

Automatización del proceso de despliegue de servicios en la nube

Miguel Zuñiga-Prieto¹, Lizandro Solano-Quinde², Emilio Insfran³, Yessenia Cabrera¹

¹ Departamento de Ciencias de la Computación, Universidad de Cuenca, Av. 12 de Abril y Agustín Cueva, Cuenca, Ecuador, 01.01.168.

² Departamento de Ingeniería Eléctrica, Electrónica y Telecomunicaciones, Universidad de Cuenca, Av. 12 de Abril y Agustín Cueva, Cuenca, Ecuador, 01.01.168.

³ Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de Valencia, Camino de Vera s/n, Valencia, España, 46022.

Autores para correspondencia: miguel.zunigap@ucuenca.edu.ec, einsfran@dsic.upv.es

Fecha de recepción: 19 de junio del 2016 - Fecha de aceptación: 24 de julio del 2016

ABSTRACT

The lack of standardization of the access methods and the management of resources in cloud environments hinders both, the portability of services between different cloud environments, and the agile deployment of services. In this context, it is important to reduce the deployment time in order to reduce the economic cost to organizations. The manuscript presents an approach for automating the deployment process of cloud services, by proposing a language that specifies the resource needs in cloud environments, in support to the logic of cloud services. The information about requirements is used for generating resource provisioning scripts and service configurations, both, specific to the deployment platform. This approach is applied to a case study in the Windows Azure© platform, used as a deployment environment and the development of a deployment script based on the cloud resource specification.

Keywords: Cloud computing, model driven software development, incremental deployment.

RESUMEN

La falta de estandarización en los métodos de acceso y gestión de recursos de entornos en la nube dificulta tanto la portabilidad de servicios entre diferentes entornos en la nube, como la entrega ágil de servicios, siendo importante reducir los tiempos de entrega para reducir los costos económicos a las organizaciones. Este artículo presenta un enfoque para la automatización del proceso de despliegue de servicios en la nube, el cual propone un lenguaje para especificar los requerimientos de recursos de entornos en la nube necesarios para soportar la lógica de servicios a ser desplegados en la nube. La información de requerimientos es utilizada para generar scripts de aprovisionamiento de recursos y configuración de servicios específicos de la plataforma de despliegue. Este enfoque se ilustra con un caso de estudio práctico en la plataforma Windows Azure© utilizada como entorno de despliegue, para lo cual en base a la especificación de los requerimientos de recursos de la nube necesarios genera el script de despliegue correspondiente.

Palabras clave: Computación en la nube, desarrollo dirigido por modelos, despliegue incremental.

1. INTRODUCCIÓN

La evolución tecnológica en el campo de la informática, el incremento en la demanda de soluciones especializadas, personalizadas y con mejoras continuas por parte de usuarios cada vez más informados, impone nuevos retos a los profesionales de la ingeniería de software, de tal manera que en la actualidad el uso de la computación en la nube es una de las tecnologías más prometedoras para el

desarrollo de software (Di Martino *et al.*, 2015).

La computación en la nube es un paradigma que permite acceso ubicuo, conveniente y a petición, a un conjunto configurable de recursos computacionales que pueden ser rápidamente aprovisionados; es decir provisto, acondicionado y habilitado, con un mínimo de esfuerzo de administración o interacción con el proveedor de recursos en la nube (proveedor de nube) (Mell & Grance, 2011). Las categorías de servicio de la computación en la nube proveen elasticidad en la adquisición de recursos tanto de infraestructura (Infraestructura como servicio - IaaS), como de plataforma (Plataforma como Servicio - PaaS) dependiendo de las necesidades computacionales de los servicios de software (servicios) a ser desplegados en la nube (servicios en la nube) y que hacen uso de esos recursos.

Este paradigma conjuntamente con los métodos de desarrollo ágil, los cuales soportan un proceso incremental de desarrollo de servicios en la nube, permiten afrontar el dinamismo en los requerimientos organizaciones (Swaminathan & Jain, 2012).

Los proveedores de nube imponen sus propios estándares tanto para el desarrollo de servicios en la nube como para el uso de sus recursos, lo que limita la portabilidad entre proveedores de nube, problema conocido también como vendor lock-in. (Chandrasekaran, 2014). En este contexto, existen propuestas que hacen frente al problema de portabilidad, tal es el caso del API unificada que integra las APIs de varios proveedores de nube (Petcu *et al.*, 2011), software comercial para acceso multiplataforma y enfoques de modelamiento de servicios en la nube mediante TOSCA y UML (Binz *et al.*, 2014) sin embargo crean dependencia del proceso de despliegue con las APIs propuestas.

Para soportar los retos antes indicados, este trabajo propone un enfoque que aplica los principios de Desarrollo de Software Dirigido por Modelos (DSDM) para la automatización de las actividades de despliegue de servicios en la nube, el cual agiliza la obtención de scripts de aprovisionamiento de infraestructura, despliegue y configuración de los servicios desplegados en la nube. El equipo de desarrollo/operaciones define el modelo de requerimientos de infraestructura de los servicios a desplegar en la nube, elige el proveedor de nube, y a través de transformaciones de modelo a texto (M2T) se obtienen los scripts específicos del proveedor de nube.

2. ESTADO DEL ARTE

Para minimizar el problema de vendor lock-in, se han realizado varios esfuerzos e investigaciones, tal es el caso de la investigación desarrollada para unificar las distintas APIs proporcionadas por los proveedores de nube (Petcu *et al.*, 2011). Dicho trabajo propone abstraer el despliegue de servicios en la nube del tipo de tecnología utilizada para acceder a cada plataforma mediante la integración de las distintas APIs proporcionadas por los proveedores de nube, en una API unificada. En el ámbito de Software Comercial, existen varias propuestas como CHEF (Chef Software, 2015) o DeltaCloud4 (Meireles & Malheiro, 2014) que consiguen abstraer el despliegue de los servicios en la nube del proveedor de nube proporcionando una plataforma de integración común a los distintos proveedores de nube. Sin embargo, estas propuestas generan una nueva dependencia tanto a su tecnología como a la compatibilidad con nuevos proveedores de nube.

TOSCA (Binz *et al.*, 2014) proporciona portabilidad en la descripción y gestión de la aplicación mediante un lenguaje estándar para modelar aplicaciones en la nube. Sin embargo, no modela el despliegue de los artefactos de manera independiente del proveedor de nube utilizado.

Un enfoque más completo proporciona portabilidad en el despliegue de aplicaciones en la nube mediante su modelamiento en UML utilizando librerías, perfiles y plantillas (Bergmayr *et al.*, 2014) a través de los cuales representa la aplicación como componentes que generan artefactos cuyo despliegue es modelado mediante plantillas, estas corresponden a la oferta de entornos de servicios en la nube proporcionada por los distintos proveedores de nube como por ejemplo el AWSM Medium de Amazon. Este enfoque proporciona portabilidad en el despliegue de las aplicaciones, sin embargo, no considera la transformación de modelos en scripts de aprovisionamiento de infraestructura y scripts para la configuración y el despliegue de servicios en la nube.

En la siguiente sección se presenta el enfoque que debe seguir el equipo de desarrollo para

desplegar servicios en la nube. El aporte de este artículo es la definición de un lenguaje de especificación de requerimientos de recursos en la nube el cual aplica el enfoque DSDM y facilita la obtención de scripts para: i) el aprovisionamiento de los recursos en la nube necesarios para el despliegue y soporte de la lógica los servicios a desplegar, ii) el despliegue servicios en la nube.

3. ENFOQUE DE AUTOMATIZACIÓN

El enfoque de automatización propuesto en este trabajo se basa en un lenguaje de especificación de requerimientos, el cual es empleado como base para automatizar el proceso de despliegue. Este enfoque propone dos actividades; en la primera el lenguaje es utilizado para describir los recursos en la nube que soportaran los servicios a desplegar, mientras que en la segunda el modelo creado al aplicar el lenguaje propuesto es utilizado como insumo para transformaciones M2T que generan scripts de aprovisionamiento y despliegue.

3.1. Especificar requerimientos de recursos en la nube

En esta actividad una vez que los equipos de desarrollo han implementado los servicios de la nube los han liberado en forma de paquetes, el equipo de operación prepara éstos artefactos para su despliegue en el entorno de producción (o entorno de pruebas u otro), mismo que será aprovisionado en el entorno de un proveedor de nube específico. Para esto, el equipo de operaciones especifica los recursos en la nube necesarios para satisfacer los requerimientos de los servicios incluidos en un paquete; documentando estos requerimientos en el *Modelo de Recursos en la Nube*. En el enfoque DSDM un meta-modelo define la sintaxis y semántica de un lenguaje de modelado; este trabajo propone un lenguaje para la creación de Modelos de Recursos en la Nube el cual es descrito por el meta-modelo de la Figura 1. Este meta-modelo, que por limitaciones de espacio incluye únicamente las meta-classes y atributos más relevantes, ha sido definido en base a patrones de diseño (Fehling *et al.*, 2014; Homer *et al.*, 2014), especificaciones y guías (Metsch *et al.*, 2010; Council, 2015), y el análisis de las ofertas de entornos en la nube como Microsoft Azure, Amazon's Elastic Cloud (EC2) y Google App Engine.

Con el propósito de hacer frente al problema del vendor lock-in, este lenguaje permite especificar de manera independiente de la tecnología del proveedor de nube los recursos a ser aprovisionados en diferentes entornos. Para ello, como se puede apreciar en la Figura 1 el paquete a ser desplegado (*DeploymentArtifact*), ubicado en *packagesPath*, puede ser hospedado en varios entornos (*CloudEnvironment*) provistos por diferentes proveedores de nube (*cloudProvider*). Adicionalmente, los servicios incluidos en el paquete hacen uso de los recursos de la nube (*Resource*) aprovisionados en un *CloudEnvironment*. Los recursos pueden ser ya sea de infraestructura (*Infrastructure*) o de plataforma (*Platform*). Adicionalmente se pueden especificar las relaciones entre los diferentes *Resource* a provisionar, por ejemplo, un *PaaS_SO* se hospeda en un *IaaS_Computing*, o un *PaaS_Runtime* como java se instala en un *PaaS_SO* que a su vez esta hospedado en un *IaaS_Computing*, o un *PaaS_DBManager* se instala en un *PaaS_SO*. Los atributos *isDynamic* (ej., *isDynamicMemory* en *IaaS_Computing*) permiten especificar características de *Resource* que cambian (ej., a través de modificar archivos de configuración del *CloudEnvironment*) sin interrumpir la ejecución del servicio que lo utiliza una vez que la instancia del recurso haya sido inicializada y el servicio en la nube se encuentra en ejecución.

Finalmente, además de especificar los requerimientos de recursos en la nube este lenguaje permite especificar ajustes (*Settings*) en la información de configuración (*Configuration*) de los servicios incluidos en *DeploymentArtifact*; los ajustes afectan a variables (*key*) invocadas desde la lógica del servicio cuyo valor podría ser modificado en tiempo de ejecución (si es que *isDynamic* = true).

3.2. Generación de scripts de aprovisionamiento y configuración

Una vez que se han especificado los requerimientos de recursos en la nube, el equipo de operaciones selecciona el o los entornos en los que desplegará los *DeploymentArtifact* y procede a refinar el *Modelo de Recursos en la Nube* de acuerdo a las características de los recursos ofrecidos por un

proveedor de nube específico y a la información necesaria para gestionar la suscripción (*Subscription*) a la plataforma. Como parte de la suscripción se especifican los parámetros (*Parameters*) específicos de cada proveedor de nube necesarios para hacer uso de sus mecanismos de acceso, creación y gestión de recursos (ej., scripts, APIs, etc.), así como las credenciales de acceso a los mismos (*Credentials*).

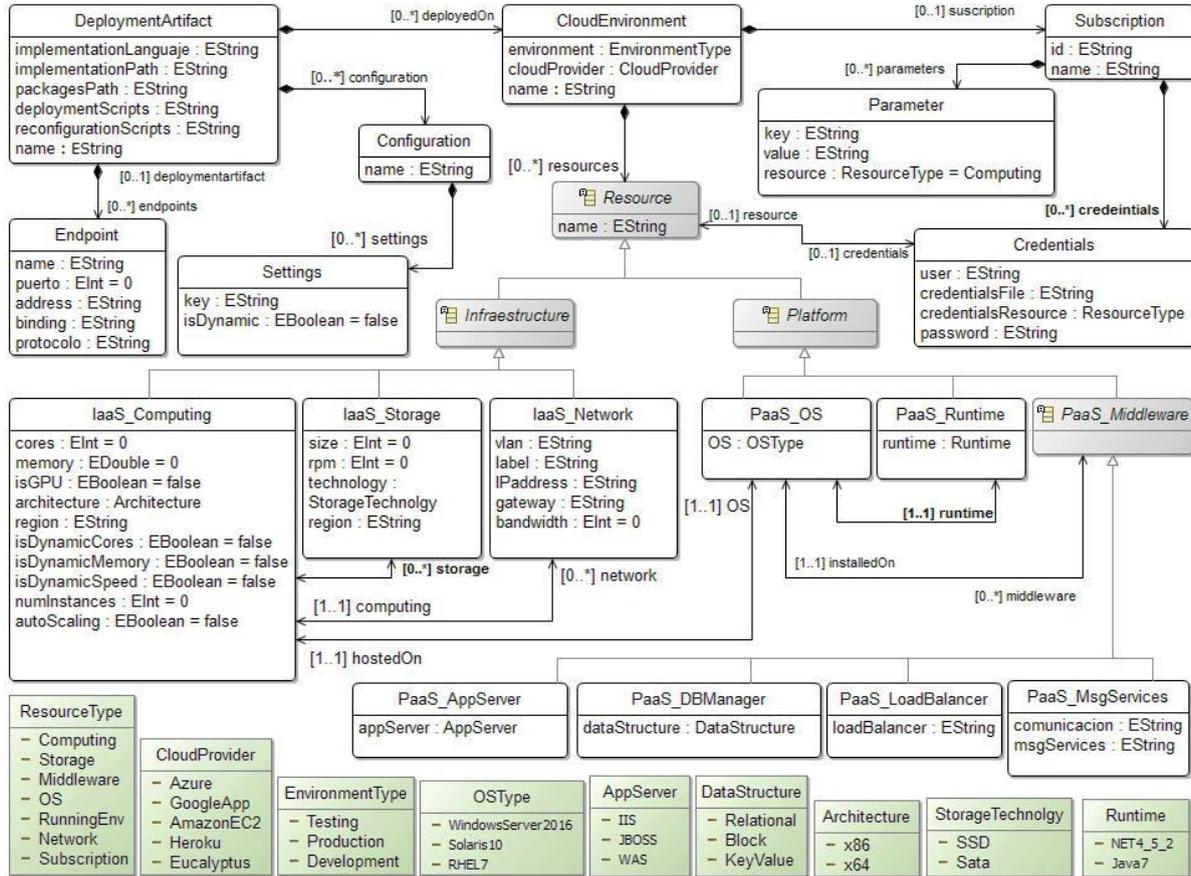


Figura 1. Extracto del meta-modelo de *Modelo de Recursos en la Nube*.

Finalmente, el equipo de operación ejecuta transformaciones de modelo a texto (M2T) que toman como entrada los requerimientos de recursos en la nube descritos en el *Modelo de Recursos en la Nube* y generan scripts (métodos que invocan las APIs específicas del proveedor de nube) para: i) el aprovisionamiento de los recursos en la nube, ii) el despliegue de los servicios incluidos en el *DeploymentArtifact*. Patrones de diseño para la nube sugieren colocar la información de configuración del entorno en la nube y de la lógica de los servicios a desplegar en un artefacto de software independiente del paquete que contiene la lógica del servicio, de tal manera que la información de configuración pueda desplegarse sin necesidad de detener la ejecución del servicio en la nube en caso de requerir cambios en la información de configuración (Fehling *et al.*, 2014; Homer *et al.*, 2014). Para satisfacer este patrón de diseño, también se generan scripts que crean artefactos de configuración o los actualizan dependiendo de los lineamientos de cada proveedor de nube. La ubicación de los scripts generados es determinada por los atributos *deploymentScripts* y *reconfigurationScripts* especificados en la meta-clase *DeploymentArtifact* (ver Figura 1).

4. APLICACIÓN DEL ENFOQUE DE AUTOMATIZACIÓN DEL DESPLIEGUE

Para ilustrar la aplicación de este enfoque nos planteamos un escenario de despliegue en el que el

servicio web *Dealer* ha sido implementado en Visual Studio 2013 y será desplegado en el proveedor de nube Windows Azure. Para éste caso de estudio se asume que el servicio ha sido implementado y empaquetado, y como entrada del proceso de automatización del despliegue se tienen los ficheros “Dealer.cspkg” y “ServiceConfiguration.Cloud.cscfg”.

El meta-modelo del *Modelo de Recursos en la Nube* fue implementado como un modelo *Ecore* en el entorno de modelado Eclipse Modeling Framework (EMF), el cual es diseñado como un componente de código abierto de Eclipse. Una vez implementado el meta-modelo, se lo utilizó para crear el *Modelo de Recursos en la Nube* del servicio *Dealer*, como se indica en la Figura 2. El modelo de la Figura 2 fue creado con el editor *Sample Reflective Ecore Model Editor*, el cual es parte de EMF y permite crear modelos que satisfacen la semántica descrita por un meta-modelo.

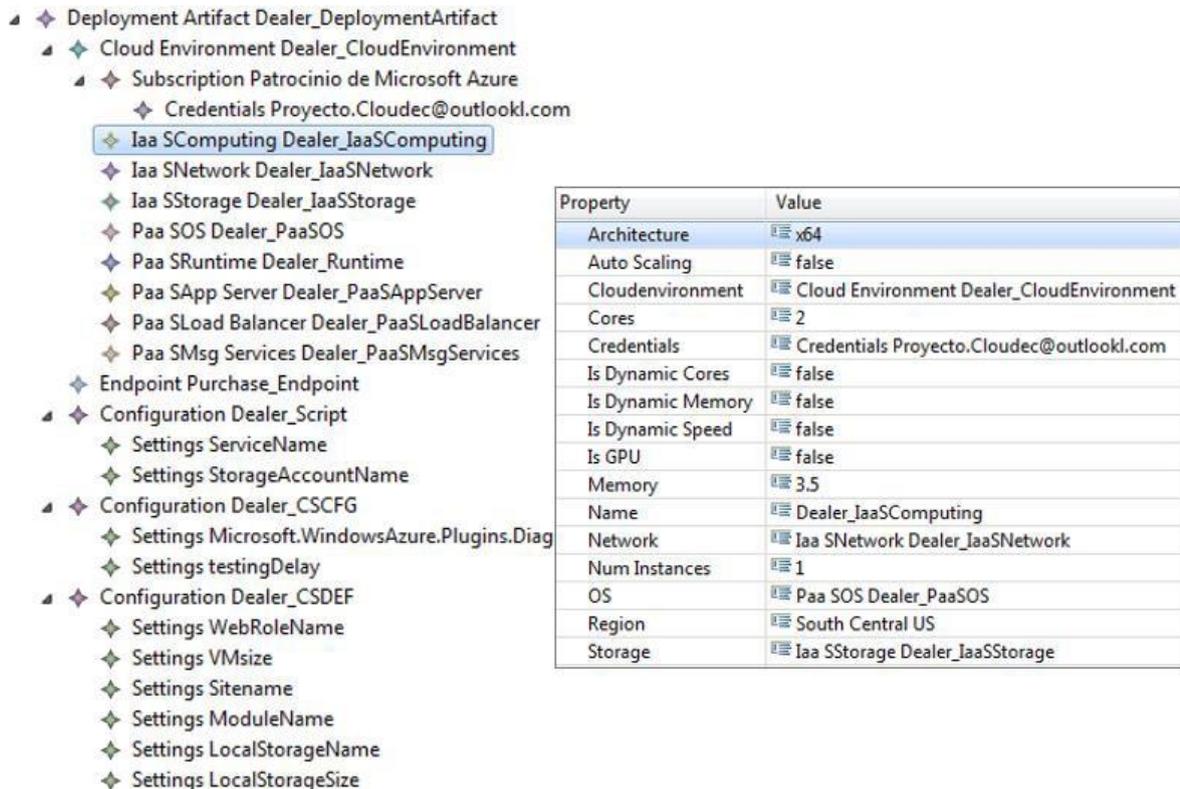


Figura 2. Modelo de Recursos en la Nube del servicio Dealer.

El entorno de nube seleccionado para el despliegue fue Microsoft Azure, el cual soporta las categorías de servicios de computación en la nube IaaS y PaaS, en donde la diferencia entre estas dos es que en la primera los desarrolladores gestionan tanto los recursos de plataforma como los servicios y datos, mientras que en la segunda los desarrolladores únicamente gestionan los aspectos relacionados los servicio y datos. Para este caso de estudio utilizamos Microsoft Azure como PaaS y nos enfocamos en generar scripts para el despliegue de servicios en la nube, sin considerar los scripts de aprovisionamiento de infraestructura y plataforma ya que estos son gestionados por el proveedor de nube. Sin embargo, los requerimientos a cerca de los recursos deben especificarse durante el despliegue de tal manera que el proveedor de nube pueda gestionarlos. Otro aspecto a considerar en el caso de Microsoft Azure (y otros proveedores de nube) es que en los scripts de despliegue o APIs no se detalla las características de los recursos, sino que se hace referencia a tipos de instancias de entornos en la nube predefinidos (ej., Tipos Instancia: Standard_A8 = 8 CPU cores, 56 GB memory, NICs 2, .NET Framework, IIS...). Sin embargo, a pesar de que en la actualidad el *Modelo de Recursos en la Nube* no soporta crear instancias predefinidas, el mapeo entre las características de los recursos especificadas en el *Modelo de Recursos en la Nube* y las instancias predefinidas en Windows Azure se realiza como parte de la transformación M2T. Para la obtención de scripts de despliegue definimos transformaciones M2T para lo cual utilizamos la extensión de Eclipse Aceleo M2T Generator.

La Figura 3 presenta un extracto del script de despliegue generado.

```
#Create Service
New-AzureService -ServiceName dealerpsec -Location "South Central us" -Label dealerpsec

#Create Deployment
$opstat = New-AzureDeployment -Slot Production -Package C:\scripts\windowsAzure\Dealer.cspkg
-Configuration C:\scripts\windowsAzure\ServiceConfig.Cloud.cscfg -Label "ContinuousDeploy-dealerpsec"
-ServiceName dealerpsec
$completeDeployment = Get-AzureDeployment -ServiceName dealerpsec -Slot Production
$completeDeploymentID = $completeDeployment.deploymentid
Set-AzureDeployment -Slot Production -ServiceName dealerpsec -Status Running
```

Figura 3. Extracto de script de despliegue generado para el servicio *Dealer*.

5. CONCLUSIONES

En este artículo presentamos un enfoque para la automatización de las actividades de despliegue de servicios en la nube, el cual aplica principios del desarrollo dirigido por modelos para hacer frente a la portabilidad de servicios entre diferentes proveedores de nube y agilizar el proceso de despliegue. El enfoque planteado en este trabajo provee un lenguaje para la especificación de requerimientos de recursos en la nube que satisfagan no solo los requerimientos específicos de los servicios a ser desplegados, sino que permite tomar decisiones de aprovisionamiento específicas de él o los proveedores de nube en los que se desplegarán los servicios. El uso del lenguaje de especificación es la base para, a través de transformaciones de modelo a texto, generar scripts de aprovisionamiento de recursos en la nube, despliegue y configuración de servicios en la nube.

Actualmente estamos trabajando en refinamiento del meta-modelo del *Modelo de Recursos en la Nube* de tal manera que soporte otros proveedores de recursos en la nube, así como en el desarrollo de casos de estudio más complejos. Se tiene previsto desarrollar mecanismos que faciliten la especificación de los recursos en la nube a través de plantillas que describan instancias de entornos nubes predefinidas. Además, se proveerá de un editor del lenguaje (meta-modelo) que soporte la especificación a través de una notación gráfica en lugar del utilizar el editor provisto por EMF.

AGRADECIMIENTOS

Investigación soportada por los proyectos DIUC_XIV_2016_038 – Universidad de Cueca y Value@Cloud (MICINN TIN2013-46300-R), y Microsoft Azure Research Awards.

REFERENCIAS

- Bergmayr, A., J. Troya, P. Neubauer, M. Wimmer, G. Kappel, 2014. *UML-based cloud app. modeling with libraries, profiles, and templates*. CEUR Workshop Proceedings, 56-65.
- Binz, T., U. Breitenbücher, O. Kopp, F. Leymann, 2014. TOSCA: Portable automated deployment and management of cloud applications. *Advanced Web Services*, Part III, 527-549.
- Chandrasekaran, K., 2014. *Essentials of cloud computing*. CRC Press Book, Taylor & Francis Group, 407 pp.
- Chef Software, I., 2015. *Chef DevOps and the cloud white paper*. Disponible en <https://www.microsoft.com/en-us/Openness/DevOps>.
- Di Martino, B., G. Cretella, A. Esposito, 2015. *Cloud portability and interoperability*. New York, NY: Springer Berlin Heidelberg.

- Fehling, C., F. Leymann, R. Retter, W. Schupeck, P. Arbitter, 2014. *Cloud computing patterns: fundamentals to design, build, and manage cloud applications*. Springer-Verlag, Wien.
- Homer, A., J. Sharp., L. Brader, M. Narumoto, T. Swanson, 2014. *Cloud design patterns: Prescriptive architecture guidance for cloud applications*. Microsoft Patterns & Practices, 232 pp.
- Meireles, F., B. Malheiro, 2014. *Integrated management of IaaS resources*. Euro-Par 2014: Parallel Processing Workshops, Pt Ii, 8806, 73-84.
- Mell, P., T. Grance, 2011. *The NIST definition of cloud computing - Recommendations of the National Institute of Standards and Technology*. U.S. Department of Commerce.
- Metsch, T., A. Edmonds, A. Others, 2010. *Open cloud computing interface - Infrastructure*. In: Standards Track, no. GFD-R in The Open Grid Forum Document Series, Open Cloud Computing Interface (OCCI) Working Group, Muncie (IN), 17 pp.
- Petcu, D., C. Craciun, M. Rak, 2011. *Towards a cross platform cloud API - Components for cloud federation*. Disponibile at <https://pdfs.semanticscholar.org/2fad/eb44dcc5b4b73a7233fac4bac7c9caeee612.pdf>, 4 pp.
- Swaminathan, B., K. Jain, 2012. *Implementing the lean concepts of continuous improvement and flow on an agile software development project: An industrial case study*. In: *2012 Agile India*, IEEE, pp. 10-19.