Diseño de una herramienta informática basada en la metodología Scrum para la gestión del desarrollo de software

M.C. José Francisco Gazga Portillo[1], Ing. José Raúl López Morales[2],
M.T.I. Juan Miguel Hernández Bravo[3] y M.I.D.S. Alma Delia de Jesús Islao[4]

***Resumen*—En este artículo, se plasma el trabajo interdisciplinario de la Maestría en Sistemas Computacionales con apoyo del CONACyT, impartida en el Instituto Tecnológico de Acapulco. El artículo tiene por meta, presentar el diseño utilizando la notación UML (Lenguaje de Modelado Unificado) que dará pauta al posterior desarrollo y codificación de una aplicación web, que permita la gestión y monitorización dedicada para proyectos de desarrollo de software basado en la metodología Scrum. Se hace mención a que este artículo forma parte de una serie de trabajos siendo el segundo de estos, el cual, da seguimiento a un artículo previo titulado: Propuesta de una herramienta basada en la metodología Scrum para la gestión del desarrollo de software, escrito por los mismos autores y que fue presentado y publicado en el congreso de Academia Journals Morelia 2019.**

***Palabras clave*— UML, Metodología Scrum, Modelado de procesos de negocio, Seguimiento del sprint.**

Introducción

El diseño de sistemas es el proceso para desarrollar modelos abstractos de un sistema, donde cada modelo presenta una perspectiva diferente de dicho sistema. En general, el modelado de sistemas se ha convertido en un medio para representar el sistema usando algún tipo de notación gráfica, que casi siempre se basa en notaciones en el **Lenguaje de Modelado Unificado** (UML, en inglés Unified Modeling Languaje). Los modelos se usan durante el proceso de ingeniería de requerimientos para ayudar a derivar los requerimientos de un sistema, también durante el proceso de diseño para describir el sistema a los ingenieros que implementan el sistema, y después de la implementación para documentar la estructura y la operación del sistema (Sommerville, 2016). Una ventaja que ofrece UML es desarrollo de aplicaciones globales para la Web, no sólo para comercio electrónico. Además, permite a los desarrolladores modelar sus aplicaciones Web como parte de un sistema completo y la lógica de negocios que se debe reflejar en las aplicaciones (Rumbaugh, Jacobson, & Booch, 2000). UML soporta la creación de diferentes tipos de modelado de sistemas, sin embargo, sólo cinco tipos de diagramas, pueden ser considerados esenciales para la representación de un sistema, estos son **los diagramas de actividad**, **de caso de uso**, **de secuencia**, **de clase** y **de estado** (Erickson & Siau, 2007). A pesar que UML provee una diversidad de diagramas, este lenguaje gráfico no cuenta con diagramas para modelar la parte de persistencia de datos (base de datos relacional) o modelar procesos en una organización para tener una perspectiva en que procesos estará involucrado un sistema a desarrollar. Para solventar tal inconveniente, se recurre al uso del **Modelo y Notación de Procesos de Negocio** (BPMN, en inglés Business Process Model and Notation) es una notación que permite el modelado de procesos de negocio, en un formato de flujo de trabajo. El BPMN se especifica como un lenguaje de propósito general, lo que significa que los conceptos de modelado son elementos genéricos en el contexto de los procesos de uso, sin ninguna característica de dominio, esto quiere decir que no está restringido a un sólo dominio (Braun & Schlieter, 2014).

Por otra parte, la base de datos es un componente del sistema de información que almacena datos del negocio, y en el ciclo de vida del desarrollo de software, la base de datos no se muestra después del proceso de recopilación de requisitos. El diseñador de base de datos tiene la responsabilidad de diseñar un diagrama denominado **Entidad-Relación** (ER, inglés Entity-Relationship) está basado en el proceso de negocio del usuario. El diagrama ER describe cualquier entidad u objeto relacionado con el sistema. Posteriormente, el diagrama ER se traduce al Modelo de Datos Conceptuales (CDM) y finalmente al Modelo de Datos Físicos (PDM) (Simanjuntak, 2015).

[1] M.C. José Francisco Gazga Portillo es docente del programa de Maestría e Ingeniería en Sistemas Computacionales del Instituto Tecnológico de Acapulco, ita.gazga@gmail.com
[2] Ing. José Raúl López Morales es estudiante de la Maestría en Sistemas Computacionales en un programa PNPC en el Instituto Tecnológico de Acapulco, jraul_lopz@hotmail.com (**autor corresponsal**)
[3] M.T.I. Juan Miguel Hernández Bravo es docente del programa de Maestría e Ingeniería en Sistemas Computacionales del Instituto Tecnológico de Acapulco, jmhernan@yahoo.com
[4] M.I.D.S. Alma Delia de Jesús Islao es docente del programa de Maestría e Ingeniería en Sistemas Computacionales del Instituto Tecnológico de Acapulco, alma.islao.ita@gmail.com

Objetivo general

El objetivo principal que se pretende lograr con la presente propuesta, consiste en desarrollar una herramienta informática que permita la gestión de proyectos de desarrollo de software que basan su construcción, en el empleo de la metodología Scrum, para ello es importante realizar el diseño, haciendo uso de la notación UML, para dar soporte al desarrollo de la herramienta propuesta.

Diseño de la herramienta propuesta

A partir de esta sección, se presentan tres tipos de diagramas UML, los cuales son diagramas de casos de uso, de clase y de despliegue; donde este último a pesar de no estar entre los cinco diagramas esenciales, se utiliza para modelar la estructura general de la herramienta donde se mostrará la interacción de los diferentes dispositivos y sus componentes para tener acceso a dicha herramienta. Además, se presentan los procesos de la propia metodología Scrum, utilizando el diagrama de modelado de negocio, esto permite tener una mejor comprensión de los procesos que se llevan a cabo durante la implementación de la metodología Scrum, y a su vez, saber en qué procesos estará involucrada la herramienta a desarrollar.

Modelado de Procesos de Negocio

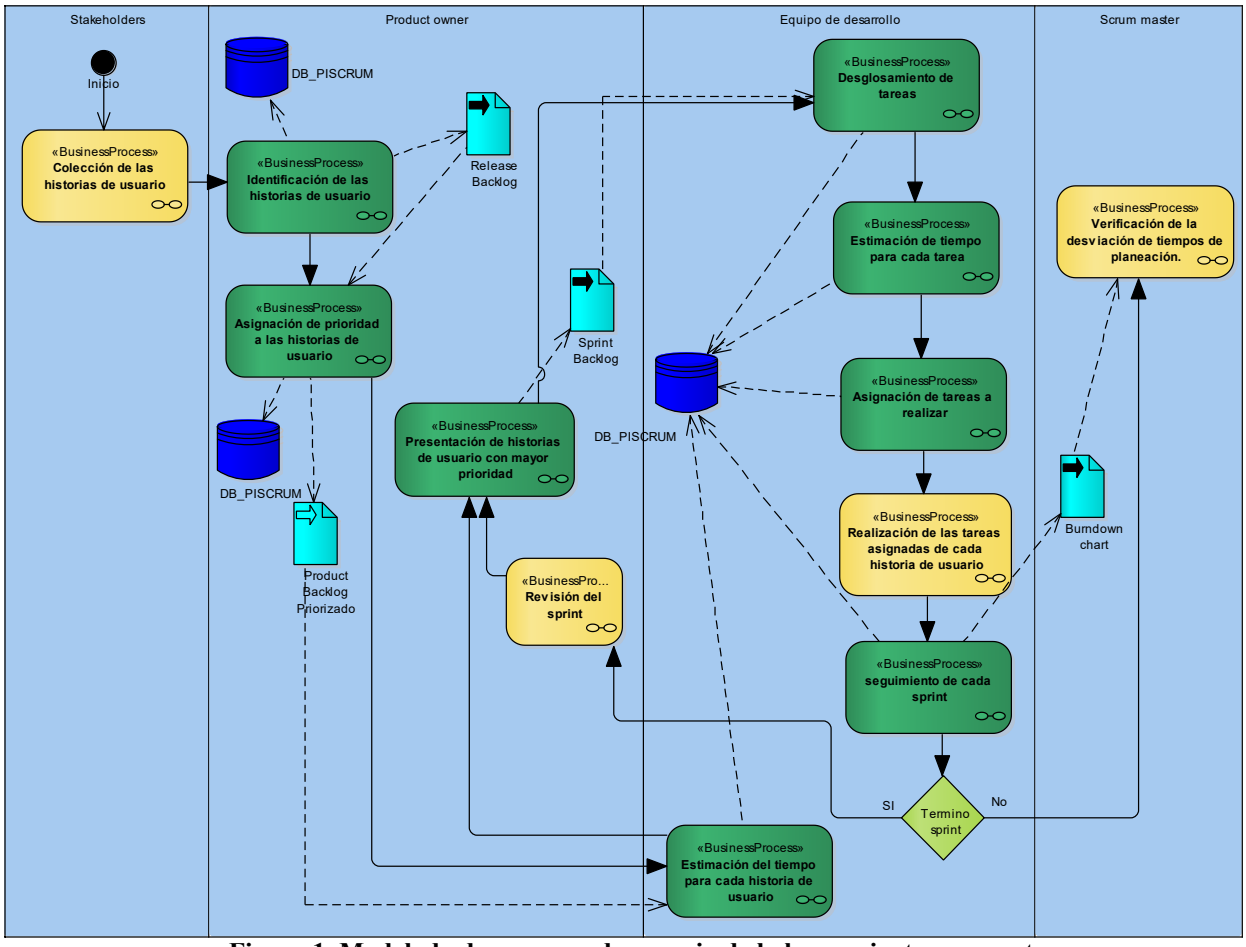

Figura 1. Modelado de procesos de negocio de la herramienta propuesta.

En el diagrama ilustrado en la figura 1, se observa cuatro carriles (divisiones verticales) que representan los roles de la metodología Scrum donde cada uno realiza determinados procesos en la metodología, así como los artefactos generados y cambios en la base de datos que se generan en procesos específicos. A continuación, se describe con un nivel de abstracción alto, cada uno de los procesos involucrados del modelado de negocio:

- *Proceso: Colección de las historias de usuario*. El **cliente** (**StakeHolder**) menciona las **necesidades** o **requisitos (historias de usuario)** al **product owner** y éste realiza la anotación de las mismas.
- *Proceso: Identificación de las historias de usuario.* El **product owner** identifica e interpreta las **historias de usuario** a través de un lenguaje de fácil entendimiento para el **equipo de desarrollo**, además éstas definirán la liberación del producto final.
- *Proceso: Asignación de prioridad a las historias de usuario.* El **product owner** prioriza las **historias de usuario** según al criterio del cliente y se obtiene un **product backlog** priorizado.
- *Proceso: Estimación de tiempo para cada historia de usuario.* El **equipo de desarrollo** junto con el **product owner,** estiman los tiempos para el desarrollo de cada **historia de usuario**.
- *Proceso: Presentación de historias de usuarios con mayor prioridad.* Se realiza una reunión antes de iniciar el **sprint**, en la cual, el **product owner** presenta las **historias de usuarios** con mayor prioridad hasta el momento al **equipo de desarrollo,** dando origen a un **sprint backlog**.
- *Proceso: Desglosamiento de tareas.* El **equipo de desarrollo** divide en pequeñas **tareas** las **historias de usuarios**.
- *Proceso: Estimación de tiempo para cada tarea.* El **equipo de desarrollo** estima un tiempo para cada **tarea** que surgió del **desglosamiento de tareas**.
- *Proceso: Asignación de tareas a realizar.* Entre los integrantes del **equipo de desarrollo** se lleva a cabo la asignación de las **tareas** a realizar para completar cada una de las **historias de usuarios**.
- *Proceso: Realización de las tareas asignadas de cada historia de usuario.* El **equipo de desarrollo** dedica tiempo diariamente a cada **tarea** para completar las **historias de usuario** según lo estimado.
- *Proceso: Seguimiento de cada sprint.* Cada integrante del **equipo de desarrollo** registra las **horas invertidas** en cada **tarea** asignada, esto sirve para saber el avance real de un proyecto.
- *Proceso: Revisión del sprint.* El **product owner** comprueba el progreso del proyecto, donde identifica las funcionalidades que se pueden considerar hechas y las que no.
- *Proceso: Verificación de la desviación de tiempos de planeación.* El **scrum master** verifica la desviación de tiempos de planeación en cada **sprint** apoyándose con el **burndown chart** y poder mitigar riesgos en caso de existir anomalías en los tiempos estimados, permitiendo que no existan atrasos en las entregas de los **sprints**.

Caso de uso del seguimiento del sprint

A continuación, se ilustra en la figura 2, el caso de uso del proceso de **hacer el seguimiento del sprint,** debido a que este proceso es uno de los más importantes de la metodología Scrum, generando el artefacto del **burndown chart,** el cual es de gran valor dado que permite mostrar el tiempo estimado requerido para alcanzar el objetivo del sprint.

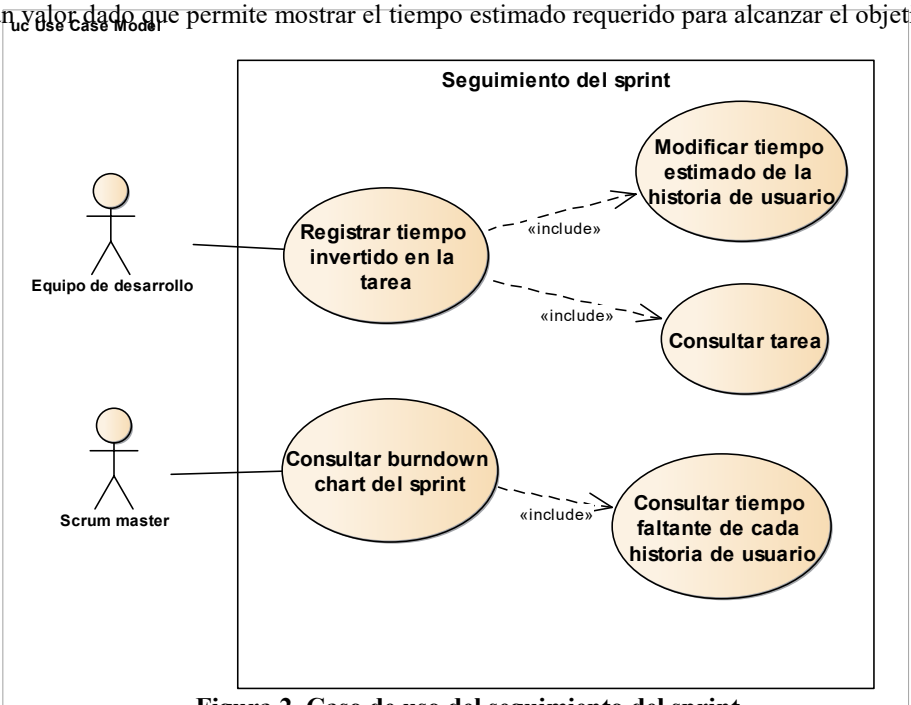

Figura 2. Caso de uso del seguimiento del sprint.

En la figura 2, se observa que cada integrante del **equipo de desarrollo** registra el tiempo invertido en cada **tarea** asignada, la cual a su vez, consulta la tarea para saber a qué **tarea** se registrará el tiempo invertido y posteriormente se actualiza el tiempo estimado de la **historia de usuario** en base a sus tareas involucradas. El **scrum master** consulta el **burndown chart** para verificar la desviación de tiempos de planeación a través de una gráfica, la cual es generada a partir de la consulta del tiempo faltante de cada **historia de usuario**.

Diagrama de clases del seguimiento del sprint

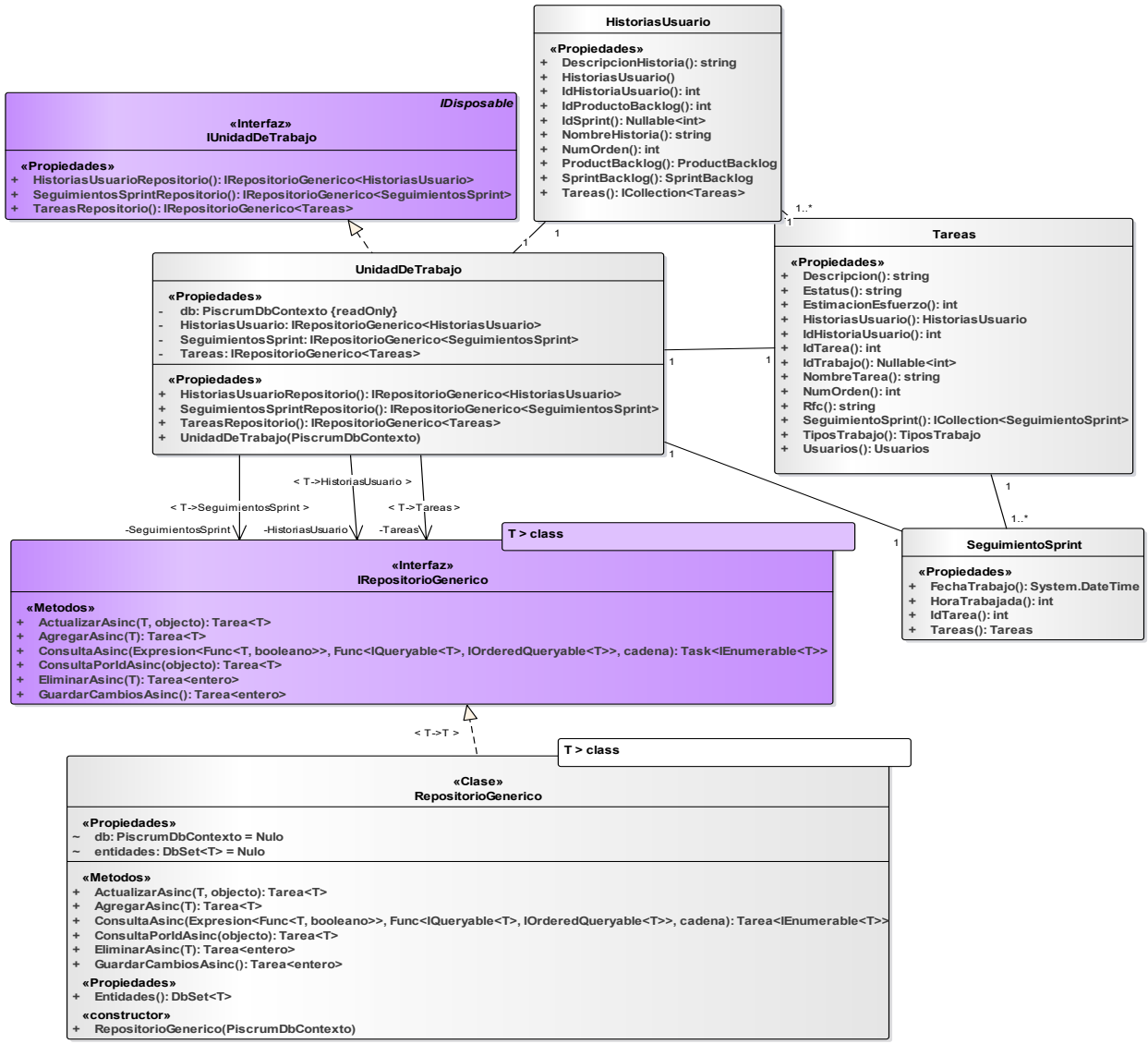

Figura 3. Diagrama de clases del seguimiento del sprint

En la figura 3 se ilustra las clases y atributos que representan el seguimiento del sprint. Como se observa en la figura 3, se modelan dos clases y dos interfaces que representan los patrones de diseño **RepositorioGenerico** y **UnidadDeTrabajo,** donde la primera tiene la función de crear una capa de abstracción entre la capa de acceso a datos y la lógica de negocio. Esta capa de abstracción implementa las operaciones CRUD (Create, Read, Update, Delete) de manera asíncrona, permitiendo hacer peticiones a la base de datos disminuyendo el tiempo de respuesta, esta capa hereda de la interfaz **IRepositorioGenerico,** la cual implementa los mismos métodos de **RepositorioGenerico,** pero esto permite la abstracción del mecanismo de los métodos que implementa **RepositorioGenerico**. **UnidadDeTrabajo** permitirá manejar transacciones durante la manipulación de datos utilizando los métodos implementados en el patrón **RepositorioGenerico**. Al realizar operaciones en la base de datos **UnidadDeTrabajo,** se mantiene una lista de los

objetos afectados (clases) por una transacción de negocio, coordinando la escritura de los cambios y la resolución de problemas de concurrencia.

Diagrama de despliegue de la aplicación

En el diagrama de despliegue se muestra las partes de la herramienta de lo que se tiene que instalar en cada nodo de la red (ver figura 4).

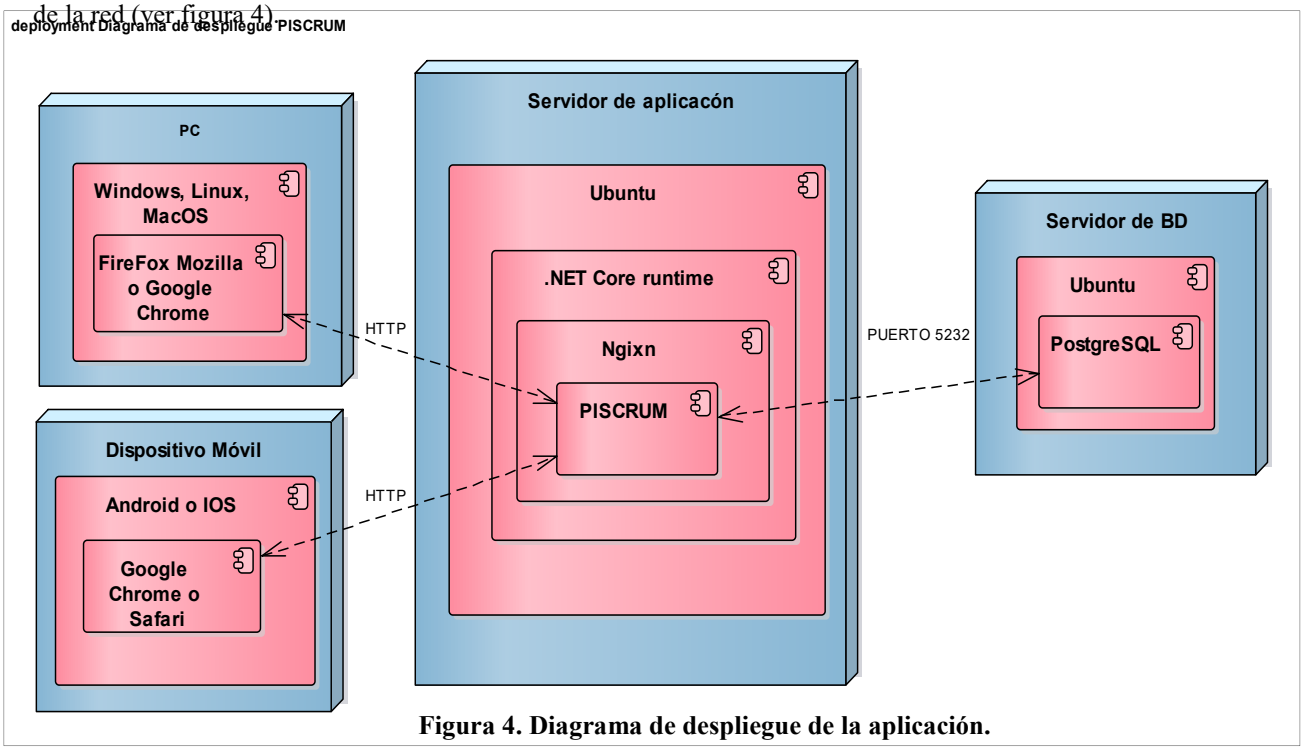

Figura 4. Diagrama de despliegue de la aplicación.

- **PC:** Es el dispositivo en el cual se manipulará la herramienta con nombre **PISCRUM** por medio del **navegador Web** (FireFox Mozilla o Google Chrome), el cual se tiene que instalar en ésta bajo cualquiera de las siguientes plataformas operativas: Windows 8.1, 10, Linux o MacOS.
- **Dispositivo Móvil:** Es el dispositivo en el cual se manipulará la herramienta **PISCRUM** por medio del **navegador Web** (FireFox Mozilla o Google Chrome), el cual se tiene que instalar en ésta y que puede contener los siguientes sistemas operativos: Android o IOS.
- **Servidor de aplicación:** El servidor de aplicación contiene el sistema operativo Linux en su distribución **Ubuntu 18.04**, en el cual se tiene instalado el **.Net Core runtime 2.1** para ejecutar aplicaciones **.Net**. En el servidor de aplicación estará desplegada la herramienta **PISCRUM,** la cual necesita el servidor Web **Ngixn**. **PISCRUM** podrá ser consultada por los dispositivos mencionados anteriormente a través del **navegador Web**.
- **Servidor de base de datos:** En este servidor se tendrá instalado el manejador de base de datos: **postgreSQL 9.5**, el cual es el encargado de administrar y manipular los datos. En dicho servidor se tiene instalado el sistema operativo Linux en su distribución **Ubuntu**.

Comentarios Finales

Conclusiones

En este artículo, se presentó el diseño del seguimiento del sprint, dicha etapa define las actividades más esenciales que contendrá la herramienta PISCRUM, donde se presentó el modelado de procesos de negocio el cual ayuda a generar un mejor panorama sobre en qué procesos de la metodología Scrum, la herramienta propuesta estará involucrada. Además, se presentó un caso de uso de la interacción entre el equipo de desarrollo con la herramienta en el proceso del seguimiento del sprint, así como las clases que estarán involucradas en este proceso y los patrones RepositorioGenerico y UnidadDeTrabajo que estarán involucrados en los demás módulos que contendrá la herramienta. Por último, se mostró el diagrama de despliegue ilustrando la estructura física donde estará desplegada

la herramienta, así como los componentes que se necesitan para que funcione correctamente. Los diagramas presentados anteriormente darán soporte al desarrollo de la herramienta a pesar que la metodología de desarrollo a implementar no exija este diseño.

Trabajos a futuro

Este artículo forma parte de una serie de trabajos que reflejan el progreso del proyecto de tesis de maestría, donde los trabajos a futuro claramente deben abordar la conclusión de la herramienta, en donde se expondrá el desarrollo de la aplicación propuesta, apoyándose en los diagramas presentados en este artículo y haciendo uso de herramientas de desarrollo de software, implementado la arquitectura propuesta en el primer artículo (López Morales, Gazga Portillo, Hernández Bravo, & de Jesús Islao, 2019), así como la metodología de desarrollo de software.

Referencias

Braun, R., & Schlieter, H. (2014). Requirements-based development of BPMN extensions: The case of clinical pathways. IEEE Xplore, 39-44.

Erickson, J., & Siau, K. (2007). Theoretical and practical complexity of modeling methods. Communication of the acm, 46-51.

López Morales, J. R., Gazga Portillo, J. F., Hernández Bravo, J. M., & de Jesús Islao, A. D. (17 de Mayo de 2019). Propuesta de una herramienta basada en la metodología Scrum para la gestión del desarrollo de software. Academia Journals, 11(2), 1597-1602.

Rumbaugh, J., Jacobson, I., & Booch, G. (2000). El lenguaje unificado de modelado. Manual de referencia. Madrid: addison wesley.

Simanjuntak, H. (2015). Proposed framework for automatic grading system of ER diagram. IEEE Xplore, 141-146.

Sommerville, I. (2016). Ingenieria de software. México: Pearson.