E2OL: Sistema de Planeamiento y Scheduling Personalizable e Integrable con ERPs

Vidoni, Melina C. – Vecchietti, Aldo R.
Instituto de Desarrollo y Diseño, INGAR UTN-CONICET

Abstract

Los sistemas conocidos como Advanced Planning and Scheduling (APS) surgieron con el objetivo de proveer funcionalidades avanzadas en el planeamiento y scheduling, intentando mejorar las capacidades que los ERP (Enterprise Resource Planning) poseen en esa área. Se han realizado muchas investigaciones pertinentes, pero aquellas que realmente lograron implementarse, fueron llevadas a cabo para empresas particulares, o como "aplicaciones empaquetadas" que utilizan modelos matemáticos genéricos para su funcionamiento. La investigación presentada en este trabajo propone un sistema denominado E2OL (ERP to Optimizer Linkage), que posee las características de los APS, pero permitiendo que la empresa que desee implementarlo pueda utilizar sus propios modelos matemáticos, a su vez de integrarlo con el sistema empresarial que utilice. E2OL no se ata a un ERP, sistema de gestión de bases de datos, optimizador o modelos matemáticos, habilitando la modificación controlada del mismo y su adaptabilidad a diversos entornos; se trata de un sistema desarrollado en Java, portable y amigable al usuario que permite considerar las características más particulares de cada negocio donde se implemente. E2OL está acompañado por su sistema complementario E2OL Configurer, el cual presenta todas las herramientas necesarias para adaptar a E2OL -de forma guiada y transparente- a la empresa donde se desee implementar, sin la necesidad de modificar el código, o invertir tiempo en la adaptación del mismo.

Palabras Clave

ERP, integración, bases de datos, gestión, APS.

Introducción

Desde que los sistemas Enterprise Resource Planning (ERP) surgieron en el mercado de los sistemas empresariales genéricos, se han convertido en el estándar *de-facto* para obtener una plataforma integrada que administre el negocio de la empresa, favoreciendo la gestión en tiempo real de la información [1]. Estos sistemas son "aplicaciones empaquetadas" pensadas para funcionar en la mayoría de las organizaciones, pero siempre necesitan cierto grado de personalización para poder funcionar adecuadamente en el negocio de la empresa donde se está implantando [2]. Sin embargo, implantar efectivamente estos sistemas en una organización no es una tarea sencilla, ya que se presentan varios desafíos para las organizaciones [3]; la mayoría de estos se encuentran en las primeras etapas de implantación, o son causados por una mala estimación de costos [4].

En lo que respecta a las empresas manufactureras industriales que deseen incorporar sistemas ERP, un punto clave a considerar es la administración del planeamiento y "scheduling" de la producción, lo que puede ser un determinante en la selección del ERP. Ya en el año 1993, McCarthy y Liu [5] hablaban de una "brecha" entre la teoría y la práctica del "scheduling", en lo que respecta a sistemas empresariales. Por su parte, en [6] los autores plantean las principales limitaciones de los ERP, asociados con su módulo central de MRP (Material Requirements Planning): trabajan con la asunción de recursos de capacidad limitada y no son capaces de considerar ubicaciones múltiples. Por esto mismo, muchas empresas han adoptado un nuevo tipo de sistema, denominado Advanced Planning and Scheduling (por sus siglas, APS), con el objetivo de mejorar el soporte en las actividades de planeamiento y "scheduling" [7] industrial.

En [8], el autor define a los APS como: *"[…] un programa de computadora que emplea algoritmos matemáticos o lógica para llevar a cabo la optimización o simulación sobre schedulings de capacidad finita […]"*. Dicho autor también menciona que los APS se integran con la cadena de suministro y con los sistemas de la empresa que los implanta, con el objeto de extraer de ellos la información pertinente para realizar el "scheduling". Estos sistemas habilitan a las empresas generar optimizaciones de acuerdo a sus objetivos

estratégicos y crear planes que satisfagan múltiples metas. A diferencia de los ERP, los APS buscan propuestas factibles y lo más cercanos al óptimo posible, al mismo tiempo que consideran -de forma explícita- los potenciales cuellos de botella.

Considerando esta problemática, se han desarrollado varios trabajos en el área. Los autores de [9], ya en el año 1993, presentaron un sistema de "scheduling" a corto plazo desarrollado específicamente para una empresa. Por su parte [10] presenta un modelo para un APS basado en algoritmos genéticos, que trabaja con órdenes de venta con fecha de entrega y que permite considerar tercerización. En [11] se propuso un modelo combinatorio, posteriormente implementado en C++, para integrar sistemas MRP con "scheduling" tipo shop-floor en ambientes discretos. Por otro lado, en [12] se desarrolló un modelo para integrar el "scheduling" m-máquina en sistemas MRP, pero no llegaron a implementar un sistema que emplee dicho modelo. Otro trabajo relacionado es el de [13], donde los autores desarrollaron un módulo ARP (Advanced Resource Planning) el cual provee un proceso de selección de parámetros con la meta de proveer información realista para el "scheduling"; también implementaron este módulo en varias empresas.

Sin embargo, la mayoría de las propuestas (e implementaciones) en el área, se limitan a una organización o producción en particular [14]. A su vez, algunas propuestas consisten en módulos planteados como "genéricos" que, si bien tienen características avanzadas respecto a los ERP, utilizan modelos matemáticos (o algoritmos genéricos) que no permiten considerar cuestiones particulares del negocio de la empresa donde se va a implementar.

Como consecuencia, la propuesta de este trabajo es un sistema denominado *E2OL* (siglas de ERP to Optimizer Linkage), el cual posee las características de un APS, pero que puede ser personalizado por la empresa que desee adoptarlo, con el objetivo de integrarse con diversos ERP y sistemas de gestión de bases de datos (SGBD), dándoles la capacidad de emplear modelos matemáticos específicos de la empresa, que contemplen las particularidades de las mismas. La adaptación del sistema se hace de forma transparente y controlada mediante su sistema complementario, denominado *E2OL Configurer*.

1. Especificación de Requerimientos

El objetivo principal de este trabajo es proponer la arquitectura de un APS, con la característica que sea personalizable y adaptable al sistema de información de una empresa industrial. Dicha "customization" debe ser adaptativa [15], es decir, tratarse de un producto estándar que los clientes puedan fácilmente reconfigurar para acomodarlo a sus necesidades, sin interacción directa con quien desarrolló el sistema.

De este modo, el primer paso para definir el alcance del sistema, fue analizar los requerimientos funcionales del mismo.

1.1 Requerimientos Funcionales

Los requerimientos funcionales de un sistema de software son comúnmente elicitados de los stakeholders del mismo [16]. Para un sistema hecho a medida, desarrollado a pedido, los mismos autores de [15] definen a los stakeholders como *"[…] personas u organizaciones que influencian los requerimientos de un sistema o que son impactados por el mismo […]"*. Sin embargo, esta situación cambia para los productos de software "empaquetados": dado que no existe una persona particular interesada en el sistema, deben analizarse las necesidades de los potenciales clientes y, a partir de estas, elicitar los requerimientos. En este caso, los trabajos académicos realizados en el área también son stakeholders del sistema, desde los cuales pueden extraerse los requerimientos.

Desde la detección de la problemática del planeamiento y "scheduling" en los sistemas ERP, se han escrito muchos trabajos de investigación, abarcando diversas propuestas y análisis. En [17], Framinan y Ruiz hicieron una revisión de muchas contribuciones, identificando un número de funcionalidades que -según la bibliografía que estudiaron-, debían ser cubiertas por un sistema de estas características. Tomando este análisis como base, se seleccionaron aquellos requerimientos -mayormente centrados en la funciones de planeamiento y "scheduling"- que serían implementados por *E2OL*:

- Planeamiento de la producción: se puede realizar optimización de muchas áreas o funciones dentro de una organización, sin embargo, el sistema debe centrarse en optimizar los recursos y el "scheduling" de la producción de bienes o servicios.

Para esto, deben emplearse las órdenes de venta (en el caso de una manufactura tipo *make-to-order*) o los pronósticos existentes (para las producciones *make-to-stock*).

- Detección del Modelo: una organización puede utilizar varios modelos matemáticos para la optimización. El sistema debe permitirle al usuario seleccionar el que desea utilizar, pero también debe guardar uno marcado "por defecto", el cual se debe usar si no hay ninguna otra selección.
- Representación de Soluciones: las soluciones provistas por el sistema deben ser traducidas en tiempo de inicio y fin para cada trabajo, y mostrarse de forma gráfica empleando diagramas Gantt. El sistema también debe proveer otros tipos de gráficos para representar datos complementarios a la solución.
- Gestión de Algoritmos de Resolución: originalmente Framinan y Ruiz definieron que el sistema debe poder agregar nuevos algoritmos resolvedores, que el usuario debe poder seleccionar el que le parezca conveniente y que, a su vez, el sistema debe mostrar un algoritmo sugerido o 'por defecto', para cada uno de los modelos matemáticos que haya disponibles. Respecto a lo primero, y con el objetivo de mejorar la integración, se decidió que *E2OL* utilizaría sistemas de modelado (por ejemplo, GAMS, AIMMS, AMPL, etc.), los cuales ya contienen múltiples "solvers". Emplear esta clase de sistemas evita problemas de sintaxis de los modelos matemáticos, al quitar la necesidad de mantener un archivo del modelo con la sintaxis necesaria de cada uno de los resolvedores.
- Evaluación de Soluciones: el sistema debe permitir evaluar un escenario con diferentes objetivos. Es decir, que para cada modelo matemático, el usuario debe poder elegir la función objetivo del mismo; para el caso de un usuario avanzado, también debe ofrecerse la posibilidad de escribir una función personalizada.
- Análisis de Escenarios: el sistema debe permitir administrar distintas soluciones para un análisis "what-if". Para eso, *E2OL* debe permitir guardar una solución "posible" con el objetivo de volver a accederla para su posterior revisión, y debe facultar al usuario a modificar los parámetros de cada modelo. Se definen como parámetros a los datos de entrada cuyos valores pueden ser especificados por el planificador, variando 'condiciones' en cada uno de los escenarios. Esto también debe poder hacerse de forma automática, de un parámetro por vez, el usuario puede especificar el tamaño y rango de la variación de los escenarios, y el sistema resolverá automáticamente todas las posibilidades, para luego mostrar los resultados pertinentes.
- Interfaz Gráfica: se debe poseer una interfaz gráfica de usuario que debe ser operada por un planificador de capacidad intermedia, pero dando las opciones adecuadas a aquellos de nivel avanzado. El objetivo de dicha GUI (siglas de "Graphic User Interface") es lograr que todos los procesos sean transparentes para el usuario planificador que opera a *E2OL*.
- Integración con Sistemas Existentes: *E2OL* debe permitir extraer la información necesaria del sistema empresarial o ERP de la empresa donde se implemente. Este punto se tratará con más detalle en la Sección 3.
- Comprobación de Errores: se debe poder efectuar un análisis de errores para evaluar si el modelo da algún tipo de resultado determinado, o si produce un error, tales como soluciones no acotadas, no factibilidad, etc.

Otros requerimientos funcionales adicionales, que se pueden plantear para el sistema son los siguientes:

- Autenticación de Usuario: antes de utilizar las funciones de *E2OL*, el usuario debe autenticarse en el sistema usando las mismas credenciales que emplea para ingresar al sistema empresarial/ERP que posee la empresa.
- Listado de Órdenes/Pronósticos de Venta: *E2OL* no debe ligarse a trabajar sólo con uno de estas entradas de ventas, sino que -a través del configurador- debe poder listar las órdenes (o pronósticos) de venta, para que el usuario seleccione aquellos que utilizará en la resolución de un "scheduling" de producción.
- Guardado en Formato XML: sólo el escenario seleccionado debe guardarse en

la BDERP. Para los demás, el planificador debe tener la opción de guardarlos como archivos tipo XML.

1.2 Actores del Sistema

E2OL también posee actores. Éstos son agentes externos con los que se relaciona, utilizando distintos medios de comunicación. A continuación se detallan los principales actores detectados:

- Planificador Intermedio: persona de existencia física que utiliza un cliente *E2OL* para realizar el "scheduling" de una producción. Interacciona con la herramienta a través de las interfaces gráficas, y utiliza sólo las funciones básicas del sistema.
- Planificador Avanzado: persona de existencia física, cuyo rol especifica el de Planificador Intermedio. Este rol posee conocimientos avanzados sobre los modelos matemáticos que pueden utilizarse, lo que le permite usar funciones avanzadas de *E2OL*, tales como escribir funciones objetivo personalizadas, seleccionar algoritmos de resolución, entre otros.
- Base de Datos del ERP: (BDERP) dado que *E2OL* debe obtener todos los datos necesarios para ejecutar el modelo desde el ERP, debe poder comunicarse con la base de datos del ERP y extraer toda la información necesaria.
- Sistema de Modelado: sistema de información con el que *E2OL* se comunica para ejecutar y resolver el modelo matemático, de modo de independizarse de la sintaxis propia de cada uno de los "solvers".

1.3 Casos de Uso

En un proyecto de software, los diagramas de Caso de Uso (CU) permiten que el cliente analice qué funciones tendrá un sistema y qué alcance tendrán sus actores, ayudan a los desarrolladores a codificar, favorecen la documentación y las pruebas del sistema.

A continuación, en la Figura 1, puede observarse el diagrama general de casos de uso de *E2OL*, que detalla las principales funcionalidades del sistema y su interacción con los actores descriptos en la subsección 1.2. Como puede observarse, tanto la BDERP como el Sistema de Modelado reciben una comunicación desde *E2OL* y la responden, pero ellos no pueden iniciar la comunicación.

Este diagrama se completa con las fichas de

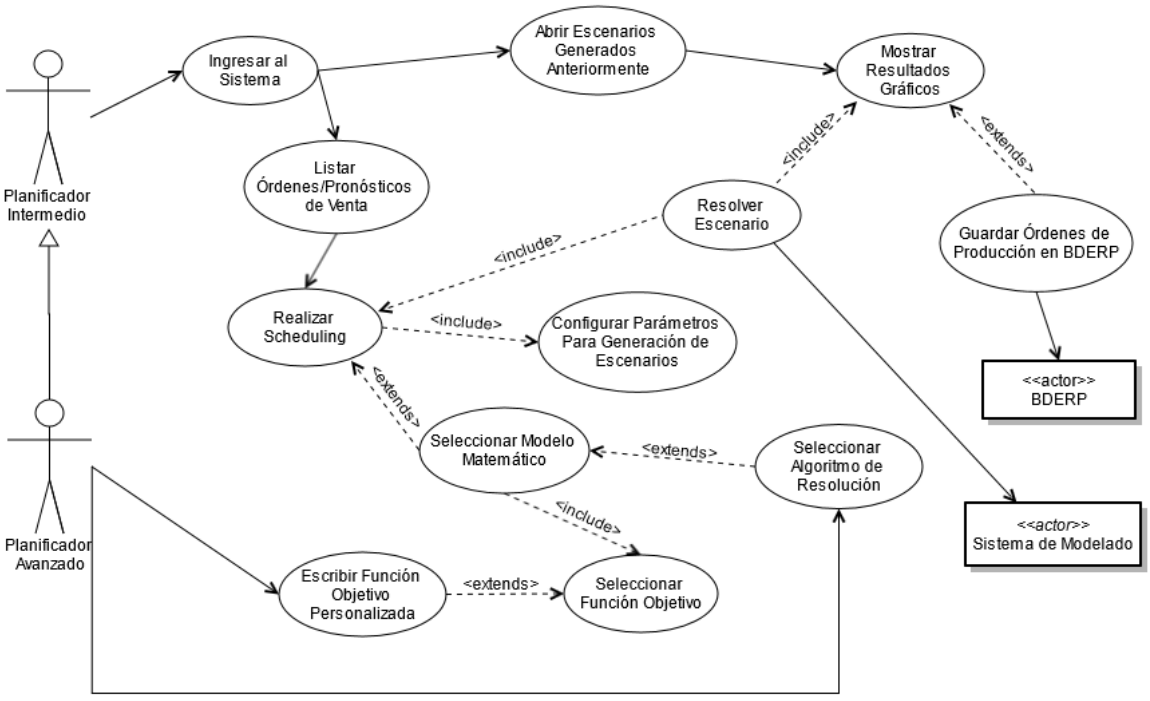

Figura 1. *Diagrama general de los casos de uso de E2OL.*

caso de uso correspondientes a cada uno de los CU detallados en la Fig. 1, pero por cuestiones de espacio, no han sido incluidas en el presente trabajo.

Los diagramas de CU fueron seleccionados

para posteriormente poder utilizar la metodología de Punto de Caso de Uso propuesta en [18], la cual permite desarrollar los casos de prueba para verificar los requerimientos funcionales de un producto de software. Para poder conocer si un sistema de información funciona correctamente, es necesario realizar una verificación del mismo, la cual se lleva a cabo usando los casos de prueba y comparando la salida esperada, contra la salida que realmente se obtuvo al ejecutar esa prueba en el sistema.

1.4 Requerimientos No Funcionales

Muy pocos de los trabajos realizados en el área hablan explícitamente de los atributos de calidad (o requerimientos no funcionales) que deben tener los sistemas de planificación y "scheduling" de la producción. Uno de ellos es [9], en el cual se analiza el desarrollo de un APS para una empresa específica; los autores de dicho trabajo plantean tres atributos de calidad: modularidad, flexibilidad y portabilidad. Sin embargo, analizando los requerimientos funcionales planteados en 1.1 y siguiendo las pautas de clasificación de atributos de calidad del estándar [19], se seleccionaron los siguientes atributos de calidad, como determinantes en el diseño y desarrollo de *E2OL*:

- Portabilidad: el sistema debe poder transferirse de un ambiente a otro, sin estar restringido ni por el hardware, ni por los otros sistemas con los que debe operar (por ejemplo: el sistema operativo, el ERP, el SGBD, y el sistema de modelado, entre otros). Como sub-atributo se destaca la *adaptabilidad*, como la capacidad del sistema de ser adecuado o reconfigurado según el entorno donde desee ejecutarse.
- Modificabilidad[1]: el esfuerzo de reconfigurar o modificar el sistema, debe ser mínimo. Los sub-atributos destacados, son: *mutabilidad*[2] (la capacidad del sistema de cambiar) y *estabilidad* (capacidad de mantenerse estable y compatible, minimizando el surgimiento de errores derivados de esos cambios).
- Funcionalidad: el sistema debe satisfacer los requerimientos funcionales planteados de forma adecuada. Como sub-atributos se destaca la *interoperabilidad* (la habilidad de *E2OL* de trabajar conjuntamente con otros sistemas).
- Performance[3]: es la relación entre los procesos que efectúa el software y la cantidad de recursos que emplea, bajo condiciones preestablecidas. Dentro de este atributo se puede evaluar: la cantidad de recursos empleados, y el tiempo ocupado para realizar una determinada acción.
- Usabilidad: *E2OL* debe ser fácil de usar por planificadores de nivel intermedio y avanzado. Como sub-atributos se destacan: *"learnability"* (facilidad para aprender a utilizar el programa), *comprensibilidad* (el sistema debe ser comprensible), *eficiencia* (velocidad con la que los usuarios pueden usar el sistema, una vez que han aprendido sus funcionalidades básicas), y también *"memorability"* (la capacidad de los usuarios de recordar las funciones y procesos del sistema).

Para poder verificar que el sistema cumple con los atributos que originalmente fueron planteados, éstos serán posteriormente evaluados utilizando métricas e indicadores.

2. Arquitectura del Sistema

En el ámbito académico se han realizado varias propuestas de arquitecturas para este tipo de sistemas. En el año 1997, Pinedo y Yen [20] detallaron tres módulos para un sistema de "scheduling" desarrollado con un lenguaje orientado a objetos; dichos módulos eran: la base de datos (almacenamiento de los datos), el generador del "scheduling" (funcionalidades necesarias para resolver el problema) y la interfaz gráfica de usuario (interfaz que proporciona la abstracción adecuada al usuario). Estos módulos constituyen los bloques principales de un sistema de soporte a la toma de decisión, y fueron analizados una vez más en el 2005 por Pinedo [21].

En [17], los autores proponen agregar un bloque más, denominado *"Business Logic Unit / Data Abstraction Management Module"* (por sus siglas, BLU/DAM), el cual se encarga de garantizar el nivel de abstracción requerido para acceder a los datos del sistema. Este módulo también debe encargarse de interconectar el sistema APS con el sistema empresarial o ERP de la empresa, y aislar las

[1] Denominado Maintainability en inglés.
[2] Denominado Changeability en inglés.
[3] También referido como Eficiencia.

rutinas relacionadas con optimización de la producción. Aún con estas propuestas de módulos o bloques que deben componer un APS muy pocos la llevaron a la propuesta de una arquitectura particular y menos aún a la implementación de la misma.

Por otra parte, las arquitecturas de los ERP y otros sistemas para empresas van desde cliente-servidor, n-bandas o la reciente tendencia de "cloud computing" [22], pero la más utilizada es de tipo cliente/servidor [23].

Este tipo de arquitectura es un patrón común compuesto de dos elementos, uno cliente -que mantiene la lógica y las interfaces del sistema- y otro servidor -encargado de la persistencia de datos-, cuya coordinación está descripta en términos del protocolo que el servidor emplea para comunicarse con cada uno de sus clientes. Pueden existir múltiples clientes, pero todos se comunican con el mismo servidor [24].

Por esto mismo, se seleccionó el patrón arquitectónico cliente/servidor, como la base de la estructura de *E2OL*.

Sin embargo, aún existe una problemática. Con esta herramienta se intenta desarrollar un producto genérico que pueda adaptarse a la situación de las empresas que deseen implantarlo, de forma rápida y con mínimos impactos en el código. Esto lleva a considerar dos factores:

- *E2OL* no puede ligarse a la estructura de base de datos de un ERP en particular, ni trabajar las sentencias SQL directamente desde el código del sistema.
- *E2OL* no debe comprometerse a trabajar con un solo Sistema de Modelado (SM).

Estos elementos resultan determinantes al momento de seleccionar una arquitectura, ya que de no cumplirlos, se comprometería los atributos de calidad de portabilidad y modificabilidad, planteados previamente.

De este modo, con el objeto de generar una arquitectura que se adaptara a las necesidades de *E2OL*, se listaron los datos que deben variar en cada implantación:

1. Datos de conexión a la base de datos del ERP (BDERP), como así también la información sobre las tablas que se deben consultar, ya sea para leer datos, como para también escribirlos (en el caso de las órdenes de producción).
2. Información sobre si el sistema debe trabajar con órdenes de venta, o con pronósticos, datos de productos, máquinas, recursos humanos, stock disponible de materias primas, insumos y productos, etc.
3. Datos pertinentes a los modelos matemáticos. Esto incluye: sus nombres, los archivos que los componen, las funciones objetivo que puede usar cada uno, los algoritmos de resolución disponibles, los datos tipos parámetro que pueden modificarse para generar escenarios.
4. Resultados: asignación de materias primas, máquinas y recursos, plan de producción, secuencia de producción.
5. Configuración de los resultados de manera gráfica, para mostrarlos mediante la interfaz de usuario, tras resolver un modelo matemático.
6. Comportamiento y procesos necesarios para resolver los modelos con un SM en particular. Esto se refiere a la lógica intrínseca que E2OL necesita para procesar los archivos que usa cada SM, traducir sus resultados a gráficos, entre otros.

Del listado anterior, el único punto que no puede ser aislado completamente del código, es número 5): el comportamiento y proceso para utilizar determinados Sistemas de Modelado.

2.1 Arquitecturas Analizadas

A continuación, se presentan las arquitecturas analizadas para el diseño e implementación de E2OL que permitan alcanzar los requerimientos funcionales y no-funcionales planteados en los apartados anteriores.

a. Clases Serializables: constituida por objetos tipo interfaces que delimitarían el comportamiento necesario y el formato de datos a pasar en cada mensaje, y la empresa que lo implante debería extender el código para personalizar el sistema. La ventaja de esta idea es que la organización puede modificar E2OL para que encaje perfectamente con sus objetivos. Sin embargo, esta idea tiene más aspectos negativos que positivos. Por un lado, implica contratar desarrolladores expertos en el lenguaje que fue desarrollado E2OL y generar un proyecto de adaptación el cual puede demorar mucho tiempo y tener un presupuesto elevado; esto también

repercute en que el tiempo de implantación es mucho mayor. Finalmente, al tener que embeber este tipo de información y lógica dentro del código hace que el sistema pierda la capacidad de adaptación y modificabilidad, haciendo que hasta un cambio en sistemas externos (como el ERP o el SM) repercuta ampliamente en el mismo.

b. Archivos Configurables: mediante un configurador de instalación, el administrador genera -de forma transparente para él- varios archivos que contienen los datos necesarios para que *E2OL* pueda efectuar todas sus operaciones; estos archivos se instalarían en cada cliente. La ventaja de esta opción es que la adaptación se realiza de forma guiada a través del configurador, este proceso también es mucho más rápido y menos costoso que la propuesta (a). Sin embargo, la gran desventaja, es que al mantener parte de la base de conocimiento en el lado cliente de la aplicación (los archivos con la información), se viola el principio del estilo cliente/servidor, además de generar redundancia de datos, lo que puede repercutir en una posible futura inconsistencia ante un cambio.

c. Base de Datos de Nomenclatura: con esta opción se implementa una segunda base de datos, denominada Nomenclatura (BDN), la cual almacenaría -en un formato estandarizado- todos los datos necesarios mencionados en la lista de la Sección 2; dicha BD sería creada y especificada mediante un configurador, pudiendo ser implementada en el mismo SGBD que la BDERP, o en uno diferente. Con esta opción se anula la desventaja de la opción (a) -modificación intensiva del código, elevados costos/tiempos de implantación- ya que se provee un sistema que guía al configurador a través del proceso de adaptación, haciéndolo más sencillo y transparente para el mismo; a su vez, se deshace la desventaja de la opción (b), ya que al poseer dos BD en el lado servidor, no se mantienen datos en los clientes, manteniendo el patrón cliente/servidor y evitando la redundancia al obligar a que todas las terminales clientes se conecten a la BDN. La desventaja de esta opción radica en que el sistema resulta más lento, debido a la doble conexión con bases de datos.

Considerando las tres propuestas, puede observarse que todas tienen aspectos positivos y negativos que balancear. La opción C es la que permite la mayor estandarización del formato en que se almacenan los datos requeridos -listados en la Sección 2- y, a través del sistema configurador, se especifica la BDN, logrando las características de modificabilidad y adaptabilidad. Por esto mismo, se decidió desarrollar el sistema "*E2OL Configurer*," complementario a *E2OL*, que permite realizar esta tarea a través de un proceso guiado y transparente. Cabe destacar que esta opción también mantiene el patrón cliente/servidor, evitando problemas con la localización de la información, y su posible redundancia, al hacer que todos los clientes se conecten a las mismas bases de datos.

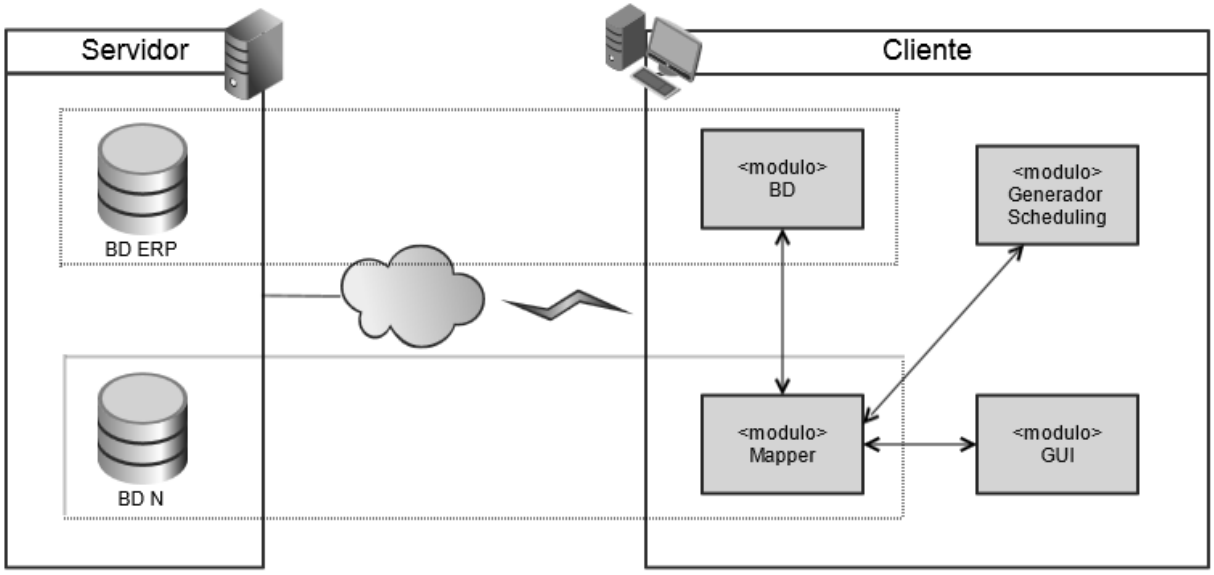

Figura 2. *Arquitectura cliente/servidor de E2OL.*

Entonces, bajo esta implementación, *E2OL* primero se conecta a la BDN para buscar la información necesaria, y luego a la BD del ERP, utilizándola para extraer los datos necesarios.

De esta forma, en la Figura 2 puede observarse la arquitectura de *E2OL*, representada en un lenguaje informal. Del lado servidor se detallan las dos bases de datos con las que interactúa la herramienta, la del ERP y la de Nomenclaturas, independientemente del SGBD en que se soporten.

Mientras, en el lado cliente, se detallan los cuatro grandes módulos en los que se propone desarrollar al sistema: un módulo de base de datos que se conecte con la BDERP, el generador del scheduling, la interfaz gráfica de usuario, y el "Mapper". Éste último módulo es el que controla las transacciones con la BDN y transforma los datos que recaba de ésta a un formato comprensible por los demás módulos.

Es importante destacar que estos módulos pueden agruparse en los cuatro bloques que proponen Framinan y Ruiz en [17], pero proveyendo una forma de implementación concreta. De esta manera, el módulo BD del cliente, y la BDERP forman parte del bloque de base de datos del que se habló en la Sección 2, mientras que el módulo "Mapper" del cliente y la BDN son la implementación del bloque BLU/DAM. Esta agrupación también puede verse en la Figura 2, indicada con dos rectángulos de líneas punteadas.

Con la estructura de módulos propuesta para el cliente, también puede solucionarse el problema de la personalización de procesos y comportamientos para adecuarlos a los SM que se desee utilizar -punto 5 de la lista mencionada en la Sección 2-. Empleando orientación a objetos, la cual favorece la encapsulación de conocimiento, su abstracción, modularidad y especificación de comportamiento en jerarquías [25], se puede desarrollar un sistema que sea adaptable de forma controlada, al generar clases tipo interfaces dentro del módulo Generador Scheduling.

En este caso, el término "interface" define un tipo abstracto que no contiene datos, pero que expone y limita comportamiento, a través de métodos. Como las interfaces son sólo definiciones de tipo, los tipos de objetos a ser intercambiados pueden especificarse en términos de interfaces, en lugar de clases concretas. Esto permite que las clases que extiendan a dichas interfaces, usen los mismos métodos, pero intercambiando diferentes tipos de objetos; de esta forma, el código obtenido resulta más genérico y reutilizable [25].

Citando a Brooch en [25], *"[…] la interface de una clase provee una mirada externa y enfatiza la abstracción, al mismo tiempo que se esconde su estructura y los secretos del comportamiento […]"*. De esta forma, se podrán implementar clases a partir de dichas interfaces, que cambien la lógica del comportamiento que une a *E2OL* con el Sistema de Modelado, sin que esta recodificación tenga un impacto elevado en el código.

A continuación, en el diagrama de clases de la Figura 3 pueden observarse las interfaces principales del módulo Generador Scheduling, detalladas para definir el intercambio de objetos que el sistema debe utilizar. A partir de estas interfaces pueden implementarse clases que contengan la lógica específica para un determinado Sistema de Modelado, las cuales pueden, a su vez, tener más métodos; sin embargo, dichos métodos deben ser privados y utilizados sólo dentro de dichas clases. Esto permite la adaptación de E2OL con el SM empleado por la empresa que lo implemente, favoreciendo una adaptación guiada y controlada, que minimiza el impacto en el código y, por ende, la cantidad de trabajo requerido para su aplicación.

2.2 Herramientas de Desarrollo

Con todas las condiciones planteadas anteriormente, se decidió programar a *E2OL* en Java. Java es un lenguaje de programación orientado a objetos, de propósito general, concurrente y basado en clases, específicamente diseñado para tener la menor cantidad posible de dependencias de implementación. En este lenguaje, el código que se ejecuta sobre una plataforma no necesita ser recompilado como para correr en otra, ya que utiliza una Máquina Virtual [26]; por esto mismo, al programar en Java se favorece el atributo de portabilidad, aun manteniendo un patrón arquitectónico tipo cliente/servidor.

Por otro lado se seleccionó la librería de conexión JDBC (sigla de "Java Database Connectivity"), que es el estándar de la industria para la conexión independiente a bases de datos relacionales, entre el lenguaje

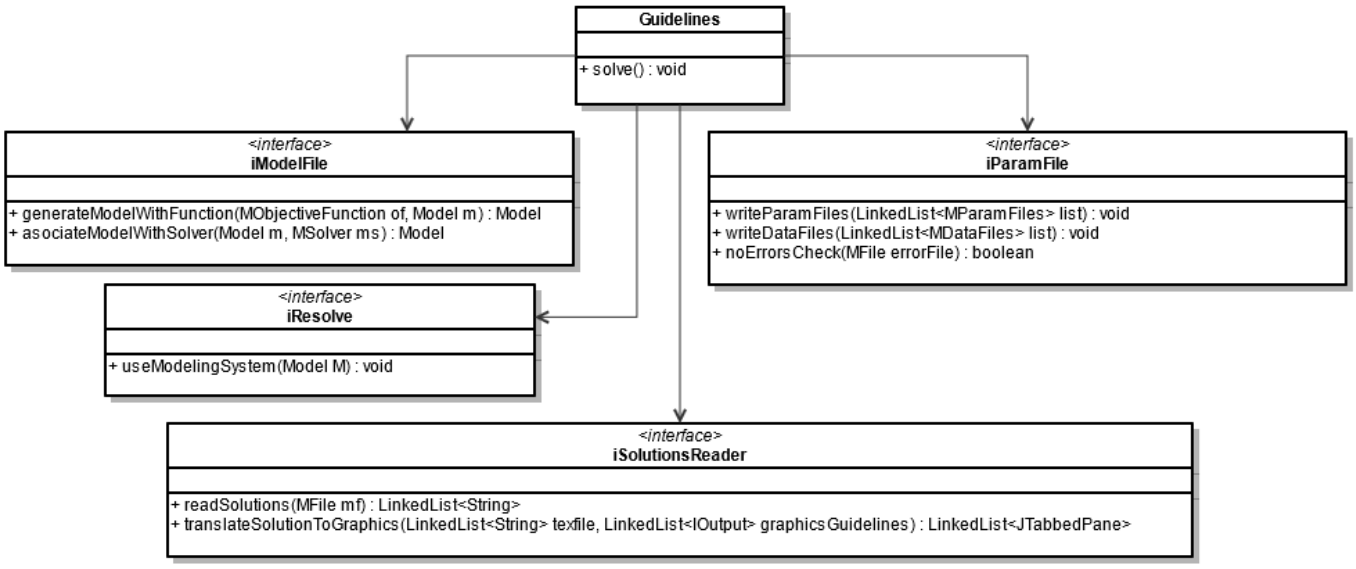

Figura 3. *Diagrama de las clases interfaces del módulo Generador Scheduling de E2OL.*

Java y un amplio rango de bases de datos [27]. Los parámetros de conexión, así como el driver específico, se configuran desde *E2OL Configurer*.

A su vez, se han usado otras librerías tanto como para la interacción con los Sistemas de Modelado (SM), la realización de gráficos y la interacción con XML al momento de guardar los escenarios obtenidos. Estas librerías no serán detalladas en el presente artículo.

3. Base de Datos Nomenclatura

La Base de Datos Nomenclatura (BDN) tiene como objetivo permitir que E2OL pueda obtener información del sistema de la empresa y operar con el SM, sin estar ligado de forma permanente a que estos dos sistemas externos tengan una estructura definida.

Para poder ejecutar los modelos matemáticos, E2OL requiere información de las órdenes/ pronósticos de ventas que utiliza la organización, información sobre las máquinas o recursos de producción, capacidad de stock, costos y otras restricciones inherentes a cada producción; como fue mencionado previamente. Con respecto a los resultados de la ejecución de los módulos de optimización, E2OL debe ser capaz de insertar en el ERP la información de las órdenes de producción seleccionadas como óptimas por el planificador. Toda esta información se extrae a partir de las consultas SQL que se almacenan en la BDN y que el "Mapper" traduce en sentencias SQL; estas consultas se definen en el momento de la especificación de la BDN en la ejecución de *E2OL Configurer*, e independizan a *E2OL* de cualquier conocimiento específico de cómo se componen dichos datos.

De este modo, considerando el listado detallado en la sección 2, la BDN tiene por objetivo almacenar tres grupos de información:

a. Datos específicos de la estructura de la BDERP. Esto se refiere principalmente a las sentencias SQL que deben realizarse para obtener o guardar información en ella. Se almacena, entre otras cosas, el tipo de sentencia (inserción o selección), las columnas que se consultan/insertan y sus respectivas tablas y, en el caso de las selecciones, las restricciones que debe tener dicha consulta.

b. Datos referidos a los modelos matemáticos. Para cada uno de ellos, se guarda información sobre los archivos que lo componen, todas las funciones objetivo disponibles, los algoritmos de resolución que pueden emplearse, entre otros. Respecto a los archivos, se incluyen los que tienen datos de entrada (sean tipo parámetro o cargados desde el ERP), los archivos del modelo en sí mismo, aquellos donde se guardan las funciones objetivo, y el/los archivo/s con los resultados de la ejecución.

c. Datos necesarios para la configuración de las interfaces de *E2OL*; esto no incluye solamente a la transformación a gráficos de los resultados de la ejecución del modelo matemático, sino también a todas las ventanas del sistema que deben mostrar datos específicos de una implementación. Ejemplo de esto, son: las tablas donde se listan las órdenes/pronósticos de venta según corresponda, las ventanas para la configuración de los parámetros del modelo matemático, etc.

3.1 E2OL Configurer

La BDN se especifica, crea y completa al utilizar a *E2OL Configurer*. Este sistema debe ejecutarse una sola vez, y se encarga de enlazar la BDN con la BDERP.

En la Figura 4 puede observarse una captura de *E2OL Configurer*, de la pantalla que permite el ingreso de los datos donde se alojará la BDN. De este modo, el usuario especializado debe ingresar el tipo de SGBD donde se insertará (con el objeto de emplear la sintaxis SQL adecuada), la ubicación (IP o DNS del servidor donde se encuentra, y el puerto de acceso), el nombre que tendrá dicha base de datos, y el usuario/contraseña del administrador. Esto es necesario para poder comprobar la conexión, y usar dichas credenciales para la creación, carga y configuración de la BDN.

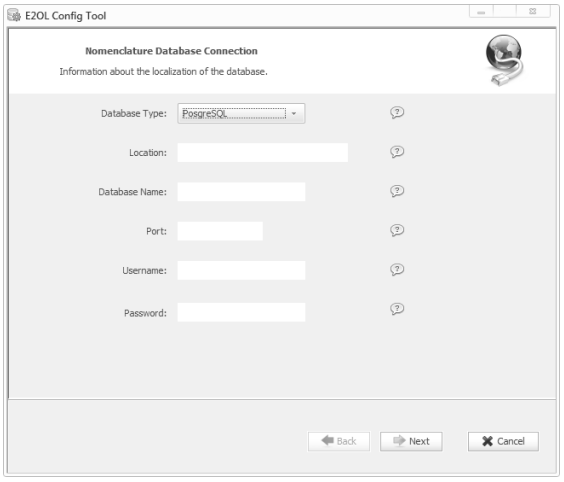

Figura 4. *Interfaz gráfica de E2OL Configurer, para la creación de la BD Nomenclatura.*

Las etiquetas con un símbolo "?" a la derecha de cada campo proveen "tooltips" (mensajes de ayuda) al usuario sobre cómo completar cada uno de los campos. Una interfaz muy similar es empleada para configurar los datos de conexión a la BDERP.

Por otro lado, *E2OL Configurer* también permite cargar las sentencias SQL para la obtención de los distintos datos, como así también las de inserción de las órdenes de producción.

A modo de ejemplo, en la Figura 5 puede observarse la ventana para la generación de las restricciones de una sentencia SQL de selección. En el panel de la derecha se pueden cargar configuraciones sobre el tipo de restricción (título Opciones; por ejemplo, si es `IN`, o `GROUP BY`, etc.), las referencias a la/s columna/s que restringe (dos columnas si es una restricción de "matching" de claves foráneas/primarias), y otros valores, como ser la función que usaría un `HAVING`, etc. Estas opciones se habilitan o deshabilitan según las opciones que haya tildado el usuario. En el panel de la izquierda, se observan las restricciones ya creadas con forma de árbol, detallando todos sus componentes; haciendo clic derecho sobre una restricción se accede a un menú que permite modificarla o eliminarla. Las columnas y tablas empleadas en las restricciones son las que posee la BDERP a la cual se conecta este configurador, permitiendo adaptar *E2OL* a cualquier sistema empresarial (sea este un ERP o no) con el que se enlace.

El actor principal de *E2OL Configurer* podría ser un miembro joven del Departamento de Sistemas de la empresa ó actor con un mínimo conocimiento sobre la estructura del ERP, ya que *E2OL Configurer* lo guía a través del proceso de especificación. Es importante destacar que este complemento de E2OL cumple un rol fundamental, ya que a partir de las definiciones que se hagan con el mismo serán los resultados que se obtengan en la operación del sistema.

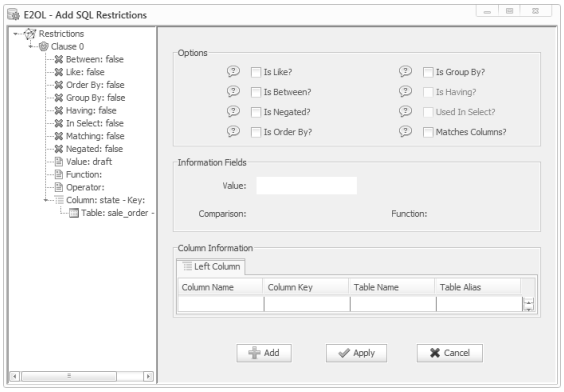

Figura 5. *Interfaz de E2OL Configurer, para crear las restricciones de una sentencia SQL de selección.*

Resultados

Un estudio detallado de trabajos previos en el área, logró encontrar que la mayoría de las propuestas eran o implementaciones específicas de sistemas APS para una empresa en particular, o propuestas de generación de APS genéricos -o el estudio de sus medios de optimización-, que ofrecían características más avanzadas respecto a los ERP, pero que no permitían considerar las características particulares de cada negocio que deseara implementarlos.

La herramienta propuesta, *E2OL*, fue

pensada con la idea de generar un sistema empaquetado que pudiera ser fácilmente adaptable por aquellos que deseen implantarlo, con el objetivo de permitir el intercambio de modelos matemáticos para lograr considerar las particularidades de cada organización, para ello se propuso una arquitectura explícita y definida, a partir de un estudio detallado de requerimientos funcionales y atributos de calidad. En estos momentos se está desarrollando la primera versión del sistema, centrándose en esta etapa en la implementación de "*E2OL Configurer"*.

Conclusiones

El presente artículo propone una arquitectura concreta para un sistema Advanced Planning and Scheduling (APS). Uno de los principales objetivos del sistema que se propone es que pueda adaptarse a los sistemas de información de cualquier empresa de manufactura y pueda sacar ventajas de sus particularidades.

La primera etapa fue la elicitación. Se extrajeron requerimientos funcionales que fueron recompilados de trabajos académicos del área y de la experiencia recogida de los trabajos realizados a empresas por los investigadores que participan de este trabajo. Se agregaron también otros requisitos no necesariamente ligados con las funcionalidades de planificación y "scheduling" de la producción pero que son características importantes del sistema que se quiere desarrollas. También se incluyeron los actores que deben interactuar con el sistema. Tanto las funcionalidades como los actores son detallados en un diagrama de Caso de Uso general, a partir del mismo se planea utilizar la medida Punto de Caso de Uso para generar escenarios de prueba y verificar el sistema. También se plantearon atributos de calidad, siguiendo la estructura propuesta por el estándar ISO/IEC, los cuales serán posteriormente verificados empleando métricas e indicadores.

La siguiente etapa fue plantear una arquitectura que permitiera implementar un sistema que cumplimentara todos los requerimientos (funcionales y no funcionales) detallados previamente. Se analizaron los análisis teóricos de otros autores, para finalmente proponer una estructura práctica que actualmente se encuentra en implementación.

E2OL posee una arquitectura cliente-servidor, dividida en cuatro grandes bloques. Uno de estos se encarga de garantizar el nivel de abstracción requerido para acceder a los datos del sistema; dicho bloque está compuesto por un módulo en el lado cliente (denominado Mapper) y una base de datos (denominada Nomenclatura) que contiene toda la información necesaria para aislar ese conocimiento de *E2OL*, haciéndolo altamente adaptable, característica que se considera fundamental para un APS que debe ser especificado en diferentes ambientes informáticos. La base de datos Nomenclatura es creada y configurada a través de un complemento del sistema, llamado *E2OL Configurer*, actualmente en desarrollo.

Trabajos Futuros

El trabajo presentado contiene varias líneas de investigación, detalladas a continuación.

Por un lado, se quieren realizar las evaluaciones concretas empleando Punto de Caso de Uso sobre E2OL, con el objeto de analizar si cumple o no los requerimientos planteados. Siguiendo esta temática, también se trabajará en el desarrollo de métricas e indicadores para evaluar los atributos de calidad propuestos y utilizarlos para medir el rendimiento de E2OL respecto de dichos atributos. Estos dos trabajos futuros favorecen la comprobación empírica del uso de E2OL.

Otra línea de trabajo, es adecuar la BDN propuesta al estándar ANSI/ISA 95, en orden de normalizar la propuesta y ampliar su alcance.

Finalmente, se abre la posibilidad de generar casos de estudio de diversa complejidad, con el fin de evaluar E2OL en ambientes de características reales, y analizar la posibilidad de transferencia del mismo.

Referencias

[1] Wang Shaojun, Wang Gang, Lü Min y Gao Guoan, «Enterprise resource planning implementation decision & optimization models,» *Journal of Systems Engineering and Electronics,* vol. 19, nº 3, pp. 513-521, 2008.

[2] M. A. Rothenberger y M. Srite, «An Investigation of Customization in ERP System Implementations,» *Transactions On Engineering Management,* vol. 56, nº 4, pp. 663-676, 2009.

[3] A. Ragowsky y T. M. Somers, «Special section: Enterprise resource planning,» *Journal of Management Information Systems,* vol. 19, pp. 11-15, 2002.

[4] J. Scott y I. Vessey, «Managing risks in enterprise systems implementations,» *Communications of the ACM,* vol. 45, nº 4, pp. 47-81, 2002.

[5] B. L. McCarthy y J. Liu, «Adressing the gap in scheduling research: A review of optimization and heuristic methods in production scheduling.,» *International Journal of Production Research,* vol. 18, nº 4, pp. 59-79, 1993.

[6] W. W. Hopp y M. Spearman, Factory Physics, New York: McGraw-Hill, 2000.

[7] H. Stadler y C. Kilger, Supply Chain Management and Advanced Planning: Concepts, Models Software and Case Studies, Tercera ed., Berlín: Springer, 2005.

[8] H. Stadler, «Supply chain management and advanced planning—basics, overview and challenges,» *European Journal of Operational Research,* vol. 163, pp. 575-588, 2005.

[9] S. Bourgeois, A. Artiba y C. Tahon, «Integration of Short Term Scheduling with an MRP-II System: Industrial Implementation,» 1993.

[10] Y. H. Lee, C. S. Jeong y C. Moon, «Advanced planning and scheduling with outsourcing in manufacturing supply chain,» *Computers & Industrial Engineering,* nº 43, pp. 351-374, 2002.

[11] C. Lalas, D. Mourtzis, N. Papakostas y G. Chryssolouris, «A combinatory approach to order release and shop scheduling in discrete manufacturing enviroments,» de *10th IEEE International Conference on Emerging Technologies and Factory Automation*, Catania, Italy, 2005.

[12] R. Masuchun, W. Masuchun y T. Thepmanee, «Intgrating m-Machine Scheduling into MRP,» de *4th International Conference on Innovative Computing, Information and Control*, 2009.

[13] I. Van Nieuwenhuyse, L. De Boeck, M. Lambrecht y N. J. Vandaele, «Advanced resource planning as a decision support module for ERP,» *Computers in Industry,* vol. 62, pp. 1-8, 2011.

[14] H.-H. Hvolby y K. Steger-Jensen, «Technical and industrial issues of Advanced Planning and Scheduling (APS) systems,» *Computers in Industry*, vol. 61, pp. 845-851, 2010.

[15] J. H. Gilmore y J. Pine, «The four faces of mass customization,» *Hardvard Business Review*, vol. 75, nº 1, pp. 91-101, 1997.

[16] M. Glinz y R. J. Wieringa, «Stakeholders in Requirements Engineering,» *IEEE Software,* vol. 24, nº 2, pp. 18-20, 2007.

[17] J. M. Framinan y R. Ruiz, «Architecture of manufacturing scheduling systems: Literature review and an integrated proposal,» *European Journal of Operational Research,* vol. 205, pp. 237-246, 2010.

[18] J. Heumman, «Generating Test Cases From Use Cases,» *The Rational Edge,* nº Junio, 2001.

[19] ISO/IEC, Software Engineering - Product Quality - Part 1: Quality Model, 9126-1:2001, 2001.

[20] M. Pinedo y B. P.-C. Yen, «On the design and development of object-oriented scheduling systems,» *Annals of Operations Research,* nº 70, pp. 359-378, 1997.

[21] M. Pinedo, Planning and Scheduling in Manufacturing and Services, New York: Springer, 2005.

[22] C. McKenna, «Cloud and Open Source Enterprise Resource Planning,» de *International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government*, 2011.

[23] J. R. Muscatello, M. H. Small y I. J. Chen, «Implementing enterprise resource planning (ERP) systems in small and midsize manufacturing firms,» *International Journal of Operations & Production Management,* vol. 23, nº 8, pp. 850-871, 2003.

[24] L. Bass, P. Clements y R. Kazman, Software Architecture in Practice, Boston, USA: Addison-Wesley Longman Publishing, 2003.

[25] G. Booch, R. Maksimchuk, M. Engle, B. J. Young, J. Conallen y K. Houston, Object-Oriented Analysis and Design with Applications, Tercera ed., Westford, Massachusetts: Addison-Wesley, 2007.

[26] Oracle Corp., «Design Goals of the Java Programming Language,» 1999. [En línea]. [Último acceso: 2013].

[27] Oracle Corp., «Java SE Technologies,» 2012. [En línea]. [Último acceso: 2013].

Datos de Contacto

Ing. Melina Carolina Vidoni. Instituto de Desarrollo y Diseño, INGAR UTN-CONICET. E-mail: melinavidoni@santafe-conicet.gov.ar.