ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

Propuesta de modelado de línea de producto de software para aplicaciones mashup

Software product line modeling proposal for mashup applications

Héctor Reinaga, Juan Enriquez, Sandra Isabel Casas
hreinaga@uarg.unpa.edu.ar, jenriquez@unpa.edu.ar, sicasas@uarg.unpa.edu.ar
GISP, Instituto de Tecnología Aplicada
Unidad Académica Río Gallegos, Universidad Nacional de la Patagonia Austral
Río Gallegos - Santa Cruz - Argentina

Recibido: 30/04/2021. Aceptado: 25/10/2021

RESUMEN

Un mashup es una aplicación compuesta que integra dos o más tipos de componentes disponibles en la Web, creando un nuevo valor a partir de los componentes o artefactos que la componen, permitiendo su reuso y proporcionando una funcionalidad que no existía antes. Las actuales herramientas y enfoques de desarrollo de estas aplicaciones carecen de algún modelo para integrar componentes similares. Las Líneas de Productos de Software es un enfoque de desarrollo de software cuyo principal objetivo es la reusabilidad, permitiendo crear una familia de productos donde cada producto posee características comunes, y difiere de otro en un conjunto de funcionalidades. En este trabajo se propone modelar una aplicación Mashup desde un enfoque de variabilidad, lo cual permitirá más adelante implementar una línea de productos de software. Para ello, se toma referencia una herramienta de desarrollo basado en Pipes, y se especifica a partir de un modelo de características. Finalmente se generan diversos ejemplos.

Palabras clave: Mashup; Línea de Producto de Software; Desarrollo de Software Orientada a Feature; Modelo de Características; Pipes.

ABSTRACT

A Mashup is a composite application that integrates two or more types of components available on the Web, creating a new value from the components or artifacts that compose it, allowing their reuse and providing functionality that did not exist before. The current tools and approaches to develope these applications do not have any model to integrate similar components. Software Product Lines (SPL) is a software development approach whose main objective is reusability, allowing the creation of a family of products where each product has common characteristics, and differs from another in a set of functionalities. This project work proposed to model a Mashup application from a variability approach, which will later allow to implement a SPL. For this, a development tool based on Pipes is referenced, and it is specified from a characteristic model, finally various examples are generated.

Keywords: Mashup; Software Product Line; Feature Oriented Software Development; Features Model; Pipes.

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

1. INTRODUCCIÓN

Actualmente la Web ha dejado de ser un medio de comunicación en un solo sentido como era en sus comienzos. Hoy existe un enorme ecosistema de aplicaciones en ejecución para diversos dispositivos, y algunas de ellas permiten la interacción con otras aplicaciones (Daniel y Matera, 2014). Con esto se ha marcado una nueva tendencia en la Web, la cual consiste en reutilizar componentes de diversas aplicaciones para formar nuevas aplicaciones que ofrezcan un valor agregado. Estos componentes o bloques, han hecho posible que aparezcan las aplicaciones denominadas Mashups, que se caracterizan por la integración de componentes de datos, lógica o interfaz de usuario. Esta nueva forma de desarrollo permite llevar a cabo la implementación de aplicaciones web nuevas y además posibilita la incorporación de contenidos actualizados en sitios web ya desarrollados. Técnicamente un Mashup se define como una aplicación compuesta que integra dos o más tipos de componentes disponibles en la Web (Daniel et al., 2012). Un componente es cualquier segmento de datos, lógica de aplicación y/o interfaz de usuario que puede ser reutilizada y que es accesible ya sea local o remotamente (Daniel y Matera, 2014). Entre las tecnologías de componentes más importantes se encuentran, los servicios web SOAP/WSDL y RESTful, feeds RSS/Atom, Servicios Web, contenidos wrapper en sitios web de terceros, API programables (como Google Maps) o fuentes XML. La mayoría de estos componentes se basan en lenguajes estándar, tecnologías o protocolos de comunicación.

La Redifusión Web (en inglés, Web Syndication) proporciona canales de suministro de noticias para publicar un resumen de los contenidos añadidos recientemente y los últimos cambios de los sitios web, así como las últimas noticias o publicaciones en foros, permitiendo a los usuarios mantenerse informados sin tener que navegar por distintos sitios web (Sikos, 2011). Aquí surge el concepto de feeds (fuente, canal), que se define como documentos utilizados para transferir las actualizaciones de los contenidos digitales a los usuarios (Yee, 2008). Los dos formatos más conocidos de Redifusión web son RSS y Atom. RSS corresponde al acrónimo Really Simple Syndication, es un dialecto del lenguaje XML. Por otro lado, Atom fue creado como alternativa a RSS, propone un estándar y soluciona los problemas de compatibilidad que presenta RSS. Se considera Atom como una alternativa mejor para la redifusión web, pero el formato RSS está mucho más extendido.

Se considera las fuentes RSS como un componente Mashup entre otras tecnologías (Tinajero Díaz, 2016), y en consecuencia, no escapa a las dificultades que se encuentran en el proceso de desarrollo de los mismos, debido al volumen y heterogeneidad de componentes mashup disponibles en la web; ausencia de algún modelo para integrar componentes similares; selección de los modelos adecuados para la integración: exploración y comprensión.

A partir de un enfoque de variabilidad en el contexto de una Línea de Productos de Software, que permita tener en cuenta el análisis sobre la heterogeneidad de los componentes en la Web, y los requerimientos de composición para integrar aplicaciones Mashup; puede proporcionar nuevas opciones de modelado, diseño e implementación para componer e integrar este tipo aplicaciones. En esta línea, se propone modelar una Línea de Productos de Software que permita la composición e integración de aplicaciones Mashup a partir de un Modelo de Características.

Una Línea de Productos de Software (LPS) es una familia de sistemas (o productos) relacionados a un dominio en particular, cuyos artefactos de implementación son compartidos (Capilla et al., 2014) (Navarro Favela y Juárez Martínez, 2013). Los productos de una misma LPS poseen un conjunto de características en común, denominado núcleo, pero cada producto difiere de otro en un conjunto de funcionalidades opcionales (variables) que implementa (Capilla et al., 2013). Esta diferencia funcional entre productos de una LPS se conoce como variabilidad (Metzger y Pohl, 2014). Para expresar características comunes se crean Modelos

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

de Características (MC) y para manejar la variabilidad entre los productos de una LPS, Modelos de Variabilidad (Acher et al., 2013) (Galindo et al., 2014).
La inexistencia de un modelo genérico para aplicaciones en este dominio es motivo por la cual se plantean las siguientes preguntas: ¿Qué herramientas existen para aplicaciones de Mashup Feeds? ¿Qué aplicaciones se pueden emplear para el modelado de características para componer e integrar un Mashup? ¿Cómo se puede especificar el modelo de características propuesto? En el caso de responder en forma afirmativa se replantea las siguientes preguntas: ¿Cómo construir aplicaciones de Mashup Feeds a partir del conjunto de funcionalidades comunes y variables?; ¿De qué manera reutilizar las características en otros productos?
En consecuencia, teniendo en cuenta una tecnología disponible y fundamental para desarrollar un Mashup como son los feeds, en este trabajo se procede a la identificación y el análisis de las funcionalidades del entorno de desarrollo visual Web, Pipes Digital; diseñar el modelo de características en base a las funciones analizadas; reflejar en el modelo las características y la variabilidad que correspondan; y por último obtener una configuración de los productos; buscando mayor flexibilidad y planteando nuevas estrategias.
La estructura del presente documento se organiza de la siguiente forma: en la Sección 2, se estudian los conceptos de Mashup, Feeds, y Línea de Producto de Software. En la Sección 3, se presentan las herramientas de desarrollo para Mashup Feeds que fueron analizadas. En la Sección 4, se describe el modelo de características propuesto a partir del cual se desarrolló la LPS, y se presenta diversos casos de estudio. Finalmente, en la última Sección, se presentan las Conclusiones y los trabajos futuros.

2. NOCIONES PRELIMINARES

En esta Sección se introducirán conceptos de Mashup, Feeds RSS-Atom; Línea de Producto de Software, su activo más importante, los modelos de características.

2.1. Mashup

La forma de acceder a la información y el formato en el cual están disponibles a través de la web ha cambiado. En consecuencia las aplicaciones web y su proceso de desarrollo también se han modificado. Las aplicaciones actuales son el resultado de la integración de múltiples recursos disponibles en la web, tales como APIs, contenidos y componentes (Metzger y Pohl, 2014). Por esta razón el proceso de desarrollo también se ha modificado. Esto ha dado lugar al surgimiento de nuevas prácticas como Mashup, que utiliza la web como plataforma de desarrollo y además combina técnicas ya establecidas como incrustación, agregación e integración para generar nuevas aplicaciones (Matera y Picozzi, 2015).
Un mashup (o Web mashup) es una aplicación que integra dos o más componentes mashup en cualquier capa de la aplicación, por ejemplo, en la capa de datos, lógica de aplicación y presentación (Daniel y Matera, 2014). Un Componente Mashup es cualquier segmento de datos, lógica de aplicación y/o interfaz de usuario que puede ser reutilizada y que sea accesible de manera local o remota. Entre las tecnologías de componentes más importantes se encuentran, los servicios web SOAP/WSDL y RESTful, fuentes RSS/Atom, servicios web, contenidos wrapper en sitios web de terceros o API programables (por ejemplo, Google Maps) o fuentes XML. La mayoría de estos componentes se basan en lenguajes estándar, tecnologías o protocolos de comunicación.
La adopción del desarrollo de mashups como un nuevo esquema para resolver problemas que involucran la integración de datos de diversas fuentes heterogéneas, dio como resultado una gran cantidad de enfoques para crear nuevas herramientas integradas que pueden cumplir con estos propósitos.

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

2.1.1. Clasificación

Existen distintas clasificaciones de mashups que se encuentran en la literatura. En muchos casos, los mashups se clasifican según su funcionalidad; por ejemplo, mashups de datos, mashups de fotos y videos, mashups de noticias y mashups comerciales (Sheong, 2008). Por otro lado, en (Mandal et al., 2019) los mashups se pueden clasificar de acuerdo a su nivel de complejidad (básico, intermedio, y avanzado), y base de integración (presentación, datos, y proceso).
Una clasificación más general y que engloba varias características, se encuentran en (Trinh, 2016) (Daniel y Matera, 2014), donde se propone una clasificación mashup basada en cubos (Figura 1). Se centra en tres perspectivas: composición, dominio y entorno.
1. La perspectiva de composición clasifica los mashups según la capa donde se realiza la integración de mashup; en la arquitectura típica de tres niveles, las aplicaciones se dividen y desarrollan en tres capas separadas, es decir, una capa de datos, una capa lógica y una capa de presentación [48]. Este modelo de desarrollo facilita la implementación y el mantenimiento de diferentes partes de las aplicaciones, ya que estas partes son independientes pero capaces de interoperabilidad. Los posibles tipos de mashups en esta perspectiva son: mashups de datos, mashups lógicos, mashups de interfaz de usuario y mashups híbridos.
2. La perspectiva de dominio clasifica los mashups de acuerdo a las funcionalidades que proporcionan los mashups. Se encuentran varios dominios de categoría de mashups[1], como: mapeo, social, herramientas, móvil, fotos, financiero, etc.
3. La perspectiva del entorno clasifica los mashups según su contexto de implementación. Los mashups web y los mashups empresariales se identifican como los dos tipos de mashups en esta perspectiva. Los mashups web (que también se conocen como mashups de consumidores) son para usuarios de Internet. Se centran en las funcionalidades más que en los requisitos no funcionales (p.e. seguridad, cumplimiento de las leyes y normativas) impuestos en el entorno empresarial.

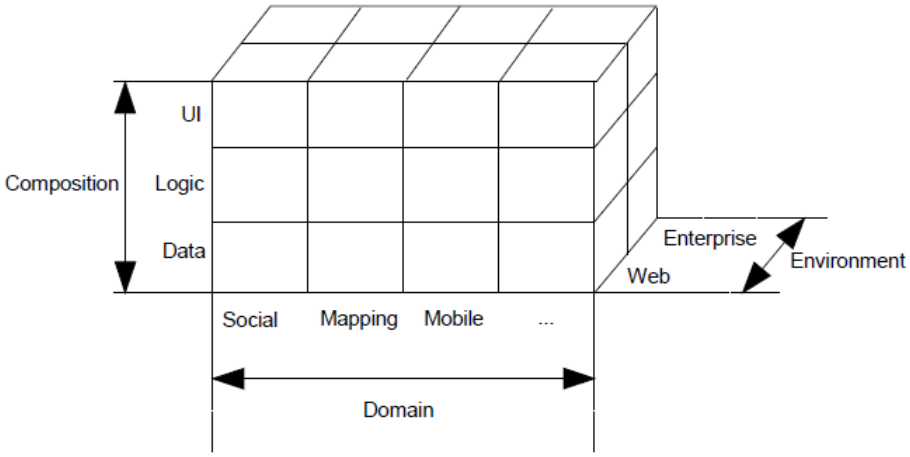

Figura 1. Clasificación de mashup por dimensiones.

2.1.2. Arquitectura de los Mashups

Existen diversas arquitecturas de software que se pueden utilizar en la construcción de aplicaciones mashup: basadas en servidor, basadas en cliente y móviles (Díaz et al., 2012).
Aunque se podría dividir una aplicación mashup en varias capas de software para un análisis detallado, generalmente se compone de tres capas principales:

[1] http://www.programmableweb.com/category

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

- **Proveedor de contenidos o capa de origen datos**: las API (Application Programming Interface); son los proveedores de contenido de los cuales se desea combinar sus datos para crear el mashup. Los proveedores brindan acceso a sus contenidos, con el interés de facilitar la recuperación de datos, a través de protocolos de Internet como REST, SOAP, servicios Web o archivos RSS.
- **El sitio de alojamiento o capa del mashup**: corresponde al lugar de alojamiento del mashup. Los mashups pueden utilizar un servidor de contenido dinámico para la generación de información de lado del servidor, aplicando tecnologías como Java, PHP o ASP, entre otras. O a su vez, si se utiliza codificación en JavaScript o utilizando applets, el contenido se lo generaría concretamente en el navegador del cliente a través de la combinación de código incrustado directamente en las vistas de la página web del mashup. Por lo general las APIs públicas tienen librerías de datos escritas en lenguaje JavaScript.
- **El navegador Web del cliente o la capa de presentación del mashup**: en el navegador o browser, es el lugar donde toma forma la interacción con el usuario y donde se genera visualmente su contenido.

2.2. Feeds

El aumento de información disponible en sitios Web es un problema por la dificultad de mantenerse actualizado debido al volumen de información nueva generada diariamente. En ese escenario surgen servicios capaces de notificar automáticamente a los usuarios sobre nuevos contenidos y actualizaciones de sitios previamente seleccionados; así surge el término redifusión (syndication, en inglés).
La Redifusión Web proporciona canales de suministro de noticias para publicar un resumen de los contenidos añadidos recientemente y los últimos cambios de los sitios web, así como las últimas noticias o publicaciones en foros, permitiendo a los usuarios mantenerse informados sin tener que navegar por distintos sitios web (Sikos, 2011).
Aquí surge el concepto de feeds (fuente, canal), que se define como documentos utilizados para transferir las actualizaciones de los contenidos digitales a los usuarios (Yee, 2008). Los dos formatos más conocidos de redifusión web son RSS y Atom.

2.2.1. RSS

RSS es un formato de distribución de contenido Web. Su nombre es un acrónimo de Really Simple Syndication. En (Singh y Sahu, 2015) lo define como una familia de formatos de fuentes web que se utilizan para publicar contenido actualizado con frecuencia de un sitio web, haciendo que las personas se mantengan al día con los sitios de interés de una manera automatizada. Asimismo, RSS resuelve el problema de la recopilación y distribución de la información, debido al aumento de tráfico y volumen de la información disponible en la Web.
Todos los archivos RSS deben cumplir con las especificaciones de XML 1.0, como se publica en el World Wide Web Consortium (W3C).
Para hacer uso de un feed RSS es necesario un software para la agregación y lectura de contenidos de interés (O'Shea, 2016). La agregación RSS es un proceso mediante el cual el software conocido como agregadores, recopila una gran cantidad de feeds RSS, blogs u otros tipos de contenido digitales, y los agrega en uno o más feeds para el consumo de los usuarios a través de los readers o lectores. La agregación puede basarse en el filtrado de palabras clave, las preferencias del usuario (ya sea definidas por el usuario o recopiladas según el uso) u otras técnicas de clasificación/extracción de texto. Los lectores RSS, que pueden ser gratuitos, pagos o basados en membresía, permiten a los usuarios registrarse o suscribirse a fuentes RSS

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

individuales o agregaciones de las mismas, u otro contenido en línea. Este proceso elimina al usuario la inconveniencia de rastrear manualmente numerosos sitios web en busca de material de interés.
En la Tabla 1 se detallan los elementos más importantes de un documento RSS 2.0[2].

ELEMENTOS BÁSICOS RSS 2.0				
<rss version ="2.0	Este *tag* indica el comienzo de un documento RSS versión 2.0.			
	<channel>	El elemento canal contiene metadatos que describen el canal en sí. Debe existir un único elemento <*channel*> en el documento.		
		<title>	Descripción del titulo del *feed*.	
		<link>	URL de la página a la que corresponde el canal.	
		<description>	Frase o resumen que describe el canal.	
		<item>	Representa el contenido del *feed*. Un canal puede contener *n* elementos *item*.	
			<title>	Título que describe al *item*.
			<link>	URL al contenido del *item*.
			<description>	Descripción del *ítem*.

Tabla 1. Elementos básicos RSS 2.0.

2.2.2. Atom

Atom es el nombre de un formato de redifusión web y metadatos basado en XML, y un protocolo de nivel de aplicación para publicar y editar recursos web que pertenecen a sitios web actualizados periódicamente (Nottingham y Sayre, 2005).
Atom fue creado como alternativa a RSS, propone un estándar dentro del mundo de la redifusión web, y soluciona los problemas de compatibilidad que presenta RSS. Atom y RSS se diferencian por el formato de fecha utilizado para transferir información.
En la Tabla 2 se explican los elementos más importantes de un documento Atom[3].

ELEMENTOS BÁSICOS ATOM 1.0			
<feed xmlns= "namespace">	El elemento raíz de todo documento *Atom* 1.0 es el elemento *feed* y establece el esquema *Atom* 1.0 como el *namespace* por defecto. El elemento *feed* es el elemento que contiene los datos y metadatos.		
	<title>	Descripción del titulo del *feed*.	
	<link>	URL de la página a la que corresponde el canal.	
	<updated>	Indica la última vez que el *feed* fue actualizado.	
	<author>	Nombre de un autor del *feed*.	
	<id>	Identifica al *feed* utilizando un sistema único y permanente *Uniform Resource Identifiers* (URI[5]).	
	<entry>	Un documento *Atom* no requiere un elemento *feed* para contener un elemento *entry*. Un elemento *entry* puede ser parte de un feed y también puede ser su propio documento. Independiente de ello el formato de los subelementos no varía. Un elemento *entry* es equivalente al elemento *item* analizado en la declaración de los formatos RSS 1.0 y 2.0.	
		<title>	Título de que describe la entrada.
		<link>	URL de la entrada.
		<id>	Identifica al elemento *entry* utilizando un sistema único y permanente URI.
		<updated>	Indica la última vez que la entrada fue actualizada.
		<summary>	Descripción o resumen de la entrada.

Tabla 2. Elementos básicos Atom 1.0.

[2] http://www.rssboard.org/rss-specification
[3] https://www.ietf.org/rfc/rfc4287

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

2.3. Línea de producto de software

Una LPS es un conjunto de sistemas de software que comparten un conjunto común y gestionado de características, que satisfacen las necesidades específicas de un determinado sector del mercado, y que son desarrollados a partir de una serie de elementos base comunes (Northrop y Clements, 2012).
El objetivo principal de una LPS es proporcionar una infraestructura adecuada para una rápida y fácil producción de sistemas de software similares, destinados a un mismo segmento de mercado.
Las LPSs son análogas a las líneas de producción industriales, los productos similares o idénticos se ensamblan y configuran a partir de piezas prefabricadas bien definidas, que son reutilizadas para la construcción de productos con características similares. Un ejemplo clásico es la fabricación de automóviles, se pueden crear decenas de variaciones de un único modelo de auto con sólo un grupo de piezas diseñadas y una fábrica específicamente concebida para configurar y ensamblar dichas piezas.
El proceso de desarrollar una línea completa de productos de software en lugar de una sola aplicación se denomina Ingeniería de Dominio, en cambio la Ingeniería de Producto incluye el desarrollo de los productos para clientes concretos (Kästner y Apel, 2011) (Figura 2).

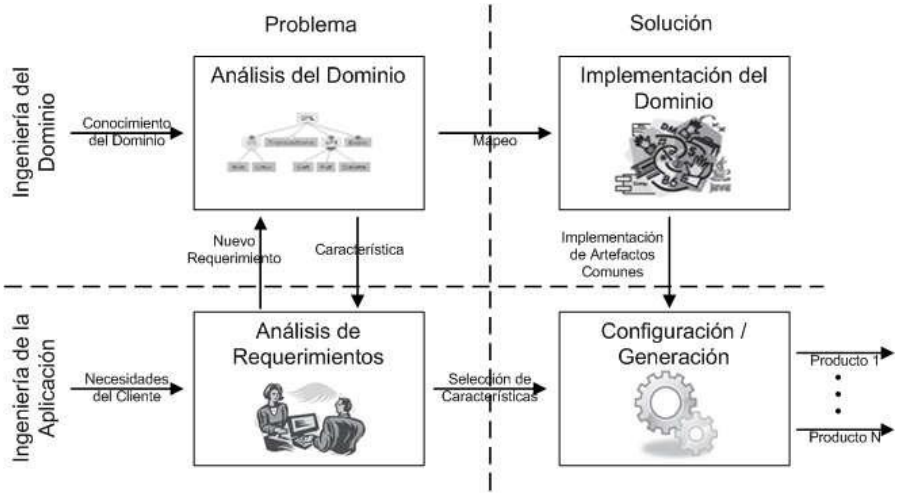

Figura 2. Desarrollo de una LPS: Ing. del dominio-Ing. de la aplicación.

Una LPS realiza inicialmente un estudio para analizar el dominio, determinar las variaciones a soportar, y desarrolla la plataforma de la LPS. El proceso general puede dividirse en dos grandes fases: la construcción del dominio y la construcción de las aplicaciones (o familia de productos).
La construcción del dominio abarca la definición y el aislamiento del marco de trabajo del proyecto. Cualquier característica, la cual se desee que forme parte de la familia de productos, debe formar parte previamente del dominio.
El proceso de construcción de aplicaciones abarca la estructuración de un modelo de aplicación el cual se ajusta a un esquema, o sea, a las características del producto previsto.
El desarrollo de LPS ha traído beneficios como: reducción en los ciclos de desarrollo y costos asociados, incrementos de productividad, y mejoras en la calidad de los productos (Sepúlveda et al., 2012).

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

2.3.1. Modelo de Características

Las LPS se representan por medio de modelos. Una de las notaciones que existen para expresar estos modelos es la notación de los modelos de características (Capilla et al., 2013). Los modelos de características (MC) especifican el conjunto de productos válidos que pueden obtenerse de una LPS (Apel et al., 2013). Estos modelos se diseñan en la primera fase de la línea de productos y por lo tanto juegan un papel central en todas las fases del desarrollo de la LPS.
Una característica (feature) es un rasgo o elemento distintivo percibido por el usuario final que representa aspectos relevantes de un software. Las características se usan para especificar y comunicar aspectos comunes y variables de la LPS (Apel et al., 2013).
Los MC organizan el conjunto de características en una estructura jerárquica en forma de árbol mediante relaciones entre ellas:
a) Relaciones jerárquica (padre-hijo):
Obligatoria (Mandatory): indica que cuando la característica padre es parte de un producto particular, la característica hija también debe ser parte del producto (Figura 3.a).
Opcional (Optional): indica que cuando la característica padre es parte de un producto particular, la característica hija, puede o no, ser incluida en el producto (Figura 3.b).
Alternativa (Alternative): es la relación entre una característica padre y un conjunto de características hijas que indica que, cuando la característica padre es parte de un producto particular, sólo una de las características del grupo de hijas debe ser parte del producto (Figura 3.c).

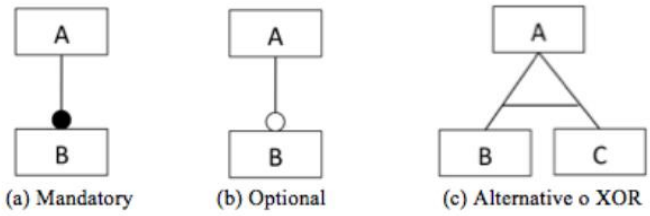

Figura 3. Representación tipos de relaciones padre-hijo.

b) Relaciones no jerárquica (restricciones cross-tree):
Excluye (Excludes): Una característica X que excluye a la característica Y significa que si la característica X es incluida en el producto, la característica Y no debe ser incluida, y viceversa (Figura 4.a).
Requiere (Requires): Una característica X que requiere de una característica Y, significa que si la característica X es incluida en el producto, la característica Y también debe ser incluida, pero no viceversa (Figura 4.b).

Figura 4. Representación tipos de relaciones no jerárquica.

Los MC se utilizan con dos propósitos principales:
1. Representar las variabilidades (requisitos comunes y variables de los productos), organizar, clasificar y representar las dependencias entre las características.
2. Servir de base como árbol de decisión para configurar (seleccionar las características) y derivar la configuración correspondiente a un producto.

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

2.3.2. Herramientas para Modelado de Feature

Existen varias herramientas para el MC, desde prototipos académicos hasta comerciales. La mayoría coinciden en permitir crear modelos de características, otros permiten hacer solo configuraciones gráficamente, y algunos solo validan los modelos (Bobadilla Montoya y Sepúlveda Cuevas, 2020).
Algunas de las herramientas disponibles son:

- CIDE (https://wwwiti.cs.uni-magdeburg.de/iti_db/research/cide/) fue desarrollado en la Universidad de Magdeburg, Alemania por Christian Kästner (ahora en la Universidad Carnegie Mellon). CIDE es una herramienta prototipo para el desarrollo de líneas de productos de software. Sigue el paradigma de la separación virtual de concerns, y forma parte de un proyecto de investigación.
- FaMa (https://www.isa.us.es/3.0/tool/fama-tool-suite/) fue desarrollado por un equipo de la Universidad de Sevilla, España. Permite el análisis automatizado de los modelos de características integrando algunos de los resolutores más comúnmente propuestos (BDD, SAT y CSP), siendo esta una característica poco común. Dispone de un plugin de Eclipse gráfico desarrollado en EMF bajo licencia Open-Source.
- FAMILIAR (http://familiar-project.github.io/) es un lenguaje para importar, exportar, componer, descomponer, editar, configurar, calcular "diferencias", refactorizar, realizar ingeniería inversa, probar y razonar sobre (múltiples) modelos de características. Todas estas operaciones se pueden combinar para realizar tareas complejas de gestión de la variabilidad. Fue desarrollado por Laboratoire d'informatique, Université Nice Sophia Antipolis, Francia.
- FeatureIDE (https://featureide.github.io/) está desarrollado por Faculty of Computer Science, Workgroup Databases & Software Engineering, University of Madeburg, Alemania. Es un IDE basado en Eclipse que integra AHEAD, FeatureC++ y herramientas de FeatureHouse como herramientas de composición y compiladores entre otros. Es tanto un editor gráfico como de texto. No dispone de features clonables, ni la posibilidad de referencias entre features.
- FeatureMapper (http://featuremapper.org/) es una herramienta que combina la Ingeniería de LPS y el desarrollo de software basado en modelos. Presenta su propio metamodelo de características. FeatureMapper fue en desarrollo en el Grupo de Tecnología de Software de la Universidad de Dresden, Alemania.
- KumbangTools (http://www.soberit.hut.fi/KumbangTools/) es un paquete de aplicación que consta de Kumbang Configurator y Kumbang Modeler. KumbangTools está diseñado para configurar familias de productos de software. La herramienta toma como entrada un modelo de configuración y ofrece al usuario la posibilidad de tomar decisiones de configuración. La herramienta se implementa como complementos para Eclipse IDE. Fue desarrollada por el Software Business and Engineering Research & Education, Department of Computer Science and Engineering, Aaito University, Helsinsky, Finlandia.
- S.P.L.O.T. (http://www.splot-research.org/) es un desarrollo de la Universidad de Waterloo de Canadá, con fines académicos. Consiste en una herramienta web que dispone de una base de datos con gran cantidad de modelos de características base. El usuario puede subir su propio modelo de características que debe de ser escrito en XSML. Además el usuario puede crear una configuración a partir de un modelo que esté en esa base de datos y validarla. Es herramienta muy simple, por lo que no se dispone de características clonables, referencias, atributos, restricciones avanzadas o interfaz grafica.
- VariaMos (https://variamos.com/home/) es una herramienta de modelado y un framework, que se puede ampliar fácilmente y que permite definir los propios modelos. En esencia,

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

VariaMos incluye soporte para modelado de características, modelado de componentes y modelado de enlaces. Por lo tanto, se puede ampliar para admitir el ensamblaje de componentes y la verificación de características. Estos modelos permiten a los desarrolladores implementar un enfoque de LPS. Desarrollado por Centro de Investigación en Ciencias de la Computación, CRI, Paris 1 University, Francia.

3. HERRAMIENTAS DE DESARROLLO PARA MASHUP FEEDS

En esta Sección se introducirán conceptos relacionados sobre el caso de estudio a desarrollar, y que sirven como base para la creación y construcción de una LPS. El enfoque metodológico que se propone aplicar, corresponde al DSR (Design Science Research) (Peffers et al., 2008; y Offermann et al., 2009), este enfoque se dirige a la producción de artefactos. Por lo tanto, el trabajo se enfocó en el entorno de desarrollo visual basado en Web para el usuario final, denominado Pipes Digital.
A continuación se introducen algunos conceptos previos, antecedentes, descripción general de Pipes Digital, y por último, un breve resumen de estas herramientas.

3.1. Antecedentes

Los pipes (tuberías en español) fueron introducidos en el año 1973 por el sistema operativo (SO) Unix, y son una forma de redirección para enviar la salida de un programa a la entrada de otro, para continuar el procesamiento (Martínez Perales, 2013).
Tomando como referencia el concepto de pipes, se encuentran algunas herramientas o aplicaciones, que permiten a un usuario establecer vínculos o cadenas a través de una serie de módulos o bloques, en el qué la salida de un bloque se pasa como entrada para el siguiente bloque. Los bloques tienen distintos propósitos, permitiendo su manipulación de diversas maneras. Este concepto básico se aplica para mezclar tipos de feeds populares, como RSS/ATOM, permitiendo crear un mashup de datos.
A continuación se describen diferentes servicios que permite combinar varias fuentes de información utilizando los feeds RSS (o ATOM), aplicando el concepto de pipes.

3.2. Yahoo! Pipes

Esta herramienta fue creada en el año 2007 por la empresa Yahoo!, y se convirtió en una de las más utilizadas y populares en su momento para la manipulación de los feeds, hasta el año 2015 que fue cerrado el proyecto. El nombre Pipes está inspirado en el concepto de tuberías del sistema operativo Unix.
Según Yahoo[4], Pipes es un servicio en línea gratuito que le permite mezclar tipos de feeds populares y crear mashups de datos utilizando un editor visual. Yahoo ofrecía 53 módulos distribuidos en nueve categorías: fuentes de datos, entrada del usuario, operadores, URL, cadena String, fecha, ubicación, número y deprecated. Cada módulo cumplía una función única, y la salida de un módulo se podría conectar como entrada a otro módulo.
Yahoo! Pipes consistía básicamente en un sitio web de acceso público que permitía a los usuarios previamente registrados, buscar pipes existentes (navegando o buscando por palabra clave) y ejecutarlas. La ejecución a menudo implicaba el ingreso de parámetros, y se obtenía como resultado una página web. Los usuarios también podían editar pipes existentes y crear composiciones nuevas, tomando como entrada de datos uno o varios archivos RSS que luego se manipulaban con ciertas reglas y condiciones de filtrado/ordenado. El resultado final era un único feed RSS personalizado.

[4] http://pipes.yahoo.com/pipes/docs

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

Como entorno de programación visual, Yahoo Pipes era muy adecuado para representar soluciones a problemas de procesamiento basados en el flujo de datos (Dinmore y Boylls, 2012).
La Figura 5 muestra la pantalla principal del sitio y una descripción general de los elementos de un pipe típico:

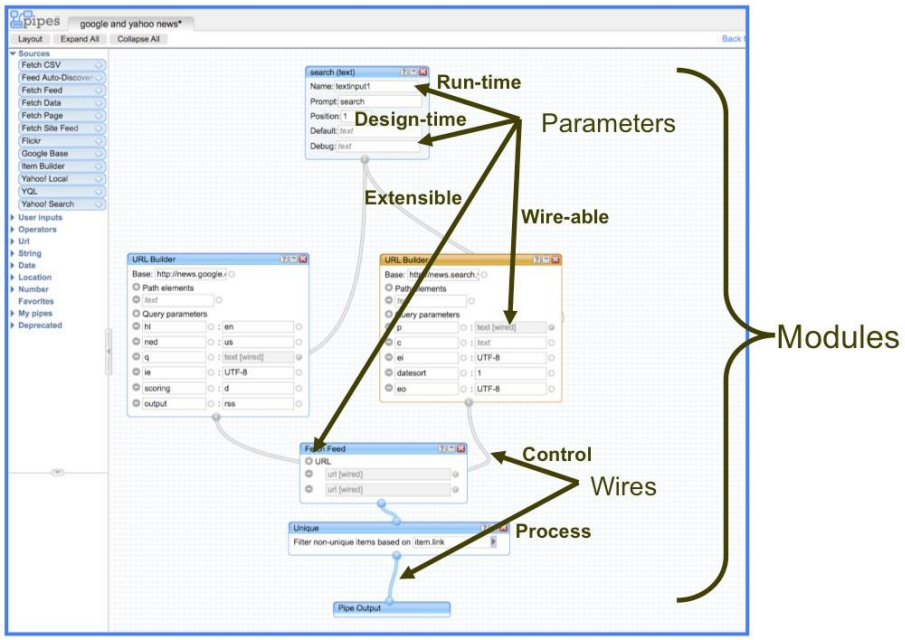

Figura 5. Vista de los elementos típicos de un Pipe.

Un pipe básicamente se creaba a partir de una selección de módulos o bloques ubicados visualmente al lado izquierdo del editor de Pipe, estos módulos se podían arrastrar al área de trabajo, donde se configuraba sus parámetros de diseño, y se podían conectar a otros módulos para formar el pipe.

3.3. Otras opciones similares a Yahoo Pipes!

Como alternativas a Yahoo! Pipes, a continuación se presenta brevemente alguna de ellas.

- **RSS Mix** (http://www.rssmix.com/) es una herramienta para mezclar o combinar hasta 100 feeds RSS en un único feed. Puede generar feeds en formato JSON para usarlo en otras aplicaciones. Las direcciones URL de las fuentes RSS combinadas se proporcionan automáticamente, lo que permite acceder a todos sus sitios web desde el mismo lugar. Es una de las herramientas más fácil de usar de este tipo.
- **Feed Informer** (http://feed.informer.com/) es una herramienta que permite mezclar y convertir feeds de manera rápida. Requiere el registro mediante una dirección de correo electrónico. Se puede elegir distintas opciones de salida (HTML a PDF), personalizar la plantilla de feed, agregar y publicar un resumen del mismo. Posee diversas herramientas que ofrecen opciones para convertir los feed Atom o RSS, en widgets que están listos para publicar en cualquier lugar de la página web. Posee completos tutoriales y documentación.
- **RSS Mixer** (https://rssmixer.com/) es una forma sencilla de combinar varios canales RSS en un solo canal. La versión gratis es limitada, y posee planes con tarifas pagas. Ofrece a los usuarios una solución rápida y simple para mezclar sus feeds de manera inmediata. La versión gratuita permite combinar hasta 3 feeds, que se actualizan solo una vez al día. En la versión paga se puede mezclar hasta 30 feeds que se actualizan cada hora. Para crear un feed se asigna un nombre al feed principal, una descripción, y se ingresan las URL que se

quieran incluir. No posee una herramienta de búsqueda, y son compatibles con muchos servicios.

- **Dlvr.it** (https://dlvrit.com/) posee varias de las capacidades de los feeds RSS que hicieron que Yahoo! Pipes fuera tan popular. Algunas de las funcionalidades que puede hacer son: ordenar y filtrar los feeds RSS para contenido seleccionado, combinar varias fuentes RSS en una nueva, agregar fuentes RSS con contenido nuevo, agregar contenido de toda la web, tweets de código geográfico, y crear widgets para un sitio web. Posee una versión pago y otra gratis.

3.4. Pipes Digital

Pipes.Digital[5] (PD), es el nombre de un servicio que permite combinar varias fuentes de información utilizando los feeds RSS o ATOM, y aplicar filtros para que el resultado sea otro feed RSS con los criterios que se definan. Funciona de manera similar a Yahoo! Pipes.
El objetivo de PD es proporcionar una infraestructura para que los usuarios trabajen con datos extraídos de la Web[6].
Básicamente consiste en un editor de programación visual para obtener datos de la web a través de los feeds, que permite combinar bloques de datos de manera intuitiva, para crear, buscar, y manipularlos de varias formas, como filtrar, extraer, fusionar y clasificar. Está centrado en el concepto de feeds, en el cual los datos fluyen elemento por elemento de un bloque a otro, con RSS como formato predeterminado.
Como formatos de entrada, Pipes admite feeds RSS, Atom y JSON, puede extraer documentos HTML y puede trabajar con archivos de texto normales. Un ejemplo simple es combinar varios canales RSS en uno y filtrar ese flujo de datos combinado para una palabra clave.
No es necesario conocimiento en programación para construir los pipes. Se puede utilizar de forma gratuita, con la limitación que solo se pueden crear hasta 3 pipes. Asimismo, posee una versión libre de código abierto denominada Pipes CE, con licencia AGPL-3.0[7].
PD posee una página de documentación que consta de información sobre pipes, una guía del usuario sobre la edición de pipes, y una guía de resolución de problemas. Asimismo posee un Blog con las últimas novedades.

Interfaz gráfica

La interfaz de PD se divide en 2 partes principales, y una barra de menú (Figura 6). Del lado izquierdo se encuentra las distintas estructuras o comandos de edición, denominados bloques, y en la parte central de la pantalla, el área de trabajo para conectar y configurar los distintos bloques para diseñar el pipe.
El proceso de creación o edición consiste en seleccionar un bloque, arrastrar y soltar en el área de trabajo, y aparecerá el bloque seleccionado con sus entradas y salidas. Luego se configura el bloque de acuerdo a criterio, para controlar su comportamiento. Para crear una conexión entre bloques, se deberá seleccionar una de las salidas del bloque, y se arrastra o selecciona la entrada de otro bloque, después se repite esta operación hasta completar la construcción del pipe. Para finalizar, el último bloque del pipe, se conecta a un círculo rojo que indica el fin de la construcción del pipe.

[5] https://www.pipes.digital
[6] https://www.pipes.digital/blog
[7] https://github.com/pipes-digital/pipes

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

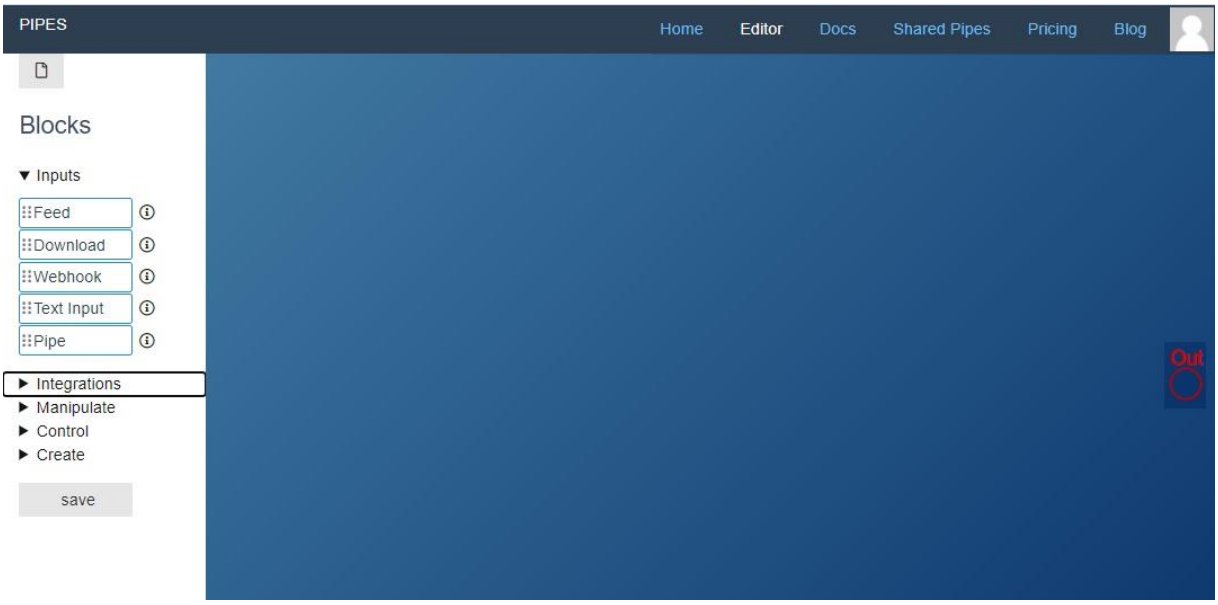

Figura 6. Interfaz principal de Pipes.Digital.

Composición de los bloques principales

Para la creación de un Pipe, el editor de PD, consta de 5 bloques, que poseen cada una de ellas, funcionalidades específicas:

a) **Input**: es el bloque principal para el ingreso de datos al pipe, posee las siguientes funciones: Feed, Download, Webhook, Text input, y Pipe. Se puede ingresar la URL de un feed, descargar una página, o hacerlo reutilizable a través de Text Input, entre otras funciones.
b) **Integrations**: permite integrar otras aplicaciones a un pipe, a través de funciones como: Dailymotion, Mixcloud, Periscope, Reddit, Speedrun, SVT Play, Tweets, Ustream, y Vimeo.
c) **Manipulate**: en este bloque se pueden realizar diferentes operaciones sobre el pipe, como buscar elementos mediante una palabra clave, eliminar elementos de feed duplicados, ordenar los datos de acuerdo a un criterio, y otras operaciones. Las funciones son: Filter, Replace, Unique, Truncate, Sort, y Shorten.
d) **Control**: permite controlar los pipes creados, combinar varios feeds en uno solo, o duplicar un feed, entre otras operaciones. Las funciones son: Combine, Duplicate, Merge Items, y Foreach.
e) **Create**: este bloque permite extraer elementos de una página o feed, para crear un nuevo feed a partir de estos elementos. Las funciones son: Extract, Images, Tables, Insert, y Build Feed.

Ejemplo

El ejemplo consiste en qué a partir de un feed de un diario digital, filtrar un feed con varias palabras claves, y generar uno nuevo de acuerdo con el criterio seleccionado.
El feed utilizado como ejemplo corresponde al diario digital denominado “Nuevo Día” de Río Gallegos, Pcia. de Santa Cruz (https://www.eldiarionuevodia.com.ar/), y se extrajo el RSS de la sección “Tecnología” del diario (https://www.eldiarionuevodia.com.ar/rss/feed.html?r=89).
Primero con el bloque *Feed*, se agrega como entrada la dirección URL del feed fuente (tecnología), luego seleccionar el bloque *Duplicate*, que permite aplicar más de un criterio de filtros, con el bloque *Filter* se ingresan los distintos criterios de filtros aplicados (Instagram, WhatsApp, Facebook), y como último paso, con el bloque *Combine*, se genera un nuevo feed con las palabras claves ingresadas (Figura 7).

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

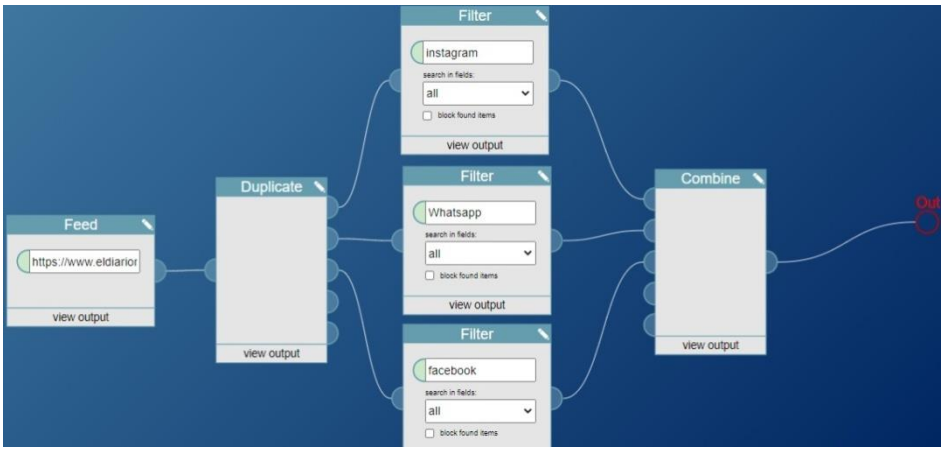

Figura 7. Vista resultado del ejemplo con Pipes.digital.

Seleccionando la opción “Pipe Output” ubicado en la sección “Block”, se genera el siguiente resultado:

1. Instagram pedirá documento para usar la red social
   2020-08-18 14:39:06 UTC
2. Comienza la fusión entre Facebook e Instagram
   2020-08-18 02:47:07 UTC
3. Así es como podés ganar dinero en Instagram sin ser influencer
   2020-08-11 11:36:19 UTC
4. De qué se trata Instagram Reels, la nueva forma de producir contenidos
   2020-08-07 13:21:26 UTC
5. Cómo escuchar un audio de Whatsapp sin que la otra persona lo sepa
   2020-08-15 19:27:10 UTC
6. Whatsapp permitirá usar la misma cuenta en distintos dispositivos
   2020-07-28 12:12:38 UTC
7. Comienza la fusión entre Facebook e Instagram
   2020-08-18 02:47:07 UTC

También posee la opción de generar o acceder al archivo RSS creado.

3.5. Resumen

En Tabla 3, se hace un resumen de las herramientas vistas teniendo presente algunas de sus características. En general, estas herramientas toman un feed como entrada, y genera uno nuevo de acuerdo a las prestaciones vistas en cada una de ellas. Así también, algunas poseen una versión paga y otra gratis, en donde la cantidad de feeds son limitadas en la versión gratis. El formato de salida puede ser RSS, XML, HTML, o PDF, y el acceso es mediante registro o confirmación por correo electrónico. Yahoo! Pipes se encuentra discontinuado, en cambio el resto se encuentran activos en la Web.

Herramienta	Versión Paga-Gratis	Acceso	Plataforma	Salida	Vigente
Yahoo!Pipes	-	Registro	Web	RSS	No
RSS Mix	Gratis	Libre	Web	XML	Si
Feed Informer	Ambas	Correo	Web	HTML/PDF	Si
RSS Mixer	Ambas	Correo	Web	XML	Si
Dlvr.it	Ambas	Registro	Web	-	Si
Pipes Digital	Ambas	Correo	Web	RSS	Si

Tabla 3. Herramientas para Mashup Feeds.

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

Tanto PD como Yahoo! Pipes en su momento, son herramientas que soportan lo que se denomina mashup de componentes (Daniel y Matera, 2014), al ofrecer entornos visuales que permiten a los usuarios seleccionar componentes predefinidos y combinarlos manualmente. No obstante, ninguna de las herramientas analizadas, carecen de algún modelo o funcionalidad para integrar componentes de manera apropiada; o seleccionar los modelos adecuados para la integración.

4. MODELADO DE UNA LPS PARA MASHUP FEEDS CON CARACTERÍSTICAS

En esta sección, se explica las funcionalidades del entorno de desarrollo visual Web, Pipes Digital, que se analizaron para extraer el MC; asimismo, se presenta el MC resultante; y se describen los resultados de verificar si el modelo propuesto posee defectos sintácticos o semánticos.

4.1. Entorno de desarrollo Web: Pipes.digital

De acuerdo a lo mencionado en la sección anterior, PD presenta cinco bloques con sus respectivas funciones. A continuación se describe las funcionalidades analizadas:

A. Input (entrada): Según el contenido se clasifican en:

1. Feed: Es el bloque por defecto como punto de partida de un pipe. Descarga un feed RSS, Atom o JSON.
2. Download: Este bloque descarga algunos datos como una página Html, que luego se transforman con otros bloques para crear un feed regular. Es un punto de partida alternativo para un pipe.
3. Webhook: Este bloque crea un feed a partir de los datos recibidos por un Webhook como entrada.
4. Text Input: Bloque permite que los usuarios ingresen datos como entrada de los bloques conectados, permitiendo cierto dinamismo de las tuberías.
5. Pipe: Permite el uso de un pipe o tubería como bloque de entrada. Esto permite la creación de pipes más complejos al combinar varios pipes, como submódulos.

B. Integrations (integración): Según el contenido se clasifican en:

1. Dailymotion: permite ingresar una dirección o url de Dailymotion.
2. Mixcloud: permite ingresar un nombre de usuario de Mixcloud o una URL.
3. Periscope: permite ingresar un nombre de usuario de Periscope o una URL.
4. Reddit: permite ingresar un subreddit (sección) de Reddit, o varios separados por coma.
5. Speedrun: permite ingresar un nombre de juego o una URL de speedrun.com.
6. SVT Play: permite ingresar un nombre de programa de SVT Play o una URL.
7. Tweets: permite recibir tweets de Twitter, como así también ingresar un término de búsqueda normal, un #hashtag o un @username.
8. Ustream: permite ingresar un nombre de canal de Ustream o una URL.
9. Vimeo: permite ingresar el nombre de un canal de Vimeo, la identificación de video o la URL de un canal o video.

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

C. Manipulate (operaciones): Según el contenido se clasifican en:

1. Filter: Retorna solo los elementos en un feed que contengan la palabra clave en su descripción, título o contenido. Se distingue entre mayúsculas y minúsculas, y buscará interpretando una expresión regular dada.
2. Replace: Reemplaza el texto en el contenido de un elemento del feed. Admite expresiones regulares.
3. Unique: Elimina los elementos duplicados de un feed.
4. Truncate: Limita la longitud de salida de un feed, por ejemplo, mostrar los 5 resultados más recientes de un feed.
5. Sort: Ordena el feed de entrada, por el item del elemento seleccionado.
6. Shorten: Acorta el contenido de un feed a una longitud determinada.

D. Control: Según el contenido se clasifican en:

1. Combine: Combina las fuentes de entrada en una única fuente. Los elementos permanecen en el orden en que están, solo se concatena todo el feed de entrada.
2. Duplicate: Dividir una fuente en varias fuentes es útil si desea manipularlos de varias formas y luego unirlos nuevamente. Un ejemplo es filtrar un feed para varias palabras clave.
3. Merge Items: Une el contenido de dos elementos del feed en uno.
4. Foreach: Este bloque se usa para repetir la acción de un bloque de descarga, fuente o tweets para cada elemento de la fuente de entrada.

E. Create (creación): a partir del diseño de pantalla seleccionado se elige la forma de navegar:

1. Extract: A través de comando XPath permite seleccionar un elemento de un archivo XML descargado o de una fuente, un selector CSS para obtener datos de un documento HTML o JSONPath para datos JSON.
2. Images: Permite extraer todas las imágenes del feed de entrada o la página.
3. Tables: Extrae todas las tablas HTML del feed de entrada o la página. Los convierte en JSON y elimina el resto del contenido.
4. Insert: Inserta un elemento en otra fuente en una posición especificada por un comando XPath.
5. Build Feed: Permite conectar el resultado del bloque de extracción a este bloque para crear una nueva fuente RSS.

4.2. Modelo de características propuesto

En este trabajo se representa a cada característica como una funcionalidad u operación de los bloques definidos por PD.
La Figura 8 presenta el MC en el dominio para la LPS aplicados a Mashup. Para el diagrama de características se utilizó el entorno visual de desarrollo FeatureIDE[8], el cual consiste en un plugins para Eclipse, es open source y acepta varios lenguajes de programación (Gherardi y Brugali, 2011).

[8] https://featureide.github.io/

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

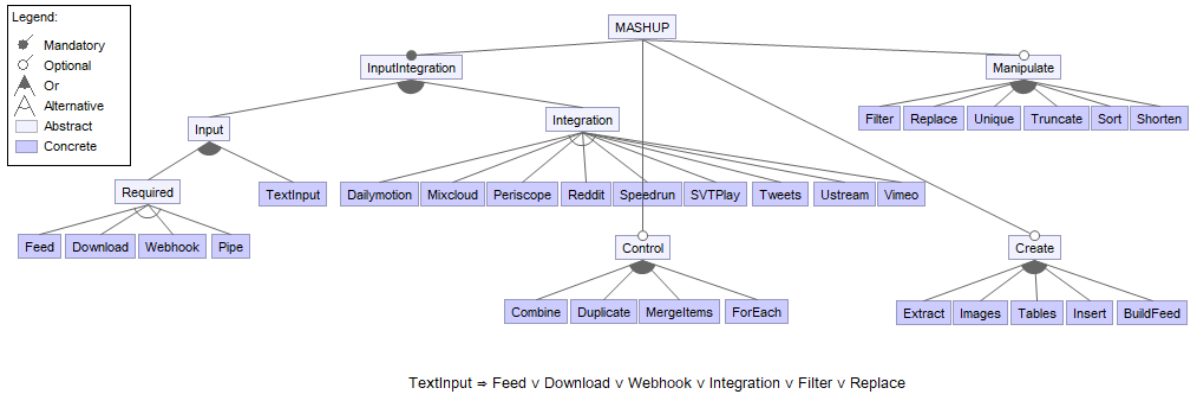

Figura 8. Modelo de características de LPS para aplicaciones Mashup.

Este MC se obtuvo luego de realizar un proceso iterativo de refinamiento. Así, inicialmente fue elaborado un MC para cada una de las características encontradas. Luego, cada uno de estos modelos fue examinado con el fin de identificar en ellos, características y dependencias comunes. Después se identificaron aquellas características que pudieran ser agrupadas en características más generales; y finalmente, se clasificaron en obligatorias (forman parte de todos los productos que contienen la LPS), opcionales (puede ser o no parte de un producto), alternativas (sólo una característica puede ser seleccionada), y disyuntivas (más de una característica puede ser elegida).

Así por ejemplo, las características que corresponden a las funciones de *Input* y de *Integration*, que se encontraban cada una en características diferentes, fueron agrupadas en un sola característica padre denominada *InputIntegration*, como hijas de la característica padre *Mashup* en el MC propuesto.

Como resultado del análisis, se obtuvo la característica obligatoria *InputIntegration*, y las características opcionales *Manipulate* (Operaciones), *Control*, y *Create* (Creación). La característica *InputIntegration* hace referencia a la funcionalidad principal de las aplicaciones analizadas para construir la LPS. En este caso, las operaciones disponibles son *Input*, e *Integration*, de las que se desprenden las características *Required* (*Feed*, *Download*, *Webhook*, y *Pipe*) y *TextInput*. Para el caso de *Integration*, se agrupan las características *Dailymotion*, *Mixcloud*, *Periscope*, *Reddit*, *Speedrun*, *SVT Play*, *Tweets*, *Ustream*, y *Vimeo*. Cualquier producto que se configure del MC deberá tener al menos un *InputIntegration* definido.

La característica *Input* se relaciona con el ingreso de datos a las operaciones, las cuales pueden ser *Required* (requerida) o *TextInput*. La característica *Required* posee 4 características hijas, *Feed*, *Download*, *Webhook*, y *Pipe*. Por otro lado, la característica *Integration*, representa las distintas formas de integración posibles del producto, de las cuales se desprenden las siguientes características, *Dailymotion*, *Mixcloud*, *Periscope*, *Reddit*, *Speedrun*, *SVTPlay*, *Tweets*, *Ustream*, y *Vimeo*.

La característica *Manipulate*, indica las operaciones disponibles que se pueden realizar dentro del contenido de un feed, como ser *Filter*, *Replace*, *Unique*, *Truncate*, *Sort*, y *Shorten*.

La característica *Control*, hace referencia a las operaciones disponibles que se pueden aplicar a un feed, como combinar, duplicar, y unir. Se desprenden las siguientes características, *Combine*, *Duplicate*, *MergeItems*, y *ForEach*.

Finalmente, la característica *Create*, se encarga de crear extraer alguna parte del documento, las imágenes o tablas, para crear nuevos feeds. Las características que dependen de *Create* son: *Extract*, *Images*, *Tables*, *Insert*, y *BuildFeed*.

La Tabla 4 describe las características representadas en la Figura 8, muestra el nivel, su característica padre, y el tipo de relación (obligatoria, opcional, alternativa y disyuntiva).

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

4.3. Configuración de productos

La instanciación de los productos del MC deben cumplir con distintos tipos de restricciones, todas las aplicaciones Mashup deben especificar alguna característica del grupo InputIntegration (Entrada de Datos e Integración), lo que constituye una característica obligatoria; asimismo, pueden especificar opciones de Input, y de Integration; para el caso de Input, se definen la característica Required (característica obligatoria) y TextInput (característica opcional); finalmente se debe especificar alguna característica para el caso del grupo Integration (característica obligatoria). Asimismo, también puede especificar opciones del grupo Manipulate (Operaciones), de Control, y de Create (Creación), siendo estas características opcionales.

Características	Descripción	Nivel	Padre	Relación
Mashup	Aplicación Mashup	0	-.-	-.-
InputIntegration	Bloque para ingreso de datos e integración	1	Mashup	Obligatoria
Input	Bloque principal para ingreso de datos	2	InputIntegration	Disyuntiva
Required	Bloque obligatorio para ingreso de datos	3	Input	Disyuntiva
Feed	Bloque por defecto	4	Required	Alternativa
Download	Bloque de descarga pagina	4	Required	Alternativa
Webhook	Crea un feed con datos de Webhook	4	Required	Alternativa
Pipe	Bloque de entrada pipe	4	Required	Alternativa
TextInput	Ingreso de datos en el bloque	3	Input	Disyuntiva
Integration	Permite integrar otras aplicaciones a un pipe	2	InputIntegration	Disyuntiva
Dailymotion	Bloque ingreso url de Dailymotion	3	Integration	Alternativa
Mixcloud	Bloque ingreso url de Mixcloud	3	Integration	Alternativa
Periscope	Bloque ingreso url de Periscope	3	Integration	Alternativa
Reddit	Bloque ingreso url de Rededit	3	Integration	Alternativa
Speedrun	Bloque ingreso url de Speedrun	3	Integration	Alternativa
SVTPlay	Bloque ingreso url de SVT Play	3	Integration	Alternativa
Tweets	Bloque ingreso tweets, #hastag, o @username	3	Integration	Alternativa
Ustream	Bloque ingreso url/canal de Ustream	3	Integration	Alternativa
Vimeo	Bloque ingreso url/canal de Vimeo	3	Integration	Alternativa
Manipulate	Bloque de Operaciones	1	Mashup	Opcional
Filter	Aplica filtro de acuerdo a criterios	2	Manipulate	Disyuntiva
Replace	Reemplaza texto de un elemento del feed	2	Manipulate	Disyuntiva
Unique	Elimina elementos duplicados	2	Manipulate	Disyuntiva
Truncate	Limita la longitud de elementos de salida	2	Manipulate	Disyuntiva
Sort	Ordena un feed de acuerdo a criterios	2	Manipulate	Disyuntiva
Shorten	Trunca el contenido de feed a una longitud	2	Manipulate	Disyuntiva
Control	Bloque de Control	1	Mashup	Opcional
Combine	Combina feeds en una única salida	2	Control	Disyuntiva
Duplicate	Divide un feed en varios fuentes	2	Control	Disyuntiva
MergeItems	Une el contenido de dos elementos del feed	2	Control	Disyuntiva
ForEach	Repite la acción de una fuente de entrada	2	Control	Disyuntiva
Create	Bloque de Creación nuevos feeds	1	Mashup	Opcional
Extract	Extrae un elemento del feed con XPath	2	Create	Disyuntiva
Images	Extrae solo imágenes del feed o pagina	2	Create	Disyuntiva
Tables	Extrae las tablas HTML del feed/página	2	Create	Disyuntiva
Insert	Inserta un elemento en otra fuente con XPath	2	Create	Disyuntiva
BuildFeed	Crea un nuevo feed conectando otras fuentes	2	Create	Disyuntiva

Tabla 4. Descripción de las características del MC para aplicaciones con Pipes.

Seguidamente, se presenta la configuración del MC de aplicaciones para Mashup (Capilla et al., 2014):

- InputIntegration =>Mashup

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

- Manipulate ∨ Control ∨ Create <=>Mashup
- (Input ∨ Integration <=> InputIntegration) ∧¬ (Input ∧ Integration)
- (Required ∨ TextInput) <=> Input) ∧¬ (Required ∧ TextInput)
- Feed ∨ Download ∨ Webhook ∨ Pipe <=> Required
- Dailymotion ∨ Mixcloud ∨ Periscope ∨ Reddit ∨ Speedrun ∨ SVTPlay ∨ Tweets ∨ Ustream ∨ Vimeo <=> Integration
- (Filter ∨ Replace ∨ Unique ∨ Truncate ∨ Sort ∨ Shorten <=> Manipulate) ∧¬ (Filter ∧ Replace ∧ Unique ∧ Truncate ∧ Sort ∧ Shorten)
- (Combine ∨ Duplicate ∨ MergeItems ∨ ForEach <=> Control) ∧¬ (Combine ∧ Duplicate ∧ MergeItems ∧ ForEach)
- (Extract ∨ Images ∨ Tables ∨ Insert ∨ BuildFeed <=> Create) ∧¬ (Extract ∧ Images ∧ Tables ∧ Insert ∧ BuildFeed)

La configuración anterior permite expresar en forma textual el MC. Todas las características que pertenecen a una aplicación están representadas en la primera y segunda línea de la configuración.

- La característica *InputIntegration* es obligatoria, y debe estar en toda aplicación de Mashup.
- Solo una de las características de *InputIntegration* puede ser seleccionadas, *Input* o *Integration*.
- Más de una de las características de *Input* pueden ser elegidas, *Required* y *TextField*.
- Solo una de las características del grupo *Required* que depende de la característica *Input*, pueden ser seleccionadas, *Feed*, *Download*, *Webhook*, y *Pipe*.
- Solo una de las características del grupo *Integration* debe ser seleccionada, *Dailymotion*, *Mixcloud*, *Periscope*, *Reddit*, *Speedrun*, *SVT Play*, *Tweets*, *Ustream*, o *Vimeo*, si hace referencia a *Integration*.
- Más de una característica del grupo *Manipulate* pueden ser seleccionadas, *Filter*, *Replace*, *Unique*, *Truncate*, *Sort*, y *Shorten*.
- Más de una característica del grupo *Control* pueden ser seleccionadas, *Combine*, *Duplicate*, *MergeItems*, y *ForEach*.
- Más de una característica del grupo *Create* pueden ser seleccionadas, *Extract*, *Images*, *Tables*, *Insert*, y *BuildFeed*.
- Por último, la característica *TextInput* requiere al menos de una de las siguientes características: *Feed*, *Download*, *Webhook*, *Integration*, *Filter*, o *Replace* (constraints).

Un producto de LPS se especifica de forma declarativa seleccionando o anulando la selección de características de acuerdo a la necesidad del usuario. Las decisiones tomadas deben respetar las limitaciones del MC.
Para detallar las características empleadas en el MC de un producto o aplicación, se especifica de la siguiente manera:
Producto={ Feed, Download, Webhook, Pipe, TextInput, Dailymotion, Mixcloud, Periscope, Reddit, Speedrun, SVTPlay, Tweets, Ustream, Vimeo, (Filter, Replace, Unique, Truncate, Sorte, Shorten), (Combine, Duplicate, MergeItems, ForEach), (Extract, Images, Tables, Insert, BuildFeed) }

4.4. Verificación del modelo de características

Verificar el MC implica detectar posibles errores sintácticos y semánticos que pueden surgir al modelar la LPS. Por una parte los defectos sintácticos se producen cuando el MC no

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

cumple con las reglas de la notación, como por ejemplo tener dos características raíz o tener características sin dependencias (Mazo et al., 2011). Por otra parte los errores semánticos son imperfecciones que afectan la capacidad del MC para representar de manera correcta el dominio de la LPS (Rincón Perez et al., 2013). Algunos de estos defectos semánticos son: características muertas, modelos vacíos, falsos modelos de líneas de productos, características falsas opcionales y dependencias redundantes.
Para evitar estos tipos de errores o validarlos, el MC se diseñó en el entorno FeatureIDE que sólo permite construir modelos de características que se adaptan correctamente a la notación, y presenta un análisis sintáctico y semántico del modelo propuesto (Figura 9).

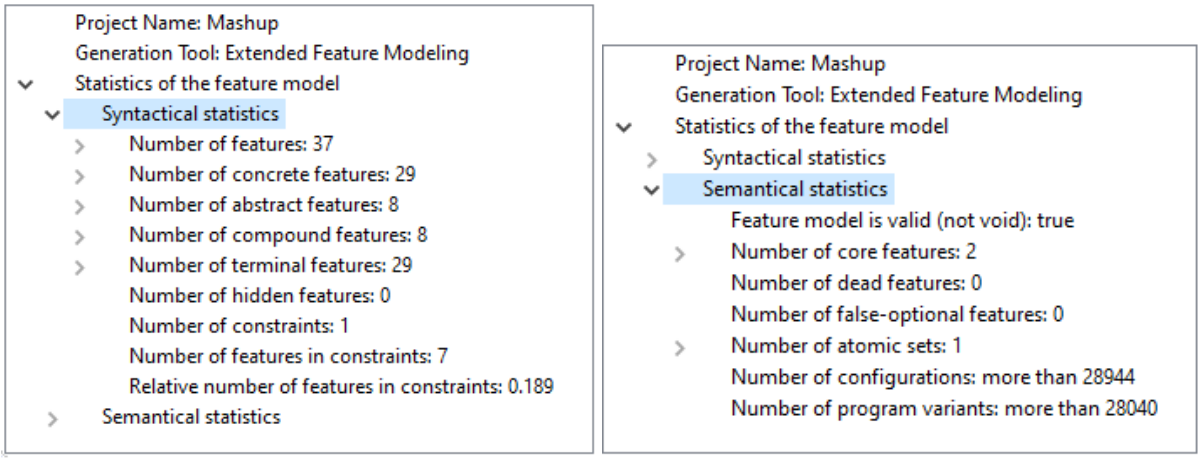

Figura 9. Información de estadística del MC propuesto.

Así también, el MC implementado también fue construido y validado en la plataforma denominada SPLOT (Software Product Lines Online Tools)[9], la cual recibe un MC junto a las restricciones que existen entre ellas, y genera un análisis automatizado de dicho modelo. El MC fue guardado en el repositorio de SPLOT y se encuentra disponible en Internet con el nombre “MashupFeed”. La Figura 10 muestra los detalles del modelo almacenado.

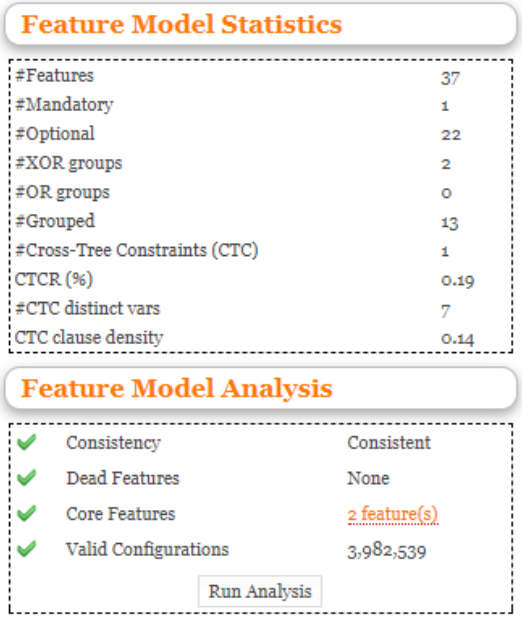

Figura 10: Resultados del análisis realizado por SPLOT al MC propuesto.

[9] http://www.splot-research.org

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

De estos datos se destaca que existen 37 features, de las cuales 22 son opcionales y 1 obligatorio, y están reunidas en 13 grupos. El número total de configuraciones es de 3.982.539.

4.5. Casos de estudios. Modelo de Características de aplicaciones Mashup

En esta Sección se describe una serie de casos que emplean características del modelo de dominio con el fin de validar la LPS presentada. Para cada producto se presenta un MC diferente. Las características empleadas por cada uno de los productos se remarcaron con un cuadro rojo.

Caso 1:
A partir de un feed RSS de un diario digital, se filtra el feed con varias palabras claves, y genera uno nuevo de acuerdo con el criterio seleccionado. La Figura 11 ilustra el MC respectivo.

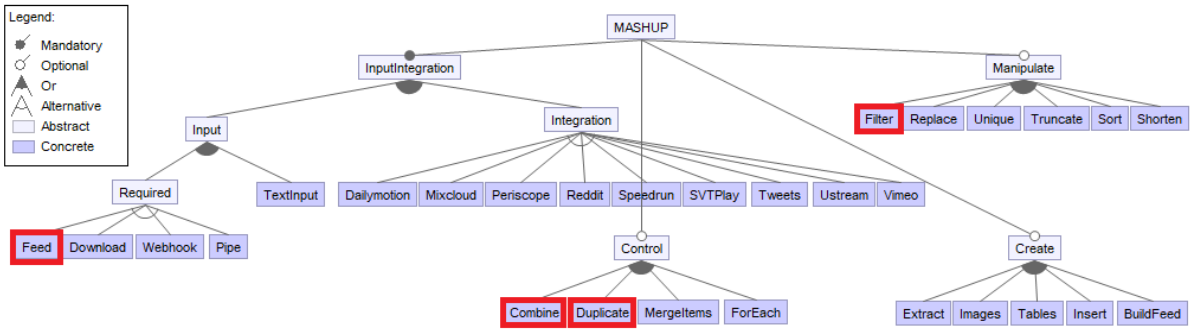

Figura 11: Modelo de Características del Caso 1.

Especificación del MC:
Caso1 = { Feed, Combine, Duplicate, Filter }

Caso 2:
Permite ingresar un nombre o alias de Twitter, y muestra los tweets del usuario ingresado. La Figura 12 ilustra el MC respectivo.

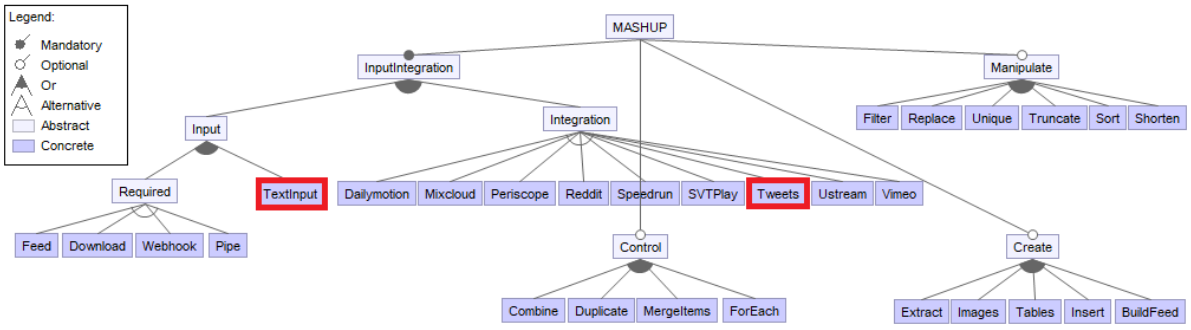

Figura 12: Modelo de Características del Caso 2.

Especificación del MC:
Caso2 = { TextInput, Tweets }

ICT-UNPA-283-2021
ISSN: 1852-4516
Aprobado por Resolución N° 0599/21-R-UNPA
http://doi.org/10.22305/ict-unpa.v13.n2.814

Caso 3:
En este caso, utilizando el bloque Download para extraer el contenido de una página y generar un nuevo Feed RSS. La Figura 13 ilustra el MC respectivo.

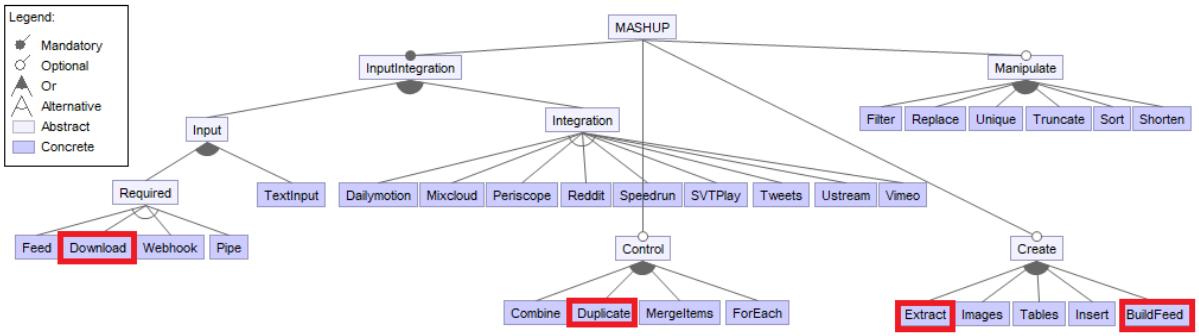

Figura 13: Modelo de Características del Caso 3.

Especificación del MC:
Caso3 = { Download, Duplicate, Extract, BuildFeed}

5. CONCLUSIONES

En este trabajo, se propone el modelado y diseño de una LPS para componer e integrar aplicaciones Mashup. Para ello, y teniendo en cuenta una tecnología disponible y fundamental para desarrollar un Mashup, como son los Feeds, se procedió a estudiar y analizar el entorno de desarrollo visual Web Pipes Digital, y a partir de la misma, modelar una línea de productos de software mediante un MC. La abstracción de los requerimientos funcionales de la aplicación Pipes Digital, permitió que el MC propuesto considerara aspectos comunes, a la vez que identificará puntos de variación entre ellas. El análisis realizado, ofrece elementos y justifica llevar a cabo la implementación de la línea de productos. En función a estos resultados preliminares, las hipótesis planteadas al inicio son afirmativas, resultando del estudio, un modelo de líneas de productos de software.
Actualmente esta propuesta se encuentra en el análisis del dominio inicial de la LPS. Como trabajo futuro se propone refinar el MC desarrollado, seguir con la etapa del diseño del modelo de características y la implementación de la Línea de Producto de Software para Mashup.

6. REFERENCIAS

ACHER, M., BAUDRY, B., HEYMANS, P., CLEVE, A. y HAINAUT, J. L. (2013). Support for Reverse Engineering and Maintaining Feature Models. Seventh International Workshop on Variability Modelling of Software-Intensive Systems (VaMoS'13), pp.20 ACM. https://doi.org/10.1145/2430502.2430530

APEL, S., BATORY, D., KÄSTNER, C., y SAAKE, G. (2013). Feature-Oriented Software Product Lines. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-37521-7

BOBADILLA MONTOYA, A., y SEPÚLVEDA CUEVAS, S. (2020). Un punto de vista de evaluación de la calidad para las herramientas de modelado de características. Revista Ibérica de Sistemas e Tecnologias de Informação. RISTI, N.º E28, 04/2020. Páginas: 123–138.

CAPILLA, R., BOSCH, J. y KANG, K. C. (2013). Systems and Software Variability Management. Springer-Verlag Berlin Heidelberg. doi:10.1007/978-3-642-36583-6. https://doi.org/10.1007/978-3-642-36583-6

CAPILLA, R., BOSCH, J., TRINIDAD, P., RUIZ CORTÉS. A. y HINCHEYD, M. (2014). An overview of Dynamic Software Product Line architectures and techniques: Observations from research and industry. The Journal of Systems and Software 91, pp. 3–23. https://doi.org/10.1016/j.jss.2013.12.038

DANIEL, F., MUHAMMAND, I., SOI, S., DE ANGELI, A., WIKINSON, C., CASATI, F., y MARCHESE, M. (2012). "Developing Mashup Tools for End.Users: On the Importance of the Application Domain", International Journal on Next-generation Computing, Vol 2. No 2.

DANIEL, F., y MATERA, M. (2014). Mashups Concepts, Models and Architectures. Springer, Heidelberg, 2014. ISBN 978-3-642-55048-5.

DÍAZ, J., QUEIRUGA, C., IULIANO, P., ROSSO, J., KIMURA, I., y BARNETCHE, M. (2012). Building Mobile Mashup Applications. Some Challenges. Encountered in Computer Science Degrees. LINTI, Computer Science School, Universidad Nacional de La Plata, La Plata, Buenos Aires, Argentina.

DINMORE, M. y BOYLLS, C. (2012). Empirically-Observed End-User Programming Behaviors in Yahoo! Pipes.

GALINDO, J. A., ALFÉREZ, M., ACHER, M., BAUDRY, B. y BENAVIDES, D. (2014). A variability-based testing approach for synthesizing video sequences. InProceedings of the 2014 International Symposium on Software Testing and Analysis, pp. 293-303. https://doi.org/10.1145/2610384.2610411

GHERARDI, L., BRUGALI, D. (2011). An eclipse-based Feature Models toolchain. An eclipse-based feature diagrams toolchain, The Sixth Workshop of the Italian Eclipse Community, pp. 242-253. In Eclipse-IT.

KÄSTNER, C., y APEL, S. (2011). Feature-Oriented Software Development: A Short Tutorial on Feature-Oriented Programming, Virtual Separation of Concerns, and Variability-Aware Analysis. In GTTSE Summer School: Generative & Transformational Techniques in Software Engineering, volume 7680 of Lecture Notes in Computer Science, Berlin/Heidelberg: Springer-Verlag, pp. 346-382. http://www.cs.cmu.edu/. https://doi.org/10.1007/978-3-642-35992-7_10

MANDAL, S., MUKHOPADHYAY, P., DUTTA A. (2019). Information mashup through application of Web 2.0 tools: services and procedures. Annals of Library and Information Studies 66(4):140-151.

MARTÍNEZ PERALES, D. (2013). Unix a base de ejemplos. España: LULU.

MATERA, M., y PICOZZI, M. (2015). Next Generation Web Apps. Towards Transformative UX. Politecnico de Milano Dpto. Electronica e Informática.

MAZO, R., LOPEZ-HERREJON, R., SALINESI, C., DIAZ, D., y EGYED, A. (2011). Conformance Checking with Constraint Logic Programming: The Case of Feature Models, in Proceedings of the 35th Annual International Computer Software and Applications Conference (COMPSAC), pp. 456 465. https://doi.org/10.1109/COMPSAC.2011.66

METZGER, A. y POHL, K. (2014). Software product line engineering and variability management: achievements and challenges. In Proceedings of the on Future of Software Engineering, pp. 70-84. http://dl.acm.org/citation.cfm?id=2593888. https://doi.org/10.1145/2593882.2593888

NAVARRO FAVELA, J. G., JUÁREZ MARTÍNEZ, U. (2013). Cómo desarrollar una línea de productos de software, un enfoque práctico. 4TH International Conference on Computer Science and Its Application (CIIA 2013). ISBN 978-607-9119-02-7, pp. 16-25. http://ciia.itsm.edu.mx

NORTHROP, L. M., y CLEMENTS, P.C. (2012). A Framework for Software Product Line Practice, Version 5.0. Software Engineering Institute-Carnegie Mellon University. Recuperado de https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=495357

NOTTINGHAM, M. y SAYRE, R. (2005). The Atom Syndication Format - RFC 4287. Retrieved from http://tools.ietf.org/html/rfc4287. https://doi.org/10.17487/rfc4287

OFFERMANN, P., LEVINA, O., SCHÖNHERR, M., y BUB, U. (2009). Outline of a Design Science Research Process. DESRIST '09, ACM, New York, p. Article No.: 7. https://doi.org/10.1145/1555619.1555629

O'SHEA, M. (2016). A series of case studies to enhance the social utility of RSS. Tesis. Department of Computer Science & Information Systems, Birkbeck, University of London.

PEFFERS, K., TUUNANEN, T., ROTHENBERGER, M. y CHATTERJEE, S. (2008). A Design Science Research Methodology for Information Systems Research. Journal of MIS (24:3), 45-77. https://doi.org/10.2753/MIS0742-1222240302

RINCÓN PEREZ, L., GIRALDO GÓMEZ, G., MAZO, R., SALINESI, C., y DIAZ, D. (2013). Subconjuntos Minimos de Corrección para explicar caracteristicas muertas en Modelos de Lineas de Productos. El caso de los Modelos de Caracteristicas, in Proceedings of the 8th Colombian Computer Conference. https://doi.org/10.1109/ColombianCC.2013.6637515

SEPÚLVEDA, S., CACHERO, C., y CARES, C. (2012). Modelado de Características para Líneas de Producto de Software: una propuesta. In International Workshop On Advanced Software Engineering, Valparaíso.

SHEONG, Ch. S. (2008). Ruby on Rails Web Mashup Projects. Publisher Packt. ISBN 9781847193933.

SIKOS, L. F. (2011). Web Standards-Mastering HTML5, CSS3, and XML. ISBN-13: 978-1-4302-4042-6. Editorial Apress.

SINGH, G., y SAHU, S. (2015). Review on "Really Simple Syndication (RSS) Technology Tools". IEEE International Conference on Computational Intelligence & Communication Technology. 978-1-4799-6023-1/15.

TINAJERO DÍAZ, I. (2016). Composición de sistemas con Mashups: El caso PhysicallTrello. Centro de Investigación de Matemática A.C. Zacatecas.

TRINH, T. D. (2016). Mashup-based Linked Data Integration. Faculty of Informatics at the Vienna. University of Technology.

YEE, R. (2008). Pro Web 2.0 Mashups: Remixing Data and Web Services. Apress.