software Development