



APLICACIÓN WEB PROTOTIPO PARA LA
EVALUACIÓN DE COSTOS Y APROVISIONAMIENTO
CON LIBCLOUD SOBRE AMAZON WEB SERVICES

PRICECLOUD

ANTEPROYECTO - TRABAJO PROFESIONAL

SEBASTIÁN AFANADOR FONTAL

sebastian.afanador@correounivalle.edu.co

Código 1629587

JOHN ALEXANDER SANABRIA ORDÓÑEZ PH.D

john.sanabria@correounivalle.edu.co

Facultad de Ingeniería
Escuela de Ingeniería de Sistemas y Computación
Universidad del Valle
Cali - Colombia

Marzo 2021 – Versión 1.4.0

El Web Scraping son un conjunto de técnicas y tecnologías para extraer información de la web usando algoritmos de exploración recursiva en todos los Endpoints de un sitio web. LibCloud permite generar una capa de abstracción a partir de una API unificada usando controladores para cada tipo de servicio que prestan muchos de los CCSPs.

RESUMEN

La aplicación web PriceCloud propone integrar la fase de evaluación de costos de los principales proveedores de servicios de CC junto con su despliegue en una única interfaz para dar un criterio informado que ayude a minimizar los costos de funcionamiento de un proyecto de software con respecto a sus necesidades iniciales.

Usando principalmente técnicas de *Web Scraping* este trabajo propone crear un servicio que sea una fuente de información veraz y actual de los costos de funcionamiento de los servicios ofrecidos por los principales CCSP y además una aplicación que presente esta información al usuario y le permita desplegar sus recursos usando la librería LibCloud¹.

¹ <https://libcloud.apache.org/about.html>

ÍNDICE GENERAL

1	INTRODUCCIÓN	1
2	PLANTEAMIENTO DEL PROBLEMA	2
3	JUSTIFICACIÓN	3
3.1	Justificación Económica	3
3.2	Justificación Académica	3
4	OBJETIVOS	4
4.1	Objetivo general	4
4.2	Objetivos específicos	4
5	RESULTADOS ESPERADOS	5
6	ALCANCES DE LA PROPUESTA	6
7	MARCO REFERENCIAL	7
7.1	Estado del arte	7
7.1.1	LibCloud	7
7.1.2	Modelos de precios	8
7.2	Marco teórico	8
7.2.1	Conceptos de Cloud	8
7.2.2	Modelos de precios	9
8	METODOLOGÍA	11
8.1	Actividades a realizar	11
8.2	Cronograma de Actividades	11
9	PRESUPUESTO	12
	Glosario	13
	Siglas	15
	BIBLIOGRAFÍA	16

INTRODUCCIÓN

El **Cloud Computing** ha facilitado el despliegue de todo tipo de aplicaciones con un uso adaptable de recursos, esto es un punto importante del cloud, permitirle al cliente iniciar con recursos de base que luego pueden ser escalados a la medida de las necesidades que la aplicación requiera. En contraposición de una infraestructura *On-premise* las tecnologías cloud se adaptan en tiempo real a los requerimientos cambiantes de la demanda de estas aplicaciones y le permiten al cliente invertir recursos vitales en procesos mas relevantes eliminando los costos de mantenimiento e infraestructura tecnológica.

El aprovisionamiento de recursos en proveedores de servicios cloud es un proceso que cada vez toma mas relevancia en aplicaciones que han apostado por la tendencia a migrarse al **CC**. Sin embargo a la hora de elegir estos recursos se deben tener en cuenta aspectos como la arquitectura del proyecto, el tipo de recursos, su ubicación geográfica, las políticas donde recidirán, la reputación del proveedor y por supuesto de su costo.

Uno de los aspectos clave para ayudar a garantizar la continuidad de un proyecto de software son precisamente sus costos fijos; el cloud los discretiza con el modelo *Pay as You Go* en términos de peticiones, duración, poder de cómputo, espacio de almacenamiento, usuarios, y demás variables que le agregan aún mas complejidad a la decisión de escoger el proveedor adecuado para los recursos requeridos por el proyecto.

PriceCloud permitirá interactuar sobre una interfaz web donde el usuario podrá tomar una decisión informada conociendo los costos de su configuración de recursos sobre varios **CCSP** y de manera complementaria gracias a los controladores que ofrece **LibCloud** podrá ser asistido en su aprovisionamiento.

El CC es una tendencia de servicios y recursos que funcionan sobre Internet para facilitar el despliegue, uso y disponibilidad de aplicaciones de Software.

En una configuración On-premise el cliente se encarga de hospedar sus aplicaciones en uno o varios servidores en su propia infraestructura.

El modelo comercial Pay as You Go permite al cliente pagar solo por lo que ha usado después de haberlo usado.

2

PLANTEAMIENTO DEL PROBLEMA

Según lo mencionado anteriormente esta propuesta de trabajo de grado busca responder las siguientes preguntas. ¿Cómo dar un criterio informado respecto a los costos de aprovisionamiento de una aplicación facilitando su despliegue?, ¿Cómo construir un servicio que brinde información actualizada de los costos de los proveedores?, ¿Cómo proponer un proyecto colaborativo que sea extensible a nuevos proveedores y servicios?, ¿Qué tecnologías y arquitectura usar para procurar su ampliación?.

JUSTIFICACIÓN

3.1 JUSTIFICACIÓN ECONÓMICA

Actualmente no existe una tecnología *Open Souce* que permita administrar recursos en **CCSPs** teniendo en cuenta los costos de servicios; gracias a que PriceCloud será una proyecto de código abierto podrá implementarse de manera gratuita por cualquier desarrollador para dar un punto de apoyo en este proceso que carece de un marco bien definido.

3.2 JUSTIFICACIÓN ACADÉMICA

Diseñar e implementar esta propuesta pone a prueba las destrezas técnicas y formación del aspirante a grado; Este proyecto de código libre permitirá tener un punto de partida para que mas interesados puedan contribuir en la solución del problema que parece tener carencia de participación.

La propuesta y desarrollo de este proyecto sugieren conocimientos de Desarrollo de Software, Bases de Datos Relacionales, Aplicaciones Web, Microservicios, Web Scraping, Crawling, Seguridad Web, Web Services, Redes de Computadores y Sistemas Operativos.

4

OBJETIVOS

El lenguaje [XPath](#) es un recurso que permite describir patrones para hallar coincidencias en los [Endpoints](#) de un sitio Web.

La recuperación de información en la web inicia con el [Crawling](#) que permite obtener las páginas web de objetivo para luego aplicar [Web Scraping](#) que recopila información de interés.

4.1 OBJETIVO GENERAL

Diseñar, e implementar un prototipo de aplicación web para informar las tarifas de aprovisionamiento de recursos tipo *Compute, Storage y Container* en el [CC](#) y administrar su orquestación.

4.2 OBJETIVOS ESPECÍFICOS

1. Definir un modelo general de evaluación de costos para el análisis de los servicios en el [CC](#) usando la mayor cantidad de información recuperable y relevante de la Web de sus proveedores.
2. Diseñar e implementar una aplicación basada en microservicios que recopile periódicamente y almacene las tarifas de los principales [CCSP](#) usando técnicas de *Web Scraping* y *Crawling* con [XPath](#), bases de datos relacionales y una [API REST](#).
3. Diseñar e implementar una aplicación web basada en microservicios para acceder a la información de costos de los proveedores y administrar los recursos de un usuario usando la [API](#) de [LibCloud](#).

RESULTADOS ESPERADOS

Cuando se culmine el proyecto propuesto en este escrito se espera cumplir los objetivos específicos mencionados anteriormente como se describe a continuación.

1. Este modelo permite tener un punto de referencia para medir los proveedores equitativamente teniendo en cuenta la mayor información que se pueda recuperar de manera pública, se espera poder incluir en todos los casos los costos para cada tipo de servicio, e información adicional como el tiempo en línea y la proximidad geográfica de cada proveedor que permitan tener un criterio mas amplio para tomar una decisión. Este modelo se incorporará al servicio web principal del proyecto.
2. Esta parte del proyecto permite manejar de manera centralizada, actualizada y con una trazabilidad la información relevante de los [CCSP](#), se pretende construir un servicio en línea que funcione sin interrupciones pues su responsabilidad es responder a las consultas hechas por las instancias de la interfaz web.
3. Esta parte de la aplicación puede ser desplegada a discreción de cualquier usuario sobre su propia infraestructura para poder usar la solución propuesta en una interfaz Web, se pretende que esté disponible en una imagen pública en [Docker Hub](#) con su respectiva documentación y parámetros de configuración usando [Docker Compose](#).

El servicio principal de la aplicación también podrá ser desplegado en un stack de contenedores dispuesto en [Docker Hub](#) solo si el usuario está interesado en recuperar por sus propios medios la información de los proveedores.

La comunicación de todos los web services estará encriptada usando el protocolo [HTTPS](#) con certificados libres de [Let's Encrypt](#), los datos sensibles almacenados usarán el algoritmo de encriptación simétrica [AES](#).

6

El archivo robots.txt ubicado en la raíz de un sitio web indica a los Bots que información es de carácter privado o no debe ser indexada.

ALCANCES DE LA PROPUESTA

PriceCloud pretende recuperar de la Web información de los principales CCSPs entre los que están [AWS](#), [Azure](#), [Google Cloud](#), [IBM Cloud](#) y [Digital Ocean](#). PriceCloud también tiene en cuenta proveedores locales (en Colombia) como [Host Dime](#), [Hosting RED](#); Se espera que el registro de nuevos proveedores se haga por cualquier usuario y se valide por el administrador de la aplicación.

PriceCloud tiene soporte básico de administración para los recursos de tipo *Compute*, *Storage* y *Container* solamente para el proveedor [AWS](#) pero con la posibilidad de extender el proyecto a nuevos tipos de servicios teniendo en cuenta las disponibilidad de controladores¹ de la API de [LibCloud](#).

La implementación de la interfaz web se construye en [React](#) a partir de la librería [CoreUI](#)², es la razón por la que esta propuesta no considera el diseño [UI](#) y [UX](#) como un componente principal.

La información que se pretende recuperar de la Web de los [CCSPs](#) tiene en cuenta las limitaciones técnicas y legales que los mismos proveedores incluyen en sus políticas y en el archivo robots.txt para evitar incurrir en problemas legales.

La implementación de la aplicación con todos sus componentes tiene como objetivo mostrar la aplicación en funcionamiento, sin embargo los aspectos técnicos adicionales referentes a su despliegue como pruebas de rendimiento, pentesting y mantenimiento no se tienen en cuenta en esta propuesta.

¹ https://libcloud.readthedocs.io/en/stable/supported_providers.html

² <https://github.com/coreui/coreui-react/>

MARCO REFERENCIAL

Este documento toma como referencia dos ejes temáticos principalmente, las variables de modelos de precios de los CCSPs y las tecnologías referentes a la administración, despliegue y orquestación de recursos de CC bajo un esquema de iteroperabilidad.

7.1 ESTADO DEL ARTE

Actualmente existen recursos de software para la orquestación de recursos en la nube, estos son bien conocidos como **Cloud Resource Orchestration Frameworks (CROFs)**, han surgido como sistemas para gestionar el ciclo de vida de los recursos de multiples CCSPs y que cada vez exigen mecanismos de orquestación de recursos capaces de tratar con la heterogeneidad subyacente. [8].

7.1.1 LibCloud

Es una librería escrita en Python tiene licencia *Apache 2.0*. Es una de las mas completas para administrar recursos de CC teniendo unicamente carencias en el soporte de DBaaS¹. Esta librería oculta las diferencias entre las API de los CCSPs y permite administrar diferentes recursos de la nube a través de una API unificada y facil de usar.

Esta librería divide sus funciones en seis principales categorías². La primera permite administrar Servidores en la nube y *Block Storage*, este componente permite ejecutar secuencias de comandos para preparar al servidor recién creado. Otro tipo de recurso administrable es el almacenamiento de objetos en la nube *Object Storage* y la administración de CDNs, Los servicios de balanceadores de carga *Load Balancer*, la API para administración de DNSs como servicio y los servicios de administración

¹ <https://medium.com/@anthonyjpshaw>

² <https://libcloud.readthedocs.io/en/stable/index.html>

de Contenedores que permiten a los usuarios implementar y administrar contenedores usando software como [Docker](#) con los proveedores que ofrecen una [API](#) de [CaaS](#).

7.1.2 *Modelos de precios*

Los [CCSPs](#) definen sus cobros a partir de diferentes

7.2 MARCO TEÓRICO

7.2.1 *Conceptos de Cloud*

Según el [NIST](#) la computación en la nube está definida como Un modelo que permite el acceso de red conveniente y bajo demanda a un grupo compartido de recursos informáticos configurables como redes, servidores, almacenamiento, aplicaciones y servicios que pueden aprovisionarse y liberarse rápidamente con un mínimo esfuerzo de gestión o interacción con el [CCSP](#)[5].

En el [CC](#) existen varios tipos de configuración según la ubicación de los servicios

Los servicios de [CC](#) ofrecen mas beneficios que la computación tradicional, ahorro de costos, escalabilidad, almacenamiento movil, acceso desde cualquier momento o cualquier lugar, mejor seguridad, ahorro de energía[7]. En terminos de taxonomía el [CC](#) está dividido en tres principales tipos de servicios, [Infrastructure as a Service \(IaaS\)](#) que puede verse como una evolución de los [Virtual Private Servers \(VPSs\)](#) la cual ofrece plataformas de virtualización donde los clientes deben configurar sus propio software, estos servicios son facturados con base en los recursos consumidos, el modelo de pago por alquiler permite usar hasta mínimo una hora en algunos casos donde la cantidad de instancias se duplican para satisfacer las necesidades los clientes. Entre los tipos de soluciones que se encuentran en esta categoría están [Amazon Elascitic Compute Cloud \(EC2\)](#), *Rackspace Cloud*, *IBM Smart Bussiness Cloud solutions*, *Oracle Cloud Computing*, *Google Compute Engine*. [7, p.2]

Otro esquema de servicios ofrecidos por el [CC](#) son las [Platform as a Services \(PaaS\)](#), como características principales tenemos que son proveidas a los clientes por una interfaz en un navegador para editar, depurar, desplegar y monitorear una

aplicación.[4, p.1] A diferencia de las **IaaS** permiten una interacción de alto nivel con el cliente porque abstraen la configuración del entorno donde se ejecutan las aplicaciones, además de permitir la preconfiguración de un entorno de ejecución para un lenguaje de programación específico. Entre los **CCSPs** mas conocidos se encuentra también **EC2** con la posibilidad de preconfigurar los entornos, *Salesforce.com* y *Google App Engine*.

El siguiente nivel en terminos de abstracción son las **Function as a Services (FaaS)** conocidas también como computación *Serverless* [6, p.1] que emerge como un paradigma conveniente para el despliegue de aplicaciones y servicios donde el desarrollador no se preocupa acerca de los aspectos operacionales, de despliegue, y mantenimiento y espera de este que sea tolerante a fallos y auto escalable especialmente el código que es escalable a cero donde no hay servidores corriendo cuando la función del usuario no está siendo usada. **FaaS** a diferencia de **Platform as a Services (PaaS)** evita que el cliente reciba cobros en los periodos de inactividad de la función.[2, p.5] Entre los servicios mas conocidos están *Azure Functions*, **AWS Lambda**, *Google Cloud Functions* y *Oracle Cloud Functions*.

Por último y mas cerca del usuario se encuentran el **Software as a Service (SaaS)**, en este modelo de servicios el cliente interactúa con una aplicación que da soporte a los procesos de su modelo de negocio o le ofrece un servicio, el cliente paga recurrentemente por las funcionalidades y la cantidad de usuarios que usan el servicio, el **SaaS** presenta ventajas frente al software tradicional, el cliente compra una suscripción y no una licencia perpetua, además el software tradicional tarda algunos años en publicar una nueva versión que el **SaaS** puede poner a disposición en cuanto esté completa [3, p.1]. Algunos ejemplos de **SaaS** son *Google Workspace*, *Microsoft 365*

7.2.2 Modelos de precios

De manera general existen dos grupos de modelos de precios, los modelos de precios dinámicos y los modelos de precios fijos [1]; En los modelos fijos el costo no cambia por un largo tiempo en contraposición de los modelos dinámicos que cambian con respecto a objetos medibles por el **CCSP** como las peticiones de los usuarios o la condición del mercado.

La mayoría de los CCSPs usan modelos de precios dinámicos con esquema de pago por uso [1], sin embargo es pertinente hacer una corta revisión para conocer los esquemas de precios de los principales proveedores. La empresa ...

7.2.2.1 *Modelo de pago por uso Pay-as-you-go*

En este modelo es el mas común y

METODOLOGÍA

8.1 ACTIVIDADES A REALIZAR

8.2 CRONOGRAMA DE ACTIVIDADES

9

PRESUPUESTO

GLOSARIO

Amazon Web Services Es una empresa lider mundial en servicios de [CC](#). [15](#)

Azure Es una empresa de Microsoft que ofrece servicios de [CC](#). [6](#)

Bootstrap Es una biblioteca de estilos para aplicaciones web definidos principalmente en lenguaje SCSS. [13](#)

Bot Es un programa de computador o script que realiza tareas periódicas de recopilación de datos generalmente sobre Internet y la web. [6](#)

Cloud Computing Provider Empresa que presta servicios de Computación en la nube. [15](#)

Cloud Computing El CC es una tendencia de servicios y recursos que funcionan sobre Internet para facilitar el despliegue, uso y disponibilidad de aplicaciones de Software. [1](#), [15](#)

CoreUI Es una biblioteca de React que ofrece una versión de uso libre de componentes gráficos escritos en Javascript y TypeScript y apoyada en en los estilos de [Bootstrap](#). [6](#)

Digital Ocean Es un proveedor de servicios de [CC](#) que proviene de New York fundada en el año 2011. [6](#)

Docker . [8](#), [13](#)

Docker Compose Es una tecnología para la automatización y orquestación de aplicaciones a partir de contenedores de [Docker](#). [5](#)

Docker Hub . [5](#)

Endpoint Comunmente las rutas en un servicio web que responden a las peticiones que le son solicitadas. [ii](#), [4](#)

Google Cloud Es una empresa de [CC](#) fundada por Google en el 2008. [6](#)

Host Dime Esta empresa tiene presencia de infraestructura de servicios de [CC](#) en Colombia. [6](#)

Hosting RED Es una empresa de [CC](#) con centros de datos ubicados en Bogotá Colombia además de Estados Unidos y Canadá. [6](#)

- Hypertext Media Transfer Protocol Secure** El protocolo que usa la web para enviar y recibir tráfico encriptado a través de una capa de sockets seguros. [15](#)
- IBM Cloud** Es el servicio de [CC](#) prestado por la empresa IBM. [6](#)
- Let's Encrypt** Es una entidad certificadora gratuita de lucro para expedición de certificados SSL/TLS. [5](#)
- LibCloud** Es una de las bibliotecas escrita en lenguaje Python mas completa hasta el momento para interactuar con los servicios de los [CCSPs](#). [ii](#), [1](#), [4](#), [6](#)
- React** Es una biblioteca de Javascript para la construcción de aplicaciones web reactivas. [6](#)
- Representational State Transfer API** Es una interfaz de programación de aplicaciones que permite la comunicación a través de la web usando el protocolo HTTP. [15](#)
- User Interface** Referente al diseño que tienen las interfaces de un proyecto de software o tecnología. [15](#)
- XPath** Es un lenguaje de programación para formular patrones que permiten encontrar coincidencias en documentos de marcado de etiquetas. [4](#)

AES Advanced Encryption Standard. 5
API Application Programming Interface. ii, 4, 7, 8
API REST [Representational State Transfer API](#). 4
AWS [Amazon Web Services](#). 6, 9

CaaS Container as a Service. 8
CC [Cloud Computing](#). ii, 1, 4, 7, 8, 13, 14
CCSP [Cloud Computing Provider](#). ii, 1, 3, 4, 5, 6, 7, 8, 9, 10, 14
CDN Content Delivery Network. 7
CROF Cloud Resource Orchestration Framework. 7

DBaaS Data Base as a Service. 7
DNS Domain Name Server. 7

EC2 Amazon Elastic Compute Cloud. 8, 9

FaaS Function as a Service. 9

HTTPS [Hypertext Media Transfer Protocol Secure](#). 5

IaaS Infrastructure as a Service. 8, 9

NIST National Institute of Standards and Technology. 8

PaaS Platform as a Service. 8, 9

SaaS Software as a Service. 9

UI [User Interface](#). 6
UX User Experience. 6

VPS Virtual Private Server. 8

BIBLIOGRAFÍA

- [1] May Al-Roomi, Shaikha Al-Ebrahim, Sabika Buqrais e Imtiaz Ahmad. «Cloud computing pricing models: a survey». En: *International Journal of Grid and Distributed Computing* 6.5 (2013), págs. 93-106.
- [2] Ioana Baldini y col. «Serverless Computing: Current Trends and Open Problems». En: *Research Advances in Cloud Computing*. Ed. por Sanjay Chaudhary, Gaurav Somani y Rajkumar Buyya. Singapore: Springer Singapore, 2017, págs. 1-20. ISBN: 978-981-10-5026-8. DOI: [10.1007/978-981-10-5026-8_1](https://doi.org/10.1007/978-981-10-5026-8_1). URL: https://doi.org/10.1007/978-981-10-5026-8_1.
- [3] Vidyanand Choudhary. «Software as a service: Implications for investment in software development». En: *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*. IEEE. 2007, 209a-209a.
- [4] George Lawton. «Developing software online with platform-as-a-service technology». En: *Computer* 41.6 (2008), págs. 13-15.
- [5] Fang Liu, Jin Tong, Jian Mao, Robert Bohn, John Messina, Lee Badger, Dawn Leaf y col. «NIST cloud computing reference architecture». En: *NIST special publication* 500.2011 (2011), págs. 1-28.
- [6] Theo Lynn, Pierangelo Rosati, Arnaud Lejeune y Vincent Emeakaroha. «A preliminary review of enterprise serverless cloud computing (function-as-a-service) platforms». En: *2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE. 2017, págs. 162-169.
- [7] Ayob Sether. «Cloud computing benefits». En: *Available at SSRN* 2781593 (2016).
- [8] Orazio Tomarchio, Domenico Calcaterra y Giuseppe Di Modica. «Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks». En: *Journal of Cloud Computing* 9.1 (2020), págs. 1-24.