UNIVERSIDAD POLITÉCNICA DE MADRID
Escuela Técnica Superior de Ingenieros Informáticos

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Máster Universitario en Software y Sistemas

Desarrollo de Software Sobre Entornos PaaS: Revisión General, Nuevas Perspectivas y Desafíos para la Ingeniería de Software

Trabajo Final de Máster

Ismael Camargo-Henríquez

Madrid, 2015

A Carmen. Te fuiste lejos y sólo tan lejos, estarás más cerca de mí.

Agradecimientos

En primer lugar, agradezco a Dios por día a día regalarme en su inmerecida misericordia, vida y salud para poder hacer cosas buenas para su honra y gloria. ¡Gracias Padre!

Quisiera agradecer a mi familia y en especial a mis papás: *"Maelo"* y *"Bacy"* por la paciencia que han tenido en el tiempo que he pasado lejos de ellos, lejos de casa. *Son mis piernas si voy lejos, es porque camino con ustedes*.

Por incondicionalmente creer en mi valor humano y profesional, por ser un maestro, mi otro padre y siempre un guía ejemplar, agradezco al profesor, Ph.D *Miguel Vargas-Lombardo*.

Agradezco también, al profesor Ph.D *Andrés Silva Vázquez*, por su entusiasta apoyo y orientación durante la preparación de este trabajo y que me ha permitido llevar el proceso de desarrollo e investigación de una manera concisa y objetiva.

Finalmente, me gustaría dar las gracias a los compañeros de clases por su amistad y ánimos durante el curso, haciendo con su compañía y aprecio, más amena la experiencia. A todos ustedes, ¡muchas gracias!

Resumen

El uso de la computación en la nube ofrece un nuevo paradigma que procura proporcionar servicios informáticos para los cuales no es necesario contar con grandes infraestructuras y sobre todo, con las complejidades de costos, seguridad y mantenimiento implícitas.

Si bien se ha posicionado en los últimos años como una plataforma innovadora en el ámbito de la tecnología de consumo masivo y organizacional, también puede ser tópico de investigación importante en ciertas áreas de interés como el desarrollo de Software, presentando en ese campo, una serie de ventajas y retos estimulantes que pueden ser explorados.

Este trabajo de investigación, sigue con dicho sentido, el objetivo de exponer la situación actual sobre el empleo de la computación en la nube como entorno de desarrollo de Software, sectorizando a través de su capa PaaS, el modelo conceptual de trabajo, las perspectivas recientes, problemas e implicaciones generales del uso de ésta como herramienta plausible en proyectos de desarrollo de Software.

El análisis de los diferentes temas abordados, tiene la intención en general, de proporcionar información objetiva, crítica y cuantitativa sobre la concentración de la investigación relacionada a PaaS, así como un marco de interpretación reciente que aporte una perspectiva referencial para futuras investigaciones asociadas.

Abstract

The use of cloud computing offers a new paradigm to provide computer services for which it is not necessary to have large infrastructure and especially with the complexities of cost, safety and maintenance implied.

While it has positioned itself in recent years as an innovative platform in the field of technology and massive organizational consumption, can also be an important research topic in certain areas of interest including, the development of Software, presenting in this field, a series of advantages, disadvantages and stimulating challenges that can be explored.

This research, following with that sense, try to present the current situation related to the use of cloud computing as a software development environment, through its sectorized PaaS layer, showing the conceptual working model, actual perspectives, problems and general implications of using this as a possible tool in Software development projects.

The analysis of the different topics covered, intends in a general form, provide objective, critical and quantitative information about the concentration of research related to PaaS, and a recent interpretation framework to provide a referential perspective for future related researches.

Contenido

Índice de Figuras

Índice de Gráficos

CAPÍTULO – I

Introducción

Este capítulo presenta el contexto de la temática abordada, la concepción investigativa que la motiva bien como las directrices u objetivos que se pretenden alcanzar. Adicionalmente, se expone de manera general, el contenido a desarrollar con el fin de enfocar y delimitar el dominio del esfuerzo de investigación.

1.1. Motivación

La computación en la nube o *Cloud Computing*, es un nuevo paradigma o modelo de computación (Buyya, Broberg, & Goscinski, 2010) que permite al usuario final acceder a una gran cantidad de aplicaciones y servicios en cualquier lugar y con independencia de la plataforma.

La palabra *nube* sugiere la idea de un ambiente desconocido, difuso, ubicuo y distante (Blacharski & Landis, 2013). Por esta razón, su empleo es muy bien utilizado en el nombramiento del modelo conceptual de cómputo que representa, en el cual toda la infraestructura y los recursos computacionales permanecen *"recónditos"* y el acceso de los usuarios se realiza utilizando una interfaz estándar a través de la cual, todo el conjunto de aplicaciones y servicios está disponible con independencia de su localización.

En esencia, la nube está representada por Internet, es decir, se apoya en la infraestructura de comunicación que compone el conjunto de Hardware, Software, redes de telecomunicaciones, dispositivos de almacenamiento, acceso y control que permiten la entrega de la información como un servicio (Armbrust, Fox, Griffith, Joseph, & RH, 2009).

Por otra parte, ha sido notable que durante la última década, la computación en la nube ha irrumpido con la expectativa de hacer cambiar muchos aspectos de la forma en que se adquiere, se gestiona y se utiliza la tecnología (Marston, Li, Bandyopadhyay, Zhang, & Ghalsasi, 2011) (Phaphoom, Wang, Samuel, Helmer, & Abrahamsson, 2015).

En este sentido, es de interés observar como el paradigma que propone, ha impactado casi todas las facetas de aplicaciones de TI (Tecnología de Información) para Ventas, Marketing, Finanzas y otras muchas (Columbus, 2014) (Marston et al., 2011), siendo un hecho frecuente el rediseño y la adaptación de éstas para aprovechar el acceso inmediato a ella.

De igual modo y casi en una tendencia de ascenso (Columbus, 2014), es apreciable la manera por la cual la computación en la nube está cambiando la forma como las aplicaciones están siendo diseñadas, probadas y desplegadas (Armbrust et al., 2009)(Bahsoon, Mistrík, Ali, Mohan, & Medvidović, 2013), lo que resulta en un cambio significativo en las prioridades del desarrollo de aplicaciones tal y como se han realizado hasta ahora.

Si bien, la reducción en costos (Armbrust et al., 2009) sigue siendo el motor más importante que acelera esta tendencia, también lo son exigencias de tipo técnicas como la agilidad, la flexibilidad, la seguridad y la velocidad para extender nuevas aplicaciones para este entorno.

Consecuentemente, su influencia también afecta las herramientas y soluciones de soporte que impulsan el desenvolvimiento de Software puesto que, con las posibilidades ofrecidas por el paradigma computacional de la nube a través de su capa PaaS (*Platform as a Service*) (Mell & Grance, 2011) y el enriquecimiento de las tecnologías Web, se abre una gama de alternativas que prometen ser la evolución en el desarrollo de Software.

De este modo, la propuesta con los entornos de desarrollo en la nube es permitir entre otras ventajas, que los desarrolladores no se preocupen por la configuración del entorno y otros detalles locales y puedan escribir código en cualquier momento y desde cualquier lugar.

En ese sentido, el presente esfuerzo de investigación surge de la inquietud por explorar en profundidad esta nueva generación de herramientas de soporte al desarrollo desde una perspectiva que describa su posicionamiento tecnológico y táctico en el mercado actual.

Para esto, inicialmente se evalúan los componentes técnicos que componen típicamente la nube, en particular, enfatizando el análisis de la capa PaaS que provee el modelo estructural (Buyya et al., 2010) para el desarrollo de Software en la nube.

Igualmente, se revisan los aspectos comunes de este entorno de desarrollo como su arquitectura de funcionamiento en relación a los modelos de herramientas para escritorio, examinando sus ventajas y particularmente, las desventajas documentadas en la literatura consultada.

El énfasis general, se realizará sobre los potenciales problemas que pudieran existir en el cambio de paradigma tanto a nivel técnico y humano como organizacional. Para esto, se procura brindar consideraciones en tendencias documentadas por estudios recientes, que permitan conducir a la identificación de criterios sobre la adopción real de la nube como entorno de desarrollo.

De lo anterior, se espera ofrecer como resultado final, un compendio apreciable de generalidades que permitan ilustrar de manera objetiva, si esta nueva evolución en herramientas de desarrollo y el aprovisionamiento de servicios de producción de Software en la nube a través de la capa PaaS, poseen un nivel de madurez, confianza, seguridad y rentabilidad, que los convierta realmente en una propuesta atractiva y seria de desenvolvimiento de proyectos de Software en los próximos años.

1.2. Objetivos

De manera descriptiva la presente labor de investigación se circunda a lograr los siguientes objetivos de trabajo:

Objetivo General

Exponer la situación actual sobre el desarrollo de Software en la nube desde la perspectiva de la adopción y uso de herramientas basadas en la capa PaaS.

Objetivos Específicos

1. Revisar de manera sistemática y general, los componentes de la arquitectura y funcionamiento del modelo de computación en la nube.

2. Examinar en profundidad la capa PaaS como modelo de aprovisionamiento de servicios de soporte en el desarrollo de Software en la nube.

3. Identificar las desventajas y problemas potenciales que pueden existir en la adopción de la capa PaaS como modelo para herramientas de desarrollo de Software bajo factores técnicos y organizacionales.

4. Analizar las funcionalidades genéricas de operación de la capa PaaS en las tareas de desarrollo de Software.

5. Describir las ventajas ofrecidas por el modelo de desarrollo basado en herramientas de PaaS en aspectos técnicos y organizacionales.

6. Proponer líneas de acción objetivas en la adopción de herramientas de desarrollo basadas en PaaS.

7. Establecer líneas de continuidad en el trabajo de investigación, sobre uno o más temas específicos abordados.

1.3. Contenido del Trabajo

El presente trabajo de investigación pretende subscribirse a la recopilación, análisis y discusión de diferentes estudios técnicos realizados sobre los temas a tratar, a partir de literatura especializada con fuentes científicas, libros, borradores técnicos de especificaciones y manuales.

Considerando la abundante literatura sobre el tópico de investigación y el amplio conjunto de temáticas específicas derivadas de diferentes enfoques y autores relativos al mismo, la metodología de segmentación de las fuentes para la organización del material a desarrollar, ha procurado seguir el criterio de una revisión sistémica a aquellas más recientes, evaluándose en este sentido, publicaciones entre los períodos de 2007 a 2015.

Igualmente, la colección bibliográfica seleccionada, ha reunido estudios y ponencias diversas, con el fin de proveer distintos puntos de opinión y discusión sin ser en este sentido, tendenciosos a favorecer alguna propuesta en particular, relativa a autores, fabricantes y proveedores citados.

De esta manera, se potencia como objeto particular de estudio, exponer y describir la situación actual y futura de la capa PaaS como la infraestructura de desarrollo de Software en la nube, destacando sus aspectos funcionales.

El aporte primordial es trazar una línea de criterios que permitan la evaluación objetiva de este tópico con miras a ofrecer un enfoque práctico, fresco y reciente sobre ésta temática.

Así, la organización propuesta para el contenido a desarrollar, se ha concentrado en ofrecer cuatro secciones o **Capítulos** y una sección de **Conclusiones** y **Trabajos Futuros** las cuales se estructuran en el siguiente orden temático general.

- **Capítulo I: Introducción**
- **Capítulo II: Estado del Arte**
- **Capítulo III: Modelo de Desarrollo de Software sobre PaaS**
- **Capítulo IV: Problemas Abiertos**
- **Conclusiones**
- **Trabajos Futuros**

A su vez, cada sección ha sido estructurada en apartados específicos relacionados al tema principal enunciado por cada Capítulo, esto con el objetivo de darle profundidad y fluidez a los temas abarcados y que se han considerado de interés en el desarrollo de este esfuerzo de investigación.

De esta manera, además del **Capítulo I** de *Introducción*, este trabajo de investigación consta de otros tres capítulos; describiéndose brevemente el contenido de cada uno de ellos a continuación.

En el **Capítulo II**, se presenta el *Estado del Arte* actual sobre la temática de estudio; se realiza para ello, una revisión descriptiva a la evolución histórica del paradigma computacional de la nube como base inicial, ampliándose hacia su concepto general, su arquitectura típica de referencia y un abordaje al enfoque del desarrollo de Software sobre ésta; el capítulo permite situar el marco conceptual para las siguientes secciones del trabajo.

Notable en el desarrollo de esta sección, es el direccionamiento estadístico actual sobre las tendencias en el uso y adopción del modelo de desarrollo propuesto por la capa PaaS, las cuales permiten establecer ciertos criterios relativos al cambio en la manera de abordar las metodologías y formas conocidas de construir Software.

Como desarrollo del **Capítulo III**, se formaliza una completa exploración a los tópicos que competen específicamente al *Modelo de Desarrollo sobre la capa PaaS*, examinando sus propiedades y características como entorno de desenvolvimiento de Software en la nube, su constitución arquetípica de funcionamiento la cual se promueve a aspectos relacionados con arquitectura y acceso.

Igualmente, se ofrecen de manera descriptiva en este capítulo, las ventajas ofrecidas por el modelo de desarrollo de Software sobre la capa PaaS, con base en las observaciones aportadas por otros estudios y literaturas especializadas.

Es de particular interés en este capítulo, las cuestiones que conciernen al potencial del modelo como entorno de desarrollo de Software, bien como las consideraciones de implementación en proyectos de Ingeniería de Software que deben ser evaluadas en su adopción.

En el **Capítulo IV**, se señalan algunos de los *Problemas Abiertos* que se extraen de esta investigación en relación al modelo de desarrollo sobre PaaS.

Se revisan sus particularidades y consideraciones al respecto, desde la base de un análisis objetivo que permita discutir críticamente el modelo como una tendencia de alternativa técnica en el desarrollo de Software para los próximos años.

Finalmente, como sección de cierre al presente trabajo, se proveen las **Conclusiones** y consideraciones finales derivadas del estudio en completitud con los objetivos propuestos y con la visión de poder brindar un aporte fresco a esta temática investigativa.

Como un espacio abierto a la continuidad de la investigación, en el tópico **Trabajos Futuros**, de manera general, se bosquejan algunas ideas sobre proyectos de trabajos *a posteriori* y que podrían ser fuente de pesquisas concretas sobre aspectos de interés como la *interoperabilidad* de aplicaciones.

CAPÍTULO – II

Estado del Arte

Se presenta en esta sección del trabajo, la situación actual de la cuestión investigada, desde la perspectiva histórica del concepto de computación en la nube, ampliando hacia su definición formal, estructura general de funcionamiento y el ciclo de desarrollo de Software sobre su contexto.

2.1. Evolución Histórica

La concepción de computación en la nube, como tantas otras de las innovaciones basadas en tecnología de la información y comunicación, es resultado de una evolución continua y mejorada de otros conceptos computacionales que han existido a lo largo de la historia (Teixeira, Pinto, Azevedo, Batista, & Monteiro, 2014) como por ejemplo: la Computación de Alto Rendimiento (High Performance Computing) (Furht et al., 2010), el concepto de Virtualización o el desarrollo de servicios Web (Marston et al., 2011). Así, la aparición de la computación en la nube *per se*, está estrechamente ligada a la madurez de este tipo de tecnologías.

En ese sentido, Buyya et al., (Buyya et al., 2010) señalan que el origen de la computación en la nube, se relaciona más específicamente con la evolución del Hardware. Concretamente la convergencia de la virtualización y los procesadores multinúcleo, complementada por avances en otras áreas como las tecnologías de Internet orientadas a los servicios Web, las arquitecturas basadas en servicios como SOA y la Web 2.0.

A esta base, se acoplan además, los fundamentos funcionales de la computación distribuida (GRIDS, Clústeres) y los sistemas de gestión automáticos como por ejemplo, la computación autonómica (Buyya et al., 2010).

Buyya et al., (Buyya et al., 2010) destacan también, que estas tecnologías fueron notorias individualmente como prometedoras en sus primeras etapas de desarrollo. Sin embargo, la atención significativa llegó de la investigación académica, al inquirir y mejorar las posibilidades de aplicación conjunta, con la posterior implementación, por parte de los grandes líderes de la industria tecnológica, por medio de procesos de especificación, estandarización y uso; conduciendo hacia la madurez y amplia adopción de éstas.

Según la descripción evolutiva propuesta por Buyya et al., (Buyya et al., 2010), la convergencia de las tecnología que conforman la nube, puede ser ilustrada según la siguiente figura.

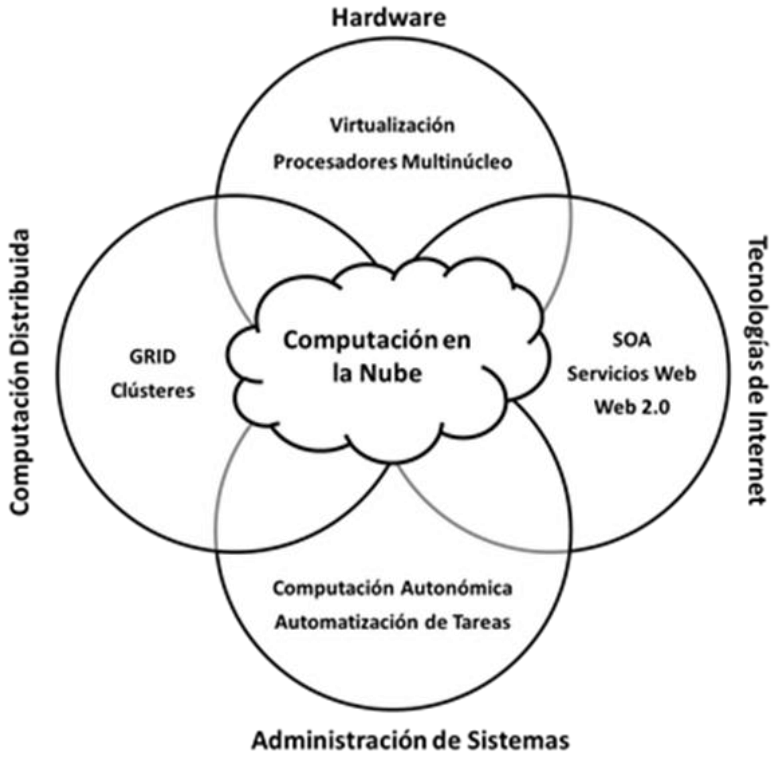

Figura 1. Esquema conceptual de la convergencia tecnológica para el modelo computacional de la nube (Basado en Buyya et al., 2010).

Otro enfoque suplementario de la evolución del concepto de computación en la nube, es el ofrecido por el informe Bankinter, en el cuál se expone en términos generales, que este modelo computacional no es sino un concepto consolidado de dos modelos previos: la Computación Centralizada y la Computación Cliente-Servidor (Bankinter, 2010).

Sobre el modelo de Computación Centralizada, el informe Bankinter, lo reseña hacia las primeras décadas de utilización masiva de los sistemas de cómputo, cifrando entre los años 50 a 70 su uso más asiduo (Bankinter, 2010).

De este modo, el modelo constituía de grandes computadores u ordenadores (Mainframes) concentrados en centros de datos específicos, los cuales eran compartidos por múltiples usuarios dado su alto costo. Una derivación menos onerosa de este modelo, se desarrolló mediante el uso de Minicomputadores o Miniordenadores con tareas muy específicas dentro de departamentos en pequeñas y grandes organizaciones, no obstante, la centralización era su principal característica.

Con el advenimiento de avances tanto en el aumento de la capacidad de cómputo bien como en la reducción física del tamaño de los procesadores alrededor de los años 80, el modelo de Computación Centralizada, evolucionó hacia un paradigma más flexible, eficiente y barato, basado en estaciones de trabajo interconectadas, originando de este modo, el paradigma de Cliente-Servidor (Bankinter, 2010).

Este paradigma en particular, aprovechaba como mejor cualidad, la optimización de funciones, la reducción de costos y potencialmente cierta escalabilidad, debido a que sus componentes *clientes*, eran baratos y podían ser adicionados tantos como fueran requeridos. Por su parte, los componentes *servidores* podían ser destinados en modo dedicado a una sola aplicación de Software y con relativa independencia física y lógica.

No obstante, a pesar de sus bondades técnicas, el modelo Cliente-Servidor presentaba el inconveniente de muchas veces, llegar a fragmentar demasiado la especialización de las aplicaciones y sistemas del lado servidor, estimándose que, alrededor de un 10% a 20% de las capacidades de procesamiento de éstos eran utilizadas, infrautilizándose el resto (Bankinter, 2010).

Otro inconveniente de la fragmentación, conducida por la *praxis* inadecuada del paradigma, fue el aumento en la complejidad de la gestión de los recursos y el aumento en costos, sobre operaciones de mantenimiento de grandes infraestructuras basadas en este modelo.

Sin embargo, es interesante notar que la evidencia demostrada del modelo Cliente-Servidor en términos de proveer independencia y una distribución de procesos, permitió el soporte de las primeras aplicaciones Web alrededor de los años 90.

Sobre lo anterior, el informe Bankinter expone que la demanda cada vez más creciente en servicios de correos, noticias, compras, búsquedas y otros; ofrecidos por los primeros grandes portales y posteriormente, el surgimiento de una industria basada en Internet y los conceptos de Web 2.0, como blogs, wikis, redes sociales, multimedia en línea, etc., abrió el compás hacia la requisición de servidores y sistemas más fiables, potentes y escalables.

Toda vez que esta nueva variedad y disponibilidad de servicios alcanzó un crecimiento cuantitativo muy crítico, el paradigma Cliente-Servidor fue provisto de nuevas caracterizaciones técnicas para hacerlo más escalable y fácil de mantener.

Sin embargo, señalan en Bankinter: *"una señal inequívoca de que el modelo cliente-servidor se estaba quedando sin aliento fue el incremento explosivo de los dispositivos móviles en los últimos años, así como la perspectiva de un número aún más grande de sensores y otras tecnologías digitales, cada una con su propia dirección IP, que empezaban a estar alojados en miles de objetos, como los electrodomésticos, los coches, las carreteras, las conducciones de gas o petróleo y los productos farmacéuticos."* (Bankinter, 2010).

De esta manera, el contexto de una escalabilidad masiva para múltiples *clientes* y servicios, precisó la creación de un nuevo paradigma capaz de optimizarse no entorno a computadores individuales, sino a Internet como una entidad; este tercer modelo computacional, surgido finalmente durante la última década, es el modelo de Computación en la Nube.

De manera ilustrativa lo anteriormente descrito, puede sintetizarse en la cronología representada por la figura a continuación.

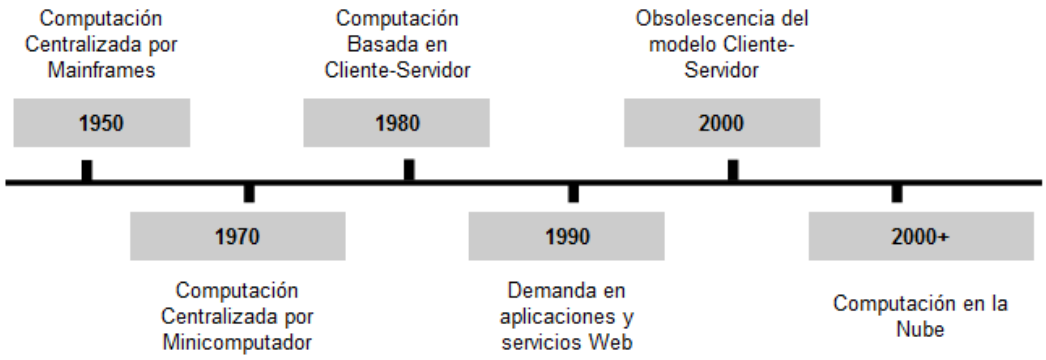

Figura 2. Cronología histórica de la evolución del modelo de computación en la nube según el informe Bankinter (Bankinter, 2010).

Una tercera visión evolutiva del modelo además de las anteriormente referenciadas, es la propuesta por Furht et al., la cual plantea que la progresión histórica del modelo computacional en la nube, es derivada de evoluciones en cinco etapas o paradigmas previos; desde terminales tontas y Mainframes, pasando por computadores personales, redes de computación y GRID, hasta finalmente llegar al paradigma de computación en la nube (Furht et al., 2010).

En la primera etapa, de manera similar a la propuesta del informe Bankinter, Furht et al., señalan la utilización de un modelo en el cual muchos usuarios comparten Mainframes centrales que utilizan terminales tontas para acceder a los procesos.

En una segunda etapa, las computadoras personales se hicieron lo suficientemente potentes como para satisfacer la mayoría de las necesidades de los usuarios en términos de procesamiento; probablemente como señalan en Buyya et al., por mejoras sustanciales al hardware, principalmente en la potencia de los procesadores (Buyya et al., 2010).

En la etapa tres de la evolución según Furht et al., las computadoras personales, los portátiles y servidores se conectan entre sí a través de redes locales para compartir recursos y aumentar el rendimiento notándose aquí, las primeras tendencias a distribuir y acceder a servicios de procesamiento remoto compartido (Furht et al., 2010). Esto último de manera comparativa, muy similar al modelo Cliente-Servidor identificado por el informe Bankinter.

En la cuarta etapa, las redes locales empiezan a interconectarse a otras redes formando conglomerados extensos geográficamente tal como hoy día lo hace el Internet, para utilizar las aplicaciones y los recursos de manera ubicua.

A modo de una quinta etapa, Furht et al., indican el surgimiento del concepto de computación mediante GRID, el cual proporciona potencia de cálculo compartida y aporta almacenamiento a través de un sistema de computación distribuida (Furht et al., 2010).

Finalmente, como conjunción final de los modelos anteriores surge la computación en la nube, que proporciona acceso a recursos compartidos en Internet de una manera escalable y simple, ofreciendo potencia de cálculo compartida, distribución ubicua e independencia geográfica, acceso virtualmente ilimitado y finalmente, transparencia sobre los dispositivos que acceden a los servicios que brinda. Lo anterior, se refleja ilustrado de manera general en la **Figura 3**.

Una acotación importante realizada por Furht et al., es señalar que la comparación de los seis paradigmas computacionales que han existido, podría sugerir que la computación en la nube es un regreso al modelo centralizado en Mainframes. Sin embargo, estos dos paradigmas tienen varias diferencias importantes.

El modelo basado en Mainframe (computación centralizada para Bankinter), ofrece finita y reducida potencia de cálculo, mientras que la computación en la nube proporciona *energía casi infinita* y *capacidad ilimitada* de procesamiento (Furht et al., 2010).

Por otra parte, en Furht et al., hace énfasis en señalar que el modelo Mainframe se vale del uso de terminales tontas que actúan como interfaz de usuario, por cuanto que en la computación basada en la nube, dichos terminales pueden ser computadores personales o cualquier otro dispositivo con conectividad a Internet.

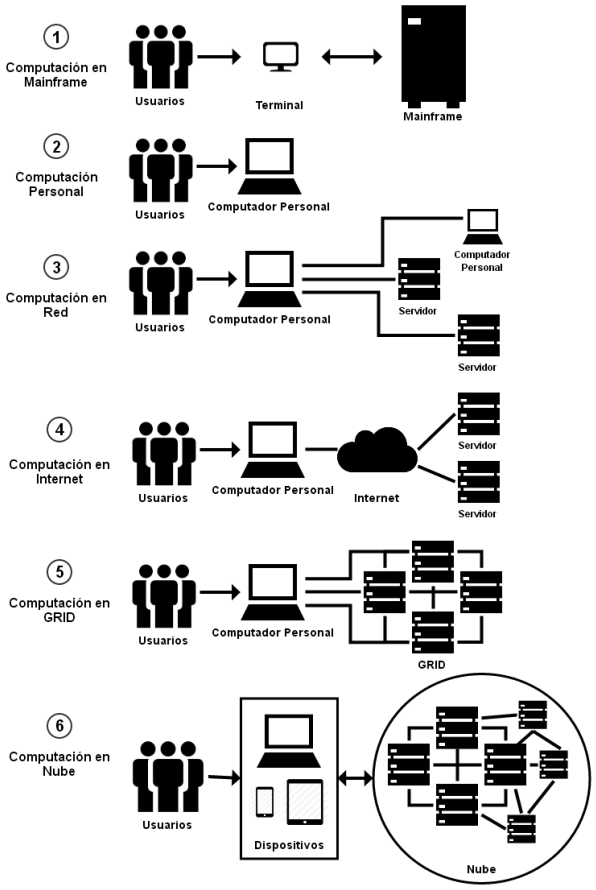

Figura 3. Visión evolutiva de los modelos computacionales según Furht et al. (Basado en Furht et al., 2010).

Como conducción final en esta sección del trabajo, se puede establecer que en las literaturas consultadas, persiste la evidente tendencia a considerar que efectivamente el modelo de computación en la nube, es una innovación producto de la evolución e integración de distintas tecnologías previas.

Si bien el enfoque de Buyya et al., providencialmente señala los aspectos más relevantes en términos tecnológicos, no produce un conflicto con la propuesta histórica de evolución del informe Bankinter.

A su vez, la versión ofrecida por Furht et al., aunque no sitúa en una línea temporal las etapas o paradigmas que plantea, enriquece los aportes anteriores pudiendo ser considerados complementarios a los objetivos a lograr en este punto del desarrollo investigativo, que es proveer una base general del origen del modelo y abordar su concepción formal en las siguientes secciones.

2.2. El Paradigma de la Computación en la Nube

Descrito en la sección anterior el contexto de origen e historia del modelo de computación en la nube, se expone en éste tópico el abordaje general a los aspectos conceptuales de este paradigma.

Inicialmente, es de considerar que el término paradigma o modelo, remite a una representación abstracta que expresa una serie de componentes y operaciones en su gran mayoría de suma complejidad. Al referirse al término *paradigma computacional en la nube*, por tanto, se aborda una naturaleza con connotaciones puramente genéricas en las cuáles el ámbito de implementación permanece invisible desde la perspectiva de quien hace uso del modelo.

En ese sentido, por ejemplo, autores como Blacharski y Landis (Blacharski & Landis, 2013) ofrecen ciertas caracterizaciones importantes a considerar sobre la palabra *nube* que forman una idea bastante apropiada del porqué el término adquiere una propiedad difusa.

Así, destacan que en primer lugar, la *computación en la nube es un término extremadamente amplio* (Blacharski & Landis, 2013) el cual puede abarcar una multiplicidad de elementos que consiguen ir desde la tecnología implementada para su soporte, tales como el Hardware, elementos de red y comunicación, almacenamiento, gestión de control y acceso, Software, etc., hasta aspectos menos intrínsecos, como los servicios y aplicaciones de uso, de los cuales el beneficiario queda exento de conocer detalles.

En esta misma publicación, los autores también asocian la palabra nube computacional o computación en la nube, con la característica de *intangibilidad*, al indicar que si bien, la *nube* es real, ésta se abstrae al punto donde no se puede ver que es como tal, por lo que se hace difícil de imaginar (Blacharski & Landis, 2013).

Como tercera y última propiedad, Blacharski y Landis indican que el término se ha popularizado a modo de expresar la relativa *simplicidad* con la que se puede publicitar el modelo por parte de las organizaciones y empresas enfocadas en el área de TI, creando una percepción ubicua, remota, transparente y poderosa del concepto sin prescribir detalles agobiantes al cliente, lo que puede ser contraproducente, al crear la expectativa que todo procesamiento remoto, es realizado en la nube.

El argumento anterior, es plausible dado que el modismo del término y su relativa novedad, bien como la forma en que ha sido promovido entre usuarios, lleva a creer erróneamente que la computación en la nube no es más que Internet con un nombre diferente.

Una contribución adicional a la confusión sobre el concepto del paradigma, ha sido que tradicionalmente, muchos dibujos representativos de arquitecturas de sistemas y servicios basados en Internet, simbolizan ésta última, como una nube (Velte, Velte, & Elsenpeter, 2010) y los autores se refieren a las aplicaciones que se ejecutan en Internet como: *"ejecutadas en la nube"*, por lo que la confusión es comprensible (Sosinsky, 2011).

Sobre esto, es significativo notar que Internet tiene muchas de las características del paradigma computacional en la nube pues ofrece abstracción, se ejecuta utilizando el mismo conjunto de protocolos y normas de comunicación y emplea la misma infraestructura de aplicaciones y sistemas operativos (Sosinsky, 2011). Por ende, al representarla se simplifican los detalles de su conjunto y no es posible diferenciar entre los sistemas físicos individuales, las redes y servicios que la componen, por lo que puede crear el estigma de que es sinónimo de *nube*.

Para efectos del presente estudio, se realiza la interpretación del concepto de computación en la nube, en concordancia con la proposición aportada por el autor Rajkumar Buyya y que expone que el paradigma computacional de la nube *"denota un modelo en el cual una infraestructura informática es vista como una 'nube', en la cual las empresas y las personas acceden a las aplicaciones desde cualquier lugar del mundo bajo demanda. El principio fundamental detrás de este modelo, es ofrecer la capacidad de computación, almacenamiento y software 'como un servicio'"* (Buyya et al., 2010).

Esta caracterización del concepto es relevante, debido a que concretiza efectivamente los componentes y objetivos del modelo más allá del empleo metafórico del nombre.

En este sentido, la exposición de Buyya et al., es específica, puesto que el paradigma computacional es en esencia, una infraestructura informática con toda la cabalidad de Hardware, Software y Redes que la componen. Diseñada, implementada e integrada para ofrecer algún tipo de servicio, de manera independiente a la geografía y ajustado a la requisición de quien lo solicita.

Importante es también recalcar el empleo del término *"servicio"* como el *postremum hoc* o meta última a la cual el modelo computacional en la nube procura atender.

Posteriormente se ampliará ésta palabra de acuerdo a la terminología conceptual apropiada no obstante, es de conocerse previamente que un *servicio* es el contexto funcional sobre el cual el modelo de computación en la nube fundamenta su propuesta.

2.2.1. Concepto General y Definición

Como anteriormente se expuso, la terminología del paradigma de computación en la nube, puede ser muy extensa en relación a las perspectivas con las que se desee abordarla, ya sea considerándola una tecnología incorpórea, fácil de popularizar y destinada a diferentes ámbitos de aplicación, o en su forma más extendida y generalmente expuesta; como una tendencia que potencia principalmente la reducción de costos en infraestructura tecnológica y mejora de procesos en las organizaciones por medio de servicios ofrecidos desde ésta.

En este sentido, muchos profesionales en los ámbitos comerciales y académicos han intentado definir exactamente *que es* el paradigma de computación en la nube y *qué características únicas presenta*.

Siguiendo este esfuerzo y en la línea de pensamiento anterior, el contexto investigativo del presente estudio, debe formalizar una definición apropiada en este aspecto. Para tal fin, se retoman las consideraciones conceptuales de diferentes fuentes a fin de establecer criterios generales sobre el concepto.

En primera instancia, se aborda una definición genérica ofrecida por Armbrust et al., quienes señalan que la computación en la nube: *"se refiere tanto a las aplicaciones entregadas como servicios a través de Internet así como el hardware y el software de sistemas en los centros de datos, que proporcionan esos servicios"* (Armbrust et al., 2009).

Otra concepción del paradigma es la ofrecida por Sotomayor et al., en la cual indican que la *"nube"* se utiliza más a menudo para referirse a la infraestructura de TI desplegada en un centro de datos del proveedor de servicio a fin de desarrollar una IaaS (Sotomayor, Montero, Llorente, & Foster, 2009).

Aportada por Buyya et al., se define el paradigma de la siguiente manera: *"La Nube es un sistema informático paralelo y distribuido que consiste en una colección de computadores interconectados y virtualizados los cuales son dinámicamente aprovisionados y presentados como uno o más recursos de computación unificados, basados en acuerdos de nivel de servicio (SLA) establecidos a través de la negociación entre el proveedor de los servicios y los consumidores"* (Buyya et al., 2010).

En relación a una definición oficial del concepto, la comisión del National Institute of Standards and Technology (NIST), provee la siguiente enunciación técnica del paradigma: *"La computación en nube es un modelo para la habilitación conveniente, de acceso por red y bajo demanda, a un conjunto compartido de recursos informáticos configurables (por ejemplo: redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser rápidamente aprovisionados y liberados con un esfuerzo mínimo de gestión o interacción con el proveedor de servicios. Este modelo de nube promueve la disponibilidad y se compone de cinco características esenciales, tres modelos de servicio y cuatro modelos de despliegue"* (Badger, Grance, Patt-Comer, & Voas, 2012).

En un ejercicio entendido como válido en el aporte a una definición del paradigma, se ofrece una noción propia, en la cual se concibe el modelo de computación en la nube como: - *un mecanismo dinámico para proporcionar un conjunto de recursos informáticos compartidos, los cuales incluyen aplicaciones, poder de cómputo, plataformas de almacenamiento, redes, infraestructuras de desarrollo y de implementación así como procesos de negocio* -.

Si bien existen un sin número de otras definiciones, el análisis de las citas utilizadas y la contribución propia que se ha realizado, proveen una idea bastante significativa del concepto sobre el paradigma de computación en la nube, que permite dilucidar algunas características importantes al estudio que se realiza.

En este sentido, parecen ser comunes a las fuentes citadas, que una nube debe tener las siguientes propiedades: capacidad de compartir recursos de Hardware y Software, pago por uso, elasticidad sobre la capacidad de los recursos (creando la ilusión de recursos infinitos), una interfaz de autoservicio con independencia del proveedor y recursos que se abstraen o virtualizan (Marston et al., 2011) (Blacharski & Landis, 2013) (Bahsoon et al., 2013).

Además de la capacidad de cómputo propiamente y el almacenamiento primario, se destaca que el modelo de computación en la nube por lo general ofrece una amplia gama de servicios de Software (Vazquez-Poletti, Moreno-Vozmediano, Montero, Huedo, & Llorente, 2013).

Sobre esto último, puede sugerirse que no solo se trata de aplicaciones pre-construidas y desplegadas para su uso inmediato, sino que también se incluyen las API (Application Programming Interface) o Interfaces de Programación de Aplicaciones y herramientas de desenvolvimiento, que permiten a los desarrolladores de Software crear aplicaciones escalables sin problemas, apoyados en los servicios ofrecidos desde la nube (Bahsoon et al., 2013) (Dillon, Wu, & Chang, 2010).

Con esta dirección, se ampliará en la sección siguiente, la integración de estos servicios en la arquitectura del paradigma, revisando la composición de sus capas o pilas bien como las conceptualizaciones iniciales que fundamentan el desarrollo de Software en la nube.

2.2.2. Arquitectura de Referencia y Componentes de Servicios

Un concepto integral tanto en la definición del paradigma de computación en la nube como en su operación e implementación, es la noción de *servicio*.

Como antes se ha mencionado, *servicio* es un término que está asociado prácticamente con toda referencia a la nube. De esta manera, en la terminología del modelo la expresión: *"como un servicio"* del inglés «*as-a-service*», de forma general se refiere a la capacidad de usar algo a través de Internet según se requiera (Velte et al., 2010).

Ampliando el término y su concepto, la frase *"as-a-service"* se utiliza ampliamente para significar que un producto de la nube (ya sea Infraestructura, Plataforma o Software) se ofrece de una manera tal que se puede *"alquilar"* por parte de los consumidores usando Internet (Blacharski & Landis, 2013).

En Blacharski y Landis se enuncia que "alquilado", conlleva el pago por el uso del servicio solicitado. A menudo también se describe como un servicio *"bajo demanda"*, porque está disponible siempre que se necesite (Blacharski & Landis, 2013).

Hay dos ventajas inmediatas con este concepto para el modelo computacional en la nube: primero, los costos tienden a ser sustancialmente menores para las organizaciones o consumidores que los solicitan; y segundo, ofrece un mayor nivel de escalabilidad pues los recursos pueden ajustarse a las necesidades del consumidor sin implicar en la subutilización de recursos (Blacharski & Landis, 2013).

En este sentido, es importante comprender esta noción para entender la organización de la arquitectura del modelo computacional. Profundizando un poco más, la delimitación de funcionalidades del paradigma en servicios, deriva de la normativa sugerida por el National Institute of Standards and Technology (NIST) la cual a su vez, retoma las conceptualizaciones del modelo SOA (Service Oriented Architecture).

Según el autor Barrie Sosinsky, una Arquitectura Orientada a Servicios (SOA) describe un método estándar para la solicitud de operaciones a través de componentes distribuidos y la gestión de los resultados que generan como respuesta. Debido a las distintas peticiones de los clientes, los componentes que proporcionan los servicios, los protocolos utilizados para entregar resultados y las respuestas en sí mismas, pueden tener una amplitud de variaciones (Sosinsky, 2011).

De esta forma, SOA viene a proporcionar una capa de traducción y gestión bajo una arquitectura que elimina la barrera implícita para que un cliente obtenga una respuesta a los servicios deseados.

Así, mediante SOA, los clientes y los componentes pueden ser escritos en diferentes lenguajes y pueden utilizar varios protocolos de mensajería y red para comunicarse entre ellos. Siendo que SOA proporciona las normas para el transporte de mensajes y provee la infraestructura para apoyar en lo posible la comunicación (Buyya, Yeo, Venugopal, Broberg, & Brandic, 2009).

SOA facilita además, acceso a servicios Web reutilizables a través de una red TCP/IP (Sosinsky, 2011), lo que hace de esta arquitectura una base importante que apoya el paradigma de computación en la nube y la consideración más relevante que se aplica en la presentación de la arquitectura de la nube computacional.

Retomando el enfoque del tópico en esta sección del trabajo, se expone el arquetipo estructural sobre el cual se desarrolla el paradigma de computación en la nube.

En relación a esto, se ha referenciado del modelo descriptivo aportado por el NIST (Mell & Grance, 2011) por ser considerado el estándar y el modelo más general de implementación, debido a que presenta un conjunto de definiciones que separan el paradigma de computación en la nube, tanto en modelos de servicios como en modelos de despliegue. Esos modelos y su relación con las características esenciales de la computación en la nube se muestran en la siguiente ilustración.

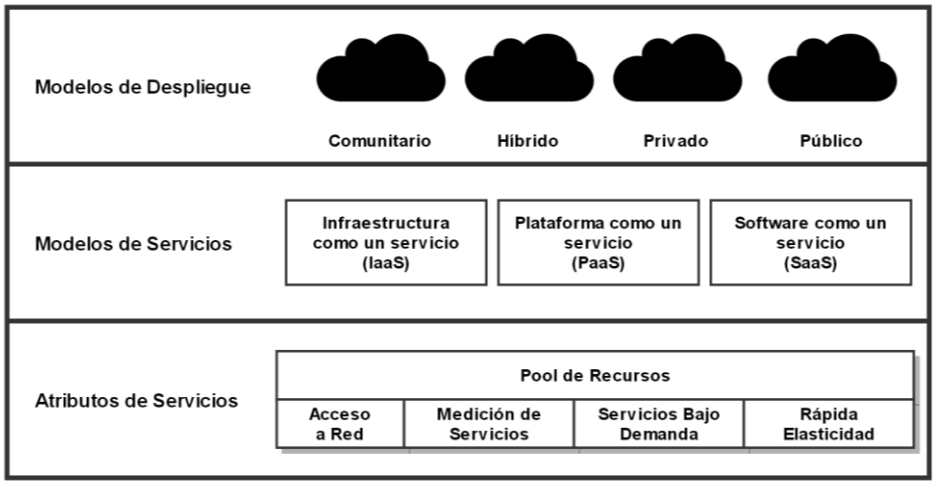

Figura 4. Modelo de arquitectura para el paradigma de computación en la nube según el NIST (Basado en Sosinsky, 2011).

Debido a que la computación en la nube está en continua tendencia a consolidarse hacia un conjunto de componentes que interactúan modularmente y basados en estándares como la Arquitectura Orientada a Servicios (SOA), es de esperar que en el futuro versiones del modelo NIST agreguen nuevas características (Sosinsky, 2011). No obstante, se describen brevemente las mostradas en el esquema anterior.

En el primer nivel, se estructuran los *Modelos de Despliegue*, que se refieren en manera general, a la ubicación y la forma de gestión de la infraestructura de la nube.

Sobre la organización de estos modelos, el NIST ofrece cuatro caracterizaciones apreciables (Mell & Grance, 2011):

- **Nube Comunitaria:** En este escenario, la infraestructura en la nube está preparada para el uso exclusivo de una comunidad específica de consumidores u organizaciones que han compartido intereses (por ejemplo: misión, requisitos de seguridad, políticas y consideraciones de cumplimiento). Puede ser propietaria, administrada u operada por una o más de las organizaciones de la comunidad, un tercero o alguna combinación de ellos y puede existir dentro o fuera de las instalaciones.

- **Nube Híbrida:** La infraestructura en la nube en este contexto, es una composición de dos o más infraestructuras distintas de nube (privadas, comunitarias o públicas) que aparentan ser entidades únicas, pero están unidas por la tecnología estandarizada o propietaria que permite la portabilidad de datos y aplicaciones.

- **Nube Privada:** Aquí, la infraestructura en la nube está preparada para el uso exclusivo de una sola organización que comprende múltiples consumidores internos (por ejemplo: unidades de negocio). Puede ser de propiedad, administrado y operado por la organización, un tercero o alguna combinación de ellos y puede existir dentro o fuera de las instalaciones de la organización.

- **Nube Pública:** La infraestructura en la nube está preparada para el uso abierto al público en general. Puede ser de propiedad, administrado y operado por una organización académica, empresa u organización gubernamental o una combinación de ellos. Existe en las instalaciones del proveedor de la nube.

Otro elemento relevante en la arquitectura expresada por el NIST, es el relacionado con los *Atributos de Servicios*, que conciernen a las características del modelo computacional que son esenciales para que los servicios que realmente representan la operatividad del paradigma, satisfagan las expectativas de los consumidores (Buyya et al., 2010).

En este sentido, el modelo arquitectónico propuesto por el NIST señala las siguientes (Mell & Grance, 2011):

- **Pool de Recursos:** Se refiere a reunir los recursos informáticos del proveedor para servir a múltiples consumidores mediante un modelo multiusuario, con diferentes recursos físicos y virtuales dinámicamente asignados y reasignados de acuerdo a la demanda del consumidor.

Hay un sentido de emplazamiento independiente sobre la ubicación exacta de los recursos proporcionados y del cual el cliente generalmente no tiene ningún control o conocimiento, pero puede ser capaz de especificar la ubicación en un nivel más alto de abstracción (por ejemplo: país, estado o centro de datos). Como ejemplos de pool de recursos se incluyen: el almacenamiento, el procesamiento, memoria y ancho de banda de la red.

- **Acceso a Red:** Es la capacidad de disponibilidad de la red y acceder a ella a través de los mecanismos normales de comunicación y el uso de plataformas heterogéneas de clientes livianos o no (por ejemplo: teléfonos móviles, tabletas, portátiles y estaciones de trabajo).

- **Medición de Servicios:** El NIST lo establece como el control automático y optimizado sobre el uso de los recursos mediante el aprovechamiento de la capacidad de medición en un cierto nivel de abstracción adecuado para el tipo de servicio (por ejemplo: almacenamiento, procesamiento, ancho de banda o cuentas de usuario activas).

  El uso de recursos se puede supervisar, controlar, e informar, proporcionando transparencia, tanto para el proveedor como para el consumidor del servicio utilizado.

- **Servicios Bajo Demanda:** La facilidad con la que un consumidor puede unilateralmente, aprovisionar las capacidades de computación, como el almacenamiento y los servidores de red, según sea necesario de forma automática sin necesidad de interacción humana, con cada proveedor de servicios.

- **Rápida Elasticidad:** Es la dinámica de poder elásticamente, aprovisionar o liberar recursos, en algunos casos de forma automática, para escalar rápidamente en consonancia con la demanda. Para el consumidor, las capacidades disponibles para la provisión de recursos a menudo deben parecer ilimitadas y aprovechables en todo momento.

En última instancia, se presentan los *Modelos de Servicios* que se pueden ofrecer en la nube. Para autores como Sosinsky, a este conjunto de servicios se le conoce como el modelo SPI (Software – Plataforma - Infraestructura) (Sosinsky, 2011) y es un término que abarca tres tipos populares de servicios ofrecidos: IaaS, PaaS y SaaS.

Dentro de esta concepción, es útil visualizar estos modelos de servicios en la nube de computación en términos de una pila compuesta por Hardware / Software (Sosinsky, 2011) los cuales proveen las operaciones universales sobre el modelo computacional.

Sobre esto, es notorio señalar que la manera en como la computación en la nube se ha desarrollado, ha permitido a diferentes proveedores ofrecer modelos de despliegue que tienen diferentes servicios asociados a ellos.

Así, la cartera de servicios ofrecidos añade otro conjunto de definiciones al modelo de servicios propuesto por el NIST, los cuales son descritos en la literatura bajo la forma: «*XaaS*» o «*'Something' as-a-service*» (Sosinsky, 2011) que referenciaría otros tipos de servicios específicos o especializados dentro de la nube, como por ejemplo: *StaaS*, Almacenamiento como Servicio; *IdaaS*, Identidad como Servicio; *CmaaS*, Cumplimiento como Servicio; etc.

Sin embargo, aunque la variedad de servicios ofertados por los proveedores puede ser abierta, esencialmente los servicios del conjunto SPI antes descrito, pueden abarcar todas las otras posibilidades (Sosinsky, 2011), por lo que su estudio se ampliará en las siguientes secciones.

2.2.2.1. La Nube como Infraestructura - (IaaS)

La descripción aportada por el NIST en relación al servicio de Infraestructura ofrecido desde la nube, destaca como característica general, la capacidad ofrecida por el proveedor de suministrar procesamiento, almacenamiento, redes y otros recursos fundamentales de computación tal que el consumidor o cliente del servicio sea capaz de desplegar y ejecutar Software arbitrario, el cual puede incluir sistemas operativos y aplicaciones específicas (Badger et al., 2012).

Igualmente, el consumidor no gestiona ni controla la infraestructura de nube subyacente, pero tiene el control de los sistemas operativos (Sosinsky, 2011), almacenamiento y aplicaciones implementadas; y el control posiblemente limitado, de componentes de red seleccionados (por ejemplo: firewalls) (Badger et al., 2012).

Generalmente un IaaS se puede obtener desde un modelo de despliegue público, privado o una combinación de ambos. Por su parte, la utilización de IaaS tiene sentido bajo una serie de situaciones que están estrechamente relacionadas con la beneficios que aporta el paradigma computacional en la nube.

Las situaciones que son particularmente adecuadas para el uso de servicios de Infraestructura de nube incluyen (Kepes, 2013):

- Cuando la demanda es muy volátil, es decir cuando en cualquier momento hay picos significativos o valles en términos de demanda sobre la infraestructura.
- Para nuevas organizaciones sin capital de inicio para invertir en Hardware (por ejemplo: startups).
- Cuando la organización está creciendo rápidamente y la ampliación del Hardware sería problemática.
- Donde hay presión sobre la organización para limitar los gastos de capital y trasladar los gastos de funcionamiento.
- Para líneas específicas de negocios, que requieren de infraestructura en carácter temporal.

Por otro lado, si bien el modelo de servicio IaaS ofrece ventajas enormes para situaciones donde la escalabilidad y el rápido aprovisionamiento son beneficiosos, hay escenarios en que sus limitaciones pueden ser problemáticas. Entre algunos de estos se podrían enunciar (Kepes, 2013):

- Cuando el cumplimiento de regulaciones de privacidad o seguridad hacen que la deslocalización o externalización de los datos, el almacenamiento y procesamiento se haga difícil.
- Cuando se requieren altos niveles de rendimiento y en las instalaciones o infraestructura organizada dedicada, no se tiene la capacidad para cumplir con las necesidades del consumidor o cliente.

2.2.2.2. La Nube como Plataforma - (PaaS)

Es la operación ofrecida al consumidor para desplegar en la nube, aplicaciones propias creadas utilizando lenguajes de programación, bibliotecas, servicios y herramientas apoyadas por el proveedor de la infraestructura adquirida (Mell & Grance, 2011).

En este modelo de servicios, el consumidor o cliente no administra o controla la infraestructura de nube subyacente como por ejemplo la red, servidores, sistemas operativos, o el almacenamiento, pero tiene plena administración sobre las aplicaciones desplegadas y la configuración del entorno que las hospeda.

El modelo de servicio PaaS, es similar en muchos aspectos a la infraestructura como servicio revisada anteriormente, se diferencia de IaaS por la adición de servicios de valor añadido los cuales pueden darse en dos formas distintas (Kepes, 2013):

1. Una plataforma de colaboración para el desarrollo de Software, centrada en workflow para organizar y controlar tareas, además de la gestión independiente de la fuente de datos que se utilice para el desarrollo de la aplicación. Un ejemplo de este enfoque sería Heroku, un PaaS que utiliza Ruby on Rails como lenguaje de desarrollo.

2. Una plataforma que permite la creación de Software utilizando datos patentados desde una aplicación. Este tipo de PaaS puede ser visto como un método para crear aplicaciones con un estándar común de datos o tipos. Un ejemplo de esta clase de plataforma sería la ofrecida por salesforce.com a través de force.com la cual se utiliza casi exclusivamente para desarrollar aplicaciones que trabajan con el salesforce.com CRM.

El uso de PaaS es especialmente útil en cualquier situación en la que varios desarrolladores están trabajando en un proyecto de desarrollo o cuando otras partes externas necesitan interactuar con el proceso de desarrollo (Kepes, 2013).

En ese sentido, PaaS es útil también cuando los desarrolladores desean automatizar los servicios de pruebas y despliegue.

Por otro lado, Kepes indica que la popularidad de desarrollo ágil de Software, un grupo de metodologías basadas en el desarrollo de Software iterativo e incremental, también aumenta la absorción de PaaS, ya que alivia las dificultades alrededor del rápido desarrollo y la iteración del Software (Kepes, 2013).

A pesar de sus ventajas hay ciertas situaciones en el uso de PaaS que pueden no ser ideales, los ejemplos incluyen (Kepes, 2013):

- Escenarios en los cuales las aplicaciones desarrolladas demanden ser altamente portables en términos de dónde se encuentren alojadas.
- Cuando los lenguajes de programación o el uso de tecnologías propietarias o muy especializadas impacten en el proceso de desarrollo.
- En situaciones donde un lenguaje propietario obstaculizaría movimientos posteriores a otro proveedor.
- Si el rendimiento de las aplicaciones requiere personalización del Hardware y el Software subyacente.

Sobre este servicio en particular, se realizará una exploración más profunda en tópicos siguientes de este trabajo de investigación. Sin embargo, muchos autores sostienen que PaaS se convertirá en el enfoque predominante hacia el desarrollo de Software dada su capacidad de automatizar procesos, utilizar componentes predefinidos, bien como la construcción y las tareas de despliegue en relación a la producción tradicional de Software (Kepes, 2013) (Buyya et al., 2009).

2.2.2.3. La Nube como Software - (SaaS)

En este modelo de servicio se habilita al consumidor o cliente con la capacidad de utilizar aplicaciones del proveedor que se ejecutan en una infraestructura de nube. Las aplicaciones son accesibles desde varios dispositivos a través de una interfaz de cliente ligero, como un navegador Web o una interfaz de programa (Mell & Grance, 2011).

Al igual que en los modelos anteriores, el consumidor no gestiona ni controla la infraestructura de nube subyacente como la red, servidores, sistemas operativos, almacenamiento o incluso las capacidades individuales de las aplicaciones, excepto ajustes de configuración específicos y limitados para usuarios (Mell & Grance, 2011).

El SaaS en particular, es un mecanismo de rápido crecimiento y uno de los más representativos del potencial del paradigma de computación en la nube en relación a la entrega de la tecnología a consumidores (Kepes, 2013).

Dicho esto, las organizaciones que consideran un movimiento tecnológico hacia la nube, deben tener en cuenta que aplicaciones pueden migrar o valerse de SaaS. Como tal, hay soluciones particulares que consideran situaciones ideales para un movimiento inicial sobre SaaS (Kepes, 2013):

- Aplicaciones donde hay interacción significativa entre la organización y el mundo exterior. Por ejemplo, el Software de correo electrónico, boletín de noticias o mercadeo.
- Las aplicaciones que tienen una necesidad significativa de acceso Web o móvil.
- El Software que es sólo utilizado para una necesidad a corto plazo. Un ejemplo sería el Software de colaboración para un proyecto específico.
- Programas o aplicaciones donde los picos de demanda aumenten de manera significativa, por ejemplo: aplicaciones de impuestos o de facturación.

Como se describió antes, SaaS es una herramienta muy valiosa para organizaciones que tienden a incursionar en el paradigma de computación en la nube no obstante, hay ciertas situaciones en las que no es la mejor opción para la entrega de Software entre las que se incluyen (Kepes, 2013):

- Aplicaciones donde se requiere procesamiento extremadamente rápido de los datos en tiempo real.
- Aplicaciones donde la legislación u otra normativa no permiten el alojamiento de datos externamente.
- Aplicaciones donde una solución existente en las infraestructuras cotidianas cubre todas las necesidades de la organización.

Finalmente, según se ha expuesto en este apartado del trabajo se ofrece en la **Figura 5**, una representación visual de los modelos de servicios en la nube, delimitando las diferencias entre PaaS, IaaS y SaaS, desde la perspectiva de gestión y acceso de los clientes y proveedores.

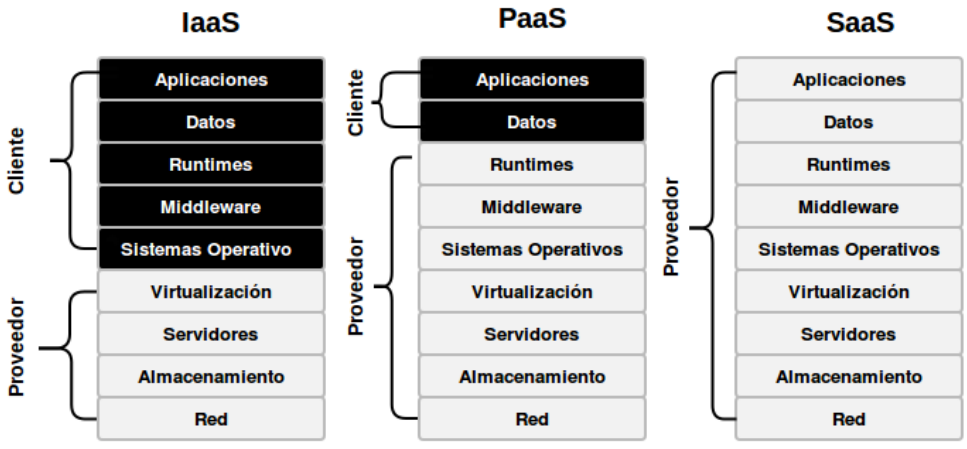

Figura 5. Representación visual de los modelos de servicios en la nube y sus delimitaciones.

2.3. La Nube y el Desarrollo de Software

Como mencionado anteriormente, el modelo de computación en la nube, está teniendo un profundo efecto sobre la gestión tecnológica de las organizaciones en distintos niveles (Marston et al., 2011) (Phaphoom et al., 2015) no siendo exentas, aquellas que incluyen dentro de sus procesos comunes, el desarrollo de Software.

Sobre esto, es importante considerar que la naturaleza basada en servicios de la nube (Sosinsky, 2011) requiere un enfoque diferente a la forma acostumbrada como se han definido los ciclos de vida de desarrollo de Software o SDLC (Software Development Life Cycle), lo cuales describen el proceso de desarrollo conjunto de Software desde su conceptualización hasta la operación y relevo (Chandrasekaran, 2015).

En un entorno de desarrollo de Software tradicional por ejemplo, se enfatizan los aspectos funcionales (Dutta, Sethi, & Khatri, 2014) debido a que se implementa sobre una infraestructura *in situ* y tangible dentro de las instalaciones de quien lo lleva adelante; con la seguridad implícita asociada (respaldos, control de versiones, etc.), el cumplimiento de ciertas regulaciones propias en el control de las configuraciones (sistemas operativos, runtimes, API, frameworks, etc.), la transparencia operativa en las etapas, entre otros.

No obstante, llevado al contexto de la nube el escenario cambia esta perspectiva, esencialmente porque como se describió en el apartado anterior, debe evaluarse el proceso desde la oferta de servicios y modalidades ofrecidos por la nube (Kadakia, 2014) (Kepes, 2013) en relación a los *Modelos de Servicios*, *los Modelos de Despliegue* y los *Atributos de Servicios* (Mell & Grance, 2011) (Sosinsky, 2011), los cuales de manera general, son ofertados por los proveedores.

En ese sentido, todas las fases generales de un ciclo de vida de desarrollo de Software que comprenda por ejemplo: recolección de requisitos, planificación, diseño, codificación, pruebas, despliegue, mantenimiento (Chandrasekaran, 2015), se abordarían desde la perspectiva de servicios en la nube, en particular desde la PaaS.

Bajo esta visión, la nube le ofrecería a los programadores acceso a las prestaciones de un entorno de desarrollo desde cualquier computador u ordenador sin necesidad de configurar nada (Kadakia, 2014), colaboración con equipos repartidos geográficamente (Blackman, Beeming, Fourie, & Schaub, 2015), facilidad de generar código para pruebas y control de calidad, bien como entornos de producción de las aplicaciones, de manera automáticamente gestionada (Kepes, 2013) (Buyya et al., 2009).

Si bien en relación al desarrollo *per se* de Software en la nube, existe una tendencia interesante hacia las llamadas metodologías ágiles (Kepes, 2013) (Kadakia, 2014) (Blackman et al., 2015) (Singh & Chana, 2013) en particular, este apartado describirá aquellos aspectos que de manera independiente al enfoque de desarrollo empleado, son los más notables en la percepción del desarrollo en la nube sobre la capa PaaS.

2.3.1. Ciclo de Vida del Software en la Nube

Aunque el paradigma computacional en la nube se ha presentado y descrito como una nueva y polivalente alternativa en la entrega de tecnología (Marston et al., 2011) y una potencial herramienta para el desenvolvimiento de Software desde una plataforma de servicios (PaaS), algunos aspectos conceptuales en particular, aquellos que están relacionados con los ciclos de vida de desarrollo de Software (SDLC) y aplicaciones, mantienen cierta invariabilidad en cuanto a su importancia (Kadakia, 2014).

Lo anterior es plausible dentro del contexto del desarrollo de Software dado que el éxito y/o la calidad de un proyecto de Software tradicionalmente se ha definido de manera general, por aspectos concretos a la forma como es desarrollado dentro del tiempo y el presupuesto; por la eficiencia, la facilidad de uso, la confiabilidad y la disponibilidad de mantenimiento posterior (Sommerville, 2007) (Chandrasekaran, 2015).

Si bien, hay muchos modelos de procesos de desarrollo o ciclos de vida para elegir en función del tamaño del proyecto, los requisito de tiempo y el tipo de proyecto a desplegar (Chandrasekaran, 2015), los aspectos anteriores se han mantenido presentes durante la evolución histórica de lenguajes, paradigmas de programación y tecnologías subyacentes que apoyan el proceso de desenvolvimiento de Software.

De esta forma, aunque la nube y su paradigma computacional, ofrece como novedad las posibilidades de un proceso de desarrollo ubicuo, transparente e independiente sobre la oferta de servicios que propone su pila PaaS, en esencia, los ciclos de vida de desarrollo de Software (SDLC), permanecen constantes, sintetizando un conjunto compuesto por distintas actividades encaminadas a describir un proceso de creación e implementación de Software (Chandrasekaran, 2015).

En este sentido, existen varios modelos o metodologías de desarrollo de Software como el modelo en Cascada, Ciclo en V, Incremental, RAD, Ágil, Iterativo y Espiral (Chandrasekaran, 2015). No obstante, todos estos SDLC son modelos genéricos no definitivos, que expresan los distintos métodos de desarrollo de Software y cómo abstracciones del proceso, se pueden utilizar para explicar diferentes enfoques en el desarrollo de Software, a modo de marcos de trabajo que pueden hacerse extensivos y adaptables para crear procesos más específicos de Ingeniería de Software (Sommerville, 2007).

Aunque existe cierto debate sobre el número apropiado de pasos, actividades o etapas y las convenciones de nomenclatura de los mismos (McMurtrey, 2013), las actividades de un ciclo de vida de desarrollo de Software, se armonizan de acuerdo con el modelo de proceso adoptado para el desarrollo de Software en particular (Chandrasekaran, 2015).

Con la propiedad anterior, proveedores y organizaciones orientadas a ofrecer plataformas de desarrollo en la nube, suelen establecer la adopción de ciclos de vida de desarrollo de Software particulares al contexto de sus entornos y herramientas (Cohen, 2013).

Una evaluación interesante de las fases de SDLC que se realizan con mayor significancia en los entornos de nube, fue realizada en 2015 por la revista especializada DZone (http://www.dzone.com).

Considerando factores como el crecimiento en las tendencias de adopción de servicios basados en PaaS, el análisis de las prácticas actuales y el uso de diferentes plataformas por parte de las organizaciones, así como la investigación de campo mediante la aplicación y recolección de encuestas aplicadas a más de 600 profesionales de TI (Bryant, Morgenthal, Haddad, Golden, & Wetherill, 2015), el estudio reveló entre otros resultados, el crecimiento de fases de SDLC como Pruebas y Evaluación de Calidad así como de Producción y Despliegue en la nube.

Los resultados mostraron que el 55% de los encuestados, están realizando fases de Pruebas y Evaluación de Calidad en la nube (11% de aumento) por cuanto que el 62% llevan a cabo etapas de Producción y Despliegue de aplicaciones (aumento del 10%) (Bryant et al., 2015).

Según la evaluación, este crecimiento no es sólo en los entornos de desarrollo y pruebas sino que incluye la producción y despliegue de las aplicaciones una vez desarrolladas, de lo que se deduce, que existe un impulso en el uso de SDLC sobre plataformas PaaS en la nube.

Es de notar la consideración del estudio realizado por DZone, al evaluar esencialmente las fases más técnicas de un proceso o ciclo de desarrollo de Software, como lo son el Desarrollo, Pruebas, Evaluación de Calidad, Producción y Despliegue antes mencionados, obviando por ejemplo, etapas como Análisis de Requisitos, Diseño, etc.

Esto podría ser visto desde la condición que las fuentes que aportan los datos (profesionales de TI), están vinculadas a organizaciones que operan en un entorno global, el cual cambia rápidamente, por lo que deben responder a las nuevas oportunidades y mercados, así como a cambios en las condiciones económicas y la aparición de productos y servicios de la competencia (Sommerville, 2007) por tanto, el uso de los beneficios de la computación en la nube y el desarrollo de Software sobre la PaaS debe responder a esos escenarios.

Así, la tendencia descrita por el estudio podría guardar relación con la introducción de metodologías ágiles en los procesos de desarrollo sobre PaaS (Kepes, 2013). Los métodos ágiles, están orientados al desarrollo rápido de Software, involucrando explícitamente el usuario en el equipo de desarrollo y reduciendo el papeleo y la burocracia en el proceso de Software (Sommerville, 2007) (Chandrasekaran, 2015) (Singh & Chana, 2013).

Debido a que estas organizaciones están operando en un entorno cambiante, a menudo es prácticamente imposible que puedan obtener un conjunto completo de requisitos de Software estables, por lo que sería factible en la práctica observada por el estudio de DZone, la influencia de este conjunto de SDLC.

Retomando los criterios de la encuesta, se consideraron para ésta tres tendencias en la utilización de la nube, relativas a la implementación de tareas pertinentes con los ciclos de vida de desarrollo de Software, evaluando en el conjunto, la integración de dichas tareas por parte de las fuentes encuestadas. De esta manera, se clasificaron dentro de los procesos de desarrollo aquellas tareas que *Actualmente son realizadas*, las que *Planean realizarse* y finalmente, aquellas que *No planean realizarse* (Bryant et al., 2015).

Bajo estos criterios, es posible explicar los comportamientos ilustradas por los gráficos a continuación y que extraen *a priori*, la situación actual del uso de ciclos de vida de desarrollo de Software según la evaluación de DZone.

El **Gráfico 1** ilustra el porcentaje de adopción y realización de actividades propias de la etapa de *Desarrollo* de Software sobre PaaS, tales como prototipos o codificación (considerando un enfoque de SDLC ágil). Siendo que el 53% de los encuestados *actualmente realizan* éstas dentro de sus procesos.

Por otra parte, 24% de la evaluación indica que las organizaciones del estudio contemplan o *planean realizar* sus actividades de desarrollo en la nube. Finalmente, existe un 23% que mantienen el criterio de *no realizar* estas tareas sobre un modelo basado en la nube (Bryant et al., 2015).

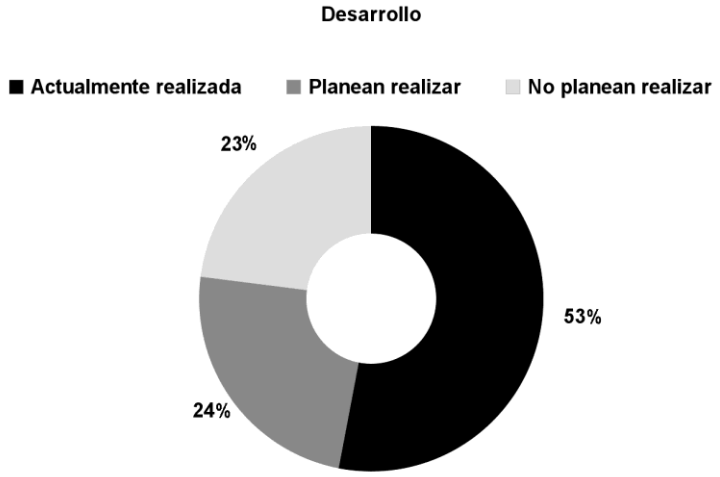

Gráfico 1. Implementación porcentual de tareas de Desarrollo en la nube [Fuente: DZone (Bryant et al., 2015)].

Otras de las fases que la encuesta de DZone reveló que se implementan con frecuencia en los entornos de desarrollo sobre PaaS, son las *Pruebas y la Evaluación de Calidad*.

Sobre estas fases, podrían definirse las tareas relacionadas con el control del código fuente, la revisión de código, pruebas de la aplicación, integración, etc., en función del SDLC adoptado.

En este sentido, el **Gráfico 2**, expone un porcentaje del 44% de adopción de tareas relacionada con *Pruebas y la Evaluación de Calidad* sobre entornos basados en PaaS. Por su parte, un 40% de los encuestados destaca que planean implementar actividades de este tipo dentro de sus procesos de desarrollo, en cuanto el 16% mantiene una posición de no realizar ninguna tarea de esa índole sobre entornos de nube (Bryant et al., 2015).

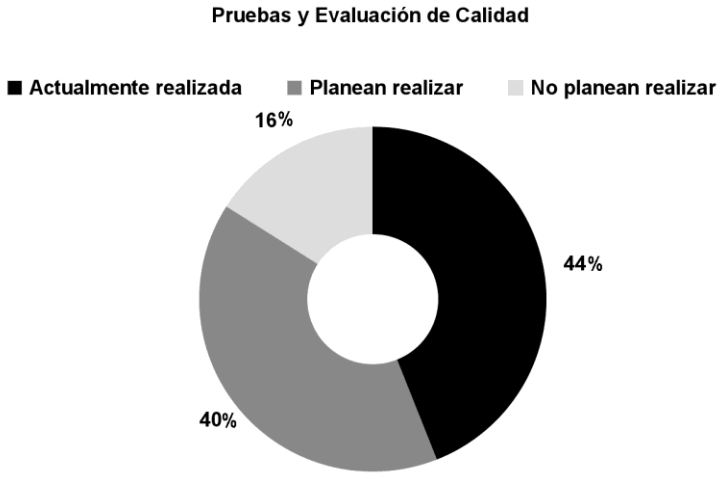

Gráfico 2. Implementación porcentual de tareas de Pruebas y Evaluación de Calidad en la nube [Fuente: DZone (Bryant et al., 2015)].

Finalmente, en relación a las fases de *Producción y Despliegue*, la encuesta de DZone indica, según el **Gráfico 3,** que el 52% del conjunto encuestado, actualmente realiza tareas dentro de estas fases, sobre plataformas basadas en la nube computacional. Un 37% a su vez, planean su adopción dentro del proceso de desarrollo de Software, mientras que el restante 11% no manifiestan planear su realización sobre PaaS como parte de sus ciclos de vida de desarrollo de Software (Bryant et al., 2015).

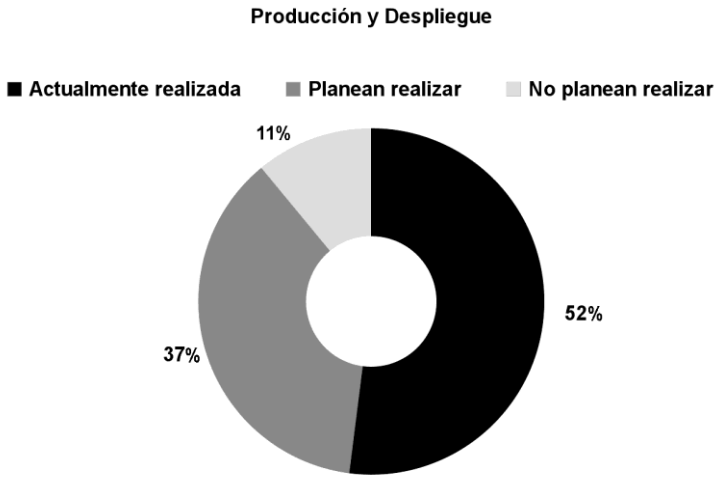

Gráfico 3. Implementación porcentual de tareas de Producción y Despliegue en la nube [Fuente: DZone (Bryant et al., 2015)].

La relativa reticencia del conjunto de encuestados que no tiene pensada la integración de sus SDLC a una PaaS, podría guardar correspondencia con la naturaleza crítica de los sistemas que desarrollan (Diez & Silva, 2014), regulaciones de estado (Marston et al., 2011) o bien porque los desarrolladores han sido reacios a migrar a las nuevas herramientas y plataformas (Cohen, 2013).

En relación a este último planteamiento, Cohen expresa que hasta hace poco, el desarrollo directo de aplicaciones empresariales en la nube sobre PaaS, se había visto obstaculizado por la falta de herramientas adecuadas y suites integrales que fueran ricas en las características necesarias para apoyar una cartera empresarial completa (Cohen, 2013). Así, un sector de los desarrolladores corporativos se mantendrían reacios a migrar a nuevas herramientas, pues estarían satisfechos con la gama de entornos tradicionales, los cuales les resultan familiares y están listos para usar en las tareas del oficio: desarrollar, codificar, probar y desplegar aplicaciones.

Sin embargo, en los últimos años los desarrolladores han expresado mayor interés en implementaciones particulares o privadas de herramientas PaaS, como alternativas a productos distribuidos públicamente inmaduros o potencialmente inseguros (Cohen, 2013). Esto por ejemplo, viene dado del esfuerzo de entornos PaaS ofrecidos por proveedores como Microsoft (Azure), Heroku o Red Hat (OpenShift), los cuales no sólo permiten gestionar tareas de desarrollo, sino además, orquestan y soportan todo el proceso de un SDLC.

Una valorización interesante, ofrecida por Cohen, es que las implementaciones de productos o herramientas de desarrollo sobre PaaS privadas, pueden ser costosas y difíciles de asimilar, por lo que los beneficios deben sopesar con las desventajas (Cohen, 2013). En la sección a seguir, se revisarán algunas de estas y otras caracterizaciones, con el fin de dirigir el enfoque y gradualmente, conducir el presente estudio a una consideración objetiva sobre estos criterios.

CAPÍTULO – III

Modelo de Desarrollo de Software sobre PaaS

En la siguiente sección, se abordan de manera concisa las principales caracterizaciones sobre el modelo de desarrollo de Software mediante el uso de PaaS. La premisa principal es ofrecer un compendio de las generalidades conceptuales, ventajas y consideraciones de adopción bajo esta plataforma ofrecida por la nube.

3.1. Características de la Pila PaaS como Middleware de Desarrollo

Como se revisó en la sección previa, las organizaciones que priman el desarrollo de aplicaciones de Software como parte de sus procesos operativos, están dando un giro sobre sus modelos de negocio tradicionales y de esta manera, cambiando sectores considerados históricamente *estables* como lo eran los métodos y tecnologías de desenvolvimiento de Software (Dutta et al., 2014).

Estas reacciones, la cuales están en tendencia ascendente (Columbus, 2014), evidencian la respuesta organizacional a las necesidades del mercado, la ampliación de la demanda (Bryant et al., 2015) (Sommerville, 2007) y la creación de nuevos servicios entre otros escenarios dinámicos, los cuales continuamente marcan el ritmo de evolución de las aplicaciones que permiten sustentar la competitividad de dichas organizaciones.

En este sentido y con el fin de facilitar y agilizar el proceso de desarrollo y entrega de aplicaciones, los agentes organizacionales están demandando cada vez más, el uso del modelo *Platform-as-a-Service* (PaaS) (Bryant et al., 2015) (Linthucum, 2014) como una solución que les permita expandir sus recursos tecnológicos en particular, sobre el desarrollo de Software.

Un respaldo al argumento anterior, es ilustrado por el **Gráfico 4**, estructurado según datos de la encuesta DZone (Bryant et al., 2015).

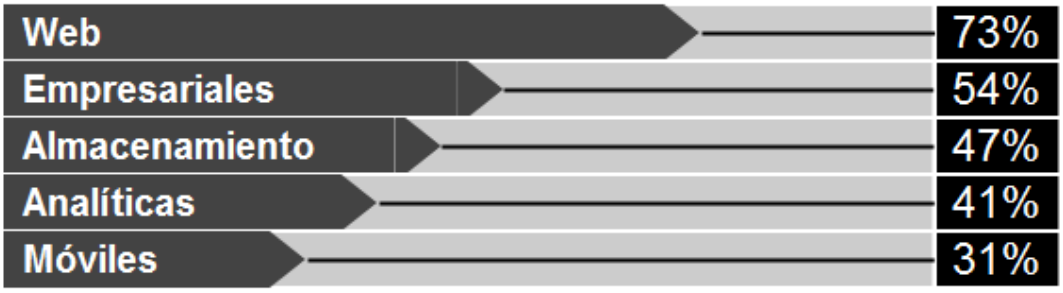

Gráfico 4. Distribución porcentual de aplicaciones desarrolladas en ambientes de nube sobre PaaS [Fuente: DZone (Bryant et al., 2015)].

Como se observa, existe un amplio porcentaje de organizaciones (73%) decantadas hacia el desarrollo prioritario de aplicaciones de Software basadas en Web, utilizando entornos de computación en la nube sobre servicios de PaaS. Esto podría por ejemplo, asociarse con el posicionamiento comercial y estratégico en pro de ofrecer productos y servicios críticos, desde portales Web de demanda crítica y accesibles desde diferentes dispositivos y lugares o de manera más técnica, con los lenguajes y plataformas más comunes ofrecidas en PaaS como PHP, Ruby, Python y Node.js (Carlson, 2013).

Igualmente, es posible notar que el conjunto particular que utilizó la revista como base estadística, coloca niveles de representatividad relativamente altos en relación con otras categorizaciones de aplicaciones que se desenvuelven sobre ambientes PaaS siendo: 54% Empresariales, 47% Almacenamiento, 41% Analíticas, 31% Móviles respectivamente.

Con base en los porcentuales encontrados sería apropiado evaluar como antes se ha dicho, que efectivamente, la propuesta ofrecida por la *plataforma como servicio* está teniendo una masiva utilización en el ámbito corporativo, consolidando las tendencias argumentadas por las literaturas consultadas en ese sentido.

Por su parte, la revisión relacionada con el **Gráfico 5**, provee interesantes indicadores adicionales que contextualizan, según la revista DZone, la poca diferenciación entre los modelos de servicios PaaS e IaaS, los cuales muestran individualmente, una distribución porcentual de 60% y 59% sobre otras modalidades de servicios implementados en el ámbito corporativo de producción de aplicaciones en la nube (Bryant et al., 2015).

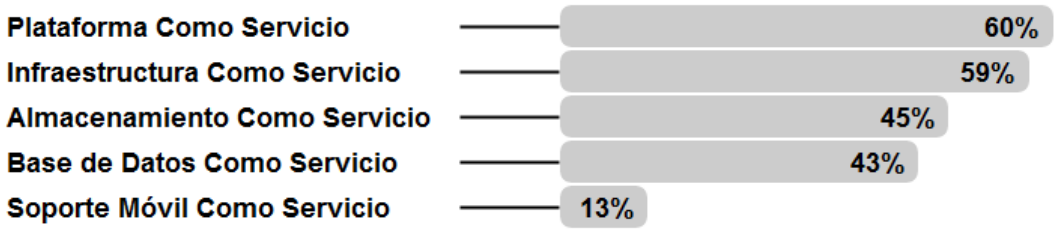

Gráfico 5. Servicios de nube usados en producción para aplicaciones [Fuente: DZone (Bryant et al., 2015)].

Estas observaciones son importantes, puesto que las segmentaciones encontradas con los valores anteriores, son verificables en la práctica debido principalmente, a que las líneas de delimitación técnica entre IaaS y PaaS no quedan completamente definidas (Kepes, 2013).

Sobre lo anterior, es importante comprender que el margen de diferencia entre ambos modelos de servicios, según muestra la encuesta, sería concordante y característico con las argumentaciones propuestas por Kepes (Kepes, 2013), Furht et al., (Furht et al., 2010) en las cuales PaaS, presenta propiedades similares a IaaS, o como afirma Furht et al., "*PaaS es un IaaS con una pila de Software a medida para una aplicación dada"* (Furht et al., 2010).

Bajo ese enfoque, por ejemplo, proveedores de soluciones computacionales en la nube como Amazon o Standing Cloud (http://www.standingcloud.com) ofrecen IaaS como parte de soluciones PaaS, que, entre otras características, permiten la capacidad automática de administrar la implementación de aplicaciones, el aprovisionamiento y la gestión de los recursos IaaS que sean requeridos para desplegar aplicaciones, de manera transparente al cliente, sin trazar ámbitos específicos de uso entre ambas.

De este modo, la enunciación hecha tanto por Kepes como por Furht et al., es admisible bajo la conceptualización adicional que la estructura de los modelos de servicios para la nube, tiende a ser representada por una jerarquía en la cual, se sitúa PaaS, como una sección intermedia entre SaaS e IaaS (Furht et al., 2010) (Teixeira et al., 2014) (Kepes, 2013) como ilustrado en la **Figura 6**.

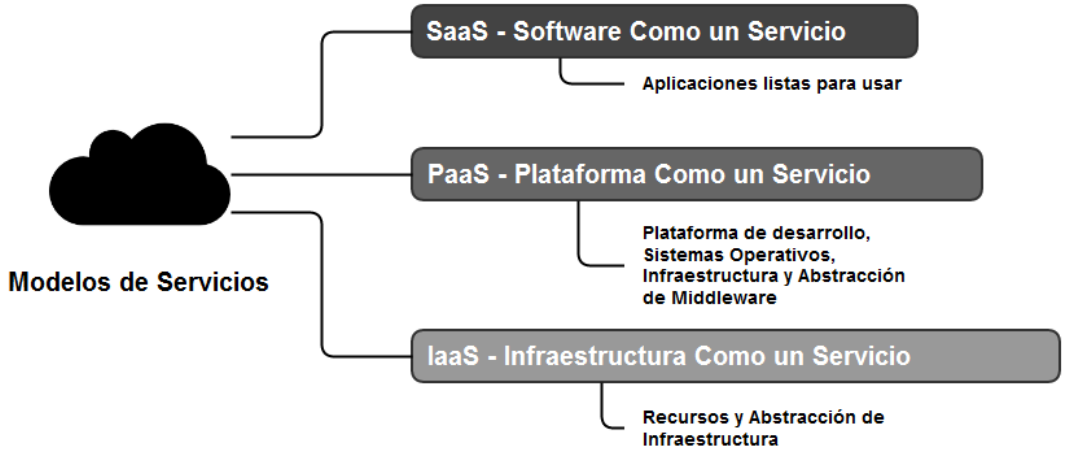

Figura 6. Organización jerárquica de los modelos de servicios para la nube (Basado en Teixeira et al., 2014 y Kepes, 2013).

Otra argumentación que apoya la consideración de los autores citados de que PaaS opera desde un nivel intermedio, viene dada de las propias particularidades conceptuales del funcionamiento de IaaS, las cuales indicarían la capacidad de ésta de ejecutar Software arbitrario, incluyendo sistemas operativos y aplicaciones específicas (Badger et al., 2012) y la habilidad del usuario o consumidor, de poder controlar dichos sistemas operativos, bien como las capacidades de almacenamiento y aplicaciones implementadas (Sosinsky, 2011).

Esta concepción de IaaS, otorgaría por tanto, la capacidad a PaaS de ser un espacio idóneo, o un *middleware* para la gestión y vínculo entre las aplicaciones *"listas para el uso"* que son provistas por la capa SaaS en la cúspide de la jerarquía y los recursos de control virtual y físico con los cuales se disponen para desplegar dichas aplicaciones desde la capa IaaS situada en la zona más baja del escalafón, según la organización de servicios presentada en la **Figura 6**.

Así, la correlación entre ambas capas estaría vinculada a la integración de PaaS con los servicios de IaaS, pues emplearía como soporte a sus propios servicios, la infraestructura subyacente proporcionada por IaaS.

Una profundización sobre lo anterior, es aportada por Blacharski y Landis (Blacharski & Landis, 2013) quienes describen las propiedades de PaaS desde su naturaleza como *"plataforma"*, concepto que consideran importante comprender para establecer una base sobre la cual discernir la importancia de PaaS, como entorno de desarrollo de Software.

En un sentido inicial, Blacharski y Landis sostienen que el término *plataforma*, generalmente se referirá a una arquitectura *"prefabricada"* de Software sobre la cual se pueden construir soluciones informáticas u otras aplicaciones. Proporcionando de esta manera, la funcionalidad básica para el desarrollo y ejecución de Software que de otro modo tendría que ser diseñada desde cero (Blacharski & Landis, 2013).

El concepto de plataforma también se aplica en la reducción de costos del Software y su mantenimiento, debido a que la responsabilidad de mantener la plataforma de código es esencialmente subcontratada a un proveedor (Blacharski & Landis, 2013).

Por otra parte, el concepto de *middleware* según descrito por Buyya et al., referencia la posibilidad de diseñar *servicios dinámicos* intermedios que permiten el uso de recursos informáticos (Hardware o Software) a potestad de la demanda, tal que cuando ya no son necesarios no se incurre en costos adicionales, representando una ventaja para los desarrolladores (Buyya et al., 2010).

De manera adicional, en Buyya et al., explican que el concepto de *middleware* como *servicios dinámicos* intermedios, puede sentar las bases de servicios adicionales más complejos. Ofreciendo a las organizaciones la capacidad de implementarlos directamente e integrarlos en su propia infraestructura operacional.

Finalmente, señalan Buyya et al., que el término común en el contexto del paradigma computacional de la nube para las arquitecturas de este tipo, es el de *plataforma como servicio* (PaaS) (Buyya et al., 2010).

Considerando entonces, las conceptualizaciones de ambas literaturas, cabría entender que PaaS como *middleware*, actúa como una capa de abstracción entre los niveles más altos de los recursos manejados por la IaaS pero a su vez, en los estratos más bajos, relativos a las aplicaciones desplegadas por la SaaS.

De esta manera, el objetivo del proveedor de PaaS, sería ofrecer un proceso abstracto y repetible para la creación y despliegue de aplicaciones de alta calidad, ocultado la complejidad de los servicios de computación subyacentes a los desarrolladores con el fin de agilizar el ciclo de vida de desarrollo de aplicaciones y el proceso de implementación en entornos de nube públicos o privados (Teixeira et al., 2014) (Cohen, 2013).

Esta abstracción, se lograría embebiendo en la estructura operativa de PaaS los elementos necesarios para que puedan brindarse como servicios, toda una serie de componentes específicos, generales o propietarios, tales como lenguajes de programación, librerías, API, máquinas virtuales y otros que permitan el desarrollo de Software constituido para soportar entre otras cosas, un gran número de clientes, procesamientos muy demandantes, grandes cantidades de datos y el acceso potencial desde cualquier punto de Internet (Badger et al., 2012).

3.2. Dinámica General de Funcionamiento

De manera universal, el concepto abordado hasta el momento en cuanto a las propiedades de funcionamiento de PaaS, es que esta plataforma permite a los desarrolladores poder diseñar, implementar, migrar, escalar y administrar sus aplicaciones en la nube (Kepes, 2013), sin necesidad de llevar a cabo configuraciones manuales (Bahsoon et al., 2013) (Dillon et al., 2010).

No obstante, en este apartado del trabajo se revisarán de manera más específica, las dinámicas de operación de PaaS desde la conceptualización técnica propuesta por el NIST como modelo estándar de implementación.

En relación a esto y de manera inicial, NIST categoriza PaaS como "*un conjunto de bloques de construcción de Software y herramientas de desarrollo, tales como lenguajes de programación, que proveen soporte en tiempo de ejecución, a entornos que facilitan la construcción de aplicaciones de alta calidad y escalables*" (Badger et al., 2012).

Una interpretación de la descripción anterior, puede ser entendida desde la perspectiva que los servicios PaaS normalmente, proporcionarán herramientas que ayuden con el despliegue de nuevas aplicaciones. Estos servicios PaaS, incluyen el diseño de aplicaciones, su desarrollo, pruebas, despliegue y alojamiento bien como, de manera adicional, herramientas de colaboración en equipo, integración con servicios Web, acceso a bases de datos, seguridad, escalabilidad, almacenamiento y control de versiones (Velte et al., 2010).

En algunos casos, según propone el NIST, la implementación de una nueva aplicación de Software en un entorno PaaS por medio de servicios bien definidos, no debe ser más difícil que subir un archivo a un servidor Web (Badger et al., 2012).

Sobre este contexto de uso, por ejemplo Beth Cohen, destaca que la nueva generación de servicios integrados alrededor de herramientas PaaS, han evolucionado al ofrecer el desarrollo de aplicaciones de manera más eficaz principalmente, mediante la implementación de normas basadas en SOA, las cuales permiten la reutilización de componentes y la integración con herramientas de desarrollo de manera transparente al desarrollador (Cohen, 2013).

Aunando en ese aspecto, los desarrolladores como consumidores de servicios, podrían por ejemplo, desenvolver y desplegar una aplicación en línea a través de herramientas de cliente, tales como una interfaz de usuario Web desde el navegador, usar consolas de línea de comando o Web CLI (Web Command Line Interface) y por medio de algún API REST (Representational State Transfer) proporcionado por el proveedor de servicios (Chandrasekaran, 2015) (Carlson, 2013) quedando la complejidad de la administración de los recursos por cuenta de la plataforma.

Esto coincidiría con lo señalado por Sosinsky que en un modelo PaaS, los clientes pueden interactuar con el Software para entrar y recuperar datos, realizar acciones, obtener resultados y en la medida en que el proveedor lo permita, personalizar la plataforma en cuestión (Sosinsky, 2011). Con estas características de facilidad en el uso de PaaS, los desarrolladores pueden concentrarse más sobre las funcionalidades concretas de la aplicación a desarrollar en lugar de invertir esfuerzo en crear adaptaciones específicas con relación a aplicaciones SaaS (Chandrasekaran, 2015).

Por otra parte, los servicios ofrecidos por PaaS en general, también incluirán los mecanismos de automatización de acceso a los recursos informáticos requeridos para el despliegue que las aplicaciones de los clientes o suscriptores necesitan utilizar, por ejemplo, el procesamiento, almacenamiento y redes (Badger et al., 2012).

En adhesión de apoyar los conceptos previamente explicados, el NIST propone un modelo de interacción dinámica que ofrece una vista simplificada de cómo funcionan los entornos PaaS, el cual permite ilustrar los aspectos clave de éste. El modelo propuesto por NIST se muestra en la **Figura 7**.

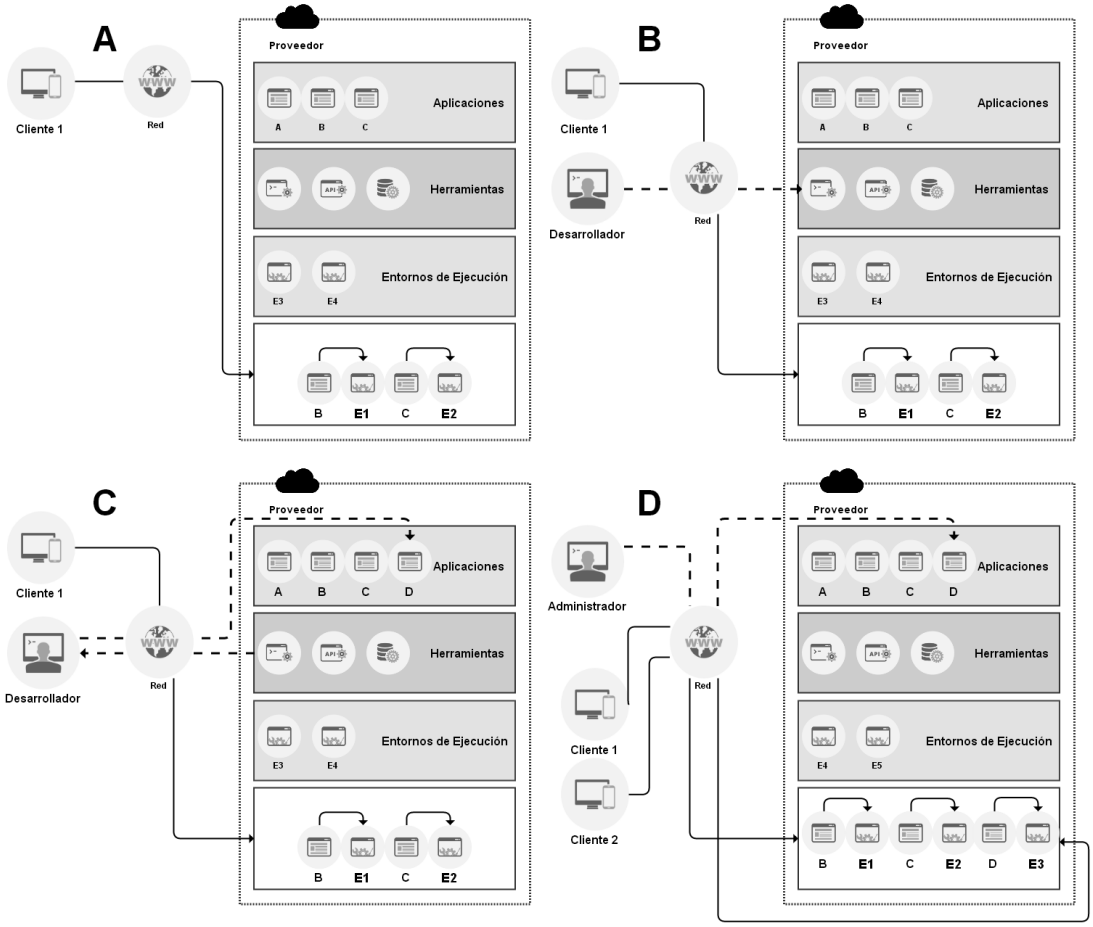

Figura 7. Modelo de Interacción en PaaS propuesto por el NIST (Basado en Badger et al., 2012).

Es importante señalar con relación a lo anterior y previo a la explicación del modelo, que hay muchas variaciones en este escenario básico de interacción. Por ejemplo, un desarrollador puede modificar una aplicación existente en lugar de crear una nueva, además, las fases normales de desarrollo de Software, incluidas las pruebas, gestión de versiones y entre otras, no son consideradas (Badger et al., 2012).

El escenario de interacción sobre el funcionamiento de PaaS, presentado en la **Figura 7**, se divide en cuatro cuadrantes (***A***, ***B***, ***C***, ***D***), que recrean distintos momentos de interactividad entre clientes consumidores de servicios sobre la plataforma.

De esta manera, en un momento inicial en el cuadrante ***A***, el proveedor de PaaS posee un inventario actualizado de tres aplicaciones desplegadas (*a, b, c*). El proveedor de la nube también puede mantener un conjunto de herramientas de desarrollo, como segundo nivel en la figura y un conjunto de entornos de ejecución (*e3, e4*) en el tercer nivel.

Al igual que en el caso de un proveedor de SaaS, un entorno de ejecución como los representados, podría ser un equipo físico, una máquina virtual, un programa de servidor que atiende solicitudes de los clientes, incluso un entorno de ejecución puede contemplar la posibilidad de iniciar una máquina virtual o alquilar los ciclos de computación y almacenamiento de otra organización (Badger et al., 2012).

Aún en el cuadrante ***A***, también se representan dos aplicaciones activas, $b \rightarrow e1$ y $c \rightarrow e2$ lo que indica que las aplicaciones *b* y *c* se ejecutan en entornos de ejecución independientes (del mismo modo lo harían en un entorno SaaS).

En el cuadrante ***B***, un nuevo cliente y un desarrollador accede a las herramientas de desarrollo del proveedor, las herramientas de desarrollo pueden incluir los lenguajes de programación, compiladores, interfaces, herramientas de pruebas y mecanismos para implementar una aplicación una vez que esté terminada.

El cuadrante ***C*** ilustra el uso de las herramientas de desarrollo. En este escenario, el desarrollador puede descargar las herramientas y usarlas en su infraestructura local o accederlas desde la infraestructura del proveedor. En cualquier caso, la salida de las acciones del desarrollador es una nueva aplicación *d*, como se muestra en la **Figura 7**, que se despliega en la infraestructura del proveedor.

En el cuadrante ***D***, un administrador se presenta interviniendo en la configuración de la nueva aplicación *d* que ha sido puesta a disposición y un nuevo cliente, *Cliente2* se muestra como usuario de la nueva aplicación.

El modelo del NIST señala finalmente, que las PaaS basadas en un modelo de despliegue *Público*, pueden proporcionar capacidades operativas similares, aunque la escala puede ser restringida en función de los términos de uso de los recursos ofrecidos (Hardware, entornos de ejecución, etc). Por su parte, en las PaaS *Privadas* o *Comunitarias*, la escala está limitada por los recursos del centro de datos del proveedor (Badger et al., 2012).

3.3. Distribución de Controles y Acceso en la Pila PaaS - Proveedor vs. Suscriptor

El acceso a los recursos de la nube (SaaS, PaaS, IaaS) por su naturaleza, es más limitado que en las aplicaciones y recursos locales. Por lo general, la disponibilidad de aplicaciones de Software *in situ*, es bastante simple y más rápido, lo que promueve que la incorporación e integración de nuevos recursos de Software y Hardware, así como el desarrollo y la personalización de aplicaciones sea más fácil (Buyya et al., 2010).

No obstante, como señalan Buyya et al., una vez que las aplicaciones se trasladan a la nube, las futuras personalizaciones y el desarrollo de nuevas deben ser proyectadas y diseñadas para apoyar la integración con la plataforma existente, puesto que ya no hay un nivel de acceso directo ni inmediato. Así, las organizaciones que colocan sus aplicaciones en la nube o los *suscriptores* de servicios basados en la nube, pasan a depender del *proveedor* para proporcionar los mecanismos de integración y las API de desarrollo (Buyya et al., 2010).

Si bien hasta el momento, se ha entendido de manera indirecta el dominio de gestión entre quienes utilizan los servicios en la nube y quienes los proveen, cabe oportuno en este apartado, emplear la terminología técnica correcta, que permita profundizar en el tópico de esta sección del trabajo. En este sentido, como antes se ha mencionado, se constituyen dos elementos en la dinámica de interacción con los servicios en la nube, los *proveedores* y los *suscriptores* (Marston et al., 2011).

Inicialmente, los consumidores, clientes o usuarios de servicios, son efectivamente denominados *suscriptores* y son los que adquieren por alquiler o compra, el uso de los servicios a los *proveedores* sobre una base de gastos operativos (Marston et al., 2011).

Por su parte, los *proveedores* de servicios de computación en la nube, son quienes poseen y operan sistemas de cómputo para entregar el servicio a terceros, en el caso contextual, a los *suscriptores*. Los proveedores llevan a cabo el mantenimiento y las mejoras en las plataformas que sustentan los sistemas de los cuales los consumidores o suscriptores, estaban a cargo. De igual modo, los proveedores son los responsables del mantenimiento del Software utilizado en la nube, junto con la fijación de precios de los servicios por su uso (Marston et al., 2011).

Una acotación importante a considerar, es que los diferentes proveedores que circundan la prestación de servicios en la nube hoy día (Amazon, Google, etc.) han desarrollado competencias relacionadas a los diferentes elementos (Software, Plataforma e Infraestructura) que componen el servicio de computación en nube.

De esta manera, la disociación conceptual entre *proveedor* y *suscriptor* se presta a la separación física y la distribución de elementos de responsabilidad a través de los límites de sus roles (Furht et al., 2010) lo que permite de manera general, proceder a explicar los controles de acceso respectivos sobre la pila de desarrollo en PaaS.

De lo anterior, el NIST específica un ámbito de dominio y acceso sobre el PaaS, en el cual el proveedor controla de manera privilegiada, las capas inferiores de la pila de Software, como se muestra en la **Figura 8**, en la cual se ilustran las responsabilidades de control y gestión entre los agentes involucrados.

Figura 8. Delimitación del control sobre la pila PaaS propuesto por el NIST (Basado en Badger et al., 2012).

En el centro, la **Figura 8** representa una pila de Software tradicional que comprende conceptualmente, capas para el Hardware, sistema operativo, middleware y aplicaciones. La figura también representa una asignación de la responsabilidad ya sea para el proveedor de la nube, el subscritor o ambos (Badger et al., 2012).

Como se indica, el proveedor opera y controla las capas más bajas, donde se sitúan por ejemplo, el sistema operativo y el Hardware. Implícito en esto, es el control de la infraestructura de redes, como LAN y los equipos de comunicación (Routers, Switches) entre los centros de datos.

En la capa de middleware, el proveedor ofrece interfaces de programación y de servicios públicos disponibles para el suscriptor, estas interfaces proporcionan el entorno de ejecución, en el cual las aplicaciones del suscriptor se despliegan y a la vez, prestan disponibilidad de acceso a los recursos necesarios, tales como ciclos de CPU, memoria, almacenamiento persistente, bases de datos, conexiones de red, entre otros, situados en la capa inferior, como antes descrito (Badger et al., 2012).

El proveedor determina el modelo de programación es decir, las circunstancias bajo las cuales el código de la aplicación del suscriptor se activa, realizando también la supervisión de las actividades de los programas de los suscriptores para fines de facturación.

Una vez que un suscriptor ha utilizado las características de los servicios de PaaS, para implementar y desplegar una aplicación, esta última es esencialmente extendida como SaaS y el suscriptor tiene control administrativo sobre ella, sujetándose únicamente al apoyo operacional que el proveedor ofrezca en términos del acuerdo de las condiciones de uso estipuladas por el *Acuerdo de Nivel de Servicio* (SLA - Service Level Agreement) (Badger et al., 2012) (Wu & Buyya, 2010) (Dillon et al., 2010).

3.4. Ventajas Ofrecidas por el Modelo

Según indican Teixeira et al., PaaS está en una fase aún temprana, especialmente cuando se compara con los estados de madurez y de desarrollo de otros componentes de la nube. Como resultado de esto, no hay de momento, un PaaS disponible que cumpla a cabalidad y simultáneamente, con las necesidades de los suscriptores, el presupuesto y la capacidad de adaptación al sistema, aunque las expectativas para obtener tales soluciones de los proveedores de PaaS son altas (Teixeira et al., 2014).

No obstante, si comparada con las soluciones de cómputo y de distribución de Software tradicionales, PaaS proporciona muchas ventajas consecuentes del modelo computacional en la nube y como revisado en la sección anterior, aporta de inmediato, un cambio significativo en las cargas de operación entre los suscriptores y los proveedores (Badger et al., 2012).

Lo anterior, resulta en una serie de oportunidades técnicas en pro de ofrecer y favorecer una mayor eficiencia en los procesos de desarrollo e implementación de Software y en algunos casos, mejoras substanciales en el rendimiento de las aplicaciones derivadas del desarrollo en PaaS.

Por otro lado, en los escenarios en que se emplea PaaS bajo cualquiera de los modelos de despliegue de servicios en la nube mencionados previamente en este trabajo (Sección 2.2.2, *Figura 4*), el proveedor es libre para ubicar su infraestructura de nube en zonas de bajo costo, permitiendo a los suscriptores acceder a sus servicios a través de Internet (Badger et al., 2012) o como señala Sosinsky, *"bajo demanda"* (Sosinsky, 2011).

Al retener el control sobre las capas inferiores de la pila de Software como se ilustró en la **Figura 8** del apartado anterior, los proveedores de PaaS, son capaces de gestionar las interioridades técnicas de los servicios ofrecidos y abstraer a los suscriptores de los servicios, de la responsabilidad en la selección, instalación, mantenimiento y operación de los componentes de la plataforma.

Considerando que si bien, los servicios ofrecidos al suscritor consumen recursos de infraestructura en alguna forma, los cargos por utilización se asocian en las tarifas que se cobran por los recursos del entorno de ejecución de PaaS que sean requeridos (por ejemplo, CPU, ancho de banda, almacenamiento, etc.) (Sosinsky, 2011) (Badger et al., 2012).

Esto ofrece en relación a los gastos de infraestructura, una ventaja económica apreciable para los suscriptores o clientes de los servicios, ya que éstos están implícitamente presentes en las ofertas PaaS propuestas por los distintos proveedores.

Aunque podrían enunciarse muchas otras ventajas adicionales ofrecidas por el modelo, se ha propuesto como delimitación para el presente esfuerzo investigativo, seleccionar tres de las más destacables, las cuales serán descritas en las siguientes secciones.

3.4.1. Bajo Impacto en el Uso de Herramientas de Desarrollo

Como describe gran parte de la literatura consultada en el presente desarrollo investigativo (Mell & Grance, 2011) (Buyya et al., 2010) (Blacharski & Landis, 2013), el principal elemento de presentación y entrega de los servicios en nube, es a través de Internet (Armbrust et al., 2009); de este modo, para los suscriptores o consumidores, que se perciben como agentes humanos, la vía natural de acceso ubicuo será el navegador Web.

Esta directriz puede ser explicada principalmente por el salto cualitativo en relación a las tecnologías Web que soportan la presentación de contenidos de manera eficiente y cada vez más interactivos como lo son HTML, JavaScript, CSS, XML, AJAX (Brockschmidt, 2014) y a los cuales los navegadores acceden de forma estándar.

De hecho, afirman Buyya et al., que *"cualquier persona con seguridad puede construir, desplegar y gestionar de manera simple, procesos de integración complejos, utilizando solamente un navegador Web"* (Buyya et al., 2010).

Como los navegadores, apoyados en las tecnologías Web, se han vuelto omnipresentes, esto los convierte en el medio idóneo para el acceso a las herramientas de PaaS, esencialmente porque, requieren poca o ninguna configuración adicional de Software del lado del cliente (Badger et al., 2012) también, por el esfuerzo de los proveedores, en brindar herramientas centradas en este paradigma de navegación, representando una ventaja apreciable.

Por otra parte, hay varios factores adicionales que contribuyen a darle valor a esta propuesta de acceso, por ejemplo, como describen Badger et al., a diferencia de las herramientas de desarrollo de Software tradicionales, generalmente empaquetadas y con gestión de dependencias (librerías, runtimes, máquinas virtuales, etc.), las aplicaciones sobre PaaS, se pueden acceder sin esperar procedimientos de instalación complejos (Badger et al., 2012).

Debido a que las aplicaciones de PaaS dejan muy pequeñas huellas en los equipos cliente que las acceden (emplean el caché del navegador), el riesgo de interferencia en la configuración entre las aplicaciones del proveedor y los equipos del suscriptor es reducido (Badger et al., 2012).

Aunado a esto, los costos de distribución y uso para el Software desarrollado en PaaS y posteriormente desplegado en SaaS, se reducen drásticamente puesto que, al disminuir los costos de utilización de las herramientas, se permite el desarrollo económico y el despliegue de Software empleando una fracción del costo pagado por la utilización de los servicios (Badger et al., 2012).

3.4.2. Gestión de Incidencias por el Proveedor

En general en los servicios PaaS contratados a los proveedores, los suscriptores no necesitan involucrarse con la gestión de la infraestructura subyacente que se les provee (Blacharski & Landis, 2013) ni entrar en contacto con tareas de orden operativo o funcionales.

En ese sentido, por ejemplo, los suscriptores no necesitan apoderarse de decisiones técnicas sobre cual sistema operativo emplear, que dispositivos de Hardware u opciones de configuración asignar, como realizar la gestión y configuración de versiones para bibliotecas específicas de Software que subyacen como parte de PaaS o intimidades relacionadas con aspectos circundantes como la seguridad (Badger et al., 2012).

En particular, como se revisó anteriormente en este trabajo (Sección 3.3, *Figura 8*), los proveedores tienen total responsabilidad de las cuestiones operativas (Marston et al., 2011) (Badger et al., 2012) tales como las copias de respaldo, el mantenimiento del sistema, la aplicación de parches de seguridad, la gestión de energía, actualizaciones de Hardware, seguridad de la planta física entre otros.

Por su parte, los suscriptores no están obligados a mantener en las instalaciones propias, personal de soporte en TI para llevar a cabo tareas relativas a solventar problemas o dar mantenimiento sobre los servicios contratados (Badger et al., 2012), salvo la excepción de que, en relación a esto, es todavía necesario conservar una conectividad constante y segura a la red desde la cual acceden los navegadores Web a la plataforma y que es pleno ámbito de los suscriptores.

Evidentemente, se debe considerar que parte del dinero que los suscriptores pagan por el servicio, va a los gastos de personal del proveedor. Pero es típicamente una cantidad mucho menor que si se hace todo el trabajo *in situ*.

Debido a que los proveedores de PaaS se manejan con libertad para implementar nuevas características en los servicios de la plataforma que ofrecen y proporcionar el Hardware que los ejecuta de manera conveniente (Furht et al., 2010) (Badger et al., 2012), también obtienen ventajas, al tiempo que mitigan la necesidad de sus suscriptores, de mejorar sus sistemas de Hardware para utilizar las nuevas características.

Esta última prerrogativa es importante, puesto que como señalan Teixeira et al., en una arquitectura de computación en la nube como PaaS, los usuarios tratan de aprovechar al máximo las mejores y menos costosas características, que los proveedores disponibles les puedan ofrecer (Teixeira et al., 2014).

3.4.3. Facilidad de Desarrollo y Despliegue de Aplicaciones

Como se trató en secciones previas, existe una tendencia defendida por autores (Buyya et al., 2009) (Columbus, 2014) (Kepes, 2013) (Teixeira et al., 2014) y sostenida por estudios recientes (Bryant et al., 2015), que señalan que PaaS se convertirá en el enfoque predominante hacia el desarrollo de Software en años próximos.

Con la capacidad de fomentar la automatización de procesos de desarrollo (Kepes, 2013), así como de desplegar sistemáticamente y en entornos de producción, aplicaciones desarrolladas sobre PaaS (Badger et al., 2012), se propone una dimensión atractiva y con valor suficiente, para hacer de esta alternativa de programación, un enfoque determinante a los suscriptores de servicios que adoptan estas tareas.

Sobre esto, como sugieren Velte et al., se pueden incluir por ejemplo, las capacidades de agregar un sistema operativo bajo demanda, la posibilidad de crear e integrar bases de datos dinámicamente, implementar sistemas para la gestión colaborativa entre los desarrolladores y la habilitación de múltiples lenguajes para la construcción de componentes lógicos (Velte et al., 2010).

En esa orientación, PaaS ha tenido un gran impulso en términos de soluciones y esfuerzos de estandarización (Buyya et al., 2010), principalmente en relación a herramientas y entornos para el desarrollo de aplicaciones, como lo son middlewares, frameworks, librerías, lenguajes, API, entre otros, los cuales procuran suplir con una variedad de instrumentales técnicos, las tareas habituales a los desarrolladores.

El surtido y disponibilidad de éstos recursos enriquece no solo la variedad propia de los desarrollos posibles sobre la plataforma sino que además, brindan de manera proactiva ventajas a los desarrolladores en la resolución de la mayoría de los problemas de gestión de dependencias (Badger et al., 2012) y agilizan los procesos de desarrollo (Kepes, 2013) sobre el modelo PaaS, presentándolo como el entorno de desenvolvimiento de Software natural para la nube (Teixeira et al., 2014).

Por su parte, otro beneficio del uso de la *plataforma como servicio*, es la posibilidad de ejecutar múltiples aplicaciones dentro de la misma instancia de recursos contratados, a modo de compartir un modelo común de seguridad, datos y de interfaz de usuario. Esta es una ventaja importante que particularmente, se encuentra en las soluciones PaaS en la nube (Velte et al., 2010).

3.5. Consideraciones en la Implementación de Proyectos de Desarrollo de Software sobre PaaS

Abordar las consideraciones para la implementación de proyectos concretos de desarrollo de Software sobre PaaS, es un tópico que puede ser evaluado bajo criterios muy similares a los que se emplean en la selección de un proveedor de servicios en la nube (Kadakia, 2014) (Kepes, 2013), considerando la relación intrínseca entre ambos conceptos previamente tratada a lo largo de este esfuerzo investigativo.

En este sentido, durante el desarrollo de la investigación, literaturas revisadas tanto en Marston et al., (Marston et al., 2011) como en Armbrust et al., (Armbrust et al., 2009) permiten estimar que es posible examinar algunas de estas consideraciones desde dos puntos de vista válidos; un primer enfoque organizacional y otro de caracterización más técnica.

En relación a lo anterior, la primera perspectiva de los autores, sugiere orientarse en el aspecto de la organización que subscribirá los servicios, siendo esta serie de criterios, más próximos a los intereses estratégicos de las empresas o instituciones, que planean la integración de servicios en la nube, como apoyo a su cartera de operaciones (Marston et al., 2011) (Armbrust et al., 2009) (Phaphoom et al., 2015).

No obstante, en circunscripción al compendio que abarca el presente trabajo, estos criterios no serán enunciados, con el fin de centralizar el interés de esta sección, en los aspectos técnicos que se tratan, considerando que, los primeros pueden ser materia de trabajos futuros y específicos.

Así, retomando un sentido técnico y tal como anteriormente se ha revisado, PaaS es el entorno de desarrollo y despliegue para las aplicaciones en la nube (Velte et al., 2010) (Badger et al., 2012) que apoyan el portafolio de operaciones de organizaciones, empresas o instituciones y como tal, los proyectos de desarrollo deben orquestarse en función de éstas.

Una sugerencia importante en este sentido, es propuesta por Mark Geene quien señala que "*no se debe construir una aplicación de nube sin seleccionar primero un PaaS. De hecho, es necesario tomar esta decisión antes de seleccionar el proveedor de Infraestructura (IaaS) que acompañe la PaaS. La selección de PaaS, debe guiar su elección de hospedaje no al revés"* (Geene, 2012).

Lo anterior, tiene un sentido práctico valioso. Considerando que si la elección de una plataforma en la nube para desarrollo y despliegue de aplicaciones, se rige más por las ventajas de infraestructura que pueda aportar el proveedor, que por la variedad y flexibilidad de herramientas y prestaciones de servicios entorno al soporte del desarrollo, se hace pobre la dinámica de los proyectos y aumenta la reticencia en la adopción de ésta por parte de los desarrolladores (Cohen, 2013).

3.5.1. Elección de la Interfaz de Acceso

El NIST describe entre sus recomendaciones, la evaluación de proveedores PaaS que permitan un soporte genérico a interfaces de infraestructura para aplicaciones tales como por ejemplo, soporte a archivos, colas, tablas hash, etc.

Esta evaluación es un criterio considerable para este organismo de estandarización, puesto que las interfaces proveídas en la plataforma que se considera adoptar, de manera nativa o con relativa facilidad, deben ser lo suficientemente genéricas para apoyar la portabilidad e interoperabilidad de las aplicaciones una vez desplegadas (Badger et al., 2012).

Lo anterior puede ser relacionado con la recomendación aportada por Mark Geene, quien señala que se debe *determinar de antemano la importancia de la portabilidad de las aplicaciones* (Geene, 2012).

Sobre esto, Geene (Geene, 2012) argumenta al igual que Carlson (Carlson, 2013), que existen dos tipos de PaaS: *portables* y *verticalmente integradas*. Las ofertas de PaaS portables son independientes del proveedor de infraestructuras (IaaS) (Geene, 2012).

Ampliando, un PaaS portable es una plataforma construida para ejecutar aplicaciones en desarrollo sin necesidad de cambios significativos en la forma en que está escrito el código. Para los desarrolladores que han creado códigos funcionales en entornos de alojamiento compartido o dedicados, trasladar ese código a una *plataforma como servicio* no se hace particularmente difícil (Carlson, 2013).

En esa línea de categorización, señala Geene, las mejores ofertas son las plataformas de código abierto las cuales incluyen productos probados y relativamente maduros como Cloud Foundry, OpenStack, OpenShift, Heroku y CloudStack. Esencialmente, como valor agregado, todas cuentan con el respaldo de los principales proveedores y son portables entre nubes con modelos de despliegue públicos y privados (Geene, 2012).

Por su parte, las ofertas PaaS integradas verticalmente, combinan una IaaS con un PaaS como una misma plataforma por obviedad, se crea una interdependencia entre los recursos disponibles y los servicios ofrecidos, por lo que no son portables (Geene, 2012). Profundizando un poco más, esta categoría de PaaS permiten construir una aplicación escribiendo código dependiente de las especificaciones únicas y las API, generalmente propietarias de esa plataforma.

Esto significa que la estructura del código debe adherirse muy estrictamente, a un cierto estándar de desarrollo específico que conlleva incluso, a utilizar lenguajes especializados que se construyen sólo para esa plataforma (Carlson, 2013).

Estas ofertas incluyen productos como Force.com, Microsoft Azure, Google App Engine (Carlson, 2013) (Geene, 2012) y en un grado cada vez mayor, según indica Geene, Amazon Web Services (Geene, 2012).

3.5.2. Lenguajes y Herramientas Disponibles

En relación a las opciones de lenguajes y herramientas que se ofertan en PaaS, Geene propone la consideración de ciertos cuestionamientos al respecto, por ejemplo, deben ser planteadas preguntas como *¿Qué frameworks de desarrollo y lenguajes son compatibles con la plataforma ofrecida? ¿Apoyan las opciones de selección de PaaS, el entorno de desarrollo actual?* (Geene, 2012).

En este sentido, seria evidente que el soporte a los lenguajes, API, frameworks, librerías y herramientas ofrecidas por el proveedor de PaaS, puedan ser ajustados a suplir las necesidades del tipo de aplicaciones que el suscriptor pretende desarrollar. No obstante, esto no es posible todo el tiempo.

Como se mostró al inicio del capítulo (Sección 3.1, *Gráfico 4*), las tendencias de desarrollo de Software sobre entornos PaaS, están potenciadas principalmente hacia proyectos Web.

Esto podría interrelacionarse con la exposición de Armbrust et al., Chee y Curtis, quienes indican que el soporte a lenguajes de alto nivel como los lenguajes Web (Ruby, PHP, etc.) es preferido para estos entornos, dada la naturaleza de éstos de abstraer aspectos que serían tediosos y en particular, sumamente complejos de manejar en desarrollos con otros lenguajes de bajo nivel como por ejemplo, lenguaje C o ensamblador (Armbrust et al., 2009) (Chee & Curtis, 2010).

Un aporte al respecto, podría ser derivado del análisis de las plataformas más populares de PaaS, las cuales muestran cierta preferencia de los proveedores a ofrecer un amplio surtido de herramientas y lenguajes, es decir, PaaS multilenguaje o *"poliglotas"*. Este indicio, es evidenciado por el portal Web http://www.paasify.it/ que muestra dicha tendencia ilustrada en el **Gráfico 6**.

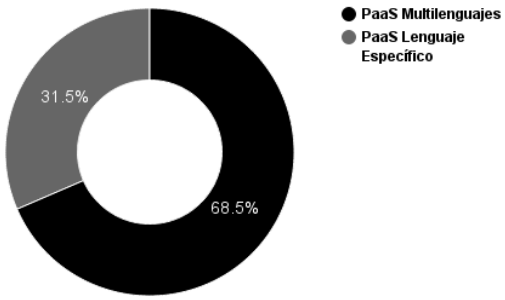

Gráfico 6. Soporte PaaS a Lenguajes Específicos vs PaaS Políglotas [Fuente: Kolb, 2015].

Como indicado en el gráfico, existe en la cuota de proveedores, una distribución porcentual del 31.5% de ofertas PaaS disponibles para lenguajes y herramientas específicas lo que podría suponer el uso de PaaS integradas verticalmente (Geene, 2012) (Carlson, 2013).

Por otra parte, existe un 68.5% de apoyo a PaaS con conjuntos más amplios y diversificados de herramientas y lenguajes (Kolb, 2015). La fuente consultada, aporta además, la frecuencia porcentual de soporte a lenguajes de programación en las PaaS, los cuales en orden descendente, se ilustran en el **Gráfico 7**.

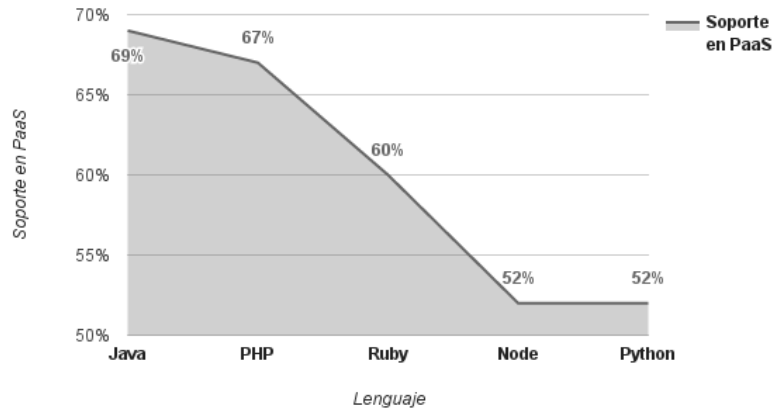

Gráfico 7. Distribución porcentual de lenguajes soportados por PaaS [Fuente: Kolb, 2015].

Los valores respectivos destacan cinco lenguajes como los más implementados, Java (69%), PHP (67%), Ruby (60%), Node (52%) y Python (52%) (Kolb, 2015). Debe notarse como interesante, que en su mayoría son lenguajes y tecnologías de desarrollo abiertas y en general, tipificadas para desarrollo Web de manera nativa (PHP, Ruby, Node) o mediante extensiones especiales (Python, Java) (Kolb, 2015).

Esto apoyaría lo sugerido por el NIST, que deben priorizarse la selección de sistemas PaaS que trabajen con lenguajes estandarizados y herramientas abiertas, a menos que las únicas opciones prácticas sean ofertas PaaS que estén restringidas a lenguajes propios y herramientas exclusivas (PaaS integradas verticalmente) (Badger et al., 2012). Finalmente, el **Gráfico 8** expresa la incidencia porcentual de otros lenguajes de desarrollo implementados en plataformas PaaS.

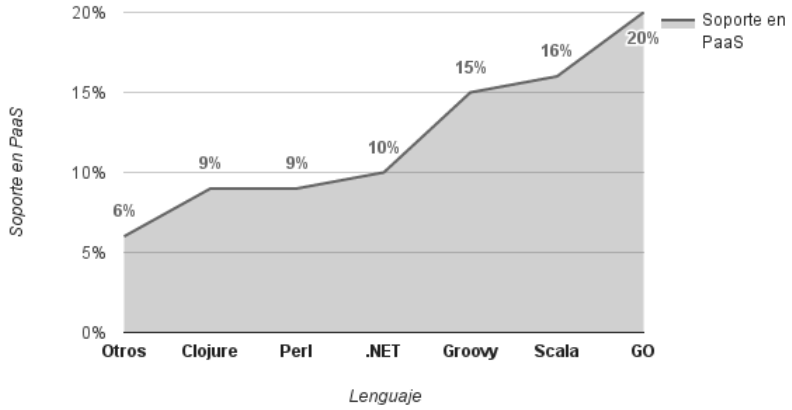

Gráfico 8. Distribución porcentual de otros lenguajes soportados en PaaS [Fuente: Kolb, 2015].

3.5.3. Acceso y Protección a los Datos

Particularmente, ningún sistema está exento de fallas. Siempre existen posibilidades de que los datos generados durante el desarrollo de un Software (archivos, compilaciones, etc.) y principalmente, en la puesta en producción de las aplicaciones derivadas, puedan perderse para siempre si no se realiza un protocolo de respaldo y restauración de datos proactivo a las eventualidades.

En concreto, al trasladar los proyectos de desarrollo de Software a la nube, se deben evaluar no sólo los mecanismos estándares y tradicionales que protegen la integridad de los sistemas y los datos *per se* (respaldos, recuperación), pero además, aquellos que procuran mantener la seguridad y la privacidad de éstos, dentro del contexto.

Una interesante concepción propuesta por Mark Geene, enuncia que una buena oferta de PaaS debe proporcionar servicios esenciales cómo; el *desarrollo de aplicaciones, acceso a bases de datos*, la *capacidad de integración* y primordialmente, un *soporte demostrable a servicios de seguridad sobre los datos* (Geene, 2012).

Sobre esta última prestación enunciada, puede argumentarse que, aunque en términos técnicos (soporte a herramientas, lenguajes, librerías y otros), existe una amplia y creciente disponibilidad de ofertas PaaS, aspectos como las copias de seguridad y la restauración de datos, merecen una especial consideración al momento de elegir un proveedor de PaaS (Linthicum, 2011).

Un estudio estadístico llevado a cabo en 2013 por la compañía *Enterprise Strategy Group* (http://www.esg-global.com), reveló que en el conjunto de organizaciones evaluadas, un 64% consideraban como vital, contar con mecanismos de respaldo y continuidad de operaciones sobre sus datos y aplicaciones, un 24% sólo empleaban un plan de recuperación y un restante 12% de los encuestados, no aplicaban esta práctica, según expone el **Gráfico 9**.

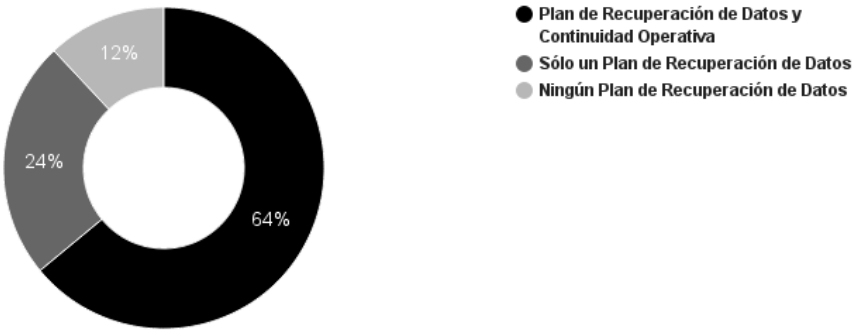

Gráfico 9. Distribución porcentual de prácticas organizacionales sobre recuperación y continuidad operativa de datos y aplicaciones [Fuente: ESG, 2013].

Lo anterior, indica que existe un priorización por parte de la mayoría de las organizaciones, en hacerse de formas sistemáticas de reponer sus datos y aplicaciones ante incidencias críticas que los comprometan.

Por otra parte, sobre el contexto de implementación de estos mecanismos, el estudio demostró que aproximadamente un 75% de las organizaciones, persistían en realizar estas tareas *in situ* es decir, en sus instalaciones físicas, por cuanto que el 25% tercerizan estos servicios, según muestra el **Grafico 10**.

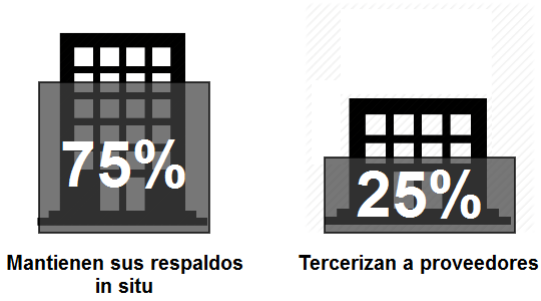

Gráfico 10. Distribución porcentual de prácticas organizacionales en relación a servicios de respaldo sobre datos [Fuente: ESG, 2013].

El estudio adicionalmente, exploró la percepción de las organizaciones en el uso de recursos en la nube para el respaldo de aplicaciones y proyectos de Software, exponiendo una tendencia interesante, ilustrada en el **Gráfico 11**.

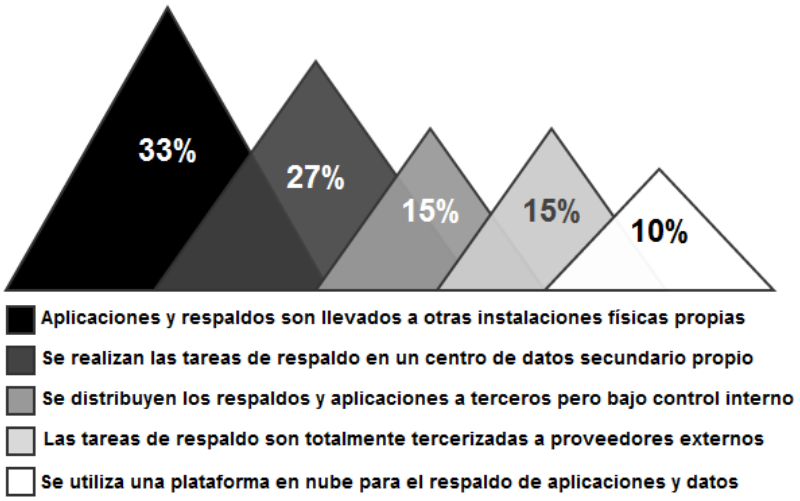

Gráfico 11. Percepción de las organizaciones en el uso de servicios de respaldo sobre datos [Fuente: ESG, 2013].

Como se percibe del **Gráfico 11**, el índice porcentual de organizaciones que están dispuestas a dejar en total responsabilidad del proveedor, el servicio de respaldo y recuperación de sus aplicaciones y datos generados está sobre el 10% decantándose aún por alternativas tradicionales o históricas (respaldo en sitio, subcontratación supervisada, etc.) las cuales asumen un porcentaje alto.

Estas son consideraciones oportunas muy importantes, dado que influyen en la forma en que pueden desarrollarse proyectos sobre PaaS esencialmente porque, si bien fue demostrado en las secciones *2.3.1* y *3.1* que las organizaciones manifiestan un movimiento hacia este paradigma por las ventajas que le ofrece en términos técnicos y económicos, conservan aún mucha cautela sobre las propiedades de respaldo y restauración como parte de los servicios PaaS.

La tendencia, podría interpretarse en dos sentidos, por una parte, podría denotarse una reticencia de las organizaciones a confiar sus datos generados, archivos fuentes y otros, a un tercero por cuestiones de seguridad y privacidad o, por otro lado, las ofertas del mercado en esa área, no aportan las garantías adecuadas a sus suscriptores (Phaphoom et al., 2015).

De esta manera, autores como Buyya et al., indican considerar proveedores con capacidades logísticas para suministrar el alojamiento de aplicaciones en varios lugares geográficamente distintos del mundo, tal que permitan la redundancia y garanticen la fiabilidad en caso de fallas (Buyya et al., 2009).

Un acercamiento es la proposición del NIST, la cual sugiere inicialmente, el análisis de los mecanismos de protección de datos ofrecidos por los proveedores PaaS, la ubicación de los datos recopilados por las aplicaciones (en términos físicos y geográficos), las configuraciones y tecnologías de base de datos empleadas para el procesamiento de las transacciones en completitud con las necesidades de confidencialidad, cumplimiento, integridad y disponibilidad de la organización que suscribe el servicio (Badger et al., 2012).

Los abordajes anteriores pueden ser genéricos y en cierto modo, distantes a ciertos movimientos del ámbito real que se han evaluado en este esfuerzo de investigación. No obstante, pueden ser usados de base y traducidos a términos específicos por el suscriptor como parte de sus necesidades de negocio, los cuales pueden ser documentados e incluidos en el SLA según las acotaciones revisadas por Wu y Buyya (Wu & Buyya, 2010).

En relación a esto, Grimes (Grimes, 2015) propone aportar una serie de cuestionamientos concretos a los proveedores evaluados, en base a conocer sus directrices respecto a por ejemplo, los datos y la frecuencia de respaldo que realizan, los mecanismos de acceso a las copias de seguridad, el cifrado o no, de los datos de respaldo, el tiempo que el proveedor mantiene los datos y dónde los almacenan, las técnicas que utiliza el proveedor para restaurar los datos, la frecuencia de comprobación de las restauraciones, la forma de asegurar la integridad de los datos de prueba, entre otros.

CAPÍTULO – IV

Problemas Abiertos

Esta sección del trabajo, expone de manera descriptiva y general, algunos de los problemas abiertos que fueron identificados en la investigación y que están relacionados con la implementación de PaaS como entorno de desarrollo de Software. Se discuten brevemente sus causas e implicaciones, como marco de referencia para trabajos futuros.

4.1. Seguridad de las Aplicaciones

Como se ha revisado durante el presente trabajo de investigación, en comparación con las soluciones de Software tradicionales, en la *plataforma como servicio* (PaaS), una organización puede crear aplicaciones personalizadas sin la sobrecarga de adquirir y gestionar una infraestructura subyacente completa (Kepes, 2013) (Blacharski & Landis, 2013).

Si bien lo anterior puede estimular múltiples ventajas para el suscriptor, sus desarrolladores y los proyectos de Software que pretenda llevar a cabo, puede por otro lado, conducir a serios conflictos relativos a la operatividad de las aplicaciones resultantes.

En primera instancia, debe considerarse que PaaS como cualquier plataforma tecnológica ofrece preocupaciones de seguridad, puesto que como fue revisado previamente (Sección 3.3, *Figura 8*), muchas de las características de implementación subyacentes están fuera del control del suscriptor al situarse en las capas inferiores de la pila (Badger et al., 2012).

En ese sentido, muchos proveedores utilizan una solución sencilla para evitar posibles problemas de seguridad en la IaaS, la cual consiste en cifrar el acceso a las capas de abstracción que acceden a los dispositivos antes de que se coloquen en la nube, esto al menos, garantiza de parte del proveedor cierta impermeabilidad a incidencias o ataques posteriores (SearchCloudSecurity, 2015).

De hecho, señalan Buyya et al., este acercamiento proviene de la propuesta de otros investigadores quienes alegan que la seguridad en la nube, no es muy diferente de las prácticas de seguridad existentes y que los aspectos de seguridad pueden ser bien administrados utilizando técnicas existentes, tales como firmas digitales, cifrado, cortafuegos, y / o el aislamiento de los entornos virtuales, entre otros (Buyya et al., 2010).

No obstante, como propone el NIST, para el caso de PaaS se emplea un modelo de interacción dinámica (Badger et al., 2012) entre sus componentes técnicos y los actores que tienen acceso a éstos (desarrolladores, clientes, etc.), lo que evidentemente, fomenta un ambiente para el acceso y procesamiento de datos de manera constante.

En estas condiciones, los entornos de PaaS deben permitir acceder, modificar y almacenar datos con cierta concurrencia, lo que conllevaría a un descifrado y re-codificación constante de toda la información generada (peticiones, acceso, respuesta), introduciendo así, problemas de gestión de claves (SearchCloudSecurity, 2015) (Ryan, 2013).

Bajo las consideraciones anteriores, el enfoque de cifrar los datos no es óptimo para la mitigación de los problemas de seguridad con las aplicaciones construidas y desplegadas sobre PaaS principalmente, porque éstas dependen de un componente lógico asociado a los procesos de depuración y corrección de errores, los cuales son inherentes a la programación de las aplicaciones desarrolladas por ende, del dominio exclusivo del suscriptor y por extensión, de sus programadores.

De esta forma, la seguridad de las aplicaciones en entornos PaaS podría tomarse de manera bidireccional, puesto que, como se argumenta una parte recae sobre los servicios ofrecidos por el proveedor y los cuales éste tiende a asegurar, por cuanto que el resto, es responsabilidad de los desarrolladores del suscriptor.

Sobre esto, más que los otros modelos de servicios ofrecidos en la nube, la seguridad en PaaS, requiere experiencia en seguridad de aplicaciones (Moyle, 2015) es decir, una cierta capacidad de introducir concretamente, diseños e implementaciones de pruebas específicas dentro del ciclo de desarrollo, las cuales permitan la reducción de vulnerabilidades sobre el Software generado.

Una técnica propuesta por Ed Moyle y que apoya directamente este concepto, es el modelado de *riesgo de amenazas de aplicación* (Application Threat Modeling), sugiriendo la implementación de los métodos propuestos por el Open Web Application Security Project (OWASP) (Moyle, 2015).

En una definición del OWASP, un modelo de riesgo de amenazas de aplicación, es un proceso esencial para el desarrollo de aplicaciones Web seguras, el cual permite a las organizaciones determinar los controles adecuados y producir contramedidas efectivas dentro de sus procesos de desarrollo de Software (OWASP, 2015). Un compendio ampliado de esta metodología y sus distintas etapas procedimentales, puede ser encontrado en: https://www.owasp.org/index.php/Threat_Risk_Modeling.

Otra sugerencia, es apoyar las actividades de pruebas durante el desarrollo, mediante el *escaneo de aplicaciones*. Una técnica por la cual herramientas de exploración de aplicaciones Web, examinan la estructura funcional de éstas, en busca de vulnerabilidades de seguridad conocidas, como por ejemplo *Cross-Site Scripting* (XSS), inyecciones SQL, ejecución de comandos remotos (Buyya et al., 2010), configuraciones inseguras de servidor entre otros (Moyle, 2015) (OWASP, 2015).

Un tópico final y tal vez, el más delicado de abordar en relación a la seguridad de las aplicaciones sobre PaaS, es el componente humano. Esencialmente, porque es crítico que los desarrolladores de aplicaciones estén versados en los principios de seguridad de aplicaciones.

Para Moyle, esto puede incluir la formación a nivel de lenguaje es decir, el conocimiento en profundidad y la aplicación de principios de codificación segura, que se emplean en relación con el lenguaje utilizado para construir las aplicaciones, así como temas más amplios, de diseño de seguridad y uso de herramientas específicas (Moyle, 2015), lo que puede ser altamente costoso en términos de tiempo y *praxis* para los desarrolladores.

4.1.1. Riesgos Basados en el Navegador Web

Debido a su ubicuidad, los *navegadores Web* son un elemento clave para el acceso de los clientes a los servicios de computación en la nube (Mell & Grance, 2011), siendo de esta manera, uno de los enfoques más convenientes e importantes relacionados con la seguridad en ella (Buyya et al., 2010).

Si bien, los clientes también pueden implicar pequeñas aplicaciones ligeras que se ejecutan en el escritorio o dispositivos móviles para acceder a los servicios (Chandrasekaran, 2015) (Carlson, 2013) (Sosinsky, 2011) en última instancia, ha sido el navegador Web el medio más común para hacerlo.

De esta manera, es a través de los navegadores Web, que los proveedores ofrecen y distribuyen herramientas del lado del cliente para la administración de la nube. Los navegadores Web, también se utilizan para la configuración y administración de los recursos, incluyendo la provisión de información necesaria para abrir y utilizar una cuenta con el proveedor (Badger et al., 2012).

No obstante, los navegadores son piezas de Software complejas rivalizando con la complejidad de los primeros sistemas operativos, demostrando a su vez, albergar fallas de seguridad y vulnerabilidades críticas (Badger et al., 2012) (Frei, Duebendorfer, Ollmann, & May, 2008).

En ese sentido, el problema se potencia debido a que los proveedores deben operar con una diversidad de navegadores Web de sus suscriptores, bajo la consideración que los sistemas y los navegadores administrados por éstos últimos, pueden no ser gestionados adecuadamente para satisfacer requisitos de seguridad o pueden no estar actualizados.

Considerando que en su gran mayoría, los navegadores Web cumplen una función general de acceso a contenido variado, apoyan su operación en el uso de *plugins* y extensiones de terceros, los cuales son conocidos por sus problemas de seguridad, falta de actualizaciones automáticas y la persistencia de vulnerabilidades (Frei et al., 2008) (Jansen & Grance, 2011), lo que induce un incremento en la posibilidad de que éstos, al ser subvertidos, comprometan ciertas operaciones delicadas en relación a los servicios, el conjunto de datos y aplicaciones confiadas al proveedor y a los que se acceden en la nube (Badger et al., 2012).

En ese sentido, investigaciones como las de Frei et al., Buyya et al., Jansen y Grance, señalan que cuando cualquier vulnerabilidad de seguridad Web se identifica, el atacante empleará distintas técnicas para aprovecharlas, clasificándose éstos en ataques de autenticación, autorización, ataques del lado del cliente, divulgación de información y los ataques lógicos, estos últimos dirigidos sobre las aplicaciones que proveen los servicios (Buyya et al., 2010) (Jansen & Grance, 2011) (Frei et al., 2008).

4.1.2. Dependencia de Conectividad a Red

Uno de los *Atributos de Servicio* propuesto como estándar del NIST y esencial para el modelo computacional en la nube, es el del *acceso a red*; el cual consiste en reunir los recursos informáticos del proveedor vía IaaS, para servir a múltiples consumidores, sobre una red desde la cual puedan acceder los distintos servicios ofrecidos (Mell & Grance, 2011) como por ejemplo, el desarrollo y despliegue de aplicaciones sobre PaaS.

En ese sentido, los suscriptores siendo clientes, necesitan trabajar sobre las garantías de una red segura para acceder a los servicios. Si la red que se le provee no es confiable, la nube no será fiable desde el punto de vista del suscriptor (Badger et al., 2012).

Dicho lo anterior, podría pensarse *a priori* que la integridad en el acceso a los servicios recae sobre la red del proveedor, si bien este es enteramente responsable de ofertar una infraestructura apropiada en términos de seguridad y continuidad (Badger et al., 2012), es oportuno considerar también, que deben existir del lado del suscriptor ciertas garantías al respecto.

Retomando el tópico que se aborda, el principal problema en el rendimiento de la conectividad de la red, puede surgir de la percepción que desde el punto de vista del suscriptor, puedan presentar algunas aplicaciones en cuanto a la completitud de ciertas tareas intensivas orientadas a transacciones y el uso masivo de datos (Furht et al., 2010) bien como por ejemplo, tareas inherentes a un proceso de desarrollo de Software, como la depuración o el despliegue de las aplicaciones.

Respecto a esto, Furht et al., señalan que los usuarios que se encuentran a una larga distancia de los proveedores pueden experimentar alta latencia y retrasos en sus operaciones (Furht et al., 2010) esencialmente, por los múltiples direccionamientos de red a nivel regional o inclusive continental, que podrían tomarse desde el origen de suscriptor hasta el destino en el proveedor y viceversa.

Por otra parte, Blacharski y Landis sugieren que la latencia en la conectividad de la red puede minimizarse mediante la selección de un proveedor cuyo centro de datos, se encuentre en una región geográfica próxima al suscriptor (Blacharski & Landis, 2013).

Esta solución, plausible en principio podría rebatirse, dado que si bien problemas como la latencia, el control de transacciones y en particular, la seguridad y el cumplimiento normativo sobre la continuidad de la red, son de particular preocupación para ambos miembros (suscriptor, proveedor), puede subyacer la influencia de problemas secundarios como por ejemplo, la calidad del servicio (Quality of Service, QoS) ofrecida por los proveedores de Internet (ISP) en ambos extremos, o la mala atención sobre el mantenimiento de las infraestructuras físicas locales en ambas partes.

4.1.3. Ingeniería de Seguridad para Aplicaciones Desarrolladas bajo PaaS

El concepto de *Ingeniería de Seguridad* está asociado con la forma de desarrollar y mantener sistemas que puedan resistir los ataques maliciosos destinados a dañarlos directamente o a sus datos. Sin medidas de seguridad proyectadas con previsibilidad, es casi inevitable que los atacantes comprometan un sistema en red, por lo que se considera la Ingeniería de Seguridad, parte integral del campo relativo a la *Seguridad Informática* (Sommerville, 2007).

En un sentido más pragmático, abarcar la Seguridad Informática en la nube, pasa por todos los temas de la Ingeniería de Seguridad, incluyendo el diseño de arquitecturas seguras, la reducción al mínimo de las superficies expuestas a ataques, la protección contra el malware y la aplicación de controles de acceso entre otros (Ryan, 2013).

Llevada al contexto del desarrollo de aplicaciones utilizando PaaS como entorno de gestión del proceso Software, la Ingeniería de Seguridad se hace esencial para diseñar sistemas y aplicaciones robustas capaces de soportar los ataques externos y con cierta habilidad de recuperarse de dichas agresiones.

Esta última, es una consideración muy importante, dado que un desarrollador de aplicaciones PaaS debe gestionar un número de riesgos de seguridad mucho más alto (Badger et al., 2012) que si lo hiciera sobre una aplicación convencional.

En primer término, generalmente, una aplicación desarrollada sobre PaaS puede requerir el uso de múltiples lenguajes y formatos, como por ejemplo, HTML, Java, JavaScript, XML, .Net, y archivos de recursos Web (Badger et al., 2012) (Brockschmidt, 2014), los cuales deben acoplarse de manera harmónica para su ejecución conjunta.

Por consiguiente, una falla potencial en alguna de las tecnologías empleadas o el inadecuado tratamiento de las pruebas con las aplicaciones desarrolladas (Moyle, 2015) sería exponerse a un escenario crítico de vulnerabilidades que podrían ser explotadas maliciosamente.

Además de las tecnologías empleadas en el desarrollo de las aplicaciones PaaS, una Ingeniería de Seguridad debe contemplar el uso explícito de criptografía y cifrado de datos en operaciones sensitivas, bien como considerar interactuar con las características de presentación de los navegadores Web comunes, que proporcionan salida visual a los suscriptores y los cuales pueden ser objeto de riesgo (Frei et al., 2008).

Lo anterior es de relevancia manifiesta puesto que, a diferencia de un ataque o vulnerabilidad que potencialmente afecte solo un entorno aislado y recursos locales como en aplicaciones de escritorio tradicionales, sobre un entorno PaaS el ataque podría propagarse y comprometer intrínsecamente, las redes de acceso, las aplicaciones, servicios y datos del suscriptor (Badger et al., 2012).

4.2. La Interoperabilidad de Aplicaciones

El ámbito de la *Interoperabilidad* se refiere a los mecanismos de funcionalidad e integración entre diferentes plataformas de computación en nube y la conexión entre una plataforma particular y los sistemas locales de una organización. El objetivo principal de la interoperabilidad, es permitir una fluidez entre los datos sin fisuras y a través de diferentes plataformas de nube y entre una nube y aplicaciones locales (Dillon et al., 2010).

Si bien la intención es válida conceptualmente, existen aún ciertas limitaciones concernientes a las implementaciones particulares de cada proveedor principalmente, en aquellos que ofrecen plataformas en modo privado con librerías, API o frameworks propietarios (Dillon et al., 2010), lo que ha hecho de esta integración operativa un problema a solventar.

Una consecuencia inmediata de la falta de interoperabilidad, es la indisponibilidad de datos y el aumento en las complicaciones sobre la portabilidad de las aplicaciones entre los proveedores de nube (Jansen & Grance, 2011) y las tecnologías empleadas en el desarrollo.

Particularmente en PaaS, la portabilidad pasa a ser una preocupación para el desarrollo de aplicaciones, especialmente cuando las plataformas contratadas requieren lenguajes propietarios y entornos propios de ejecución y despliegue.

Esto, de manera inherente hace prohibitiva la capacidad de los suscriptores a elegir entre posibles alternativas ofertadas por otros proveedores, simultáneamente, reduce las posibilidades de optimizar los recursos de las aplicaciones y ajustarlas a diferentes niveles funcionales requeridos dentro de una organización (Dillon et al., 2010).

Incluso, cuando se utilizan lenguajes estándares como sugiere el NIST (Badger et al., 2012), las implementaciones de servicios de la plataforma pueden variar ampliamente entre los proveedores por ejemplo, los archivos gestionados por una plataforma o las estructuras de datos complejas, como las tablas hash, pueden no ser compatibles con las de otra (Badger et al., 2012). Existe además, el escenario en el cual si el proveedor se retira del negocio, se pierdan todas las aplicaciones y sus datos (Velte et al., 2010).

Por otra parte, los suscriptores que crean nuevas aplicaciones pueden mitigar los riesgos de portabilidad mediante la creación de interfaces generalizadas a los servicios de la plataforma en lugar de crear implementaciones especializadas para proveedores de plataformas específicas (Badger et al., 2012).

Esta estrategia, sin embargo, incurre en costos y tampoco aminora totalmente los riesgos, pues desde una interfaz general que esconde variaciones específicas del proveedor, es probable que se limite el uso de propiedades de valor añadido específicas ofrecidas por el proveedor, por lo que resultaría en una minimización de características para la aplicación que se desarrolla.

Conclusiones

Durante el desenvolvimiento de este trabajo investigativo, se ha procurado abarcar de manera general, los tópicos considerados más relevantes a la temática abordada. En esta sección final, se sintetizan las principales observaciones derivadas de este esfuerzo, con el fin de fomentar una serie de criterios objetivos, que permitan valorar la situación actual de los entornos PaaS como recursos plausibles en la realización de proyectos de desarrollo de Software.

Inicialmente, fue comprobado mediante la revisión conceptual de Buyya et al., (Buyya et al., 2010), Furht et al. (Furht et al., 2010) entre otros, que el modelo de computación en la nube, ha demostrado ser una innovación tecnológica fruto de la evolución de otros fundamentos computacionales que existieron.

La terminología computación en la *nube*, por otro lado, ofrece un compendio muchas veces mal empleado y ambiguo que en opinión de autores como Blacharski y Landis (Blacharski & Landis, 2013), expresa un amplio espectro de posibilidades debido a las propiedades como la intangibilidad, la ubicuidad y la simplicidad conceptual del modelo.

Una acotación propia de este esfuerzo de investigación, ha sido ofrecer una descripción general del concepto de computación en la nube, el cual se ha definido como - *un mecanismo dinámico para proporcionar un conjunto de recursos informáticos compartidos, los cuales incluyen aplicaciones, poder de cómputo, plataformas de almacenamiento, redes, infraestructuras de desarrollo y de implementación así como procesos de negocio* -.

Otro aspecto derivado de este estudio investigativo, ha sido la consideración del concepto de *servicio* como el eje primordial sobre el cual gira toda la complejidad y esencialmente, todo el engranaje tecnológico de la computación en la nube.

En relación al uso de la nube como plataforma de desarrollo mediante el empleo de modelos de servicios como PaaS, es notable que implica un profundo cambio en la metodología y técnicas tradicionales de desarrollo que, en muchos escenarios, aún no han sido asimiladas totalmente por las organizaciones y de manera concreta, por los desarrolladores.

PaaS como entorno de desarrollo conlleva la adopción de toda una nueva línea de pensamiento sobre la re-implementación de métodos de ingeniería de Software que resultaban efectivos y factibles, como por ejemplo, los ciclos de desarrollo de Software tradicionales.

Dicho cambio en la forma de hacer las cosas si bien no altera significativamente las tecnologías de fondo, como los lenguajes o herramientas, si genera una perspectiva sustancialmente distinta bajo las consideraciones de que en un contexto como el propuesto por PaaS, se deben adecuar no el *"que"* se hace, sino el *"cómo"* se hace.

Sobre esto, se pudo determinar que en gran medida la introducción de metodologías ágiles de desarrollo ayuda en la transición de los proyectos de desarrollo de Software utilizando PaaS, se aprecia también, el esfuerzo de proveedores de herramientas y soluciones de ofrecer entornos potentes y robustos apoyados en tecnologías Web enriquecidas como HTML, AJAX, JavaScript y otros.

Un aspecto interesante en el sentido de las aplicaciones que son desarrolladas en entornos PaaS, puede derivarse de los índices estadísticos aportados en el Capítulo III, que permiten visualizar un significativo porcentaje de aplicaciones Web que se realizan para este entorno.

De lo anterior, podría señalarse que dicho comportamiento podría estar asociado a la oferta de lenguajes y plataformas disponibles las cuales brindan de manera general, las prestaciones óptimas para este tipo de desarrollos.

Otra consideración es que, en la mayoría de los proveedores PaaS se suele optar por alternativas de lenguajes variadas y abiertas, como recomendado por el NIST (Badger et al., 2012) y comprobado por Kolb (Kolb, 2015).

Existen también, propuestas PaaS sustentadas por proveedores privativos que proponen un ámbito de confiabilidad muy alto, pero que técnicamente pueden ser restrictivos en relación a las capacidades de desarrollo, despliegue y compatibilidad de plataformas.

De manera general, como fue expuesto en el Capítulo II, la recepción de PaaS en el ámbito de las organizaciones es bastante aceptable y positiva, no obstante, como también se señaló en el Capítulo III, aún existen ciertas reticencias sobre una aceptación total de la propuesta por parte de los suscriptores de servicios.

Las reacciones en el sentido de no adoptar los servicios de desarrollo de Software ofrecidos sobre PaaS, pueden ser acotados esencialmente a ciertas resistencia de los desarrolladores por salir de los entornos con herramientas conocidas.

Por otra parte, el fenómeno se extiende a los *modus operandi* tradicionales de las organizaciones, en relación a la potestad de los métodos de respaldos y reintegración de datos, los cuales persisten en retener con cierta exclusividad.

Esto puede ser visto como una cautela en las garantías sobre la privacidad de sus datos, códigos fuentes y otros; o por otro lado, como un manejo proactivo ante las dudas que suelen ser no solventadas durante la evaluación de proveedores PaaS o no especificadas de manera concreta en los Acuerdos de Nivel de Servicios (*Service Level Agreement*, SLA).

En relación a las problemáticas abiertas tratadas en el Capítulo IV, es notorio evaluar que están por hacerse avances sobre éstas ya que de momento, las propuestas de solución ofrecidas son parciales y de manera substancial, mejorables en la práctica, esto abre la posibilidad de trabajos futuros al respecto.

Particularmente, los enfoques de interoperabilidad y seguridad, son aspectos críticos que deben ser madurados mediante nuevas técnicas y metodologías como parte del enfoque para el desarrollo de aplicaciones en el entorno PaaS, esencialmente por el ámbito y la trascendencia de las implicaciones asociadas al respecto.

Si bien la complejidad implícita que se identifica *a priori* como parte de un desarrollo sobre PaaS, puede conllevar el desarrollo de otras habilidades y conocimientos por parte de los desarrolladores, el grado de comprensión de sus conceptos, la apertura de ofertas y la competitividad de servicios que se brinden, serán una modalidad cuya integración en los procesos de desarrollo futuros, evolucionará hasta hacerse natural al nuevo paradigma.

Finalmente, es la consideración de esta investigación, concluir que si bien existen barreras técnicas y de reticencia humano-organizacionales en la adopción completa de PaaS como un entorno de desarrollo, existen indicadores irrefutables que es un movimiento creciente, que en la medida que se extienda solventando las actuales dificultades, se posicionará como un marco de referencia para proyectos de variada índole y con una diversidad de implementaciones en diferentes áreas, dejando de ser una promesa para hacer parte integral de los recursos tecnológicos de la Ingeniería de Software.

Trabajos Futuros

Como se describió en el Capítulo IV, existen algunos problemas interesantes que representan posibilidades de continuación en la investigación sobre el desarrollo de Software en entornos PaaS. Una primera línea en ese sentido, es profundizar en la *Interoperabilidad* de las aplicaciones desarrolladas.

Si bien este tópico ha sido abordado por otros autores, el problema central continúa a ser persistente, debido a la falta de una normalización que permita la operación de las aplicaciones desarrolladas sobre PaaS, en distintos entornos o plataformas de computación en la nube. (Machado, Hausheer, & Stiller, 2009) (Badger et al., 2012),

La estructuración de esta problemática estriba en su complejidad. Entendiendo como señalado por Machado et al., (Machado et al., 2009) que cada plataforma y proveedor, ofrecen distintas API de implementación.

Aunque existen iniciativas, propuestas y discusiones abiertas llevadas a cabo por organismos establecidos entre proveedores y suscriptores (Lewis, 2012), en el sentido de reglar y extender las características operativas entre plataformas, no hay consenso en la manera apropiada de hacerlo.

De este modo, diseñar y desplegar una aplicación sobre un entorno particular de PaaS en un primer momento y posteriormente, implementarla en otro ambiente concreto y distinto del primero, con la expectativa de rendimiento inicial, conlleva un esfuerzo enorme debido a las consideraciones técnicas que se plantean.

En ese sentido, por ejemplo, Wang et al., describen que un mecanismo de interoperabilidad debe llevar en cuenta los protocolos de comunicación de mensajes entre plataformas como XMPP (Extensible Messaging and Presence Protocol), la transmisión de los datos, la transferencia de las máquinas virtuales y de los runtimes de las aplicaciones (Wang, Ding, & Niu, 2012).

Acercarse a una solución global sobre este problema sería una manera idónea y potencialmente interesante, de mejorar y ampliar la versatilidad del modelo de desarrollo de Software propuesto por la nube principalmente, porque permitiría ofrecer un avance sobre este campo en particular tan demandado en la industria.

Una experimentación *a priori* que puede ser planteada como dirección de la investigación futura, podría ser el establecimiento de un análisis sobre las características comunes que deben y pueden ser estandarizadas.

Un apoyo importante sería partir de la clasificación preliminar aportada por el NIST, quienes definen 21 casos de uso para la interoperabilidad (Lewis, 2012). Esta selección por ejemplo, serviría de base o marco referencial para abstraer algunas tipologías interesantes y técnicamente realizables, que pudieran servir en la completitud de normas al respecto de la interoperabilidad.

De igual manera, un esfuerzo de investigación futuro, cabe sobre una revisión a los estándares en desarrollo que actualmente se proponen como por ejemplo, Open-SCA (Open-Service Component Architecture) de la iniciativa OASIS (http://www.oasis-opencsa.org/sca), USDL (Unified Service Description Language) de W3C para la definición de servicios o de OCCI (Open Cloud Computing Interface) (http://occi-wg.org/) y TOSCA (Topology and Orchestration Specification for Cloud Applications) de OASIS, para la gestión de interfaces.

Éstos podrían ser un punto de partida para la búsqueda y desarrollo de métodos de integración de aplicaciones sobre el modelo de computación en la nube. La idea sobre investigaciones en ese sentido, tendría que delimitarse a aspectos como por ejemplo, la integración de estos estándares desde una perspectiva técnica, la mejora sobre factores de optimización y principalmente, sobre la compatibilidad de los mismos en entornos de distintos proveedores.

Una conducción final de los estudios anteriores, permitiría llevar adelante la realización y publicación de una propuesta investigativa más ambiciosa a mediano plazo, como el *Desarrollo de un Modelo de Interoperabilidad Semántica para la Capa PaaS*.

Sobre esto, el ideario principal sería proponer un mecanismo centrado en la abstracción de algunas entidades referentes como las estructuras de las API y los metadatos requeridos, tal que se permitiera modelar la semántica pretendida. Un modelo emergente a esta propuesta, podría ser el uso de la semántica Web utilizando el lenguaje OWL (Web Ontology Language) normalizado por la W3C.

En el sentido práctico de aplicación, la propuesta podría acercar posturas usuales mediante una taxonomía y ontología común, lo que permitiría expresar el modelo computacional de la nube y sus partes, en términos de un modelo de datos consensuado.

Así, la semántica de los datos y de las acciones sobre éstos, permitiría derivar el vocabulario en el que estas acciones se expresan, pudiendo constituir el inicio de un lenguaje de computación inter-nubes, el cual podría ser aplicado a las API como parte de un esfuerzo más amplio por crear una interfaz o capa de abstracción única, definida y programable para todas las otras API de proveedores.

Si bien aún el proceso sugerido debe identificar problemas y conflictos relativos a este abordaje, en completitud podría ser una propuesta interesante que contribuya a resolver los conflictos de interoperabilidad que se susciten durante el despliegue o la migración de una aplicación desarrollada en PaaS sobre entornos de nube heterogéneos.

Referencias

Armbrust, M., Fox, A., Griffith, R., Joseph, A., & RH. (2009). Above the clouds: A Berkeley view of cloud computing. *University of California, Berkeley, Tech. Rep. UCB* , 07–013. Retrieved from http://scholar.google.com/scholar?q=intitle:Above+the+clouds:+A+Berkeley+view+of+cloud+computing#0

Badger, L., Grance, T., Patt-Comer, R., & Voas, J. (2012). DRAFT Cloud Computing Synopsis and Recommendations. *NIST Special Publication*, *146*, 86.

Bahsoon, R., Mistrík, I., Ali, N., Mohan, T. S., & Medvidović, N. (2013). The future of software engineering in and for the cloud. *Journal of Systems and Software*, *86*(9), 2221–2224. http://doi.org/10.1016/j.jss.2013.05.061

Bankinter, F. (2010). *Cloud Computing: La tercera ola de las tecnologías de la información* (Vol. 13). Fundación Bankinter. http://doi.org/10.1007/978-3-642-10665-1

Blacharski, D., & Landis, C. (2013). *Cloud Computing Made Easy*. Virtual Global, Inc.

Blackman, B., Beeming, G., Fourie, M., & Schaub, W.-P. (2015). *Managing Agile Software Projects with Microsoft Visual Studio Online Professional*. Redmond, Washington: Microsoft Press.

Brockschmidt, K. (2014). *Programming Windows Store Apps with HMTL, CSS and Javascript*. Redmond, Washington: Microsoft Press.

Bryant, D., Morgenthal, J., Haddad, C., Golden, B., & Wetherill, J. (2015). *The Dzone Guide to Cloud Development*. DZONE.

Buyya, R., Broberg, J., & Goscinski, A. (2010). *Cloud Computing: Principles and Paradigms*. Hoboken, New Jersey: John Wiley & Sons, Inc. Retrieved from http://books.google.com/books?id=S1NvRRd77rQC&pgis=1

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, *25*(6), 17. http://doi.org/10.1016/j.future.2008.12.001

Carlson, L. (2013). *Programming for PaaS*. Sebastopol, California: O'Reilly Media Inc.

Chandrasekaran, K. (2015). *Essentials of Cloud Computing*. Florida: Taylor & Francis Group.

Chee, B., & Curtis, F. (2010). *Cloud Computing:Technologies and Strategies of the Ubiquitous Data Center*. New York: CRC Press.

Cohen, B. (2013). PaaS: New opportunities for cloud application development. *Computer*, *46*(9), 97–100. http://doi.org/10.1109/MC.2013.323

Columbus, L. (2014). *Roundup Of Cloud Computing Forecasts And Market Estimates*. *FORBES - Financial Times*.

Diez, O., & Silva, A. (2014). Resilience of cloud computing in critical systems. *Quality and Reliability Engineering International*, *30*(3), 397–412. http://doi.org/10.1002/qre.1579

Dillon, T., Wu, C. W. C., & Chang, E. (2010). Cloud Computing: Issues and Challenges. *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, 27–33. http://doi.org/10.1109/AINA.2010.187

Dutta, M., Sethi, K. K., & Khatri, A. (2014). Web Based Integrated Development Environment. *International Journal of Innovative Technology and Exploring Engineering*, *3*(10), 56–60.

Frei, S., Duebendorfer, T., Ollmann, G., & May, M. (2008). Understanding the Web browser threat: Examination of vulnerable online Web browser populations and the "insecurity iceberg." *Proceedings of Defcon 16*, 10. Retrieved from http://www.defcon.org/images/defcon-16/dc16-presentations/defcon-16-frei-panel.pdf

Furht, B., Escalante, A., Jin, H., Ibrahim, S., Bell, T., Gao, W., … Wu, S. (2010). *Handbook of Cloud Computing*. (Springer, Ed.)*Handbook of Cloud Computing*. New York. http://doi.org/10.1007/978-1-4419-6524-0

Geene, M. (2012). 5 Steps To Selecting a Platform-as-a-Service (PaaS). *Cloud Elements*, 1. Retrieved from http://cloud-elements.com/

Grimes, R. (2015). Staying secure in the cloud. *InfoWorld DeepDive*, 9.

Jansen, W., & Grance, T. (2011). Guidelines on Security and Privacy in Public Cloud Computing. *NIST Special Publication*, *144*(7), 800–144. http://doi.org/10.3233/GOV-2011-0271

Kadakia, C. (2014). *SDLC: Still Critical under PaaS*.

Kepes, B. (2013). *Understanding the Cloud Computing Stack SaaS, Paas, IaaS*.

Kolb, S. (2015). Platform as a Service Provider Comparison. Retrieved May 8, 2015, from http://www.paasify.it/

Lewis, G. (2012). *The Role of Standards in Cloud- Computing Interoperability The Role of Standards in Cloud-Computing Interoperability*. Retrieved from http://www.sei.cmu.edu/reports/12tn012.pdf

Linthicum, D. S. (2011). Running your business in the clouds. *Cloud ComputingDeep Dive*, (June), 2–6. http://doi.org/10.1109/DEST.2010.5610668

Linthucum, D. (2014). *Why PaaS Growth is Disproportional to Other Sectors*.

Machado, G. S., Hausheer, D., & Stiller, B. (2009). Considerations on the Interoperability of and between Cloud Computing Standards. *G2C-Net Workshop: From Grid to Cloud Networks*, (Section 4), 1–4. Retrieved from http://www.csg.uzh.ch/publications/ogf27-g2cnet-discussion-cc-standards-finalversion.pdf

Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing - The business perspective. *Decision Support Systems*, *51*(1), 176–189. http://doi.org/10.1016/j.dss.2010.12.006

McMurtrey, M. (2013). A case study of the application of the stystems development life cycle (SDLC) in 21st century health care: Something old, something new? *Journal of Southern Association for Information Systems*, *1*(1), 14–25. Retrieved from http://quod.lib.umich.edu/cgi/p/pod/dod-idx/case-study-of-the-application-of-the-systems-development.pdf?c=jsais;idno=11880084.0001.103

Mell, P., & Grance, T. (2011). The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology. *Nist Special Publication*, *145*, 7. http://doi.org/10.1136/emj.2010.096966

Moyle, E. (2015). Five steps for achieving PaaS security in the cloud. Retrieved May 16, 2015, from http://searchcloudsecurity.techtarget.com/tip/Five-steps-for-achieving-PaaS-security-in-the-cloud

OWASP. (2015). Threat Risk Modeling - OWASP. Retrieved May 13, 2015, from https://www.owasp.org/index.php/Threat_Risk_Modeling

Phaphoom, N., Wang, X., Samuel, S., Helmer, S., & Abrahamsson, P. (2015). A survey study on major technical barriers affecting the decision to adopt cloud services. *Journal of Systems and Software*, *103*, 167–181. http://doi.org/10.1016/j.jss.2015.02.002

Ryan, M. D. (2013). Cloud computing security: The scientific challenge, and a survey of solutions. *Journal of Systems and Software*, *86*(9), 2263–2268. http://doi.org/10.1016/j.jss.2012.12.025

SearchCloudSecurity. (2015). An examination of PaaS security challenges. Retrieved May 6, 2015, from http://searchcloudsecurity.techtarget.com/tip/An-examination-of-PaaS-security-challenges

Singh, S., & Chana, I. (2013). Introducing Agility in Cloud Based Software Development through ASD. *International Journal of U- and E- Service, Science and Technology*, *6*(5), 191–202. http://doi.org/10.14257/ijunesst.2013.6.5.17

Sommerville, I. (2007). *Software Engineering 8th* (8th ed.). Edinburgh: Addison-Wesley.

Sosinsky, B. (2011). *Cloud Computing Bible*. Indianapolis, Indiana: Wiley Publishing Inc.

Sotomayor, B., Montero, S., Llorente, I. M., & Foster, I. (2009). An Open Source Solution for Virtual Infrastructure Management in Private and Hybrid Clouds. *Internet Computing, IEEE*, *13*(5), 14–22. http://doi.org/http://dx.doi.org/10.1109/MIC.2009.119

Teixeira, C., Pinto, J. S., Azevedo, R., Batista, T., & Monteiro, A. (2014). The building blocks of a PaaS. *Journal of Network and Systems Management*, *22*(1), 75–99. http://doi.org/10.1007/s10922-012-9260-2

Vazquez-Poletti, J. L., Moreno-Vozmediano, R., Montero, R. S., Huedo, E., & Llorente, I. M. (2013). Solidifying the foundations of the cloud for the next generation Software Engineering. *Journal of Systems and Software*, *86*(9), 2321–2326. http://doi.org/10.1016/j.jss.2013.05.063

Velte, A. T., Velte, T. J., & Elsenpeter, R. (2010). *Cloud Computing: A Practical Approach*. *Journal of the Electrochemical Society*. McGraw-Hill. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0

Wang, J. K., Ding, J., & Niu, T. (2012). Interoperability and Standardization of Intercloud Cloud Computing. *CoRR*, 4. Retrieved from http://arxiv.org/pdf/1212.5956v1.pdf

Wu, L., & Buyya, R. (2010). Service Level Agreement (SLA) in Utility Computing Systems. *arXiv:1010.2881*. Retrieved from http://arxiv.org/abs/1010.2881\nhttp://www.arxiv.org/pdf/1010.2881.pdf