Una Aproximación de Línea de Producto para la Generación de Informes de Bases de Datos

Felipe I. Anfurrutia, *Student Member, IEEE*, Oscar Díaz, y Salvador Trujillo

***Resumen*—La generación de informes es una demanda habitual para los programadores de las bases de datos. A diferencia del análisis de datos dinámicos, la generación de informes tiende a ser previsible. Aún así, el personal dedica su tiempo a programar repetidamente informes similares, dónde los informes son frecuentemente construidos desde el inicio, con poca reutilización, en el caso de que la haya. Basándose en la similitud entre los informes, este trabajo presenta una aproximación de línea de producto para la generación de informes de bases de datos. Además, describe el modelo de características, los elementos comunes y el plan de producción de la línea de producto, y realiza una discusión de la arquitectura utilizada. Este trabajo ha sido realizado bajo un requisito principal: la no disponibilidad de un almacén de datos (warehouse). El coste o la falta de personal formado, son razones que llevan a una pequeña-y-mediana empresa (PYME) a resistirse en el uso de estas sofisticadas herramientas, mientas que la generación de informes simples es todo lo que ellos necesitan. En este escenario, la aproximación de línea de producto puede ser una solución efectiva para conseguir la reutilización.**

***Palabras claves*— Automatic programming, Business data processing, Manufacturing data processing, Software reusability, Specification languages**

I. Introducción

Muchas pequeñas organizaciones, confeccionan sus informes de datos directamente desde la base de datos transaccional (OLTP). Este enfoque puede ser suficiente si no se requiere de un análisis de datos sofisticado, o los recursos no están disponibles. La tecnología de almacén de datos (warehouse) ofrece un soporte eficaz para el análisis, pero ello también implica tener personal y equipamiento cualificado que pequeñas organizaciones no siempre pueden permitirse.

Acceder directamente al sistema OLTP para la generación de informes, incurre en dos importantes limitaciones (aparte de soportar el histórico de datos y el asunto del ETL). En primer lugar, el rendimiento del sistema OLTP se ve degradado al tener que soportar tanto las aplicaciones de OLTP como los generadores de informes.

Este trabajo ha sido parcialmente financiado por el Ministerio de Educación y Ciencia (MEC) del Gobierno Español bajo el contrato TIC2002-01442. Salvador Trujillo disfruta de una beca doctoral del MEC.

Felipe. I. Anfurrutia (felipe.anfurrutia@ehu.es), Oscar Díaz (oscar.diaz@ehu.es) , y Salvador Trujillo (struji@ehu.es) pertenecen al grupo de investigación ONEKIN (http://www.onekin.org) de la Universidad del País Vasco, San Sebastián (España).

En segundo lugar, una complejidad adicional asociada a generar los informes directamente desde el sistema OLTP es que el esquema de la BD, rara vez, es intuitivo para el gerente de la empresa interesado en obtener el informe. Los datos han sido almacenados para optimizar la entrada de datos, no para optimizar el acceso a los datos. Esto a menudo conduce a dedicar personal específico para la generación de informes. Este personal está cada vez más atareado con la generación de nuevos informes o adaptar informes previos a nuevas necesidades de análisis, con lo que se tiene que detraer nuevos recursos que se obtienen a costa de ralentizar el desarrollo del resto de las aplicaciones [12].

Sin embargo, y a diferencia del análisis de datos dinámicos, la generación de informes tiende a ser previsible. Esta característica junto con la similitud existente entre distintos informes, avalan la oportunidad de una aproximación de línea de producto para la generación de informes de bases de datos.

Las Líneas de Producto Software (LPS) se definen como *"un conjunto intensivo de sistemas software que comparten un conjunto común y gestionado de características (features), dónde estas características están pensadas para satisfacer las necesidades especificas de una misión o de un segmento de mercado. Asimismo, los productos son desarrollados de una forma pre-establecida a partir de un conjunto común de componentes"* [6].

En este caso, una familia de informes es un conjunto de informes que comparten algunos componentes. El diseñador de la línea de producto identifica cuáles son estos componentes comunes, y cuáles las partes variantes. De esta forma, un informe concreto es un producto de la línea, donde el producto queda singularizado por los datos que utiliza y su maquetación.

Por tanto, este trabajo parte de las siguientes bases:

1. **no está disponible una tecnología de almacén de datos (warehouse)**. El coste del software o la falta de personal formado, hacen a las PYMES reticentes al uso de estas sofisticadas herramientas, mientas que un simple generador de informes pueden solventar la mayor parte de sus necesidades de análisis de datos.
2. **evitar ralentizar el sistema OLAP, separando todo lo posible el generador de informes de la base de datos**. Para este fin, los datos objeto del análisis son extraídos y guardados como ficheros XML. Estos datos vendrían a ser un tipo de caché de datos [1]. En un entorno PYME, la obsolescencia de esta caché, tiende a ser bastante grande para permitir recalcular la caché en modo de

Producción PSC
Fecha: 2006-02-24

Control de los Paros

Desde:10/11/2004
Hasta:15/11/2004

onekin

Sección
Maquina
CategoríaA de estados
Estado
duración

			PRETENS					GRUESO						FINO				Total maquinas	
			PG85	PG91	PG92	Total PRETENS		PG99	PES5	PG96	PES4	Total GRUESO		PG76	PG77	Total FINO			
						horas	%					horas	%			horas	%	horas	%
NO IMPUTABLES	Falta de aire		0.0	0.6	0.0	0.6	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6	0.4
	Falta de agua		0.0	0.0	0.0	0.0	0.0	0.8	0.0	0.0	0.0	0.8	0.7	0.0	0.0	0.0	0.0	0.8	0.6
	Falta de hileras		0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.2	0.2	0.2	0.2
	Total NO IMPUTABLES		*0.0*	*0.6*	*0.0*	*0.6*	*0.5*	*0.8*	*0.0*	*0.0*	*0.0*	*0.8*	*0.7*	*0.2*	*0.0*	*0.2*	*0.2*	*1.6*	*1.1*
						horas	%					horas	%			horas	%	horas	%
IMPUTABLES	Averia		4.4	4.5	0.3	9.2	7.7	6.3	0.6	4.1	0.5	11.5	9.6	10.6	1.4	12.1	10.1	32.7	22.7
	Rotura		2.5	1.1	2.0	5.6	4.6	0.0	0.0	0.0	0.2	0.2	0.2	0.0	1.2	1.2	1.0	7.0	4.8
	Salida Carrete		11.1	4.1	0.1	15.3	12.7	8.9	8.4	11.0	19.4	47.8	39.8	15.3	8.8	24.2	20.1	87.2	60.6
	Parada no codificada		5.3	6.2	7.0	18.4	15.3	10.2	7.9	6.7	10.9	35.8	29.8	7.1	8.5	15.6	13.0	69.7	48.4
	Total IMPUTABLES		*23.2*	*15.9*	*9.4*	*48.5*	*40.4*	*25.4*	*17.0*	*21.9*	*31.1*	*95.3*	*79.4*	*33.1*	*19.9*	*53.0*	*44.1*	*196.7*	*136.6*
Total Paradas			*25.1*	*16.4*	*9.7*	*51.2*	*42.7*	*27.5*	*22.5*	*29.6*	*34.1*	*113.7*	*94.7*	*38.2*	*39.5*	*77.7*	*64.8*	*242.6*	*168.5*

Fig. 1 Ejemplo de un informe.

proceso por lotes (batch) (p.ej. normalmente esto implica ejecutar la consulta por la noche). Observar que en las pequeñas organizaciones el volumen de datos es bastante pequeña para que este proceso sea viable.

3. **evitar detraer recursos de programación, mediante la disponibilidad de generadores de informes directamente manejables por los propios gerentes.**

Respecto a la implementación, el sistema realiza una utilización intensiva de la tecnología XML. Los diferentes componentes, a saber, la caché (*XCube* [13]), el modelo de características (basado en XML [3]), el formateador (*XSL* [17]) y el proceso de construcción (*Ant* [9, 16]) son descritos utilizando vocabularios XML. Esto hace que el sistema sea independiente de plataforma.

El resto del artículo se estructura de la siguiente forma. La sección 2 presenta un ejemplo. La sección 3 hace una introducción a las líneas de producto. Las secciones 4, 5 y 6 describen los diferentes aspectos de la línea de producto. La sección 7 concluye el artículo.

II. CASO DE EJEMPLO

Este trabajo has sido guiado por el sistema TRACELIA, el cual se encarga de administrar el flujo de información en plantas industriales. Los aspectos del proceso de producción incluyen el seguimiento del proceso de producción, control de inventario a nivel de máquina o el control de mantenimiento. En este escenario, el gerente puede estar interesado en *"el tiempo total transcurrido en cada estado, agrupado por categoría, para cada máquina de todas las secciones durante marzo de 2005"*, dónde *estado* se refiere a si la máquina está parada, fuera de servicio, etc. La figura 1 muestra un informe de ejemplo, dónde los estados han sido etiquetados como *"Falta de aire"*, *"Falta de agua"*, etc.

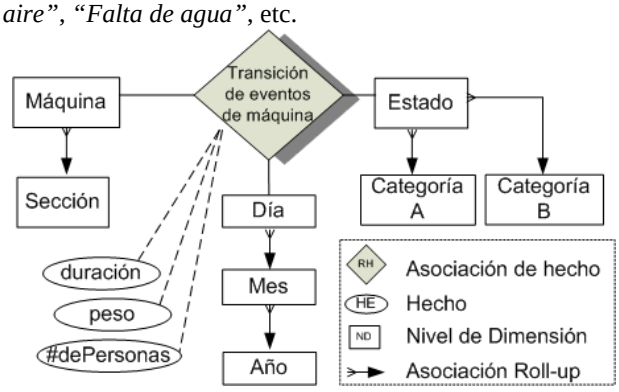

Fig. 2 Modelo ME/R para el cubo de datos de la Fig 3.

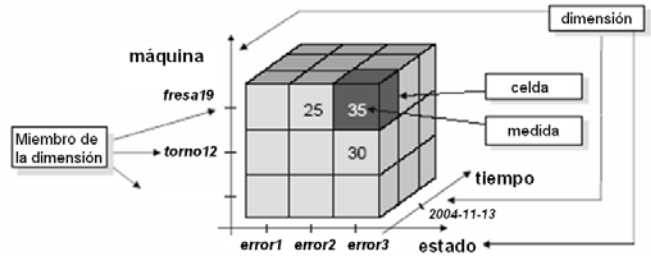

Fig. 3 Metáfora del cubo

El análisis requiere primero identificar el sujeto del análisis: el **hecho**. En nuestro caso, el hecho corresponde a las interrupciones en las que puede haber incurrido una determinada máquina. En la figura 2 esto corresponde a la asociación "transición de eventos de máquina" que liga una máquina con el estado en el que se encuentra. La transición

sería el suceso que ocasiona el que la máquina cambie de estado.

Una vez identificado el objeto de análisis, tenemos que caracterizarlo en términos de unas **medidas.** Para nuestro ejemplo, las medidas incluyen el tiempo transcurrido en un estado, el peso del producto producido o el número de personas que han participado durante este estado. La figura 2 ilustra el modelo conceptual para este ejemplo utilizando el diagrama M/ER[15]. Esta notación representa las medidas como atributos de la **asociación** "transición de eventos de máquina" (representado como un diamante grande en la figura 2).

Una vez establecido el hecho y sus medidas, indicamos las diferentes dimensiones a lo largo de las cuales se va a distribuir o desglosar este hecho. En nuestro ejemplo, estas dimensiones corresponderían a la fecha en la que se produjo la transición, la máquina afectada, y el estado o tipo de interrupción.

Estas dimensiones permiten realizar el análisis en diferentes niveles de granularidad, pudiendo dar lugar a jerarquías de inclusión entre los distintos niveles (p.ej. *año*, *mes*, *día*). Los diferentes niveles corresponden a diferentes niveles de granularidad de los datos (p.ej. cifras diarias vs. cifras mensuales) y modos de clasificación (p.ej. distribución de las máquinas a lo largo del tiempo vs. distribución de las máquinas por estado). Un nivel A se puede compactar (roll-up) en un nivel B si los hechos clasificados bajo A se pueden a su vez clasificar dentro de una misma categoría B (p.ej. el nivel *día* se compacta en el nivel *mes*; ya que las transiciones que se producen dentro de un día siempre están dentro del mismo mes).

El modelo ME/R utiliza la **asociación 'roll-up'** (compactar) para enlazar distintos niveles de granularidad para una dimensión concreta. Cada dimensión es representada por un subgrafo que empieza en el nivel correspondiente al grano más fino (p.ej. *día* para la dimensión *tiempo*).

El diseño multidimensional resumido en los párrafos anteriores, establece el espacio de análisis. El informe de la figura 1 muestra un caso concreto dónde la medida *duración* es analizada por las dimensiones *estado* y *maquina*. A menudo, para representar la vista de este dato se utiliza la **metáfora del cubo**, tal y como se muestra en la figura 3. En la metáfora, los ejes soportan las dimensiones mientras las celdas contienen las medidas.

Pero un informe son más que los datos que contiene. La maquetación del informe también es importante. Esta maquetación incluye la disposición de los datos dentro del informe (o dimensiones) y el estilo gráfico utilizado.

Por todo ello, un informe concreto como el mostrado en la figura 1, se podría caracterizar por los datos utilizados, las dimensiones y el estilo:

Informe = datos del cubo + dimensiones + estilo

Este trabajo tiene como objetivo realizar una línea de producto que obtenga generadores de informes.

III. UN RESUMEN SOBRE LINEAS DE PRODUCTO SOFTWARE

Una Línea de Producto Software (LPS) es *"un conjunto intensivo de sistemas software que comparten un conjunto común y gestionado de características (features), dónde estas características están pensadas para satisfacer las necesidades especificas de una misión o de un segmento de mercado. Asimismo, los productos son desarrollados de una forma pre-establecida a partir de un conjunto común de componentes"*. [6].

Las LPS proporcionan economías de escala. Esto significa que se pueden obtener ventajas económicas del hecho de que la mayoría de nuestros productos son muy similares, no por accidente, sino porque se ha planificado de esa forma.

Las diferencias entre posibles productos de la línea de producto pueden ser discutidas en términos de **características**. Una característica (feature) es *"una propiedad del producto que se utiliza para distinguir entre los diferentes productos dentro de una familia de productos relacionados"* [2]. Las características se organizan en **grupos de características**, formando una jerarquía de composición. Por consiguiente, una LPS soporta la variabilidad para estas características que tienden a diferenciarse de producto en producto.

Partiendo de estas características, se elaboran los componentes comunes. Estos componentes tienen que desarrollarse de forma que permitan acometer la variabilidad descrita en el modelo de características. Finalmente, el ensamblado y adaptación de los diferentes componentes se hace conforme a **un plan de producción**. Un plan de producción es *"una descripción de como los componentes comunes deben ser utilizados para desarrollar un producto de la línea de producción"* [4]. Para nuestro ejemplo, el plan de producción consiste en:

- la selección de las características deseadas para nuestro producto, es decir el informe deseado.
- la adaptación de los componentes comunes a las características seleccionadas.
- la construcción del producto mediante la ejecución del plan de producción.

La siguiente sección describe el modelo de características, los principales componentes comunes y el proceso de producción de la línea de producto para generar informes (**LPGI**).

IV. LPGI: MODELO DE CARACTERÍSTICAS

Un **modelo de características** facilita un resumen y una sintaxis concisa para expresar la parte común y variable de un dominio específico (p.ej. el generador de informes de bases de datos). Para conseguir esto, se sigue un análisis de dominio orientado a la reutilización [8, 11].

La figura 4[1] representa el modelo de características para el LPGI. Las características tienen cardinalidad [14] (p.ej.[1..*], [2..2][2]). Para generar informes de bases de datos se identifican los siguientes grupos:

- **Dimensión analítica**. Especifica las dimensiones analíticas involucradas en el problema de ejemplo. Este grupo se obtiene desde el modelo multidimensional (ver figura 2) dónde los niveles más finos de las dimensiones corresponden con las características[3].
- **CeldaDeCubo**. Esta característica incluye la medida a obtener (p.ej. *duración*, *númeroDePersonas*, *peso*) y la función de agregación a utilizar (p.ej. *suma*, *minino*, *media*, etc).
- **Estilo**. Establece el aspecto visual (*look-and-feel)* del informe. Es similar a la funcionalidad que ofrece Excel para presentar los datos en una hoja de cálculo. Las alternativas incluyen: *clásico* y *moderno.*
- **FormatoDeSalida**. Indica el formato de entrega del informe. Las alternativas incluyen: *pdf* y *xhtml*.

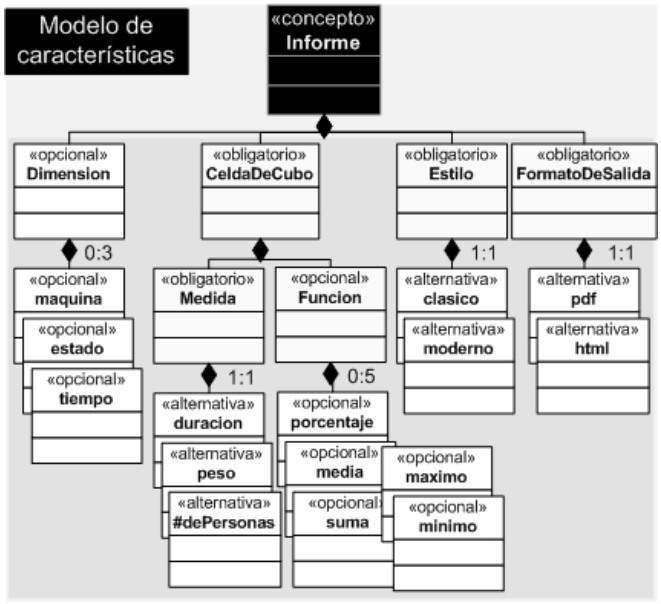

Fig. 4 Un modelo de características para informes de bases de datos utilizando la notación descrita en [5]

Un producto se singulariza por las características que exhibe. Por ejemplo, el informe mostrado en la figura 1 se caracteriza como un informe en formato *PDF* utilizando un estilo *clásico* para la visualización del hecho *duración* organizado por dos dimensiones, *estado* (*Id. de estado* y su *categoría*) y *máquina* (*Id. de máquina* y su *localización*), respectivamente. El número de diferentes productos que una LPS soporta se obtiene como la multiplicación de las variantes existentes en cada una de sus características. En nuestro ejemplo, el LPGI puede generar más de cien informes distintos.

V. LPGI: Los componentes comunes

Un **componente común (*core asset*)** es *"un artefacto o recurso que se utiliza en la producción de uno o mas productos en una línea de producto"* [6]. Estos pueden ser una arquitectura, un componente software, un proceso, un documento, o cualquier otro artefacto útil en el desarrollo de un sistema. A continuación, se introducen los principales componentes nucleares del LPGI.

A. El componente director: cuboDeDatos

Nuestra aproximación no depende de la existencia de un almacén de datos (warehouse). Más bien el análisis es guiado directamente desde la base de datos OLTP. Esto ralentizará las transacciones OLTP. Además, las bases de datos OLTP están diseñadas para optimizar transacciones OLTP. Es decir, el diseño va encaminado a agilizar las operaciones de altas, bajas y modificaciones sobre las tablas en vez de facilitar las complejas consultas necesarias para la generación de informes.

Estas observaciones justifican la necesidad de un tipo de "vistas materializadas" dónde las tablas están diseñadas para agilizar las consultas de análisis. Ya que muchos Sistemas de Gestión de Bases de Datos (SGBDs) carecen de un mecanismo de vista materializada, y con el doble objetivo de separar el SGBD del y LPGI, se define un componente común que da soporte a la vista-materializada: el componente *cuboDeDatos*.

El componente *cuboDeDatos* incluye una definición del esquema multidimensional, un proxy al SGDB, y un cubo de datos, el cuál contiene los datos. Para obtener la interoperabilidad, se utiliza la propuesta de *XCube*[4]. *XCube* [13] es un formato libre, independiente de fabricante y una familia de documentos basados en XML para guardar, intercambiar y consultar los datos de un almacén de datos (warehouse). Esta propuesta incluye *XCubeSchema* y *XCubeDimension* para la descripción del esquema multidimensional (ver figura 6), y *XCubeFacts* para describir el hecho. La figura 6 muestra un documento *XCubeFacts* para los datos del cubo de nuestro caso de ejemplo.

Utilizar XML tiene importantes ventajas, a saber, la interoperabilidad y la existencia de una amplia serie de herramientas para el procesamiento de XML. Pero también acarrea inconvenientes, en concreto el voluminoso tamaño de los documentos XML. Sin embargo, creemos que las ventajas prevalecen considerando el constante crecimiento

[1] Para una completa explicación sobre modelado de características consultar en [czarnecki00, czarnecki04, Gomaa04]. [Cechticky04] provee un vocabulario XML para la especificación de las características.

[2] Las características obligatorias y opcionales se pueden considerar casos especiales de las características con cardinalidad [1..1] y [0..1], respectivamente.

[3] Si no se selecciona ninguna dimensión, el informe es reducido a un único valor (p.ej. el total de la duración, independiente de las dimensiones maquina, tiempo y categoría de estados). Si uno de ellos es seleccionado, el informe es una línea de valores. Si más de uno son seleccionados, el informe se convierte multidimensional.

[4] http://www.xcube-open.org/

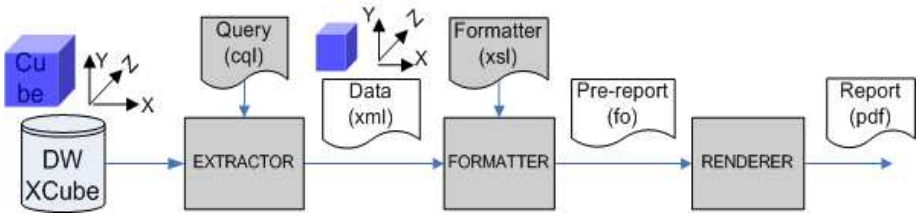

Fig. 5 Un proceso de construcción para generar informes de bases de datos

del ancho de banda de la red, el trabajar en un entorno local, y la existencia de métodos de compresión de XML cada vez más eficaces.

```
<?xml version="1.0" encoding="UTF-8"?>
<cubeFacts xmlns="http://xcube-open.org/V0_4/XCubeFacts_base.xcsd"
version="0.5">
<cube id="TransicicioDeEventosDeMaquina">
<cell>
<dimension id="maquina" node="fresa19"/>
<dimension id="estado" node="error2"/>
<dimension id="dia" node="2004-11-12"/>
<fact id="duracion" value="25"/>
</cell>
<cell>
<dimension id="maquina" node="fresa19"/>
<dimension id="estado" node="error3"/>
<dimension id="dia" node="2004-11-12"/>
<fact id="duracion" value="35"/>
</cell>
<cell>
<dimension id="maquina" node="torno12"/>
<dimension id="estado" node="error3"/>
<dimension id="dia" node="2004-11-12"/>
<fact id="duracion" value="30"/>
</cell>
</cube>
</cubeFacts>
```

Fig. 6 Los hechos del cubo de datos utilizando el vocabulario XCubeFacts

B. La hoja de estilos para el formato

El artefacto *formatter.xsl* es el encargado de definir la distribución/composición de una página y de asociar un estilo de formato a los datos. De este modo, este artefacto es utilizado para construir un primer esbozo del informe, el cuál es un documento XML llamado *pre-report.fo*. El documento *pre-report.fo* es descrito utilizando el estándar XSL-FO[5]. Este estándar es un formato independiente del dispositivo, el cuál ayuda a soportar la variabilidad en el formato de salida del informe.

C. El proceso de construcción

La figura 5 ilustra el proceso de construcción del informe:

- **Extractor**. La consulta (*query.cql*) se realiza sobre la vista materializada (*data.xcb*) para recoger solamente los datos requeridos (*data.xml*), dependiendo de las características que identifican al producto.
- **Formatter**. Este paso transforma el *cuboDeDatos* obtenido (*data.xml*) en un primer esbozo *(pre-report.fo),* aplicando para ello el transformador *formatter.xsl*.
- **Renderer**. Finalmente, este primer esbozo (*pre-report.fo*) se plasma en un formato de salida concreto.

El proceso de construcción se representa mediante un documento *Ant*. *Ant* es una herramienta basada en Java para programar procesos de construcción [9, 16]. Los programas se especifican utilizando sintaxis XML, y a menudo se llaman *build.xml*.

VI. LPGI: SOPORTANDO LA VARIABILIDAD

Los componentes comunes constituyen la plataforma o base común sobre la que se construyen los diferentes productos de la línea de producción. Seguidamente es necesario indicar cómo cada variante o alternativa dentro de una característica va a extender/adaptar esta plataforma común.

La tabla I resume cómo las características afectan a los componentes de la plataforma común. En este caso, el impacto puede conllevar el cambio de una configuración de parámetros (p.ej. la *función* de agregado a utilizar), añadir un nuevo plug-in (para la característica de *salida*) o extender las plantillas de XSLT (para la característica del *estilo*).

TABLA I
CARACTERÍSTICAS VS COMPONENTES COMUNES

	CuboDeDatos	**Formateador**	**Programa**
Dimensión			
máquina	X	X	
estado	X	X	
tiempo	X	X	
Función	X	X	
Hecho			
duración	X	X	
#dePersonas	X	X	
peso	X	X	
Estilo			
clásico		X	
moderno		X	
Salida			
pdf			X
html			X

[5] http://www.w3.org/TR/xsl/

VII. Conclusiones

Las LPS persiguen lograr una re-utilización efectiva planificando de antemano cuál va a ser la variabilidad soportada, indicando la gama de productos a ser construidos por esta "cadena de montaje". En este sentido, este trabajo muestra un ejemplo para la obtención de generadores de informes. El producto que sale de esta cadena de montaje, el informe, no se construye desde cero, sino en base a una sería de componentes comunes que constituyen la plataforma o chasis básico de esta cadena de montaje.

Este artículo muestra un caso de generación de informes de bases de datos. El sistema ha sido construido según los tres requisitos principales, a saber, no se utiliza tecnología de almacén de datos (warehouse), separar todo lo posible el generador de informes de la base de datos, y promover la generación de informes directamente por parte de los usuarios finales.

Sin embargo, la cuestión es qué pasa si el usuario final quiere un informe fuera del ámbito de la línea de producto, p.ej. con una característica que no se ha incluido en el modelo de características. Debería de decirse que el análisis de dominio que se ha llevado como parte del desarrollo de la línea de producto aspira a prever los distintos requisitos, de los cuales sólo un conjunto de ellos es seleccionado para que finalmente sean soportados por la línea de productos (p.ej. debido a razones estratégicas o de rentabilidad). Aún así, podría ser necesario modificar la línea de producto, si bien en un grado y con una frecuencia menor que con otro tipo de software. Pero incluso en este caso, la separación de asuntos que caracteriza a la arquitectura de la línea de producto facilita las extensiones.

VIII. Agradecimientos

Nuestros agradecimientos a Rosa Dunis y Unai Bergara de la empresa Mondragón Sistemas de Información, Soc. Coop. por compartir con nosotros sus percepciones acerca de la generación de informes en PYMEs, y proveer el caso de ejemplo para *Tracelia*.

IX. Referencias

[1] J. Albrecht, A. Bauer, O. Deyerling, H. Günzel, W. Hümmer, W. Lehner, and L. Schlesinger. "Management of multidimensional aggregates for efficient online analytical processing." In *IDEAS '99: Proceedings of the 1999 International Symposium on Database Engineering & Applications*, page 156, Washington, DC, USA, 1999. IEEE Computer Society.

[2] D. Batory, J.Neal Sarvela, and A. Rauschmayer. "Scaling Step-Wise Refinement." *IEEE Transactions on Software Engineering*, 30(6), pp.355–371, June 2004.

[3] V. Cechticky, A. Pasetti, O. Rohlik, and W. Schaufelberger. "Xml-based feature modelling." In *Software Reuse: Methods, Techniques and Tools: 8th International Conference, ICSR 2004, Madrid, Spain, July 5-9, 2009. Proceedings*, volume 3107 of *Lecture Notes in Computer Science*, pp. 101–114. Springer, 2004.

[4] G. Chastek and J.D. McGregor. "Guidelines for Developing a Product Line Production Plan." Software Engineering Institute, Technical report CMU/SEI-2002-TR-06, June 2002.

[5] M. Clauss. "Modelling Variability with UML." In *Proceedings of Young Researchers Workshop GCSE'01, the Third International Symposium on Generative and Component-Based Software Engineering*, En-furt,Germany, September 2001.

[6] P. Clements and L.M. Northrop. "*Software Product Lines - Practices and Patterns*." Addison-Wesley, 2001.

[7] K. Czarnecki and U. Eisenecker. "*Generative Programming*." Addison-Wesley, 2000.

[8] K. Kang et Al."Feature Oriented Domain Analysis Feasability Study." Software Engineering Institute, Technical report CMU/SEI-90-TR-21, November 1990.

[9] Apache Software Foundation. "Apache Ant." [Online] Available: http://www.ant.apache.org/.

[10] H. Gomaa. "*Designing Software Product Lines with UML*." Addison-Wesley, 2004.

[11] Martin L. Griss. "Implementing Product-Line Features with Component Reuse." In *Proceedings of the Sixth International Conference on Software Reuse*, pp. 137–152, Vienna, Austria, June 2000.

[12] A. Grohe. "Reporting Architectures", 2002. [Online] Available: "http://www.dmreview.com/article_sub.cfm?articleId=4903".

[13] W. Hümmer, A. Bauer, and G. Harde. "Xcube: Xml for data warehouses." In *DOLAP 2003, ACM Sixth International Workshop on Data Warehousing and OLAP, New Orleans, Louisiana, USA, November 7, 2003, Proceedings*, pp. 33–40. ACM, 2003.

[14] K.Czarnecki, S.Helsen, and U.W.Eisenecker. "Staged Configuration Using Feature Models." In *Software Product Lines, Third International Conference on Software Product Lines, SPLC 2004*, pp. 266–283, 2004.

[15] C. Sapia, M. Blaschka, G. Höfling, and B. Dinter. "Extending the E/R Model for the Multidimensional Paradigm." In *Int. Workshop on Data Warehousing and Data Mining (DWDM 98), Singapore*, volume 1552 of *Lecture Notes in Computer Science*, pp. 105– 116. Springer, 1998.

[16] N. Serrano and I. Ciordia. "Ant: Automating the Process of Building Applications." *IEEE Software*, 21(6), pp. 89–91, November/December 2004.

[17] W3C. "XSL eXtensible Style Language Transformations (XSLT) Working Draft Version 2.0", 2005. [Online] Available: http://www.w3.org/TR/xslt20/.

X. Biografías

Felipe I. Anfurrutia es profesor colaborador del departamento de Lenguajes y Sistemas Informáticos de la Universidad del País Vasco. Ingeniero en Informática por la Universidad del País Vasco en 1999. Desde entonces pasó a formar parte del grupo ONEKIN donde realiza actualmente su tesis doctoral. Sus intereses son la programación generativa, específicamente en la construcción de aplicaciones Web utilizando tecnologías XML. Es miembro estudiante del IEEE.

Oscar Díaz es catedrático del departamento de Lenguajes y Sistemas Informáticos de la Universidad del País Vasco. Licenciado por la Universidad del País Vasco en 1985 y doctor por la Universidad de Aberdeen en 1992, dirige actualmente el grupo ONEKIN con quince personas en el área de Ingeniería Web. Asimismo, ha participado como miembro del comité de programa de congresos internacionales (VLDB, CAiSE, Data Engineering, etc) y es autor de más de 50 publicaciones (p.ej. ACM Computing Surveys, IEEE Software, Information System Journal, ACM TOIT etc).

Salvador Trujillo disfruta de una beca FPI en el departamento de Lenguajes y Sistemas Informáticos de la Universidad del País Vasco. Ingeniero en Informática por la Universidad de Mondragón en 2002. Desde entonces pasó a formar parte del grupo ONEKIN donde realiza actualmente su tesis doctoral. Sus intereses son las líneas de producción software, específicamente como aplicar estas técnicas a las aplicaciones de Portales corporativos..