Curso de Posgrado: Computación Orientada a Servicios

DATOS GENERALES	
Responsable a cargo de la actividad curricular	**Dr. Martin Garriga**
Área temática[1]	Ingeniería de Software

RESUMEN DE CONTENIDOS

El paradigma de Computación Orientada a Servicios (SOC), promueve el desarrollo rápido y de bajo costo de aplicaciones de distribución masiva, interoperables y capaces de evolucionar, que permite crear procesos de negocio dinámicos que interconectan organizaciones y plataformas informáticas heterogéneas. SOC permite efectivizar el crecimiento dinámico de portfolios de servicios organizacionales, y la combinación de estas soluciones con componentes externos que posiblemente residen en forma remota (Papazoglou et al., 2007). Una Arquitectura Orientada a Servicios (SOA) representa un modelo arquitectónico cuyo propósito es mejorar la agilidad y la rentabilidad de una empresa, y reducir toda la carga de materialización de las TICs en una organización (Erl et al., 2008). Por lo tanto, SOA se considera un marco general adecuado como solución tecnológica a la interacción de servicios, donde se identifican los siguientes actores: un cliente, un proveedor, y un agente de servicios (broker) (Zimmerman et al., 2005).

Una Aplicación Orientada a Servicios puede verse desde una perspectiva arquitectónica como una aplicación basada en componentes que es construida mediante el ensamblaje de dos tipos de componentes: internos (localmente embebidos en la aplicación), y externos (estática o dinámicamente ligados a un servicio) (Crasso et al., 2010). Tales componentes pueden ajustarse a la definición de un modelo off-the-shelf (OTS) de componentes de software (por ejemplo, Java Beans) de acuerdo con el paradigma de Desarrollo de Software basado en Componentes (CBSD) (Kung-Kiu et al., 2007), pero también pueden asumir la forma de piezas comunes de software en una arquitectura.

El paradigma SOC ha sido adoptado en la industria principalmente a través de la tecnología de Servicios Web, que está definido por la W3C como “un sistema software (identificado por un URI), diseñado para soportar la interacción máquina-a-máquina sobre una red interoperable. Tiene una interfaz descripta en un formato procesable por máquina (específicamente WSDL), y otros sistemas interactúan con el Servicio Web de la manera que prescribe su descripción, utilizando (por lo general) mensajes SOAP transmitidos mediante HTTP con una serialización XML, junto con otros estándares de la Web”.
De esta manera, en este curso se analizará la evolución de las arquitecturas de objetos y componentes hasta alcanzar el paradigma de servicios. Luego, se brindarán las nociones y herramientas tecnológicas para desarrollar Servicios Web Java, como artefactos experimentales. Se podrán utilizar componentes y programas existentes, y ajustarlos (*servificarlos*) para ajustarlos al formato de Servicios Web.

CONOCIMIENTOS PREVIOS REQUERIDOS

Se requieren sólidos conocimientos en metodologías de desarrollo de software orientada a objetos. Lenguaje de modelado UML. Manejo de IDE Eclipse o similar. Es deseable conocimientos de Arquitecturas Web (cliente-servidor).

OBJETIVOS

El objetivo general del curso es conocer y profundizar la base metodológica para el desarrollo de aplicaciones orientadas a servicios, utilizando las últimas tendencias tecnológicas y conceptuales de Servicios Web. Además, se espera que los estudiantes incorporen nociones teórico-prácticas, desarrollando el ciclo de vida de al menos un servicio durante el curso (crear, publicar y consumir un servicio).
Los objetivos específicos son:

- Entender aspectos básicos de la Computación Orientada a Servicios y los Servicios Web.
- Adquirir las herramientas arquitectónicas para comprender y desarrollar Arquitecturas Orientadas a Servicios.
  Aplicar los conocimientos teóricos para desarrollar Servicios Web basados en Java utilizando el IDE Eclipse.

---

1 Corresponde a uno de los siguientes tópicos: Algoritmos y Lenguajes; Teoría de la Computación; Ingeniería de Software, Bases de Datos y Sistemas de Información; Arquitecturas, Sistemas Operativos y Redes.

CONTENIDOS

Unidad I: INTRODUCCIÓN: DESARROLLO DE SOFTWARE BASADO EN REUSO

Desarrollo de Software Tradicional vs. Desarrollo de Software basado en reuso. Historia y evolución. Paradigmas. Desarrollo de software orientado a servicios, a componentes y a dominios.

Unidad II: COMPUTACIÓN ORIENTADA A SERVICIOS

Concepto de Computación Orientada a Servicios. Historia y evolución. Ciclo de vida. Aspectos principales. Arquitecturas orientadas a servicios. Cliente, Proveedor y Registro de servicio.

Unidad III: SERVICIOS WEB

Características de los Servicios Web. Tecnologías de Implementación de Servicios Web. Especificaciones WSDL. Protocolo SOAP y sus implicaciones. Servicios SOAP vs. Servicios RESTful.

Unidad IV: DESARROLLO DE APLICACIONES ORIENTADAS A SERVICIOS

Desarrollo de Servicios Web en Java. Eclipse IDE. Servificación usando AXIS2 y servidor Apache Tomcat. Prueba de Servicios utilizando SoapUI. Desarrollo del cliente java usando AXIS2.

MODALIDAD DE EVALUACIÓN Y CONDICIONES DE ACREDITACIÓN[2]

Certificación de Postgrado: 80% de asistencia, realización de los trabajos prácticos propuestos en clase y elaboración de un trabajo final que involucra tareas de investigación y desarrollo, que deberá ser realizado una vez terminado el curso (cuyo tema y fecha de entrega se acordará con los participantes). La nota de los trabajos obligatorios y del trabajo final debe ser no menos a 7 (siete) puntos.

BIBLIOGRAFÍA DE LECTURA OBLIGATORIA CORRESPONDIENTE A CADA UNIDAD Y GENERAL

1. Erl, T., Kamarkar, A., Walmsley, P., Haas, H., Yalcinalp, U., Liu, C., Orchard, D., Tost, A., Pasley, J. (2008) Web Service Contract Design & Versioning for SOA, vol. 1. First edn., Prentice Hall.
2. Jarzabek, S. (2007). Effective Software Maintenance and Evolution: A Reuse-Based Approach. Auerbach Publications. Taylor & Francis Group.
3. Kung-Kiu, L.; Zheng, W. (2007). Software Component Models. IEEE Transactions on Software Engineering, 33(10):709-724.
4. Nagappan, R.; Skoczylas, R.; Sriganesh, R. (2003). Developing Java™ Web Services: Architecting and Developing Secure Web Services Using Java. Wiley Publishing Inc.
5. Papazoglou, M.; Traverso, P.; Dustdar, S.; Leymann, F. (2007). Service-Oriented Computing: State of the Art and Research Challenges. IEEE Computer, 40(11):38-45.
6. Weerawarana, S.; Curbera, F.; Leymann, F.; Storey, T.; Ferguson, D. (2005). Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More. Prentice Hall PTR.
7. Zimmerman, O.; Tomlinson, M.; Peuser, S. (2005) Perspectives on Web Services – Applying SOAP, WSDL and UDDI to Real-World Projects. Springer-Verlag.

Bibliografia de consulta:

1. Crasso, M.; Mateos, C., Zunino, A.; Campo, M. (2010). Empirically assessing the impact of dependency injection on the development of Web Service applications. Journal of Web Engineering, 9(1):66–94.
2. Wirfs-Brock, R.; Wilkerson, B.; Wiener, L. Designing Object-Oriented Software. Prentice Hall, 1990.
3. Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D. (2004). Web Services Architecture. W3C Working Group. February. http://www.w3.org/TR/ws-arch/
4. Eclipse Home Page (2014). The Eclipse Foundation. Eclipse.org. http://www.eclipse.org/

2 Son condiciones mínimas para la aprobación de todos los cursos: cumplir con un mínimo del 80% de asistencia a las clases, realizar las tareas y aprobar las evaluaciones que se hayan propuesto en el programa, con una calificación no menor a 7 (puntos). Los trabajos de evaluación pautados y la calificación de los alumnos deberán realizarse dentro de los 60 días posteriores a la finalización del curso.