UNIVERSIDAD AGRARIA DEL ECUADOR

FACULTAD DE CIENCIAS AGRARIAS

CARRERA DE COMPUTACIÓN E INFORMÁTICA

ANÁLISIS COMPARATIVO DE LOS SISTEMAS DE GESTIÓN DE FLUJO DE TRABAJO TRIANA, KEPLER Y TAVERNA UTILIZADOS EN LA MODELIZACIÓN DE TRABAJOS CIENTÍFICOS

MONOGRAFÍA

Trabajo de titulación presentado como requisito para la
obtención del título de
TECNÓLOGA EN COMPUTACIÓN E INFORMÁTICA

AUTORA

CUENCA RIVERA ESTHER MARGARITA

TUTORA

LOOR CAICEDO GINA MSc.

BALZAR – ECUADOR

2020

UNIVERSIDAD AGRARIA DEL ECUADOR

FACULTAD DE CIENCIAS AGRARIAS

CARRERA TECNOLOGÍA EN COMPUTACIÓN E INFORMÁTICA

ANÁLISIS COMPARATIVO DE LOS SISTEMAS DE GESTIÓN DE FLUJO DE TRABAJO TRIANA, KEPLER Y TAVERNA UTILIZADOS EN LA MODELIZACIÓN DE TRABAJOS CIENTÍFICOS

MONOGRAFÍA

INGENIERÍA DE SOFTWARE

AUTORA

CUENCA RIVERA ESTHER MARGARITA

BALZAR – ECUADOR

2020

UNIVERSIDAD AGRARIA DEL ECUADOR

FACULTAD DE CIENCIAS AGRARIAS

CARRERA DE TECNOLOGÍA EN COMPUTACIÓN E INFORMÁTICA

CERTIFICACIÓN DE ACEPTACIÓN DEL TUTOR

Yo, LOOR CAICEDO GINA JAZMIN MGE., docente de la Universidad Agraria del Ecuador, en mi calidad de Tutora, certifico que el presente trabajo de titulación: ANÁLISIS COMPARATIVO DE LOS SISTEMAS DE GESTIÓN DE FLUJO DE TRABAJO TRIANA, KEPLER Y TAVERNA UTILIZADOS EN LA MODELIZACIÓN DE TRABAJOS CIENTÍFICOS, realizado por la estudiante CUENCA RIVERA ESTHER MARGARITA; ha sido orientado y revisado durante su ejecución; y cumple con los requisitos técnicos exigidos por la Universidad Agraria del Ecuador; por lo tanto se aprueba la presentación del mismo.

Atentamente,

Lcda. Gina Loor Caicedo MGE.
TUTORA

Guayaquil, 10 de abril del 2019

UNIVERSIDAD AGRARIA DEL ECUADOR

FACULTAD DE CIENCIAS AGRARIAS

CARRERA DE TECNOLOGÍA EN COMPUTACIÓN E INFORMÁTICA

APROBACIÓN DEL TRIBUNAL DE SUSTENTACIÒN

Los abajo firmantes, docentes miembros del Tribunal de Sustentación, aprobamos la sustentación del trabajo de titulación: ANÁLISIS COMPARATIVO DE LOS SISTEMAS DE GESTIÓN DE FLUJO DE TRABAJO TRIANA, KEPLER Y TAVERNA UTILIZADOS EN LA MODELIZACIÓN DE TRABAJOS CIENTÍFICOS, realizado por la estudiante CUENCA RIVERA ESTHER MARGARITA, el mismo que cumple con los requisito exigidos por la Universidad Agraria del Ecuador.

Atentamente,

_____

ING. REAL AVILES KARINA MSc.

PRESIDENTE

_____ _____

ING. LAGOS ORTIZ KATTY MSc. LCDA. LOOR CAICEDO GINA MGE.

EXAMINADOR PRINCIPAL **EXAMINADOR PRINCIPAL**

Guayaquil, 10 de abril del 2019

Dedicatoria

Dedico este trabajo a Dios, a mis padres, a mi familia que siempre me han dado las fuerzas para salir adelante y he podido conseguir.

Agradecimiento

A Dios por estar siempre conmigo en los buenos y malos momentos de mi vida por dame inteligencia y sabiduría para resolver los problemas que se me han presentado en mi camino.

Además agradezco de la manera más sincera:

A la Universidad Agraria del Ecuador.

Al Sr. Ing. Jacobo Bucaram Ortiz. Rector Fundador de la Universidad.

A la PhD Martha Bucaram de Jorgge, Rectora de la Universidad.

Al PhD Javier Del Cioppo Morstadt, Vice-Rector de la Universidad

A los maestros del Programa Regional de Enseñanza Balzar de la Universidad Agraria del Ecuador.

A mi tutor quien me ha guiado en mi trabajo monográfico con profesionalismo y dedicación

A todos mis compañeros y amigos.

Autorización de Autoría Intelectual

Yo CUENCA RIVERA ESTHER MARGARITA, en calidad de autora del proyecto realizado, sobre “ANÁLISIS COMPARATIVO DE LOS SISTEMAS DE GESTIÓN DE FLUJO DE TRABAJO TRIANA, KEPLER Y TAVERNA UTILIZADOS EN LA MODELIZACIÓN DE TRABAJOS CIENTÍFICOS” para optar el título de TECNÓLOGA EN COMPUTACIÓN E INFORMÁTICA, por la presente autorizo a la UNIVERSIDAD AGRARIA DEL ECUADOR, hacer uso de todos los contenidos que me pertenecen o parte de los que contienen esta obra, con fines estrictamente académicos o de investigación.

Los derechos que como autor me correspondan, con excepción de la presente autorización, seguirán vigentes a mi favor, de conformidad con lo establecido en los artículos 5, 6, 8; 19 y demás pertinentes de la Ley de Propiedad Intelectual y su Reglamento.

Guayaquil, 30 de enero del 2019

_____

CUENCA RIVERA ESTHER MARGARITA

C.I. 1310423320

Índice general

Índice de tablas

Índice de figuras

Resumen

Este trabajo monográfico está orientado al análisis comparativo de los sistemas de gestión de flujo de trabajo Triana, Kepler y Taverna utilizados en la modelización de trabajos científicos, en base a las características sintácticas, comportamiento de control y datos, ductos computacionales y de análisis de datos, integración de servicio y barridos de parámetros durante el diseño de los trabajos científicos. Se ha dividido la investigación en tres secciones la primera detalla la funcionalidad del sistema de gestión de flujo de trabajo Triana; la segunda sección se enfoca en la funcionalidad del WFMS Kepler; mientras que la tercera sección se explica la funcionalidad de Taverna; en cada una de las secciones de puntualiza la arquitectura de enrutamiento para el diseño de procesos científicos, su ejecución y prototipo de modelado.

Palabras claves: Arquitectura, ejecución, flujo, funcionalidad, patrón.

Abstract

This monographic study is oriented to the comparative analysis of the workflow management systems Triana, Kepler and Taverna that is used in the modeling of scientific works, based on the syntactic characteristics, control behavior and data, computational ducts and data analysis, service integration and parameter sweeps during the design scientific work. The research has been split divided into three sections. The first section details the functionality of the Triana workflow management system; the second section is focused on the functionality of the Kepler WFMS; while the third section explains the functionality of Taverna; in each of the sections, the routing architecture is specified for the design of scientific processes, their execution and modeling prototype.

Keywords: Architecture, execution, flow, functionality, pattern.

1. Introducción

1.1 Importancia o caracterización del tema

Las aplicaciones basadas en datos están cada vez más desarrolladas en la ciencia para explotar la gran cantidad de datos digitales disponibles en la actualidad, necesitándose mecanismos adecuados de composición del flujo de trabajo científicos para respaldar el complejo proceso de gestión, que incluye la creación, reutilización y modificaciones realizadas en el flujo de trabajo a lo largo del tiempo. Esto se refiere a redes de procesos que normalmente se utilizan como canalizaciones de análisis de datos o para comparar datos observados y pronosticados, y que pueden incluir una amplia gama de componentes, por ejemplo, para consultar bases de datos, para transformación de datos y pasos de minería de datos para su ejecución de códigos de simulación en computadoras de alto rendimiento, entre otros.

El valor de un Workflow Management Systems - WFMS científico obedece primordialmente a las funciones particulares de un dominio científico, así como de la unificación de instrumentos específicos de dominio. Además de optimizar la actuación de flujo de trabajo y el empleo de recursos de una manera diáfana, en trabajos de programación con arquitecturas distribuidas en un entorno de red.

La multiplicidad de tareas que desarrolla la comunidad científica dentro de los diversos campos como biología, física, entre otros, los ha conllevado a crear nuevas soluciones de flujo de trabajo que se acoplen a sus requerimientos, en vez de ampliar los sistemas de flujo ya existentes (Curcin y Ghanem, 2013).

1.2 Actualidad del tema

Los sistemas de gestión de flujo de trabajo se han venido empleando en la automatización de los diferentes procesos en negocios y actualmente se utilizan en la gestión de trabajos científicos, integrando diversos recursos como bases de datos, servidores, entre otros que facilitan la comprensión de labores de los científicos en temas de biología molecular, investigación clínica, ciencia computacional, física, química o estadística. Esto les permite a los investigadores el acceso a herramientas que les facilita la reunión de diversos datos, a través de la implementación de procedimientos para el análisis científico.

1.3 Novedad científica del tema

CloudWF puede ejecutar flujos de trabajo compuestos por MapReduce y programas heredados (los programas existentes no se crean utilizando la API de MapReduce). Cada conector de flujo de trabajo contiene una dependencia de bloque a bloque que puede implicar copias de archivos entre bloques conectados. Los sistemas de archivo distribuido - DFS siglas del inglés Distributed file sistems, se usan como intermediario para organizar archivos entre bloques que se pueden ejecutar en diferentes nodos de la nube. Tanto los bloques como los conectores se pueden ejecutar independientemente sin importar a qué flujo de trabajo pertenecen, mientras que cada uno de los árboles de dependencia de bloques de flujo de trabajo se mantiene y reconstruye implícitamente en base a los registros HBase de los componentes de flujo de trabajo. Como resultado, las ejecuciones de flujo de trabajo están descentralizadas, en el sentido de que no hay un control de ejecución separado para cada instancia de flujo de trabajo para realizar un seguimiento de las dependencias: el sistema CloudWF programa bloques y conectores de

todos los flujos de trabajo que se ejecutan en un momento dado de manera uniforme. Esto permite una ejecución altamente paralela y escalable de múltiples flujos de trabajo al mismo tiempo (Zhang y De Sterck, 2017).

1.4 Justificación del tema

Existen muchas implementaciones comerciales y de código abierto para la utilización de flujos de trabajo científicos que permiten la automatización de procesos, siendo los más utilizados por la comunidad de código abierto Kepler, Triana, Taverna, Pegasus, Weka4WS, Gwes, DVega, entre otros, con nuevos entornos de trabajo que aparecen continuamente (Salado-Cid, Luque, y Romero, 2015). En esta investigación monográfica se pretende analizar comparativamente los sistemas de gestión de flujo de trabajo Kepler, Triana y Taverna, dentro de la modelización de trabajos científicos, a través de los siguientes objetivos:

1.5 Objetivos

1.5.1 Objetivo general

Analizar sistemas de gestión de flujo de trabajo Triana, Kepler y Taverna describiendo su funcionalidad dentro de la modelización de trabajos científicos.

1.5.2 Objetivos específicos

- Detallar la funcionalidad del sistema de gestión de flujo de trabajo Triana describiendo su arquitectura de enrutamiento para el diseño de procesos científicos.
- Examinar la funcionalidad del sistema de gestión de flujo de trabajo Kepler a través de la descripción de su arquitectura de enrutamiento para el diseño de procesos científicos.

- Explicar la funcionalidad del sistema de gestión de flujo de trabajo Taverna puntualizando su arquitectura de enrutamiento para el diseño de procesos científicos.
- Establecer las diferencias entre los sistemas de gestión de flujo de trabajo Triana, Kepler y Taverna realizando cuadros comparativos basados en los factores de interoperabilidad, interacción, esquema, procesamiento y reproducción durante el diseño de procesos científicos.

2 Aspectos metodológicos

2.1 Materiales

2.1.1 Recursos bibliográficos

- Artículos Web
- Páginas Web
- Periódicos Web
- Consultas en buscador de internet
- Libros Web
- Libros

2.1.2 Materiales y equipos

- Computadora
- Impresora
- Unidad USB Flash Drive
- Lápiz
- Cuaderno de apuntes
- Hojas A4
- CD

2.1.3 Recursos humanos

Para llevar a cabo la elaboración de esta investigación estuvieron involucrados:

El alumno como proponente de la investigación.

El docente encargado para realizar las debidas sugerencias y ser una guía en la etapa de elaboración del trabajo.

2.2 Métodos

2.2.1 Modalidad y tipo de investigación

- Bibliográfica
- Explicativa
- Descriptiva

2.2.2 Tipos de métodos

2.2.2.1 Método inductivo

Esta investigación monográfica se enfocó en analizar los WFMS Taverna, Triana y Kepler, para luego debatir criterios que permitieron la comprensión con referente a este tema de actualidad.

2.2.2.2 Método deductivo

Este método nos permitió extraer conclusiones de las funciones específicas de los patrones de código abierto Triana, Kepler y Taverna para el diseño de procesos científicos.

2.2.2.3 Método analítico

La información recopilada tuvo su respectivo análisis con el propósito de comprender los patrones del sistema de gestión de flujo de trabajo científico.

2.2.2.4 Método síntesis

Este método consistió en discernir cómo se puede utilizar los sistemas de gestión de flujo de trabajo científico para optimizar y automatizar los procesos.

2.2.3 Técnicas

Con la finalidad de contar con procedimientos e instrumentos que apoyen a los métodos para que de forma sistemática, racional y reflexiva poder acceder al conocimiento, el presente trabajo monográfico utilizó la técnica bibliográfica, pues

esta técnica permitió tomar información de investigaciones realizadas anteriormente por otras personas.

2.3 Marco legal

Esta investigación monográfica se basó en los siguientes artículos:

Constitución de la República del Ecuador:

Art. 385, establece que: "…el sistema nacional de ciencia, tecnología y saberes ancestrales, en el marco del respeto al ambiente, la naturaleza, la vida, las culturas y la soberanía, tendrá como finalidad: generar, adaptar, y difundir conocimientos científicos y tecnológicos; recuperar, fortalecer, y potenciar los saberes ancestrales; desarrollar tecnologías e innovaciones que impulsen la producción nacional, eleven la eficiencia y productividad, mejorar la calidad de vida y contribuyan a la realización del buen vivir...";

Art. 386, determina que: "…el sistema nacional de ciencia, tecnología y saberes ancestrales, comprenderá programas y políticas, recursos, acciones, e incorporará a instituciones del Estado, universidades y escuelas politécnicas, institutos de investigación públicos y particulares, empresas públicas y privadas, organismos no gubernamentales y personas naturales o jurídicas, en tanto realizar, actividades de investigación, desarrollo tecnológico, innovación y aquellas ligadas a los saberes ancestrales…" (Presidencia de la república, 2010).

Ley Orgánica de Educación Superior:

Art. 13.- Funciones del Sistema de Educación Superior.- Son funciones del Sistema de Educación Superior: a) Garantizar el derecho a la educación superior mediante la docencia, la investigación y su vinculación con la sociedad, y asegurar crecientes niveles de calidad, excelencia académica y pertinencia; b) Promover la creación, desarrollo, transmisión y difusión de la ciencia, la técnica, la tecnología y la cultura; c) Formar académicos, científicos y profesionales responsables, éticos y solidarios, comprometidos con la sociedad, debidamente preparados para que sean capaces de generar y aplicar sus conocimientos y métodos científicos, así como la creación y promoción cultural y artística; d) Fortalecer el ejercicio y desarrollo de la docencia y la investigación científica en todos los niveles y modalidades del sistema; e) Evaluar, acreditar y categorizar a las instituciones del Sistema de Educación Superior, sus programas y carreras, y garantizar independencia y ética en el proceso; f) Garantizar el respeto a la autonomía universitaria responsable; g) Garantizar el cogobierno en las instituciones universitarias y politécnicas; h) Promover el ingreso del personal docente y administrativo, en base a concursos públicos previstos en la Constitución; i) Incrementar y diversificar las oportunidades de actualización y perfeccionamiento profesional para los actores del sistema; j) Garantizar las facilidades y condiciones necesarias para que las personas con discapacidad puedan ejercer el derecho a desarrollar actividad, potencialidades y habilidades; k) Promover mecanismos asociativos con otras instituciones de educación superior, así como con

unidades académicas de otros países, para el estudio, análisis, investigación y planteamiento de soluciones de problemas nacionales, regionales, continentales y mundiales (Presidencia de la República del Ecuador, 2010).

Normativa de ética para los profesos de investigación y de enseñanza – aprendizaje – prácticas – comprensión de la Universidad Agraria del Ecuador:

Artículo 7. Investigación para el aprendizaje: La organización de los aprendizajes en cada nivel de formación de la Universidad Agraria del Ecuador se sustentará en el proceso de investigación correspondiente y propenderá al desarrollo de conocimientos y actitudes para la innovación científica, tecnológica, humanística y artística, conforme a lo siguiente:
Investigación en nivel técnico superior y tecnológico, o sus equivalentes en la UAE.- Se desarrollará en el campo formativo de creación, adaptación e innovación tecnológica, mediante el dominio de técnicas investigativas de carácter exploratorio.
Investigación en nivel de grado de la UAE.- Se desarrollará en el marco del campo formativo de la epistemología y la metodología de investigación de una profesión, mediante el desarrollo de proyectos de investigación de carácter exploratorio y descriptivo. Estas investigaciones se realizarán en los contextos de prácticas pre profesionales.
Investigación en nivel de posgrados de la UAE.- Se desarrollará en el marco del campo formativo de investigación avanzada y tendrá carácter analítico, explicativo y correlacional, de conformidad a los siguientes parámetros:
Investigación en especializaciones de posgrados de la UAE.- Este tipo de programas deberán incorporar el manejo de los métodos y técnicas de investigación para el desarrollo de proyectos de investigación de nivel analítico.
Investigación en maestrías profesionales de la UAE.- Este tipo de programas deberán profundizar el conocimiento de la epistemología del campo profesional y desarrollar proyectos de investigación e innovación de carácter analítico, que pueden utilizar métodos multi e inter disciplinar.
Maestrías de investigación de la UAE.- Este tipo de programas deberán profundizar en la epistemología de la ciencia y desarrollar proyectos de investigación de carácter explicativo o comprensivo con un claro aporte al área del conocimiento; podrán ser abordados desde métodos inter disciplinarios y trans disciplinarios (Universidad Agraria del Ecuador, 2014).

3. Análisis y revisión de la literatura

3.1 Funcionalidad del sistema de gestión de flujo de trabajo Triana

3.1.1 Sistema de flujo de trabajo científico

Los sistemas de gestión de flujos de trabajo - workflow management systems (WfMS) son aplicaciones software que permiten la definición, creación y gestión de la ejecución de un conjunto de tareas que trabajan coordinadamente para alcanzar un objetivo común, denominado como flujo de trabajo. Para ello, utilizan uno o más motores de ejecución que permiten gestionar los recursos disponibles e invocar las tareas especificadas (Salado-Cid, Luque, y Romero, 2015).

Un sistema de flujo de trabajo científico es una forma especializada de un sistema de administración de flujo de trabajo diseñado específicamente para componer y ejecutar una serie de pasos computacionales o de manipulación de datos, o flujo de trabajo, en una aplicación científica, por ejemplo, recuperar datos de un instrumento o una base de datos, reformatear los datos y ejecutar un análisis.

Estos sistemas hacen posible integrar diferentes tipos de recursos disponibles, como bases de datos, servidores o servicios, facilitando el intercambio de conocimiento entre áreas tan variadas como la biología molecular, investigación clínica, ciencia computacional, física, química o estadística. Los investigadores tienen acceso a distintas herramientas que permiten incorporar todo tipo de datos que se encuentran distribuidos en diversos entornos, para implementar sus propios procedimientos para el análisis científico. También existen otras herramientas independientes del dominio, que accede a la definición y configuración de los elementos funcionales propios de cada área,

independientemente del WfMS donde se van a emplear, y de esta manera conseguir herramientas ya adaptadas a uno o varios dominios (Salado-Cid, Romero, y Ventura, 2016).

Mediante estos sistemas los investigadores pueden integrar todos los datos para el procedimiento de ejecución de los análisis científicos, utilizando bases de datos, servidores entre otros recursos que permiten el acceso a elementos funcionales para las diferentes áreas de investigación, por ejemplo biología molecular, clínica, bioestadística.

3.1.2 Sistema de flujo de trabajo Triana

Triana es un entorno para la resolución de problemas, que proporciona herramientas para el análisis de datos y cuenta con un gran número de componentes para trabajar en dominios como el procesamiento de audio, texto, entre otros. La principal limitación de estas herramientas radica en la complejidad para ser adaptadas a un dominio específico, y en la imposibilidad de generar aplicaciones particularizadas con elementos de trabajo ya configurados. Por ello, resultaría de interés contar con una metaherramienta en el campo de los workflows científicos que facilitará esta adaptación (Salado-Cid, Romero, y Ventura, 2016).

El sistema de flujo de trabajo Triana es un entorno de resolución de problemas de código abierto que combina una interfaz visual intuitiva con potentes herramientas de análisis de datos. Se puede utilizar para una variedad de tareas, como el procesamiento de señales, texto e imágenes; siendo su primordial limitación la complejidad de adaptación a un dominio específico, así como la dificultad de componer aplicaciones con elementos configurados (ver figura 1).

Triana proporciona una interfaz gráfica de usuario para componer aplicaciones científicas. Un componente es la unidad de ejecución más pequeña escrita como clase Java. Cada componente tiene una definición codificada en XML. Dicho gráfico de la aplicación creada se puede ejecutar a través de la red Grid utilizando la interfaz GAP (Shields, 2012).

Triana es un entorno de resolución de problemas, completo e integrado para componer, compilar y ejecutar aplicaciones científicas, escrito en el lenguaje de programación Java; permitiendo al usuario ejecutar un servicio compuesto en la red Grid, donde los datos se pueden distribuir y procesar de forma remota, devolviendo un pequeño gráfico simple al proceso de visualización y control de imágenes del cliente.

Triana es un entorno de resolución de problemas basado en el flujo de trabajo visual desarrollado en la Universidad de Cardiff. Un componente de flujo de trabajo en Triana se llama unidad y las unidades se pueden conectar entre sí mediante cables dirigidos; puede proporcionar soporte para el flujo de control mediante el uso de algunos mensajes especiales que pueden activar el control entre unidades. También hay nodos especiales para bifurcar y bucles, y se pueden combinar con otras unidades funcionales para construir formas más sofisticadas de flujo de control (Curcin y Ghanem, 2013).

Este sistema fue desarrollado en la Universidad de Cardiff, con la finalidad de brindar un soporte para el flujo de control entre unidades, mediante nodos que pueden ser combinados para generar formas sofisticadas de flujos de control.

3.1.3 Características del sistema de flujo Triana

Triana puede usarse para proporcionar mejores herramientas de administración y despliegue. Además de eso, no posee ningún tipo de herramientas o

información de registro o gestión; pudiendo extenderse para usar Monitoreo de estampida, al igual que Pegasus. Al integrar Triana al modelo de datos Stampede, cada tarea dentro de un gráfico de tareas se ejecuta localmente. Stampede es una infraestructura que proporciona monitoreo interoperable utilizando un modelo de tres capas:

- Un modelo de datos común para describir el flujo de trabajo y las ejecuciones de trabajo.
- Herramientas de alto rendimiento para cargar registros de flujo de trabajo que se ajusten al modelo de datos en un almacén de datos.
- Una interfaz de consulta común (Vatri et al., 2013).

El sistema de flujo Triana aporta con herramientas de administración y despliegue, careciendo de información de registro o gestión por lo que puede ampliarse utilizando el monitoreo de estampida, ejecutando localmente cada tarea dentro de un gráfico.

“Triana se centra principalmente en la ejecución de flujos de trabajo en la red o como servicios web. Tiene dos políticas de distribución: paralelo y de canalización, así como dos modos de ejecución: dinámico y estático” (Churches et al., 2011).

Este sistema primordialmente se enfoca en la construcción de flujos de trabajo en la red o como servicios de web, empleando políticas de distribución por paralelo o canalización, además su ejecución puede ser dinámica o estática.

Triana proporciona tolerancia a fallas (capacidades de prevención y recuperación) en el nivel de tarea, nivel de flujo de trabajo y nivel de usuario. También puede detectar fácilmente fallas en el nivel de hardware, pero no tanto

en el nivel del sistema operativo (Costan, Stratan, Tirson, Ionut, y Cristea, 2011).

Tiene la capacidad de prevenir y recuperar a nivel de tarea, flujo de trabajo, usuario y hardware, por su tolerancia a fallas; sin embargo no puede realizarse a nivel de sistema operativo. El usuario debe seleccionar los recursos para la ejecución. Un usuario, por ejemplo, seleccionará un grupo de tareas que quiera ejecutar en paralelo. La unidad principal de la ejecución distribuida de Triana son esas tareas grupales; los datos se distribuyen en consecuencia. En el enfoque dinámico, los flujos de trabajo se envían a servicios que pueden ejecutar cualquier subflujo y comunicarse con otros servicios de Triana a los que están conectados. En el enfoque estático, un usuario puede elegir iniciar una unidad de grupo como un servicio remoto específico, por lo que las unidades o grupos de Triana se pueden implementar como servicios web.

3.1.4 Arquitectura de Triana

El componente funcional en Triana se llama unidad. Las unidades se conectan mediante cables dirigidos que entran y salen de sus puertos para formar flujos de trabajo. Existe una unidad de grupo de nivel superior con el propósito de integrar flujos de trabajo entre sí como unidades (Migliorini, Gambini, La Rosa, y Ter-Hofstede, 2014).

Triana tiene un componente utilizable llamado unidad, que se conectan a través de cables para constituir los flujos de trabajo.

Los componentes del flujo de trabajo de Triana se desarrollaron para el procesamiento de señales, imágenes y audio y el análisis estadístico, agrupados en cajas de herramientas de componentes relacionados. Además de estos, se desarrolló un conjunto genérico de componentes para la

integración de código Java, aplicaciones heredadas, WS – RF, P2P o servicios web de descripción de lenguaje WSDL. Las implementaciones de cable se resuelven en tiempo de ejecución según los tipos de unidades que están conectadas. Por ejemplo, un cable entre dos unidades locales hará que un archivo se mueva de una ubicación en el sistema de archivos a otra, mientras que el cable entre dos herramientas remotas iniciará una transferencia Grid FTP (Abdul, 2015).

Una capacidad notable de Triana es modificar y volver a publicar cualquier nodo. El código fuente de cada nodo en la caja de herramientas se puede ver, modificar y recompilar dentro del entorno, lo que permite un rápido desarrollo de nuevos componentes en Java puro.

Un servicio de Triana consta de tres componentes: un cliente, un servidor y un servidor de procesos de comando. En una red típica, solo un servidor de proceso de comando estará activo. El componente cliente del Servicio de Triana en contacto con el controlador de Triana luego canaliza los datos y el programa a los componentes del servidor de otros servicios de Triana. Estos servicios se ejecutan las tareas asignadas y transfieren datos según lo prescrito por el controlador de Triana (Sonntag, Karastoganova, y Leimann, 2010).

Triana está constituida por tres componentes que son cliente, servidor y servidor de procesos de comando, que generalmente se encuentra activo. El componente cliente se pone en contacto con el controlador para canaliza los datos y el programa hacia el servidor, ejecutando las tareas determinadas y transfiriendo los datos de acuerdo a lo establecido por el controlador (ver figura 2).

Triana se enfoca en servicios de soporte en múltiples entornos, como peer-to-peer (P2P) y Grid, al integrarse con varios tipos de kits de herramientas de

middleware. Triana GUI proporciona una interfaz gráfica fácil de usar para construir una red. Alternativamente, una interfaz de línea de comandos también está disponible para este propósito. Una vez que se crea la red, se genera un Task Graph basado en XML. Este controlador de Triana ejecuta este Task Graph a través de una puerta de enlace del Servicio de Triana (Taylor, Shields, Wang, y Harrison, 2011).

El enfoque de Triana se basa en múltiples entornos como peer to peer y grid. Integrándose con diversos kits de middleware. Triana GUI provee de una interfaz gráfica y una interfaz de línea de comandos que facilitan la construcción de una red, posteriormente se genera un task graph asentado en XML, que se ejecuta mediante un enlace.

Sobre la base de esta arquitectura, se pueden identificar tres herramientas principales:

- **Triana GUI:** Triana GUI: una herramienta de modelado de procesos que proporciona al usuario una interfaz gráfica de arrastrar y soltar fácil de usar para construir redes.
- **Representación de Task Graph:** una definición del proceso que comprende varios pasos discretos y se expresa en forma de texto. En el caso de Triana, esto se representa en formato XML WSFL.
- **Ejecución del proceso:** similar a un motor de flujo de trabajo, una puerta de enlace del servicio de Triana ejecuta el Task Graph, realizando llamadas al GAT según sea necesario (Taylor, Majithio, Shields, y Wang, 2013).

Triana tiene tres herramientas importantes que son Triana GUI que permite el modelado de los procesos; el task graph que involucra los diversos pasos del

flujo; y la ejecución del proceso que es la puerta de enlace para la ejecución del task graph.

3.1.4.1 Triana Gui

La GUI de Triana proporciona al usuario un medio para construir una red mediante una simple acción de arrastrar y soltar, conectar y agrupar tareas, establecer y cambiar parámetros, entre otros. Se planea implementar al menos tres tipos de GUI: una versión estándar que se ejecutará en las máquinas, una versión liviana para ejecutar en dispositivos con recursos limitados y una vista del navegador web para informar sobre los parámetros de progreso / cambio. La GUI de Triana también permite al usuario especificar la máquina en la que se ejecutará el Task Graph completo o parcial (Talia, 2013).

Triana Gui facilita que el usuario realice un análisis pudiendo realizar cambios en el flujo de trabajo agregando, eliminando o cambiando la secuencia de ejecución simplemente reorganizando el flujo de trabajo. Existen tres tipos el estándar que se realiza en la máquina, la versión liviana para los dispositivos, y la del navegador web que permite observar los parámetros del progreso (ver figura 3).

3.1.4.2 Representación de Task Graph

En Triana, un flujo de trabajo se representa mediante un formato de representación similar a WSFL basado en XML. El lenguaje de flujo de servicios web (WSFL) es un lenguaje basado en XML para componer y coreografiar el flujo de servicios web. WSFL se utiliza para producir una representación basada en XML de un proceso. Esta representación se alimenta luego en una aplicación de middleware diseñada para invocar y administrar el proceso. WSFL usa WSDL para la descripción de las interfaces de servicio y

sus enlaces de protocolo. WSFL también se basa en un "lenguaje de descripción de punto final" previsto para describir las características no operativas de los puntos finales de servicio, como las propiedades de calidad de servicio - Web Services Endpoint Language (Guan et al., 2010).

Se utiliza una representación XML similar a WSFL para representar el gráfico de tareas. Task Graph tiene tres tipos de elementos: tareas, enlaces de control y enlaces de datos.

- Una tarea representa una operación.
- Los enlaces de control definen la secuencia de tareas en el modelo.
- Los enlaces de datos describen el flujo de datos entre tareas (Liu, Pacititti, Valdariez, y Maltoso, 2015).

Según lo expresado por los autores el task graph genera comandos de acción realizados por el usuario durante una sesión interactiva, como avanzar, detener, entre otros. La combinación de estos permite la simulación de comandos de usuario desde la GUI, lo que permite a Triana ejecutar la aplicación independiente directamente. Al implementar las mismas interfaces que los lectores de gráficos de tareas incluidas utilizan en el servicio del controlador de Triana, otro mecanismo de distribución y ejecución podría insertarse con su propia administración y programación de recursos, pero aprovechando las características de inicio de sesión remoto y la interfaz de usuario de la interfaz gráfica de usuario y el subsistema distribuidos.

3.1.4.3 Triana service

El Servicio de Triana es responsable de validar el Task Graph entrante y ejecutarlo. Además, el Servicio de Triana distribuye tareas de gráficos parciales

a otros servicios de Triana que se ejecutan en servidores remotos a través del GAT. El servicio de Triana es responsable de:

- Interpretación de la definición del proceso;
- Control de instancias de tareas;
- Navegación entre tareas (secuencial, paralela, condicional, entre otros.)
- Mantenimiento de datos de control de flujo de trabajo;
- Mantenimiento de auditoria y detalles históricos (Ferreira et al., 2017).

De acuerdo a lo señalado por los autores el servicio de Triana permite la interpretación del proceso, navegación ya sea secuencial, paralela, condicional, entre otras; así como el mantenimiento de los datos, auditoría y el histórico.

3.1.5 Ejecución de la aplicación

Triana se basa en un entorno de resolución de problemas que permite la ejecución de la aplicación científica de uso intensivo de datos. Para la cuadrícula, tiene una capa de middleware de abstracción independiente denominada prototipo de aplicación de cuadrícula (GAP). Esto permite a los usuarios anunciar, descubrir y comunicarse con la Web y servicios P2P (peer-to-peer). Triana también utiliza el RLS para administrar los datos en tiempo de ejecución (Yuan, Yang, y Chen, 2013).

De acuerdo a lo indicado la ejecución de la aplicación Triana para flujo de trabajos científicos consiste en una colección de interfaces que se unen a diferentes tipos de middleware y servicios, como la interfaz del Prototipo de aplicación de cuadrícula (GAP) y sus enlaces a servicios P2P e integración a servicios web,

- **Control de flujo**

La ejecución en Triana se basa en el empuje, y la salida de cada componente se envía por los cables a los componentes receptores, que luego comienzan a ejecutarse. A pesar de ser un sistema de flujo de datos, Triana brinda soporte para el control de flujo a través de mensajes especiales que activan el control entre unidades. Además de estos, hay nodos dedicados para la bifurcación, que pasan los datos solo a uno de los destinatarios, y en bucle. Estas construcciones se pueden combinar libremente con componentes funcionales. La selección condicional se basa en el uso acoplado de los componentes If y Switch. If pasa los datos a uno de dos nodos en función de alguna condición, mientras que Switch selecciona una de las dos entradas también en función de alguna condición (Triana, 2016).

De acuerdo a lo manifestado además del envío de la información desde los componentes de entrada hacia los receptores, para realizar la ejecución del flujo de datos, Triana también permite el control del flujo mediante mensajes que impulsan el control entre las unidades, existiendo además nodos bifurcan el mensaje llegándole solamente a un destinatario. Estas combinaciones se realizan con componentes funcionales, como son If, que permite el paso de la información al nodo establecido en la función condicionada; y Switch facilita la selección de las entradas según la función condicionada.

- **Flujo de datos**

El flujo de datos predeterminado en Triana es desde la unidad de origen a la unidad de destino, que representa la composición de la función. Sin embargo, la adición de numerosas estructuras de control rompe la metáfora funcional a través del no determinismo. Efectivamente, un historial de componentes ejecutados representa la composición funcional que se realizó (Joyce, 2016).

El flujo de datos en Triana se encuentra predeterminado desde el origen hasta el destino, no obstante puede cambiarse el esquema a través de las estructuras de control.

> El mecanismo de flujo de datos estándar completa la ejecución de un nodo antes de pasar el resultado al siguiente. Este no es el único paradigma compatible, y la transmisión es posible utilizando un conjunto de componentes dedicados, como secuencia, bloque, combinación y otros. A través de estas construcciones, es posible programar la ejecución paralela de varias unidades en el gráfico en diferentes subconjuntos de datos (Atkinson, Gesing, Montagnot, y Taylor, 2017).

Antes de pasar los datos de un nodo a otro tiene que completarse la ejecución; además la transmisión puede realizarse mediante componentes dedicados como secuencia, bloque, combinación, programando su ejecución en diversos subconjunto de datos.

3.1.6 Construcciones de control

Hay tres construcciones de control que deben representarse dentro de Triana:

1. Procesamiento paralelo,
2. Procesamiento condicional y
3. Iteración (Triana, 2016).

Según el autor en Triana existen tres construcciones que son procesamiento paralelo, procesamiento condicional e iteración.

3.1.6.1 Procesamiento paralelo

Por ejemplo, dos tareas B y C deben ejecutarse, pero el orden de ejecución es arbitrario. Para modelar dicho enrutamiento paralelo, se utilizan dos bloques de construcción:

- La división AND y
- El AND-join (Wildeet al., 2016).

Lo señalado por los autores implica que para ejecutarse un procesamiento paralelo se debe utilizar los bloques de construcción and y and-join.

3.1.6.2 Procesamiento condicional

El procesamiento condicional se utiliza para permitir un enrutamiento que puede variar entre los casos. De esta manera, el enrutamiento de un caso puede depender de los atributos del flujo de trabajo. Para modelar una elección entre dos o más alternativas, se utilizan dos bloques de construcción:

- La división IF y
- La unión IF (Deelman et al., 2017).

Una división IF puede ser modelada por una unidad con varios cables salientes, una unión IF es modelada por una unidad con múltiples cables entrantes. La elección entre alternativas a menudo depende de los atributos del flujo de trabajo. Si la elección se basa en atributos de flujo de trabajo, es una elección determinista. El usuario puede especificar el parámetro que se va a probar y, en consecuencia, los datos se envían a la unidad apropiada. Además, la construcción de control SWITCH se usa para especificar cuál de las dos entradas usará una red visual, dependiendo de una expresión condicional. Los datos se obtendrán de la primera entrada si una condición especificada es VERDADERA; se obtendrá de la segunda ruta si la condición especificada es FALSA (Abdul, 2015).

La realización de un enrutamiento bajo un procesamiento condicional dependerá de los atributos del flujo de trabajo científico, utilizando los bloques de construcción división if y unión if, la primera se modela con cables salientes y la

segunda con cables entrantes; es el usuario que deberá especificar los parámetros de la investigación. Switch se utiliza para especificar la entrada a utilizarse para la red visual, realizando la expresión condicional, así por ejemplo si una condición especificada es verdadera los datos se obtienen de la primera entrada; y si es falsa de la segunda.

3.1.6.3 Iteración

Dos posibles construcciones looping son:

- **Count Loop**

Count Loop hace que el flujo de datos se repita en una sección particular de la red un número específico de veces. Cuando los datos están disponibles en las conexiones de entrada de la unidad de bucle de conteo, los eventos ocurrirán de la siguiente manera:

- Primero, se inicializará el contador.
- A continuación, el valor actual del contador se comparará con el valor final especificado; Si el valor actual es menor que el valor final, se ejecutará la red contenida en el bucle de conteo.
- Después de que se ejecuten las unidades dentro del ciclo de conteo, el valor de incremento especificado se agregará al valor actual de la variable de conteo.
- El nuevo valor de la variable de conteo se comparará nuevamente con el valor final especificado; Si el valor actual es aún menor que el valor final, la red se reprogramará. Este proceso se repetirá hasta que el valor actual de la variable de conteo cumpla o exceda el valor final especificado.

- El flujo de datos luego avanzará en sentido descendente desde la conexión de salida de la unidad de bucle de conteo al resto de la red (Taylor, Majithio, Shields, y Wang, 2013).

Las construcciones looping hacen que el flujo de control del programa visual se repita en una sección particular de la red varias veces; éstas dependen del valor de las variables para determinar cuántas veces se repetirá el flujo de control en la sección especificada de la red; es decir se automatiza el proceso de varios pasos organizando secuencias de acciones por lotes y agrupando las partes que deben repetirse. Count loop se ejecuta durante un número determinado de veces, según lo controlado por un contador o un índice, incrementado en cada ciclo de iteración.

- **While Loop**

El While Loop hace que el flujo de datos se repita en una sección particular de la red siempre que se cumpla una condición particular. Cuando los datos están disponibles en la conexión de entrada de la unidad looping, los eventos ocurrirán de la siguiente manera:

- Se evalúa la condición de salida; si se evalúa como FALSO, se ejecuta la red contenida en el While Loop.
- Después de que se ejecuten las unidades en el While Loop, la expresión condicional se evaluará nuevamente; si la expresión condicional sigue siendo FALSA, la red será reprogramada; el proceso se repetirá hasta que la expresión condicional se evalúe como VERDADERA.
- El flujo de datos luego avanzará hacia abajo desde la conexión de salida de la unidad de While Loop al resto de la red (Migliorini, Gambini, La Rosa, y Ter-Hofstede, 2014).

De acuerdo a lo manifestado por los autores dentro de Triana, el usuario crea un grupo seleccionando las unidades a las que se realizará el While Loop. El Grupo pone todos los parámetros a disposición del usuario para que el usuario pueda configurar las condiciones de salida, basándose en el inicio y la verificación de una condición lógica. La condición se prueba al inicio o al final de la iteración.

3.2 Funcionalidad del Sistema de gestión de flujo de trabajo Kepler

> Kepler es un proyecto colaborativo de código abierto, que pretende proporcionar un "entorno de modelado y resolución de problemas". Más concretamente, se trata de un sistema diseñado para crear modelos ejecutables utilizando una representación visual de los procesos que implican. La representación gráfica de estos modelos, también llamados flujos de trabajo muestra el flujo de datos entre los distintos componentes del análisis. Kepler está especialmente diseñado para dar soporte al flujo de datos en distintos dominios técnicos y científicos, como la bioinformática, la ecoinformática y la geomática entre otros, pero sus características pueden ser aplicadas a cualquier campo que requiera flujos de trabajo con datos para resolver problemas (García, Casado, Pérez, y Benito, 2012, pág. 41).

El Proyecto Kepler es de código abierto que provee de un entorno de modelado con la finalidad de establecer modelos ejecutables a través de la representación gráfica o flujos de trabajo, de esta manera facilita a técnicos y científicos la resolución de problemas en áreas como bioinformática, ecoinformática, geomática entre otros.

> Kepler está dedicado a promover y respaldar las capacidades, el uso y el conocimiento de la aplicación de flujo de trabajo científico de código abierto y gratuito, Kepler. Kepler está diseñado para ayudar a los científicos, analistas y

programadores de computadoras a instituir, producir y participar modelos e investigaciones en una gran escala de áreas científicas y de ingeniería. Kepler puede operar con información recopilada en una diversidad de formatos, por Internet o en redes internas, y es un entorno efectivo para integrar componentes de software dispares, como la combinación de scripts "R" con el código "C" agrupado, o ayudar en la realización remota y distribuida de modelos utilizando la interfaz gráfica de Kepler, los usuarios simplemente seleccionan y luego conectan los componentes analíticos pertinentes y los archivos con la información para establecer un flujo de trabajo científico, una forma operable de obtener resultados siguiendo las fases requeridas (Kepler, 2016).

Kepler, como se presenta en su sitio web, está diseñado para ayudar a los científicos, analistas y programadores de computadoras a crear, ejecutar y compartir modelos y análisis en una amplia gama de disciplinas científicas y de ingeniería, permitiendo operar con la información en diversos formatos, ya sea por internet o redes internas; además es un entorno seguro para combinar componentes de software dispares, como scripts "R" con código "C", o realizando modelos mediante la interfaz gráfica de Kepler, seleccionando y conectando los componentes y la información para la ejecución del flujo de trabajo científico y obtención de resultados.

3.2.1 Características de Kepler

Kepler se basa en el sistema Ptolemy II, una plataforma madura que admite múltiples modelos de cómputo adecuados para distintos tipos de análisis.

- Kepler es gratuito bajo la Licencia BSD.
- Kepler provee de una interfaz gráfica.

- Los flujos de trabajo de Kepler permite crear sub-flujos de trabajo modulares y reutilizables.
- Los flujos de trabajo de Kepler pueden aprovechar el poder computacional de las tecnologías de red (por ejemplo, Globus, SRB, Web y Soaplab Services), así como aprovechar el soporte nativo de Kepler para el procesamiento en paralelo.
- Los flujos de trabajo de Kepler se pueden guardar, reutilizar y compartir con colegas utilizando el formato de archivo de Kepler (KAR).
- Kepler incluye una biblioteca con capacidad de exploración con más de trescientos cincuenta componentes de procesamiento llamados actores, listos para usar, los mismos que se pueden personalizar, conectarse y ejecutarse desde una computadora (Nguyen, Crawl, Mosoumi, y Altintas, 2016).

De acuerdo con lo expuesto por los autores Kepler es un sistema basado en la plataforma Ptolemu II con licencia BSD gratuita, provee de una interfaz gráfica, con mecanismos simples que permiten la creación de sub-flujos de trabajos modulares, aprovechando tecnologías de red como Globus, SRS, Web y Soaplab service. Además los trabajos realizados pueden ser guardados reutilizados y compartidos en formato de archivo Kepler (ver figura 4).

3.2.2 Prototipo de modelado

Los sistemas de flujo de trabajo científico pueden beneficiarse enormemente de estas tecnologías web, ya que las visualizaciones de alta calidad pueden dar a los científicos una mejor comprensión de los resultados que conducen a nuevas perspectivas sobre sus datos o modelo. Los flujos de trabajo a menudo administran simulaciones de larga ejecución, y las visualizaciones basadas en

datos pueden mostrar los resultados al finalizar el trabajo o en tiempo real mientras se ejecuta el flujo de trabajo. Además, los componentes interactivos proporcionan un mecanismo conveniente para inspeccionar o consultar información de un flujo de trabajo en ejecución o del sistema de procedencia. La habilitación del acceso web aumenta el número y los tipos de interfaces posibles para el sistema de flujo de trabajo científico; Se pueden ensamblar diferentes interfaces para diferentes dispositivos, tamaños de pantalla y escenarios de uso (Lee et al., 2012).

Los sistemas de flujo de trabajo constituyen un beneficio para los científicos proporcionando una visualización de los resultados que permiten una mejor comprensión; estos flujos pueden realizar simulaciones cuya visualización puede darse en el transcurso de la ejecución en tiempo real; a través de los componentes interactivos se puede realizar inspecciones o consultar información existente. El acceso a la web incrementa el número y tipo de interfaces de acuerdo a los dispositivos, tamaños de pantalla y escenarios de uso.

Los actores Kepler en un flujo de trabajo pueden anotarse con uno o más atributos de WebView que especifican componentes HTML arbitrarios. Un servidor web ligero se ejecuta dentro del proceso Kepler y proporciona comunicación en tiempo real entre los atributos de WebView y los componentes HTML que se ejecutan en los clientes web. El atributo WebView supervisa el comportamiento de su actor, por ejemplo, los datos leídos en puertos, ejecuciones de actores, entre otros. e informa de estos eventos al componente HTML correspondiente que se ejecuta en el cliente web (Migliorini, Gambini, La Rosa, y Ter-Hofstede, 2014).

En un flujo de trabajo Kepler, los actores pueden valerse de los atributos de WebView que detallan componentes HTML, suministrando información en tiempo real que pueden ejecutar los clientes web. Estos atributos inspeccionan el comportamiento de su actor, informando al componente HTML correspondiente.

Un flujo de trabajo tiene una serie de componentes formalmente descritos, algunos están representados visualmente sobre la interfase de Kepler; estos componentes son Director, actor, actor compuesto, receptores, parámetros, relaciones, puertos, canal, paquete de datos, cuyas funciones se especifican en la tabla 1.

3.2.3 Responsabilidades de los directores

Kepler permite ejecutar flujos de trabajo utilizando distintos modelos de computación, representados por los Directores, de acuerdo al que le sea asignado.

El Director de Process Network (PN), a diferencia del Director SDF, no calcula estáticamente los horarios de disparo. En cambio, en un flujo de trabajo de PN, cada actor tiene un hilo de Java independiente y el flujo de trabajo es impulsado por la disponibilidad de datos: los tokens se crean en los puertos de salida siempre que los tokens de entrada estén disponibles y se pueda calcular la salida. Los tokens de salida se pasan a los actores conectados, donde se mantienen en un búfer hasta que el siguiente actor recolecte todas las entradas necesarias y pueda disparar. El PN Director termina de ejecutar un flujo de trabajo solo cuando no hay nuevas fuentes de token de datos en ninguna parte del flujo de trabajo (Atkinson, Gesing, Montagnot, y Taylor, 2017).

En la tabla 2 se indica un resumen de las características de cada uno de los directores como son el director de Process Network (PN), que es una opción

popular para los diseñadores de flujos de trabajo científicos, ofreciendo un diagrama de la semántica. En esta semántica, los actores son procesos independientes que se ejecutan simultáneamente, cada uno con su propio hilo de control, y se comunican mediante el envío de tokens a través de canales unidireccionales. También está, el director SDF (Synchronous Data-Flow), que se puede utilizar para redes de proceso especializadas con producción fija de tokens y tasas de consumo. El director SDF (Synchronous Data Flow) que realiza un análisis estático en un flujo de trabajo garantizando la ausencia de puntos muertos, determina los tamaños de búfer necesarios y optimiza la programación de la ejecución del actor. Se han construido otros directores para modelar sistemas de eventos discretos (DE), modelos de tiempo continuo (CT). La selección de director para un flujo de trabajo se presenta en la figura 5.

3.2.4 Responsabilidades de los actores

Un actor de servicio web para acceder y ejecutar servicios web definidos por WSDL y devolver los resultados de ejecución para su posterior procesamiento dentro de un flujo de trabajo. Un actor de la tabla de lectura para acceder a los datos heredados almacenados en archivos Excel. Un actor de ejecución externa para ejecutar aplicaciones de línea de comandos desde un flujo de trabajo. El Repositorio de componentes de Kepler proporciona un servidor centralizado donde los componentes y flujos de trabajo se pueden cargar, descargar, buscar y compartir con la comunidad o los usuarios designados (Lee et al., 2012).

Un actor de servicio web permite el acceso y ejecución de los servicios web reenviando los resultados para ser procesados en el flujo de trabajo. El actor de tabla de lectura permite ver los resultados en Excel. El actor de ejecución externa

facilita la ejecución de las aplicaciones desde el flujo de trabajo; Kepler permite cargar, descargar, buscar y compartir los flujos de trabajos a los usuarios establecidos.

Mediante el uso de uno de varios actores específicamente diseñados para ingerir y generar datos, los flujos de trabajo pueden acceder y utilizar una amplia variedad de fuentes de datos. Actualmente, Kepler admite datos descritos por Ecological Metadata Language (EML), datos accesibles mediante el protocolo DiGIR, el protocolo OPeNDAP, DataTurbine, GridFTP, JDBC, SRB y otros. Kepler proporciona acceso directo a Earth Grid, una red distribuida que brinda a los científicos acceso a datos analíticos y de datos ecológicos, de biodiversidad y ambientales (Ludascher et al., 2013).

Los actores de Kepler se han planteado para introducir y componer datos, los flujos de trabajo pueden utilizar diferentes fuentes de datos. En la actualidad este sistema admite datos definidos por Ecological Metadata Language (EML), mediante los protocolos DiGIR, OPeNDAP, DataTurbine, GridFTP, JDBC, SRB y otros; además tiene acceso directo a Earth Grid, que es una red distribuida que proporciona a los científicos datos analíticos y ecológicos, de biodiversidad y ambientales.

Los distintos actores pueden ofrecer sus salidas en ventanas diferentes según se trate de texto, gráficos descriptivos o imágenes.

3.2.5 Flujos de trabajo de muestra

Kepler viene con una serie de flujos de trabajo de demostración que se pueden encontrar en su directorio Kepler Data / workflows / module / outreach / demos.

3.2.5.1 Lotka-Volterra Workflow

El flujo de trabajo de Lotka-Volterra resuelve el clásico modelo de dinámica de presas de predadores de Lotka-Volterra, que describe las poblaciones relativas de un depredador y su presa a lo largo del tiempo utilizando dos ecuaciones logísticas que consideran tanto la competencia intraespecífica como interespecífica, donde la tasa de incremento poblacional sobre el número de individuos de población es igual a la tasa intrínseca de crecimiento de población por el número de individuos de la población que multipllica a la capacidad de carga para la población (κ) menos el número de individuos, menos el coeficiente de competencia (α). La ausencia de competencia interespecífica hace que la población crezca de manera logística hasta llegar a un punto de equilibrio. Los resultados se trazan a medida que se calculan, mostrando las dos poblaciones frente al tiempo, utilizando el actor TimedPlotter; y, entre sí utilizando el actor XYPlotter. El flujo de trabajo se muestra arriba en la interfaz de Kepler, y los componentes principales del flujo de trabajo -actores, puertos, parámetros, entre otros, se identifican con llamadas (Williams, 2018).

El flujo de trabajo de Lotka-Volterra soluciona el modelo de dinámica de presas y predadores de Lotka-Volterra, mediante el uso de dos ecuaciones diferenciales articuladas, donde una detalla el curso de la población de depredadores y la otra de presas; los resultados se calculan frente al tiempo a través del actor Timed Plotter; y, entre sí mediante el actor XY Plotter. En la interfaz de Kepler se exhibe el flujo de trabajo y los componentes se identifican con llamadas.

3.2.5.2 Flujo de trabajo de servicios web

El flujo de trabajo de los servicios web utiliza el actor del servicio web de Kepler para invocar un servicio web de datos genómicos, que accede y consulta una base de datos de genómica remota y devuelve una secuencia genética. El nombre de la secuencia se pasa al actor de servicios web por un actor de constante de cadena. Una vez que el servicio se ha ejecutado, el actor del servicio web genera la secuencia del gen recuperado para que se pueda mostrar en múltiples formatos utilizando tres actores de visualización diferentes: uno para XML, uno para una secuencia de elementos extraídos del XML, y uno para un documento HTML que se puede mostrar en un sitio web. Además, el flujo de trabajo utiliza un cuarto actor de visualización para mostrar los errores devueltos por el servidor remoto (Altintas, 2018).

El actor del servicio web de Kepler permite solicitar un servicio web de datos genómicos teniendo acceso a la base de datos para realizar consultas, generando una secuencia genética que puede ser mostrado en diversos formatos valiéndose de tres actores de visualización, para XML, para una secuencia de elementos extraídos del XML, y para un documento HTML; existiendo además un cuarto actor que permite observar los errores reenviados por el servidor remoto (ver figura 6).

3.2.6 Escenarios de uso

3.2.6.1 Entornos informáticos de portátiles

Los marcos de trabajo para portátiles, como IPython / Jupyter, Beaker y Zeppelin proporcionan un entorno computacional interactivo, en varios idiomas y basado en navegador. Un solo cuaderno puede combinar texto, código y visualizaciones de datos, incluyendo componentes HTML y JavaScript. Al

integrar Kepler en un entorno de portátil de este tipo, podemos permitir que los científicos aprovechen las fortalezas de Kepler utilizando plataformas de ejecución dinámica reproducibles (Perrier, 2018).

IPython / Jupyter, Beaker y Zeppelin son marcos de trabajo que proveen de un entorno interactivo en diversos idiomas al utilizar portátiles; combinando texto, código y visualización de los datos en HTML y Java Script, haciendo la ejecución dinámica reproducible.

Los usuarios pueden hacer llamadas a una API de Kepler en el cuaderno que ejecuta el flujo de trabajo, inspecciona los actores, verifica la procedencia o muestra los resultados. Estas llamadas a la API se envían al servidor JupyterHub, que a su vez las envía al servidor web en el proceso Kepler. La API incluye funciones para ver qué componentes HTML proporciona el flujo de trabajo, es decir, la lista de atributos de WebView y la capacidad de mostrar esos atributos dentro del cuaderno (Crawl, Singh, y Altintas, 2016).

La programación científica en la ciencia de datos se ocupa más de la exploración, la experimentación, la realización de demostraciones, la colaboración y el intercambio de resultados. Es esta necesidad de experimentos, exploraciones y colaboraciones la que abordan los cuadernos para la computación científica. Los cuadernos son entornos de colaboración basados en la web para la exploración y visualización de datos, mediante una API que se envía al servidor Jupyter Hub y éste al servidor web; a través de las funciones de la API se aprecia que componente HTML proporciona el flujo de trabajo. Estos atributos se muestran en el cuaderno.

3.2.6.2 Interactuando con Kepler en un cluster computacional

Cuando se ejecuta Kepler en un recurso HPC como XSEDE, los usuarios se conectan a un nodo de inicio de sesión y envían trabajos a un programador como PBS o LSF; cuando comienza un trabajo, el programador asigna uno o más nodos de proceso en el clúster a la aplicación del usuario. Kepler se ejecuta en uno de estos nodos de cómputo y administra los otros nodos de cómputo (Tabares, 2016).

Kepler para su ejecución utiliza XSEDE que es un recurso HPC como estrategia de alto rendimiento, conectándose al nodo de inicio que envía la información al programador PBS o LSF que asigna los nodos a utilizarse durante el proceso, de los cuales uno funciona como administrador.

El marco de Kepler WebView permite al usuario interactuar con el flujo de trabajo que se ejecuta en un nodo de cómputo. Se puede usar un túnel SSH para conectar el cliente web del usuario con el servidor web que se ejecuta en Kepler. Al conectarse, el cliente web puede proporcionar instantáneamente al usuario el estado del flujo de trabajo. Dependiendo de los componentes HTML utilizados por los atributos de WebView del flujo de trabajo, el cliente web puede mostrar, por ejemplo, gráficos de datos que muestran los resultados de los cálculos que se ejecutan en el nodo de cómputo, o componentes interactivos para dirigir la ejecución del flujo de trabajo. La conexión entre el cliente web y el servidor web es temporal y el flujo de trabajo puede continuar la ejecución después de que se cierre (Crawl, Singh, y Altintas, 2016).

Kepler WebView accede la interactuación en el flujo de trabajo al usuario utilizando un túnel SSH para la conexión con el servidor web, suministrando inmediatamente el estado en que se encuentra el flujo; esta información depende

de los componentes HTML empleados por los atributos de WebView. Una vez que el cliente cierre la conexión la ejecución del flujo de trabajo continúa.

3.2.6.3 Kepler como servicio

Los clientes web se conectan a un equilibrador de carga de front-end, que reenvía las conexiones a un proceso Kepler que se ejecuta en un contenedor Docker o una máquina virtual. Cada cliente se asigna a un Kepler diferente y los procesos de Kepler están aislados entre sí. A medida que cambia la demanda, el equilibrador de carga puede ajustar el número de procesos Kepler (Yatsyk, 2016).

Los clientes web se conectan a una interfaz que funciona como equilibrador de carga reenviando esta conexión al proceso Kepler ejecutado de manera virtual o contenedor Docker; cada proceso es aislado y de acuerdo a la demanda el número de procesos es ajustado por el equilibrador de carga. Ejemplos de flujos de trabajos implementados en Kepler se aprecian en las figuras 7, 8, 9 y 10.

3.3 Funcionalidad del Patrón de Código Abierto Taverna

La mesa de trabajo de Taverna sigue un modelo explícito para la creación de flujos de trabajo. Su punto de entrada principal es el editor de flujo de trabajo que permite a los usuarios arrastrar, soltar y conectar componentes que representan diferentes fuentes de datos y herramientas. El sistema se basa en una arquitectura desacoplada que separa el editor del motor de promulgación (Oinn et al., 2011).

El sistema de flujo de trabajo Taverna tiene un modelo explícito para la creación de los flujos, cuyo acceso es a través del editor que facilita a los usuarios arrastrar, soltar y conectar los componentes; la arquitectura de este sistema separa el editor del motor de promulgación.

Los flujos de trabajo en Taverna se representan internamente en el SCUFL (Simple Conceptual UnifiedFlow Language) para representar flujos de trabajo como DAG (gráficos acíclicos dirigidos). SCUFL soporta predominantemente el modelo de flujo de datos de ejecución. Los nodos en el gráfico representan procesadores que transforman los datos de entrada en datos de salida. Un procesador sin entrada actúa como un origen de datos y un procesador sin salidas actúa como un receptor de datos (Russell, Ter-Hoftede, Aolst Wil, y Mulyar, 2010).

En Taverna la representación de flujos de trabajo como gráficos acíclicos dirigidos – DAG es a través del lenguaje de flujo unificado conceptual simple - SCUFL (Simple Conceptual UnifiedFlow Language) para representar flujos de trabajo como DAG (gráficos acíclicos dirigidos); donde los nodos constituyen los procesadores que pueden transformar los datos de entrada en datos de salida. Cuendo el procesador no tiene entrada actúa como origen de datos y sin salidas como receptor de datos.

3.3.1 Paradigma de modelado

Las barras direccionales entre los nodos son generalmente canales para pasar la salida de un procesador como entrada a otro. Taverna admite la ejecución iterativa de un procesador al proporcionar un conjunto de estrategias de iteración configurables que especifican cómo iterar sobre una lista de entradas. Además, Taverna admite varias construcciones de flujo de control para organizar operaciones de flujo de control. Por ejemplo, un arco que conecta dos nodos puede simplemente indicar una dependencia secuencial entre dos nodos sin datos que fluyen en él, y la separación condicional se logra al pasar

un token especial de "falla" en uno de sus ramas de salida (Sroke y Hidders, 2012).

Los canales de entrada y salida de un procesador se representan con barras direccionales; la ejecución en Taverna es iteractiva suministrando estrategias de iteración configurables, las mismas que son detalladas en una lista de entradas; para la organización de las operaciones de flujo de control se desarrollan varias construcciones de flujo (ver figura 11).

Un flujo de trabajo puede tener cero o más entradas formales que se representan como fuentes. Del mismo modo, las salidas globales de un flujo de trabajo se representan como sumideros. La ejecución de un flujo de trabajo comienza desde las fuentes y termina cuando todos los sumideros han producido su salida o han fallado. Contrariamente a Kepler, Taverna proporciona sólo un modelo de computación (Lee et al., 2012).

En el flujo de trabajo las entradas son representadas como fuentes, mientras que las salidas como sumideros, iniciando la ejecución desde la fuente y culminando todas las salidas en los sumideros o su fallo. Taverna solamente proporciona un modelo a diferencia de Kepler.

La ventaja de la característica es que hace que la dependencia de los parámetros sea explícita y permite que sean controlados fácilmente por el usuario o por otros procesadores que permiten al usuario controlar explícitamente cómo se maneja cada salida. La desventaja clave es que confiar en gran medida en su uso termina generando flujos de trabajo con una gran cantidad de nodos, incluso para una tarea simple (Wohed, Van-der Aalst, Dumas, Ter-Hofstede, y Russell, 2010).

Una ventaja del flujo de trabajo Taverna es la claridad de los parámetros lo que facilita el control por parte del usuario; no obstante la desventaja de este sistema es que genera durante el proceso muchos nodos inclusive para una simple tarea.

3.3.2 Productos de taverna

Hay dos productos Taverna, que son los más adecuados dependiendo del tipo de uso:

- El Workbench o banco de trabajo Taverna es una aplicación que se ejecuta en el escritorio del usuario y se utiliza para crear flujos de trabajo en un entorno de interfaz gráfica de usuario. Workbench es un desarrollo basado en Java y se ejecuta en todos los sistemas operativos dentro de una Máquina Virtual Java.
- El servidor Taverna es para ejecutar flujos de trabajo de forma remota. El servidor permite crear y ejecutar flujos de trabajo que combinan los servicios del investigador y lo ocultan al usuario (Helio, 2015).

Taverna utiliza dos productos, el workbench utilizado en flujos de trabajo con interfaz gráfica, cuyo desarrollo se basa en java ejecutable en los sistemas operativos de una máquina virtual.

3.3.3 Fases en el uso de flujo de trabajo Taverna

Proporciona a los usuarios soporte para varias fases en el uso de flujos de trabajo: descubrimiento de servicios, composición y orquestación, acceso a datos e invocación segura de servicios, que la comunidad caGrid ha identificado como un desafío en un dominio multi-institucional y multidisciplinario:

- Descubrimiento del servicio: dónde encontrar servicios que son relevantes para la investigación científica del usuario.

- Acceso a datos: qué tipo de datos (tipos de datos) se pueden obtener de un servicio determinado y cómo transferir datos desde y hacia ella.
- Interacción del servicio - cómo invocar servicios y mantener la información de la sesión en múltiples intercepciones
- Cumplimiento de la seguridad - cómo forzar la autenticación y autorización en las invocaciones de servicio y la privacidad y la integridad en las transferencias de datos (Tan et al., 2013).

Dentro de las fases para los flujos de trabajo Taverna provee a los usuarios de los soportes respectivos como servicios relevantes para la investigación científica, acceso a la información y la forma de transferirla, interacción de los servicios, mantenimiento de la información dentro de la sesión con múltiples intercepciones, seguridad con autenticación, privacidad e integración en la transferencia de los datos.

3.3.4 Especificación de flujo de trabajo Taverna

Un flujo de trabajo de Taverna se define en términos del lenguaje SCUFL2, un formato XML que se extiende a W3C PROV. Los flujos de trabajo de Taverna son principalmente impulsados por datos, aunque se pueden expresar para cada bucle (Socland, Bacall, y Holubowicz, 2014).

Taverna utiliza lenguaje SCUFL2 con formato XML que se extiende a W3C PROV, donde los flujos de trabajo son impulsados por datos, que pueden ser expresados para cada bucle. Ejemplos de flujos de trabajo implementados en Taverna se aprecian en las figuras 12 y 13

Los procesadores pueden asociarse con programas locales (script bean, R-script) o programas remotos (servicio REST, servicio en la nube). Un usuario puede especificar un flujo de trabajo desde cero o utilizar flujos de trabajo

existentes como flujos de trabajo integrados. Taverna es en principio independiente del dominio. Sin embargo, se ha utilizado ampliamente en las ciencias de la vida. Taverna también incorpora capacidades de búsqueda para recuperar flujos de trabajo y herramientas de myExperiment y biocatalogue, respectivamente (Cohen et al., 2017).

En este sistema puede existir asociación de los procesadores con programas tanto locales como remotos; pudiendo iniciar un flujo de trabajo o integrar uno ya existente. A pesar de ser de dominio independiente se emplea generalmente en ciencias de la vida, incorporando capacidades de búsqueda en la recuperación de flujos en myExperiment y herramientas en biocatalogue.

3.3.5 Ejecución de flujo de trabajo Taverna

Taverna está equipada para capturar la procedencia retrospectiva de las ejecuciones de flujos de trabajo utilizando Taverna-PROV, un modelo que extiende PROV y WfPROV para capturar características que son específicas de los flujos de trabajo de Taverna, en particular iteraciones; proporciona un panel de resultados básico para examinar los resultados de las ejecuciones de flujos de trabajo. No se admiten las consultas sofisticadas para rastrear el linaje de artefactos de datos o comparar los resultados de la ejecución múltiple de flujos de trabajo iguales o diferentes (Belhajjame et al., 2013).

De acuerdo a lo expresado por los autores el panel se usa principalmente para mostrar una entrada de procesador determinada, así como los artefactos de datos que se utilizan durante la ejecución.

Taverna aún no admite la virtualización mediante un mecanismo como Docker, aunque existen esfuerzos de desarrollo actuales que permitirán a los usuarios crear una imagen de Docker que ejecute un procesador Taverna. Además

viene con un complemento para usar los recursos de grilla de UNICORE. Con respecto al empaquetado científico, se ha demostrado cómo los flujos de trabajo taverna pueden integrarse en objetos de investigación reproducibles (Belhajjame et al., 2015).

Taverna no utiliza mecanismo Docker para la virtualización, empleando recursos de grilla de Unicore; los flujos de trabajo pueden integrarse en objetos de investigación reproducibles.

3.3.6 Limitaciones en el contexto de la reproducibilidad

Taverna permite a los diseñadores de flujos de trabajo utilizar servicios web de terceros en la composición de sus flujos de trabajo. Esta dependencia de servicios web volátiles de terceros significa que los flujos de trabajo no pueden ejecutarse debido a la falta de disponibilidad de uno o más servicios utilizados dentro del flujo de trabajo en cuestión (Zheng, Ratnakar, Gil, y McWeeney, 2015).

Los diseñadores de flujos de trabajo Taverna pueden usar servicios web que implica que no se podría ejecutar si no existiera la disponibilidad de uno o más servicios que se utilicen en el desarrollo del flujo.

El flujo de trabajo de la taverna tiende a contener un número importante de procesadores que se utilizan para realizar la transformación del formato de datos básicos. De hecho, la dependencia de servicios web de terceros que adoptan diferentes estructuras de datos para su entrada y salida, generan la necesidad de utilizar correcciones de compatibilidad para resolver las discrepancias entre los servicios en un flujo de trabajo (Albouelhoda, Issa, y Ghamann, 2012).

Generalmente el flujo de trabajo de taverna utiliza algunos procesadores para la trasformación de datos; además la dependencia de servicios web crea la necesidad de realizar correcciones de compatibilidad con la finalidad de solucionar las incompatibilidades entre los servicios dentro del flujo de trabajo.

Como resultado, la especificación del flujo de trabajo se vuelve más compleja de lo que debería ser, ya que contiene procesadores que no contribuyen al análisis general de datos implementado por el flujo de trabajo y, por lo tanto, pueden dificultar que un usuario potencial entienda el experimento implementado por el flujo de trabajo basado únicamente en la especificación del flujo de trabajo (Zhao et al., 2012).

Estos inconvenientes hacen compleja la especificación del flujo de trabajo, dificultando al usuario entender la investigación científica implementada por el flujo.

3.3.7 Presentación de la ejecución del flujo de trabajo

Taverna brinda poco apoyo para su exploración, por ejemplo, es difícil rastrear el linaje de un resultado dado. Este es particularmente el caso cuando algunos de los procesadores de flujo de trabajo se ejecutan varias veces, a través de iteración, dentro de una sola ejecución (Kurs, Simi, y Campagne, 2016).

La exploración en Taverna es complicada, debido a que este sistema no brinda el apoyo necesario, sobretodo cuando algunos procesadores se ejecutan simultáneamente mediante la iteración.

Taverna también tiene un soporte limitado para la virtualización. Aunque actualmente hay esfuerzos del equipo de Taverna en esta dirección; por otro lado este sistema no brinda soporte para entornos de nube (Albouelhoda, Issa, y Ghamann, 2012).

Además este sistema brinda un escaso soporte para la virtualización y ninguna para entornos de nube.

3.4 Diferencias entre los sistemas de gestión de flujo de trabajo Triana, Kepler y Taverna

Triana es un entorno de solución de problemas gráficos de código abierto que permite a los usuarios ensamblar y ejecutar un flujo de trabajo a través de una interfaz gráfica de usuario mientras minimiza la carga de la programación. Triana es utilizada principalmente por la comunidad informática para respaldar la investigación del flujo de trabajo.

Kepler es un sistema de gestión y modelado de flujo de trabajo científico que permite a los usuarios, independientemente de la experiencia de programación, configurar los procesos de análisis de datos.

Taverna es un sistema de gestión diseñado para ensamblar, ejecutar, documentar y compartir flujos de trabajo científicos.

En la tabla 3 se describen las características de cada uno de los sistemas de flujo de trabajo científico, como son Kepler, Triana y Taverna, detallando los desarrolladores, versión, plataforma, lenguaje, licencia y dominio.

La comparación de los sistemas de flujo de trabajo científico Kepler, Triana y Taverna en base a la integración estratégica, procesos de integración y reutilización se presenta en la Tabla 4, donde se puede apreciar que las tres herramientas utilizan como estrategia de integración ambiental while box más encapsulación. En relación al modelo de datos tanto Taverna como Triana se basan en cliente y servicio, mientras que Kepler tiene una interfaz base extensible basada en actor y parámetro. Taverna y Triana basan sus servicios en WSDL, el primero en Biocatálogo y el segundo en repositorios o servicios P2P; mientras que

Kepler cuenta con interfaces de programación orientada al actor, contando con soporte para servicios web, OGC y Cloud al igual que Taverna; mientras que Triana cuenta con soporte para servicios web, P2P, Grid, WSRF. En relación al lenguaje de programación las tres herramientas utilizan Java. Cada una de lso work Flow tienen diferentes métodos de composición; Kepler y Triana cuentan con un buen soporte de procedencia de los datos como parte del flujo de trabajo; mientras que Taverna se soporta conectado a MiExperimento y los datos se manejan fuera del sistema. El dominio tiene un foco genérico en los flujos Kepler y Taverna, pero el fuerte de Kepler ees el procesamiento molecular y la biología; Taverna se enfoca en las Ciencias de la vida y la bioinformática; distintamente el dominio de Triana es en minería de datos y estadísticas. Los tres workflow disponen de herramientas front-end, pero Kepler y Taverna cuentan con código fuente y binario en el sitio web; y, Triana cuenta con un código fuente disponible desde el servidor y el binario en forma de sitio web.

Luego de la presentación de estos tres sistemas, se comparan con respecto a algunas características sintácticas y aspectos de control y flujo de datos.

- **Características sintácticas**

Varias características sintácticas son relevantes para la discusión.

La capacidad de un nodo para tener múltiples puertos que producen diferentes tipos de salida, está presente en todos los sistemas. El espacio variable compartido no está presente en los sistemas de flujo de datos enumerados aquí, ya que la comunicación de datos se realiza a través de enlaces de componentes. Finalmente, Triana y Kepler adjuntan información de tipo a los datos que se pasan y aseguran que la composición sea segura.

- **Comportamiento de control**

Los elementos de control y separación de datos existen en estos sistemas de flujo de trabajo en tres formas distintivas: como diferentes tipos de flujo de trabajo, como diferentes subconjuntos de nodos que implementan tanto el control como la funcionalidad de datos en los mismos flujos de trabajo, o como diferentes tipos de enlaces que transportan control o datos.

- **Capas separadas.** Kepler define el control utilizando directores que determinan el comportamiento del nodo de flujo de trabajo.
- **Nodos separados.** Triana tiene un conjunto de nodos de control que se utilizan para lograr la ramificación, el paralelismo y el bucle.
- **Control dedicado y enlaces de datos.** Taverna utiliza enlaces de control, que no pasan datos, para sincronizar la ejecución de componentes sin dependencia de datos. No hay ningún bucle presente.

- **Comportamiento de los datos**

La ejecución del flujo de datos puede organizarse de una manera controlada por datos o puede ser ejecutada al proporcionar a cada nodo instrucciones detalladas sobre cómo comportarse cuando los datos estén disponibles.

- **Ejecución basada en datos.** Taverna y Triana mantiene ejecución en todos los nodos sin predecesores y continúa hasta que no haya más nodos para ejecutar.
- **Ejecución basada en instrucciones.** Cada nodo recibe instrucciones sobre cómo comportarse con respecto a los datos. En Kepler, este comportamiento es común a todos los nodos en la capa y está definido por el director.

- **Incrustación**

Cuando se trata de combinar aspectos de control y datos, Triana y Kepler proporcionan dichos mecanismos a través de construcciones de incrustación en las que un flujo de trabajo de un tipo se integra en el flujo de trabajo de otro tipo.

Triana realiza el control del flujo según patrón computacional asociando la lógica de coordinación explícita con un flujo de trabajo integrado mediante el uso de scripts; el uso principal de este mecanismo es crear construcciones de bucle avanzadas, pero generalmente permite la creación de cualquier lógica imperativa.

- **Ductos computacionales y de análisis de datos**

El sistema Kepler integra mecanismos para admitir flujos de datos de tuberías. La comunicación entre actividades en Kepler se realiza mediante la conexión de puertos de entrada y salida entre ellas. Cada tarea puede leer desde sus puertos cuando una secuencia de datos esté disponible para ella.

- **Integración de servicios**

Taverna se enfoca en el nivel de servicio web para permitir a los investigadores realizar experimentos que involucran el uso de recursos locales y remotos en biología. Triana admite la integración de múltiples servicios e interfaces que incorporan la verificación de tipos de tiempo de ejecución y la conversión de tipos de datos para admitir tipos de datos complejos

- **Barridos de parámetros**

Nimrod es una familia de herramientas para el barrido exhaustivo de parámetros en flujos de trabajo donde se realiza un solo cálculo muchas veces. El soporte para flujos de trabajo está disponible a través de Nimrod / K, que incorpora las capacidades de Kepler en la creación de flujos de trabajo

complejos, y la capacidad de Nimrod para ejecutar barridos sobre recursos distribuidos (Kitowski, 2016).

En flujos de trabajo se utiliza Nimrod para el barrido de parámetros cuando se repite muchas veces un mismo cálculo, cuyo soporte se encuentra en Nimrod / K, incluyendo las capacidades de Kepler en la creación de flujos de trabajo complejos,

En la tabla 5 se observa un análisis de los sistemas de flujo de trabajo científico en base a la arquitectura, donde Kepler, Triana y Taverna proporcionan un buen soporte de interacción del usuario, sin embargo es deficiente el soporte para la personalización de la interfaz de usuario, debido a un estrecho acople entre las interfaces del sistema y los subsistemas de tiempo de ejecución. Los tres sistemas soportan actualmente la procedencia, enfatizando la importancia de la procedencia en SWFMS. No obstante, la reutilización del subsistema de procedencia en otros SWFMS es difícil, debido a que el módulo de procedencia está estrechamente relacionado con su propietario. Taverna, Kepler y Triana tienen soporte parcial para la integración de servicios heterogéneos y herramientas de software. En cuanto a la gestión de productos de datos, Kepler y Taverna trabajan en los niveles de los mensajes XML, archivos y registros de la base de datos. Kepler admite la noción de colecciones de datos anidados mediante el uso de actores orientados a la recopilación personalizada - coactores. En lo relacionado a la computación de alta gama, Kepler y Taverna proporcionan tareas personalizadas para comunicarse con el entorno Grid, mientras que Triana usa la interfaz GAT para acceder a los trabajos Grid. Actualmente estos tres SWFMS brindan cierto grado de soporte para la supervisión del flujo de trabajo y

el manejo de fallas; y, por último la interoperabilidad está mal soportada en todos estos SWFMS.

3.4.1 Ejemplo de un modelo conceptual de flujo de trabajo científico para ser utilizado en los sistemas Kepler, Taverna y Triana

Verdi, Ellis, y Gayk (2015) desarrollaron el proceso de construcción del modelo conceptual del flujo de trabajo científico para el análisis biomoleculares utilizando espectroscopía de RMN. El proceso consta de las siguientes fases de modelado: el contexto modelo que proporciona una vista estructural de los procesos y sus sub-procesos; el proveedor-productor-Input-Output de los Consumidores (SIPOC) modelo que pone de manifiesto que los datos son producidos y consumidos y la forma en que las corrientes durante un experimento, y el control de flujo modelo que demuestra el flujo y la orden de los pasos en el experimento. Cada fase consta de un proceso de dos fases de diseño y validación del modelo, como se muestra en la figura 14.

El primer paso en el proceso de determinación de la RMN experimento modelo de flujo de trabajo es crear un modelo de contexto, que proporciona una perspectiva estructural en el experimento proceso mediante la descripción de los principales procesos y sus sub-procesos en un flujo de trabajo. Se ha modelado el experimento usando una vista jerárquica de la reunión de alto nivel de los procesos, para demostrar la relación entre el super-flujo de trabajo y sub-flujo de trabajo, como se muestra en la figura 15.

En la segunda fase de modelado, los procesos identificados en el contexto modelo son más detallados correspondientes con los productores, los datos de entradas y salidas, y los consumidores utilizando un modelo SIPOC, que se basa

en el proveedor-productor-cliente (CPE) la cadena, una probada técnica de modelado utilizados en Proceso de Mejora Continua.

La Figura 16 muestra el modelo SIPOC para el análisis de la muestra de nivel superior proceso identificado en el contexto de modelado fase; recogiendo los datos de entrada de muchos tipos de proveedores; *r*ecoger parámetros de la muestra, recoger datos de RMN y segregar el control de datos. Otros aportes provienen de fuentes humanas: la RMN Analizador de *proceso* de datos, así como información de referencia depositado en la literatura y el internet. Todos estos datos son requeridos por el proceso de analizar la muestra para producir un resultado final, que se compone tanto de los datos resultantes, así como la solución para su publicación.

En la tercera fase del proceso se captura el flujo de control del flujo de trabajo científico, utilizando una notación para representar a los flujos de trabajo científicos como diagramas de flujo, siendo un sencillo y eficiente método para identificar las actividades necesarias para ejecutar un proceso. Descomposición de diagramas de flujo es posible si el proceso primario no proporciona suficiente detalle para el análisis del proceso. El menor de descomposición que necesitábamos para capturar a nuestros conceptual a nivel de vista del proceso fue de cuatro niveles de descomposición.

La figura 17 muestra el control de flujo para el proceso de análisis de la muestra, que emplea los siguientes componentes:

- Inicio - un círculo se utiliza para representar el punto de partida del proceso.
- End - un óvalo se utiliza para indicar el punto de terminación del proceso.

- Actividad - una tarea o paso necesario para ejecutar el proceso, representados por un simple rectángulo.
- Tratamiento externo - un proceso que es externo al flujo de trabajo que indica el control de flujo de trabajo de las salidas a un proceso externo. Representado por un rectángulo.
- Flujo - Para la secuenciación de las actividades del proceso, representado por una flecha dirigida. Un ejemplo de las corrientes utilizadas en los flujos de trabajo es la conexión entre el *constructo del espectro* y *analizar* las actividades *del espectro.*
- La Decisión - un punto de decisión en el flujo de control donde el control puede ir una de las dos direcciones, representado por un diamante. La decisión identifica la dirección en la que la corriente se mueve sobre la base de la respuesta a un verdadero / falso. Por ejemplo, los picos son asignados hasta que los resultados se validan como demuestran los *resultados validados?* Decisión.

3.4.1.1 Enfoque conceptual de modelado

Esto corresponde a la creación de flujos de trabajo científico conceptual que abarca todo el proceso de un experimento. Esta implementación podría llevarse a cabo utilizando un sistema de gestión de flujo de trabajo científico, como Kepler, Taverna o Triana. Una vez que se elige un sistema, los modelos conceptuales se convertirán en los modelos de implementación correspondientes, comenzando con el proceso de muestra analizada descompuesto. Los diagramas de actividad de UML 2.0 permitirían modelar tanto el flujo de control como el flujo de datos en un diagrama de modelo y UML es el estándar de factor de la industria para los

requisitos de software. Además, los flujos de trabajo de nivel conceptual se pueden convertir fácilmente a notación UML 2.0.

3.4.2 Impacto de los sistemas de flujo de trabajo en el Ecuador

En el país existen trabajos de workflow o sistemas de flujo de trabajo empleados en actividades administrativas; sin embargo para la elaboración de investigaciones científicas no se ha encontrado ningún repositorio que conlleve el uso de sistemas de flujos de trabajo; como se puede apreciar en diferentes trabajos investigativos en Europa o norte América, tales como:

- Gestión de la información ambiental en los espacios protegidos y en las redes de seguimiento del cambio global - España
- Análisis espaciales con énfasis en modelos de nicho ecológico – México
- Modelos de distribución de especies – España
- La Red de Información Ambiental de Andalucía – España
- Resumen de la estrategia de especies amenazadas - Australia

En diferentes trabajos de titulación presentados en universidades del Ecuador solamente se utilizan los sistemas de flujo de trabajo para casos administrativos como son:

- Implantación de un workflow basado en la arquitectura SOAP, utilizando un modelo de servicios de negocios en la Jefatura de Estado Mayor Institucional del Comando Conjunto de las Fuerzas Armadas del Ecuador
- Metodología de implementación de un sistema de Workflow para mejorar el proceso administrativo y comunicativo en la Facultad de ciencias de la educación de la Universidad Estatal de Bolivar.
- Aplicación de sistemas workflow en la gestión académica.

- Implementación de un sistema workflow en el proceso de adquisición de bienes y servicios en el departamento de Compras Públicas de la Universidad Estatal Amazónica de la ciudad de Puyo.
- Evolución de diseños basados en la naturaleza: diseño de un juguete didáctico para entender a las arañas.
- Desarrollo e implementación de un workflow para la unidad de titulación de la carrera de Ingeniería en Sistemas computacionales de la Universidad Católica Santiago de Guayaquil

4. Conclusión

Para cerrar esta monografía se puede decir que estos sistemas de flujos de trabajo funcionan de acuerdo a las necesidades que se requieran, ya que cada uno tiene su particularidad.

Triana utiliza su propio lenguaje de flujo de trabajo personalizado, aunque puede usar otras representaciones de lenguaje de trabajo externas. Triana viene con una amplia variedad de herramientas integradas para el análisis de señalización, la manipulación de imágenes, la publicación de escritorio, entre otros, y tiene la capacidad para que los usuarios integren fácilmente sus propias herramientas. El marco de Triana se basa en una arquitectura modular en la que la GUI se conecta a un motor de Triana, llamado Triana Controlling Service (TCS), ya sea de forma local o remota. Un cliente puede iniciar sesión en un TCS, componer y ejecutar una aplicación de forma remota y luego visualizar el resultado localmente.

Kepler se basa en el concepto de directores, que dictan los modelos de ejecución utilizados en un flujo de trabajo. Los pasos de flujo de trabajo individuales se implementan como actores reutilizables que pueden representar fuentes de datos, sumideros, transformadores de datos, pasos analíticos o pasos de cálculo arbitrarios. Cada actor puede tener uno o más puertos de entrada y salida, a través de los cuales fluyen flujos de tokens de datos, y pueden tener parámetros para definir un comportamiento específico. Kepler es un sistema de flujo de trabajo de código abierto basado en Java y en el sistema Prolemy II y presenta una arquitectura monolítica con varios módulos de extensión para las funcionalidades necesarias para los flujos de trabajo científicos. Una vez definido,

los trabajos de Kepler se pueden intercambiar mediante un formalismo basado en XML.

El sistema Taverna demuestra que la capa de flujo de ejecución es responsable de la programación del flujo de trabajo, el descubrimiento de servicios, la administración de datos y metadatos; y la capa Invocación del procesador es responsable de la invocación de servicios concretos. El sistema Triana tiene una interfaz gráfica de usuario sofisticada para la composición y modificación del flujo de trabajo, incluidas las funciones de agrupación, edición y zoom. Desde el campo de la onda gravitacional, el sistema contiene un gran repositorio de herramientas para el análisis y procesamiento de datos.

El objetivo principal de este sistema es apoyar a la comunidad de ciencias de la vida (biología, química y medicina) para diseñar y ejecutar flujos de trabajos científicos y apoyo en la experimentación silica, donde la investigación se realiza a través de simulaciones por computadora con modelos que responden al mundo real. Aunque la mayoría de las aplicaciones de Taverna se encuentran en el dominio de la bioinformática, se puede aplicar a una amplia gama de campos, ya que puede invocar cualquier servicio web simplemente proporcionando la URL. Esta característica es muy importante para permitir a los usuarios reutilizar el código que está disponible en Internet. Por lo tanto, el sistema está abierto a código heredado de terceros al proporcionar interoperabilidad con servicios web.

En el Ecuador el uso de sistemas de flujo de trabajo se observa en actividades administrativas; no se ha encontrado ningún repositorio que evidencie su utilización en investigaciones científicas, a diferencia del continente europeo y de los países de norte América, donde sí se emplean estos sistemas en diversos estudios ambientales, ecológicos, informáticos, entre otros.

5. Recomendaciones

De acuerdo a lo expuesto en el presente trabajo monográfico se realizan las siguientes recomendaciones:

Fomentar el uso de los sistemas de flujos de trabajo científicos, ya que esto permitirá a los investigadores organizar sus proyectos y datos, al tiempo que mantienen un registro continuo de cómo se obtuvieron los resultados mediante una combinación de operaciones manuales y flujos de trabajo científicos automatizados.

Capacitar el uso de aplicaciones analíticas avanzadas como son la integración de datos, flujos de trabajo y la interoperabilidad entre diferentes sistemas.

Profundizar estudios sobre los sistemas de gestión de flujos de trabajo como herramienta en los procesos de investigaciones científicas.

6. Bibliografía

Abdul, M. (2015). Sistema de gestión de flujos de trabajo científicos escalables SWFMS. *Revista internacional de informática avanzada y aplicaciones*, Vol. 7 pp 11.

Albouelhoda, M., Issa, S., Ghamann, M. (2012). Tavaxy: integración de flujos de trabajo taverna y galaxy con soporte de computación en la nube. *BMC Bioinformatica*, 13.

Altintas, I. (2018). *El flujo de trabajo de servicios web*. Obtenido de Universidad de San Diego: https://www.sdsc.edu/

Atkinson, M., Gesing, S., Montagnot, J., Taylor, I. (2017). Flujos de trabajo científicos: Pasado, presente y futuro. En *Sistemas de computación de generación futura* (págs. 216-227). Elsevier.

Belhajjame, K., Chao, J., Garijo, D., Gamble, M., Hellme, K., Palma, K., . . . Goble, C. (2015). Utilizando en conjunto de ontologías para preservar objetos de investigación centrado en el en el flujo de trabajo. *Revista de Semántica Web*, Vol. 32 pp 16-42.

Belhajjame, K., Graham, K., Garijo, D., Corcho, O., García, E., Palma, R. (2013). *Especificaciones de flujo de trabajo - wfprov*. Obtenido de http://wf4ever.github.io/ro/#wfprov

Bhatt, Gupta, Kitchens. (2010). *Sistemas de Flujo de Trabajo.*

Caro, J. (2014). Tecnología Workflow: Estado actual de la investigación. *Universidad de Málaga*.

Churches, D., Gombas, G., Herrison, A., Maassen, J., Robinson, C., Shields, M., . . . Wang, I. (2011). Programación de flujos de trabajo científico y distribuido con los servicios de triana. *Concurrencia y cálculo*, Vol. 18 pp 1021-1037.

Cohen, S., Belhajjame, K., Collin, O., Chopard, J., Froidevoux, G., Gaignard, A., . . . Lemoine, F. (2017). *Flujos de trabajo científicos para la reproducibilidad computacional en las ciencias de la vida, estado, desafíos y oportunidades.* Elsevier.

Costan, A., Stratan, C., Tirson, E., Ionut, M., Cristea, V. (2011). *Hacia una plataforma grid para getión de flujos de trabajos científicos.* USA.

Crawl, D., Singh, A., Altintas, I. (2016). Kepler web view: un marco ligero y portatil para construir interfaces web en tiempo real de flujos de trabajo científico. *Procedia Comput. Sci*, Vol. 80 pp 673-679.

Curcin, V., Ghanem, M. (2013). Sistemas de flujo de trabajo científico: ¿Puede un tamaño adaptarse a todos? *Conferencia de Ingeniería biomédica* (págs. 1-9). Cairo: CIBEC.

Deelman, E., Carother, C., Mandal, A., Tiernez, B., Vetter, J., Baldín, I., . . . Lynch, V. (2017). Panorama: una aproximación al modelado del rendimiento y el diagnóstico de flujos de trabajo de escala extreme. *Revista internacional de aplicación informática de alto rendimiento*, Vol. 31 pp 4-18.

ERP Kepler. (2019). *Kepler características*. Obtenido de https://www.kepler.com.mx/erp.html#textosencillo

Ferreira, R., Filgueira, R., Pietri, I., Jiang, M., Sakellarious, R., Didman, E. (2017). *Una caracterización de los sistemas de gestión de flujo de trabajo para*

aplicaciones de escala extrema. California, EEUU: Universidad del Sur de California.

García, F., Casado, R., Pérez, R., Benito, B. (2012). *Diseño y creación de un repositorio de modelos para la red de información ambiental de Andalucía. Análisis y evaluación del sistema Kepler.* Granada, España: Universidad de Granada.

Goble, C. A., Bhagat, J., Aleksejevs, S., Cruickshank, D., Michaelides, D., Newman, D., . . . Li, P. (2010). MyExperiment: un repositorio y una red social para el intercambio de flujos de trabajo de bioinformatica. *Nucleic Acids Res.*, Vol 38 pp W677 . W682.

Guan, Z., Hernández, F., Bangalore, P., Gray, J., Skjellum, A., Velusamy, V., Liu, Y. (2010). *Grid - flow: un sistema de flujo de trabajo científico habilitado para la red con una interfaz basada en red de petri.* Alabama, EEUU: Universidad de Alabama.

Helio. (2015). *Capacidades de flujo de trabajo*. Obtenido de http://helio-vo.eu/capabilities/workflows.php

Joyce, J. (2016). *Gestión del flujo de trabajo*. Obtenido de https://www.labmanager.com/laboratory-technology/2016/02/managing-workflow#.XGZo5VVKjIU

Kano, Y., Dobson, P., Nakanishi, M., Tsuju, J., Ananiadou, S. (2010). Extracción de texto cumple con flujo de trabajo: vinculando U-compare con taverna. *Bioinformatics*, Vol 26 pp 2486 - 2487.

Kepler. (2016). *Biokepler 1.2 publicado*. Obtenido de https://kepler-project.org/users/whats-new/biokepler-1.2-released

Kepler. (2016). *Proyecto Kepler*. Obtenido de https://kepler-project.org/

Kitowski, K. (2016). Servicios autoescalables en software orientado a servicios para el cultivo de datos rentables. *Sistemas de computación de generaciones futuras*, Vol. 54 pp 1-15.

Kurs, J., Simi, M., Campagne, F. (2016). *Flujos contínuos en work bench: flujos de trabajo reproducibles y reutilizables para principiantes y expertos*. Obtenido de https://www.biorxiv.org/

Lee, E. A., Hylands, C., Janneck, J., Davis, J., Liu, J., Liu, X., . . . Whitaker, P. (2012). Descripción general del proyecto Ptolomeo. *Informe Técnico UCB/ERL M01/11*. Berkeley, USA: Universidad de California.

Liu, J., Pacititti, E., Valdariez, P., Maltoso, M. (2015). Una encuesta sobre gestión de flujo de trabajo científico intensivo en datos. *Revista de computación Grid*, 13 (4) pp 457-493.

Ludascher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., . . . Zhao, Y. (2013). Workflow Management científico y el sistema Kepler. *Concurrencia y Conputación: Práctica y Experiencia.*, (págs. 18: 1039-1065).

Migliorini, S., Gambini, M., La Rosa, M., Ter-Hofstede, A. H. (2014). *Evaluación de patrones. Sistema de gestión base de flujo de trabajo científico.* Australia: Universidad de Queensland.

Nguyen, M., Crawl, D., Mosoumi, T., Altintas, I. (2016). Aprendizaje automático integrado en el sistema de flujo de trabajo científico kepler. *Conferencia internacional de ciencia computacional.* (págs. Vol. 80 pp 2443-2448). Elsevier.

Perrier, A. (2018). *Jupyter, Zepelin, Beaker: el ascenso de los cuadernos. Ciencia de datos abiertos*. Obtenido de Corporación Medium: https://medium.com

Presidencia de la república. (2010). Ley de Educación Superior. *Registro Oficial No. 298*. Quito, Ecuador: Lexis S.A.

Presidencia de la República del Ecuador. (2010). Ley Orgánica de Educación Superior. *Registro Oficial. Suplemento 298*. Quito, Ecuador.

Russell, N., Ter-Hoftede, A. H., Aolst Wil, Mulyar, N. (2010). Los patrones de flujo de trabajo de control de flujo: Una visión revisada. *Informe técnico BPM 06-22*. Centro de Informes.

Rybinski, M., Lula, M., Banasik, P., Lasota, S., Gambin, A. (2012). Tav 45B: herramientas integradas para el análisis de modelos cinéticos de sistemas biológicos. *BMC Systems Biologics*, Vol 6 pp. 25.

Salado-Cid, R., Luque, G., Romero, J. (2015). Sistemas de gestión de flujos de trabajo para la definición visual de aplicaciones basadas en algoritmos evolutivos. *Acta del XVI Conferencia CAEPIA* (págs. 261-270). España: CAEPIA.

Salado-Cid, R., Romero, J., Ventura, S. (2016). *Metaherramienta para la generación de aplicaciones cinetíficas basadas cen workflows .* España: Universidad de Cpordova.

Shields, M. (2012). Programación del flujo de trabajo científico y distribuido con los servicios de treiana. *Taller de flujo de trabajo*. Berlín, Alemania.

Socland, S., Bacall, F., Holubowicz, P. (2014). Scuft 2 - wfdes 0.3.

Sonntag, M., Karastoganova, D., Leimann, F. (2010). *Las características faltantes de los sistemas de flujo de trabajo para cálculos científicos.* Alemania: Universidad de Stattgart.

Sroka, J., Hidders, J., Missier, P., Goble, C. (2010). Una semántica formal para el modelo de trabajo Taverna. *Journal of computer and system sciences*, 76 (6) 490 - 508.

Sroke, J., Hidders, J. (2012). Hacia una semantica formal para el modelo de proceso del taverna workbench. *Fundamenta Informaticae*, 92: 279 - 299.

Tabares, R. (2016). *Programación paralela sobre arquitecturas heterogéneas.* Manizales, Colombia: Universidad Nacional de Colombia.

Talia, D. (2013). Sistemas de flujo de trabajo para la ciencia: conceptos y herramientas. *International scholarly research notices.*

Tan, W., Madduri, R., Nenadic, A., Soiland, S., Sulakhe, D., Foster, I., Goble, C. (2013). CaGrid Workflow: Una herramienta de flujo de trabajo basada en Taverna para la cuadrícula de cáncer. *BMC Bioinformática*, Vol. 11 (1) pp 542.

Taverna. (2015). *Guía de inicio rápido.* Obtenido de http://www.taverna.org.uk/documentation/taverna-2-x/quick-start-guide/

Taylor, I., Majithio, S., Shields, M., Wang, I. (2013). *Especificaciones de sistema de flujo de trabajo Triana.* USA: Grid Lab.

Taylor, I., Shields, M., Wang, I., Harrison, A. (2011). El entorno de flujo de trabajo de triana: arquitectura y aplicaciones. En D. Gannon, M. Shields, *Flujo de trabajo para la e-ciencia.* USA: Springer.

Triana. (2016). *Triana, software de resolución de problemas de código abierto*. Obtenido de www.trianacode.org

Truszkowske, A., Jayaseelan, K. V., Neumann, S., Willighagen, E. L., Zielesny, A., Steinbeck, C. (2011). Nuevos desarrollos en el entgorno de flujo de trabajo abierto de Chemiinformatics CDK - Taverna. *J. Chemisiform.*, Vol. 3 pp 54.

Universidad Agraria del Ecuador. (2014). *Normativa de ética para los procesos de investigación y de enseñanza - aprendizaje - práctica - comprensión*. Obtenido de http://www.uagraria.edu.ec/documentos/reglamentos/NORMATIVA-DE-ETICA-PARA-LOS-PROCESOS-DE-INVESTIGACION.PDF

Vatri, K., Harvey, I., Samak, T., Gunter, D., Evans, K., Rogers, D., . . . Al-Shakarchi, E. (2013). Un estudio de casos sobre el uso de infraestructura común de monitoreo de flujo de trabajo para flujos de trabajo científicos. *Journal of grid computing*, Vol. 11 pp 381-406.

Wilde, M., Hategan, M., Wozmiak, J., Clifford, B., Katz, D., Foster, I. (2016). Swift: Un lenguaje para escritura paralela distribuida. *Computación paralela*, 37 (9) pp 633-652.

Williams, R. (2018). *Lotka-Volterra Workflow*. Obtenido de Centro Nacional de análisis y síntesis ecológicos: https://www.nceas.ucsb.edu/

Wohed, P., Van-der Aalst, W. M., Dumas, M., Ter-Hofstede, A. H., Russell, N. (2010). Sobre la idoneidad de BPMN para la modelización de procesos de negocios. *Conferencia Internacional Business Process Management BPM* (págs. Vol. 4102 pp 161-176). Viena, Austria: Springer.

Wolstencroft, K., Haines, R., Felloub, D., Williams, A., Winthers, D., Owen , S. (2013). El paquete de flujo de trabajo de taverna: diseño y ejecución de flujos de trabajo de servicios web en el escritorio, la web o la nube. *Nucleic Acids Research*, Vol. 41.

Workflow Patters. (2011). *Pattern 5 (Simple Fusionar). Animación flash de patrón simple fusionar.* Obtenido de http://www.workflowpatterns.com/patterns/control/basic/wcp5.php

Yatsyk, Y. (2016). *Gestión elástica de clusters de contenedores.* Valencia, España: Universidad Politécnica de Valencia.

Yuan, D., Yang, Y., Chen, J. (2013). *Cálculo y almacenamiento en la nube.* USA: Elsevier.

Zhang, C., De Sterck, H. (2017). *Cloud WF: un sistema de flujo de tgrabajo computacional para nubes basado en Hadoop*. Obtenido de https://www.researchgate.net/profile/Chen_Zhang47/publication/225269059.pdf

Zhao, J., Gómez, J., Belhajjame, K., Klyne, G., García, E., Garrido, A., . . . Goble, C. (2012). Por qué se rompen los flujos de trabajo: comprender y combatir la descomposición en los flujo de trabajo de Taverna. *8° conferencia internacional IEEE sobre E-ciencia* (págs. 1-9). Chicago. EEUU: IEEE.

Zheng, C., Ratnakar, V., Gil, Y., McWeeney, S. (2015). Uso de flujos de trabajo semánticos para mejorar la transparencia y la reproducibilidad en ómicas clínicas. *Genome Medicina*, Vol 7.

7. Glosario

Actividad automática: una actividad automática es un paso de un proceso de flujo de trabajo que está completamente automatizado y bajo circunstancias normales no es necesaria la intervención humana para la terminación de tal paso. Un paso de actividad automática invoca un método de la aplicación para realizar algún proceso necesario como parte del proceso de negocio global.

Administración de flujo de trabajo: el sistema de gestión de trabajo de Cúram proporciona funciones de administración de flujo de trabajo que permiten a los administradores supervisar y controlar las instancias de proceso que son ejecutadas por el motor de flujo de trabajo.

Asignación de trabajo: cuando se crea una tarea (o se entrega una notificación) a raíz de la ejecución de una actividad, se debe direccionar esa tarea o notificación a un usuario o grupo de usuarios específicos para que se accione.

Beanshell: es un lenguaje de scripts Java. Un servicio Beanshell en Taverna le permite escribir fragmentos de código Java simples y ejecutarlos como parte de sus flujos de trabajo.

Diagrama de flujo de trabajo: contiene la imagen del diagrama de flujo de trabajo. Se puede utilizar para modificar el flujo de trabajo: al hacer clic con el botón derecho en los elementos individuales del flujo de trabajo o en el lienzo blanco, aparecerá un menú emergente con las opciones de edición disponibles.

Flujo de trabajo: Una ejecución de una sola instancia de flujo de trabajo. Esta información incluye qué datos de entrada se proporcionaron.

GroupTask / Tarea de grupo: una composición recursiva de tareas, es decir, una tarea que contiene otras tareas

Motor de flujo de trabajo: el motor de flujo de trabajo proporciona el entorno de ejecución en tiempo de ejecución para una instancia de proceso. Gestiona los datos que se pasan en la instancia de proceso, ejecuta y gestiona las distintas actividades del proceso y gestiona también la ruta seguida a través del proceso evaluando las transiciones entre las actividades que existen en el proceso.

Procedencia: es un historial o una traza de (en este caso) una ejecución de flujo de trabajo. Los datos de procedencia del flujo de trabajo le permiten averiguar qué flujos de trabajo se han ejecutado, con qué datos y cuáles fueron los resultados intermedios y finales.

Puerto: es un conector, desde una entrada o salida de un servicio. Normalmente, un puerto de entrada es una entrada de datos o un parámetro de parámetro de entrada.

Renderizador: es un complemento que controla cómo mostrar los datos en estos formatos especializados. Por ejemplo, Taverna puede mostrar imágenes en 3D Jmol de estructuras de proteínas, imágenes y árboles XML.

Servicio: una instancia de una descripción de servicio dentro de un flujo de trabajo. Servicios web: utilizan principalmente las especificaciones XML, SOAP y WSDL y permiten que los almacenes de datos distribuidos y las herramientas de análisis se puedan acceder y utilizar desde las computadoras de escritorio de los científicos.

Sucesos: los sucesos proporcionan un mecanismo para que las partes débilmente acopladas de la aplicación comuniquen información sobre los cambios de estado del sistema.

Tarea: un objeto que hace algo, tiene entradas y salidas, por ejemplo, Grapher, FFT.

Task Graph / Gráfico de tareas: una definición de proceso de flujo de trabajo: define las tareas que deben ejecutarse y el orden en que se ejecutan.

WSDL: Significa el lenguaje de descripción de servicios web. Es un formato XML que es la interfaz de un servicio web. Es la descripción legible por máquina de las operaciones (o funciones) ofrecidas por el servicio.

8. Anexos

Tabla 1. Componentes de un flujo de trabajo kepler

Componentes	
Director	Controla la ejecución del flujo de trabajo basándose en un modelo de computación concreto, y determina en que modo los “actores” se comunican entre sí.
Actor	Es el elemento fundamental de un flujo de trabajo. El actor es una unidad computacional que tiene puertos de entrada a través de los cuales se alimenta de datos, un núcleo de procesamiento que opera con los datos, y puertos de salida, que ofrece los datos transformados a un nuevo actor.
Actor compuesto	Es un conjunto de actores unidos en un subflujo de trabajo que se representa como un solo actor pero computa operaciones complejas.
Receptores (receivers)	Median en la comunicación entre actores, y son proporcionados por el director.
Parámetros	Son valores configurables que pueden ser asignados a un flujo de trabajo, a un director o a un actor.
Relaciones	Permiten ramificar flujos de datos, para enviar los mismos datos a distintos lugares del flujo de trabajo.
Puertos	Conectan los actores entre sí, para la entrada y salida de datos. Los puertos pueden ser de entrada, salida, o entrada/salida. Además, cada puerto puede conectarse a un solo puerto de otro actor, o a varios puertos de varios actores (multipuerto).
Canal	Es el vínculo que representa el flujo de datos entre puertos de actores distintos.
Paquetes de datos (tokens)	Los canales transportan los datos encapsulados como “tokens”. Cada token tiene asignado un tipo de datos concreto.

Se describen los componentes de un flujo de trabajo científico Kepler
Cuenca, 2019

Tabla 2. Características de los directores

Director	Características
Director PN (Process Networks)	Los actores funcionan como procesos independientes.
	El volumen de escritura de datos en un canal no está limitado, la lectura de datos si puede estarlo.
	No pre calcula el plan de trabajo de los actores.
	Tienen muy pocas restricciones, pero pueden se muy ineficientes
Director SDF (Synchronous Data-Flow)	Se utiliza para redes de procesos especializadas Permite análisis estáticos en flujos de datos.
	Eficiente para sistemas de flujo de datos constantes y conocidos.
	Controla el número de veces que el flujo es iterado.
Director DE (Discrete Event systems)	Diseñado para la simulación y modelado de sistemas dependientes del tiempo.
	Los actores se comunican a través de secuencias de eventos situados en una línea de tiempo real.
Director CT (Continuous-Time models):	Diseñado para modelar sistemas dinámicos gobernados por sistemas de ecuaciones diferenciales lineales y no lineales.
Director DDF	Ejecuta el flujo en un solo hilo, como el SDF
	No pre calcula el plan de ejecución.
	Es útil para diseñar flujos con estructuras de control que no requieren procesamiento en paralelo

Se describen las características de los directores de un flujo de trabajo Kepler
Cuenca, 2019

Tabla 3. Características de los sistemas de flujo de trabajo científico

Características	Triana	Kepler	Taverna
Desarrolladores	Universidad de Cardiff	Kepler / CORE UC Davis financiado por la NSF, UC Santa Barbara y UC San Diego	MyGrid Team Universidad de Manchester, Reino Unido
Versión	4.0	1.0.0	2.1
Plataformas	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X
Lenguaje de desarrollo	Java	Java	Java
Idioma workflow		MoML (basado en XML)	Scufl
Licencia	Apache licencia de código abierto versión 2	Licencia BSD	LGPL
Sitio web	http://www.trianacode.org/	Http://kepler-project.org/	Http://www.taverna.org.uk
Dominio de aplicación	Bioinformática	Física, Ecosistemas, Bioinformática	Biología, Bioinformática, Quimioinformática Astronomía, Ciencias Sociales y Música

Se describen las características de los sistemas Triana, Kepler y Taverna
Cuenca, 2019

Tabla 4. Comparación de los flujos de trabajo científico Kepler, Triana y Taverna

Herramienta	Integración estratégica		Proceso de integración			Reutilización			
	Estrategia de integración ambiental	**Estrategia de integración de terceros**	**Modelo de datos**	**Método de integración**	**Lenguaje de programación**	**Método de composición**	**Soporte de procedencia.**	**Dominio específico o genérico**	**Herramientas de soporte y documentación**
Kepler	Whitebox	Plena integración Encapsulación.	Interfaz base extensible (Actor, Parámetro).	Interfaces de programación orientadas al actor. Soporte para servicios web, servicios OGC, Cloud. Servicios basados en WSDL o servicios en BioCatalogo	Java	Interfaz base extensible (puerto, canal) Directores (flujo de control). Reutilización por actores compuestos o sub flujos de trabajo. Enlaces de datos entre procesadores.	Bien, la procedencia de los datos como parte del flujo de trabajo. Muy bien, bien conectado a mi Experimento.	Foco genérico pero fuerte en el procesamiento molecular y la biología.	Código fuente y binario disponible en el sitio web dedicado. Herramientas front-end disponibles. Documentación, tutorial, lista de correo, entre otros. disponible en línea. http://kepler-project.org Código fuente y binario disponible en el sitio web dedicado.
Taverna	Whitebox	Full Integración + Encapsulación.	Un tipo de "cliente de servicio" o "tipo de servicio" (procesadores) .	Procesadores + descripción del documento basado en XML (SCUFL) Soporte para servicios web, servicios OGC, Cloud, Grid.	Java	Descripción del documento basado en XML + restricciones de coordinación (flujo de control). Reutilización por procesadores de tipo de flujo de trabajo anidados.	Centrarse en flujos de trabajo, servicios, entradas, salidas. Los datos deben ser manejados fuera del sistema.	Foco genérico pero fuerte en las ciencias de la vida y la bioinformática.	Herramientas front-end disponibles. Documentación, tutorial, lista de correo, entre otros. disponible en línea. http://www.taverna.org.uk

				Servicios basados en WSDL (repositorios UDDI) o servicios P2P		Enlaces de datos entre unidades.			Código fuente disponible desde el servidor de subversión - SVN. Código binario en forma de sitio web dedicado.
Triana	Whitebox	Integración completa + encapsulación.	Un tipo de "cliente de servicio" o "tipo de servicio" (Unidades).	Soporte para servicios web, P2P, Grid, WS-RF Recurso de Servicios web estándar	Java	Control del flujo mediante caja de herramientas incorporada. Reutilizado por la publicación de flujos de trabajo como WS (web services) en los repositorios UDDI (Descripción, descubrimiento e integración universales)	Bien, la procedencia de los datos como parte del flujo de trabajo.	Minería de datos, estadísticas.	Herramientas front-end disponibles. Mala documentación y tutoriales. http://www.trianacode.org

Tabla 5. Comparación de los SWFC Kepler, Triana y Taverna en base a su arquitectura

Requerimientos	Kepler	Triana	Taverna
Personalización de la interfaz de usuario y soporte de interacción del usuario.	Parcial	Parcial	Parcial
Soporte de reproducibilidad.	Completo	Completo	Completo
Integración de servicios heterogéneos y distribuidos y herramientas de software.	Parcial	Parcial	Parcial
Gestión de productos de datos heterogéneos y distribuidos.	Parcial	No	No
Soporte informático de gama alta.	Parcial	Parcial	Parcial
Monitoreo de flujo de trabajo y manejo de fallas.	Parcial	Parcial	Parcial
Interoperabilidad.	No	No	No

Se describe el análisis de Kepler, Triana y Taverna en base a los requisitos arquitectónicos.

Cuenca, 2019

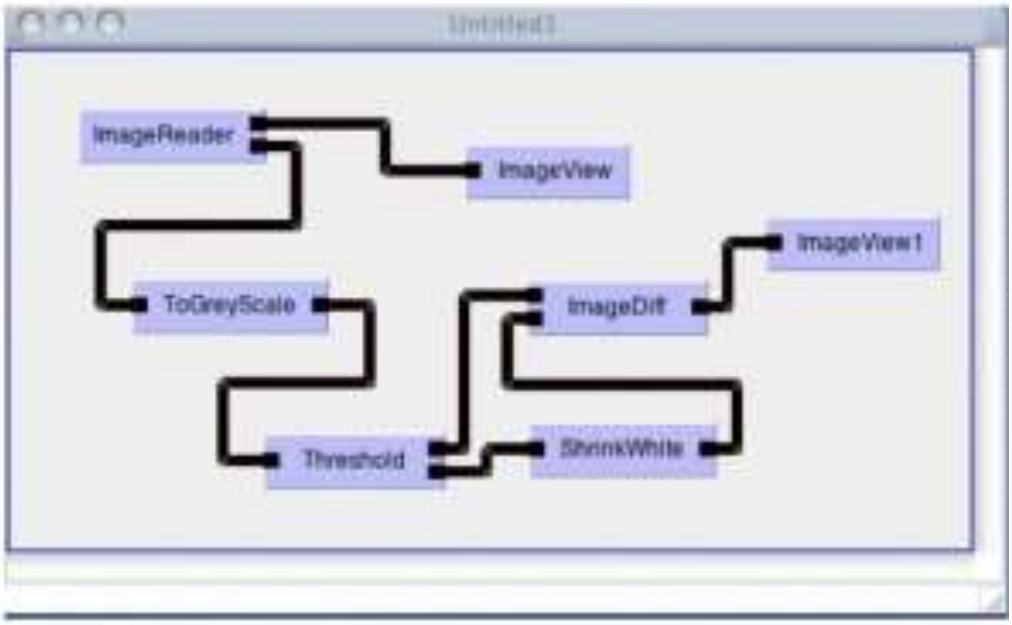

Figura 1. Triana work Flow

(Curcin, 2013)

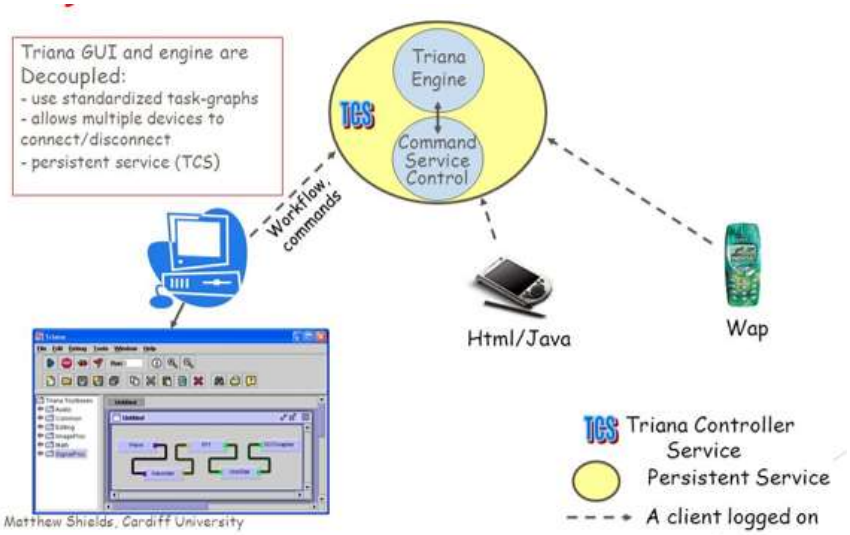

Figura 2. Triana controller service

(Shields, 2017)

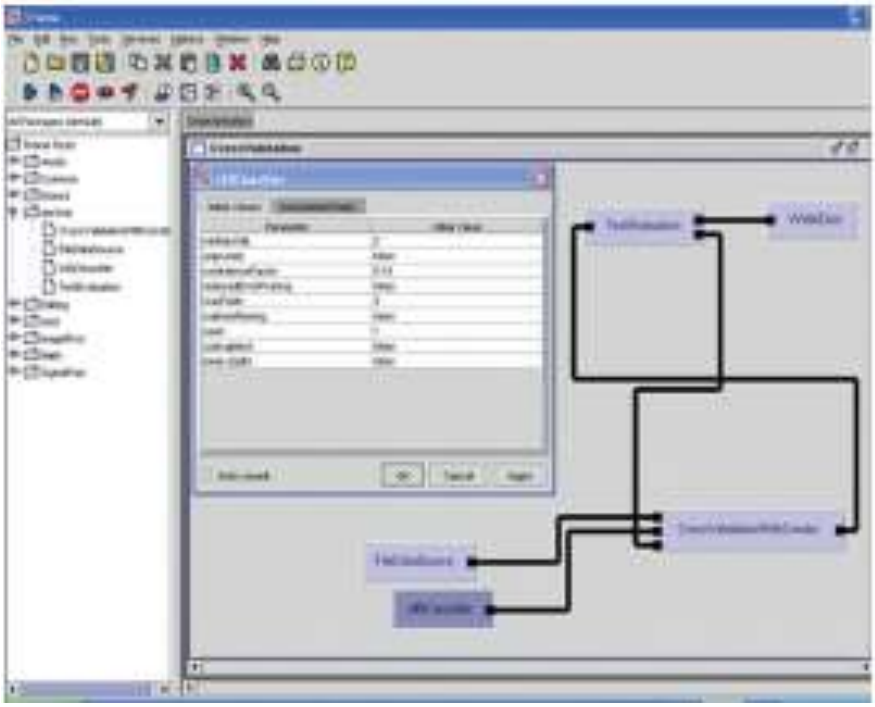

Figura 3. Una instantánea de la GUI de Triana utilizada para componer datos

(Talia, 2013)

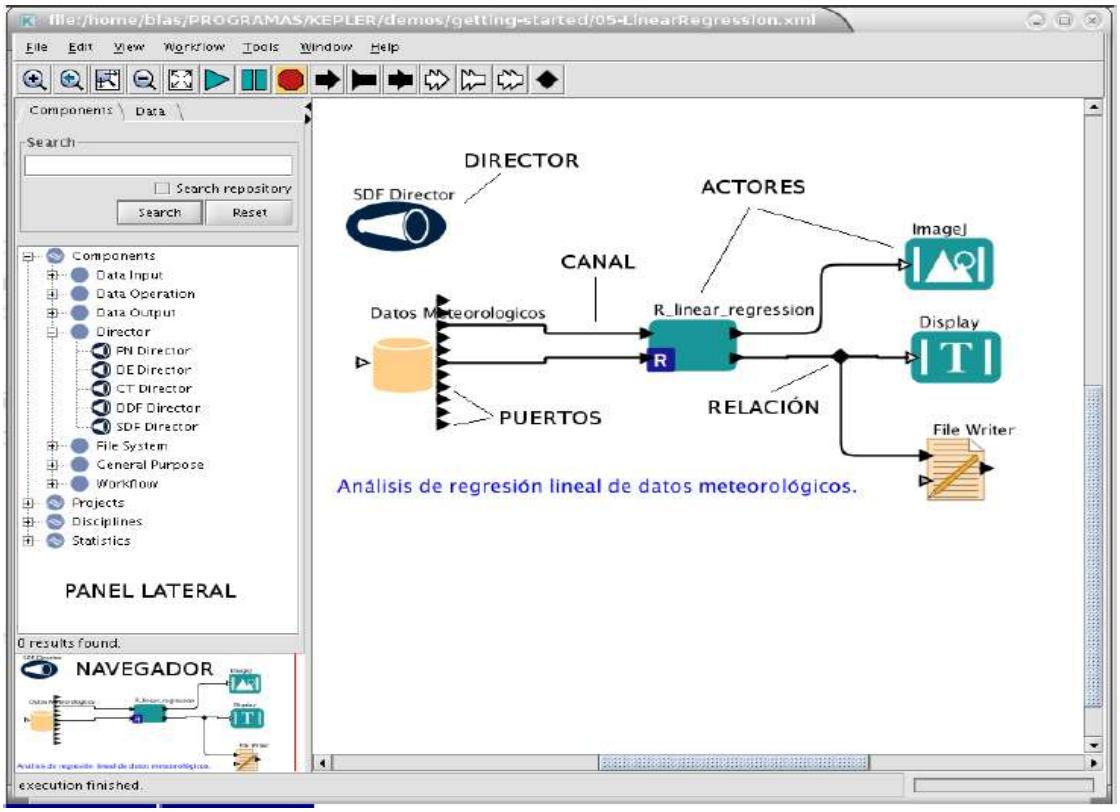

Figura 4. Componentes de Kepler

(Kepler, 2016)

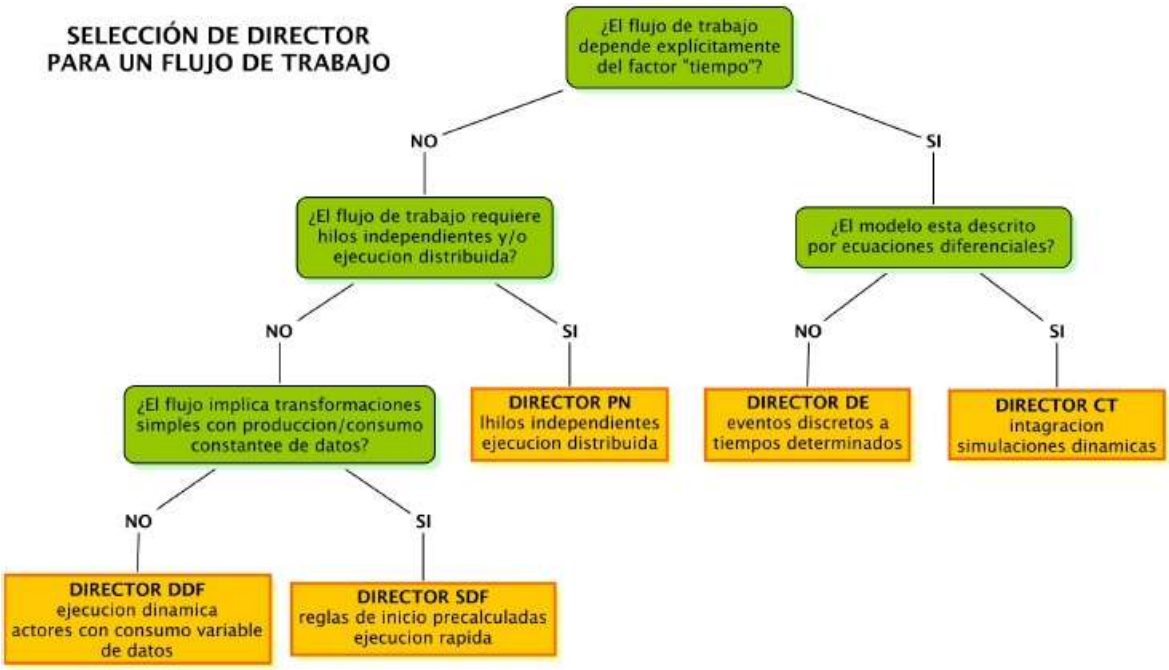

Figura 5. Selección de director para un flujo de trabajo en Kepler

(BPM Center report, 2008)

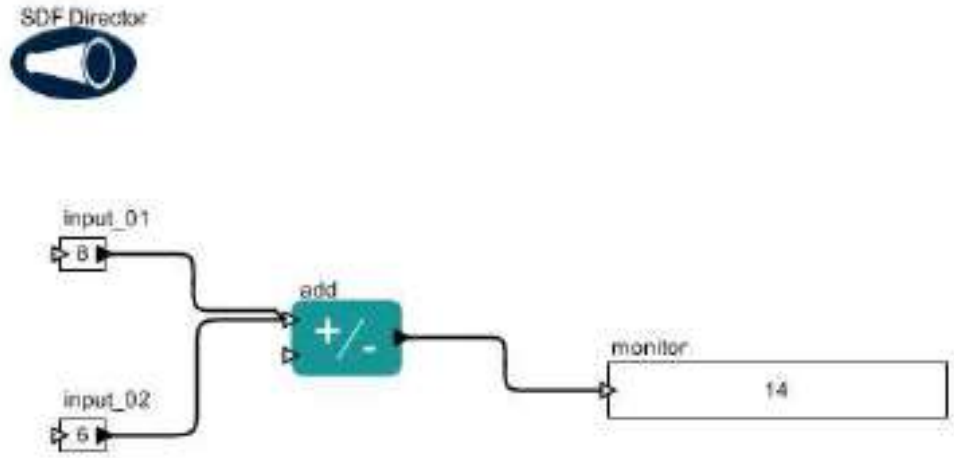

Figura 6. Patrón de secuencia Wcp-01 en Kepler

(BPM Center report, 2008)

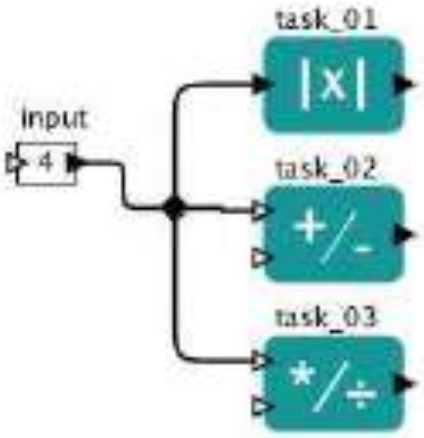

Figura 7. Ejemplo de Wcp-02 Parallel Split implementado en Kepler

BPM Center report, 2008

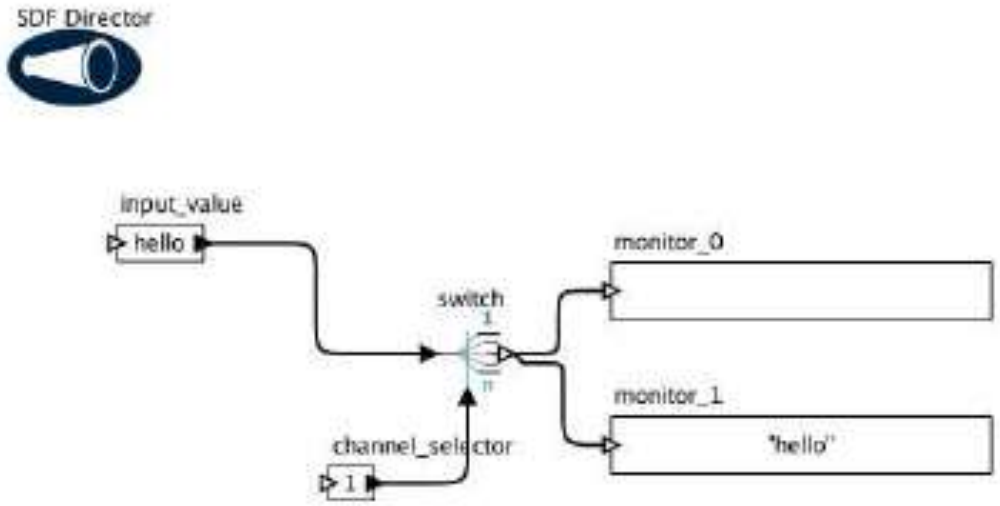

Figura 8. Ejemplo de elección exclusiva Wcp-04 implementada en Kepler

BPM Center report, 2008

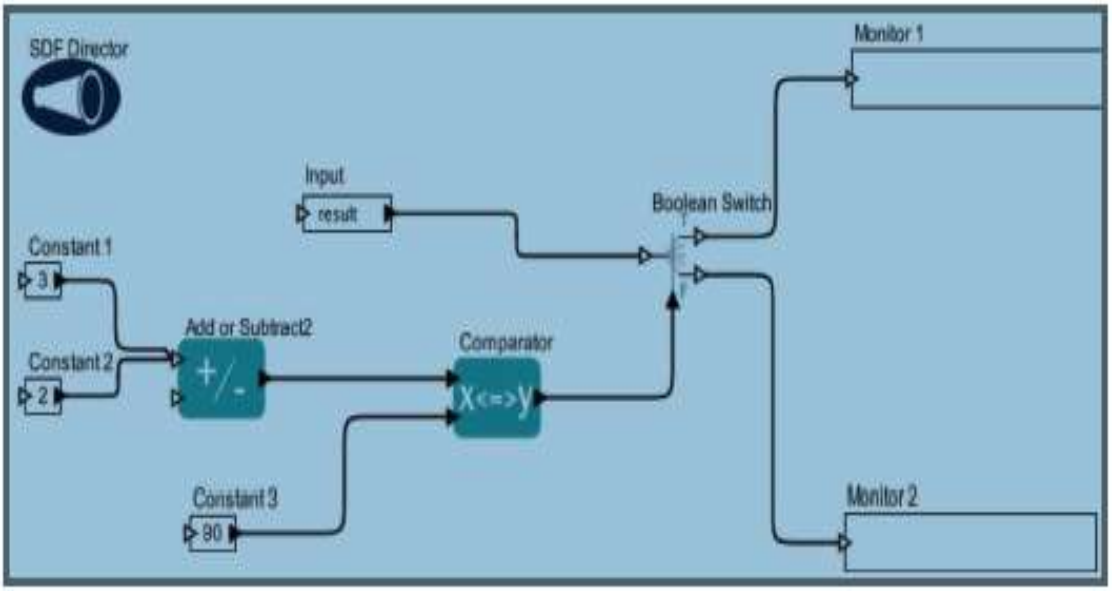

Figura 9. Ejemplo de Wcp-04 Exclusive Choice implementado en Kepler (BPM Center report, 2008)

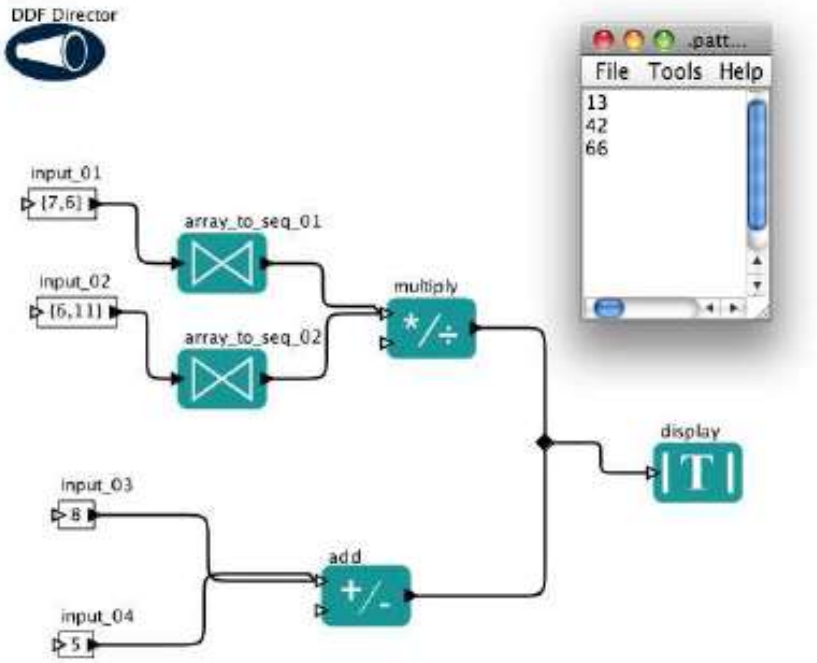

Figura 10. Ejemplo de Wcp-05 Simple Merge implementado en Kepler (BPM Center report, 2008)

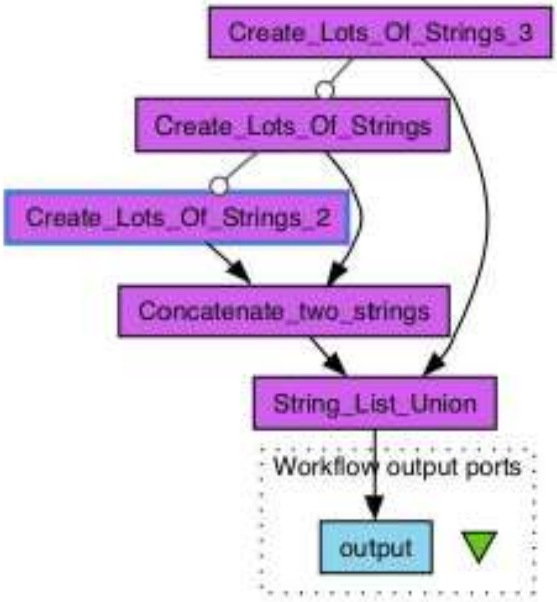

Figura 11. Patrón de secuencia Wcp-01 en Taverna
(BPM Center report, 2008)

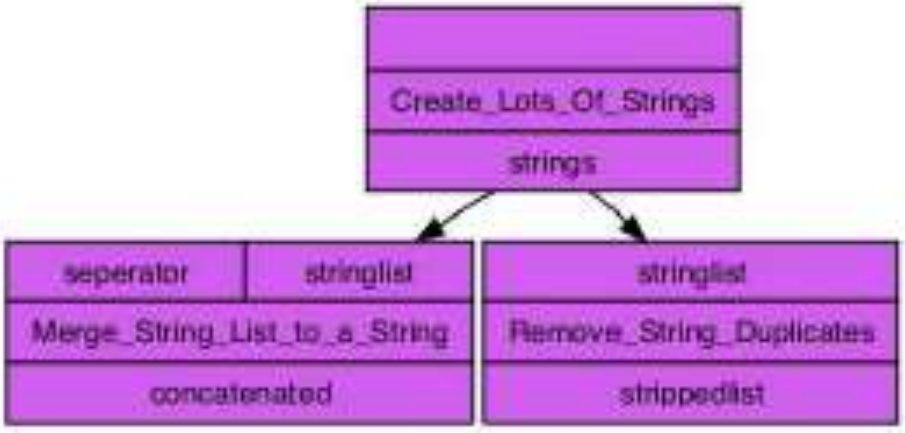

Figura 12. Ejemplo de Wcp-02 Parallel Split implementado en Taverna
(BPM Center report, 2008)

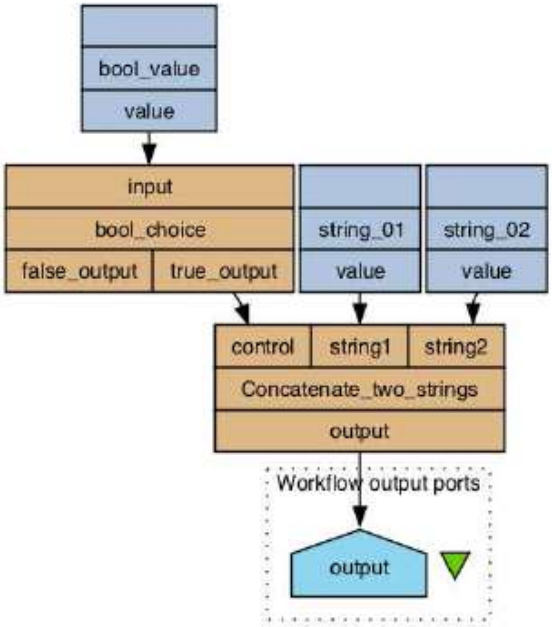

Figura 13. Ejemplo de Wcp-04 Exclusive Choice implementado en Taverna

BPM Center report, 2008

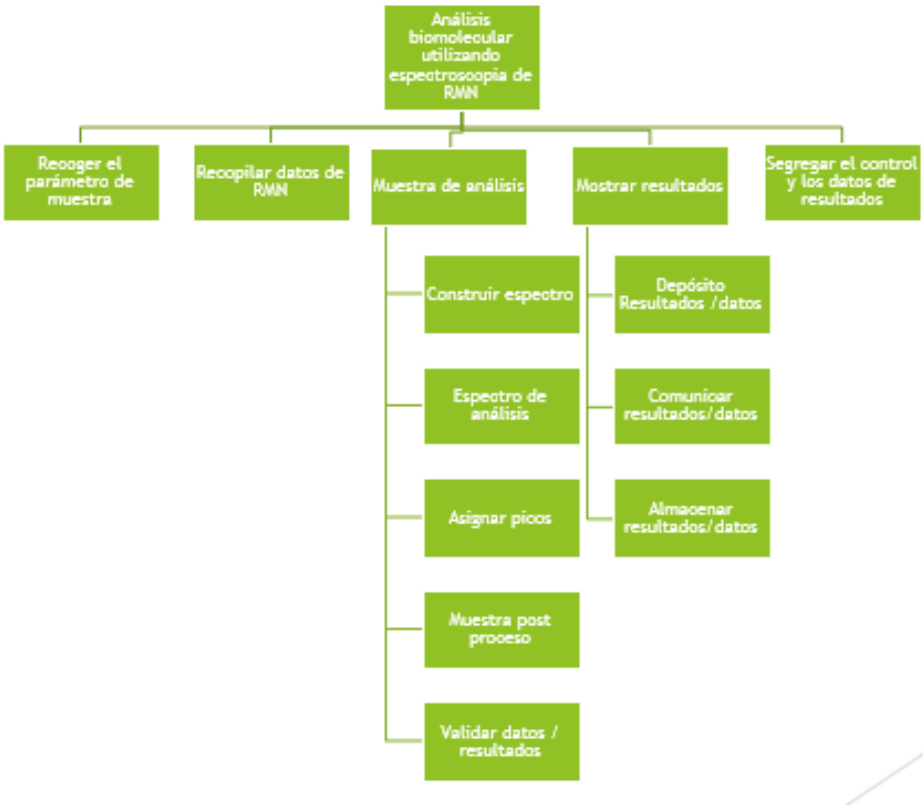

Figura 14. Modelo de contexto para el proceso de análisis biomolecular mediante Resonancia magnética nuclear

Cuenca, 2019

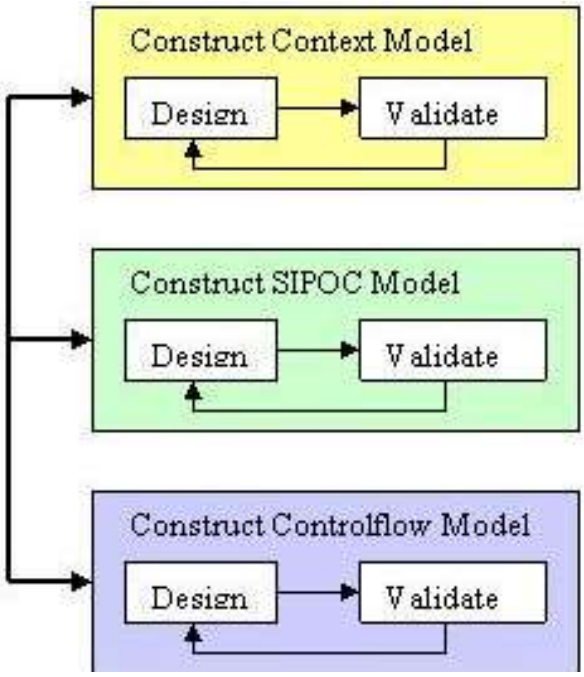

Figura 15. Modelo de contexto para el proceso de análisis biomolecular

(Verdi, Ellis, y Gayk, 2015)

X

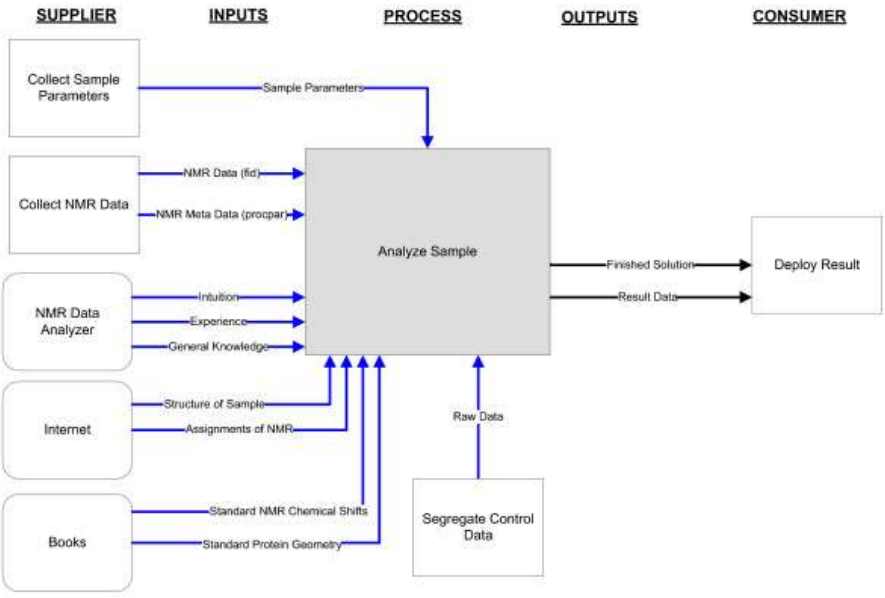

Figura 16. Modelo SIPOC para analizar la muestra Proceso del nivel superior del proceso de análisis biomolecular mediante RMN

(Verdi, Ellis, y Gayk, 2015)

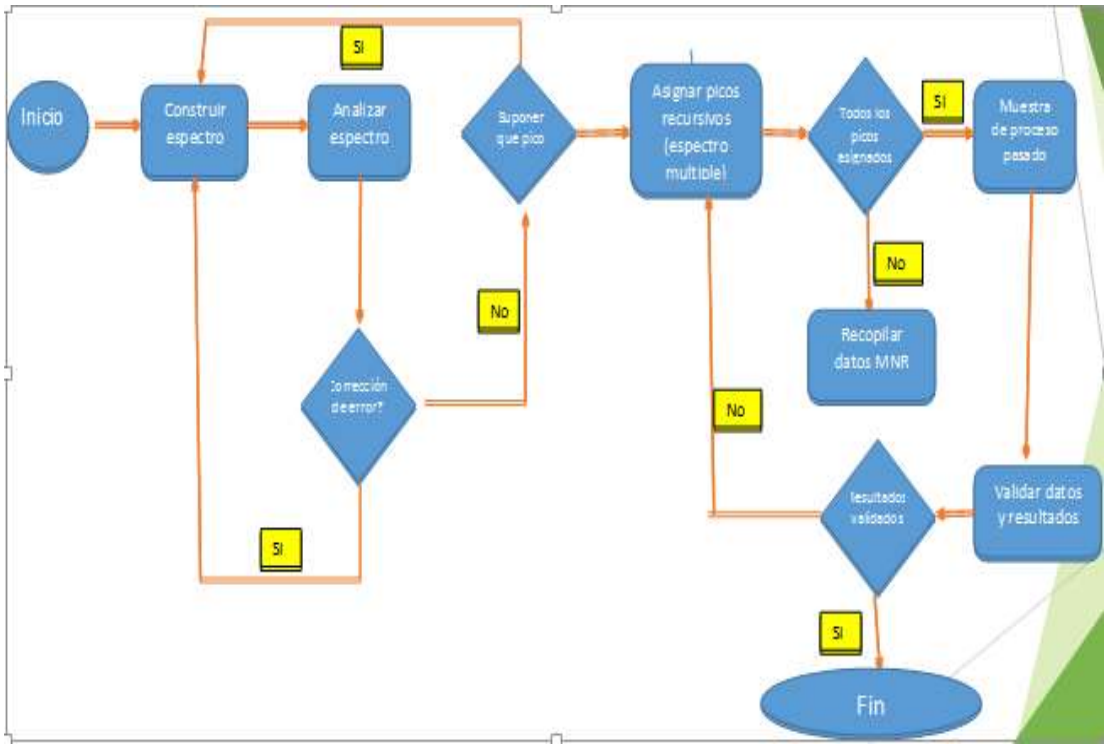

Figura 17. Modelo de flujo de control para el proceso de muestra de análisis del proceso de análisis biomolecular mediante RMN

Verdi, Ellis, y Gayk, 2015