IX Congreso de Ingeniería de Organización
Gijón, 8 y 9 de septiembre de 2005

Modelado de procesos y desarrollo de sistemas software: integración entre UML y EPC

Carlos Parra Calderón[1], Rafael Ruiz-Usano[2], Paz Pérez González[1]

[1] Hospitales Universitarios "Virgen del Rocío", Sevilla, 41013 Sevilla. carlos.parra.sspa@juntadeandalucia.es, paz.perez.exts@juntadeandalucia.es
[2] Escuela Superior de Ingenieros. Isla de la Cartuja, 41092 Sevilla. usano@esi.us.es

Resumen

*Los objetivos de **UML** (Unified Modeling Language – Lenguaje Unificado de Modelado) y de la **EPC** (Event-driven Process Chain - Cadena de Procesos guiada por Eventos), están bien diferenciados: mientras que UML se enfoca al diseño de sistemas de información (SI), las EPCs se emplean para el modelado de procesos de negocios (**BPM**) dentro de la metodología ARIS. No obstante, es evidente la relación entre ambas técnicas: por una parte, un correcto diseño de SI debe basarse en los requisitos definidos por el modelo de procesos de negocio. Por otra parte, las mejoras de los procesos existentes a menudo deben llevarse a cabo mediante el desarrollo o modificación de los SI que soportan dichos procesos. En esta comunicación se analizan las posibilidades de integración de ambas técnicas de modelado.*

Palabras clave: Modelado de procesos (*Business Process Modelling*), UML, EPC.

1. Introducción

UML (Unified Modeling Language) es un lenguaje de modelado estándar en el área de desarrollo de software. Fue desarrollado por Booch et al (1998), y posteriormente aceptado como estándar en modelado orientado a objetos por Object Management Group (OMG) en noviembre de 1997, Ferdian (2001). Booch et al (1999) lo definen como un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. Proporciona una forma estándar de escribir planos de un sistema, cubriendo las partes conceptuales (funciones del sistema, y en principio también los procesos de negocios, aunque el alcance para esto no es el suficiente como veremos más adelante), y las cosas concretas (clases escritas en un lenguaje de programación específico, esquemas de bases de datos, componentes software reutilizables).

Un proceso de negocio es una colección de actividades que toma una o más clases de entradas y crea una salida que es de valor para el cliente, Hammer (1990). La captura, documentación y análisis de dichos procesos se denomina modelado de procesos de negocios (BPM – Business Process Modelling), Davis (2001). Para modelar dichos procesos hay numerosas metodologías, técnicas y herramientas. La cadena de procesos guiada por eventos (EPC) es la técnica de BPM dentro de la metodología ARIS (ARchitecture of Information Systems), Scheer (1992).

Este trabajo se deriva de la participación de sus autores en un proyecto de investigación financiado por el Fondo de Investigación Sanitario con referencia G03/117, titulado "Nuevos modelos de prestación de servicios sanitarios utilizando telemedicina"

La herramienta que recoge dicha metodología es ARIS Toolset™, de IDS Scheer. Gracias a la integración de los tres conceptos IDS Scheer es considerado el líder del sector de BPM, con 40000 licencias vendidas hasta 2002, Gartner Group (2002), y que han seguido incrementando en los últimos años, Paul y Britz-Averkamp (2005).
Loos y Allweyer (1998), Ferdian (2001), Loos y Fettke (2001) están de acuerdo en que el buen desarrollo de sistemas de información (Sis) en los negocios requiere un enfoque integrado, que incluye el diseño unificado de los procesos de negocio y el desarrollo de los SIs soportando dichos procesos. Para estos autores los procesos de negocio son modelados en un marco orientado a procesos, mientras que por otro lado, los métodos de modelado orientados a objetos sólo se usan en el aspecto de implementación software, no en procesos de negocio. Pero los dos campos están unidos, los enfoques orientados a objetos se pueden usar en el nivel de negocios para identificar objetos de un proceso y sus relaciones, aunque es importante tener en cuenta que no es un mapeado uno a uno, Eriksson y Penker (2002), sino que hay que hacer un análisis crítico del modelo de negocio para ver su aplicación para el diseño del SI específico, es decir, identificar qué tramos del proceso son automatizables.

2. Integración del modelado de procesos de negocios con EPC y el diseño de sistemas de información con UML

Con las técnicas de BPM no se puede desarrollar software directamente como con UML, y UML no cubre todos los aspectos necesarios para el BPM, Loos y Allweyer (1998). Por esto, autores como Eriksson y Penker (2002) han creado una extensión de UML para BPM (Extensión de negocios de Eriksson y Penker), pero aún así no tiene el alcance de la EPC, ya que ésta en su forma extendida integra las vistas de funciones, datos, organización en un único tipo diagrama, y la extensión de Erikkson y Penker se basa en todos los diagramas de comportamiento. De hecho, Kim et al (2003) cuestiona la validez de los intentos de extensiones de UML para el BPM. Por otro lado, desde la EPC no se puede generar código como con UML, Scheer (2002a). Pero la cuestión es saber en qué punto se separan o se unen estas dos técnicas. Muchos autores han estudiado la relación entre ellas mostrando distintos enfoques para integrarlas. Del análisis de estos estudios se concluye que hay varios tipos de relaciones entre los diagramas de UML y la EPC (Tabla 1). Hemos definido estos tipos de relaciones de la siguiente forma:
Comparación: diferencias y analogías entre los diagramas
Transformación: traducción de los diagramas para tenerlos en los dos tipos de formatos
Integración: diagrama resultante de la unión de las técnicas (diagrama con objetos de la EPC y objetos de UML)
Combinación: uso de los diagramas UML y la EPC complementariamente que da una visión desde todos los niveles (para tener la descripción del proceso de negocio, y la información necesaria para desarrollar el sistema de dicho negocio)

DE: diagrama de estado
DA: diagrama de actividades
DC: diagrama de clases
DCU: diagrama de casos de uso
DS y DCol: diagrama de secuencia y colaboración

Tabla 1. Relaciones entre la EPC y los diagramas UML de las referencias más significativas

	Comparación	Transformación	Integración	Combinación
Loos y Allweyer (1998)	EPC - diagramas comportamiento	EPC ➔ DE EPC ➔ DA		EPC - DCU EPC - DC

Nüttgens et al (1998)		EPC → DC EPC → DA EPC → DCU	OEPC	
Hahn et al (1998)		EPC → DA		
Ferdian (2001)	EPC - DA	EPC ↔ DA	oEPC	
Loos y Fettke (2001)	EPC - DA	EPC → DE EPC → DA		EPC - DCU EPC - DC EPC - DS y Col
Söderström et al (2002)	EPC - DE			
Scheer (2002a)			EPC - objetos UML	EPC - diagramas UML
Scheer (2002b)		EPC ↔ DE		
Scheer (2003)			EPC-objetos UML	EPC - diagramas UML
Knott et al (2003)	EPC - DA EPC - DS EPC - DE			

En los siguientes sub-apartados se presenta una breve explicación de cada propuesta.

2.1 Comparación

El primer estudio sobre la relación entre los diagramas de UML y la EPC fue de Loos y Allweyer (1998), que más tarde se completó en Loos y Fettke (2001). De la comparación de los diagramas de comportamiento con la EPC estos autores afirman que los diagramas de comportamiento sólo cubren algunos aspectos del proceso que describen, como los detalles técnicos, pero no el flujo total del proceso, ni todos los aspectos organizacionales. Se centran en la comparación del DA con la EPC, identificando problemas en el uso del DA en BPM:

- Todos los operadores lógicos no pueden modelarse de una forma directa para dividir y unir el flujo de control, como con la EPC. De hecho no hay equivalente en el DA para el operador O de la EPC.
- Las unidades organizativas responsables de cada actividad se expresan mediante swimlanes, sin embargo no es información suficiente, ya que no muestran todas las relaciones organizativas como “es responsable para”, “proporciona soporte para”, “debe ser informado sobre el resultado de”, o “debe aprobarse”.
- No hay soporte para modelado de información adicional como la disponible en la EPC, como portadores de información, recursos materiales u otro tipo de recurso.

En conclusión, estos autores observan que la EPC es más completa que el DA a la hora de representar un modelo de proceso de negocios.

Ferdian (2001), basándose en el estudio previo de Loos, va más allá haciendo un análisis exhaustivo para comparar el DA con la EPC, identificando algunos aspectos desde los que podemos ver las correspondencias y diferencias entre las dos técnicas, principalmente son:

- Contexto.
- Exactitud/ ambigüedad.
- Notación/ terminología. Ambos diagramas tienen conceptos similares (tales como dividir/ unir, rama/ fusión, actividad atómica/ extendida, etc) pero son representadas usando notación y terminología distintas. Algunas notaciones no tienen un homólogo en el otro diagrama. Esto indica las diferencias semánticas entre ellos.

Así, este autor describe en una tabla cada uno de los aspectos para cada tipo de diagrama.

Tabla 2. Comparación entre EPC y DA, Ferdian (2001)

	EPC	Diagrama de Actividades de UML
Contexto	Modelado orientado a procesos (orientado a negocios)	Modelado orientado a objetos (orientado a TI)
Exactitud/ ambigüedad	Eventos iniciales en cualquier punto del proceso, operadores lógicos, bucles, puntos muertos	
Notación/ Terminología		
Elemento activo	Función	Actividad/ Estado de acción
Elemento pasivo	Evento	
Cadena de proceso	Flujo de control	Transición
Conectores lógicos		
Rama/ combinación	O exclusivo	Diamante de decisión
Bifurcación/ Unión	Y	Barra de sincronización
O inclusivo	O	
Actor	Unidad organizativa	Swimlane
Iteración		'*' Signo de Multiplicidad

A partir de este análisis, Ferdian (2001) crea una metodología de transformación entre estos tipos de diagramas, que se verá en el siguiente apartado.
Dentro de un marco de comparación de técnicas de modelado de procesos, entre las que se encuentran la EPC y el DE de UML, Söderström et al (2002) proponen el uso de un metamodelo ante la ambigüedad y los diferentes usos y definiciones de los conceptos de las técnicas, ya que esto dificulta la comparación entre éstas y la integración de modelos de procesos. Para estos autores el uso del metamodelo hace más sencilla la comparación, y proporciona una ilustración gráfica de los conceptos básicos y sus relaciones, que es fácil de entender incluso para no expertos. El metamodelo está detallado en la referencia Söderström et al (2002). Las conclusiones que obtienen son:

Tabla 4. Comparación entre la EPC y el DE, Söderström et al (2002)

Concepto	EPC	Diagrama de Estados de UML
Evento	Usa 2 tipos de eventos simples: 1)Evento pre-actividad 2)Evento post-actividad	Usa 7 tipos de eventos simples o combinados
Estado		Concepto de "estado" y "estado de espera"
Actividad	Concepto de "función" de EPC	Conceptos de "acción de entrada", "hacer actividad" y "acción de salida"
Proceso	Conjunto parcialmente ordenado de "funciones" lógicamente dependientes	Conjunto parcialmente ordenado de "acciones/ actividades" lógicamente dependientes
Reglas	Conjunto de pre- y post- condiciones (eventos) que deben verificarse antes y después de una función	Conjunto de condiciones en las "transiciones"
Recurso	Concepto de "recurso" de EPC	Concepto de "clase"
Actor		Concepto de "objeto" capaz de producir un "evento"

Una comparación menos detallada es la presentada por Knott et al (2003), en la que estudian las ventajas y desventajas de UML y las EPC de ARIS. Las conclusiones que obtienen son:

Tabla 3. Comparación entre EPC y los diagramas UML, Knott et al (2003)

Enfoque	Teoría subyacente	Ventajas	Desventajas
EPC ARIS	Redes de Petri	Muy popular en Europa, soportada por herramientas ARIS, método fácil y comprensivo para expertos del dominio	Débil relación con las técnicas de desarrollo de software posterior, análisis lento, baja expresividad para grandes modelos
Diagrama de actividades de UML	Cadenas de flujos	Estándar industrial, soportado por muchas herramientas CASE	Demasiada orientación a software, difícil de entender por expertos de dominio
Diagrama de secuencia y de estados	Máquinas de estado	Estándar industrial, soportado por muchas herramientas CASE	Demasiada orientación a software, difícil de entender por expertos de dominio

2.2 Transformación

Para Loos y Allweyer (1998) y Loos y Fettke (2001) el único diagrama de UML que se puede transformar de forma directa a EPC es el DA, pero no propone ninguna forma para llevarla a cabo. Detecta algunos problemas en la transformación que más tarde Ferdian (2001) describirá más explícitamente. Además no consideran relevante la transformación de EPC a DCU por la falta del flujo de control de éstos que hacen perder la visión sobre la secuencialidad de las acciones, por tanto la relación entre estos diagramas está basada en la combinación (apartado 2.4). Estos autores proponen la transformación de la EPC en DE, introduciendo en la EPC el objeto estado con conexiones de entrada y salida a las funciones correspondientes, y cada objeto estado de la EPC se transforma en un estado del DE. Las funciones de la EPC se corresponden con transiciones de estados.

Para Nüttgens et al (1998) la mayoría de diagramas EPC también pueden transformarse de forma más o menos sencilla en DA, sin tener en cuenta que los objetivos de modelado con cada uno de ellos es diferente. Hacen la siguiente correspondencia sin detallar: las funciones de la EPC con las actividades del DA, los eventos con los estados, y los operadores de enlace con las transiciones sincronizadas o divididas. Proponen además la transformación de EPC en DCU, pero no tiene en cuenta que éstos tienen distinto alcance que los DA, por lo que no tendría sentido transformar los dos tipos de diagramas de UML en EPC (se puede suponer que dependerá del nivel de detalle de la EPC). Para ello consideran cada función en la que el sistema esté implicado como un caso de uso, y las unidades organizacionales implicadas como los actores. Además proponen que, ante la gran cantidad de información que dan los procesos sobre objetos, sus estructuras y sus relaciones, las clases pueden derivarse desde los objetos de información y las funciones de la EPC.

Otro enfoque de transformación de la EPC en DA lo realizan Hahn et al (1998), sin considerar los swimlanes. Identifican algunos problemas, fundamentalmente la falta de información del DA (eventos y objetos de información) a la hora de crear la EPC, por lo que resulta incompleta. Los autores sugieren algunas posibles soluciones, como omitir los eventos finales de las funciones en la EPC, y proponen una extensión de UML mediante barras de transición etiquetadas con el tipo de operador para diferenciarlos al hacer la transformación, representar

la regla O exclusivo mediante operadores de decisión, y denotar un subproceso de una función de la EPC mediante un asterisco en la actividad correspondiente, entre otras.
Ferdian (2001) explica la transformación directa de EPC a DA, y viceversa, usando los objetos correspondientes del cuadro de comparación (tabla 2). Da una serie de pasos para hacer la transformación e introduce también la de sentido contrario, observando que se pierde información al crear el DA a partir de la EPC.
Con respecto a la transformación de EPC a DE, y viceversa, Scheer (2002b) propone una metodología mediante la integración de dos herramientas, ARIS para el modelado de la EPC y Business Ware para los DE. La transformación se hace por objetos según la siguiente tabla:

Tabla 4. Transformación EPC-diagramas de estado UML, Scheer (2002b)

ARIS	Business Ware
eEPC (o eEPC en columnas o filas)	Modelo de proceso
Función con subproceso	Estado anidado
Función con sistema de aplicación asignado	Estado normal
Función con unidad organizacional asignada	Estado de actividad
Función con sistema de aplicación y unidad organizativa asignados	Estado de actividad
Otra Función	Estado de acción
Evento inicial	Evento sobre transición de estado inicial
Evento final	Evento sobre transición de estado final
Conector	Transición
Regla Y/ O de apertura	División
Regla Oexcl de apertura	Estado de acción
Regla Y de cierre	Unión
Regla O/ Oexcl de cierre	Estado de acción

2.3 Integración

El propósito del concepto de EPC orientado a objetos, oEPC, según Nüttgens et al (1998) es, por un lado preservar el potencial y la aceptación del usuario final del método estándar EPC, y por otro lado la integración de métodos orientados a objetos. Junto con Ferdian (2001), que también apuesta por el modelado de objetos de negocios y de procesos de negocios con la oEPC, describen los objetos de negocios y sus relaciones dentro de un proceso en la oEPC, integrando así la técnica EPC con objetos UML, ya que para el BPM la interacción entre objetos tiene lugar por medio de intercambio de mensajes, denotado como flujo de control guiado por eventos, y los objetos de negocios pueden agruparse dentro de clases, cada clase de objeto de negocio tiene atributos y métodos o funciones.
BPM con ARIS HOBE (House of Business Engineering of Architecture of Information Systems), y el diseño de sistemas con RUP (Proceso Unificado de Rational), son las metodologías en las que están basadas el enfoque de Scheer (2002a), describiendo qué información se requiere para el análisis y el diseño de SIs, y cómo un proceso puede dar ese diseño. Fundamentalmente propone añadir información a los modelos de procesos, incluyendo objetos orientados a UML, para generar modelos UML directamente desde modelos de procesos (como proponen Loos y Allweyer (1998), Loos y Fettke (2001) con la introducción de los objetos del DC en la EPC). Describe la forma de uso combinado de la herramienta ARIS Toolset™ con Rational para obtener la mejor forma posible de soportar proyectos de diseño de SIs, integrando las metodologías. Más tarde, Scheer (2003) presenta ARIS UML Designer, herramienta que integra las metodologías añadiendo los objetos de UML a los diagramas del modelado de procesos, incluyendo a la EPC.

2.4 Combinación

Loos y Allweyer (1998) y Loos y Fettke (2001) proponen la combinación de la EPC con los DC, DCU, y con DS y DCol. Para la combinación con los DC, explican que por medio de la conexión de EPCs y éstos es posible definir qué clase de objetos se requieren, crean, o cambian por una función, y qué operaciones y atributos son necesarios. Su enfoque se basa en, dado un DC, usar sus objetos en la EPC de forma que mantenga la lógica según unos niveles de detalles dados (ver referencias Loos y Allweyer (1998), Loos y Fettke (2001)), teniendo en cuenta las relaciones entre los objetos. Por lo que se concluye que son necesarios los dos tipos de diagramas para tener la información completa.
En relación a los DCU, estos autores afirman que solo es posible el uso combinado de éste con la EPC, ya que tienen distintos alcances, los casos de uso normalmente no representan funciones de negocios, pero interaccionan con un SI, y una función de la EPC puede contener varios casos de uso, por lo tanto es posible detallarla con un DCU. Por otra parte, proponen definir los casos de uso de forma más completa, con la ayuda de una EPC, sustituyendo la descripción escrita por la gráfica con la EPC.
Para los DS y DCol, Loos y Fettke (2001) afirman que no es razonable transformar las EPCs en estos directamente, aunque haya algunas redundancias en ambos diagramas, ya que principalmente están conectados por medio del uso de los mismos elementos. Por lo tanto afirma que es responsabilidad del modelador asegurar que el intercambio de mensajes representado en un DS o DCol sean consistentes con los procesos que van a soportar.
El proceso general de Scheer (2002a), combinando la EPC y los diagramas UML, y basándose en la metodología RUP para describir el diseño de sistemas, es:
- Análisis y diseño de los procesos de negocio usando modelos de alto nivel y EPCs.
- Crear un diseño detallado para transformar modelos EPCs en eEPCs.
- Relacionar las clases y los paquetes UML, así como otros objetos de ARIS, a tramos del proceso para mostrar la integración con sistemas y aplicaciones.
- Esos objetos formarán el concepto básico del nuevo sistema en clases UML y diagramas de componentes.
- Las tareas específicas se describirán con detalle sobre el nivel de interacción de usuario en la jerarquía del proceso.
- Los atributos UML pueden mostrar información de entrada y salida; las operaciones UML muestran reacciones del sistema en la actividad del usuario.
- El diseño puede completarse añadiendo otros objetos relevantes como roles, posiciones, estados de objetos, etc; esos objetos tienen importancia para desarrollar el mecanismo de control, conceptos de autorización, o modelos de flujo de trabajo.
- Copiar o transformar los objetos relevantes del diseño de proceso en clases UML, componentes, actividades o DE.
- Esos objetos pueden usarse para desarrollar el modelo base para la siguiente fase de diseño.
- El diseño de sistema y proceso se transformará en varias formas de documentación que construye la definición de requisitos.

El problema del enfoque de Scheer (2002a) es que, aunque dice qué hay que hacer, no describe la forma de llevar a cabo el proceso.

3. Integración de BPM y desarrollo de sistemas con otras técnicas

Otros autores han estudiado la integración del BPM y el desarrollo de sistemas con otras técnicas, aunque básicamente para este trabajo se han analizado aquellas que usan UML, ya que es lo más extendido. De dicho análisis destacamos McSheffry (2001) y Kim et al (2003).

McSheffry (2001) transforma cadenas de procesos en DCU y DA, partiendo de que toma el concepto de proceso como intrínsecamente jerárquico, desde el proceso de negocio de la empresa, hasta cada proceso que realiza un empleado (en el sentido de lo que usualmente llamamos actividad o función), agrupando los procesos de bajo nivel que tienen una meta común. Para evitar modelar en un grado de detalle inapropiado propone identificar en el nivel más bajo del proceso los llamados Procesos de Negocios Elemental (Elemental Business Process, EBP), definiéndolos como " Un proceso realizado por una persona en un lugar en un período de tiempo, que aporta un valor significante y que deja los datos en un estado consistente". McSheffry (2001) toma el concepto EBP para buscar la relación entre los DCU y las cadenas de procesos, asignando un caso de uso con cada EBP, y detallando los caso de uso si se desea, para que la relación con el soporte EBP sea clara.
Kim et al (2003) hacen otra aportación comparando UML con IDEF, y una de las conclusiones principales a las que llegan es que, en general, no puede ser definido explícitamente un mapeado uno a uno entre los elementos de los distintos tipos de modelos, ya que la vista del modelador puede ser diferente dependiendo del propósito del modelo, por lo tanto las semánticas no serán necesariamente las mismas. Sin embargo, el mapeado entre algunas partes específicas del algunos tipos de modelos pueden ser explícitas y representarse consistentemente. Por lo que presenta una metodología de trabajo, modelando el proceso de negocio con IDEF, y posteriormente, modelando con UML utilizando toda la información de los modelos de IDEF para ello, explicando detalladamente como hacerlo (indicando los diagramas, elementos y relaciones útiles de IDEF para crear los diagramas de UML).

4. Conclusiones

El BPM es una herramienta útil para hacer una toma de requisitos en un proyecto de reingeniería o mejora de un proceso de una empresa. Para mejorar dicho proceso podemos hacerlo de varias formas, automatizándolo, es decir, no variar el proceso y automatizar las funciones que lo forman; o cambiando el proceso, de forma radical (reingeniería) o sólo realizando mejoras puntuales. Para cada tipo de mejora la toma de requisitos es distinta, ya que no es lo mismo si el objetivo es desarrollar software, integrar un software estándar, o integrar aplicaciones heredadas tanto para automatizar el proceso completo como tramos del mismo. Sea cual sea el objetivo, el modelado del proceso es fundamental, sirve para conocerlo, analizarlo y poder así proceder a su mejora, creando dos modelos, el "as is" (que representa el proceso antes de la mejora), y posteriormente el "to be"(modelo de cómo será el proceso después de mejorarlo).
Si el objetivo es desarrollar un SI personalizado es necesario añadir a la toma de requisitos el diseño mediante diagramas UML, que no sólo documentan el sistema, sino que herramientas como Rational transforman directamente en código.
A partir de este análisis se plantea encontrar el límite existente entre el BPM y el diseño de sistemas, para crear una forma óptima de trabajar, sin duplicar información y tener una visión completa de las necesidades del proceso y mejorarlo mediante un SI. La gran variedad de relaciones entre las técnicas de BPM con las de modelado de sistemas que presentan los autores hace pensar que no es algo sencillo de delimitar. El enfoque de Loos y Allweyer (1998) fue el primer trabajo presentado en el que se discute este tema con la EPC y los diagramas de UML. Es el más completo (sobre todo a partir del segundo trabajo de Loos y Fettke (2001)) por que tiene en cuenta todos los diagramas de UML, pero no da una solución directa a la forma de trabajo para diseñar el sistema con base en el BPM. El trabajo de comparación de Knott et al (2003) nos da una visión clara de la disyunción entre las dos metodologías (la orientación a procesos de la EPC y la orientación a software de UML), y una de las conclusiones básicas que obtenemos de su análisis es el papel de los expertos del

dominio en el modelado, ya que los modelos de los procesos son fácilmente entendibles por éstos, y los modelos de software no.
Con respecto a las distintas relaciones analizadas, la comparación nos ayuda a estudiar el alcance de ambas formas de modelado, y la posibilidades de ampliación de alguna de ellas para cubrir todos los aspectos necesarios para el buen desarrollo de los SIs. Según Kim et al (2003), la transformación de cualquier tipo de técnica de BPM en otra de diseño de software no tendría sentido, ya los alcances de ambas técnicas son diferentes, por lo tanto transformar diagramas UML a EPC tampoco, pues el modelado de procesos con la EPC sería un paso previo al modelado con UML. Plantear la transformación contraria, desde la EPC a algún diagrama UML tampoco sería válida, ya que no se cubren todos los aspectos necesarios para una documentación completa del sistema a desarrollar. Con respecto a la combinación y la integración, por independiente no se pueden considerar válidas, la combinación de las técnicas deja aspectos que la integración puede cubrir, es decir, usar la EPC y todos los diagramas de UML necesarios para un trabajo puede ser factible, pero si además en la EPC se incluyen datos que puedan reaprovecharse en los diagramas UML (integración de objetos de UML en la EPC), se hace una toma de requisitos muy completa para que los analistas creen los diagramas de UML de una forma sencilla y directa.
En conclusión, la cuestión está en facilitar el modelado del sistema a desarrollar mediante el BPM. El problema está en cómo hacer esto. Scheer (2002a) nos da una primera idea, sistematizando la forma de trabajar con las metodologías ARIS y UML, basándose en RUP, y usando la integración de las técnicas, al introducir objetos de UML en la EPC, y la combinación de los diagramas, para conseguir completitud. Desde nuestro punto de vista es la forma más lógica de trabajar, pero sería necesaria una guía detallada y un metamodelo definido para esta sistematización, como la que presentan Kim et al (2003) para IDEF con UML, que nosotros desechamos ante la complejidad de los modelos de IDEF.
Las ideas fundamentales en las que se basarán la guía y el metamodelo son las propuestas por Loos y Allweyer (1998), Loos y Fettke (2001) y Scheer (2002a).
Los puntos fundamentales para analizar son:

- El nivel de detalle, que vendrá dado por el tipo de proyecto (desarrollo de software aplicado al rediseño o mejora), y del nivel de automatización del proceso de partida (puede estar parcialmente automatizado).
- Los diagramas de ARIS, sus objetos y relaciones entre éstos, necesarios para tener un análisis completo del proceso en el modelo “to be”, y así obtener la mayor información que se reaproveche en los diagramas de UML, identificando exactamente dicha información en los objetos de los distintos diagramas de UML.
- Diagramas de UML necesarios según el tipo de proyecto y del nivel de automatización para evitar información redundante.

Por todo ello, consideramos que ARIS Toolset es la herramienta adecuada, ya que nos proporciona los medios necesarios para poder trabajar con las propuestas de Loos y Allweyer (1998), Loos y Fettke (2001) y Scheer (2002a), y más autores de los analizados, ya que proporciona los diagramas de UML y la eEPC, además identifica las operaciones de las clases con funciones, es decir, son los mismos tipos de objetos, por lo tanto, viendo las relaciones de los objetos clase se identifican automáticamente las operaciones de dicha clase; identifica los casos de uso con las funciones también, y los actores con los distintos tipos de objetos organizacionales. Por lo que ARIS nos puede ayudar a integrar las técnicas y combinarlas.

Referencias

Booch, Grady, Jacobson, Ivar, and Rumbaugh, James. (1998). El Lenguaje Unificado de Modelado. Manual de Referencia. Addison Wesley, Madrid.

Booch, Grady, Jacobson, Ivar, and Rumbaugh, James. (1999). El Lenguaje Unificado de Modelado. Addison Wesley, Madrid.
Davis, Rob. (2001). Business Process Modelling With ARIS: A Practical Guide. Springer Verlag
Eriksson, Hans-Erik and Penker, Magnus, (2002). Business Modeling with UML. Report.
Ferdian. (2001). A comparison of Event-driven Process Chains and UML Activity Diagram of Denoting Business Processes. Thesis/Dissertation, Technische Universität Hamburg-Harburg.
Gartner Group. (2002) Gartner Group. http://www.gartner.com.
Hahn, E. v.; Paech, B.; Bock, C. (1998) Reengineering Conventional Data and Process Models with Business Object Models: A Case Study Based on SAP R/3 and UML. Lecture Notes in Computer Science, Vol 1507, pp. 393- 406.
Hammer, M. (1990) Reengineering work: Don't automate, obliterate. Harvard Business Review, Vol 68, No.4, pp. 104- 112.
Kim, C.-H. et al. (2003) The complementary use of IDEF and UML modelling approaches. Computers in Industry, Vol 50, No.1, pp. 35- 56.
Knott, Roger P., Merunka, Vojtech, and Polak, Jiri (2003). The role of object-oriented process modeling in requirements engineering phase of information systems development, EFITA 2003 Conference, Debrecen, Hungary.
Loos, Peter and Allweyer, T (1998). Process orientation and object-orientation: an approach for integrating UML and Event-Driven Process Chains (EPC), in Kobryn, C., Atkinson, C., and Milosevic, Z. (Eds.), Enterprise Distributed Object Computing (2nd International Workshop EDOC'98, La Jolla, California, U.S.A.), (IEEE Service) Piscataway.
Loos, Peter and Fettke, Peter (2001). Towards an Integration of Business Process Modeling and Object-Oriented Software Development, in Rosca, I. and Ivan, I. (Eds.), Information Society-Proceeding of the Fifth International Symposium on Economic Informatics (IE 2001), Bucharest, Romania.
McSheffry, Eugene. (2001). Integrating Business Process Models with UML System Models. Thesis/Dissertation, Popkin Software.
Nüttgens, M, Feld, T, and Zimmermann, V (1998). Business Process Modeling with EPC and UML. Transformation or Integration?, in Schader, M. and Korthaus, A. (Eds.), Workshop des Arbeitskreises "Grundlagen objektorientierter Modellierung" (GROOM) der GI-Fachgruppe 2.1.9 ("Objektorientierte Softwareentwicklung").
Paul, S. ; Britz-Averkamp, I. (2005) IDS Scheer Increases Volume of License Sales as well Total Revenue by 27% – Dividend to Increase to 0.16 EUR – Increase of at least 200 New Employees in 2005. www.ids-scheer.com.
Scheer, August-Wilhelm. (1992). Architecture of Integrated Information Systems. Springer Verlag, Berlin.
Scheer, August-Wilhelm, (2002a). ARIS and UML - System Design with ARIS HOBE and Rational Unified Process. Report.
Scheer, August-Wilhelm, (2002b). ARIS Integrator for Vitria - Process-oriented Integration of IT-Applications. Report.
Scheer, August-Wilhelm, (2003). ARIS UML Designer - Business-Driven Software Engineering. Report.
Söderström, Eva, Anderson, Birger, Johannesson, Paul, Perjons, Erik, and Wangler, Benkt (2002). Towards a Framework for Comparing Process Modelling Languages, in Banks Pidduck, A., Mylopoulos, J., Woo, C. C., and Tamer Ozsu, M. (Eds.), 14th International Conference, CAiSE 2002.