Aprenda la disciplina ejerza el arte y contribuya con sus ideas en el nuevo Centro de Recursos de Arquitectura.

Recursos que sirven de base.
microsoft.com/architecture

THE ARCHITECTURE JOURNAL
Journal 5

Integración e Intercambio

Microsoft

Indice

Reconocimientos

Editor Ejecutivo & Gerente de Programación
Arvindra Sehmi
Architect, Developer and Platform Evangelism Group, Microsoft EMEA
www.thearchitectexchange.com/asehmi

Editor Administrador
Graeme Malcolm
Principal Technologist, Content Master Ltd

Consejo Editorial
Christopher Baldwin
Principal Consultant, Developer and Platform Evangelism Group, Microsoft EMEA
Felipe Cabrera
Architect, Advanced Web Services, Microsoft Corporation
Gianpaolo Carraro
Architect, Developer and Platform Evangelism Group, Windows Evangelism, Microsoft Corporation
Mark Glikson
Program Manager, Developer and Platform Evangelism Group, Architecture Strategy, Microsoft Corporation
Simon Guest
Program Manager, Developer and Platform Evangelism Group, Architecture Strategy, Microsoft Corporation
www.simonguest.com
Neil Hutson
Director of Windows Evangelism, Developer and Platform Evangelism Group, Microsoft Corporation
Terry Leeper
Director, Developer and Platform Evangelism Group, Microsoft EMEA
Eugenio Pace
Program Manager, Platform Architecture Group, Microsoft Corporation
Harry Pierson
Architect, Developer and Platform Evangelism Group, Architecture Strategy, Microsoft Corporation
devhawk.net
Michael Platt
Architect, Developer and Platform Evangelism Group, Microsoft Ltd
blogs.msdn.com/michael_platt
Beat Schwegler
Architect, Developer and Platform Evangelism Group, Microsoft EMEA
Philip Teale
Partner Strategy Manager, Enterprise Partner Group, Microsoft Ltd

Gerencia de Proyecto
Content Master Ltd
www.contentmaster.com

Dirección de Diseño
Venturethree, London
www.venturethree.com

Orb Solutions, Londres
www.orb-solutions.com

Organización
Richard Hughes
Program Manager, Developer and Platform Evangelism Group, Architecture Strategy, Microsoft Corporation

Colaborador de Prefacio
Gurpreet Pall
Senior Director, Architecture Strategy Team, Microsoft Corporation

Nota del Editor

Estimado Arquitecto,

En los últimos 18 meses he tenido la fortuna de presenciar la evolución de la Revista para Arquitectos desde sus comienzos hasta esta edición actual - *Journal 5*. Ha crecido en muchos aspectos, en especial como un medio para que los arquitectos del mundo entero compartan sus ideas e incorporen perspectivas nuevas y únicas. Un indicador clave que muestra que este medio tiene vida por sí mismo es el sentido de continuidad que surge a través de los artículos - referencias y comentarios sobre artículos de revistas previas y el compromiso de los autores de escribir una segunda parte de los temas que ellos presentan en esta edición. Esto me motiva a explorar las ediciones anteriores y aumenta mi curiosidad respecto de las futuras.

La arquitectura en el mundo de los sistemas es tan extensa como la galaxia, y los arquitectos que operan en ella enfrentan desafíos únicos no comparables a ningún otro reto imaginable. Lo que hace esto aún más desafiante es que a penas unas pocas instituciones educacionales ofrecen un título formal en esta disciplina. A diferencia de sus equivalentes en el mundo de la edificación que tienen sus diplomas encuadrados colgando contra la pared, el conocimiento de la arquitectura no es tan fácil de adquirir. Se trata de una disciplina refinada que han elegido ejercer sólo los valientes Caballeros Jedi que poseen habilidades especiales. Con el veloz ritmo de cambios en el mundo de los negocios y la rapidez de las innovaciones de la arquitectura, los arquitectos necesitan un medio para mantenerse actualizados, intercambiar ideas con sus pares y crecer. Sólo los individuos que se sienten seguros trabajando con la ambigüedad, aquellos que poseen conocimiento y experiencia en varias disciplinas y los que disfrutan hacer malabares con los requerimientos de los diversos participantes tienden a aventurarse en este exigente y excitante límite. El Journal es un medio clave que recorre un largo camino para que esto sea posible.

Al leer los seis artículos y admirar los gráficos meandros de esta edición, aprendí nuevas cosas y encontré una visión sobre los interrogantes desconcertantes que a veces me quitan el sueño. En el artículo de Richard Veryard y Philip Boxer tal vez vaguemos por las calles de una Metrópolis dinámica buscando paralelos con el espacio del gobierno de SOA. Seguí paso a paso la secuencia de una lista de comprobación preliminar completa para la evaluación de tecnologías por Anna Liu y Ian Gorton para mis necesidades de integración basadas en el servicio. Si tuviera que seleccionar las alternativas de transporte adecuado para los Servicios Web, tomaría el camino de "Aviones, Trenes y Automóviles" de Simon Guest. Por un momento, pensé que estaba leyendo sobre la planificación de transportes metropolitanos.
Cada artículo de esta revista es rico y conducente y establece el ritmo para "Pensar, Aprender y Solucionar", tema del recientemente inaugurado Centro de Recursos para la Arquitectura [Architecture Resource Center] en Microsoft.com.

Estoy segura de que está edición lo hará pensar, aprenderá algo nuevo y lo ayudará a evolucionar como lo ha hecho conmigo. También espero que lo motive a utilizar el Journal como un medio para compartir sus conocimientos y perspectivas.

Disfrútela!
Gurpreet S. Pall
Director Senior, Architecture Strategy Team
Microsoft Corporation

Prólogo

Estimado Arquitecto,

Es muy claro que a partir de la segunda edición del Journal, ahora conocido como el Journal de Arquitectura, la demanda ha sido cada vez mayor con respecto a este tipo de publicaciones de Microsoft. Como editor, he recibido mucho incentivo por parte de mis amigos, colegas y clientes para realizar una iniciativa permanente en vez de un arranque de entusiasmo. Mi objetivo ha sido siempre desarrollar una comunidad sólida de personas que posean un espíritu similar al que trata esta revista. Para ser aceptado en el mercado esto debe proporcionar una plataforma acreditada de manera tal que los autores expresen sus opiniones y puntos de vista con un amplio criterio y la menor interferencia de Microsoft posible, y que al mismo tiempo sean creíbles y económicamente factibles debido al gran apoyo de Microsoft. Me alegra decir que el Journal estará con nosotros por mucho tiempo.
En el período desde que se publicó el Journal 4 y se dio a conocer esta última edición, Journal 5, la revista ha evolucionado desde su etapa de incubación para convertirse en una parte clave de las estrategias de desarrollo de la comunidad de arquitectos de Microsoft. El Journal destaca en primer plano el Centro de Recursos de Arquitectura de Microsoft (visitar www.microsoft.com/architecture), y ahora proporcionamos suscripción gratuita para la versión impresa. Invertimos en el profesionalismo que requiere el Journal creando un grupo dedicado para su producción y publicación en colaboración con Fawcette Technical Publications. El objetivo de estos hechos es asegurar que usted, nuestros lectores y autores obtengan experiencias y valores destacados al leer el Journal. Lo mejor de la versión temporaria del Journal nos ha llevado a pasar con éxito del formato "grande" original al presente formato normal de revistas. Este año se han distribuido decenas de miles de copias en las conferencias TechEd que se llevaron a cabo en EE.UU. y en Europa! Al mismo tiempo observamos mucha inercia e interés en la arquitectura, a punto tal que las charlas de arquitectura de los especialistas en estas conferencias compiten a la par con las charlas tradicionales de los desarrolladores ¡Me entusiasma el hecho de estar aquí en este espacio y pueden contar con el Journal para liderar el camino!
Finalmente, notarán que en esta oportunidad escribo el Prólogo y no un editorial como es de esperar. Bien, esto se debe a que Gurpreet Pall realizó un muy buen trabajo con su versión de la introducción, por lo que decidimos cambiar.
Como siempre, si está interesado en escribir para esta publicación, por favor envíeme un breve resumen del tema y su CV a asehmi@microsoft.com.
¡Le deseo una buena lectura!

Arvindra Sehmi

Introducción a Mapas Conceptuales

Kal Ahmed y Graham Moore

Introducción

Este artículo introduce los Mapas Conceptuales estándares internacionales ISO. El paradigma de los mapas conceptuales describe la forma en que las relaciones complejas entre conceptos abstractos y los recursos del mundo real pueden describirse e intercambiarse utilizando una sintaxis XML estándar.

Los Mapas Conceptuales fueron en un principio desarrollados a fines de 1990 como una forma de representar las estructuras de índices al final de los libros para que de esta forma los múltiples índices de distintos sistemas pudieran fusionarse. No obstante, los desarrolladores pronto se dieron cuenta de que con una pequeña generalización adicional, podían crear un modelo meta de aplicación potencialmente mucho más amplia. El resultado de este trabajo se publicó en 1999 como ISO/IEC 13250-Topic Navigation Maps.

Además de describir el modelo básico de los mapas conceptua les y los requerimientos necesarios para un procesador de mapas conceptuales, la primera edición de ISO 13250 incluyó una sintaxis de intercambio basada en SGML y el lenguaje de enlace hipermedia conocido como HyTime. La segunda edición, publicada en 2002 [1], agregó una sintaxis de intercambio basada en XML y XLink. Hasta la fecha, esta es la sintaxis que cuenta con el más amplio soporte en productos de procesamiento de mapas conceptuales, y es la sintaxis que se describe en este artículo.

Hoy en día existen numerosas implementaciones de lo estándar, ya sea un código abierto o de propiedad exclusiva, para un sin fin de lenguajes y plataformas, incluida la plataforma .NET.

Lo esencial de los Mapas Conceptuales puede resumirse muy brevemente: un mapa conceptual consiste en un conjunto de topics (temas), cada uno de los cuales representa un concepto. Los temas se relacionan entre sí por medio de asociaciones, que son clasificadas como combinaciones de asociaciones n-arias. Un tema puede también estar relacionado a cualquier otro recurso por medio de sus ocurrencias.

La figura 1 muestra los tres fundamentos de los mapas conceptuales.

También expone la forma en la que la distinción entre las relaciones tema-para-tema y tema-para-recurso permiten una partición del modelo en un espacio para temas que contiene sólo temas y asociaciones entre temas y un espacio para recursos que contiene los recursos relacionados a los temas.

Esta partición es interesante ya que permite que un mapa conceptual desarrollado para un conjunto de recursos se reaplique para indexar un conjunto de recursos diferente.

Figura 1. Temas, asociaciones y ocurrencias

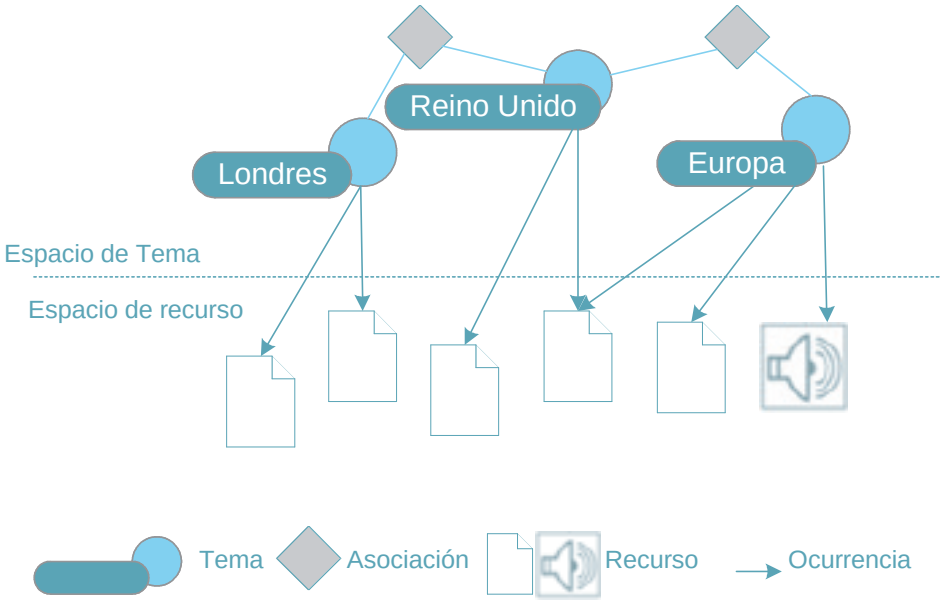
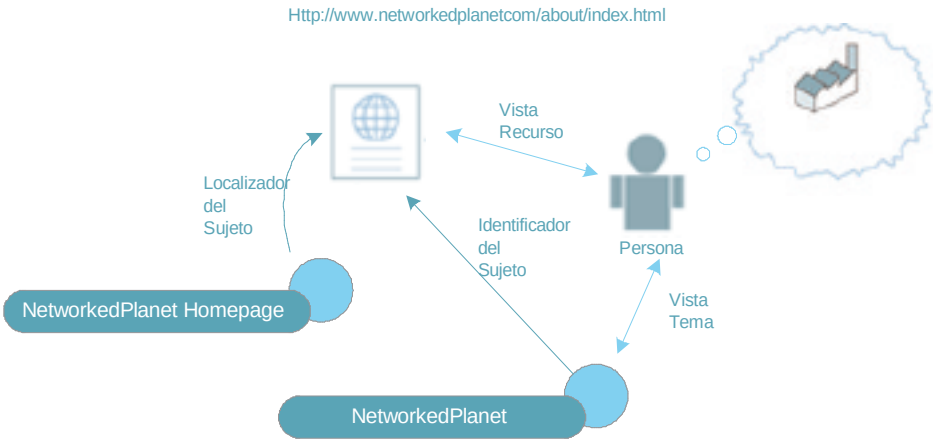


Figura 2. Localizador e identificador del tema



De esta manera, se puede considerar que el mapa conceptual es una forma portátil de conocimiento.

A diferencia de los modelos específicos de dominio, el modelo de mapas conceptuales no posee un conjunto de tipos pre-definidos. En cambio, los autores de mapas conceptuales individuales o grupo de autores en una comunidad de práctica pueden definir el modelo para su dominio de interés y compartirlo con otros autores de otros dominios.

Creemos que para muchos usuarios finales, una buena aplicación de mapas conceptuales ocultará mucho, o todos los mecanismos de los mapas conceptuales, permitiendo que los usuarios se concentren en el modelo de dominio con el que trabajan. Sin embargo, los modelos de mapas conceptuales y los Mapas Conceptuales estándar proporcionan gran cantidad de beneficios que pueden dar cierta clase de superficie a las aplicaciones y pueden ser puntos de venta únicos.

La metáfora central de los mapas conceptuales de temas, ocurrencias y asociaciones encuentra un equilibrio entre lo compacto y lo fácil de comprender y proporcionando la suficiente infraestructura básica para permitir que los usuarios traduzcan los modelos que tienen en sus mentes respecto de un dominio en un modelo de mapa conceptual. Otras formas de organización de la información y de los datos, como RDF y el modelo de relación puede tener aún un modelo más simple, pero luego requerirá que el usuario cree infraestructura para procedimientos comunes como por ejemplo etiquetar un artículo con nombres, definiendo una estructura de clase o creando relaciones n-arias entre los artículos.

Tal como se ha descripto anteriormente, los modelos de mapas conceptuales poseen una clara distinción entre el modelo de dominio, expresado como temas y asociaciones entre temas, y los recursos indexados, que se expresan como ocurrencias que vinculan los temas con los recursos. De esta estructura derivan tres beneficios principales:

- El mapa conceptual puede actuar como una visión general de alto nivel del conocimiento del dominio contenido en un conjunto de recursos. De esta manera, el mapa conceptual puede no sólo servir como guía para ubicar los recursos para el experto, sino también como forma para que los especialistas modelen su conocimiento de manera estructurada. Esto permite que quienes no son expertos comprendan los conceptos básicos y sus relaciones antes de sumergirse en los recursos que proporcionan más detalles.
- Un mapa conceptual puede dividirse con facilidad dependiendo de los recursos que los hacen disponibles. Algunos editores utilizan mapas conceptuales basados en el índice de grandes conjuntos de recursos, y luego, de manera dinámica, crean el índice adecuado al publicar el subconjunto de estos recursos. Con alguna creación cuidadosa de modelos, aún es posible diseñar diferentes niveles de detalle en un mapa conceptual, y diferenciar entre productos de información basados en el indexado y las características de navegación que proporcionan así como también los contenidos de información del producto.
- Los mapas conceptuales que indexan diferentes conjuntos de recursos pueden combinarse fácilmente. Esta característica puede utilizarse para permitir que las organizaciones "importen" datos e índices de terceros e integren de manera homogénea sus propios datos e índices.

Tal como se ha mencionado, los Mapas Conceptuales estándar no poseen una ontología predefinida1. No existe restricción alguna sobre los dominios a los cuales se les puede aplicar mapas conceptuales y existen aún menos restricciones sobre el enfoque de modelado que se escoge.

Fácil Intercambio

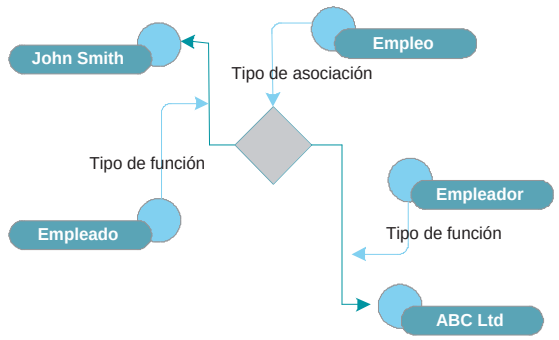
Hemos visto que los mapas conceptuales se utilizan para modelar relaciones temporales entre eventos; relaciones entre conceptos abstractos y su representación ; y formas de lógica de primer orden así como también otras relaciones tradicionales como tesauros, vocabulario controlado e información de la empresa.

Para muchos usuarios, el hecho de que los mapas conceptuales puedan intercambiarse utilizando una sintaxis basada en XML estándar proporciona beneficios sólidos mejorando la portabilidad de los datos entre las aplicaciones y las plataformas. Además, la sintaxis de intercambio XML permite una fácil integración del intercambio de información de los mapas conceptuales dentro de una arquitectura de servicios web.

Existen tres beneficios principales con los cuales los arquitectos de sistemas y desarrolladores pueden favorecerse a partir de la aplicación del paradigma del mapa conceptual, y pueden resumirse como "Flexibilidad, Flexibilidad y Flexibilidad".

Los mapas conceptuales proporcionan el modelo meta sobre el cual se puede construir un modelo de aplicación totalmente flexible.

Figura 3. Estructura de una asociación



Es posible crear nuevos tipos de objetos empresariales al agregar datos a la ontología que construye el mapa conceptual. Debido a que la ontología se expresa a sí misma como temas o asociaciones entre temas, la extensión de la ontología es el resultado de la incorporación de datos y no del rediseño del sistema de almacenamiento subyacente. Esto permite modificar el modelo de datos que utiliza una aplicación sin necesidad de actualizar los sistemas de almacenamiento persistentes ya implementados.

Almacenar los datos de la aplicación en un modelo-meta extensible y estandarizado, posibilita una extensión e integración más simple de aplicaciones de terceros. Este modelo podría permitir que otros desarrolladores definan sus propios datos y extensiones de códigos para una aplicación sin la necesidad de depender del sistema de aplicación central que soporta la extensión de estructuras de datos específicos.

Además de permitir extensiones de terceros para el sistema de aplicaciones, la flexibilidad de los mapas conceptuales puede utilizarse para permitir que los usuarios creen sus propias extensiones. Esto tiene dos efectos:

- Posibilita el nivel más alto de personalización de las aplicaciones: por ejemplo, posibilita una integración mucho más fácil de los sistemas de datos específicos del cliente. Consideramos que esta es la clave para que las aplicaciones verticales puedan desplegarse con más facilidad para clientes múltiples. Por ejemplo, un editor de información legal produjo una aplicación de mapa conceptual exitosa para el mercado de servicios financieros. El único punto de venta de su producto basado en el mapa conceptual es que puede integrar la documentación de marketing y procedimiento de sus clientes con la información legal que le proporcionan.
- Permite el desarrollo de aplicaciones horizontales que pueden integrarse con mucha más facilidad a entornos existentes.

Un tema es la representación de un concepto procesable por máquinas. Los Mapas Conceptuales estándar no restringen de manera alguna el conjunto de conceptos que pueden representarse como temas. Por lo general, los temas se utilizan para representar recursos electrónicos (como documentos, páginas en Internet, servicios web) y recursos no electrónicos (como personas o lugares). De igual manera, los temas pueden utilizarse para representar cosas que no tienen una forma tangible en sí, como por ejemplo empresas, eventos y conceptos abstractos como "Pensiones" o "Seguros".

Formas de Identidad

Los temas poseen cuatro formas de identidad principales. Un tema puede tener o no alguna de estas formas de identidad, y por lo tanto puede ser identificado dentro de un sistema de mapa conceptual a través de muchas formas diferentes:

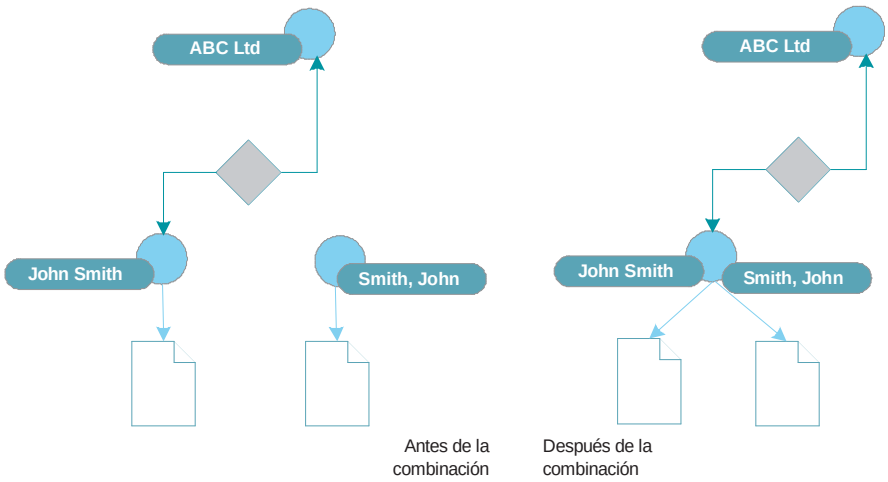
- La identidad como un recurso de tema en un mapa conceptual en serie. Cuando un mapa conceptual se representa en forma serializada para el intercambio, se le asigna a cada tema un identificador URI que es único a lo largo del mapa conceptual. Estos URIs se utilizan principalmente para deserializar las referencias entre temas. A estos identificadores se los llama localizadores de recursos.
- La identidad como una etiqueta legible por el usuario. Un tema puede tener un sin fin de nombres. Los nombres ofician de etiquetas para el consumo humano y pueden ser de texto o referencia de alguna representación no textual (por ejemplo, un icono, un clip de sonido, un clip animado, etc.). El mecanismo de alcance (descripto más adelante) permite el caso de homónimos (donde una misma palabra se utiliza para referirse a dos o más conceptos diferentes).
- Identidad por referencia. Cuando un tema ser utiliza para representar un recurso que ya posee su propio URI, este URI puede usarse como parte de la identidad del tema. Esto es simplemente una forma de decir al agente de procesamiento "este tema simboliza este recurso". En el mapa conceptual estándar, esta forma de identificador se conoce como localizador del tema.
- Identidad por descripción. Los temas pueden utilizarse para representar un concepto que no posee su propio URI. Muchas de estas cosas que un tema puede representar tal vez nunca tengan un único URI ya que no son cosas a las que una computadora les pueda establecer un vínculo de referencia. Por ejemplo, una persona puede tener muchos registros de bases de datos propios o biografías online o fotos, pero ninguno de estos recursos personalizados son la persona -sino una mera descripción de la persona. En los Mapas Conceptuales estándar, esta forma de identificador se conoce como identificador del tema, y el recurso al cual identificador del tema se vinculó se denomina indicador de tema. Los mapas conceptuales permiten el uso de referencias URI para estos recursos descriptivos como una forma de identidad. Obviamente, es importante que el autor del mapa conceptual seleccione recursos descriptivos que no sean ambiguos a tal fin, y éste es un tema que volveremos a tratar más adelante.

La distinción de las dos últimas formas de identidad puede ser confusa.Tener presente la URL http:// www. networkedplanet.com /about/index.html. Ésta es una página en Internet que describe la empresa NetworkedPlanet. Por lo tanto, esta URL podría utilizarse como identificador de tema para un asunto llamado "La empresa NetworkedPlanet", ya que establece un vínculo con un recurso que describe el concepto de la empresa. Sin embargo, si deseáramos hablar acerca del concepto "La página 'About' en el sitio de Internet www.networkedplanet.com", desearíamos un tema cuyo asunto sea en realidad el recurso en la dirección http://www.networkedplanet.com/about/index.html y por consiguiente podamos utilizar más tarde el mismo URI como localizador del tema.

Tipos y Nombres

La diferencia esencial entre el identificador del tema y el localizador del tema reside en que el identificador de tema requiere de la interpretación humana de un recurso para determinar el concepto que representa un tema, mientras que el localizador sólo

Figura 4. Combinación de temas



indica el concepto que representa el tema. Esto se representa en la Figura 2. La flecha de línea gruesa muestra el uso de un recurso como localizador del tema.

La flecha de línea fina muestra el uso del mismo recurso como un identificador del tema. La flecha de línea punteada muestra la función de las personas en la interpretación de un identificador de tema.

Si bien un único tema puede tener varias formas de identidad, es importante observar que cada identificador individual puede establecer un vínculo sólo con un tema. Las normas de combinación de los mapas conceptuales (descriptas más adelante) imponen esta relación de uno a muchos (one-to-many) entre los temas y sus identificadores.

Además de estas formas de identidad, un tema puede también tener muchos temas y un sin fin de nombres.

Los tipos de tema definen la clase (o clases) de concepto al cual pertenece de acuerdo al tema que lo represente. Los tipos se tratan en los mapas conceptuales como conceptos por sí mismos; por lo tanto cada tema representa un tipo. El tipo de un tema es representado simplemente por una forma privilegiada de relación entre el tema que representa la instancia, y el tema que representa el tipo.

Los nombres de un tema definen un conjunto de niveles para el tema. Cada nombre posee una estructura jerárquica. En la raíz se encuentra el nombre de base, que posee una representación de secuencias. Para determinar la identidad del tema por nivel se utiliza el valor de la secuencia del nombre de base. El nombre de base es también un contenedor para muchas formas alternativas (conocidas como variantes del nombre). Las formas alternativas de un nombre pueden ser valores de secuencia o referencias a recursos; lo que permite que representaciones como por ejemplo íconos o clips de sonido sean referenciados como variantes del nombre. Se puede otorgar un contexto (o alcance) que sea válido para el nombre de base y las variantes del nombre, permitiendo que una aplicación capaz de distinguir temas relacionados pueda seleccionar el mejor nombre para presentar a un usuario en una situación determinada. El tema del alcance se trata más adelante.

Las asociaciones son la forma general de representación de las relaciones entre los temas y el mapa conceptual. Una asociación puede pensarse como un agregado n-arias de temas. Es decir, una asociación es una agrupación de temas que no poseen implícitos una dirección u orden y no tienen restricción respecto de la cantidad de temas que pueden agrupar.

Se puede asignar un tipo a una asociación (nuevamente definido por un tema) que especifica la naturaleza de la relación representada por la asociación. Además, cada tema que participa en la asociación cumple una función tipificada que especifica la forma en la que participa el tema.

Por ejemplo, para describir la relación entre una persona, "John Smith," y la empresa para la que trabaja, "ABC Limited," se crearía una asociación clasificada con el tema "Empleo" y con los tipos de función "Empleado" (para la función que cumple el señor "John Smith") y "Empleador" (para la función que cumple "ABC Limited").

Al igual que los nombres, se puede asignar a una asociación un alcance en el cual sea válida, y que tal vez pueda ser utilizada por una aplicación capaz de distinguir temas relacionados para determinar si se mostrará o no a un usuario en una situación determinada la información representada por la asociación.

Ocurrencias

Las ocurrencias se utilizan para representar o referirse a información acerca de un concepto representado por un tema. Las ocurrencias pueden usarse para almacenar datos de secuencias en un mapa conceptual, o para referirse a cualquier tipo de recurso externo al mapa conceptual que se pueda ubicar por Internet. No se establecen restricciones respecto del tipo de recurso al que se dirige una ocurrencia. Puede ser una página HTML estática, una página HTML generada por ASP, un servidor de Web o cualquier otro tipo de recurso. Tampoco están restringidas las ocurrencias para el protocolo HTTP -se puede utilizar cualquier dirección codificada como un URI para dirigirse a un recurso externo. Una vez más, las ocurrencias pueden ser clasificadas si se utiliza un tema para expresar el tipo de ocurrencia y también se puede asignar a las ocurrencias un alcance de validez.

El ámbito es el término que utilizan las normas de los mapas conceptuales para referirse a una restricción o a un contexto en el cual se dice algo sobre un tema. La manera en la que se realizan estos juicios acerca de los temas es agregando un nombre al tema; especificando una ocurrencia para un tema; o creando una asociación entre temas (caso en el que el juicio aplica para todos los temas que se encuentran en la asociación).

En muchos casos, los juicios no siempre son verdaderos, pero dependen de un contexto. Por ejemplo, emitimos juicios como

"ABC Limited era el proveedor top de componentes en el Q2 del 2004," o "Fred dice que ABC Limited es un buena inversión". En estas opiniones, el contexto se muestra en cursiva -un contexto temporal en el primer caso y una cita textual en el segundo. Dicho de una manera más prosaica, el contexto se utiliza para facilitar las interfaces multilingües, por lo tanto el concepto "perro" puede tener la etiqueta "dog" en el contexto del idioma inglés, "le chien" en Francés, y "das Hund" en Alemán.

En un mapa conceptual, el ámbito se define por medio de un conjunto de temas que pueden ser asignados a un nombre, una ocurrencia o una asociación. El ámbito por defecto (que no tiene asignado un conjunto) se conoce como ámbito ilimitado y simplemente significa que los nombres, ocurrencias o asociaciones son siempre válidos.

Cuando una aplicación capaz de distinguir temas relacionados se encuentra con un nombre, ocurrencia o asociación que tiene asignado un ámbito, la aplicación debe hacer uso de la información que posee acerca del contexto operativo en curso y comparar esta información con la información del ámbito contenida en el mapaconceptual para determinar si el modelo es válido y si debe o no ser presentado al usuario.

En la actual edición de ISO 13250, los mecanismos para el procesamiento del ámbito contra un contexto de aplicación no están restringidos por la norma, y para muchos desarrolladores de mapas conceptuales esto está visto como un defecto ya que puede dificultar el intercambio de mapas conceptuales que usan un ámbito. En la próxima revisión a la norma se recomendará que un ámbito que está compuesto de múltiples temas debe ser procesado de manera tal que el modelo alcanzado sea válido sólo si la aplicación determina que todos los temas en el ámbito son de aplicación para el contexto de la aplicación en curso.

Combinación de temas

La combinación de temas automática es una función clave de los mapas conceptuales y proporciona muchos beneficios para su desarrollo y también para las aplicaciones que los utilizan en la administración e intercambio de datos.

El principio detrás de la combinación de temas es que en cualquier mapa conceptual determinado, cada uno de los asuntos que éste describe, debe estar representado por un único tema. Esto significa que es responsabilidad del procesador del mapa conceptual identifica la situación en la cual dos temas representan el mismo asunto y procesarlos de manera tal que sólo un tema permanezca. Este es el proceso de combinación.

Se puede lograr identificar si dos temas representan el mismo asunto por medio de la aplicación de la heurística. La norma de los mapas conceptuales define un conjunto de heurísticas básicas:

1. Si dos temas comparten el mismo localizador de recurso, entonces se han analizado desde el mismo recurso de mapa conceptual y debe considerarse que representan el mismo concepto.
2. Si dos temas tienen el mismo localizador de asunto, entonces ambos localizan al recurso de red como lo que ellos representan.
3. Si dos temas tienen el mismo indicador de asunto, entonces ambos utilizan el mismo recurso para describir el concepto que representan y debe considerarse que representan el mismo concepto.
4. Si dos temas tienen cada uno un nombre de base con la misma representación de secuencia y el entorno de los nombres de base pertenecen al mismo conjunto de temas, entonces debe considerarse que los temas representan el mismo concepto.
5. Finalmente, una aplicación de mapa conceptual puede usar cualquier información específica de dominio que posea para determinar que dos temas representan el mismo concepto.

En el punto (3) de la lista que precede surge la importancia de seleccionar un buen recurso como descripción para un concepto. Si la descripción es de alguna forma ambigua o el recurso presentado no está lo suficientemente bien definido, es posible que dos autores de mapas conceptuales diferentes puedan usar el mismo recurso como descriptor para diferentes conceptos, lo que resultará en una combinación no deseada. De acuerdo a nuestra experiencia, los buenos recursos para descriptores de un asunto son aquellos creados específicamente para describir un único asunto -las páginas en wikipedia.org, por ejemplo, o las páginas que crean los autores de mapas conceptuales o una comunidad de profesionales para definir un vocabulario controlado.

El punto (4) ha probado ser polémico en la comunidad de mapas conceptuales ya que depende de lo que algunos consideren que es una forma relativamente débil de identidad -el nombre del concepto en algunos leguajes. La asignación de palabras para conceptos en un lenguaje es un asunto complejo y se enfrenta el desafío de palabras múltiples que poseen diferente significados (homónimos), sin olvidar los desafíos que representa la localización! En la próxima versión de la norma ISO, las restricciones sobre la identidad basada en nombres serán aún más rigurosas para requerir que un autor marque de manera explícita todo nombre de tema que se deberá utilizar para asignar una identidad (ya que la opción predeterminada será que un nombre no conferirá identidad a su tema).

El punto (5) permite que las aplicaciones amplíen los criterios de combinación de las normas de los mapas conceptuales por medio de criterios específicos de aplicación. Esto incluye criterios basados en más de una comparación directa o URI. Por ejemplo, una aplicación puede saber que "The Duke" y "John Wayne" son nombres que se refieren al mismo actor y combinan dos temas sobre esta base. Una vez identificados los temas a combinar, el proceso de combinación define el proceso que reemplaza estos dos (o más) temas por un único tema. Este único tema que resulta del proceso de combinación posee todos los identificadores, nombres (incluyendo variantes del nombre), y ocurrencias de los temas que se combinan. Además, el tema resultante reemplaza los temas que se combinan cada vez que se los consulta (es decir en cualquier asociación, ámbitos o tipos en los que aparecen). Este proceso se muestra de forma esquemática en la figura 4.

La combinación de dos (o más) mapas conceptuales es sólo el proceso de combinaciones de temas y asociaciones y la aplicación al resultado de las normas de combinación de temas.

Intercambio y la sintaxis XTM

Como hemos observado más arriba, las normas de Mapas Conceptuales ISO definen dos sintaxis de intercambio estándar, una basada en SGML y la otra basada en XML. La sintaxis XML define un elemento de mapa conceptual que contiene infinitos temas y elementos de asociaciones. A continuación se muestra un ejemplo simple de la sintaxis XTM en un mapa conceptual.

```
<topicMap xmlns=http://www.topicmaps.org/xtm/1.0/
xmlns:xlink="http://www.w3.org/1999/xlink">
<topic id="band">
<baseName>
<baseNameString>Band</baseNameString>
</baseName>
</topic>
<topic id="person">
<baseName>
```

Lista 1 . La sintaxis XML define un elemento de mapa conceptual que contiene infinitos temas y elementos de asociaciones;en este caso hay una sintaxis para membership, group, singer, y guitarist.

```
-->
<!-- The Clash is a Band -->
<topic id="clash">
<instanceOf>
<topicRef xlink:HREF="#band" TARGET="_self"/>
</instanceOf>
<baseName>
<baseNameString>The Clash</baseNameString>
</baseName>
</topic>
<!-- Joe Strummer is a Person (note multiple names)
-->
<topic id="joe-strummer">
<instanceOf>
<topicRef xlink:HREF="#person" TARGET="_self"/>
</instanceOf>
<baseName>
<scope>
<topicRef xlink:href="stage-name"/>
</scope>

<baseNameString>Joe Strummer</baseNameString>
</baseName>
<baseName>
<baseNameString>Joseph Mellor</baseNameString>
</baseName>
</topic>
<!-
Joe Strummer is a member of The Clash -->

Note separate member elements used for the different
roles played
-->

<association>
<instanceOf>
<topicRef xlink:HREF="#membership" TARGET="_self"/>
</instanceOf>
<member>
<roleSpec>
<topicRef xlink:HREF="#group" TARGET="_self"/>
</roleSpec>
<topicRef xlink:HREF="#clash" TARGET="_self"/>
</member>
<member>
<roleSpec>
<topicRef xlink:HREF="#singer" TARGET="_self"/>
</roleSpec>
<topicRef xlink:HREF="#joe-strummer" TARGET="_self"/>
</member>
<member>
<roleSpec>
<topicRef xlink:HREF="#guitarist" TARGET="_self"/>
</roleSpec>
<topicRef xlink:HREF="#joe-strummer" TARGET="_self"/>
</member>
</association>
</topicMap>
```

```
<baseNameString>Person</baseNameString>
</baseName>
</topic>
<!--
```

No entraremos en detalles respecto de la sintaxis aquí. El lector interesado puede consultar las especificaciones originales de Mapas Conceptuales XML [2] que presentó TopicMaps.org (las cuales posteriormente fueron adoptadas por ISO).

Se debe tener en cuenta que la sintaxis XTM no impone las restricciones de combinación que requiere un procesador de mapa conceptual. Esto permite crear XTM fácilmente, pero requiere que el procesador que lee un archivo XTM detecte los temas que se deben combinar y aplique las normas de combinación en la medida que el archivo XTM se analiza. Cuando un archivo de XTM está "totalmente combinado" (es decir, no incluye elementos de temas que representan temas que deben combinarse), el mapa conceptual que contiene puede consultarse con facilidad utilizando herramientas de procesamiento de XML estándar como XSLT y XQuery. Sin embargo, esto no significa que las herramientas de procesamiento de XML estándar se pueden aplicar fácilmente a los archivos de XTM en los casos en los que se requiere una combinación.

A pesar de los problemas de la combinación, la sintaxis XTM cumple con la necesidad básica de permitir el intercambio entre aplicaciones de procesamiento de mapas conceptuales a las que se someten. Además, la sintaxis y las normas de combinación en conjunto son lo suficientemente flexibles aún para permitir que las partes de un mapa conceptual sean serializadas en documentos XTM por separado y se recombinen posteriormente por medio de la combinación [3].

Tal como lo hemos demostrado con optimismo hasta el momento, las normas de Mapas Conceptuales proporcionan una arquitectura de base muy flexible para una gran variedad de aplicaciones de administración de información y conocimiento. Esta flexibilidad puede llevar a confusión y una constante reinvención de enfoques de modelado básicos. Para referirnos a este tema, recomendamos el desarrollo y uso de patrones dentro de las aplicaciones de mapas conceptuales. Dividimos los patrones en dos grandes categorías: Patrones de Diseño de Mapas Conceptuales utilizados para modelar datos de mapas conceptuales; y Patrones de Aplicación de Mapas Conceptuales que son los patrones arquitectónicos que utilizan los sistemas de procesamiento de mapas conceptuales.

El concepto básico de los Patrones de Diseño de los Mapas Conceptuales adopta gran cantidad de los patrones de diseño de la ingeniería del software. Un patrón de diseño de mapa conceptual proporciona una ontología orientada y reutilizable que trata un único tema. Sin embargo, existen algunas diferencias importantes.

- Un patrón de diseño de mapa conceptual puede ser más preceptivo que un patrón de diseño de software, ya que debe especificar el identificador del tema URIs para los temas claves que utiliza el patrón. De este modo, cada implementación de un patrón en particular en un mapa conceptual puede reconocerse instantáneamente ante la presencia de temas que poseen URIs que especifica el patrón.
- Ya que un mapa conceptual está compuesto simplemente de datos, los comportamientos relacionados con el patrón de diseño de mapas conceptuales no se implementan en el mapa conceptual en sí mismo sino en el software de procesamiento que hace uso de los datos del mapa conceptual. Algunos patrones de diseño pueden prescribir un conjunto en particular de comportamientos para las aplicaciones de procesamiento; otros describen sólo el modelo de datos y dejan pendiente la forma en la que la aplicación lo procesa.

Árbol de temas

Algunos patrones básicos que se han tomado de Biblioteconomía han sido definidos por uno de los autores y admitidos por diversas aplicaciones de procesamiento de mapas conceptuales [4]. Estos patrones incluyen patrones para clasificación jerárquica y facetada. Aquí se muestra un ejemplo de uno de estos patrones.

El patrón de clasificación jerárquica utiliza una propiedad para modelar muy útil de los mapas conceptuales y es que cada tipo de tema, asociación y ocurrencia es en sí mismo un tema.

Esta función permite que la ontología de un mapa conceptual se anote utilizando la misma estructura que se utiliza para introducir datos en la ontología en sí misma, y se puede utilizar para producir grandes efectos en los patrones de diseño, permitiendo que una ontología de mapa conceptual existente sea anotada con el patrón "metaontología" en vez de ser modificada.

Este patrón permite que una aplicación procese un conjunto de asociaciones entre temas de manera tal que representen una jerarquía. Por ejemplo, puede mostrar los temas organizados en un diagrama de árbol.

Los patrones de aplicación de mapas conceptuales proporcionan patrones arquitectónicos de alto nivel y principalmente se concentran en la integración de los sistemas de procesamiento de mapas conceptuales con otros sistemas de datos y aplicaciones. Estos patrones incluyen patrones para representar información desde sistemas de datos externos como por ejemplo datos de mapas conceptuales; patrones para importar información desde sistemas de datos externos; y patrones para exportar y mostrar los datos de los mapas conceptuales.

Analizaremos las aplicaciones de los mapas conceptuales con más detalles en un próximo artículo.

En el momento de redacción de este documento, ISO sigue trabajando en la norma para Mapas Conceptuales en sí misma así como también en el conjunto de normas complementarias.

A pesar de que ISO/IEC 13250 ha estado en revisión, lo más importante de la norma ha permanecido sin cambios desde 1999 -un buen grado de estabilidad comparado con algunas normas de Internet. Sin embargo, el comité de ISO ha decidido que la próxima versión de la norma contendrá una revisión importante respecto de la forma en que se presenta y revisiones menores en la norma en sí misma.

La norma ISO/IEC 13250 debe ser dividida en varias partes separadas: una introducción no normativa, una descripción formal que fundamente los modelos de datos de los mapas conceptuales, una sintaxis de intercambio basada en XML/Xlink con una descripción del proceso de deserialización de la sintaxis en una instancia del modelo de datos y serialización del modelos de datos en un documento conforme a la sintaxis de intercambio;

Patrón de clasificación jerárquica

Definición del problema. Muchos sistemas de clasificación constan de una o más jerarquías de temas. Existe una gran cantidad de relaciones jerárquicas diferentes "parte-todo", "general-específico", etc. A pesar de que estas relaciones pueden ser diferentes, la semántica jerárquica permanece igual. Una aplicación que trata con tipos de relaciones jerárquicas múltiples requiere de una forma para identificar todos esos tipos.

Descripción del patrón. El patrón presentado aquí para modelar un sistema de clasificación jerárquica utiliza un tema que representa cada clase en el sistema. Entonces, la jerarquía se modela al crear asociaciones entre clases subordinadas y superordinadas. Sin embargo, existe una gran variedad de relaciones jerárquicas diferentes. Por esta razón, el tipo de asociación entre clases subordinadas y superordinadas no está definido por estos patrones. En cambio, este patrón define todos estos tipos y todos los tipos de funciones ya sean subordinadas o superordinadas.

El otro punto es la forma de relacionar los temas clasificados por este sistema (instancias) con los temas que representan las clases. Si una instancia es representada por un tema, entonces debe realizarse una asociación entre el tema que representa la clase y el tema que representa la instancia. A tal fin, se introducen los tipos de temas para representar la clasificación de una instancia ("clasificada como") y las funciones que cumplen en esta relación ("Clasificación" e "Instancia"). Si la instancia no se representa como un tema, entonces se debe utilizar una ocurrencia, caso en el que el tipo de "Instancia" puede utilizarse como un tipo de ocurrencia en vez de un tipo de función de asociación

Análisis. Este patrón crea el medio para marcar un tipo de asociación como una relación jerárquica e indicar cuál es la función superordinada y cuál la subordinada. Esto puede realizarse de forma externa al mapa conceptual que define la asociación y tipos de funciones, lo que permite que un mapa conceptual pre-existente pueda integrarse sin la necesidad de editarlo.

La semántica de clasificación de los tipos "clasificada como", "Clasificación" e "Instancia" pueden omitirse en este patrón cuando la clasificación no es el propósito de la jerarquía. Por este motivo, estos temas se definen en un conjunto PSI (con una base URI diferente) separados de la jerarquía que los define.

PSIs para patrones de clasificación jerárquica.

Los patrones de clasificación jerárquica utilizan los siguientes PSIs

- Tipo de relación jerárquica

http://www.techquila.com/psi/hierarchy/#hierarchical-relation-type

Tipo de función de asociación. Las asociaciones que son tipificadas por un tema que es una instancia de este tipo representan una relación parent-child (de subordinación) entre dos o más temas.

- Tipo de función superordinada

http://www.techquila.com/psi/hierarchy/#superordinate-role-type

Tipo de función de asociación. Si cumplen una función tipificada por una instancia de este tipo en una asociación del tipo de Relación Jerárquica representan el elemento superior en la jerarquía.

- Tipo de función subordinada

http://www.techquila.com/psi/hierarchy/#subordinate-role-type

Tipo de función de asociación. Si cumplen una función tipificada por una instancia de este tipo en una asociación del tipo de Relación Jerárquica representan el elemento subordinado en la jerarquía.

- Clasificada como

http://www.techquila.com/psi/classification/#classified-as

Tipo de asociación que afirma la relación entre el tema que representa una clase en un sistema de clasificación (cumple la función de Clasificación) y uno o más temas que representan instancias de esta clase (cumple la función de Instancia).

- Clasificación

http://www.techquila.com/psi/classification/#classification

Función que cumple un tema que representa una clase en un sistema de clasificación.

- Instancia

http://www.techquila.com/psi/classification/#instance

Función que cumple un tema que representa un asunto clasificado según un sistema de clasificación.

y un algoritmo de canonicalización para que el modelo de datos pueda utilizarse en la evaluación de conformidad de los mapas conceptuales. Se espera que esta organización elabore normas de fácil uso para el usuario y que agregue características que originalmente faltaban y se consideran importantes para los nuevos avances (en especial la especificación del modelo formal y el algoritmo de canonicalización).

Los cambios a la norma incluyen la capacidad para aplicar los tipos de datos a los valores de ocurrencias, incluyendo la capacidad para insertar XML; capacidad para manifestar un subconjunto de nombres de un tema como nombres que se utilizarán para determinar la identidad de un tema; un modelo más claro de ámbito; y una definición de la sintaxis de intercambio en W3C XML Schema y Relax-NG, así como también XML DTD.

Tareas por cumplir

Además de los cambios para ISO/IEC 13250, el comité ha comenzado a trabajar sobre dos normas complementarias. ISO/IEC 18408: Topic Maps Query Language (TMQL) definirá un lenguaje para consultar el modelo de datos de mapa conceptual, que permita seleccionar los modelos de mapas conceptuales (como temas y asociaciones) y los datos contenidos en ellos (nombre del tema o valores de ocurrencia, por ejemplo).

ISO/IEC 19756: Topic Maps Constraint Language (TMCL) define el lenguaje del sistema para los mapas conceptuales que permitirán al autor del sistema limitar los modelos que pueden aparecer en un mapa conceptual y la manera que deben relacionarse entre ellos. Al igual que con XML un un lenguaje del sistema para los mapas conceptuales permite la validación y también aplicaciones de edición orientadas al servicio más inteligentes.

En la actualidad estas dos normas se encuentran en las primeras etapas de desarrollo con requerimientos definidos y en el caso de TMQL, se ha creado una propuesta inicial para el lenguaje.

Pueden seguirse los avances sobre la norma central y los lenguajes de restricción y consulta en la página en Internet de Mapas Conceptuales de ISO [5].

Síntesis

Este artículo presenta el paradigma de los mapas conceptuales en el contexto de la norma ISO. Planteamos los componentes principales del modelo de mapa conceptual, mostrando la manera en la que los componentes de procesamiento estándar de combinación de ámbitos semánticos y temas le otorgan poder adicional a este modelo.

En un próximo artículo plantearemos algunos casos de uso concreto para los mapas conceptuales y mostraremos la forma en la que el modelo de mapa conceptual puede utilizarse para tratar la organización de la información y las necesidades de intercambio del entorno de las empresas modernas.

Notas al pie

1.La palabra "ontología" en este contexto significa el sistema de tipos de temas, ocurrencias y asociaciones que juntos definen las clases de cosas y relaciones entre las cosas que son documentadas por un mapa conceptual.

Referencias

1.Biezunski M., Newcomb S., Pepper S. (ed.). ISO/IEC 13250:2002, Topic Maps [online]. ISO. Formato PDF.

2.Moore G., Pepper S. (ed.), XML Topic Maps (XTM) 1.0 [online], TopicMaps.Org. Formato HTML.

3.Ahmed K., TMShare-Topic Map Fragment Exchange in a Peer-To-Peer Application. Formato HTML.

4.Ahmed K., Topic Map Design Patterns for Information Architecture. Formato HTML.

5.http://www.isotopicmaps.org/

Sobre los autores

Kal Ahmed ha trabajado durante 10 años en la administración de la información en SGML y XML information, desempeñándose tanto en el desarrollo de software como en la consultoría. Es muy conocido en la comunidad de mapas conceptuales debido a su trabajo sobre el Juego de Herramientas de Código Abierto en Java para mapas conceptuales, TM4J y por sus contribuciones en el desarrollo de la norma ISO. Kal ha publicado varios artículos sobre mapas conceptuales y temas relacionados con estos y es habitual orador en conferencias.

En la actualidad es co-fundador de Networked Planet Limited, una empresa que desarrolla herramientas para la construcción de mapas conceptuales y aplicaciones basadas en estos para la plataforma .NET.

Graham Moore ha trabajado durante ocho años como desarrollador, investigador y asesor en las áreas de administración de la información, contenido y conocimiento. Ha desempeñado una función protagónica como CTO de STEP, Vice President Research & Development empolis GmbH and Chief Scientist Ontopia AS. Ha sido responsable del desarrollo de productos de administración del conocimiento incluyendo K42 Topic Map Engine, X2X Link Management Engine y e:kms knowledge suite. Graham es co-editor de XTM 1.0 XML Topic Maps standard y ISO13250-1 y -2 (Topic Map Data Model and Syntax), también es co-editor de TMCL (Topic Map Constraint Language).

En la actualidad Graham es co-fundador de Networked Planet Limited.

Metrópolis y Gobierno de SOA

Richard Veryard y Philip Boxer

Introducción

La tecnología determinante en el último milenio fue el reloj, y el principio tecnológico dominante ha sido ahorrar tiempo para lograr que las cosas fueran más rápidas o mejores que antes.

Los relojes mecánicos modernos con péndulo se han inventado alrededor de mil años atrás. Por lo general, se atribuye la invención al Papa Silvestre II, quien se ha consumado como matemático y científico antes de ser papa. Según la Regla de San Benito, los monasterios medievales utilizaban el reloj para regular el trabajo y las oraciones. Lewis Mumford investigó que los orígenes de la Revolución Industrial se remontan a la Regla de San Benito y a la dominación del reloj. La película Tiempos Modernos de Charles Chaplin muestra (de manera exagerada) el poder incesante del reloj sobre un trabajador de una cadena de producción. A fines del siglo veinte, la ingeniería de procesos empesariales se centró en reducir el ciclo del tiempo, eliminando la espera. El slogan clave: Justo a tiempo.

Por supuesto, es muy pronto para decir cuál será la tecnología que distinguirá el próximo milenio. Pero existen aún algunos signos de un cambio del énfasis en el tiempo hacia el énfasis en la diferencia. Internet, por ejemplo, crea nuevas clases de diferencia en la relación entre las personas y las organizaciones; las empresas operan como nodos diferenciados o grupos dentro de un ecosistema en red complejo; la cohesión institucional y social está organizada contra las coordenadas de dimensiones abstractas complejas de diferencia.

La comprensión de la complejidad en sí misma se basa en reconocer que una vez que se ha ido más allá de un umbral determinado de diferencia en el comportamiento de los sistemas, se torna imposible predecir su comportamiento compuesto. Por lo tanto, en la economía de servicio, esperamos que surjan sistemas orientados al servicio que sean cada vez más grandes y complejos, pero que también tengan capacidad de funcionamient o cada vez más diferenciado. Como veremos, éste es uno de los desafíos claves de la Arquitectura Orientada a Servicios (SOA).

La ciudad o metrópolis es otro gran sistema complejo que muchos de nosotros encontramos en nuestra vida cotidiana y que experimentamos como diferenciado en su comportamiento. El conocimiento de las ciudades hace de la metrópolis un buen lugar de comienzo para lograr el mejor enfoque en la construcción y control de estos grandes sistemas complejos. Además, algunos de los temas claves para el diseño y control de estos grandes sistemas complejos orientados al servicio surgen también en el campo del diseño urbano, en el cual han sido debatidos por décadas (a pesar de que no se ha llegado a ningún consenso).

Resulta que el aporte de peso más reciente al debate sobre arquitectura física y diseño urbano ha venido de Christopher Alexander, quien finalmente publicó su tan esperado trabajo sobre la Naturaleza del Orden. Alexander ha tenido una profunda influencia sobre la ingeniería del software por varios años -uno de sus primeros trabajos sobre la Síntesis de la Forma ha influenciado los métodos estructurados de Yourdon y deMarco, mientras que gran cantidad de ingenieros del software han aceptado con entusiasmo su trabajo posterior sobre Patrones, en especial en el mundo OO.

De las ciudades a SOA

En el siglo XX, Lewis Mumford y Jane Jacobs fueron dos de los mejores escritores sobre la naturaleza de las ciudades. Mumford pensaba que una ciudad bien ordenada necesitaba una planificación e infraestructura central, mientras que Jacobs adoptó una posición más anarquista.

A continuación se exponen algunos de los asuntos planteados en el debate de la ciudad [1]:

Adaptabilidad. Durante el siglo XIX en Inglaterra, Manchester se adaptó muy bien a la industria del algodón, pero no logró adaptarse a movimientos de industrialización subsiguientes.

Cuadro 2. Preguntas de Pat Helland sobre la administración de SOA

(En contraste con las iniciativas de una ciudad metropolitana...) las iniciativas de IT no están muy desarrolladas.	**Las empresas pueden aprender mucho al observar la manera en que las ciudades administran el difícil proceso de asignación de recursos.**
¿Quién toma las decisiones más difíciles en IT? El CEO, el CIO, los líderes de la unidad empresarial, expertos informáticos o tal vez comités?	¿Qué propuestas se proyectan para autofinanciarse?
¿Se establecen las prioridades basadas en los costos, flexibilidad o en la explotación de activos?	¿Cuál es el tiempo programado y análisis de riesgo de estas proyecciones?
¿Qué es el éxito y cómo se mide?	¿Qué es sagrado para una empresa?
¿Se buscan reducciones de costos, transparencia en los procesos empresariales o ventaja competitiva?	¿Qué recursos permanecen una vez que se han consolidado estos esfuerzos?
	¿Qué balance a corto plazo, largo plazo e inversiones especulativas son correctos dentro de una cultura empresarial específica?
	Estos problemas son comunes para los entornos metropolitanos e IT

Mientras tanto, Birmingham se adaptó con mucha más facilidad, lo que le permitió acomodarse a una serie de innovaciones industriales.

Complejidad. Una ciudad posee una gran cantidad de interacciones sociales y comerciales. Una ciudad dinámica permite diversos niveles de interacción, así como también el surgimiento de importantes grupos y sub-grupos, los cuales forman una jerarquía abstracta.

Gobierno. La planificación de una ciudad requiere la organización de pequeños y grandes desarrollos, equilibrando la iniciativa y la autonomía local con la coherencia global.

Existen fuertes paralelos entre la planificación de una ciudad y la arquitectura orientada al servicio, por lo que resulta razonable traducir ideas y experiencias del diseño urbano al dominio SOA.

Distribución del diseño. Dentro de las diferentes organizaciones, se toman decisiones de diseño detalladas y cada una sigue sus prioridades (metas comerciales o políticas, por ejemplo).

Constancia del cambio. Los elementos del todo se rediseñan y reconfiguran constantemente así como también se agregan nuevos elementos. Las estructuras deben evolucionar de forma sólida.

Necesidad de una mejora progresiva. Cada incremento de un diseño debe no sólo realizar mejoras locales sino que también debe tener un efecto positivo en el todo.

Naturaleza recurrente de la arquitectura. Las tareas similares de diseño deben realizarse en diferentes escalas (niveles de granularidad).

En un reciente artículo del Microsoft Architects Journal, Pat Helland presenta un extenso análisis en el que compara la planificación y gestión de IT con la planificación y gestión de las ciudades. Demuestra que la administración de IT tiene mucho que aprender de la administración de las ciudades (Cuadro 2), y plantea algunos paralelismos interesantes entre el diseño urbano y SOA (Cuadro 3).

El artículo de Helland realiza el siguiente razonamiento

- El progreso requiere estandarización. (Según Helland, la gente ni siquiera se lavaba como es debido hasta que tuvo vestimenta uniforme.)
- La estandarización está asociada con la comoditización.
- La estandarización requiere concentración de poder (y si ello implica una distorsión patológica de las relaciones socioeconómicas, que así sea).
- La infraestructura requiere inversión central. (Puesto que podemos considerar la infraestructura como un acto de estandarización local, se deduce que debe comprender concentración de poder.)
- La inversión central preserva lo "sagrado".

Veamos los pasos del razonamiento de Helland en detalle.

Estandarización. El progreso debe suponer el enriquecimiento de la vida de las personas, aunque que varios actos de estandarización las empobrecen. No todos están dispuestos a considerar Wal-Mart como el epítome del progreso. Los sistemas dinámicos son heterogéneos.

La utopía de Helland parece ser relativamente homogénea. Los ciudadanos aún tienen el mismo olor. La eliminación de olores antisociales se obtiene mediante vestimentas normalizadas intercambiables (a pesar de que sin duda el jabón y el agua limpia también contribuyen).

Los métodos modernos de producción permiten la personalización masiva. Esto implica separar la producción en dos niveles -un nivel homogéneo de producción masiva y un nivel heterogéneo de personalización.

En el diseño urbano, lo estandarizado es aquello que va por debajo del pavimento -suministros normales como el agua y la energía, por ejemplo. Todo lo humano va por arriba del pavimento. Decidir que debe/debería ir por debajo del pavimento, es asunto de la gobernabilidad de la ciudad.

Cuadro 4. La Naturaleza del Orden. Manifiesto de Christopher AlexanderHelland

Naturaleza del Orden: Christopher Alexander

La Naturaleza del Orden (en sus cuatro volúmenes completos) posee repercusiones en la participación de las personas al diseñar construcciones y en las formas detalladas en las que esta participación puede resultar exitosa o no. Posee implicaciones para el flujo de dinero, así como también para la manipulación de detalles de la arquitectura y para una integración exitosa de la ingeniería estructural dentro de un marco de diseño. ... Afecta de manera virtual cada parte de la profesión conocida en la actualidad como arquitectura e indica la necesidad de un cambio en casi todas sus partes. No hay duda que, bajo el impacto de esta teoría, la arquitectura cambiará rigurosamente y para mejor.

Por varias décadas, Christopher Alexander ha investigado las alternativas para la práctica de la arquitectura convencional. Su último trabajo, La Naturaleza del Orden, se publicó el año pasado.

Una cuestión de arquitectura

La articulación en capas de un sistema complejo (uno homogéneo, uno heterogéneo) es una pregunta de la arquitectura. Obviamente existen compañías con un importante interés comercial en la pregunta. Por lo tanto, no está mal sospechar que una articulación se presenta como dada en algunas tradiciones históricas o imperativos técnicos. [3]. Las opiniones de arquitectura en apariencia puramente tecnocráticas ocultan con frecuencia un plan comercial. Una de las funciones de la administración es mantener un "campo de juego nivelado" entre los diferentes programas comerciales. Por este motivo, quienes regulan muchas veces aspiran a intervenir a nivel arquitectónico.

Es razonable que algo parezca homogéneo desde el punto de vista de la oferta y heterogéneo desde el punto de vista de la demanda, o viceversa, por lo tanto, el límite es en sí mismo relativo a alguna formulación de oferta/demanda. La tecnología crea divisiones homogéneas/heterogéneas constantemente -por ejemplo, la voz en la tecnología IP crea una división entre los dispositivos que determinan si un flujo de bits representa voz o información (por lo tanto, considera el tráfico como heterogéneo), y los dispositivos que no (y así, considera el tráfico como homogéneo).

Un punto de partida posible para la articulación de capas es la tasa diferencial de cambio. Parecería tener sentido estandarizar y regular la capa de movimiento lento, y permitir una mayor diversidad en la capa de movimiento rápido. Pero una perspectiva ecológica nos dice que el movimiento lento domina el movimiento rápido. Esto comprende una nueva función para la arquitectura:

- Mantener la estratificación adecuada de capas y conexión entre los elementos que se encuentran en y a través de ellas.
- Operar a un nivel más alto de abstracción, implementando políticas de diseño basadas en la evidencia.

Los enfoques actuales para definir arquitecturas tal vez no funcionen muy bien, aún para los bits homogéneos. Desde luego no funcionan para los bits heterogéneos, y tampoco colaboran al definir el límite entre la capa(s) homogénea y la capa(s) heterogénea. Una consecuencia de nuestro argumento es que el límite

entre lo homogéneo y lo heterogéneo (tal como se explica más arriba) es un foco de atención adecuado para la arquitectura. Más adelante veremos lo que sucede en su aplicación práctica.

Comoditización. No podemos evitar la comoditización de nuestras vidas -pero debemos estar atentos a sus peligros [4].

Afortunadamente, ya no es necesario soportar un software estándar. El software situado-software diseñado en y para una situación social o contexto en particular- resiste la presión de la ingeniería del software tradicional hacia la generalización, y por lo visto, no tiene en cuenta la economía de escala o ámbito. En cambio, funciona solo dentro de un sistema socio-técnico colaborador (la "comunidad"); las condiciones para el éxito del software (incluidos el significado y la confianza) son creadas en conjunto por los miembros de la comunidad.

Una de las primeras formas de software situado fue la hoja de cálculo. Los usuarios avanzados creaban ellos mismos estructuras complicadas con Visicalc o Lotus 123 o Excel. Éstos eran esencialmente objetos no transferibles con algunas hipótesis ocultas, pero servían para un propósito útil dentro de un contexto determinado. Esto demuestra el hecho de que el software situado es asistido por herramientas y plataformas que le proporcionan el soporte generalizado.

Admite una gran diversidad

El éxito rotundo de la hoja de cálculo se debió al hecho de que cumplía una función útil, mientras que el usuario podía libremente crear un sentido específico según el contexto. Pero también, la hoja de cálculo era limitada por el hecho de que estos sentidos creados eran privados y no se documentaban, y los intentos de transformarlas en objetos compartidos por lo general fracasaron.

Aquí es donde aparecen las empresas productoras de software. Existe una gran oportunidad de producir herramientas de software capaces de admitir una gran variedad de demandas del usuario final. El Lenguaje Específico de Dominio (DSL) puede ser una forma de unir y mantener abierto el espacio entre lo general / público y el contexto específico/privado, así como también conservar la interacción dinámica entre la oferta y la demanda.

Tabla 2. Paralelismos claves entre las ciudades e IT según Pat Helland

Ciudades	Seminarios IT
Fábricas o edificios	Aplicaciones
Transporte	Comunicación
Mercaderías fabricadas	Información estructurada
Conjuntos fabricados	Empresas virtuales
Venta y Distribución	Proceso empresarial
Infraestructura urbana	Infraestructura IT
Administración de la ciudad	Administración IT

Esta dinámica debe estar orientada por la definición que crea el usuario final respecto de la relación entre los dominios y su empresa como un todo.

La economía de servicio es un ecosistema complejo. Las soluciones orientadas al servicio son esencialmente sistemas de sistemas, y su composición debe ser consciente de la teoría de los sistemas complejos. Para mantener la variedad de requisitos (y por lo tanto la supervivencia de los mejor adaptados) en estos ecosistemas, es necesaria la diversidad en todos los niveles de abstracción.

Concentración del Poder. El sistema económico de Wal-Mart es insostenible. Destruye la fabricación de pequeños comercios de los cuales depende gran parte de la vida urbana.

Las ciudades son en algún punto paradójicas. Por otro lado, una ciudad ya es en sí misma una concentración de vida humana. Pero los procesos de concentración son inestables y pueden resultar en formas urbanas muy disfuncionales.

Históricamente, las ciudades tenían murallas que mantenían alejados a los visitantes no deseados. En alguna otra parte, Helland ha defendido el modelo aislado de la informática. Pero aquí, parece prever un tejido social continuo, en el que se desdibujan los límites entre una ciudad y otra (al igual que Manchester se une a Salford).

Inversión central. El alegato de Helland respecto de la inversión central (posición de Lewis Mumford) proporciona justificación para el planeamiento e inversión central de la empresa en IT. Muchas pequeñas y grandes empresas tratan de imponer el planeamiento IT central. Sin embargo, en muchas organizaciones es una batalla que se pierde. La situación real de la industria IT surge de millones de pequeñas decisiones de compra, y está más cerca de la idea de compras anárquicas (posición de Jane Jacobs).

Salingaros se basa en Alexander para describir la forma de resolver los argumentos de Mumford y Jacobs -pero adoptando un enfoque que va más allá de un mero intento de reconciliar lo descendente con lo ascendente.

Una versión moderna de lo sagrado puede encontrarse en la noción de Borgmann sobre las cosas y prácticas centrales. La función del gobierno urbano/de sistemas sería crear/preservar un espacio en el que estas cosas y practicas centrales puedan desarrollarse y respetarse.

En una economía de demanda dinámica, la fuente de lo sagrado es la demanda. Esto contrasta con una lógica desde el punto de vista de la oferta basada en una presunción de demanda simétrica, en la cual los mercados están definidos para reflejar al proveedor, para que así, las formaciones de la demanda sean simétricas a las formaciones de la oferta.

Al hablar de empresas virtuales, Helland expone: "Debe considerarse el contexto en el cual se utilizará la parte. ¿Será el objetivo principal el peso o la durabilidad?" Helland sostiene que los estándares son la clave que permiten que los proveedores de componentes nivelen el costo de optimización a través de un

Tabla 3. Implicaciones de la asimetría - los tres dilemas

Simetría	Suposición	Consecuencias de la asimetría
Tecnología = Producto	La primera suposición simétrica es que los primeros tres desafíos están alineados. Por lo tanto, estos tres desafíos colapsan en una única dimensión definida por la arquitectura.	Con SOA, enfrentamos cada vez más empresas que son nada menos que plataformas de tecnología para otras empresas (desde Microsoft mismo hacia abajo). Por lo tanto, la alineación simple no funciona. En cambio son insertados en alguna forma de estratificación.
Negocio=Solución	Las normas y procedimientos empresariales que adopta el proveedor concuerdan con las formas en las que se utilizarán los servicios.	Se supone que el mantenimiento de los carrilles proporciona ferrocarriles seguros y confiables. En el Reino Unido, Network Rail (ex-Railtrack) toma servicios de entrada de las compañías de ingeniería y los transforma en servicios de salida para las compañías operativas de trenes. Esto ha probado que es muy difícil alinear los requerimientos de entrada con los requerimientos de salida.
Demanda del cliente=Experiencia del cliente	La utopía bancaria del procesamiento directo y completo se basa en la simetría, y los valores compartidos se encuentran todos a los largo de la cadena de valores.	Concuerda con la situación en la industria farmacéutica en la que el conjunto de relaciones de una compañía de drogas con GPs y los farmacéuticos es de naturaleza bastante diferente a las relaciones de GPs y los farmacéuticos con sus pacientes, a pesar de que la tendencia de las compañías de drogas es creer que esto es de otra manera.

mercado más amplio, y esto puede considerarse como la lógica de la oferta. No obstante, este tema va más allá de la estandarización (aquí se extiende hasta los modelos de componentes empresariales, y les permite ser encapsulados como capacidades de los componentes y organizados como parte de conjuntos más grandes de procesos) y establece la granularidad de las capacidades de los componentes con respecto a las otras. Es decir, el repertorio disponible de capacidades de los componentes alternativos. Esto también debe comprenderse desde el punto de vista de la demanda.

La fuerza orientadora

Al considerar el proceso empresarial, este argumento está extendido por analogía a la necesidad de estandarización e intercambiabilidad de la información y las operaciones: "las personas aceptan con gusto las cosas estándar y la personalización es poco frecuente y costosa. Pero el proceso empresarial todavía es en su mayor parte artesanal. Existen estándares escasos..." y por lo tanto el argumento busca la estandarización proporcionando una base de extensión de la lógica desde el punto de vista del proveedor mucho más profunda dentro de la provisión de servicios, en los que los procesos empresariales se tornan la fuerza impulsora que dicta la configuración y la forma de las aplicaciones" tan cierto como que Wal-Mart impulsa los estándares para muchos, muchos productos fabricados."

Esto no cumple con el desafío del ciclo de venta minorista mencionado anteriormente. [5]. El ciclo describe la aparición de una nueva forma de relación oferta-demanda (destino), la cual se expande para convertirse en una nueva forma de oferta en conjunto con las otras (comparación), antes de sean comoditizados (costo), y por último se fija en el contexto del usuario (costumbre/uso/hábito).

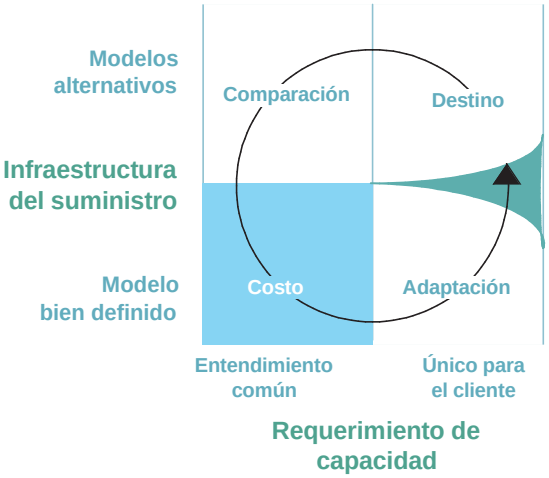
> "La ciudad o metrópolis es otro gran sistema complejo que muchos de nosotros encontramos en nuestra vida cotidiana y que experimentamos como diferenciado en su comportamiento."

Desde aquí, se prepara el campo para un nuevo ciclo (período de transición), y demás. Este ciclo es un proceso dinámico, en el cual el punto de vista de la oferta está en constante aprendizaje de nuevas formas en respuesta a una demanda en continua evolución -y nunca se satisface totalmente. La demanda asimétrica describe la demanda en su particular contexto de uso, y esta situación permanente de que "siempre hay algo más que desear" es un déficit estructural que siempre está allí conduciendo el desarrollo de los mercados. La comoditización es sólo la parte de la historia desde el punto de vista de la oferta, el problema real es la forma en la que la dinámica de la formación de la demanda en sí misma debe admitirse.

Según Alexander, un proceso de diseño convencional no puede producir grandes sistemas complejos -ya sea descendente o ascendente. Por el contrario, surgen de un proceso más amplio y en conjunto (evolutivo). El orden y la coherencia surgen de las normas que gobiernan este proceso evolutivo.

Este proceso evolutivo puede desglosarse en pasos diferenciados que pueden ya sea preservar la estructura e integridad, o destruirla. La estructura e integridad se articulan como un sistema recurrente de centros.

Figura 1. Requerimiento de capacidad



Los servicios pueden fácilmente considerarse como centros de valor [6]. También pueden estar constituidos dentro de servicios compuestos, con una organización (con un poco de suerte) de rendimiento coherente entre los niveles recurrentes.

Una empresa orientada al servicio puede entonces entenderse como una Red empresarial continua de servicios. Las propiedades de la arquitectura de esta empresa dependen de los numerosos procesos que colaboran y provocan su composición. Si estos procesos preservan su estructura de una forma adecuada, entonces la empresa puede llegar a ser cada vez más diferenciada y más integrada, sin perder coherencia.

Alexander es muy crítico respecto del gobierno convencional sobre la planificación de las ciudades y el diseño urbano, y sumamente crítico de los resultados inflexibles e inhumanos del diseño dirigista. Sus ideas respecto del diseño y del orden concuerdan, creemos, con las necesidades de composición en conjunto, tal como se resume en este artículo.

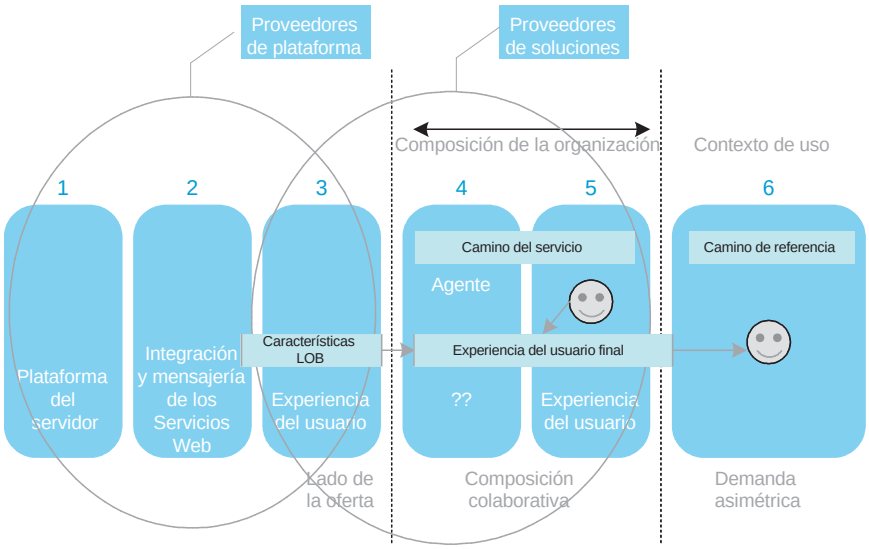
Implicaciones estructurales

Una empresa o cadena de valor está constituida en una estructura geométrica. En SOA, se diseña un negocio o cadena de valor como una red de servicios. Este es un patrón geométrico fuerte. Pero pueden existir varias geometrías de redes posibles capaces de satisfacer un determinado requisito de la empresa, las cuales se considera que satisfacen los principios de SOA. Por ejemplo: hub/spoke o punto a punto.

Una característica clave de SOA es la estratificación. Un proceso empresarial está compuesto por servicios desde un conjunto de servicios de más bajo nivel, conocidos como plataformas. Un buen ejemplo de una plataforma empresarial es el conjunto de servicios de venta minorista que ofrecen Amazon e eBay. Otros proveedores de servicios han creado más servicios de venta por menor/logística sobre la base de las plataformas de Amazon/eBay.

Cada plataforma a su vez se basa en servicios aún más bajos. En los niveles más bajos, pueden existir colecciones de servicios basados en IT, conocidos como Bus de Servicio de la Empresa (ESB). También pueden existir plataformas de servicios socio-técnicos, como centros de llamadas. Algunos de estos niveles bajos de la pila informática pueden parecer ser puramente servicios técnicos; sin embargo, una imagen más completa debe revelar la existencia de una organización IT que mantenga la plataforma, en otras palabras, esto también es una plataforma socio-técnica

Figura 2. Composición colaborativa de la Cadena de Valor de Seguros (basado sobre Microsoft IVC)7



que incluye sus administradores y programadores.

Por lo tanto, tenemos una geometría estratificada, en la cual una persona que enfrenta un problema en un nivel determinado se presenta ante un conjunto de servicios disponibles, que se forman dentro de una plataforma virtual. Esto puede pensarse como una pila de negocios, con una plataforma colocada arriba de la otra. Y mientras que los principios de SOA pueden proporcionar alguna orientación geométrica y mantener cierto patrón de geometría, aún se necesita un trabajo de diseño para determinar la forma de la geometría más adecuada para admitir el servicio solicitado. Este trabajo de diseño puede ser fácil cuando los requerimientos son triviales, pero se torna más difícil en tanto su complejidad aumenta.

En varias situaciones, el punto de vista de la oferta posee más variaciones que las que un diseñador humano (o grupo de diseño) puede adaptar. (Esto se caracteriza como una asimetría de la oferta, lo que exige un proceso de diseño asimétrico). Dadas las circunstancias, es necesario avanzar aún más y comenzar a pensar en soluciones de geometría variable, en las que la geometría en sí misma pueda adaptarse de acuerdo a las necesidades.

Por ejemplo, en el pasado suponíamos que la granularidad debía fijarse en el momento en el que se diseñaba. Pero podemos concebir que una plataforma de un Servicio Web detecta patrones de composición desde el punto de vista de la oferta, define nuevos servicios de componente de manera automática, describiéndolos y publicándolos en tiempo real, y notifica a posibles usuarios del nuevo servicio, que incluye un incentivo para cambiar a nuevas formas de administrarlos que ayudan a la composición desde el punto de vista de la oferta. Se puede formar un concepto de esta plataforma del Servicio Web analizando el contenido del mensaje de un determinado servicio y producir otro substituto que cuente con superficies de apoyo más pequeñas que satisfagan a la mayoría de los usuario de una forma más sofisticada.

El término configuración de los valores se utiliza para referirse a la distribución del costo, beneficio y conocimientos técnicos a través de un ecosistema de mercados complejos, como la industria de seguros, para un determinado nivel de riesgo. La tecnología (incluida SOA) influye la geometría del negocio, porque no sólo afecta los costos de transacción sino también el modo en el que los conocimientos técnicos pueden nivelarse en relación con la demanda. La forma de la configuración de los valores cambia (ya ha comenzado a cambiar) a partir de B2B, B2C, P2P y BPO. Las empresas que alguna vez ocuparon posiciones de mercado seguras pueden encontrar que su ventaja comercial está decayendo, o tal vez se encuentren aislados de sus anteriores clientes o cadenas de suministro.

Identificar Objetivos

Supongamos que una empresa de seguros posee los siguientes objetivos estratégicos:

- Rentabilidad, viabilidad a corto plazo. Brindar el mayor valor en servicios con la mejor relación costo-beneficio posible, utilizando servicios de entrada disponibles y tecnologías lo más eficaces posible, con el menor costo/riesgo de cambio.
- Adaptabilidad, viabilidad a mediano plazo. Comprender y responder a la demanda cambiante para los servicios de seguros y para las tendencias en el costo y el riesgo, tanto internamente como a través de la industria. Desarrollar y desplegar nuevos servicios para explotar las nuevas oportunidades empresariales y evitar las amenazas que puedan surgir.
- Supervivencia, viabilidad a largo plazo. Asegurar que la propuesta central de la empresa permanece válida, y no dejarse vencer por los más ágiles. Tomar una acción estratégica en relación a los cambios estructurales en la industria del seguro.

Si se realiza una zoometría de negocio para una empresa de seguros, es necesario considerar a la industria del seguro como un ecosistema que evoluciona. Se necesita un modelo ACTUAL del presente ecosistema (basado en gran parte sobre tecnologías pre-SOA) y un modelo FUTURO de un ecosistema evolutivo (basado sobre los efectos de SOA). Esperamos que el ecosistema pre-SOA se desarrolle en alguna forma de ecosistema post-SOA, a pesar de que no se tiene mucha idea de cuál de los posibles cambios sucederá primero. Para cumplir con los tres objetivos estratégicos, es necesario que la empresa de seguros explote el ecosistema pre-SOA, y se prepare para el ecosistema post-SOA.

Cabe observar que esta situación puede obligar a la empresa de seguros a implementar una geometría variable a través de su pila de negocios, tanto en la plataforma de organización como en la plataforma subyacente IT. De lo contrario, deberá operar ya sea de manera subóptima por un período prolongado de tiempo, o

incurrir en importantes costos de organización y costos IT cada vez que la industria avance otro paso hacia SOA. La geometría variable comprende la colaboración dinámica (composición en conjunto) entre una oferta eficiente y una demanda variable (asimétrica), tal como se muestra en la Figura 2 [7].

Asimetría significa que las formas de la demanda son cada vez más específicas para el contexto en el que surgen.

La primera asimetría implica separar la tecnología de la oferta de productos específicos.

Esto requiere la modelación de comportamientos posibles que puedan ser soportados (por lo tanto Microsoft o los fabricantes de automóviles deben utilizar un sistema modular para permitir conjuntos de usos tecnológicos).

La segunda asimetría requiere la separación de modelos empresariales que puedan organizar la oferta respecto de soluciones ofrecidas. Esto implica el modelado de formas posibles de geometrías de negocios (por lo tanto, el mantenimiento de ferrocarriles o los servicios de venta minorista deben utilizar un modelo de franquicia para permitir la variación en la organización de la empresa y así adaptar la variedad de formas en las que es necesario implementar un servicio).

Finalmente, la tercera asimetría necesita la separación de los diferentes contextos de uso. Esto requiere el modelado de formas posibles de demanda (de manera que los servicios financieros o de la atención están teniendo que considerar la forma en que las riquezas/condiciones son administradas de manera que respondan a las diferentes formas de contexto de uso).

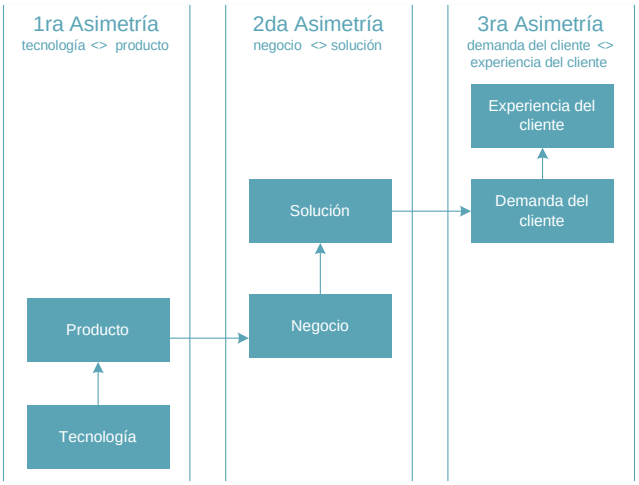
Superar la competencia

Estas asimetrías se sintetizan en la Figura 3 y es importante considerar lo que sucede si se ignoran. En la primera asimetría, significa definir el producto por la tecnología. Esto es típico de las primeras etapas de evolución de las nuevas tecnologías. (¿Recuerdan cómo solíamos utilizar los teléfonos celulares?). En la segunda asimetría, significa definir la solución para el cliente de acuerdo a la forma en la que la empresa está organizada. (¿Recuerdan la manera en la que las grandes empresas solían relacionarse con sus clientes antes de la aparición de CRM?). Y en la tercera asimetría, se supone que la solución al problema presentada por el cliente es lo que el cliente necesita en realidad. (¿Han recibido alguna vez de un doctor una indicación que resultó sólo tratar el síntoma?). El mayor impacto competitivo de SOA es que cambia lo que el proveedor puede permitirse ignorar desde la perspectiva del cliente.

Para obtener mejores capturas de las formas asimétricas de la demanda, una organización necesita liderazgo que le permita realizar dos cosas:

- Hacer llegar el poder a la periferia de la organización: Las personas en la periferia de la organización que tengan relación con la demanda asimétrica deben poder decidir cuál es el modelo empresarial más apropiado para capturar la demanda.
- Desarrollar una infraestructura dinámica: proporcionando servicios empresariales que puedan organizarse y componerse en la periferia en respuesta a las formas particulares de demanda a las cuales apuntan. Entonces, esto permite que el punto de vista de la oferta de una empresa extraiga economías de escala o de
ámbito al proporcionar respaldo entre los múltiples modelos empresariales.

Figura 3. Las tres asimetrías de la demanda



"Es razonable que algo parezca homogéneo desde el punto de vista de la oferta y heterogéneo desde el punto de vista de la demanda, o viceversa"

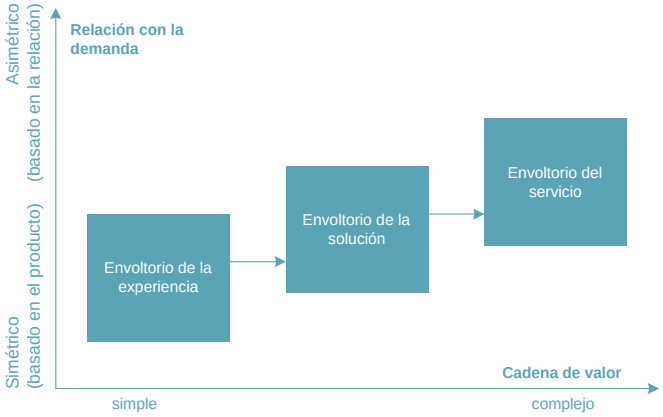
La asimetría en la variedad de condiciones que poseen las personas y en las formas en las que se manifiestan en las vidas de las personas a través del paso del tiempo, está siempre en aumento. Mientras tanto, los hospitales y las clínicas deben ser cada vez más eficientes en la forma que administran caminos de atención en particular.

Este desafío se manifiesta en particular en relación a condiciones que son crónicas. Por lo tanto, los pacientes no presentan condiciones que son adecuadas para los tratamientos disponibles, por lo que la organización de sistemas precisos de atención en torno a tratamientos de condiciones crónicas puede ser exorbitablemente costosa. Los pacientes sufren condiciones fundamentalmente asimétricas desde el punto de vista de los especialistas de la medicina que las tratan. Por ejemplo, las cirugías agudas pueden por lo general remontarse a un fracaso anterior para proporcionar un tratamiento preventivo oportuno.

Lograr el poder a la vanguardia en el cuidado de la salud significa que los recursos se dispongan de conformidad con los que el paciente necesita, y que los médicos posean la capacidad de formar un plan de tratamiento que sea específico para las condiciones del paciente.

Esto genera un doble desafío para el suministro del cuidado de la salud. No sólo se debe aumentar la flexibilidad con la que sus servicios de componentes deben ponerse a disposición de los pacientes, sino que también los médicos deben tener una participación mucho mayor al formular estrategias permanentes de suministro de atención médica que puedan adaptarse con el tiempo a la condición del paciente, y por el suministro del cual pueden ser responsables. Si esto se logra, el costo total del cuidado se reduce.

Figura 4. Esquematización de la Oferta-Demanda



Abordad la asimetría

La agencia encargada de compras y provisiones médicas (PASA) era responsable del suministro de equipos para el servicio clínico. Estaban preocupados por los efectos que produciría el minimizar los costos de estos suministros -redujeron la inversión en la industria y el círculo vicioso del deterioro de la calidad del servicio clínico en sí mismo. Decidieron que el diseño de simetría convencional no funcionaba para ellos; en un intento por mejorar la calidad del servicio clínico, optaron por considerar el desarrollo de un enfoque que se remitiera a la naturaleza asimétrica de las exigencias clínicas. Llevaron a cabo un estudio piloto inicial para establecer la factibilidad de un proceso de diseño asimétrico.

PASA admitió la necesidad de remitirse a las exigencias de la clínica y establecer la mejor forma de satisfacer sus necesidades. Dentro del contexto de uso, podían entonces remitirse al problema del costo de la oferta.

La agencia de modernización estableció un proceso nacional. El proceso organizó seis proyectos preliminares, cada uno de los cuales estaba orientado a establecer la forma en la que el cambio se haría efectivo en cada contexto. La tarea del proyecto nacional era asegurar el apoyo y financiamiento a largo plazo para el proceso de cambio en vista del aprendizaje y los resultados de lo explorado. Por último, esto comprendía el modelado del impacto de los cambios regionales y nacionales sobre los presupuestos de NHS y Servicios Sociales.

Desde el punto de vista del suministro a los centros de salud, la modelación desde el lado de la demanda utilizó los circuitos de derivación y los servicios que ofrecía cada clínica en respuesta a las demandas que surgían de estas vías de derivación. La modelación desde el punto de vista de la demanda se centró en la organización de la clínica en sí, junto con su uso de suministros para establecer la forma en la que uno se alineaba con el otro. El punto en el que este "corte" se interponía entre el lado de la oferta y el lado de la demanda dependía del cliente y de su agenda de cambios. No obstante, al examinar las vías de derivación y las formas en particular en las que habían sido "colonizados" por los proveedores, surgieron más preguntas respecto de la organización de la atención primaria en sí misma. Sin embargo, estas preguntas se dejaron para que las interprete un sistema de cliente diferente en un futuro -y la que deberían remitirse a los intereses de las Autoridades Sanitarias estratégicas.

El desafío clave residía en otorgarle a los clínicos el "control del diseño" sobre la manera en la que las clínicas operaban (es decir, el poder a la vanguardia). Algo fundamental para esto era comprender desde el punto de vista de la oferta la naturaleza de multiepisodios crónicos de las condiciones que se trataban en la clínica, y desde el punto de vista de la oferta, los procesos de delegar y/o trabajar en equipo las responsabilidades de la clínica respecto de las condiciones de los pacientes. Los clínicos no poseían los medios para definir la diferencias de lo primero y manejar las complejidades de lo segundo. Además, sin los medios para realizar estas cosas, no existía manera práctica de responsabilizar a los médicos por los resultados clínicos del centro. La solución era construir una plataforma de informe que pudiera respaldar la práctica de estas cosas.

Este modelado comprendía la definición de los caminos referidos y sus características reales. A partir de esto surgió el requisito para cambiar la manera en la que la clínica se relacionaba con la demanda.

El modelado implicaba definir las propuestas de servicios y los modelos empresariales clínicos necesarios. En el primer caso, significaba establecer protocolos de episodio para las diferentes condiciones, y en el ultimo significaba cambiar los procesos del flujo de trabajo entre la clínica y los procesos front y back office que lo soportan. No se intentó reajustar las infraestructuras de la oferta, ya que se obtenían beneficios considerables simplemente a través de la forma en que se administraba la alineación de los proveedores para la oferta. La plataforma de informe construida proporcionó los medios para lograrlo. La plataforma permitió que la clínica defina sus propios protocolos de tratamiento en relación a sus propias definiciones de las condiciones referidas. El sistema informático podía levantar información sobre los pacientes, citas, y demás, fuera del entorno NHS.

Esquematización de la oferta-demanda

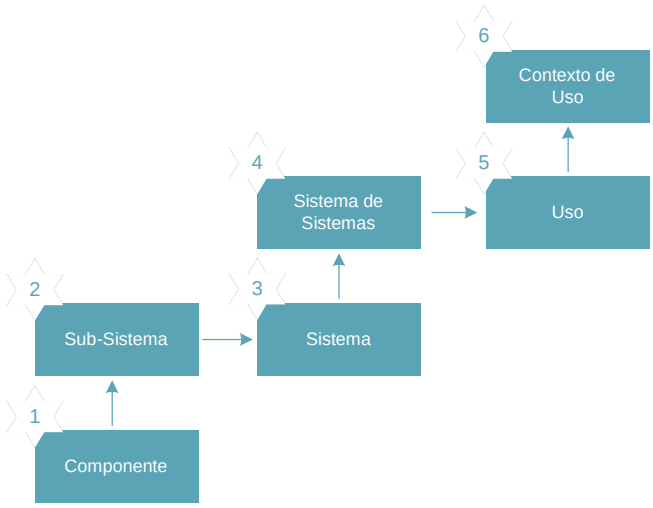
La Figura 4 muestra una esquematización típica para las empresas basadas en el servicio en la medida que responden al impacto competitivo de SOA. La empresa resuelve una forma de asimetría por vez:

1.Utiliza un "envoltorio del servicio" para desacoplar el producto de la tecnología. Esto incluye la definición de un modelo de objeto diferente desde el punto de vista de la demanda, separando la información "sin procesar" de la información "procesada".

2.Utiliza un "envoltorio de la solución" para desacoplar la solución del negocio. Esto incluye la definición de diferentes normas para la demanda, separando la lógica del negocio de la organización de las diferentes soluciones.

3.Utiliza un "envoltorio de la experiencia" para desacoplar la experiencia del cliente en curso de las soluciones en particular presentadas por el cliente en cualquier momento. Esto incluye nuevas formas de modelado del proceso para comprender la experiencia de soluciones del cliente dentro de un contexto de uso en particular.

Figura 5. Estratificación de los modelos



El punto a considerar acerca de esta evolución es que enfrenta la oferta con la necesidad de administrar la complejidad creciente (y la concurrencia) de la forma en que la cadena de valor se relaciona al cliente. De ahí la importancia competitiva de SOA.

En la primera simetría, la empresa sólo necesita un modelo, ya que la demanda puede deducirse de la oferta (o eso parece hasta el momento!). Cuando se rompe esta simetría, se necesitan dos modelos: uno para administrar la tecnología, y el otro para administrar la empresa. ¿Cuántas veces un inversor de capital de riesgo le ha enseñado esto a un novato en los negocios?

Si se realiza el potencial completo de una propuesta empresarial, también debe romperse la segunda simetría (esto es algo que toda persona que posee un Master en Administración de Empresas debe aprender). Ahora, existen tres modelos: uno para administrar la tecnología, uno para administrar la empresa, y uno para administrar la solución de la empresa. Por lo tanto, en el ejemplo del ferrocarril en el Cuadro 6, es necesario separar la forma en la que se brinda el servicio debido a que no es posible obtener niveles del servicio de salida para las compañías operativas de ferrocarriles simplemente desde los requerimientos del servicio de entrada de la infraestructura de los ferrocarriles (es decir, al menos que se esté preparado para ofrecer el más alto nivel de calidad bajo todos los conceptos).

¿Pero qué efecto surge al romper la tercera simetría? Lo que CRM nos dice es que la "solución del mercado" es también una relación con el cliente. Pero al romper la tercera simetría debemos aprender que cada cliente incorpora la solución en un contexto de uso que es diferente. Por lo tanto, en el cuidado de la salud, esto no es más una cuestión de brindar determinados tratamientos con la mejor relación costo-beneficio, sino también brindar el tratamiento adecuado, en el momento adecuado dentro del contexto de una condición de paciente que evoluciona - como la tercera edad, por ejemplo!

De esta forma, en el caso farmacéutico, no se puede obtener el conocimiento de la salida (lo que GPs y los farmacéuticos necesitan saber respecto de la condición del paciente) desde el conocimiento del lado de la entrada. Se necesitan al menos dos modelos ya que no se pueden deducir los efectos del tratamiento simplemente desde el conocimiento de las interacciones entre la prescripción y la situación tal como se la presenta cuando se prescribe -requiere un modelo de los efectos dinámicos de la prescripción dentro de su contexto de uso.

Entonces, en general, romper las simetrías requiere de la separación de seis diferentes niveles de modelos (ver Figura 5). A pesar de esto, cada uno de los estratos puede contener diferentes niveles dentro de el, la agilidad de la infraestructura dependerá de la cantidad de variabilidad que permitan en la geometría de cómo pueden componerse.

¿Qué es entonces lo que todo esto requiere del "arquitecto IT" a favor de estas nuevas formas de dinámica. Nuestro argumento más importante es que no existe un único nivel de granularidad. Existe un dilema de oferta versus demanda al cual es necesario que se remita un método orgánico (Véase Alexander) en el que los niveles de estratificación en los cuales se realiza un cambio deben siempre también remitirse a su impacto sobre los niveles superiores e inferiores. Como resultado, se debe realizar un intercambio entre dos tipos de presión a un nivel determinado de estratificación, uno que estandarice y el otro que requiera mayor diversidad. Esto impone un tipo diferente de demanda sobre la función para la arquitectura.

Tal vez pensamos que la función de la arquitectura consiste en emitir ciertos tipos de juicio (estructural) [8]. Vemos a la arquitectura como algo que surge de un proceso en conjunto, que puede (pero no debe) involucrar personas llamadas "arquitectos" en su profesión. Tal vez consideramos a estas personas arquitectos profesionales, a pesar de que en la ingeniería del software y de los sistemas no existe un organismo que otorgue este estatus oficial a la profesión. Estos juicios pertenecen a un régimen de diseño asimétrico.

Podemos entonces separar los productos y procesos de la actividad de la arquitectura de las responsabilidades profesionales de los Arquitectos calificados de forma adecuada. Las propiedades estructurales de una ciudad dependen del proceso complejo en conjunto, en el cual los juicios profesionales de los arquitectos se oponen a un gran número de intereses políticos y comerciales. Los arquitectos profesionales rara vez dominan el diseño de las ciudades, a pesar de que a veces ejercen cierta influencia.

Y tal vez observemos una situación similar dentro de IT. A pesar de que existen numerosas personas en la industria IT que se llaman a sí mismos "arquitectos", las propiedades estructurales de los extensos sistemas IT por lo general dependen en gran medida de las decisiones que se toman en otro lugar. Por ejemplo, el grado de acoplamiento entre dos módulos puede tener un impacto importante sobre la cohesión y la flexibilidad estructural de un gran sistema, pero esto puede estar determinado por elecciones de desarrollo de un nivel bastante bajo que ni siquiera son visibles para los arquitectos del software.

En muchas organizaciones, los arquitectos del software pueden ser desplazados y marginados por coaliciones de desarrolladores y usuarios. Y en la medida en que las tercerizaciones aumentan, la posición del arquitecto del software puede aún debilitarse -en especial en aquellas instancias en las que las especificaciones contractuales se centralizan en los requerimientos funcionales y no detallan las propiedades estructurales; y en el caso que existan mecanismos insuficientes para que los arquitectos verifiquen las propiedades estructurales del software que se brinda. (Por ejemplo, acoplamiento oculto que compromete la flexibilidad prevista de un artefacto de software.)

Por lo tanto, el desafío para los arquitectos del software es permanecer relacionados con el mundo SOA, un mundo de producción y servicios distribuidos, prestando atención a los problemas estructurales verdaderos que surgen en este mundo "en-demanda". De lo contrario, no podrán contribuir con nada de valor para el diseño y administración de los sistemas a pedido de los siste

mas. Esto conduce a la necesidad de formas de análisis que admitan un régimen de diseño asimétrico, y puedan permitir otorgar consideraciones explícitas para que ser realicen elecciones implícitas con respecto a la geometría.

Hemos visto que algunos proveedores reconocen el aspecto de que la oferta compatibilice servicios múltiples de la Red (IBM Rational, por ejemplo), mientras que otros proveedores de plataforma crean las condiciones para el vertiginoso aumento de los dominios (conductas) que necesitan ser puestos en relación entre sí (Microsoft, por ejemplo). En ambos casos, la empresa orientada al servicio se configuró como una estructura continua de servicios -"La Red corporativa." Pero esto nunca puede lograrse en un gran proyecto ambicioso. Debe lograrse progresivamente mediante un flujo continuo de proyectos pequeños y medianos.

Producción del servicio

En el enfoque de planificación en conjunto, el orden y la coherencia surgen de una actividad distribuida, que no posee una autoridad de diseño central. Cada unidad de compra, desarrollo o actividad de mantenimiento debe considerarse como un proyecto, donde los resultados de los proyectos se constituyen como servicios. Por lo tanto, cada proyecto aporta algo positivo a la red corporativa de servicios emergente. Entonces ¿qué forma de gobierno se necesita para mantener el orden de la arquitectura?

Se necesita del gobierno de SOA para asegurar que cada proyecto cumpla con las demandas globales de la Red corporativa, y para garantizar la existencia de un conjunto de proyectos bien balanceados -diferentes tipos así como también diferentes escalas (grande, mediana y pequeña).

El análisis en la Parte I ha resumido los límites de un enfoque oferta-demanda para el gobierno de SOA -la composición dirigida está limitada en su capacidad para responder a la heterogeneidad completa de la demanda. Esto deja demasiado déficit de valor en relación a la demanda que es cada vez más heterogénea, asimétrica y especial, así como también diferenciada con el tiempo. Por lo tanto, es necesario hacer llegar el gobierno a la periferia de la organización, reconociendo que trabajamos con procesos de diseño asimétricos, y prepararnos para enfrentar el siglo veintiuno según sus propias condiciones. En la Parte 2 de este artículo, se dejará abierta la pregunta respecto de qué significa hacer llegar el gobierno a la "periferia" tanto para el diseño de infraestructuras SOA como para la relación con la demanda.

Notas al pie

1.Para un consultar un resumen práctico, ver Coward & Salingaros.
2.Pat Helland, Metropolis. Microsoft Architects Journal [2], Abril de 2004. Disponible en http://msdn.microsoft.com/architecture/journ
3.Ejemplo práctico -la compañías telefónicas desearían considerar la ubicación de mástiles de telefonía móvil como una mera infraestructura, analizarlo desde el punto de vista de la tecnología y sin requerir la consulta pública. No obstante, las personas parecen interesarse en la radiación de estos mástiles, en especial los ubicados cerca de escuelas y hogares, y esto politiza la ubicación de los mástiles.
4.Albert Borgmann proporciona una posible reconciliación de la comoditización con los valores humanos.
5.Para consultar una version de este ciclo tal como se aplica en el cuidado de la salud, ver "Triply Articulated Modelling of the Anticipatory Enterprise" de P Boxer y Prof B. Cohen, Proceedings of the International Conference on Complex Systems, Boston 2004.
6.Ver el artículo de Murphy.
7.Imagen tomada del informe sobre Service-Based Business in Insurance, CBDI Forum Noviembre de 2004.
8.Este ejemplo se extrajo de un trabajo realizado por PASA. http://www.pasa.nhs.uk/orthotics/Orthotic_Pathfinder_Report_July_2004.doc
9.Las opiniones aquí se utilizan tal como las definió Vickers, para incluir un juicio de valor y reconocimiento así como también un juicio de actuación.

Referencias

Christopher Alexander, The Nature of Order.4 Volúmenes. The Center for Environmental Structure, 2002-2004.

Albert Borgmann, Technology and the Character of Contemporary Life. Chicago, 1984.

Philip Boxer & Bernie Cohen, Triple Articulation. BRL Working Paper, revisada en el 2004.

Andrew Coward & Nikos Salingaros, The Information Architecture of Cities, Journal of Information Science, Volumen 30 No. 2 (2004), pp. 107-118. Reimpresa en Salingaros 2005.

Jack Greenfield & Keith Short, Software Factories (Wiley, 2004). Pat Helland, Metropolis. Microsoft Architects Journal [2], Abril 2004. Disponible en http://msdn.microsoft.com/architecture/journ/

Jane Jacobs, The Death and Life of Great American Cities (Vintage Books, Nueva York, 1961).

Lewis Mumford, The Culture of Cities. (Secker & Warburg, 1938).

Richard C. Murphy, Centers: The Architecture of Services and the Phenomenon of Life. FTP Online Marzo 2004. http://www.ftponline.com/special/soa/murphy/

Nikos Salingaros, Principles of Urban Structure. Delft University Press, Delft, Holanda, 2005, en prensa. http://www.math.utsa.edu/~salingar/urbanstructure.html

Richard Veryard, Component-Based Business (Springer 2001).

Richard Veryard & David Sprott, The Service Based Business (CBDI Journal, 2003). SOA Governance and Business Driven SOA (CBDI Journal, 2004).

Gracias a Bernie Cohen, Pat Helland y Nikos Salingaros por sus comentarios a los primeros borradores.

Tecnologías de Integración Basadas en Servicios

Anna Liu y Ian Gorton

Introducción

Construir una solución de integración que se aplique a la empresa es una tarea difícil. Estas soluciones necesitan integrar sistemas empresariales múltiples que no están creados para trabajar juntos.

La integración de estos sistemas es difícil por varios motivos. Esto incluye la heterogeneidad de plataformas y lenguajes de programación, la diversidad y complejidad de cada sistema empresarial individual y la dificultad de comprensión de los requerimientos para la solución integrada resultante. Los arquitectos de software realizan un sin fin de tareas cruciales durante el diseño de aplicaciones empresariales integradas. Entre ellas se incluyen las siguientes:

- Colaborar en la comprensión de requerimientos funcionales y de calidad para las aplicaciones integradas.
- Crear el proyecto de arquitectura inicial para las aplicaciones integradas.
- Seleccionar tecnologías de integración adecuadas que cumplan con los requerimientos de la aplicación.
- Confirmar que la combinación de la arquitectura y la tecnología de integración que se utilizan para construir la aplicación para toda la empresa pueden ser exitosas antes de realizar una gran inversión para su implementación.

Este artículo describe un método comprobado que ayuda a los arquitectos a evaluar tecnologías de integración aplicadas a la empresa. En particular, centramos nuestro debate sobre la evaluación de tecnologías de integración, para implementar la integración basada en servicios.

SOA para la integración

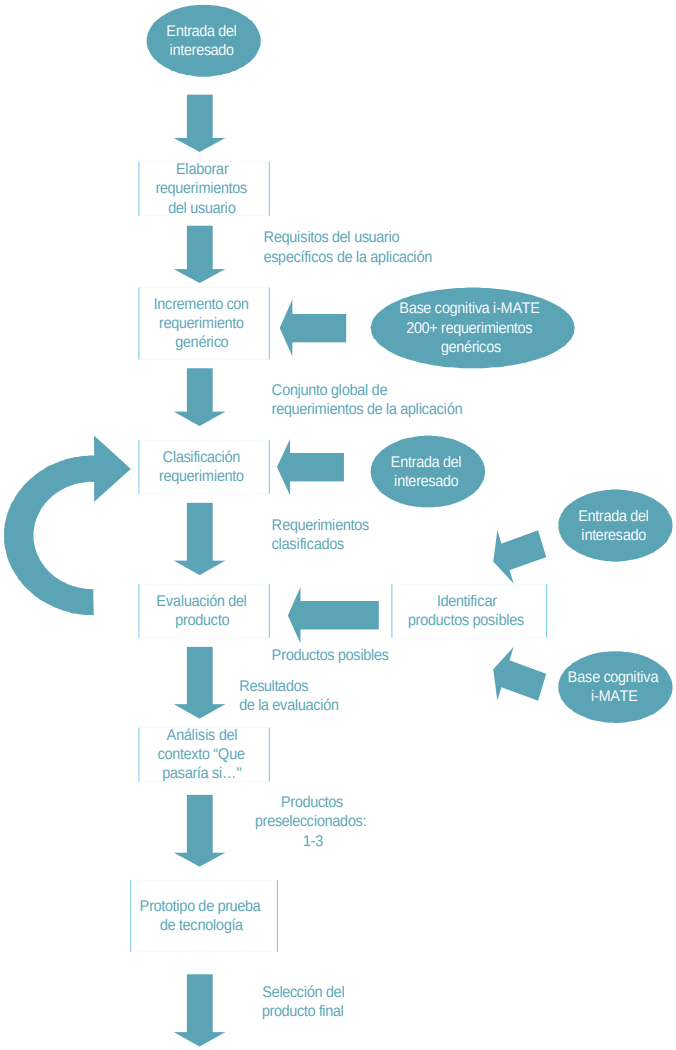
Con la llegada de estándares de la industria como por ejemplo Servicios Web, la Arquitectura Orientada a Servicios está impulsando un cambio de paradigma en algunas áreas, incluyendo la integración de aplicaciones empresariales.

La arquitectura de integración orientada al servicio consiste en integrar entidades informáticas usando interacciones de servicios. Este modelo basado en los servicios se refiere los problemas de integración de sistemas heterogéneos inflexibles y anticuados permitiendo que las organizaciones IT ofrezcan la funcionalidad bloqueadas de aplicaciones existentes como servicios reutilizables.

A diferencia de la tradicional Integración de Aplicaciones Empresariales (EAI), las características principales del enfoque basado en el servicio para la integración son:

- Interfaz estándar bien definida-Se proporciona a los consumidores acceso sistemático y de fácil comprensión para los servicios subyacentes
- Opacidad-La tecnología y ubicación de la aplicación que proporciona la funcionalidad están ocultas detrás de la interfaz. De hecho, no existe la necesidad de un proveedor de servicios fijos.
- Flexibilidad-Tanto el proveedor de servicios como el consumidor pueden cambiar-la descripción del servicio es la única constante. Siempre que el proveedor y el consumidor sigan fieles a la descripción del servicio, las aplicaciones seguirán funcionando.

Figura 1. Proceso de evaluación



Las tecnologías para la construcción de Integración basada en Servicios necesita contar con las siguientes funcionalidades básicas:

- Entrega del mensaje
- Enrutamiento inteligente
- Servicios para eventos
- Adaptadores para la Aplicación
- Transformación de datos/Traducción XML
- Procesamiento de Normas
- Soporte de Servicios Web
- Organización del Servicio/Proceso
- Gestión de procesos empresariales
- Seguimiento de la actividad empresarial

Además, para asegurar el éxito de la integración basada en servicios, es necesario que la tecnología de integración posea las siguientes cualidades:

- Escalabilidad
- Alto rendimiento
- Seguridad
- Manejabilidad

Como podemos observar, uno de los puntos principales se concentra en utilizar los estándares de la industria como los Servicios Web para permitir la entrega del mensaje real y otros diversos servicios avanzados, evitando los problemas de las tecnologías EAI tradicionales -es decir, el uso de protocolos de propiedad exclusiva para el intercambio de mensajes. De esta manera, la integración basada en servicios es un patrón de diseño que asegura la interoperabilidad e integración verdadera en cualquier contexto empresarial heterogéneo.

Existen diversas implementaciones variadas de tecnologías de integración que proporcionan las funcionalidades enumeradas más arriba. Estas abarcan desde las tecnologías EAI tradicionales que han incluido prestaciones de Servicios Web, hasta implementaciones nuevas de marcas que poseen soporte de Servicios de Web inherentes.

Desafortunadamente, seleccionar una implementación de tecnología de integración apropiada no es una propuesta simple para la mayoría de las organizaciones IT. Existen muchos motivos para esto, pero generalmente se centran en lo siguiente:

- Complejidad de la tecnología: Los productos de integración son muchos, diversos y realmente poseen un sin fin de características e interfaces de programación de la aplicación. Son complejos de comprender y el bajo nivel de detalles puede provocar serios efectos sobre la forma en la que el producto se comporta. El problema en realidad está en el detalle.

- Diferenciación del producto: Existen decenas de miles de productos que compiten en el entorno de la integración. En un nivel superficial, algunos poseen conjuntos de características y capacidades casi idénticas. La diferencia de precios puede por lo general ser importante, lo que complica aún más el problema de selección y adquisición.

- Conocimiento institucional de IT: Las organizaciones del usuario final rara vez cuentan con arquitectos o ingenieros que posean en el profundo y amplio conocimiento necesario de todos los productos y tecnologías de integración. Es por lo tanto una pérdida de tiempo y una práctica costosa para las organizaciones adquirir estos conocimientos y así poder seleccionar un producto de integración adecuado. Esto también distrae la atención del staff de ingeniería clave para quienes el objetivo principal son las tareas centradas en la aplicación.

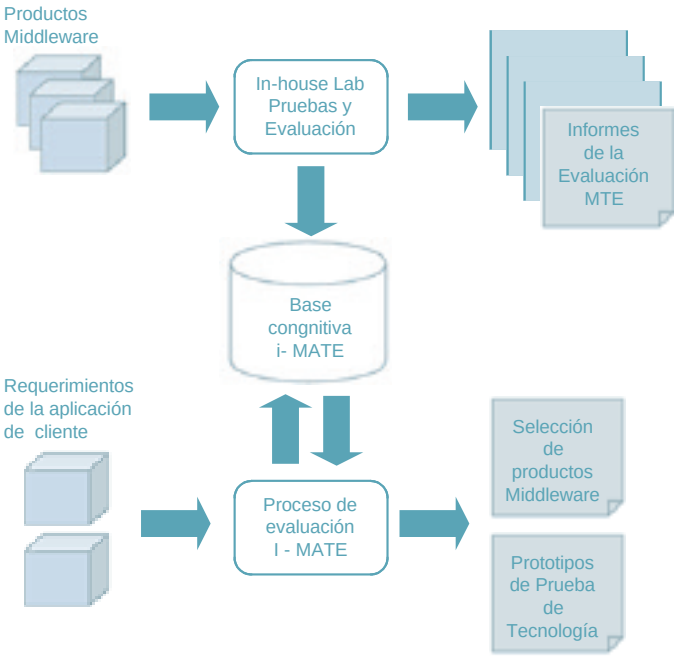
Evaluación

i-MATE (Middleware Architecture and Technology Evaluation in Internet time) es un proceso de ingeniería de software especializado que se usa para evaluar la reutilización de componentes externos [Off The Shelf (COTS)]. Es ideal para las organizaciones que operan en el Nivel 3 del Software Acquisition Capability Maturity Model del Software Engineering Institute [1], en especial de según el soporte para las áreas de proceso clave de User Requirements and Acquisition Risk Management. La eficacia y novedad de i-MATE se encuentra en la combinación de:

- Un proceso definido: Esto comprende una serie simple de pasos del proceso bien definido para recopilar, clasificar y ponderar los requerimientos de la aplicación para el middleware de integración.
- Una base cognitiva: Esto contiene cientos de requerimientos genéricos para las diversas clases de productos de middleware COTS, incluidos aquellos únicos para implementaciones de integración basada en el servicio.
- Una herramienta de análisis de requerimientos: La herramienta de análisis permite una rápida valoración, experimentación y presentación de la forma en la que los productos de middleware que se están evaluando se comparan con los requerimientos del proyecto.

Las próximas secciones describen las características exclusivas de i-MATE, que lo hacen muy adecuado para la evaluación de tecnologías de integración dentro del contexto de construcción de soluciones de integración basada en el servicio.

Figura 2. Formación de la base cognitiva



Requerimientos específicos SOI

- El producto debe admitir el conjunto de especificaciones de Web Service Basic Profile (por ejemplo, WSDI, SOAP, UDDI, XML)
- El producto debe admitir las especificaciones WS-* avanzadas (incluidas WS-Security, WS-Coordination, WS-ReliableMessaging, etc.)
- El proveedor debe implementar el conjunto de especificaciones WS-* por medio de descargas de cajas de herramientas dentro de los seis meses de la fecha de información de la especificación.
- El producto debe admitir el soporte del Servicio Web personalizado al exponerlo a SDK APIs o proporcionar mecanismos de intercepción
- El producto debe tener soporte del Servicio Web inherente, en vez de ser un producto añadido.
- Debe admitirse el acceso a las diversas características de Servicios Web (por ejemplo, Seguridad, garantías transaccionales) a través de APIs sistemáticos así como también medios declarativos
- El producto debe admitir una "implementación del Servicio Web" fácil de las interfaces de servicio empresarial existentes.
- El producto debe interoperar con todas las aplicaciones que exponen los servicios a través de las interfaces estándar de Servicios Web.
- El producto debe contar con extensiones de propiedad exclusiva que puedan dividir los requerimientos de interoperabilidad basados en los Servicios Web.
- La caja de herramientas de los Servicios Web debe haber experimentado la evaluación rigurosa del seminario de interoperabilidad de WS para demostrar su interoperabilidad con los otros Servicios Web.
- El proveedor debe ser miembro activo de la organización WS-I

El proceso que se utiliza en i-MATE es similar a aquellos descriptos en [2,3]. Define una serie de pasos que se realizan en i-MATE, y las decisiones tomadas y objetos producidos en cada etapa. Ellos se describen en la figura 1 y se explican brevemente a continuación:

Elaborar requerimientos del cliente. Este primer paso produce un documento que captura los requerimientos del cliente. Debido a que los problemas de aplicación y la tecnología son complejos, por lo general nos encontramos con que los requerimientos globales no se comprenden en su totalidad. Por consiguiente, se llevan a cabo varios seminarios con los interesados en la aplicación para obtener estos requerimientos. Los interesados involucrados en el mejor de los casos incluyen los grupos empresariales e IT. El documento resultante detalla los requerimientos técnicos y empresariales que son específicos para la necesidad de tecnología de integración en este entorno de integración basada en el servicio. Cada requisito se expresa como un único ítem que puede ser evaluado contra una tecnología de integración específica.

Aumentar los requerimientos genéricos. Esto presenta la base de conocimientos de i-MATE de casi 200 requerimientos aplicables amplios y genéricos para tecnologías de integración. Esto aumenta el conjunto de requerimientos específicos de aplicación con requerimientos de integración genéricos. El resultado de este representa los requerimientos de aplicación global para la tecnología de integración basada en el servicio, que se representan como puntos de requisitos identificados individualmente.

Clasificar los requerimientos globales. El conjunto general de requerimientos globales se clasifica trabajando en conjunto con los usuarios principales y otras partes interesadas de la aplicación. En un basto nivel, cada requisito es considerado como obligatorio, conveniente, de baja prioridad/no aplicable. Dentro de cada una de estas categorías, se asignan ponderaciones importantes para realizar un control de textura fina sobre la clasificación de requerimientos, de una forma similar a [4,5]. Resulta de este paso un conjunto de requisitos ponderados que se almacenan en la herramienta de análisis de requerimientos de i-MATE.

Identificar posibles productos. Este paso identifica de tres a cinco productos de integración que son los que muy probablemente sean aplicados a los requerimientos de aplicación global. En algunos casos, el cliente ya ha identificado una preselección basada en razones empresariales y técnicas. En otros casos, utilizamos nuestra experiencia para trabajar con el cliente en la identificación de las opciones más probables.

Evaluación del producto. Hemos evaluado cada uno de los productos posibles en comparación con los requisitos globales en seminarios realizados con los clientes interesados claves y los representantes de proveedores del producto. Se asignaron puntuaciones en comparación con cada punto de requisito para cada producto y se capturaron en la herramienta de análisis de requerimientos. Esto implica análisis técnicos en profundidad, así como también pasar por escenarios de aplicación relevante para comprender de manera precisa la forma en la que los productos de integración se comportan en la actualidad. En algunos casos, las capacidades y características del producto pueden causar que el proceso repita y depure las clasificaciones de requerimientos. Una vez que se han evaluado todos los productos, la herramienta de análisis de los requerimientos calcula automáticamente un resumen de las puntuaciones ponderadas en base a los puntos de los requerimientos individuales y las ponderaciones. Se crean también de manera automática cuadros sintéticos para permitir presentaciones e informes eficaces.

Análisis del contexto. Al variar las ponderaciones de requerimientos, la herramienta de análisis de requerimientos considera trivial explorar diversos contextos "Qué pasaría si..." ["what-if"] y relaciones de intercambio [trade-offs]. Esto puede utilizarse para diferenciar aún más entre productos posibles, o confirmar la idoneidad de ciertos productos sujetos a diversos requerimientos. El resultado de este paso es la recomendación de uno o más productos que pueden cumplir con los requerimientos de aplicación.

Creación de un prototipo

Prototipo de Prueba de Tecnología. Cuando el resultado de la evaluación del producto no está 100% completo, se desarrolla un Prototipo de Prueba de Tecnología. El prototipo por lo general implementa un contexto crítico que ejercitará y/o destacará los requerimientos que se considera que poseen la más alta prioridad. Aún los más simples prototipos son herramientas muy eficaces que proporcionan una evidencia indiscutible y concreta de las posibilidades del producto. En varios proyectos i-MATE, los resultados de los prototipos han proporcionado la diferenciación final requerida para ultimar la selección del producto. De hecho, siempre se recomienda una etapa de creación de prototipos en I-MATE, aún cuando surge del proceso un producto como líder claro. Sin embargo, cuando se considera sólo un producto, la tarea de creación de prototipos no es competitiva y puede ser definida y estructurada más hacia la validación de requerimientos claves de aplicación.

Con respecto a los recursos, las etapas de Evaluación del Producto y Prototipo de Prueba de Tecnología siempre consumen la mayor parte del esfuerzo en I-MATE. La evaluación del producto lleva, en promedio, entre uno y tres días de trabajo por producto que se evalúa, sujeto a la familiaridad del equipo de i-MATE con el producto en particular. La creación de prototipos es más variable y depende de la complejidad del prototipo deseado. En la mayoría de los casos, un sistema simple es suficiente y la etapa de creación del prototipo dura menos de una semana. En otras aplicaciones en las que el riesgo es mayor, la creación de prototipos se ha extendido hasta un mes.

La base cognitiva de i-MATE contiene un amplio conjunto de requerimientos genéricos para tecnologías middleware más genéricas, así como también aquellos específicos para tecnologías que admiten la orientación al servicio. Estos requerimientos genéricos derivan de la experiencia práctica del Proyecto de Evaluación de Tecnología Middleware de CSIRO [6], que trabaja con proveedores del producto y la participación de consultorías para los clientes, como se muestra en [7].

Debido a que existen diferentes clases de productos de middleware, hay una ejemplificación diferente de la base cognitiva global para cada clase de producto. Por ejemplo, la base cognitiva está versionada para tecnologías de integración basadas en servicios, Integración de Aplicaciones Empresariales (EAI), tecnologías Application Server y CORBA. Veremos la base cognitiva de la integración basada en el servicio en el próximo ejemplo.

Un análisis detallado del conjunto de requerimientos genéricos ha resultado en la estructuración de cada base cognitiva como un conjunto de categorías de alto nivel, que encapsulan diversos ítems de requerimientos individuales. La presentación de la base

cognitiva es su totalidad va más allá del alcance de este artículo, pero a modo de ejemplo, las categorías podrían aparecer de la siguiente manera para una versión de tecnología de integración basada en los servicios:

Categorías de evaluación de alto nivel. Cada categoría de alto nivel contiene por lo general entre 10 y 20 requerimientos individuales que lo relacionan a esta categoría. Por ejemplo, la categoría Soporte de Servicios Web contiene requerimientos de tecnología individual.

Soporte de Servicios Web. Estos puntos de requerimientos cubren características detalladas y de bajo nivel de las tecnologías de integración. Todas las soluciones de integración basadas en el servicio requieren inevitablemente algunas o todas estas capacidades. Durante un proyecto de i-MATE, el cliente es conducido a través de los contenidos de la base cognitiva y la importancia de cada requisito para determinar la aplicación de cliente. En algunos proyectos, el cliente conoce la tecnología y el proceso es claro y sin complicaciones, lo que lleva menos de un día. En otros proyectos, el cliente depende del equipo de i-MATE para comprender las implicaciones de algunos de los requerimientos y su importancia relativa se establece de manera conjunta.

Además de los requerimientos categorizados, la base cognitiva de i-MATE está integrada por evaluaciones de las diversas versiones de los productos más importantes de middleware. Cada producto en la base cognitiva se clasifica en base a una escala de 1-5 contra los requerimientos individuales. Se dan las clasificaciones y se mantienen vigentes por medio de dos mecanismos, tal como se explica más adelante y se representa en la figura 4.

El primero es el proyecto MTE, que evalúa de forma rigurosa las tecnologías de middleware utilizando un enfoque repetible y definido [6]. Los resultados de las evaluaciones de MTE se introducen directamente en la base cognitiva de i-MATE. El segundo mecanismo son los proyectos de i-MATE en sí mismos. Los clientes por lo general requieren que una tecnología de integración u otro producto middleware o versión que no ha sido evaluada con anterioridad sea examinada durante un proyecto de adquisición. En estas circunstancias, el equipo de i-MATE trabaja con el proveedor del producto para clasificar las características del mismo. La evaluación resultante extiende el alcance de los productos en la base cognitiva y esto puede utilizarse en proyectos posteriores.

Al reutilizar los requerimientos genéricos en i-MATE, las organizaciones ahorran el costo de desarrollo de su propio conjunto de requerimientos de tecnología de integración. Por lo tanto, los esfuerzos pueden concentrarse en capturar los requerimientos específicos de la aplicación y planeamiento y diseño para la arquitectura orientada al servicio en toda la empresa. Esto ahorra tiempo y esfuerzo y ayuda a proporcionar un resultado de bajo riesgo.

Se ha construido una herramienta personalizada de análisis de requerimientos para admitir el análisis de intercambio como parte del proceso de i-MATE.

La funcionalidad de la herramienta básica proporciona lo siguiente:

Tabla 1. Análisis de contextos

Determinación de costos:	Costos de capacitación/servicios/tecnología básica
Mensaje XML y Gestión del servicio:	Disponibilidad de servicios para la definición del formato del mensaje, interfaz del servicio/definiciones del contrato, gestión del servicio
Arquitectura de integración:	Características de arquitectura central, flexibilidad, servicios de evento, forma en que se descubren/integran/invocan los servicios
Adaptadores:	Variedad y calidad de los adaptadores disponibles para la integración de sistemas externos
Transformación de datos y traducción XML:	Posibilidades de las herramientas de traducción XML internas. Características de rendimiento del procesamiento, posibilidades de clasificación de los datos
Calidad de entrega:	Fundamento del bus de servicio, modo de operación, calidad del servicio (ej. transmisión del mensaje confiable), posibilidades de ruteo
Soporte del Servicio Web:	¿Se cumple conforme a las normas de Servicios Web? ¿Qué tan completo es el soporte del Servicio Web? ¿Es el soporte del Servicio Web inherente o añadido?
Desarrollo y admisión:	¿Cómo se desarrollan, depuran las aplicaciones?
Rendimiento:	Rendimiento original y temas de escalabilidad
Seguridad:	Autenticación, autorización, encriptación, servicio de registro único
Servicios de transacción:	Servicios disponibles para la admisión del comportamiento transaccional
Flujo de trabajo:	Características de la automatización y gestión del proceso empresarial, Seguimiento de la Actividad Empresarial
Gestión del sistema:	¿Cómo se versionan, gestionan y utilizan las aplicaciones?
Técnico:	Requerimientos técnicos diversos

- Captura puntos de requerimientos individuales, tanto genéricos como específicos para la aplicación, estructurados en categorías de alto nivel.
- Captura de las clasificaciones del producto y de las ponderaciones de los requerimientos.
- Cálculo instantáneo de los porcentajes ponderados para las categorías de requerimientos.
- Cálculo instantáneo e informe de los resultados de la evaluación utilizando gráficos y cuadros.

En la figura 5 se muestra una imagen de captura del análisis de intercambio. Está basada en un programa de hoja de cálculo. La fuerza principal de este enfoque se demuestra durante las etapas de Evaluación del Producto y el Análisis del escenario What-if de i-MATE. Debido a que la hoja de cálculo es en "en vivo", cualquier cambio que se realice a las clasificaciones de categorías o ponderaciones de ítems de requerimientos se refleja inmediatamente en los gráficos representando la puntuación de la evaluación.

Por ejemplo, en la figura 6 se muestra la pantalla para establecer las ponderaciones de categorías de requerimientos. En este proyecto, las categorías Rules Engine, Development and Support, y System Management se consideran de alta prioridad. Estas configuraciones generan un conjunto de gráficos que representan las clasificaciones del producto una vez que se ha completado su evaluación. En este momento, por lo general es conveniente explorar la forma en la que podría variar el resultado de la evaluación global si una de estas categorías que posee una prioridad alta ser reduce a un nivel de prioridad medio. Al cambiar cualquier valor de prioridad, la hoja de cálculo refleja de manera instantánea en los resultados de la evaluación los cambios a las prioridades. Esto posibilita una rápida exploración de las alternativas y confirmación de los resultados de la evaluación bajo escenarios alternativos diversos.

El proceso i-MATE ha sido descripto como un proceso para facilitar la evaluación de tecnologías de integración, dentro del contexto de la implementación de una arquitectura orientada al servicio.

Las tecnologías de integración son grupos de productos diversos, altamente técnicos y complejos que por lo general operan en entornos de suma importancia para el funcionamiento de la empresa. También es una inversión importante para IT ya que asegura una integración futura sin problemas.

Figura 5. Herramienta del Análisis de Intercambio

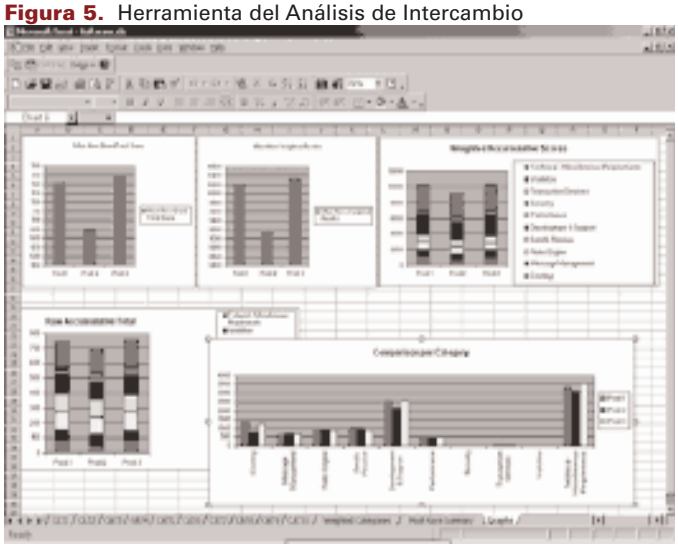
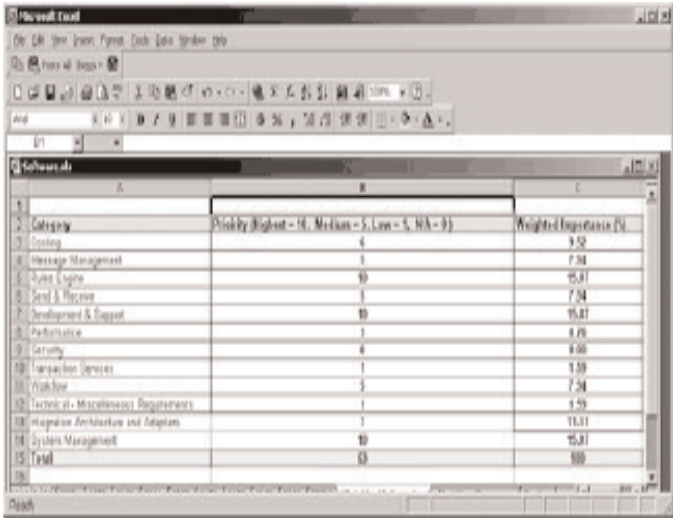


Figura 6. Categoría de ponderaciones



La contribución clave de i-MATE al facilitar el proceso de evaluación de la tecnología de evaluación reside en la combinación de lo siguiente:

- Un conjunto de requerimientos genéricos prefabricados y reutilizables basados en el análisis de las características de componentes de middleware.
- Un proceso para la incorporación de requerimientos específicos de aplicación, ponderando los requerimientos individuales.
- Soporte de la herramienta para capturar y explorar de manera rápida requerimientos intercambio y generar informes que muestren la forma en que los productos middleware se comparan con los requerimientos.

El enfoque de integración basado en el servicio posee la clave para una interoperabilidad e integración futura sin problemas. Si las cosas se hacen correctamente, no deberíamos tener que enfrentar más los problemas tradicionales de la integración de aplicación empresarial. Con la llegada de los servicios Web y con la participación y colaboración de la industria por primera vez en los esfuerzos de estandarización de nuestra industria IT, los servicios Web y la arquitectura orientada al servicio prometen resolver los problemas del desafío de la integración de la aplicación empresarial. La integración basada en los servicios es un patrón importante para la implementación de esta visión. La selección cuidadosa de una tecnología de integración para este propósito es absolutamente importante para contribuir en el éxito de esta iniciativa de ingeniería del software.

Referencias

1.J. Cooper and M. Fisher, Software Acquisition Capability Maturity Model (SA-CMM), Versión 1.03, CMU/SEI-2002-TR-010, Marzo de 2002, en http://www.sei.cmu.edu/publications/documents/02.reports/02tr010.html

2.S. Comella-Dorda, J. C. Dean, E. Morris, P. Oberndorf, A Process for COTS Software Product Evaluation, en Actas del 1st International Conference on COTS-Based Systems-ICCBSS 2002 Orlando, Florida, pp.86-96, Febrero 4-6, 2002.

3.J. Kontio, A Case Study in Applying a Systematic Method for COTS Selection, en Actas del 18th International Conference on Software Engineering, pp 201-209, IEEE, Berlin, Marzo de 1996.

4.J. Kontio, A Case Study in Applying a Systematic Method for COTS Selection, en Actas del 18th International Conference on Software Engineering, pp 201-209, IEEE, Berlin, Marzo de 1996.

5.Patricia K. Lawlis et al, A Formal Process for Evaluating COTS Software Products, Computer, Vol. 34, No. 5, Mayo de 2001.

6.I.Gorton, A.Liu, Software Component Quality Assessment in Practice: Successes and Practical Impediments, in Proceedings of the International Conference on Software Engineering, Orlando, May 2002, IEEE, páginas 555-559.

7.Ian Gorton, Anna Liu, Streamlining the Acquisition Process for Large-Scale COTS Middleware Components, en Actas del 1st International Conference on COTS-based Software Systems, Florida, Volumen 2255, pp 122-131, Lecture Notes in Computer Science, Springer-Verlag, Feb 2002.

Sobre los autores

Anna Liu
Microsoft Australia
Anna Liu es Arquitecta para el Developer and Platform Evangelism Group en Microsoft Australia. Se especializa en proyectos de integración de aplicación empresarial y es una apasionada respecto de la codificación de mejores practicas para la ingeniería del software, así como también acelera la adopción de estas mejores prácticas y aprendizajes. Anteriormente, Anna fue Ingeniera de Investigación en CSIRO donde asumió un compromise fulltime y ocupó el cargo de científica invitada en el Software Engineering Institute, CMU. Posee un Doctorado en Ingeniería Informática.

Ian Gorton
National ICT Australia
Ian Gorton es Investigador Senior en National ICT, Australia. Hasta marzo del 2004 se desempeñó como Chief Architect in Information Sciences and Engineering en el Laboratorio Nacional Pacific Northwest del Departamento de Energía de EE.UU. Anteriormente trabajó para Microsoft e IBM, así como también en otros cargos de investigación, incluyendo CSIRO. Sus intereses incluyen las arquitecturas de software, en particular aquellas para sistemas de información de alto rendimiento a gran escala que utilizan tecnologías middleware para "productos comerciales fuera de góndola" (COTS). Realizó un doctorado en Ciencias de la Computación en la Universidad Sheffield Hallam.

Aviones, Trenes y Automóviles

Simon Guest

Introducción

Muchas implementaciones de Servicios Web existen en la actualidad a lo largo de entornos y plataformas múltiples. La mayoría de ellos comparten una cosa en común todos utilizan HTTP como el transporte básico. La naturaleza ubicua de HTTP ha ayudado a que los Servicios Web logren su actual nivel de adopción.
A pesar de esto, ¿Es HTTP un gran ajuste para cada problema? ¿Existen arquitecturas de aplicación que podrían beneficiarse con el uso de otros transportes? ¿Cuáles son las ventajas y desventajas de hacerlo?
Este artículo presenta la respuesta a estos interrogantes y aún más, con la mirada hacia contextos en los que los transportes alternativos para los Servicios Web pueden ofrecer una mayor solución sobre HTTP.

Como muchas personas saben, HTTP ha tenido una larga historia antes de la aparición de los Servicios Web. Ha sido el transporte predeterminado para explorar páginas de Web desde que las primeras versiones sobre NCSA Mosaic se dieron a conocer al mundo.

HTTP tal y como está funciona bastante bien para los Servicios Web. Siendo global, por lo general funciona bien entre Firewalls y Servidores Proxy, elementos (como WSDL) son fácil de evaluar si se utiliza HTTP, y algunos servidores y pilas de HTTP están disponibles para la construcción de implementaciones. También, si observamos algunos de los "objetivos de diseño" iniciales para los Servicios Web, veremos mucha inercia respecto a los Servicios Web que se presentan públicamente. Como resultado, HTTP contribuye a la elección perfecta.

A pesar de la generalidad de HTTP, no obstante, existen contextos en los que no siempre se ajusta. En mi experiencia, he visto esto clasificado en tres categorías:

- Conexiones asíncronas
- Aplicaciones locales y offline
- Informática punto a punto

Veamos cada una de estas áreas en forma individual y examinemos contextos en los que HTTP puede no ser la combinación perfecta.

La consultoría Contoso, una organización ficticia, emplea casi 5000 consultores en todo lo el mundo. En los últimos años han experimentado un crecimiento fenomenal, contratando varios miembros a su staff hacia el final de la época de las empresas puntocom. Debido a este aumento en el personal de la empresa, uno de los problemas que enfrentan hoy en día es la notificación de las horas trabajadas. Todas las semanas, cada consultor debe enviar una planilla con las horas trabajadas para que la facturación del cliente sea precisa y oportuna.

Su modelo actual comprende una planilla de horas trabajadas (creada en una plantilla de Microsoft Excel) que se envía por e-mail al departamento de contabilidad. Una vez que se recibe esta planilla, un empleado del grupo de contaduría la ingresa en el sistema de contabilidad para registrar las horas que se deben facturar. Este sistema de contabilidad está en la actualidad basado en un sistema central.

Como podemos imaginar, este modelo para el envío de las horas trabajadas no se adapta bien con su expansión actual. A pesar de la contratación de más personal en el departamento de contabilidad, el método para administrar las planillas entrantes con el uso de e-mal se torna difícil debido al proceso manual que esto implica al tomar los datos desde Excel e ingresarlos al sistema de contabilidad.

Para colaborar con esto, el IT interno ha diseñado un nuevo servicio de envío de las planillas con las horas trabajadas. Si se utilizan los Servicios Web, este servicio se situará frente al sistema central, acepta la planilla del consultor y automáticamente ingresa los detalles en el sistema de contabilidad. El diseño del Servicio Web se ha mantenido simple y se ha desarrollado una aplicación de Cliente Inteligente para el envió de estos datos, tal como se muestra en la Figura 1.

Uno de los objetivos de diseño iniciales, sin embargo, ha sido asegurar que el envío de las planillas con las horas trabajadas se realice de manera asíncrona. El sistema de contabilidad se está tornando anticuado, y pensar en la idea de que 5000 consultores presenten sus planillas todos el último día de la semana es de alguna manera abrumador. Para superar esto, el arquitecto para el sistema se ha decidido implementar una cola de mensajes entre el Servicio Web y el sistema de contabilidad, como se muestra en la Figura 2.

La responsabilidad de la cola será agrupar en lotes las solicitudes de los clientes antes de enviarlas al sistema de contabilidad. Este diseño asíncrono evitará que el sistema de contabilidad se sobrecargue con tantas solicitudes en curso, y al mismo tiempo "libera" la aplicación de Cliente Inteligente para que realice más tareas. (es decir, el Cliente Inteligente no tiene que esperar hasta que el sistema de contabilidad haya procesado la planilla antes de pasar a otras tareas)

Todo esto parece bueno en teoría, pero el grupo de IT percibe un problema potencial. ¿Qué sucede si hay un problema con el envío de la planilla?

Figura 1. Se utiliza un Web Services Façade para exponer el Servicio Web

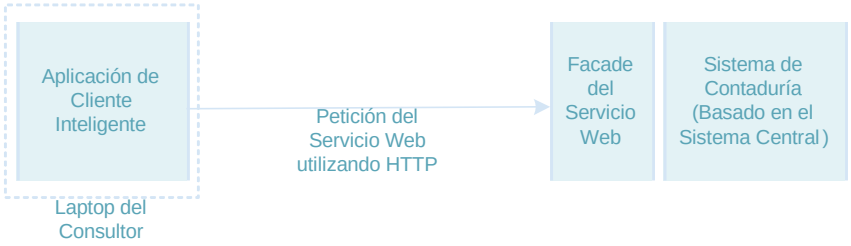


Figura 2. Se utiliza una cola de mensajes para agrupar las peticiones desde los clientes hacia el sistema central

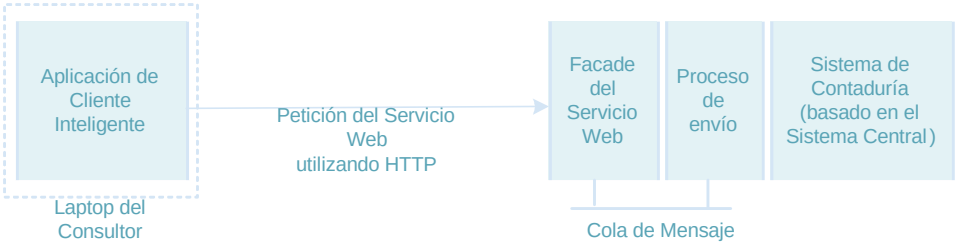
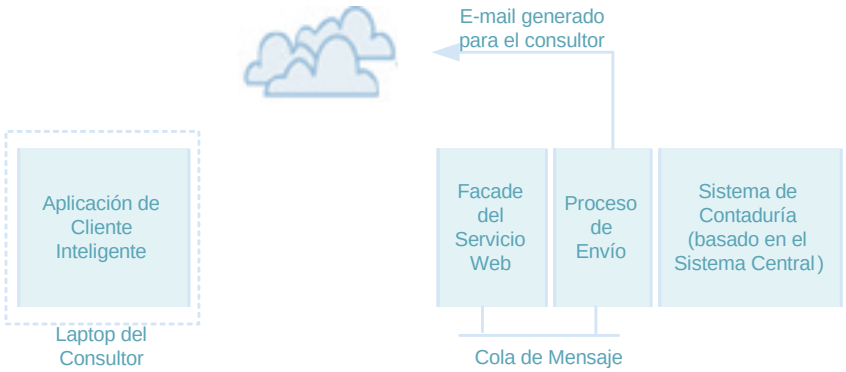


Figura 3. La respuesta asíncrona al cliente es difícil



Imaginemos la siguiente secuencia: Un consultor envía la planilla utilizando el Servicio Web de la figura 2. Todo funciona bien y la solicitud con la planilla se coloca en la cola de mensajes. Después de una horas (es viernes a la noche y el sistema está ocupado), el sistema de contaduría lee la planilla de la cola de mensajes. En este proceso se descubre que el consultor ha asignado algunas horas de manera incorrecta a un proyecto que previamente se había marcado como finalizado. El sistema de contaduría necesita información antes de poder continuar con el procesamiento de la petición.

Si se utiliza el diseño actual ¿Cuáles son las opciones?. El sistema de contaduría podría enviar un alerta a un miembro del staff de contaduría (tal vez un mensaje del sistema) para indicar que se requiere más información. El miembro del staff de contaduría puede entonces contactar al consultor para obtener los datos correctos. Esto cierra el ciclo, pero es aún un trabajo manual -y se expandirá sólo en la medida que lo haga la cantidad de personal en la organización.

Otra opción sería que el sistema de contabilidad envíe un alerta directamente al consultor tal vez enviando un e-mail al consultor en el que solicite más información. Una vez más, esto cierra el ciclo, pero la solicitud de información está aún desconectada del proceso original. ¿Cómo relaciona el consultor el e-mail con la planilla enviada? ¿De qué manera adecuada puede el e-mail describir la información que se necesita? ¿De qué manera puede el sistema de contaduría relacionar el envío de una planilla nueva o modificada con la planilla anterior y con el e-mail que se envió? ¿Qué seguimiento puede realizar el sistema de contaduría si el consultor simplemente borra el e-mail? ¿Qué sucede con el envío existente? Existe la posibilidad de que este ciclo nunca se cierre.

Vayamos un paso atrás y preguntemos en primer lugar ¿Por qué es necesario un e-mail? ¿Por qué el sistema de contaduría no comunica directamente la información a la aplicación de Cliente Inteligente? La respuesta: HTTP es un protocolo de petición/respuesta.

Una vez que se ha enviado la planilla y la petición/respuesta de HTTP se ha completado, como se muestra en la figura 3, es casi imposible volver a comunicarse con el cliente para solicitarlemás información. Aún si el cliente está ejecutando un Servicio Web local (para aceptar solicitudes de Servicios Web entrantes), ¿Qué sucede si se desconectan o el sitio del cliente está detrás de un firewall?

¿Qué pasa si su Dirección IP y/o hostname ha cambiado desde la última comunicación? Aún mas aterradora es la pregunta de ¿Quién administra las implementaciones del servidor Web sobre cada una de las 5000 laptops de los consultores.

Puesto que HTTP es un protocolo de petición/respuesta, y como tal es muy difícil para el servicio hacer un seguimiento con el cliente, deben tomarse medidas asíncronas alternativas (es decir, el e-mail). Lamentablemente, debido a que este e-mail está realmente "desconectado" de la solicitud original, puede por lo general implicar más trabajo relacionar lo que es necesario que suceda.

Para ver la forma en la que podemos diseñar una solución utilizando otros transportes que no sea HTTP, veamos algunos modelos alternativos.

Servicios Web que utilizan Colas de Mensajes. Fue fácil llegar a la conclusión de que HTTP fue el responsable. Nuestro primer modelo para la solución requiere pensar en un medio de transferencia alternativo para reemplazar o adaptar en nuestro diseño las conexiones HTTP.

Tomemos nuestro diseño existente y reemplacemos la transmisión HTTP por una cola de mensajes. La implementación no es importante en este momento (podría ser MSMQ, IBM MQ Series,

Figura 4. Se envía una planilla directamente a la cola de mensajes

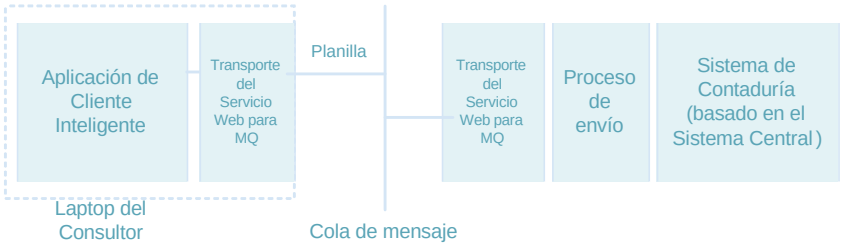


Figura 5. El sistema de contaduría solicita más información utilizando la cola de mensajes

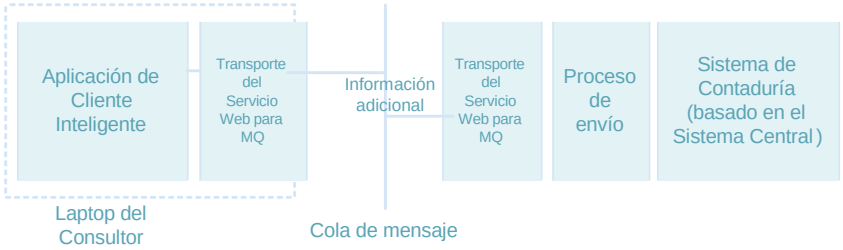


Figura 6. Uso de SMTP para las Peticiones de los Servicios Web

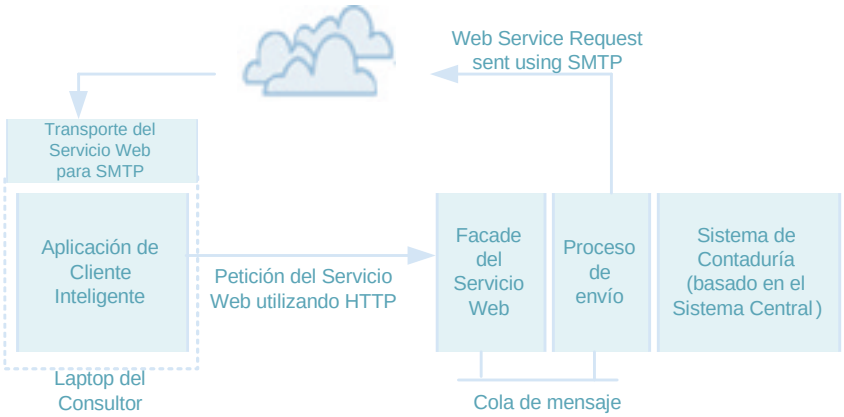
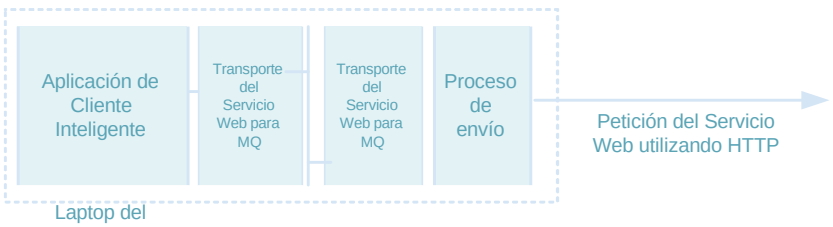


Figura 7. Se utiliza una cola de mensaje local para continuar con la solicitud sin conexión



Tuxedo, etc.), siempre y cuando sea confiable para manejar solicitudes asíncronas.

Es importante observar que todavía utilizaremos Servicios Web enviamos y recibimos muchos mensajes SOAP, con la excepción de que se enviarán utilizando una cola de mensajes asíncrona a diferencia de HTTP. Tal como se ha dado el ejemplo, podemos utilizar transportes permitidos por el Servicio Web para la cola de mensajes. Por lo tanto, ¿Cómo funciona esta arquitectura con nuestro nuevo contexto?

El Cliente Inteligente envía una planilla (y se crea una petición del Servicio Web correspondiente). Esta petición se ubica directamente en la cola de mensajes, en lugar de ser enviada en HTTP. Una vez que el mensaje se ubica en la cola, el cliente puede desconectarse sin peligro alguno.

El servidor basado en el Servicio Web a su vez tomará conocimiento de este mensaje, y luego se pondrá en contacto con el sistema de contaduría para que procese la petición. Si el sistema de contaduría necesita pedir información adicional, se coloca un nuevo mensaje en la cola (una petición al consultor) el destinatario es la aplicación de Cliente Inteligente.

Este mensaje permanecerá en la cola hasta que el Cliente Inteligente se vuelva a conectar. Para asegurar que el mensaje se identifique sin demoras , tal vez consideremos un servicio en segundo plano en base al cliente que se conecta a la cola de mensajes y envía el cuadro de diálogo "falta información" en la aplicación de la planilla para el consultor.

Esto proporciona una solución para nuestro para nuestro problema de ciclos cerrados, y puede ofrecer un modelo más automatizado pero esto tiene un defecto. El Cliente Inteligente debe poder conectarse a la cola de mensajes para procesar las peticiones que ingresan desde los sistemas de contaduría. La mayoría de los proveedores de colas de mensajes realizan esto por medio del acceso a alguna cola de mensajes APIs de propiedad exclusiva. ¿Qué sucede si el consultor está en una carretera? ¿Qué sucede si el consultor sólo tiene acceso desde un aeropuerto a través de la Web o un e-mail? Estos mensajes no serán entregados hasta que se conecte a la red corporativa, lo que puede ser inaceptable.

Utilizando otro transporte, veamos el segundo modelo:

Servicios Web que utilizan Transportes HTTP y SMTP. Uno de los problemas más importantes con el diseño original es que una vez que el sistema de contaduría envió el e-mail al consultor solicitando más información, realmente se crea un ciclo abierto. Este diseño depende de que el consultor tenga que asociar manualmente el mensaje de e-mail con texto entrante con el proceso de la aplicación.

El transporte en sí mismo, por el contrario, es eficaz de un modo razonable. Con la confiabilidad que representan los e-mail hoy en día, es mucho más que probable que el consultor haya recibido el e-mail.

Teniendo esto en cuenta, podríamos considerar un nuevo diseño a partir de lo siguiente:

Aquí la solicitud del Servicio Web sobre HTTP aún se utiliza para enviar la planilla. De igual manera que con el diseño original, esto está comprometido a una cola de mensajes para liberar la conexión del Cliente Inteligente. En este diseño, si ocurre algún problema con la planilla, se envía un e-mail -pero no al consultor. Por el contrario, se genera un e-mail que contiene la solicitud del Servicio Web para que se cree la aplicación de Cliente Inteligente. Lo que hacemos es iniciar una solicitud de información adicional del Servicio Web, pero utilizando SMTP como transporte.

El Cliente Inteligente necesita algunas modificaciones para hacer que esto funcione. Es necesario que exista algún modo de recuperar la solicitud de SOAP utilizando un e-mail -esto puede ser o un filtro en la bandeja de entrada del consultor o una cuenta de e-mail por separado para la aplicación de Cliente Inteligente. En segundo lugar, una vez que se ha recibido el e-mail, la aplicación de Cliente Inteligente debe procesarlo y provocar que suceda la acción correcta sobre el cliente (por ejemplo, un diálogo para preguntar al consultor por la información faltante). Una ventaja de estos modelos es que utilizan los transportes existentes, permite que la aplicación de contaduría inicie una solicitud a la aplicación de Cliente Inteligente y (sujeto a que podamos acceder al e-mail desde una ubicación remota) no restringe al consultor a tener que conectarse a la red corporativa para enviar los gastos. Hay que recordar también que debido a que la solicitud es un Servicio Web, se pueden aplicar otras normas de igual manera (Como WS-Security) -que proporcionan integridad y confidencialidad para el mensaje a pesar de que éste se envíe a través de servidores SMTP públicos.

Figura 8. Dos aplicaciones en comunicación en la misma máquina utilizando http

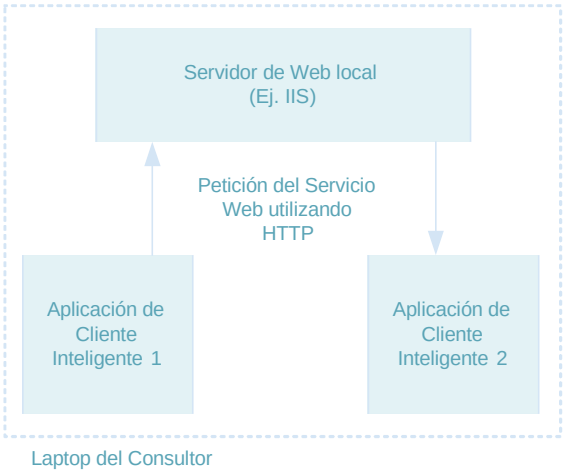


Figura 9. TCP o In Process proporciona una forma más directa para la comunicación entre aplicaciones locales

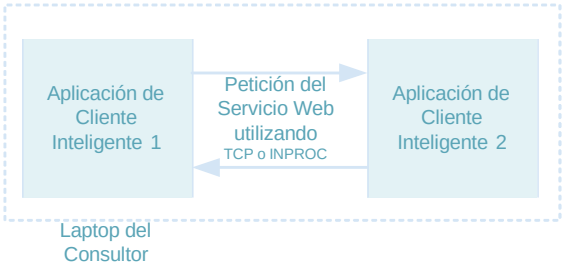


Figura 10. El transporte para el Servicio Web se utiliza para registrar el mensaje en SQL

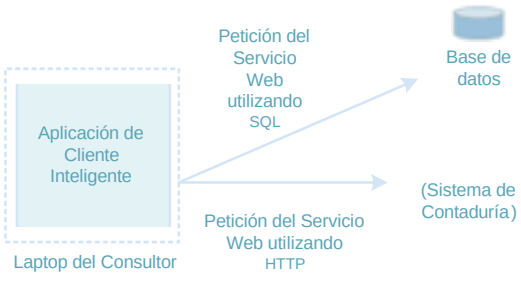


Figura 11. El Servicio Web de Bob procesa peticiones y respuestas en SMTP

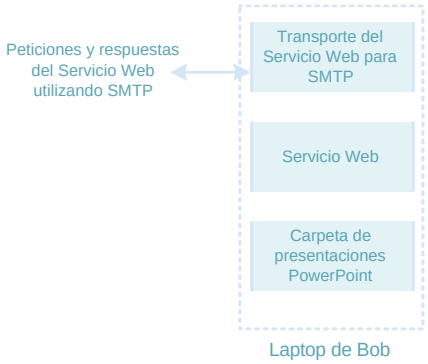


Figura 12. La Laptop de Joe envía 50 solicitudes del Servicio Web utilizando SMTP

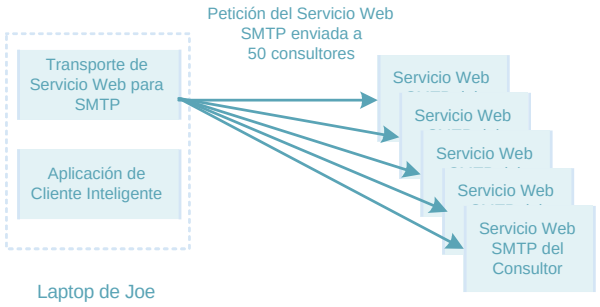
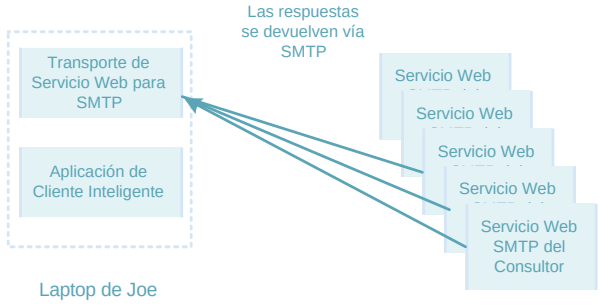


Figura 13. La aplicación de Cliente Inteligente procesa las respuestas que regresan



Este concepto de conexiones asíncronas puede del mismo modo aplicarse sobre el cliente. Si tomamos nuestro ejemplo previo, imaginemos que el consultor está por enviar una planilla. Se genera la planilla dentro de la aplicación de Cliente Inteligente y luego se envía (utilizando HTTP) al Servicio Web.

Esto funciona perfectamente bien sujeto a que exista una conexión para el Servicio del Web. ¿Qué sucede si el consultor envía la planilla desde un lugar donde no existe conectividad alguna (entre los sitios de los clientes, cuando está a bordo de un avión a 30000 pies, por ejemplo). En este caso deberíamos pensar en algún tipo de modelo sin conexión. Una vez que sea a clikeado el botón "Enviar", el diseño del Cliente Inteligente deberá detectar que no existía conexión en red y la operación sería suspendida o guardada en alguna cola o base de datos local.

Otra alternativa es considerar un segundo transporte. En vez de utilizar directamente HTTP desde el Cliente Inteligente, podríamos considerar un transporte de cola local para proporcionar esta funcionalidad sin conexión de manera automática.

Aquí, se instala una cola local (utilizando MSMQ, por ejemplo) en la laptop del consultor. En vez de utilizar HTTP, la solicitud de SOAP se ubica en la cola por como opción predeterminada. Un segundo proceso, que se ejecuta potencialmente en un segundo plano de la máquina del consultor, podría controlar la instancia MSMQ local para los mensajes nuevos y de modo frecuente, controlaría si es posible estableces una conexión para el Servicio Web basado en HTTP. Una vez que se han logrado estas dos conexiones, el mensaje se reenvía entre los transportes.

Para las aplicaciones de Cliente Inteligente, el uso de transportes para Servicios Web alternativos también ofrece otras opciones: Imaginemos que se tienen dos aplicaciones de Cliente Inteligente ejecutándose en la misma máquina que necesita la conexión. Tratar de iniciar llamadas a través de los Servicios Web sobre HTTP podría ser excesivo, ya que requeriría una instancia local de servidor de Web, y cada solicitud probablemente podría atravesar la pila de la red en la máquina.

Una forma más eficaz de realizar esto sería utilizar un transporte basado en TCP (socket) o un inprocess (o transporte de memoria compartida). Aquí, tal como se muestra en la Figura 9, las dos aplicaciones en la misma máquina se pueden comunicar utilizando peticiones y respuestas de Servicios Web estándar, pero por medio de un transporte ligero y manejable.

Además, al utilizar nuestro ejemplo de la planilla, ¿Qué sucedería si deseáramos implementar una forma de conectar todas las solicitudes del Servicio Web (por razones de auditoría). Con probabilidad logremos esto al crear un registro del mensaje antes de que éste emigre al servicio, lo que requeriría un filtro o clase para llevar el mensaje a la base de datos.

Tabla 1. ¿ Cuando implementar transportes en HTTP?

Problemas al utilizar HTTP	Transporte alternativo	Ventaja	Desventaja
Dificultad para comunicarse de nuevo con el cliente una vez que se ha realizado la petición	Cola de mensaje (p.e., MSMQ o IBM MQ Series)	Conexión asíncrona verdadera	Dificultad para acceder a una ubicación remota (requiere la visibilidad de la cola)
Dificultad para comunicarse de nuevo con el cliente una vez que se ha realizado la petición	SMTP	Conexión asíncrona verdadera	Requiere un filtro o buzón de correo para procesar los mensajes entrantes
Necesita gestionar de manera correcta el estado de conexión en el caso de que el servicio HTTP no esté disponible	Cola de mensaje local	Admite mensajes aún sin conexión	Requiere la instalación local de la cola de mensajes, además del proceso de control
Acceder a las solicitudes de los Servicios Web requiere códigos adicionales	SQL	Admite mensajes directamente para el servidor SQL como un transporte alternativo	Necesita un código en el servidor SQL para clasificar las solicitudes del Servicio Web en el esquema de la base de datos
Se requiere un Servidor de Web para dos aplicaciones en la misma máquina	TCP o In Process	Comunicación directa sin un servidor adicional	Requiere la administración de conexiones abiertas en el equipo local (por ejemplo pooling, firewall local, etc.)
Dificultad al exponer un Servicio Web punto a punto entre las organizaciones (a m se abran agujeros de seguridad en el Firewall	SMTP	Se requiere muy poca infraestructura adicional (si se asume que el servidor de Email ya existe)	Seguridad difícil de controlar. Sólo bueno para async, contextos de ejecución potencialmente largos

Esto funciona, pero un enfoque más fácil (tal como se muestra en la Figura 10) podría ser implementar un transporte para el Servicio Web. Un transporte podría utilizar una base de datos SQL para registrar las solicitudes de salida, aunque para la aplicación de Cliente Inteligente esto parece simplemente otro transporte.

Aquí, la solicitud de Servicio de Web se envía utilizando dos transportes. El primero va al destinatario previsto (por medio de HTTP). El segundo se envía a la base de datos para registro por medio de un transporte SQL.

Informática punto a punto

Finalmente, otra área que posee un gran potencial en los transportes para Servicios Web alternativos es la informática punto a punto. Veamos un ejemplo:

Bob es consultor para Contoso. En su laptop tiene un directorio de diapositivas de PowerPoint que utiliza para las presentaciones que hace a sus clientes. Este directorio viaja con él a cualquier lado que vaya. Está en constante uso y debe funcionar en contextos online y offline.

Debido a que es un buen ciudadano, Bob desea compartir estas presentaciones en Power Point con sus compañeros de trabajo - ya sea dentro de la empresa así como también con miembros de otras organizaciones. Muchas personas en la actualidad le envían e-mails para preguntarle si tiene alguna presentación de Power Point en particular sobre algún tema -y mientras esto sucede, la búsqueda de la información y la respuesta de estos e-mails le consume a Bob mucho tiempo.

Bob está pensando en construir un Servicio Web centralizado para alojar sus presentaciones de Power Point. Debería estar disponible para cualquier persona, incluso él mismo debe poder acceder a estos archivos en contextos offline. Considera entonces los pasos requeridos para implementar este servicio.

Configuración de un Servidor Central. Bob tomará su directorio con presentaciones de Power Point y lo alojará centralmente en algún lugar. Esto implicará no sólo encontrar el suficiente espacio de disco, sino también una consideración para la administración de copias de seguridad y actualizaciones con las últimas versiones.

Presentar un Servicio Web. Con la configuración del servidor, Bob deberá tener que instalar un servidor Web en la máquina, crear un Servicio Web y trabajar con el grupo IT local para asegurarse de que esté alojado de manera correcta detrás de los Firewalls de Contoso (probablemente en el DMZ).

Crear una Aplicación de Cliente Inteligente para el Acceso. Bob está pensando en crear una aplicación de Cliente Inteligente que le permita a él guardar en un instante una versión offline de las presentaciones que necesita.

Bob considera lo siguiente esto parece implicar muchísimo trabajo, además de que no está seguro de lo bien que se adaptará esto. ¿Qué sucede si los otros 5000 consultores de la organización también desean hacer algo similar? ¿Deberán experimentar el mismo método? ¿Qué sucede si no poseen los mismos conocimientos de tecnología que Bob?

Bob retrocede un paso y considera el motivo por el cual desea realizar esto -el sistema actual funciona bastante bien, es simplemente porque se siente inundado con tanto e-mail que solicitan las presentaciones que él ha dado o dará.

Él podría crear potencialmente un Servicio Web en su laptop para manejar estas solicitudes entrantes -el Servicio Web podría buscar su directorio de presentaciones en Power Point y recuperar algunas para los clientes. El problema de este método que utiliza HTTP es que la laptop de Bob debe estar encendida y accesible para poder realizar este trabajo. Por lo general, Bob está mucho fuera de su oficina y ¿Cómo puede él permitir el acceso de clientes externos a su laptop a través de un Firewall? Esto parece un tanto difícil de gestionar.

Después de haber leído respecto del uso de transportes alternativos, Bob propone un nuevo diseño:

Creará un Servicio Web para su laptop, pero en vez de aceptar las conexiones entrantes a través de HTTP, utilizará SMTP (e-mail), tal como se muestra en la Figura 11. Los clientes pueden enviarle solicitudes del Servicio Web para buscar y recuperar su almacenamiento local de archivos de Power Point. Para realizar esto, Bob creará una aplicación de Cliente Inteligente que genera estas peticiones.

La ventaja de este diseño es que Bob y otras personas pueden ahora beneficiarse con las ventajas de funcionalidad distribuida que proporciona el e-mail. Bob comparte su nueva aplicación del Servicio Web con otros 50 consultores en Contoso. Lo que tenemos ahora es una forma muy dinámica de utilizar Servicios Web para buscar archivos de Power Point que se almacenan localmente en una serie de máquinas.

Por ejemplo, Joe está buscando una presentación de Power Point sobre el tema de C#. Él ingresa la consulta "C#" en la aplicación de Cliente Inteligente. Esto crea una solicitud del Servicio Web que se envía utilizando SMTP hacia una lista de distribución de e-mails que contiene los 50 consultores que ejecutan el Servicio Web de Bob.

Una vez que se recibe el mensaje, el Servicio Web que se ejecuta sobre cada una de estas laptops realiza una búsqueda basada en el criterio de Joe. La lista con los resultados es entonces enviada a la aplicación de llamada de Joe, que puede visualizarlas en la medida que se reciben las respuestas (una vez más, usando SMTP).

Joe puede ahora comenzar la búsqueda a través de los resultados que ha recibido de su consulta (hay que recordar que al igual que un e-mail, él no necesita que todos le respondan -simplemente las personas necesarias para que él tenga la presentación de Power Point que está buscando). Cuando la encuentra, se puede realizar una petición similar usando el Servicio Web sobre SMPT para realmente adquirir la presentación.

Tabla 2. Implementación de Transportes Alternativos

Transporte	Descripción
Indigo	Indigo es el codename del entorno de informática distribuida de la próxima generación de Microsoft. Indigo ofrece la promesa de transportes múltiples para Servicios de Web, junto con un modelo de programación unificado. Indigo en forma nativa soporta HTTP, TCP, y MSMQ en el CTP de marzo de 2005. El modelo de programación permite una interfaz fácil de extender para otros transportes.
WSE (Web Services Enhancements)	Para aquellos que quieran implementar esto en la actualidad, también se pueden realizar transportes alternativos utilizando WSE. WSE proporciona un API llamado transporte personalizado, que permite utilizar transportes que no sean HTTP. Los transportes personalizados de hoy incluyen ejemplos para MSMQ ww.codeproject.com/cs/webservices/SoapMSMQ.asp), IBM MQ Series (http://workspaces.gotdotnet.com/wsemqs), SMTP (http://hyperthink.net), UDP (http://dynamic-cast.com), TCP, e In Process (ambos ejemplos se envían con WSE).* *Nota: La interoperabilidad usando TCP no es soportada utilizando WSE.
JMS (Java Message System) API	Una serie de proveedores de servidores de aplicaciones Java están en la actualidad proporcionando soporte de Servicios Web a través de JMS. Esto permite las peticiones y respuestas de SOAP sean procesadas en una cola JMS.
JAXMail (Java API for XML Mail)	JAXMail (parte de Sun JWSDP) es una extensión de JAX-RPC (Java Web Services) para proporcionar soporte para el protocolo SMTP.

Los modelos que hemos visto en este artículo pueden generar más preguntas que las respuestas que proporcionan. Es de esperar, no obstante, que se vea que utilizar Servicios Web con transportes alternativos puede ofrecer una nueva variedad de aplicaciones que hasta el momento habían sido restringidas por el uso de HTTP.

Una de las preguntas que pude surgir es "¿Cuándo se debe implementar un transporte que no sea HTTP?" Para colaborar con la respuesta y para sintetizar los contextos enumerados en este artículo, pueden referirse a la Figura 14.

Sobre el autor

Simon Guest es Program Manager en el grupo de Architecture Strategy en Microsoft Corporation, y se especializa en la interoperabilidad y la integración. Simon realizó un Master sobre Seguridad IT en la Universidad de Westminster, Londres, y es el autor de Microsoft .NET y de J2EE Interoperability Toolkit (Microsoft Press, Sept. 2003).

Se puede llegar a Simon a través de su blog en http://www.simonguest.com.

Relación de la estrategia de producto con la arquitectura

Charlie Alfred

Introducción

Existen sistemas para generar valor para sus interesados. Lamentablemente, este ideal sólo se cumple en forma limitada. Los métodos actuales de desarrollo, como cascada, espiral, ágil, con frecuencia brindan una dirección incompleta e inadecuada a los interesados, arquitectos, y personas a cargo del desarrollo.
El presente documento presenta dos conceptos esenciales: modelos de valor y estrategia de arquitectura, que no figuran en muchos procesos de desarrollo.

La creación de modelos de valor bien definidos brinda una dirección que mejora la calidad de las decisiones de intercambio, en especial en sistemas configurados para muchos usuarios en diversas configuraciones. La existencia de una estrategia de arquitectura claramente establecida brinda una dirección coherente y de alto nivel para el sistema, de la misma forma que la Constitución de Estados Unidos lo hace para su nación. Finalmente, el presente documento mostrará cómo estos dos conceptos se pueden integrar de forma eficaz con los métodos de espiral, cascada o ágil.

Nuestras maneras actuales de construir sistemas complejos de software intensivo no son efectivas. Esto no significa que sean inadecuadas. Muchos sistemas hechos utilizando métodos de cascada, espiral o ágiles se configuran con éxito y pueden satisfacer a sus interesados. Sin embargo, muchos no lo logran, y por causas que pueden corregirse.

Los procesos tradicionales para construir sistemas de software intensivo, como los métodos de cascada y espiral, se basan en los requerimientos para brindar una dirección. Una creencia errónea típica es que los requerimientos son afirmaciones que describen el problema. Según Greenfield y Short [1], no es así. Ellos definen la solución desde un punto de vista de los usuarios y el patrocinador del sistema.

Los Requerimientos tienen algunas fallas notables:

- Los requerimientos con frecuencia utilizan una estructura binaria. Funcionan como calificación de aprobado/no aprobado en un curso escolar, y brindan poca ayuda para la toma de decisiones de intercambio. Por supuesto, estas decisiones de intercambio se deben tomar en alguna parte del proceso. Con frecuencia se toman en forma implícita, y sin considerar totalmente las consecuencias.
- Los requerimientos con frecuencia se utilizan como base para especificar los criterios de aceptación evaluables para un sistema. En el proceso de especificarlos, se toman en forma implícita importantes decisiones sobre el diseño, sin considerar en forma total las implicancias. Con el tiempo, estas decisiones se deberán revertir con un costo significativo, o sino terminarán limitando el potencial del sistema.
- Los requerimientos tienden a considerar a todas las personas de un determinado tipo de usuario como iguales. Por ejemplo, el uso de escenarios típicos para un sistema médico podría referirse a médicos y enfermeras, mientras que los que se usan para sistemas inmobiliarios podrían referirse al comprador, vendedor, agente o prestamista. El problema es que dos médicos no son el mismo, y no necesariamente tendrán las mismas necesidades. Existe una buena razón por la cual los restaurantes más conocidos tienen muchas opciones de platos en el menú.
- Con frecuencia, no está definida la información necesaria para tomar decisiones de arquitectura de software efectivas. Todos los sistemas se configuran en entornos que colocan obstáculos importantes en su camino. Superar estos obstáculos es responsabilidad de cada sistema, y tener éxito a pesar de estos es indicio de un sistema eficaz. Sin embargo, salvo que las personas a cargo del desarrollo tengan un entendimiento extremadamente profundo del dominio del problema, no tienen la sagacidad adecuada para realizar juicios valorativos correctos.

Figura 1. Agilidad del Sistema

Caracteristica	Mecanismo	Ejemplo
Brindar Funciones Útiles	Brindar una Nueva Capacidad Importante	Given Imaging Inc desarrolló la cápsula de 11x26 mm que contiene una cámara digital capaz de atravesar y tomar fotos de partes del tracto superior Gastrointestinal del paciente.
	Mejorar la Calidad de Capacidades Existentes	Intel Pentium 4 CPU utiliza una regla de diseño de 90 mm (reducida de los 130 mm) y puede llevar a cabo 13 mil millones de instrucciones por segundo.
Superar obstáculos	Factores de Limitación de Dirección	Muchos fondos comunes de inversión y carteras privadas gestionadas están obligados a cumplir con limitaciones de inversión. Los sistemas de cumplimiento pre-comerciales analizan actividades propuestas para verificar que la cartera siga en cumplimiento
	Identificar y Mitigar Riesgos	Un sistema de detección corta-viento en aviones comerciales detecta la presencia de microráfagas que pudieran provocar estallido del avión en proceso de aterrizaje
Sobrellevar los cambios	Oportunidades de explotación	eBay reconoció que la población creciente de consumidores con acceso a Internet creó una oportunidad para brindar capacidad de remate electrónico
	Adaptarse Rápidamente a las Nuevas Condiciones	Eastman Kodak reconoció el salto tecnológico que permitió el uso de la fotografía digital y logró la penetración en el mercado en este segmento para compensar la baja en ventas de película. Polaroid no tuvo tanto éxito en este aspecto.

Al mismo tiempo, los usuarios influyentes y los patrocinadores del sistema generalmente tienen esta experiencia, pero con frecuencia no tienen la pericia en cuanto a tecnología y sistemas como para saber cuándo será necesario.

Desarrollo ágil

Los métodos ágiles como PROGRAMACIÓN EXTREMA y SCRUM tienen un enfoque levemente diferente. Estos métodos ponen énfasis en algunos cambios útiles, como la colaboración alta entre interesados y personas a cargo del desarrollo, y repeticiones de proyectos muy cortos para conseguir un feedback continuado. La teoría es que la interacción continua entre interesados y las personas a cargo del desarrollo constituye un método más confiable para la navegación del proyecto que una alta inversión inicial en requerimientos por escrito.

Además, los métodos ágiles tienden a favorecer los enfoques más orgánicos y reactivos (refabricación) por encima de los que tienen una guía más prescriptiva (arquitectura). Los que proponen métodos ágiles tienen la teoría de permitir la evolución de la arquitectura de un sistema. En algunos casos, este enfoque puede ser eficaz. Un ejemplo es cuando las necesidades del usuario o las condiciones de competitividad cambian rápidamente. No obstante, hay muchos casos en donde este enfoque puede ser riesgoso. Uno en particular sería cuando el producto debe desarrollarse para funcionar en muchos entornos diferentes y/o satisfacer a los interesados con diferentes necesidades y prioridades.

El problema principal con los enfoques de cascada, espiral y ágiles es que el desarrollo de software con frecuencia procede sin información crítica y sin las herramientas necesarias para reunir esta información. Un bote valioso, una radio que funcione, y un conjunto completo de velas son todos necesarios, pero no necesariamente son suficientes. Un marinero experimentado no pensaría zarpar del puerto sin un buen conjunto de mapas náuticos, pronóstico del tiempo a largo plazo y una manera confiable de rastreo y localización del bote.

El presente documento debate dos procesos: modelación del valor y estrategia de arquitectura. Demostrará cómo el uso efectivo de estas técnicas podrá:

- Captar información esencial sobre el alcance del problema que permita a usuarios y personas a cargo del desarrollo realizar intercambios efectivos.
- Permitir que se puedan identificar y priorizar con éxito los obstáculos significativos.
- Permitir que se exprese la estrategia de arquitectura de manera clara y concisa para que todos los interesados la entiendan.

Los sistemas intencionales se desarrollan para crear valor para sus interesados. En la mayoría de los casos, este valor se considera beneficioso porque estos interesados juegan roles importantes en otros sistemas. A su vez, estos otros sistemas existen con el fin de crear valor para sus interesados. Esta calidad recurrente de sistemas es una clave en el análisis y entendimiento de los flujos de valor. La siguiente sección (Descubrimiento de Modelos de Valor) trata este tema en más detalle.

La lista en la página anterior presenta tres características vitales de un sistema intencional. Se describen dos mecanismos para lograr cada característica y un ejemplo realista se brinda para cada mecanismo.

Estas tres características son la base del modelo de valor. Para identificar y trabajar más fácilmente con estos, necesitamos reducir a cada uno a su forma elemental:

Expectativa de Valor . Expresa una necesidad de una función en particular, incluido lo que se suministra (capacidades), cuán bien se suministran (atributos de calidad), y cuán beneficiosos son los diversos niveles de calidad (función de utilidad). Por ejemplo, un conductor de automóvil podría tener una expectativa de valor para una manera rápida y segura de que el auto pueda detenerse cuando circula a 60 millas por hora.

Fuerza Opuesta. Representa la fuerza natural o impuesta en un entorno donde se configura un sistema que dificulta el buen cumplimiento de la expectativa de valor. Por ejemplo, cuán eficazmente podrá un auto detenerse a 60 millas por hora dependerá del tipo de superficie (pavimento o grava), la inclinación (pendiente abajo o arriba), las condiciones (seco, húmedo, hielo) y el peso del vehículo.

Catalizador de Cambio. Representa la fuerza o situación en el entorno que causa el cambio de las expectativas de valor, o un diferente impacto de factores restrictivos. Por ejemplo, las disminuciones en el costo del chip de memoria y los aumentos en la densidad de almacenamiento son un catalizador para la fotografía digital.

En lo que resta del presente documento, nos referiremos a las fuerzas opuestas y catalizador de cambio como factores restrictivos, y nos referiremos a los tres en conjunto como orientadores de valor.

No es tan simple

Si un sistema debe ser eficaz en cuanto a satisfacer los modelos de valor de sus interesados, necesitará poder identificarlos y analizarlos. Los enfoques tradicionales, como escenarios o requisitos comerciales o de marketing, comienzan enfocándose en los tipos de actores con los cuales interactúa el sistema. Este enfoque tienen varias limitaciones importantes:

Desafíos de Arquitectura en Sistemas de Cumplimiento Pre-Comercial para Cartera

Las reglas de cumplimiento deberán especificar limites superiores e inferiores sobre el porcentaje de bienes de cartera que se pueden invertir en categorías específicas, como ser tipo de título valor, sector de la industria, o región geográfica. Los porcentajes actuales de distribución en una cartera pueden variar en base a cambios de precios, comercios y acciones corporativas. Si una cartera de clientes no cumple con sus normas y pierde dinero, se le podrá exigir indemnizar a sus interesados. En situaciones donde la cartera esté dentro de los límites de cumplimiento, existe un bajo riesgo de que el comercio particular cause un incumplimiento. Sin embargo, si el mercado se mueve con rapidez o el volumen de actividad comercial es pesado, una cartera cercana a uno de más límites tiene mayor riesgo de caer en incumplimiento.Un desafío mayor es que los mercados que se mueven rápidamente o los plazos de volumen alto de actividad comercial sean exactamente el escenario donde los comerciantes deban responder con rapidez, para conseguir los mejores precios. Aun así, esto puede ser la misma situación cuando existen muchas actividades pendientes y situaciones de cambio de precio a evaluar, haciendo más compleja la verificación de cumplimiento.En este caso, el tamaño de la cartera, volumen de actividad comercial y volatilidad del mercado se combinan para crear un conflicto entre la necesidad para verificar cumplimiento (técnica de mitigación de riesgo) y la necesidad de comerciar en forma eficiente (que impacta en la cartera ROI). Para ser eficaz, la arquitectura de la organización de gestión de cartera y su sistema de información deberán encontrar la manera de equilibrar el intercambio entre riesgo de cumplimiento y actividades comerciales oportunas.

- Se enfoca más en lo que hacen los actores, y menos en porqué lo hacen.
- Tiende a estereotipar actores en categorías, donde todos los de un tipo son en esencia lo mismo (comerciantes, gestores de cartera, o administradores de sistema por ejemplo.).
- Tiende a ignorar diferencias en factores restrictivos (por ejemplo: ¿un comerciante de acciones en Nueva York es lo mismo que uno en Londres? ¿Comerciar en el mercado abierto es lo mismo que comerciar durante el día?) .
- Se basa en resultados binarios: el requisito se cumple o no. El caso de uso se completa satisfactoriamente o no.

Existe una razón práctica y lógica por la cual este enfoque es popular. Utiliza razonamiento secuencial y basado en clasificación, y es más fácil de enseñar y explicar, y puede producir un conjunto de objetivos fáciles de verificar. Por supuesto, si la simplicidad fuera la única meta considerada, todavía estaríamos caminando o cabalgando para transportarnos de un lugar a otro. En su libro Ventaja Competitiva, Michael Porter [7] trata el con cepto de cadenas de valor en el contexto de planificación estratégica corporativa:

"Aunque las actividades de valor son los bloques de construcción de la ventaja competitiva, la cadena de valor no es un conjunto de actividades independientes, sino un sistema de actividades interdependientes. Los enlaces son relaciones entre la manera en que se realiza una actividad de valor y el costo o rendimiento de otra."
"Los enlaces no solo existen dentro de una cadena de valor de una empresa (enlaces horizontales) sino entre la cadena de valor de una empresa y las cadenas de valor de proveedores y canales (enlaces verticales). La manera en que se realizan las actividades de los proveedores y canales afecta el costo o rendimiento de las actividades de una empresa (y vice versa)".
Si uno piensa en una empresa (o cadena de suministro) como sistema, y cada actividad de valor importante (fuentes, recepción, fabricación y demás) como subsistema, podemos generalizar la noción de cadenas de valor y enlaces:

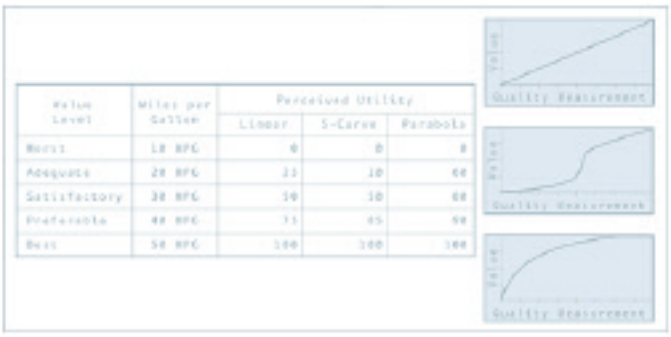
- Cada entidad (actividad de valor) tiene su propio modelo de valor para representar sus expectativas de valor y factores restrictivos.
- Cada enlace describe cómo el modelo de valor de una entidad se relaciona con el modelo de valor de la entidad a la cual está ligado.
- Cada enlace entre dos entidades en el mismo sistema es lo que Porter considera enlace horizontal. Cada enlace entre entidades en diferentes sistemas es un enlace vertical.

Porter también se refiere al concepto de diferenciación, cuando dos entidades que realizan el mismo conjunto de actividades de valor tienen diferente comportamiento. Un ejemplo simple podría ser un taxi y un colectivo. Mientras que ambos brindan transporte por tierra, estos dos contextos tienen diferentes características. El colectivo es relativamente económico y sigue una ruta y programa pre determinados. El taxi está disponible a pedido (salvo cuando uno realmente lo necesita), funciona de un punto a otro, es más costoso, y tiene un número limitado de pasajeros. Cuando llueve, el costo extra de un taxi carecerá de importancia.

Problema de equilibrio

En lo sucesivo en el presente documento, usaremos el término grupo de valor para referirnos a una entidad abstracta

Figura 2. Curvas de Utilidad



que realice un tipo general de actividad de valor. Contexto de valor se usará para referirse a una forma específica de grupo de valor que tenga diferencias significativas en expectativas de valor, fuerzas opuestas, y catalizadores de cambio con respecto a otros contextos en el mismo grupo.

Tanto los grupos de valor como los contextos de valor tienen sus propios modelos de valor. El modelo de valor de un grupo representa los aspectos comunes de todos los contextos que especializan a tal grupo. Cada contexto de valor especializa el modelo de valor de su grupo. El conjunto de modelos de valor para todos los contextos en un grupo brinda ideas importantes en las diferencias entre lo que cada uno espera, y cómo se ve afectado por su entorno.

¿Porqué es importante esto? La arquitectura de un sistema debe realizar un acto de equilibrio delicado que involucre a sus orientadores de valores. Esto puede ser engañoso en un sistema de un solo contexto. Igual que los editores de texto simples, analistas de diferencias de archivos, y muchas otras utilidades de escritorio de PC. En un sistema de contexto simple, es igualmente posible tener interdependencias y conflicto entre las combinaciones de expectativas de valor y factores restrictivos.

> "El principal problema de los métodos de cascada, espiral o ágil reside en que los desarrollos de sofware por lo general siguen adelante sin obtener alguna información muy importante y sin las herramientas necesarias para reunirla."

No obstante, se vuelve más desafiante. La mayoría de los sistemas complejos tienen múltiples contextos. En otras palabras, al considerar diferentes entornos de configuración, ellos tienen una variación importante en las expectativas de valor, fuerzas opuestas, y catalizadores de cambio. Al aumentar cualquiera de los números de contextos, o disminuir su grado de compatibilidad, se vuelve más difícil satisfacerlos a todos con una sola arquitectura. Mientras que existen muchas técnicas para tratar esta situación, el primer paso es reconocer cuándo enfrentarlas.

Muchos sistemas tienen solo algunos contextos. Esto ocurre mayormente con sistemas que se configuran para uso interno dentro de una organización. Los diferentes entornos de configuración pueden tener diferentes factores restrictivos. Por ejemplo, un sistema para envío de controladores de equipaje

de aerolínea se ve afectado por el clima o un sistema internacional se ve afectado por normativa local. Otras veces, los entornos de configuración tienen diferentes expectativas de valor. Esto es en particular cierto cuando hay diferencias culturales o internacionales. Las enfermeras que trabajan con máquinas de hemodiálisis para pacientes con trastorno renal crónico en un hospital del estado en Europa tendrán diferentes deseos y prioridades de las que realizan la misma tarea en una clínica privada pequeña y ambulatoria en EE.UU. (donde las empresas de seguro privadas pagan los tratamientos).

Muchos otros sistemas tienen gran número de contextos. Esto ocurre más frecuentemente con los productos de centros de tecnología que se desarrollan para la venta o alquiler a una amplia gama de clientes. Las mismas condiciones que causan la variación en los sistemas de leve contexto, se producen en grandes cantidades debido a:

- El número de contextos de configuración puede ser miles o millones de veces más grande,
- Las organizaciones o sistemas en que los interesados participan pueden tener muchos grupos diferentes de expectativas de valor,
- Los catalizadores que provocan un cambio significativo en cada entorno de configuración puede que sean muy diferentes.

En resumen, un modelo de valor capta a los orientadores que determinan cuán satisfecho está un particular segmento del mercado, y cuán difícil será satisfacerlos.

Curvas de Utilidad

La sección anterior hizo referencia a un concepto importante llamado curva de utilidad. Simplemente, una curva de utilidad es un plano de una escala de medición a una segunda. La primera escala representa una variable de resultado que se podrá cuantificar. La segunda escala es el nivel de valor (satisfacción, utilidad) que se genera. El ejemplo más común de una curva de utilidad es la utilizada para planificar puntajes de evaluaciones en grados de letra para exámenes de la escuela secundaria o universidad. Según lo que le mostramos, un buen plano de curvas de utilidad es absolutamente esencial para tomar decisiones de intercambio efectivas.

Figura 2 ilustra un ejemplo simple. La primera escala representa la economía de combustible de autopista y ciudad combinados EPA para un automóvil. La segunda escala representa cinco valores cualitativos:

Peor. El requisito mínimo transitable. Se pierde poco o ningún valor con resultados por debajo de este nivel.
Adecuado. Este resultado representa un resultado por debajo del promedio, decepcionante pero aceptable.
Satisfactorio. Este es el resultado esperado, ni mejor ni peor.
Preferible. Este resultado representa un resultado por encima del promedio, satisfactorio y suficiente, pero no mucho más allá del rango de lo común.
Mejor. El mejor resultado esperable. Se gana poco o ningún valor con resultados que excedan éste.

La figura muestra tres curvas de utilidad distintas. Hay muchas otras figuras posibles, estas representan a tres comunes. La primera curva es lineal, la segunda es en S y la tercera es una parábola. Las tres tienen exactamente los mismos valores peores y mejores. Lo interesante son los valores intermedios. Un aumento de 10 a 20 mpg rinde el 10 por ciento de un valor disponible para la curva en S pero un 60 por ciento para la parábola.

En un sistema de contexto simple el uso de curvas de utilidad para analizar estrategias de arquitectura es directo. El método de Análisis de Decisión descripto por Kepner y Tregoe [2] se puede utilizar para este fin. Cada opción se evalúa con relación a cada expectativa de valor. Las curvas de utilidad se usan para plano del valor de la medida cuantitativa lograda por cada opción con respecto a su valor correspondiente. Entonces, los niveles de valor se miden por prioridad de expectativa, y se suman. Las opciones más preferibles tendrán un total más alto.

El aspecto más desafiante de este método es elegir un mecanismo apropiado para evaluar cada opción con respecto a cada meta deseada. El mejor escenario es cuando el mecanismo provee una medida objetiva (como MPG o caballos de fuerza para un motor de automóvil). En algunos casos el mecanismo puede ser subjetivo. El costo de tener una medición objetiva adecuada debe estar equilibrado contra la extra precisión y la objetividad suministrada. En algunos casos, una evaluación inicial se puede realizar con pruebas subjetivas. Si los resultados están cerca, las mediciones objetivas se pueden realizar para poder elegir entre las mejores opciones.

Figura 4. Formulación de Estrategia de Arquitectura

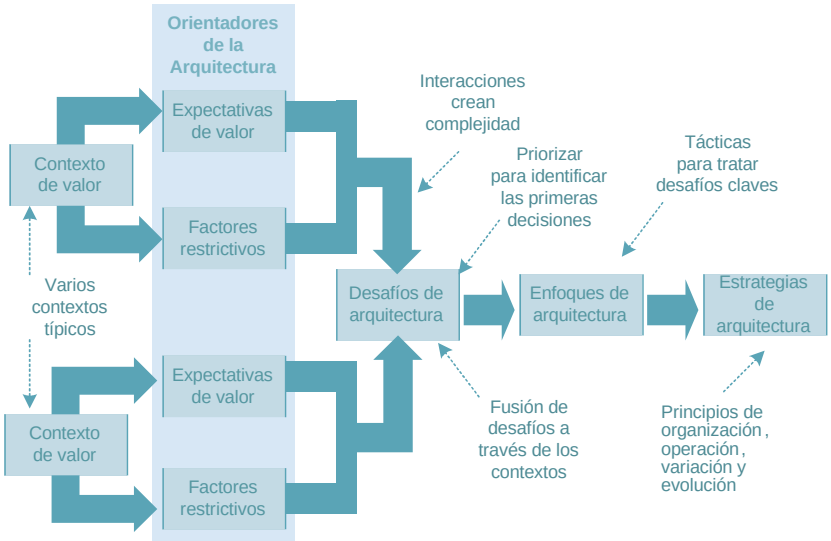
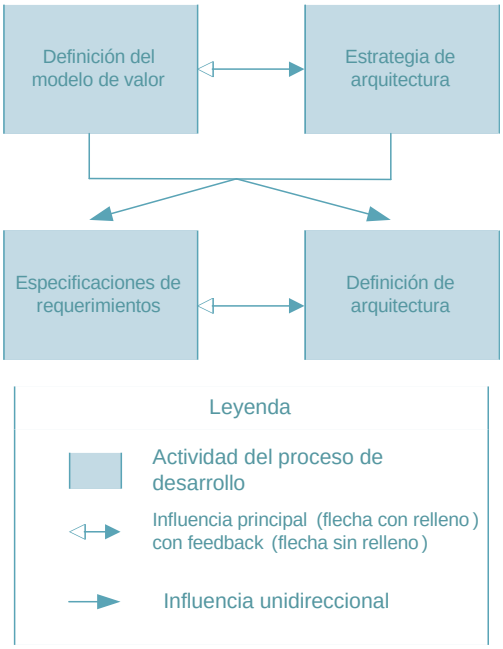


Figura 5. Arquitectura sobre Creación de Valor con Métodos Tradicionales



Desafíos de Arquitectura

Un desafío de arquitectura es una situación donde uno o más factores restrictivos dificultan más satisfacer una o más expectativas de valor. Para decirlo más sencillamente, un desafío de arquitectura es un obstáculo o barrera que el sistema debe superar para brindar valor. Esto es clave. Los obstáculos y expectativas de valor son como el yin y el yang. Si los obstáculos no están presentes, caen los valores, porque el resultado es fácil y cualquiera puede lograrlo. El agua embotellada es una excepción notable a esta regla.

Dentro de cualquier contexto, la identificación de desafíos de arquitectura implica evaluar:

- ¿Qué factores restrictivos impactan en una o más expectativas de valor?
- Si se observan impactos, ¿estos hacen que cumplir las expectativas de valor sea más fácil (impacto positivo) o más difícil (impacto negativo)?
- ¿Cuánto facilita o dificulta las cosas cada impacto? Es suficiente en este caso la escala de bajo, medio o alto.

La Figura 3 describe algunos desafíos de arquitectura que ocurren en un sistema de cumplimiento pre-comercial de gestión de cartera. Un debate más profundo de los desafíos de arquitectura y un estudio de situación se podrán encontrar en [4].

Los desafíos de arquitectura deben considerarse dentro de sus propios contextos. Mientras es posible promediar curvas de utilidad con contextos, no se puede hacer lo mismo con los impactos de factores restrictivos sobre expectativas de valor. Por ejemplo, en caso de un servidor de Internet que brinda páginas a usuarios en dos contextos. Un contexto accede a información estática, como documentos de referencia. Desean un tiempo de respuesta entre 1 y 3 segundos. El otro contexto accede a información muy dinámica, como puntaje de eventos deportivos en curso. Están conformes con un tiempo de respuesta del rango entre 3 y 6 segundos.

Ambos contextos están sujetos a limitaciones de CPU, memoria, disco y red. Sin embargo al aumentar los volúmenes de pedido por un factor de 10 o 100 estos dos contextos se encontrarán probablemente con obstáculos de escala muy diferentes. En un caso de contenido dinámico, la sincronización de actualizaciones y accesos se convierte en factor restrictivo bajo carga pesada. Para el contenido estático, la carga pesada se puede superar guardando en la memoria las páginas más leídas.

Existe un punto final que debería mencionarse sobre desafíos de arquitectura y sistemas de contexto múltiple. En muchos casos, parecería que un sistema simple es capaz de soportar diferentes contextos. Sin embargo, los contextos de arquitectura que surgen de cada contexto son una herramienta muy buena para evaluar cuán compatibles son estos contextos entre sí. Cuando la misma arquitectura se dirige a contextos incompatibles el resultado nunca es que ambos estén satisfechos. O uno sufre a expensas del otro, o ambos están comprometidos. Un ejemplo de esto es la herramienta semiconductora que intenta soportar contextos de producción e investigación con una arquitectura sola. Dados los diferentes grupos de expectativas de valor (confiabilidad versus flexibilidad), fuerzas opuestas (fab versus lab), y catalizadores de cambio (producción versus experimentos) era improbable que esta unión se pudiera salvar.

Estrategia de Arquitectura

Según lo descripto en anteriores secciones, para formular una estrategia de arquitectura de un sistema se comienza por:

- Reconocer los contextos de valor adecuados y priorizarlos;
- Definir curvas de utilidad para expectativas de valor y priorizar estas expectativas en cada contexto.
- Identificar y analizar fuerzas opuestas y catalizadores de cambio en cada contexto.
- Detectar casos en que los factores restrictivos dificultan cumplir con las expectativas de valor.

La Figura 2 ilustra este proceso. La lista previa de actividades nos muestra la casilla de Desafíos de Arquitectura en la mitad del diagrama. En este punto, trabajamos con una lista de desafíos de arquitectura que se han reunido de todos los contextos. Cada uno de estos desafíos representa un impacto de uno o más factores restrictivos en uno o más expectativas de valor.

> "Si la simplicidad fuera el único objetivo con valor , todavía estaríamos todos caminando o montando caballos para trasladarlos de un lado al otro."

Según muestra el diagrama, antes de empezar a tratar cada desafío, necesitamos priorizarlos. Las siguientes observaciones explican porqué:

- Cuanto antes se toma una decisión, es probable que se puedan delimitar más cantidad de asuntos.
- Cuanto más tarde se toma una decisión, menos opciones estarán disponibles.

Como resultado, solo tiene sentido reservar las primeras decisiones de arquitectura como las que rendirán el mayor valor.

Existen varios criterios a utilizar para priorizar los desafíos de arquitectura. Recomendamos un equilibrio entre lo siguiente:

Importancia. ¿Cuán alta es la prioridad de las expectativas de valor que se ven afectadas por el desafío? Si estas expectativas de valor son específicas de algunos contextos, ¿cuál es la prioridad relativa de estos contextos?
Magnitud. ¿Cuán grande es el impacto en las expectativas de valor causado por factores restrictivos?
Consecuencia. ¿Cuántas opciones realistas parece haber? ¿Estas opciones tienen diferencias importantes en dificultad o efectividad?
Aislamiento. ¿Cuán aislado está el impacto de las opciones más realistas? Cuanto más divulgado esté el impacto, más peso tendrá el factor.

Una vez priorizados los desafíos de arquitectura, se formulan los enfoques para los de mayor prioridad. Mientras que las técnicas como estilos y patrones de arquitectura [3] [8] pueden ayudar, esta es un área donde la vasta experiencia en temas de problema-solución es invaluable. Los enfoques eficaces a desafíos importantes son resultado de un trabajo habilidoso, profundo, de esfuerzo y meticuloso. Esto es verdad, independientemente de si el problema tiene que ver con cirugía, gerencia ejecutiva o arquitectura de software.

Al tratar cada desafío, su enfoque delimitará las soluciones para otros desafíos, y a veces creará nuevos. Si las prioridades de desafío de arquitectura son correctas, la mayoría de las limitaciones posteriores serán adecuadas. Sin embargo, en algunos casos el enfoque a un desafío de alta prioridad impactará negativamente en varios desafíos de prioridad levemente menor. La prioridad combinada de desafíos impactados compensará al desafío de mayor prioridad. En este caso, es aconsejable respaldar y formular un enfoque diferente al desafío original.

Levar anclas

Finalmente, una vez formulados los enfoques al conjunto de desafíos de alta prioridad, se podrá expresar la estrategia de arquitectura. El arquitecto analiza el conjunto de enfoques y considera el conjunto de principios de guía en las siguientes áreas:

Organización. ¿Cómo se organiza el sistema en subsistemas y componentes? ¿Cuál es la composición y responsabilidades de cada uno? ¿Cómo se puede configurar el sistema en una red? ¿Qué tipos de usuarios y sistemas externos existen? ¿Dónde se ubican y cómo se conectan?
Operación. ¿Cómo interactúan los componentes? ¿En qué casos es sincrónica la comunicación? ¿En qué casos es asincrónica? ¿Cómo se coordinan las acciones de los componentes? ¿Cuándo es aceptable configurar un componente o realizarle un diagnóstico? ¿Cómo se detectan errores, cómo se diagnostican y corrigen?
Variabilidad. ¿Cuáles funciones importantes del sistema se permiten para variar de un entorno de configuración a otro? ¿Qué opciones se soportan para cada función, y cuándo se puede optar (por ejemplo, compilar, enlace, instalación, puesta en marcha, o tiempo de funcionamiento)? ¿Qué dependencias existen entre puntos de variación?
Evolución. ¿Cómo está diseñado el sistema para soportar el cambio mientras mantiene su estabilidad? ¿Qué tipos específicos de cambio importante se han anticipado, y cuales son las maneras preferibles de tratarlos?

En resumen, la estrategia de arquitectura es el timón y la quilla del barco, brindando dirección y estabilidad. Se espera que sea una declaración breve y de alto nivel de la dirección que todos los interesados deberán entender, y deberá ser relativamente estable durante la vida útil del sistema.

La Figura 3 muestra cómo los Modelos de Valor y Estrategia de Arquitectura se relacionan con los métodos de cascada y espiral. Los Modelos de Valor y Estrategia de Arquitectura operan al primer punto y a un mayor nivel que estos métodos. Cuando se estudian los modelos de valor y se formula la estrategia de arquitectura, brindan una gran base para especificar requerimientos y definir una arquitectura más detallada. El modelo de valor conduce los requerimientos, e influye en la definición de la arquitectura al brindar información para realizar intercambios. La estrategia de arquitectura brinda una más detallada definición de la arquitectura, y un conjunto de requerimientos derivados necesarios para superar los obstáculos conocidos.

Una analogía adecuada es ver a la estrategia de arquitectura como planificación estratégica, y los modelos de valor como análisis de mercado. En esta perspectiva, los requerimientos son objetivos y políticas corporativas. La definición de arquitectura es la organización comercial y el plan operativo, y los casos de uso son equivalentes a procesos comerciales.

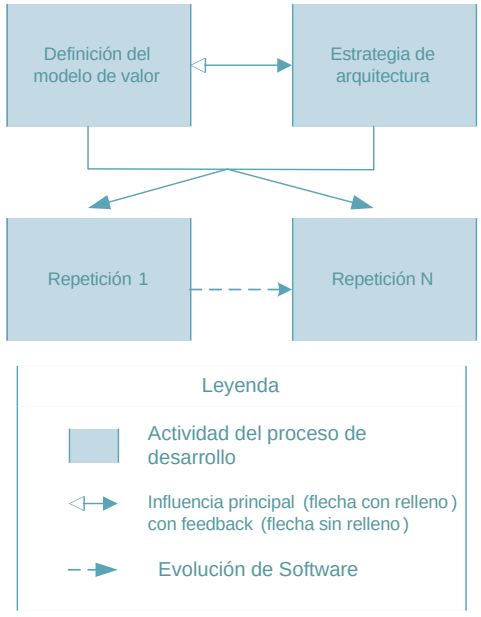
Pocas empresas establecen objetivos corporativos, estructura organizativa, planes operativos, y procesos comerciales sin primero tener una idea clara de su misión, mercados, competidores, recursos y estrategia. Incluso menos empresas efectivas hacen esto.

La Figura 4 muestra cómo los Modelos de Valor y Estrategia de Arquitectura se relacionan con métodos ágiles. PROGRAMACIÓN EXTREMA y SCRUM hacen concesiones para una definición de arquitectura. SCRUM lo hace en forma expresa, esperando que se defina la arquitectura en la primera repetición de 4-5 semanas.

PROGRAMACIÓN EXTREMA lo hace en forma implícita. Uno de los 12 principios clave de PROGRAMACIÓN EXTREMA se llama System Metaphor (Metáfora de Sistema).

Este principio no se usa tan frecuentemente ni se entiende tan bien como sus parientes famosos: Small Releases, Pair Programming, Test Driven Development. En los comienzos de PROGRAMACIÓN EXTREMA, el equipo que trabajaba en el Chrysler Payroll System grande y complejo necesitaba una buena forma de describir la gestión de flujo de trabajo a las personas a cargo del desarrollo Chrysler.

Figura 6. Integración de Arquitectura sobre Creación de Valor con Métodos Ágiles.



Alguien tuvo la idea de esbozar una analogía entre flujo de trabajo de la nómina de empleados y una línAlguien tuvo la idea de esbozar una analogía entre flujo de trabajo de la nómina de empleados y una línea de montaje de un automóvil. La metáfora cuadró y las personas a cargo del desarrollo Chrysler lo pudieron entender. ea de montaje de un automóvil. La metáfora cuadró y las personas a cargo del desarrollo Chrysler lo pudieron entender.

Enhebrando una historia

El sitio Web de PROGRAMACIÓN EXTREMA [6] define al System Metaphor como:

Lo que Programación Extrema (EXTREME PROGRAMMING) usa en vez de una arquitectura formal. Una historia compartida simple de cómo funciona un sistema, una metáfora. Esta historia típicamente involucra un montón de clases y patrones que dan forma al flujo principal del sistema que se construye.

Lo que PROGRAMACIÓN EXTREMA menciona como "arquitectura formal" es más lo que nosotros mencionamos antes como definición de arquitectura. Una estrategia de arquitectura juega el mismo papel que una metáfora de sistema, sin ser una metáfora. Esto constituye una ventaja importante, ya que las metáforas realmente efectivas (como la utilizada en Chrysler) pueden ser difíciles de conseguir. En comparación, los principios clave, concisos y claros son fáciles de determinar y fáciles de entender. Una persona no necesita salir y ver la película Hidalgo para entender lo que significa "vida, libertad, y búsqueda de la felicidad".

En resumen, el modelo de valor nos ayuda a entender y comunicar información importante sobre fuentes de valor. Algunos de los temas importantes que trata son cómo fluye el valor, porqué se producen las semejanzas y diferencias en las expectativas de valor y factores externos, y qué subconjunto de ese valor busca nuestro sistema para ser satisfactorio. Es trabajo del arquitecto satisfacer estas expectativas de valor resolviendo las fuerzas que influyen en el sistema en general, fuerzas que son específicas de ciertos contextos y fuerzas que se espera que cambien con el tiempo. En este aspecto, la arquitectura es similar a viajar en avión jet, el piloto debe transportar pasajeros en forma segura a un destino conocido, mientras equilibra las leyes de aerodinámica, las capacidades del avión, y las condiciones climáticas actuales y futuras. El enlace entre modelos de valor y arquitectura de software es claro y lógico, y se puede expresar en nueve puntos listados a continuación:

1. Productos y sistemas de software intensivo existentes para brindar valor.

2. El valor es una cantidad escalar que incorpora percepciones de utilidad marginal e importancia relativa a través de las distintas metas. Los intercambios entre metas son una consideración extremadamente importante.

3. El valor existe en múltiples niveles, algunos contienen el sistema objetivo como proveedor de valor. Los modelos de valor para estos alcances contienen los orientadores primarios de arquitectura de software.

4. Los modelos de valor que se encuentran encima de estos en la jerarquía pueden causar que los modelos de valor de sus hijos cambien. Estos es un aporte importante para la formulación de principios de evolución para el sistema.

5. Para cada grupo, los modelos de valor son homogéneos. Contextos de valor, expuestos a diferentes condiciones de entorno, tienen diferentes expectativas de valor.

6. El patrocinador de desarrollo para el sistema tiene diferentes prioridades para tratar de satisfacer a los diferentes contextos de valor.

7. Los desafíos de arquitectura son causa del impacto de factores del entorno en las expectativas de valor dentro de un contexto.

8. Los enfoques de arquitectura buscan aumentar el valor tratando primero los desafíos de arquitectura de mayor prioridad.

9. Las estrategias de arquitectura se sintetizan de los enfoques de arquitectura de más alta prioridad teniendo en cuenta normas comunes, políticas y principios de organización, operación, variación y evolución.
Las mayores contribuciones de este enfoque son:

- Las fuentes de valor en el sistema se modelan como conceptos de primera clase. Las expectativas de valor asocian un pequeño número de capacidades con atributos de calidad, curvas de utilidad, y factores externos. Las expectativas de valor están apoyadas en el dominio del valor y contextos y estos dominios capturan los aspectos comunes de las expectativas de valor, mientras que los contextos capturan las importantes variabilidades dentro de un dominio.
- La localización de razonamiento arquitectural también es una entidad de primera clase. Las expectativas de valor se relacionan con los desafíos de arquitectura, que se relaciona con el enfoque de arquitectura, que se relaciona con estrategias de arquitectura. Los interesados ahora pueden ver el proceso pensado detrás de la solución.
- Un efecto colateral muy útil de esta localización es la aumentada capacidad de revisión de arquitectura de software. Ya que el razonamiento detrás de la decisión se hizo explícitamente, es más fácil para otros interesados (patrocinadores de proyecto, expertos de dominio, expertos de tecnología, usuarios finales) identificar aspectos que podrían estar omitidos o incorrectos.

Notas al Pie

Se describe una técnica equivalente para plano de métrica cuantitativa a escala de utilidad en [5].

Referencias

[1] Greenfield, J. y Short, K., Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools,Wiley, 2004. ISBN: 0-471-2084-3.

[2] Kepner, C., Tregoe, B., The New Rational Manager, Kepner-Tregoe Inc., 1997,
ISBN: 0971562717.

[3] Gamma, E., Helm, R., Johnson, R., Vlissides, J., Design Patterns: Elements of Reusable Object- Oriented Software, Addison Wesley 1995, ISBN: 0201633612.

[4] Alfred, C., "Using Architecture Challenges to Formulate Software Architecture ", 2002, Foliage Software Systems, Inc. white paper.

[5] Kazman, R., Asundi, J., and Klein, M. "Making Architecture Decisions, an Economic Approach", SEI Technical Informe: CMU/SEI-2002-TR-35, 2002.

[6] Extreme Programming Core Practices

[7] Porter, M., Competitive Advantage: Creating and Sustaining Superior Performance, MacMillan Inc. 1985, ISBN: 0029250900.

[8] Buschmann, F. et al., Pattern-Oriented Software Architecture, Volumen 1: A System of Patterns, 1996, John Wiley and Sons, ISBN: 0471958697.

Sobre Foliage Software Systems

Foliage brinda ventaja competitiva a través de estrategia de tecnología, arquitectura de software, y desarrollo personalizado de software. Fundado en 1991, Foliage completó más de 175 proyectos para clientes en servicios financieros, semiconductores, atención sanitaria, aviación y comercio electrónico. Los arquitectos, gerentes de proyecto e ingenieros de software en Foliage tienen un promedio de más de 20 años de experiencia y aumentan su conocimiento al producir software de calidad destacada para cumplir con las metas de presupuesto y plazos. La empresa fue nominada para el Deloitte & Touche Fast 50 por tres años consecutivos y para el Software Magazine's Software 500 por dos años.

Permitir una informática a escala Internet

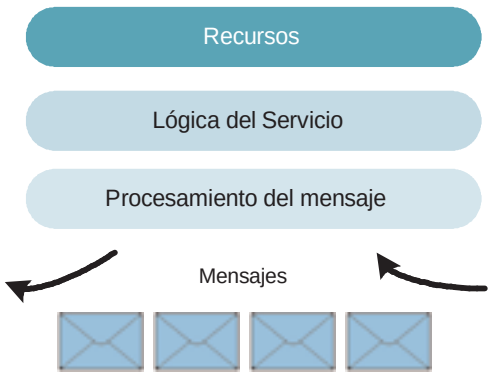
Savas Parastatidis y Jim Webber

Introducción

La Informática de Alto Rendimiento ha evolucionado desde una disciplina que se ocupaba sólo de la ejecución eficaz de un código sobre arquitecturas paralelas a ser una disciplina más estrechamente alineada con el campo de sistemas distribuidos. La HPC moderna está mucho más dedicada a acceder dispositivos especializados y de datos en amplias redes así como también a procesar datos numéricos tan pronto como sea posible. El objetivo de HPC ha cambiado para permitir la utilización más eficaz y transparente de la gran variedad de posibilidades que están disponibles en las redes, de una manera homogénea en la medida que la red eléctrica envía electricidad. Esta visión requiere una inversión arquitectónica e intelectual significativa. En este artículo se analiza un enfoque orientado al servicio para permitir aplicaciones de alto rendimiento a escala Internet en base a un trabajo que se ha completado como parte del programa de e-Science del Reino Unido de £250 millones.

Desde las redes para estaciones de trabajo hasta Internet [1], la comunidad de informática de alto rendimiento se ha dedicado por largo tiempo a crear recursos de informática individual con el objeto de proporcionar la más alta calidad de servicio (en lo que se refiere a tiempo de procesamiento, tamaño del almacenamiento de datos, banda ancha/tiempo de descarga de datos, acceso remoto, algoritmo de integración especial, etc., por ejemplo).

Figura 1. Estructura arquetípica de un servicio



En los últimos años, este avance ha sido dirigido por la visión de "Informática Distribuida" en la que la eficacia computacional, eficacia de almacenamiento y servicio especializado de dispositivos en red arbitrarios será puesto a disposición, de acuerdo a la demanda, de cualquier otro dispositivo con conexión que le sea permitido accederlo.

Al mismo tiempo, la comunidad de sistemas distribuidos ha estado trabajando sobre las tecnologías y principios de diseño para la integración a escala Internet (Web y Servicios Web, por ejemplo). Recientemente, ha surgido el término "Arquitectura Orientada al Servicio (SOA)" como parte de la terminología popular, en algunos sitios debido a la novedad en los entornos de la presentación de los Servicios Web. Si bien los Servicios Web se consideran una tecnología que permite la creación de aplicaciones orientadas al servicio, deberían tratarse como una tecnología de implementación de un conjunto de principios que constituye la orientación al servicio. La promesa de SOA y los Servicios Web es permitir acoplamiento escaso, robustez, escalabilidad, extensibilidad e interoperabilidad. Estas son precisamente las características que requiere un sistema global para "Informática Distribuida" -una palabra de moda popular utilizada para referirse a la informática de alto rendimiento distribuida o informática a escala Internet.

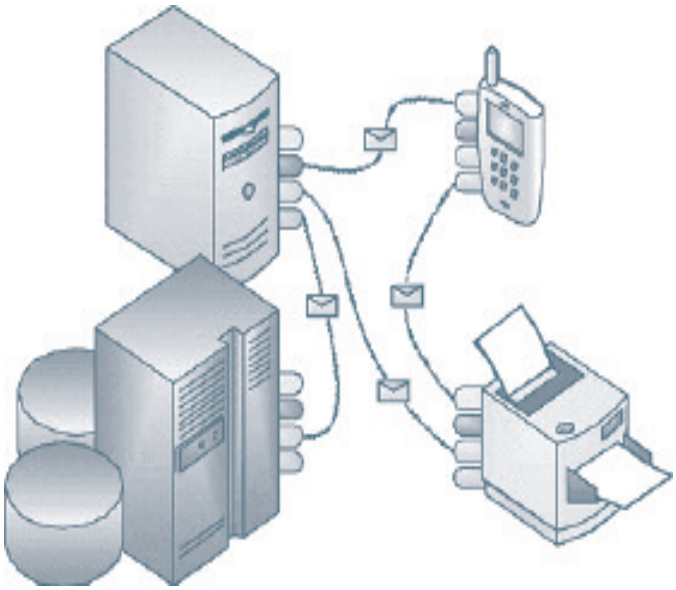
En las próximas secciones se describe la informática distribuida y la orientación al servicio. Con posterioridad, se analiza la forma en la que las aplicaciones de alto rendimiento se pueden diseñar, implementar y mantener utilizando una integración basada en el protocolo y la orientación al mensaje. Seguido de eso, presentamos nuestro enfoque respecto de la manera en la que las aplicaciones HPC de gran escala pueden tratar con un sin fin de recursos y definirse de acuerdo con los principios de SOA.

Informática Distribuida

El término "Informática Distribuida" está sobrecargado y posee diferentes significados para las diferentes comunidades (y proveedores). A continuación se detallan algunas interpretaciones comunes:

- Informática bajo demanda
- Informática como un servicio público
- Informática homogénea
- Interconexión de supercomputadoras
- Informática virtual mundial
- SETI@home y ClimatePrediction.net
- (BOINC-style projects [2])
- Organizaciones virtuales

Figura 2. Las aplicaciones en red se construyen por medio del intercambio de mensajes entre los servicios alojados en dispositivos. En este ejemplo, una aplicación que se ejecuta sobre un dispositivo móvil hace uso de los recursos distribuidos a través de los servicios que se ejecutan en la estación de trabajo (ejecución de un trabajo, por ejemplo), una base de datos y una impresora



Hemos adoptado la posición de que informática distribuida es un sinónimo para la informática a escala Internet con atención sobre la explotación dinámica de los recursos distribuidos para la informática de alto rendimiento. Al construir aplicaciones e infraestructura distribuida fomentamos la aplicación de los mismos principio, técnicas y tecnologías que son típicas en la práctica de los sistemas distribuidos modernos, con la orientación al servicio como el paradigma arquitectónico de elección y los Servicios Web como la tecnología de implementación.

Si bien la orientación al servicio no es un paradigma nuevo de la arquitectura, la llegada de los Servicios Web ha revigorizado el interés en este enfoque. No obstante, es una idea falsa que los Servicios Web son una forma de mágica de software que de algún modo encamina automáticamente al arquitecto hacia una solución de acoplamiento escaso que sea escalable, robusta y dependiente. Ciertamente es posible (y por lo general muy deseado) construir aplicaciones orientadas al servicio utilizando herramientas y protocolos de Servicios Web; sin embargo, de igual manera es posible construir aplicaciones que violen cada dogma y principio arquitectónico de SOA.

De la misma manera que los investigadores y los desarrolladores han remarcado su trabajo para estar en línea con las últimas palabras de moda, el término Arquitectura Orientada al Servicio (SOA) se ha vuelto tenue e impreciso. Debido a que no existe una definición de un servicio aceptada de forma general, proponemos lo siguiente:

Un servicio es la manifestación lógica de algún recurso físico o lógico (por ejemplo, bases de datos, programas, dispositivos, personas, etc.) y/o alguna lógica de aplicación que se expone en la red e interactúan intercambiando mensajes.

Los servicios constan de algunos recursos (datos, programas o dispositivos, por ejemplo), lógica del servicio y una capa de procesamiento del mensaje que se ocupa del intercambio de mensajes (Figura 1). Los mensajes llegan al servicio y actúan según la lógica del servicio, utilizando los recursos del sistema (si existe alguno) tal como es requerido. Las implementaciones del servicio pueden ser de cualquier escala: desde un único proceso del sistema operativo hasta procesos empresariales para la totalidad de la empresa.

Los servicios pueden estar alojados en dispositivos de capacidad arbitraria (por ejemplo, estaciones de trabajo, bases de datos, impresoras, teléfonos, agenda electrónica, etc.) proporcionando diferentes tipos de funcionalidad para una aplicación basada en la red. Esto fomenta el concepto de un mundo conectado en el que ningún dispositivo y/o servicio único está aislado. Por medio de la composición de los servicios y el intercambio de mensajes se construyen aplicaciones interesantes (Figura 2).

Un mensaje es una unidad de comunicación entre servicios. Los sistemas orientados al servicio no exponen abstracciones como clases, objetos, métodos, procedimientos remotos, pero en cambio se basan en torno al concepto de transferencia del mensaje. Por supuesto, las transferencias de mensajes únicos poseen una utilidad limitada por lo tanto existe una preferencia para que varias trasferencias de mensajes sean agrupadas de manera lógica y de esta manera formen patrones de intercambio de mensajes (MEPs) (por ejemplo, un mensaje entrante y saliente que están relacionados pueden formar un MEP "petición-respuesta") para admitir interacciones más variadas. Los MEPs se agrupan para formar protocolos que capturan el comportamiento de mensajería de un servicio (a veces conocido como conversación) para una interacción específica. Estos protocolos pueden posteriormente describirse en contratos y publicarse para ayudar la integración con el servicio (por ejemplo, en WSDL o SSDL [3]).

Protocolos y Contratos

El comportamiento de un servicio en una aplicación distribuida es capturado a través del conjunto de protocolos que éste admite. La idea de protocolo es una separación del mundo orientado al objeto tradicional en el que las semánticas de comportamiento están asociadas con los tipos, expuestas a través de los métodos e integradas con puntos finales en particular (el punto de acceso para un instancia en particular). En cambio un protocolo describe el comportamiento de un servicio visto desde afuera sólo en relación a los mensajes, patrones de intercambio de mensajes y ordenación de aquellos MEPs que son soportados por el servicio.

Figura 3. Integración de recursos empresariales para cumplir con los requerimientos de alto rendimiento de las aplicaciones

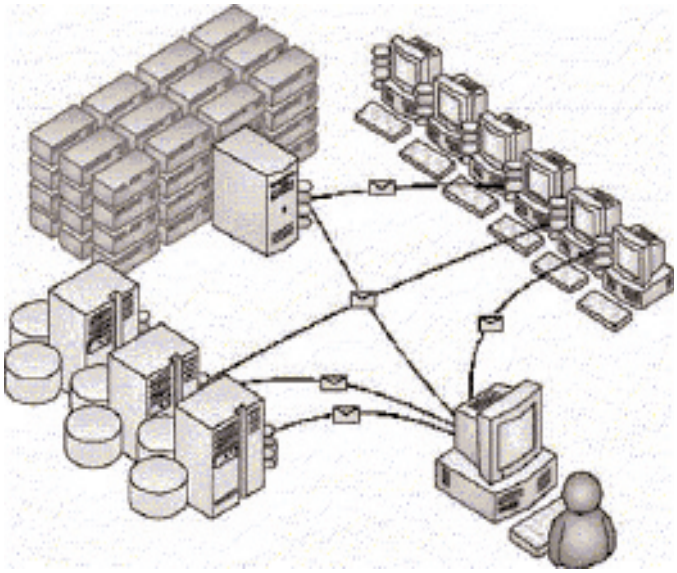
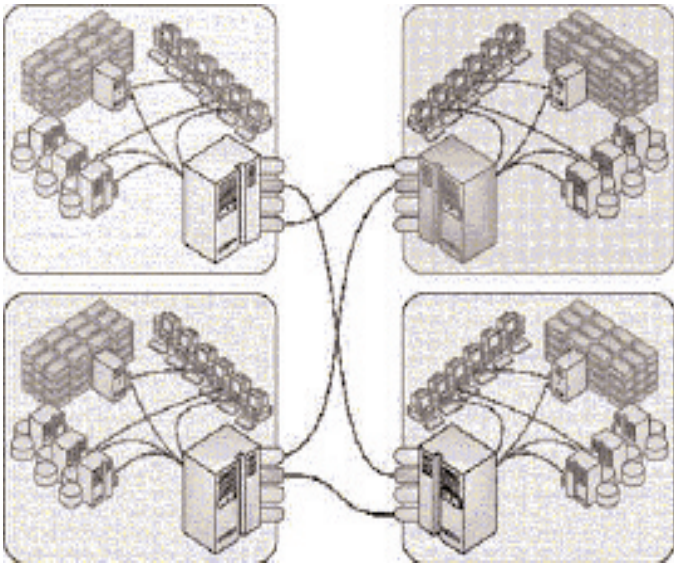


Figura 4. Ejemplo de una aplicación orientada al servicio inter-empresarial con diferentes partes de la empresa representadas como servicios



Los protocolos por lo general se describen a través de contratos a los cuales se adhieren los servicios. Un contrato es una descripción de una política (requerimientos de seguridad o posibilidades de encriptación, por ejemplo) y calidad de las características del servicio (soporte para transacciones, por ejemplo) que un servicio admite y/o requiere, además del conjunto de mensajes y MEPs que transportan información funcional para y desde el servicio.

Muchas organizaciones se están dando cuenta de los beneficios de costo que se obtienen al utilizar grupos de estaciones de trabajo como plataformas alternativas para las instalaciones de supercomputadoras especializadas para sus necesidades informáticas de alto rendimiento. Hasta hace poco estas soluciones basadas en grupos habían sido tratadas como recursos de almacenamiento y/o informáticos dedicados. Las empresas en la actualidad buscan beneficios en términos de bajo costo y rendimiento al utilizar la capacidad inactiva de procesamiento, capacidad de almacenamiento distribuida y otras capacidades disponibles en su infraestructura de terminales existentes, un enfoque que comúnmente se denomina "informática en red dentro de la empresa".

A continuación analizamos los grupos dedicados y el modo en el que la orientación al servicio puede utilizarse para construir estas soluciones antes de proponer un enfoque para construir arquitecturas de alto rendimiento, distribuidas dentro de la empresa.

Las soluciones comerciales específicas basadas en el hardware para la informática de alto rendimiento no son poco comunes dentro de un dominio administrativo de organizaciones con requerimientos para informática de alto rendimiento. Estas soluciones por lo general se implementan de a uno o más grupos de estaciones de trabajo con interconexiones de alta velocidad (por ejemplo Myrinet, SCI, Gigabit Ethernet, etc.). Tales soluciones intentan proporcionar una imagen de computadora única para las aplicaciones por medio de la implementación -en hardware o software- de técnicas que ocultan la distribución de CPUs, memoria y almacenamiento. Los desarrolladores se presentan ante una abstracción de programación típica, la de multiprocesamiento simétrico de memoria compartida. No obstante, estos modelos tienen tendencia a limitar la escalabilidad de los nodos informáticos, que pueden ser un problema para ciertos tipos de aplicaciones paralelas. Las soluciones de middleware orientadas al mensaje especializadas (MPI, por ejemplo) se emplean por lo general para tratar el problema de escalabilidad, pero al costo de requerir administración explícita del grado de paralelismo por parte de la aplicación. Hay una gran cantidad de trabajos en la literatura informática paralela al analizar las ventajas/desventajas de la memoria compartida versus los paradigmas de transmisión del mensaje en las aplicaciones paralelas.

Los grupos dedicados para la informática de alto rendimiento con por lo general considerados y administrados como recursos simples. Para permitir una mejor utilización e estos recursos, es preferible un modelo basado en el servicio. Por ejemplo, el acceso a los recursos es normalmente controlado por un servicio de programación/cola/presentación de un trabajo lo que asegura la utilización óptima del servicio. Las tecnologías de Servicios Web pueden utilizarse para la implementación de estos servicios, y desde luego pueden beneficiarse de la importante inversión en modelación, soporte de tiempo de ejecución eficaz, documentación y educación del usuario en el área de Servicios Web. La naturaleza redactable de las tecnologías de los Servicios Web facilitan las necesidades de implementación para que la calidad de servicio no funcional (por ejemplo, mensajería confiable, seguridad, transacciones, etc.) se incorpore de manera más fácil en un entorno heterogéneo.

Robo de Ciclos de Estaciones de Trabajo del Personal

Un enfoque que ha ganado recientemente un impulso significativo es el desarrollo de tecnologías de robo de ciclos implementadas por el middleware especializado, como el Condor [4]. Este tipo de middleware posibilita la distribución y administración de trabajos informáticos en estaciones de trabajo inactivas, mientras permite que en forma casi instantánea que su propio usuario reclame la estación cuando comienza a utilizar la computadora, al suspenderse, eliminarse o migrar el robo de ciclo a otra estación de trabajo. Sin embargo, las implementaciones más actuales de software de middleware que soporta tales instalaciones no aprovechan los protocolos de calidad de servicio interoperativos y redactables. Como resultado se dificulta la creación de soluciones integradas e interoperativas para informática de alta performance dentro de la empresa.

En informática de alto rendimiento dentro de la empresa en cada estación de trabajo, el sistema de administración de base de datos, mecanismos y demás, muestran una funcionalidad como servicio. Construir tal middleware utilizando los principios de orientación hacia el servicio puede dar como resultado desarrollos con una magnitud de miles de estaciones de trabajo. Un enfoque orientado al servicio puede aumentar la flexibilidad, manejo y valor de tales soluciones, ya que se pueden aprovechar grandes grupos de unidades ampliamente aceptadas de funcionalidad/comportamiento, disponibles como protocolos (Por ejemplo, seguridad, transacciones, mensajes confiables, organización, etc.) y ciertamente, existen esfuerzos que se pueden realizar para hacer sólo eso con sistemas existentes (por ejemplo, el Condor BirdBath [5]).

Las futuras instalaciones de red de suministro dentro de la empresa se construirán alrededor de los servicios estándar suministrados por los sistemas operativos esenciales. Los protocolos de aplicación como WS-Eventing y WS-Management se implementarán y se suministrarán como estándar que las soluciones de red de suministro se pueden implementar y desarrollar fácilmente.

Enfoque Conceptual para Informática HPC Dentro de la Empresa Basada en Windows.

El conjunto de estaciones de trabajo basadas en Windows utilizado por el personal es parte de un dominio de Directorio Activo "CPU sub-utilizadas". La empresa desea aprovechar las capacidades informáticas de estas estaciones de trabajo durante su período inactivo (por la noche, por ejemplo). El administrador del dominio impulsa la implementación .NET de un conjunto de servicios de red que brindan envío, monitoreo, y administración de trabajos fuera de las estaciones de trabajo a través de un directorio activo. Estos servicios aprovechan la plataforma de middleware de Servicios Web esencial (Indigo, por ejemplo) para sus requisitos de seguridad y calidad de servicio (por ejemplo, notificación, mensajes confiables, transacciones y demás). Sólo los usuarios que pertenecen al dominio "CPUs sub-utilizadas" tienen permiso para enviar trabajos. WS-Security se utiliza para requisitos de autenticación y encriptación de mensajes con comprobantes Kerberos salvados del Directorio Activo. Además de las estaciones de trabajo, existe también un grupo dedicado a los requisitos de alto rendimiento de la empresa y centros de datos. Servicios Web instalados en estos recursos exponen funciones de almacenamiento de datos e informática en la red. Las aplicaciones de la empresa se escriben de una manera tal que pueden en forma dinámica mostrar y utilizar cualquier recurso informático distribuido dentro de la empresa. Por lo tanto, en cuanto se activa la función, las aplicaciones intensivas a nivel informático se inician en forma automática para aprovechar la infraestructura distribuida. Los usuarios de tales aplicaciones no tienen conocimiento de estos recursos que se aplican en realidad.

Un ejemplo de enfoque conceptual a la informática HPC dentro de la empresa con uso de Servicios Web se describe en la barra lateral y se ilustra en Figura 3.

Integración estándar

Identificamos un conjunto de servicios genéricos no exhaustivos, funcionalidades y características que pueden ser ofrecidas y/o admitidas por cada dispositivo en la red (Tabla 1). Por supuesto, las funcionalidades específicas del dominio de la aplicación también tendrán que ser soportadas (por ejemplo, un servicio BLAST que se instala en un servidor muy eficaz para realizar un análisis de bioinformática, o un servicio que implementa un algoritmo de estimación para el uso del petróleo.

Las grandes empresas pueden no sólo estar interesadas en implementar una única solución de grupo o simplemente reclamar la capacidad inactiva de procesamiento de partes de su infraestructura organizacional. En cambio, pueden querer focalizarse en el encapsulamiento de conjuntos completos de recursos informáticos detrás de los servicios de alto nivel que, al ser compuestos en conjunto, pueden permitir un nivel de integración que antes era difícil y consumía mucho tiempo debido a la diferente cantidad de tecnologías implementadas.

El modelo para soluciones de arquitectura de alto rendimiento dentro de la empresa es similar a aquél en el que el objetivo es construir soluciones basadas en grupos analizado previamente (los problemas de escalabilidad, acoplamiento escaso, composición, etc., aplican de la misma manera). La calidad de los protocolos de servicio como seguridad (para autenticación, autorización y contabilidad), transacciones, mensajería confiable, notificaciones, etc., son todos parte de una infraestructura subyacente y puede utilizarse tal cual está sin importar el tipo de solución que se implementa. Además, el conjunto de servicios que se utiliza en las soluciones dentro de la empresa puede también utilizarse sin modificaciones (por ejemplo, administración de credencial de usuario, administración de sistemas, implementación de servicios y aplicaciones, soporte del flujo de trabajo, archivo y almacenamiento de datos, mensajería, etc.).

Al igual que previamente, existe aún la necesidad de que los servicios ofrezcan acceso a recursos informáticos y de datos, implementen la planificación, visualización, funcionalidad de algoritmo especializado y demás, dependiendo del tipo de aplicaciones que se implementarán. Esta vez, sin embargo, los servicios se encuentran en un nivel más alto de abstracción ya que el conjunto completo de recursos está encapsulado (Figura 4).

Observamos que a pesar de que la complejidad de los servicios ha aumentado en comparación con aquellas que utilizábamos al construir una solución de grupo, la complejidad de la arquitectura no ha incrementado y los principios y pautas son los mismos. Nuestra aplicación distribuida aún obliga a que los mensajes sean intercambiados y no se presupone que todos dentro de la empresa conozcan las interfaces y funcionamiento de los diversos componentes. Los arquitectos diseñan aplicaciones por medio de la descripción de mensajes y la definición de protocolos que capturan el comportamiento del servicio.

Percibimos que debido a que la granularidad de los servicios aumenta, también la necesidad de incrementar la granularidad de los intercambios de mensajes es mayor. La red es costosa y por lo tanto los arquitectos necesitan diseñar sus protocolos y mensajes de manera adecuada. Del mismo modo que el grado de distribución de una aplicación aumenta, también lo hace la necesidad de un acoplamiento escaso. Mientras que en los grupos únicos o en entornos de empresas más pequeñas, el control total de la infraestructura y el conjunto de tecnologías implementadas es posible, en una solución que afecte a toda la empresa (o mayor) es imprescindible que la integración suceda por medio de protocolos estándares.

También, es muy claro que a medida que la complejidad de la escala de una aplicación aumenta, la funcionalidad de sus servicios se torna aún más abstracta. Desde los servicios que exponen una funcionalidad específica para la red (ejecución de proceso remoto o administración de estaciones de trabajo, por ejemplo) o que proporcionan acceso a un recurso (sistemas de bases de datos o archivos, por ejemplo), hemos avanzado a servicios que admiten el agregado de funcionalidad (colas de trabajo, colas de mensajes, administración de grupos, por ejemplo) o agregado de recursos (red del área de almacenamiento y la base de datos federada, por ejemplo).

Debemos diseñar aplicaciones utilizando bloques de construcción más grandes, ya que las interacciones entre servicios se vuelven cada vez más generales para poder minimizar el efecto de los costos de comunicación entre las diferentes partes de las aplicaciones, y los servicios se tornan cada vez más abstractos y de aplicación más general respecto de los recursos que encapsulan. Cuanto más grande es la escala de una aplicación distribuida, más importante es diseñar mecanismos de aplicación más general, basados en protocolos y declarativos para describir el comportamiento. En este momento es cuando los flujos de trabajo, contratos y políticas se vuelven aún más importantes. La organización del servicio y los procesos empresariales abstractos son necesarios y por lo tanto las tecnologías relevantes como WS-BPEL se vuelven una parte importante del conjunto de herramientas de los arquitectos.

Del mismo modo que ningún dispositivo único es una isla inaccesible dentro de un dominio alternativo, tampoco existen empresas y organizaciones aisladas.

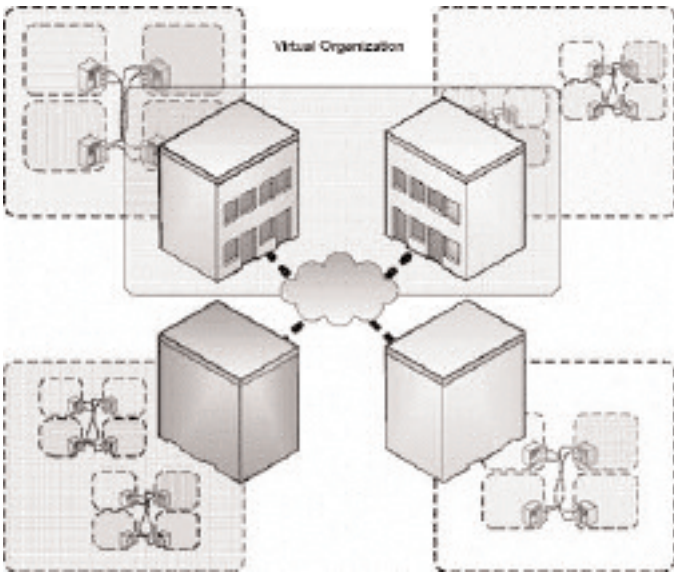
Tal como es el caso de nuestro mundo físico, las empresas hacen negocios con otras personas, las organizaciones interactúan y las instituciones gubernamentales colaboran. Los servicios y las interacciones son fundamentales para nuestras actividades cotidianas (el servicio bancario, el servicio de correo postal, un servicio de una agencia de viajes, por ejemplo). Es simplemente natural que cuando modelamos estas actividades en un mundo computarizado, seguimos un modelo de arquitectura similar al que se adopta en el mundo real.

Como cabría esperar, cuando se trata de aplicaciones de escala muy grande, donde el objetivo principal es proporcionar alto rendimiento, la naturaleza de la aplicación puede conducirnos a diferentes diseños que tienen en cuenta diferentes estrategias si la comparamos con una situación dentro de la empresa. Tal como es el caso del mundo real, los acuerdos a nivel contractual y de servicios existen para gobernar las interacciones entre las empresas. Las organizaciones virtuales pueden establecerse -de la misma forma que se crean alianzas entre las empresas en el mundo físico- para cumplir con las necesidades de alto rendimiento de las aplicaciones de las entidades participantes. De hecho, una aplicación de alto nivel distribuida puede reflejar una alianza entre empresas del mundo real tal como se ejemplifica en la Figura 5 (una serie de institutos de investigación que aúnan sus esfuerzos para resolver un gran problema científico, por ejemplo).

Para que las organizaciones virtuales sean posibles, se deben tratar asuntos como representaciones digitales de acuerdos, negociación de contratos, aceptación de interacciones, federación de credenciales de usuarios, políticas y provisión de la calidad de servicio acordada. Las descripciones basadas en el flujo de trabajo de alto rendimiento pueden existir para representar el comportamiento de las organizaciones virtuales o para coordinar las interacciones de negocio de empresas cruzadas. La aplicaciones en este entorno, cuando se diseñan e implementan de forma adecuada, pueden ser sin duda a escala Internet.

Mientras que los tipos de servicios que encontramos dentro de un empresa, como cola de trabajo, programación, comercialización de recursos, acceso e integración de datos, visualización, etc., pueden aún ser necesarios cuando nos movemos a escala Internet, se debe tener cuidado en la manera en que se utilizan e implementan estos servicios. Deben evitarse las soluciones centralizadas (un servicio de almacenamiento de datos único, por ejemplo) o comportamientos con un alto grado de acoplamiento (transacciones perdurables entre las organizaciones o exposición directa de estado, por ejemplo). Por supuesto que uno puede opinar que Google y Amazon son ejemplos espectaculares de depósitos digitales centralizados. Esto es cierto, pero también es cierto que son depósitos que ya utilizaban soluciones escalables para sus implementaciones; y ya habían sido construidos en función de centros de datos de sincronización escasa, reproducidos y distribuidos. Google y Amazon pueden ser vistos como buenos ejemplos de un servicio de acceso a datos virtualizado que han sido diseñados e implementados teniendo en cuenta la escalabilidad y el rendimiento.

Figura 5. Ilustración de la forma en que las empresas pueden colaborar



Con las aplicaciones a escala Internet, las alianzas y colaboraciones entre organizaciones se forman utilizando contratos digitales. Estos contratos representan un conjunto de acuerdos a nivel servicios que deben llevarse a cabo para que los trabajos informáticos viajen desde una organización hacia otra, para que las fuentes de datos puedas ser visibles, para que la funcionalidad de determinados equipos esté disponible para los miembros, para el nivel de confianza sobre las funciones de los usuarios acordadas, políticas de seguridad y requerimientos, etc. Las organizaciones virtuales necesitan contratos para administrar sus operaciones al igual que sucede con cualquier colaboración entre empresas en el mundo real. Los contratos se leen, validan y ejecutan por medio de middleware de soporte especializado.

Además, en un mundo digital en el que cada tipo de recurso es accesible, los requerimientos de la aplicación y ofrecimientos de servicios se describen utilizando lenguajes declarativos. El middleware de soporte es responsable de combinar dinámicamente los requerimientos de una aplicación con el ofrecimiento de un servicio. Si es necesario, también deben implementarse acuerdos a nivel servicio y negociación dinámica de pagos.

Por ejemplo, una aplicación puede promocionar que necesita un servicio que ofrezca recursos informáticos con requerimientos de software y hardware específicos, un servicio de almacenamiento de datos de un tamaño determinado, un motor de visualización con un tiempo de respuesta específico y un servicio de resolución de ecuación que posea un tiempo en actividad garantizado. Estos requerimientos se expresan utilizando un vocabulario XML. El documento resultante se envía a un registro (distribuido) y se descubre un conjunto de servicios disponibles. El middleware subyacente negocia el acuerdo a nivel servicio y detalles del pago con los servicios resultantes conforme a los requerimientos de la aplicación y dentro de una serie de límites que establece el usuario. Una vez que se ha establecido el acuerdo, que puede utilizarse en controversias futuras, lo firman todas las partes implicadas.

El SETI@home, ClimatePrediction.net, y otros proyectos similares han demostrado que a través de las redes comunitarias es posible reunir recursos para resolver grandes problemas. Si se ignoran los entornos controvertidos respecto del uso de tecnologías P2P para compartir archivos, la promesa de la informática distribuida de colaborar con los problemas empresariales y científicos es una buena elección para las posibilidades de las tecnologías P2P o redes comunitarias. Las aplicaciones Web HPC futuras deben considerar las tecnologías P2P como las tecnologías que permiten compartir y transferir archivos, descubrir recursos, distribución de informática, etc. Por ejemplo, podemos imaginar una red P2P que permite enviar trabajos y que los recursos adecuados para la ejecución sean descubiertos automáticamente.

Las redes de P2P pueden implementarse utilizando las mismas

tecnologías del Servicio Web para utilizar la enorme inversión que se realice en los protocolos de calidad de servicio fundamentales.

A pesar de que el concepto de informática distribuida surgió de la comunidad de supercomputadoras, los negocios también se dan cuenta de su valor comercial. Por cada acceso a recursos basado en un pago o suscripción (en especial recursos informáticos de alto rendimiento) están comenzando a surgir como un modelo empresarial viable de grandes compañías que ya han implementado los servicios y tecnologías que se ofrecen. Por supuesto, la integración de estas implementaciones en aplicaciones debe ser general y transparente en su totalidad para los usuarios finales y que de esta manera, la visión de "informática de servicios en red" o "informática como un servicio" se hagan realidad.

Esto produce un sin fin de oportunidades valiosas. Obviamente, existen aquellas compañías que podrán obtener los beneficios de alojar recursos informáticos que poseen una buena relación costo-beneficio para que otras personas puedan integrarse en sus entornos sobre una base ad hoc. Lo recíproco de esta situación es que existirán oportunidades para que las empresas programen sus gastos sobre una infraestructura IT de la manera más eficaz y decidan si las inversiones de capital inicial pueden ser suplantadas por el modelo de pago por tiempo de uso (o no), además de la agilidad empresarial general que se obtendrá si se cambia a una arquitectura orientada al servicio.

El conjunto de funcionalidades/servicios típicos que se presentó anteriormente (Tabla 1) también son necesarios en las aplicaciones HPC a escala Internet. No obstante, son más abstractos y deben realizar diferentes suposiciones respecto del entorno en el que son implementados. Por ejemplo, el problema de a quién se le permite enviar trabajos para ejecutar en un centro de computación de la organización será definido a través de contratos digitales, mientras que la calidad de servicio que recibirá cada interacción (asignación de almacenamiento de datos y CPU, por ejemplo) será controlado por acuerdos a nivel servicio en el mismo contrato. Sin embargo, además de la Tabla 1, también observamos el conjunto de comportamientos/servicios típicos para HPC a escala Internet que se muestra en la Tabla 2.

Ya que hemos analizado a un muy alto y abstracto nivel la arquitectura de las aplicaciones distribuidas, de alto rendimiento, orientadas al servicio que pueden escalar por Internet, ahora mencionaremos algunas consideraciones importantes sobre el diseño y la implementación.

A pesar de las constantes mejoras en aumento de la banda ancha y el tiempo de espera en la red, la comunicación sobre infraestructuras de red comerciales es muchísimo menos eficaz que sobre interconexiones especializadas o arquitecturas de bus de memoria. Por consiguiente, se debe tener cuidado al crear, diseñar y construir aplicaciones distribuidas HPC, para minimizar el costo asociado con el intercambio de mensajes entre los componentes de una aplicación.

Tabla 1. Servicios característicos/funcionalidades/características que pueden soportar los dispositivos

Caracteristica	Descripción
Seguridad	Todos los aspectos de la seguridad (por ejemplo autenticación, autorización, auditoría, confidencialidad, privacidad, etc.) se tratan al utilizar protocolos redactables e interoperables como WX-Security, Liberty-Alliance, SAML, XACML, etc. La administración de la identidad y las soluciones federadas también necesitan estar en su lugar (por ejemplo Directorio Activo).
Gestión del trabajo	Aquellos recursos en la red que son capaces de alojar trabajos para ejecución, ofrecen un servicio de administración del trabajo.
Programación	Se reúne la información de utilización del recurso y luego se utilice en el proceso de decisión de distribución de trabajos para los recursos disponibles.
Acceso a datos	Aquellos recursos en la red que proporcionan acceso a almacenamientos de datos (por ejemplo sistemas de bases de datos relacional, sistemas de archivos, redes de áreas de almacenamiento) necesitan exponer servicios apropiados.
Registros de recursos y seguimiento	Tanto P2P como las soluciones centralizadas necesitan ser implementadas para permitir el descubrimiento y seguimiento de los dispositivos en la red y su estado.
Administración de dispositivos	Los dispositivos que se encuentran en la red pueden tener que ser administrados como un conjunto o de manera individual (como por ejemplo Directorio Activo, WS-Managmente).

Principios de SOA

Además de los gastos de la red, la comunidad HPC está interesada en los gastos informáticos en los que se incurre debido al procesamiento de XML. Sin embargo, este es un aspecto que está tratando la comunidad SOAP. De hecho, las implementaciones SOAP correctas ya tratan el rendimiento de mecanismos binarios para mensajes cortos [6], lo que implica que con el tiempo, el factor limitador para la transmisión de mensajes ya sea en formato SOAP o binario será la banda ancha y el tiempo de espera en la red. Si bien no deseamos desacreditar la importancia del bajo tiempo de espera y la alta capacidad de procesamiento para las aplicaciones HPC, es claro que el nivel conservador que SOAP

Figura 2. Prestaciones/funcionalidades/servicios característicos para la informática HPC a escala Internet

Caracteristica	Mecanismo
Comercialización	Los servicios que actúan como intermediarios para otros servicios serán implementados para permitir un descubrimiento dinámico de los recursos o recursos agregados para de esta manera poder proporcionar una mejor calidad.
Pago	Será necesaria una infraestructura común para los pagos, similar a la que se utiliza para pagos con tarjeta de crédito en el mundo real .
Servicios de lógica de almacenamiento/ informáticos	Estarán disponibles los servicios que ofrecen acceso a recursos de almacenamiento e informáticos, aún si estos recursos no pertenecen a una única entidad, de la misma forma que existen compañías que ofrecen servicios de electricidad en el mundo real.
Registros de recursos y servicios de dominio específico y global	Serán necesarios directorios globales como Google para la ubicación de recursos y otros servicios en la Red. Los dominios de aplicaciones pueden utilizar sus propios registros especializados como una forma de agregar valor para sus usuarios de dominio (por ejemplo, un registro de servicios relacionados con la bioinformática.
Transferencia de datos	Cuando se necesitan transferir grandes bases de datos a través la Web, es necesario utilizar tecnologías de transferencia de alto rendimiento, especializadas. La negociación del tipo de tecnología de transferencia que se utilizará deberá realizarse sobre protocolos estándares. También se pueden utilizar tecnologías P2P.
Contratos y políticas	Los vocabularios y el software middleware utilizados para crear, negociar, ejecutar y controlar los contratos y políticas serán fundamentales en un entorno en el que se formen organizaciones virtuales dinámicas.
Organización	En la medida que los servicios están disponibles en Internet, es necesario el uso de tecnologías para organizarlos y combinarlos en formas específicas de aplicación.
Tecnologías relacionadas con la semá	En un entorno en el que están disponibles una gran cantidad de recursos y servicios, es extremadamente importante razonar de un modo universal respecto de la información disponible.

ha atraído es en algún punto inmerecido.

El acoplamiento escaso y la escalabilidad son los resultados de un diseño con principios y una arquitectura de software sensata. Adoptamos los siguientes principios para construir aplicaciones orientadas al servicio:

- La colección de protocolos que admite un servicio determinan la semántica de su comportamiento.
- Los Servicios se vinculan al mensaje y a la información que ellos llevan y no para estados y términos en particular.
- Los mensajes que se intercambian entre servicios son auto-descriptivos (es decir, como en REST5) en la medida que estos transporten la información suficiente como para permitir al destinatario que establezca un contexto de procesamiento así como también la información necesaria para ejecutar la acción deseada.
- Los servicios se implementan y evolucionan independientemente uno del otro.
- La integración de los servicios se lleva a cabo a través de acuerdos basados en contratos.

Además de los principios expuestos, también fomentamos la siguiente lista de pautas que se debe tener en cuenta al construir sistemas orientados al servicio:

Sin Estado. Esta propiedad se refiere al principio de naturaleza auto-descriptiva ya mencionado. Los servicios deberán apuntar a un intercambio de mensajes que demuestren toda la información necesaria para recibir servicios para re-establecer el contexto de una interacción en una conversación multi-mensaje. Los servicios sin estado son fáciles de escalar y simplifican mucho la tolerancia de fallas mediante redundancia.

Mensajes Enriquecidos. Los costos de la comunicación generalmente son elevados. Por lo tanto, debemos apuntar a los protocolos que impliquen mensajes enriquecidos que den como resultado interacciones detalladas, disminuyendo de forma eficaz el número de veces que un servicio tiene que cruzar toda la red.

Gestión de Estados. En cuanto al tradicional diseño de aplicación en N, la implementación de servicio deberá delegar todos los aspectos de la gestión de estado en almacenes de datos especializados y exclusivos (según lo mostrado en la Figura 1)

Envío de Mensajes. No deberá haber presunciones sobre los mecanismos de envío empleados por los servicios. Como resultado, ninguna información específica de envío se deberá filtrar de las implementaciones de servicio, cruzar los límites y mostrarse a través de contenidos del mensaje (por ejemplo, SOAP-RPC, SOAP estilo RPC, WSDL de estilo Document-Wrapped, o nombres de métodos mostrados como atributos soap:action o wsa:action)

Acoplamiento de Roles. Los arquitectos deberán tener siempre en mente que en una arquitectura orientada al servicio no existen "consumidor", "proveedor", "cliente", "servidor" y demás. Estos son roles que existen a nivel de aplicación y no en los bloques de construcción de la arquitectura, los servicios. En SOAs, solo existen los servicios que intercambian mensajes. Considerar a un par de servicios como cliente/servidor presenta una forma de acoplamiento que podría en definitiva ser difícil de romper.

Los servicios son una abstracción sensata para encapsular y administrar el creciente nivel de complejidad de las aplicaciones distribuidas. Lo bueno de la orientación al servicio es que las pautas y principios de la arquitectura son sistemáticos en su totalidad desde un proceso de sistema operativo hasta un servicio que encapsula un proceso empresarial completo o aún toda una organización. Los requerimientos de la arquitectura de alto rendimiento a escala Internet o informática "distribuida" no son diferentes a los de los negocios empresariales focalizados en la informática, y por lo tanto se deben utilizar pautas y principios idénticos.

Notas al pie

1. http://boinc.berkeley.edu
2. http://ssdl.org
3. http://www.cs.wisc.edu/condor/
4. http://www.cs.wisc.edu/condor/birdbath/
5. http://mercury.it.swin.edu.au/ctg/AWSA04/Papers/ng.pdf
6. http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Lecturas relacionadas

1. Open Grid Services Architecture
2. Globus Toolkit
3. OGSA Data Access and Integration
4. Web Services Grid Application Framework
5. Semantic Web
6. W3C Web Services
7. OASIS Web Services

Agradecimientos

Los autores desean agradeces al Prof. Paul Watson (Escuela de Ciencias de la Computación, Universidad de Newcastle) por sus comentarios de gran utilidad durante la preparación de este artículo.

Sobre los autores

Savas Parastatidis
Escuela de Ciencias de la Computación
Universidad de Newcastle
Jim Webber
ThoughtWorks Australia

Microsoft

THE ARCHITECTURE JOURNAL