Despliegue de un Sistema Web Vertical Rígido como Sistema Web Elástico en la Nube

Fernando Invernizzi, Sebastián Ortiz, Carlos Núñez, and Cristian Cabanellas

Universidad Nacional de Asunción, Facultad Politécnica, Paraguay
{ferinver92,sebastianortizsantacruz}@gmail.com
{cnunez,cristian.cabanellas}@pol.una.py
http://www.pol.una.py/

Resumen La computación en la nube es una tecnología que se encuentra en crecimiento, la cual tiene la capacidad de escalar y aprovisionar recursos en forma dinámica sin la gestión de la complejidad subyacente. La necesidad de acceder a la información desde cualquier lugar y momento han llevado a repensar las formas en que se despliegan las aplicaciones actuales, esencialmente aquellas del tipo web. Los beneficios que ofrece la nube hacen de ella una candidata excelente para tomarla como solución. Entonces, ¿Cómo migramos desde el entorno tradicional?
Este trabajo implementa una metodología de migración extendiendo a un esquema conceptual base existente, presentando las adversidades durante el proceso y las conveniencias del mismo.
El estudio lanzó resultados alentadores, principalmente en inversión y eficiencia, representando una sólida justificación para la adopción del paradigma. Se espera que la presente investigación motive la migración de aplicaciones verticales rígidas a la nube.

Keywords: Computación en la nube, Metodología de migración, Aplicación nativa de la nube, Cloud Ready, IaaS, TICs

1. Introducción

Analizando cifras de mercado se comprueba que 65 % de las inversiones de TI se destinan al mantenimiento de infraestructuras y servicios ("Run the business spendings" expuesto en [1]). Esta es una de las principales razones por las que muchas organizaciones de TI muestran interés en una teconología que les permita disminuir y hacer variables los costes de infraestructuras y mantenimiento [2].

Una vía para incrementar la productividad es optimizando la inversión en infraestructuras acudiendo a servicios en la nube [2]. Al notar su atractivo, la pregunta que viene inmediatamente es: ¿Cómo migramos allí?

Este trabajo tiene como objetivo general contribuir con la implementación de una metodología de migración de un sistema web convencional alojado en infraestructura tradicional, a una infraestructura de computación en la nube, tomando como base un esquema conceptual propuesto por AWS [3], de manera a comprobar empíricamente las limitaciones y conveniencia de la migración en términos de coste y beneficios.

La metodología propuesta está orientada a la adopción de una arquitectura *Cloud Ready*, es decir, una arquitectura nativa de la nube. La validación *Cloud Ready* es uno de los principlales aportes del estudio.

El resto de este documento se estructura como sigue: los antecedentes y motivaciones en la sección 2. En la sección 3, se define *Cloud Ready* y el algoritmo de validación de aplicaciones. En la sección 4, se realiza la descripción de la metodología propuesta. En 5 se aplica la metodología a un caso de estudio. Finalmente, en la sección 6 se exponen las conclusiones del trabajo.

2. Antecedentes y Motivación

Normalmente se piensa que la migración a la nube no es más que la instalación en el entorno. Este concepto es erróneo ya que se confunde con una replicación, la cual no aprovecharía al máximo a la nube. Actualmente existen esquemas propuestos para migrar, pero los mismos sólo se presentan de manera conceptual.

En [4] se propone un modelo de cascada para la migración a la nube basado en el modelo cascada iterativo del Ciclo de Vida de Desarrollo Software (SDLC). El artículo destaca desafíos en el proceso sin incluir aspectos de implementación.

El framework CloudGenius [5] ofrece un proceso de migración y soporte para toma de desiciones, con un enfoque que considera solo el servidor web, además de asumir que primero se trabaja y elige una imagen de máquina virtual.

En [6], se concluye la clara necesidad de un proceso bien definido para la migración de aplicaciones a la nube y de los beneficios que se obtendrían.

[7] y [8] manifiestan la falta de apoyo para la migración a la nube en forma de frameworks, herramientas y métodos de ayuda para la toma de decisión.

En [9] los autores presentan una metodología para migración a la nube, pero para ambientes PaaS. Este trabajo pretende extender el mismo concentrándose en las aplicaciones mediante el aporte de una metodología orientandada a la adopción de una arquitectura nativa de la nube *Cloud Ready*.

3. Cloud Ready

CloudGenius [5] y *Cloud Migration Research: ASystematic Review* [8] mencionan la falta de apoyo para la adaptación arquitectónica de sistemas que se buscan llevar a la nube, logrando aprovechar de la mejor manera el entorno.

Cloud Ready es aquel servicio diseñado para trabajar a travéz de internet [10]. Es una aplicación que puede ser efectivamente desplegada en la nube [11].

La validación *Cloud Ready*, aporte de este trabajo, permite una arquitecura óptima para la nube mediante el cumplimiento de los principios *Cloud Ready*,

3.1. Principios Cloud Ready

Los principios *Cloud Ready* son el conjunto de características que necesita un aplicativo para ser considerado poseedor de una arquitectura nativa de la nube.

Estos principios se elaboraron a partir de una serie de artículos y estudios [11][12][13][14][15][16][17][18]. A continuación se expone el conjunto formulado:

1. **No escribir la aplicación para una topología específica.** No suponer la existencia fija de nodos [11][14][18].
2. **No asumir que el sistema de archivos local es permanente.** No suponer el ciclo de vida de los archivos en el sistema de archivos [11][18].
3. **No mantener una sesión en la aplicación.** Una aplicación del tipo stateful limita la escalabilidad, se sugiere que sea stateless [11][12][13][18].
4. **No escribir logs en el sistema de archivos.** Con logs escritos localmente, se pierde información y trazabilidad si la máquina falla [11][12][13][14][18].
5. **No utilizar APIs de infraestructura ni características propias del sistema operativo.** Evitar utilizar "APIs de la infraestructura o de SO" ya que dificultan el monitoreo por parte del proveedor de nube [11][14].
6. **No utilizar protocolos oscuros.** Evitar protocolos que requieran de alguna configuración especial que pueda afectar la flexibilidad [11][12][13][18].
7. **No instalar manualmente la aplicación.** La instalación sencilla permite adoptar diferentes técnicas de automatización [11][12][13][14][15][17][18].
8. **Código base.** Cada aplicación desplegable se debe mantener en un código único bajo un control de revisión [12][13][18].
9. **Servicios de soporte.** Los servicios de soporte deben tratarse como recursos adjuntos, de consumo idéntico en los entornos de despliegue [12][13].
10. **Concurrencia.** Dos o más procesos son concurrentes cuando se ejecutan al mismo tiempo y no existen dependencias entre los mismos [12][13].
11. **Desechabilidad y robustez.** Es el manejo de entradas inválidas, permitiendo así confiabilidad y recuperación frente a accidentes [12][13][15][17][18].
12. **Procesos administrativos.** Procesos de "una sola vez"deberían ejecutarse en entornos idénticos a los procesos regulares de larga duración [12][13][18].
13. **Escalabilidad.** Los sistemas que se espera crezcan con el tiempo necesitan ser construidos sobre una arquitectura escalable [14][15][16][17].
14. **Procesos stateless.** Se debe tener componentes débilmente acoplados, que idealmente no compartan nada [16][17][18].
15. **Paralelización.** Busca analizar cómo superponer operaciones para mejorar el rendimiento y el uso de infraestructura al realizar una tarea concreta [16].
16. **Técnicas y estrategias de tolerancia a fallos.** Describe un sistema diseñado de modo a que en caso de fallo puede restablecerse y alertar sin la pérdida del servicio [14][15][16][18].
17. **Evitar adivinar la necesidad de capacidad.** Una aplicación orientada a la nube debe prepararse para usar recursos bajo demanda [14][16][18].
18. **Aplicar seguridad en todas las capas.** Se debe proteger la integridad y privacidad de la información almacenada en un sistema [14][15][18].

3.2. Validación Cloud Ready

El algoritmo de validación *Cloud Ready* (presentando en Alg. 1) utiliza los principios *Cloud Ready* para comprobar si una aplicación efectivamente cuenta con una arquitectura nativa de la nube.

Antes de iniciar, la validación requiere de una buena comprensión del aplicativo a migrar. Se sugiere su descomposición y análisis en detalle.

Por cada principio, se evalúa el cumplimiento de la arquitectura del sistema. Si cumple, se pasa al siguiente principio. En caso de no cumplirlo, se lo ajusta.

Algoritmo 1 Validación Cloud Ready

Entrada: Aplicativo A, Conjunto de Principios CP
Salida: Aplicativo con arquitectura Cloud Ready.

```
1: para todo P en CP hacer
2:    si A cumple P entonces
3:       obtener siguiente principio
4:    si no
5:       ajustar A para que cumpla P
6:    fin si
7: fin para
```

El empleo ideal del algoritmo *Cloud Ready* ofrece un aplicativo con la arquitectura capaz de explotar de la mejor manera las características de la nube.

3.3. Extensión a la validación Cloud Ready en caso de limitaciones

En algunos casos, por diversos motivos (dinero, esfuerzo, capacidad, tiempo, entre otros), los principios no pueden ser alcanzados en su totalidad. Para estas situaciones, el proceso a seguir para determinar si es beneficioso continuar con la migración está compuesto por las siguientes tareas:

- **Recuento de principios**, tarea que buscar obtener el porcentaje *Cloud Ready*, el cual se calcula a partir de la sumatoria de valoración de los principios. Cada principio se evalúa según el grado de cumplimiento que tenga (Alta, Media, Baja) y, a partir de ese grado, se suma su correspondiente valoración (1, 0.5, 0) al total.
- **Conclusión a partir del recuento**, tarea que busca determinar a partir del porcentaje *Cloud Ready*, si el aplicativo se encuentra apto para ser migrado. Si el porcentaje *Cloud Ready* es superior al 80 %, el aplicativo es apto para la migración, si se encuentra entre 60 % y 80 % es medianamente apto y, en caso de ser inferior al 60 %, no resulta ser apto.

Si se tuviera un aplicativo que no cumple plenamente con los 18 principios, y cuya sumatoria de valoración, arroje un total de 15, su porcentaje *Cloud Ready* estaría definido por: 15/18*100 = 83,3 %, con lo que se puede concluir, que el aplicativo de igual modo se encuentra en condiciones para ser migrado.

4. Metodología propuesta de migración

El modelo conceptual de Amazon [3] fue la base para la implementación de la metodología de migración. La elección se debe a que su enfoque se corresponde con nuestras percepciones iniciales de la necesidad existente, envolviendo aspectos de caracterización, con la visión de explotar los beneficios de la nube.

La metodología propuesta se ocupa de comprender a la organización en la que uno se encuentra, conocer los diferentes sistemas junto a la complejidad de la arquitectura subyacente de cada una y su relación con los demás. Plantea una ruta para reconocer las virtudes del proveedor de nube, migrar el aplicativo según la estrategia mas conveniente y añadir servicios que aprovechen a la nube.

Este trabajo aporta lo siguiente:

- Evalúa y enfoca al aplicativo a una arquitectura *Cloud Ready*.
- Discute aspectos técnicos y limitaciones (tiempo, dinero) para la toma de desiciones durante la migración.
- Presenta una perspectiva general para la selección de los servicios ofrecidos por el proveedor de nube.
- Incluye un esquema de pruebas para validar los beneficios de la nube en el aplicativo migrado.

Como resultado, la metodología migra un aplicativo a la nube capaz de aprovechar al máximo de sus beneficios. La Fig. 1 presenta la metodología final.

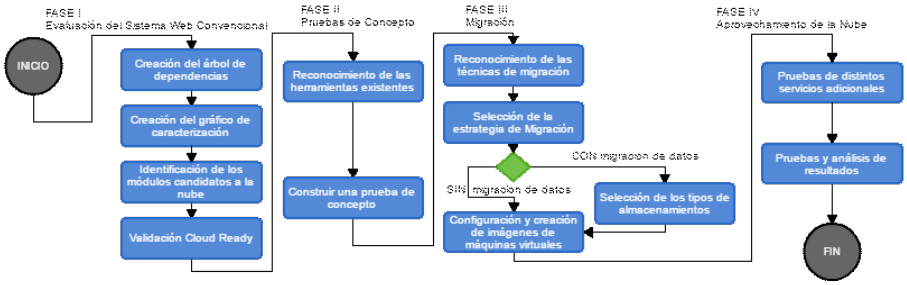

Figura 1. Metodología Propuesta

La metodología utiliza el proveedor AWS [19] para su ejemplificación ya que cuenta con importantes informes elaborados por Gartner y Forester que destacan su integridad, capacidades críticas y seguridad [20][21][22][23].

4.1. Evaluación del Sistema Web Convencional

La fase inicial busca conocer los sistemas de la organización, analizar las conexiones existentes, caracterizar a los componentes y, partiendo de la caracterización, identificar los mejores candidatos para la selección y ajuste del mismo.

Creación del árbol de dependencias

Esta etapa reliza un examen de las conexiones lógicas existentes entre los sistemas mediante la elaboración de un árbol de dependencias, el mismo permite identificar las aplicaciones y sus dependencias sobre otros componentes. Los nodos representan los componentes y las aristas las relaciones.

Creación del gráfico de caracterización
El gráfico de caracterización se resume en un diagrama de las aplicaciones de la organización. Se conforma por los módulos identificados en el árbol de dependencias, donde cada módulo se caracteriza según su papel (dependencias).

Identificación de los módulos candidatos a la nube
Los mejores candidatos para la nube, sin tener en cuenta modificaciones estructurales en su actual disposición, son aquellas aplicaciones con pocos componentes; aplicaciones con necesidad de escalar y que corren por sobre su capacidad. Definida la lista de candidatos, se debe priorizar la migración según convenga.

Validación Cloud Ready
Con el aplicativo seleccionado, lo siguiente es emplear la validación *Cloud Ready* (sección 3) sobre el mismo. El objetivo de la etapa es que el aplicativo a migrar posea una arquitectura capaz de aprovechar eficientemente a la nube.

4.2. Pruebas de Concepto

Esta fase se centra en conocer las características del proveedor y obtener confianza en sus servicios dando los primeros pasos sobre el mismo.

Reconocimiento de las herramientas existentes
Esta actividad consiste en familiarizarse con las herramientas del proveedor. Los servicios básicos a aprender deberían ser: creación de máquinas virtuales, gestión de imágenes virtuales, asignación de volúmenes de almacenamiento, carga de datos a la nube, gestión de base de datos, entre otras cosas.

Construir una prueba de concepto
Al conocer los servicios básicos del proveedor, se debe continuar con la construcción de una prueba de concepto. La prueba consiste en representar al aplicativo en la nube mediante los servicios reconocidos.

4.3. Migración

Esta etapa presenta las principales estrategias de migración de aplicaciones y los aspectos a tener en cuenta para la selección de tipos de almacenamiento.

Reconocimiento de las técnicas de migración
Una estrategia de migración representa un plan que permite trasladar a la nube un aplicativo. Las principales estrategias que se sugiere conocer y tener en cuenta para la migración son la híbrida, de reemplazo, forklift y cloudify [8] [24].

Selección de la estrategia
La selección implica decidir la estrategia de migración que más convenga. *Cloud Ready* sugiere la estrategia cloudify. Si la estrategia requiere de almacenamiento se debe continuar con la "Selección de los tipos de almacenamientos", de otro modo con la "Configuración y creación de imágenes de máquinas virtuales".

Selección de los tipos de almacenamientos
Para la selección del tipo de almacenamiento adecuado, se deben realizar las compensaciones justas entre las diversas dimensiones existentes (costo, disponibilidad, durabilidad, latencia, performance, capacidad de cache y consistencia).

Configuración y creación de imágenes de máquinas virtuales
La creación de imágenes virtuales y procesos de despliegue automatizados reducen el tiempo y esfuerzo de construcción de una aplicación [3]. Se recomienda servirse de herramientas de administración de despliegue y configuración.

4.4. Aprovechamiento de la Nube

Esta fase se centra en incorporar aptitudes del proveedor de la nube a la arquitectura migrada y la creación de un esquema de pruebas cuyo objetivo es el de determinar el éxito de la migración. Ambas tareas se encuentran fundamentadas sobre las cinco características esenciales definidas por el NIST [25].

Pruebas de distintos servicios adicionales
El apartado se centra en realizar pruebas sobre servicios que permitan fortalecer al aplicativo desplegado inicialmente con características de la nube. Se recomienda incluir servicios que potencien a las prestaciones (autoescalamiento, balanceo de carga y otros).

Pruebas y análisis de resultados
El esquema de pruebas para determinar el éxito de la migración es como sigue:

1. *Emparejamiento*: consiste en asociar los componentes de la arquitectua en el entorno tradicional con los del entorno en la nube.
2. *Definición de las funcionalidades de prueba*: la selección debe incluir a aquellas que causen más carga en el sistema y se utilicen con mayor frecuencia.
3. *Selección de la herramienta de pruebas*: la herramienta debe permitir inyectar cargas reales al sistema. Debe incluir características como concurrencia, definición de secuencia de tareas y la generación de reportes representativos.
4. *Evaluación*: se divide en la estimación del costo de los servicios utilizados y la medición del éxito de la migración con respecto a las características nube.
   a) *Costo de servicios*: El objetivo es tener un estimativo del costo de los servicios utilizados en la nube y su rentabilidad en el tiempo.
   b) *Características nube*: El éxito de la migración se evalúa a partir del comportamiento del aplicativo con respecto al cumplimiento de las características esenciales de la nube definidas por el NIST [25].

5. Caso de estudio

El entorno seleccionado para el estudio es el de una entidad financiera. Fue elegido por la cantidad de usuarios que maneja y la alta demanda de recursos.

El árbol de dependencias y el gráfico de caracterización permitieron detectar todos los módulos y conexiones dentro de la entidad. Ante la alta demanda del particular *módulo de transferencias*, se lo priorizó para buscar otorgarle flexibilidad y elasticidad mediante la nube. La validación *Cloud Ready* (sección 3) no pudo ser empleada idealmente por falta de tiempo y capital. De igual forma, se realizó el cálculo del porcentaje *Cloud Ready* obteniendo un 91,6 %, por lo que se concluyó que el mismo se encontraba apto para la migración.

En las pruebas de concepto, se reconoció y estudió a los servicios *Elastic Compute Cloud* (EC2) [26], *Elastic Block Store* (EBS) [27], *Relational Database Service* (RDS) [28] y *Simple Storage Service* (S3) [29] ofrecidos por AWS.

Para la migración, la técnica utilizada fue *cloudify*, reemplazando todos los componentes por servicios de AWS. El detalle es como sigue: a) el módulo core instalado en una instancia EC2, b) anexo de volumen EBS a la instancia EC2 en la cual se situarán los archivos de configuración de arranque, c) base de datos migrada como una instancia RDS, y d) archivos estáticos y logs en un volumen S3. Por otra parte, la migración no utilizó herramientas de despliegue.

Para el aprovechamiento de la nube, se utilizó el Grupo de Autoescalado [30] y el *Elastic Load Balancing* (ELB) [31]. La calculadora de Costo Total de Propiedad (TCO) [32] con los parámetros que reflejan la infraestructura antes de la migración, estima un ahorro del 97 % de los costos en 3 años. Finalmente, las figuras 2 y 3 comprueban el éxito de la migración con la obtención del autoservicio bajo demanda y la rápida elasticidad del aplicativo en la nube.

Figura 2. Autoservicio bajo demanda. El eje Y representa la *Carga de CPU (%)* y el eje X la cantidad de *Usuarios por segundo*. Luego del punto determinado por 150 usuarios por segundo y 62 % de carga, la CPU disminuye debido a su autoescala en base a la demanda de los usuarios, por lo que queda demostrada la característica.

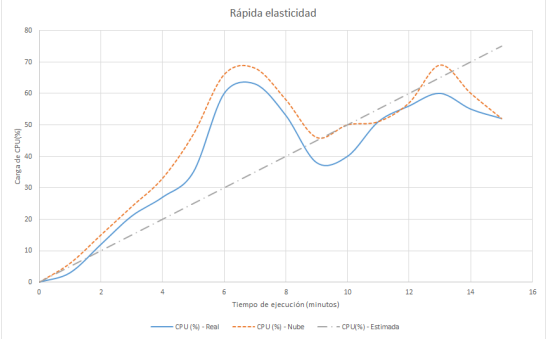

Figura 3. Rápida elasticidad. Se aprecian 3 curvas de la *Carga de CPU (%)*, la lisa representa a una instancia única, la de guión corto a la aplicación en la nube (con autoescalado y balanceo) y la de guión largo a la carga estimada. Como la curva de la aplicación en la nube es similar a las de instancia única y estimada, se comprueba la característica ya que, independientemente a la demanda, la adaptación es dinámica.

6. Conclusión

Las organizaciones encuentran atractiva a la nube por los múltiples beneficios que ofrece, principalmente con la facilidad de gestión de recursos, la reducción de costos mediante el modelo *Pay as you go* y el incremento de la productividad.

En la actualidad no se cuenta con una guía, a nuestro entender lo suficientemente completa y actualizada, que permita migrar a la nube aplicaciones que son desplegadas en entorno tradicionales. Este trabajo ofrece una metodología que se encarga de cubrir dicha necesidad.

La metodología implementada se encuentra basada en el esquema conceptual de Amazon [3], aunque con ciertas modificaciones debido al direccionamiento hacia sus productos y la falta de detalles técnicos para la toma de decisiones. La metodología propone la definición de un esquema *Cloud Ready*, permitiendo así obtener el mayor de los beneficios del nuevo entorno.

La utilización de la metodología en el caso de estudio arroja resultados alentadores, consolidando al proceso y los cambios hechos al esquema base adoptado, favoreciendo al entorno nube por sobre el tradicional haciendo notar principalmente la reducción de costos y la mejora del rendimiento frente a cargas variables.

Referencias

1. Gartner IT Key Metrics Data, 2012 IT ENTERPRISE, SUMMARY REPORT, *http://www.gartner.com/resId=1872515*.
2. Management Solutions, *La nube: oportunidades y retos para los integrantes de la cadena de valor*, 2012.
3. Jinesh Varia, *Migrating your Existing Applications to the AWS Cloud - A Phase-driven Approach to Cloud Migration*, October 2010.

4. Rashmi, Dr. Shabana Mehfuz, Dr.G.Sahoo, *A five-phased approach for the cloud migration*, ISSN 2250-2459, Volume 2, Issue 4, April 2012.
5. Michael Menzel, Rajiv Ranjan, *CloudGenius: Decision Support for Web Server Cloud Migration*, April 16–20, 2012, Lyon, France.
6. Rashmi Rai, Dr. Shabana Mehfuz, Dr. G. Sahoo, *Efficient Migration of Application to Clouds: Analysis and Comparison*, GSTF Journal on Computing (JoC), Vol. 3 No. 3, Dec 2013.
7. Raghavan P, Shiva Murthy G, *Migration of Legacy Application to Cloud Environment: A Survey.*
8. Pooyan Jamshidi, Aakash Ahmad, and Claus Pahl, *Cloud Migration Research: A Systematic Review*, IEEE.
9. Juan Marcelo Pintos, Carlos Núñez Castillo, Fabio López Pires, *Evaluation and Comparison Framework for Platform as a Service Providers.*
10. PCMAG CR, *http://www.pcmag.com/encyclopedia/term/67090/cloud-ready.*
11. Kyle Brown, Mike Capern, *Top 9 rules for cloud applications*, IBM, April 09, 2014.
12. Matt Stine, *Migrating to Cloud-Native Application Architectures (Twelve factor)*, O'Reilly.
13. The twelve-factor App, *https://12factor.net/.*
14. AWS Well-Architected Framework. November 2016.
15. Architecting for the Cloud - AWS Best Practices. February 2016.
16. Jinesh Varia, *Cloud Architectures*, June 2008.
17. Maram Mohammed Falatah, Omar Abdullah Batarf, *Cloud Scalability Considerations*, International Journal of Computer Science & Engineering Survey (IJCSES) Vol.5, No.4, August 2014.
18. James Hamilton - Windows Live Services Platform, *On Designing and Deploying Internet-Scale Services.*
19. Amazon Web Services, *https://aws.amazon.com/es/.*
20. Gartner: Magic Quadrant for Cloud Infrastructure as a Service, Lydia Leong, Mayo, 2015, *https://aws.amazon.com/es/resources/gartner-2015-mq-learn-more/.*
21. Gartner: Magic Quadrant for Public Cloud Storage Services, Lydia Leong, Junio, 2015, *http://aws.amazon.com/es/resources/gartner-storage-mq-learn-more/.*
22. Gartner: Critical Capabilities for Public Cloud Infrastructure as a Service, Lydia Leong, Octubre, 2015, *https://aws.amazon.com/es/resources/gartner-cc-learn-more/.*
23. Forrester Wave: Public Cloud Service Providers' Security, *https://aws.amazon.com/es/resources/forrester-security-wave-learn-more/.*
24. Tobias Binz, Frank Leymann, David Schumm, *CMotion: A Framework for Migration of Applications into and between Clouds*, Institute of Architecture of Application Systems, University of Stuttgart, Germany.
25. National Institute of Standards and Technology, *The NIST Definition of Cloud Computing*, NIST Special Publication 800-145.
26. Amazon EC2, *https://aws.amazon.com/es/ec2/.*
27. Amazon Elastic Block Store, *https://aws.amazon.com/es/ebs/.*
28. AWS Relational Database Service (RDS), *https://aws.amazon.com/es/rds/.*
29. Amazon S3, *https://aws.amazon.com/es/s3/.*
30. Amazon Auto Scaling, *https://aws.amazon.com/es/autoscaling/.*
31. AWS Elastic Load Balancing, *https://aws.amazon.com/es/elasticloadbalancing/.*
32. AWS Total Cost of Ownership (TCO) Calculator, *https://awstcocalculator.com/.*