OLC y Monitoreo de procesos en el *cloud*: un caso de estudio

José Martínez Garro
Universidad Nacional de La Plata
UNLP
La Plata, Argentina
josemartinezgarro@gmail.com

Patricia Bazán
LINTI –UNLP
Universidad Nacional de La Plata
La Plata, Argentina
pbaz@ada.info.unlp.edu.ar

Javier Díaz
LINTI – UNLP
Universidad Nacional de La Plata
La Plata, Argentina
jdiaz@ada.info.unlp.edu.ar

***Abstract*—el uso de BPM (Gestión de procesos de negocio) en ambientes de *cloud computing* ha producido cambios y nuevas arquitecturas en pos de conceptos como descomposición y monitoreo de procesos. La aplicación del concepto de descomposición sobre procesos de negocio ha permitido su ejecución en entornos tanto embebidos como basados en *cloud*, posibilitando obtener las ventajas de ambos enfoques bajo criterios tales como sensibilidad en los datos, alta capacidad de cómputo y portabilidad de los sistemas. Luego de la fase de ejecución, el ciclo de vida de los procesos prosigue con el monitoreo; en este caso, el mismo debe ser efectuado conservando la visión del proceso original que ha sido descompuesto. Los procesos de negocio ejecutan actividades y transiciones que generan cambios de estado en los datos asociados. Estas unidades de datos resultan de interés para el monitoreo debido a que contienen información complementaria asociada a la ejecución del proceso. En el presente trabajo se aplica una arquitectura de ejecución y monitoreo de procesos descompuestos definida en trabajos previos sobre un caso de estudio concreto en un entorno de investigación, empleando a su vez objetos de datos OLC (Objetos de ciclo de vida) para la ampliación de los resultados obtenidos.**

***Keywords*—OLC,descomposición, monitoreo, cloud, BPM.**

I. INTRODUCCIÓN

El monitoreo es una de las etapas más importantes del ciclo de vida de los procesos de negocio, asegurando el objetivo de mejora continua que propone la solución de problemas con enfoque en la gestión de procesos. Además, la utilidad de dicho monitoreo se incrementa en un entorno de ejecución automatizado donde los BPMSs son los protagonistas principales.

Ambos aspectos, tanto la ejecución como el monitoreo de procesos de negocios, cobran un significado diferente si se trasladan desde una arquitectura centralizada clásica hacia un ambiente colaborativo híbrido, donde se combinan elementos de tipo embebido con componentes *cloud*, y a su vez estos últimos pueden ser públicos o privados.

Ante este contexto de descentralización es importante conservar la perspectiva del modelo de proceso original con una visión centralizada, y por otro lado, enriquecer los rastros que dejan los procesos mediante objetos de datos OLC, aunque en un entorno distribuido. Estos datos resultan de importancia para abarcar la mayor cantidad de información posible y que esta resulte útil para la medición y mejora de los procesos.

El presente trabajo aplica una arquitectura de ejecución y monitoreo de procesos de negocios descompuestos, planteada en trabajos anteriores, a un caso concreto incorporando además el uso de objetos OLC. El artículo se organiza de la siguiente manera: en la sección 2 se analiza la bibliografía actual considerando conceptos asociados a descomposición de procesos en un ambiente híbrido, el monitoreo de procesos consecuente a dichas ejecuciones, y la necesidad de unir los procesos con objetos de datos de manera de obtener información relevante para complementar el monitoreo. En las secciones 3 a 6 se retoma la arquitectura para la ejecución y monitoreo de procesos descompuestos, planteada en trabajos previos [8], y se incorpora a la misma el uso de objetos OLC. En la sección 7 se aplica la arquitectura a un ejemplo concreto construido en nuestro entorno de investigación, desarrollando las tres premisas que nos ocupan actualmente: ejecución, monitoreo y uso de objetos OLC en procesos descompuestos. Para finalizar, se presentan conclusiones acerca del estado del arte y propuestas de trabajo futuro para la presente línea.

II. TRABAJOS RELACIONADOS

Es posible encontrar en la bibliografía actual diversas investigaciones relacionadas con los temas principales del presente trabajo. En primer lugar, en [1] se detalla el modelo de *cloud computing* y su tecnología, así como también el despliegue de un entorno de BPM, ubicando en cada modelo de servicios las distintas componentes de un BPMS. Este trabajo es una puesta en escena inicial acerca de las futuras posibilidades que brinda el *cloud* al paradigma BPM. En contraposición a esto último, no aborda cuestiones de distribución de procesos como tampoco de un monitoreo distribuido, tal cual es el foco de nuestro trabajo.

Por otra parte, en [2] se aborda el impacto del *cloud* en los BPMS como un salto en la escala de madurez para la adopción de BPM, considerando la distribución de sus componentes bajo cuatro esquemas diferentes y analizando su aplicación acorde al dominio del problema. Asimismo refiere el concepto de iBPMS como una evolución de dicha tecnología hacia

entornos sociales, colaborativos y por ende, vinculados al *cloud*. Sin embargo no plantea un escenario concreto de distribución de procesos ni analiza las peculiaridades acerca del cumplimiento de uno de los aspectos más importantes de la mejora continua de los mismos: el monitoreo de los procesos distribuidos en el ambiente en cuestión.

En [3] se presenta una propuesta de distribución de procesos que hemos evaluado muy conveniente. De hecho, la misma ha sido tomada como base para una alternativa propia, donde se combinen diferentes estrategias con el fin de definir una arquitectura de monitoreo distribuido [7].

A su vez en [4] se presenta una extensión a los BPMS aportando la capacidad de llevar el rastro de eventos, además de actividades que utilizan la idea de CEP (*Complex Event Processing*). Este enfoque en efecto trata el tema del monitoreo en profundidad mediante el mismo mecanismo propuesto por nuestro trabajo: el de enriquecer los rastros que dejan las ejecuciones de los procesos mediante datos del negocio, aunque sigue sin abordar este mismo enfoque de manera distribuida.

En [5] y [6], por otra parte, se plantea un mecanismo de monitoreo enriquecido por OCL al igual que en el presente trabajo, pero lo realiza desde una visión centralizada. Estos trabajos han sido fuente de inspiración para elaborar una propuesta distribuída sobre un caso real.

III. COMBINACIÓN DE ENTORNOS EMBEBIDOS Y BASADOS EN CLOUD

La protección de la privacidad es una de las barreras principales para ejecutar BPM en un ambiente basado en *cloud*. No en todos los casos es posible colocar los datos sensibles fuera de la organización, a menos que se trate de un entorno de *cloud* privado. Por otro lado, se hace necesario observar la portabilidad de los productos y sus versiones, así como su disponibilidad en un sistema basado en *cloud computing*. Otro problema no menor en el presente contexto es la eficiencia a la que se ven sometidas las tareas de un proceso en un ambiente distribuido, esencialmente debido a la transferencia de datos.

Las actividades de alta intensidad de cómputo pueden obtener beneficios en el *cloud* debido a la escalabilidad y a la alta disponibilidad de fuerza de cómputo. Por el contrario, las tareas que no poseen alta intensidad computacional no siempre sacan ventajan de este contexto. La performance de una actividad tradicional que se ejecuta en un ambiente embebido debería ser mejor en comparación al *cloud*, debido a las cantidades de datos que deben ser transferidas para ejecutarla. Este último tipo de actividades puede incluso resultar siendo más costoso debido a que la transferencia de datos es un criterio de facturación primario en la nube [1] [2].

Hablando en términos de implementación acerca de una arquitectura híbrida, existen formas de calcular la distribución óptima, siendo el esquema de costos en el *cloud* un tópico investigado en diversos trabajos. Se han presentado varias fórmulas que permiten calcular la distribución óptima de las actividades, debido a que las mismas pueden ser ubicadas en la nube o en un sistema embebido. Los cálculos toman en consideración diferentes ítems, tales como tiempo, costos y riesgos en la privacidad. Mediante el uso de dichas fórmulas los usuarios pueden efectuar estimaciones de costos acerca de cómo desplegar parte de sus aplicaciones en un sistema basado en c*loud* o embebido en forma alternativa [4] [5].

IV. IMPLEMENTACIÓN DE UN ESQUEMA HÍBRIDO

Es posible generalizar el mecanismo de distribución e identificar un patrón donde el motor de procesos, las actividades y los datos se despliegan tanto en el *cloud* como en el usuario final en forma simultánea. Esta solución presenta dos beneficios potenciales:

1) El motor de procesos regula los flujos de control y de datos. Una actividad recibe datos desde el motor de procesos y, luego de su ejecución, los resultados son pasados nuevamente a dicho motor.
2) Cuando la nube no se encuentra accesible, los usuarios pueden ejecutar los procesos de negocio de manera completa en el sistema embebido, hasta que el anterior se encuentre disponible nuevamente.

Para poder ejecutar un proceso de negocio en dos motores separados, el mismo debe ser dividido en dos procesos individuales. La comunicación entre ambos sistemas se puede describir usando un lenguaje de orquestación de procesos, como por ejemplo BPEL. El monitoreo de procesos de negocio se vuelve más complicado ahora, debido a que el proceso ha sido dividido en dos o más partes que deben ser consideradas en forma conjunta. Como solución, se puede desarrollar una herramienta de monitoreo para observar el proceso original, a través de la combinación de los detalles recolectados de los procesos individuales. Este punto será analizado con más detalle a posteriori.

Tal como se ha presentado previamente en [8], hay diversas aproximaciones que implementan descomposición de procesos en una arquitectura híbrida, considerando aspectos tales como sensibilidad de datos, portabilidad de aplicaciones y altas prestaciones de cómputo. Una vez que el criterio de descomposición fue establecido, es necesario sincronizar las distintas partes en tiempo de ejecución. En nuestro caso, tal como fue presentado en [8], este hecho fue implementado mediante el uso de Bonita BPMS [20] y sus conectores de proceso, de manera tal de iniciar una nueva parte del proceso en otro motor una vez que la parte anterior fue finalizada. El resultado de esta implementación es un conjunto de nodos unidos por conectores de proceso en tiempo de ejecución, logrando así la puesta en marcha del modelo de proceso original [1] [2] [6].

V. MONITOREO DE PROCESOS EN SISTEMAS HÍBRIDOS

Tal como fue visto previamente en [7] y [8], los mayores problemas que surgen al utilizar un modelo de proceso particionado son, además de la ejecución propia del mismo, la

recolección y el monitoreo de las distintas instancias distribuidas (se encuentren en un sistema embebido o en el *cloud*), así como su visualización bajo la óptica del proceso original al cual pertenecen. Con este propósito, cada nodo de la arquitectura debe ser capaz de establecer comunicación con el siguiente de manera de iniciar nuevas instancias, y recolectar luego información sobre las mismas. Normalmente, dada una nueva instancia que fue iniciada en un nodo de la arquitectura, debería ser posible obtener no solo sus propios datos sino aquellos de cada instancia generada por esta en otro servidor [3] [4].

Interfaz centralizada

Tal como fue descripto con anterioridad en [8], es necesario desarrollar una aplicación de monitoreo de manera de mostrar datos en forma integrada en relación a las instancias distribuidas. Observando la traza de ejecución, resulta de importancia que cada instancia sea capaz de almacenar, no solo su propia información sino también de acceder a toda aquella asociada a las demás instancias iniciadas por ella en otros servidores. De esta manera, mediante el acceso a la primera instancia del proceso según el modelo de distribución, se hace posible recuperar la información asociada con el caso siguiente, y así en adelante hasta obtener la traza completa.

Una vez que la información es recuperada desde los distintos servidores, es necesario proveer una aplicación a cargo del proceso de recolección de datos y la visualización de los mismos en forma integrada. De esta manera, la aplicación web deberá concatenar la información recibida sobre las diferentes instancias y permitir a los usuarios observar los detalles de monitoreo de una manera transparente e integrada [3] [7] [8].

Arquitectura de la aplicación

La solución descripta en [8] se compone de tres nodos primarios: el *cloud*, los sistemas tradicionales o embebidos y la aplicación de monitoreo. El *cloud* trabaja como un contenedor de varios elementos: el BPMS, la aplicación de monitoreo y la API REST utilizada por los desarrolladores para integrar aplicaciones con el motor de procesos. Por otro lado se encuentran los componentes de tipo embebido, normalmente aplicaciones BPM tradicionales que pertenecen a la organización y por distintas razones tales como sensibilidad en los datos y portabilidad, las mismas no pueden ser extraídas de la infraestructura donde se hallan. El tercer componente se encarga de la función de monitoreo. Está a cargo de retornar información acerca de las instancias y sus actividades, las cuales fueron ejecutadas en cada nodo de la arquitectura [7] [8] [9].

Comunicación de los componentes

Se observan en dicha aplicación diversos componentes, así como diferentes perfiles de usuario involucrados en la ejecución de los mismos. Mientras que el rol preponderante en la ejecución del proceso es el participante de cada actividad, los resultados del sitio de monitoreo son importantes para el analista de negocio, así como para los administradores de la arquitectura que pueden optimizar servicios o componentes de proceso. Una característica en común entre la aplicación de ejecución de procesos y la de monitoreo es la transparencia en la ubicación. Los usuarios no deberían ser necesariamente notificados sobre los cambios en el ambiente de ejecución, en caso que estemos considerando un proceso descompuesto donde las actividades se ubican en diferentes servidores [8] [10].

VI. USO DE OBJETOS DE DATOS PARA EL MONITOREO DE PROCESOS

Fundamentos

En el campo de BPM el monitoreo es utilizado para observar el comportamiento de los procesos, así como para probablemente reaccionar a eventos y predecir futuros pasos del proceso durante su ejecución. Los procesos que son automatizados usando sistemas de información tales como motores de BPM, pueden se correctamente monitoreados debido a que dichos sistemas comúnmente ofrecen capacidades de mantener *logs*, y como consecuencia de esto el proceso puede ser fácilmente reconocido. En contraste, en ambientes donde los procesos deben ser ejecutados manualmente en una gran porción, como por ejemplo en una dependencia de cuidados de la salud, un gran número de eventos no se pueden capturar en forma automática. Un punto de monitoreo de eventos está relacionado a ciertos eventos capturados por un sistema IT conectado a una fuente específica (por ejemplo una base de datos o un lector de códigos de barra) y encargado de informar cuando ciertas transiciones de estado (por ejemplo habilitado, iniciado o finalizado) ocurren en una actividad de proceso. En este caso, "probabilístico" significa que es posible proveer un índice indicativo acerca del progreso del proceso, pero en este contexto esto resulta solo una aproximación.

Sin embargo, el reconocimiento del progreso del proceso en forma completa, así como una visión holística del mismo siguen siendo objetivos a lograr aún en ambientes de ejecución de procesos manuales. La introducción de más documentación y el registro de actividades durante la ejecución del proceso pueden ser una solución, aun cuando no es definitiva debido al esfuerzo extra que se introduce para los participantes del proceso, como por ejemplo los doctores y enfermeras en el caso anteriormente citado de la dependencia de salud. De esta manera, se introduce un enfoque que utiliza eventos extraídos de la creación o modificación de objetos de datos, incrementando de esta manera el número de eventos observables utilizados para el monitoreo y la predicción del progreso. Estos objetos son llamados eventos de datos de transición de estados.

Luego de grabar un objeto de datos con un estado específico asignado, podemos asumir su existencia. Además, discutiendo acerca del progreso del proceso, la registración de eventos ayuda a decidir acerca de futuras ejecuciones del mismo, la oportunidad de completarlo propiamente, o incluso la posibilidad de alcanzar una actividad determinada. Adicionalmente, esta aproximación puede ser utilizada para

identificar algún comportamiento incorrecto de una manera similar a lo que se realiza en el campo de la conformidad de datos [10] [11].

El enfoque presentado permite aproximaciones acerca de la ejecución del proceso basadas en información sobre los objetos de datos. Un objeto de datos OLC puede ser representado mediante una red de Petri, donde un nodo describe un estado de datos y una transición representa un cambio entre estos, desde el predecesor hacia el sucesor. Basándonos en este OLC, se seleccionan las transiciones que pueden monitorearse. Se considera un evento como un hecho que ocurre en un punto del tiempo particular, en un cierto lugar y con un contexto específico que se representa en el sistema IT. Las transiciones de estado de los datos que son observables han sido conectadas a los eventos que proveen información sobre el desencadenamiento de las mismas. Dicha conexión se realiza mediante la unión de las transiciones OLC particulares a una implementación que extrae información desde una fuente de eventos, incluyendo la correlación a un objeto de datos específico en tiempo de ejecución. Mientras se diseñan varios modelos de proceso, los objetos de datos se pueden asignar a actividades particulares, donde las mismas leen objetos de datos en un cierto estado, realizan inserciones en un estado de datos específico, o simplemente ejecutan transferencias de un estado a otro. Se asume entonces que usar un objeto de datos en un proceso se correlaciona correctamente con el OLC, lo cual significa que el proceso solamente usa los estados de datos específicos y sus transiciones. Sin embargo, no todos los objetos de estados de datos en el OLC necesitan ser reflejados en el modelo de proceso.

Sobre la base de la asignación de objetos de datos a actividades, es posible proveer información acerca de la ejecución del proceso en tiempo real. Es posible asumir que una actividad está habilitada si la misma puede ser ejecutada con las especificaciones de control de flujo, y los objetos de datos de entrada se encuentran disponibles en el estado requerido. Por otro lado se considera que una actividad se encuentra completada tan pronto como los objetos de datos de salida alcanzan un cierto estado. En tiempo de diseño, las transiciones de estado de datos se pueden marcar como "observables", haciendo referencia así a las capacidades de monitoreo antes mencionadas. En tiempo de ejecución se puede reconocer el progreso del proceso a través de eventos de transición de estados de datos que ocurren en un almacén determinado. Para obtener detalles no observables en el monitoreo de procesos a través de la manipulación de datos, el enfoque introducido debería ser combinado con otras técnicas complementarias pertenecientes a BI (*Business Intelligence*) o BAM (*Business Activity Monitoring*) [10] [11] [12].

Gestión de OLC en Cloud BPM

En términos de implementación, la aplicación de monitoreo debería ser capaz de recolectar información del estado del proceso, y al mismo tiempo, toda aquella información marcada como observable en las transiciones del mismo. Para realizar esto es necesario unir la información en la base de datos del motor de procesos con la información relacionada a los distintos objetos de datos OLC identificados a lo largo de la fase de definición. En la Fig. 1 es posible ver como los diferentes estados son atravesados por los eventos y la registración de información en el ambiente.

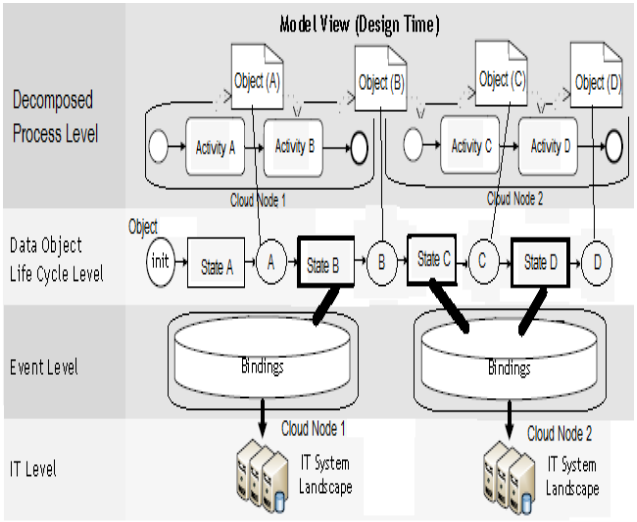

Fig.1. Vista del modelo descompuesto en tiempo real [6]

Es importante en este caso unir las actividades del proceso y las transiciones con los objetos OLC respectivos de manera de recolectar toda la información necesaria, y recuperar todos los ítems de una sola vez durante el proceso de monitoreo. La aplicación web que se ha considerado antes para el monitoreo debería ser modificada de manera de recolectar esta nueva información [8] [15] [16].

En la Fig, 2 se observa como las diferentes transiciones insertan información en el almacén de eventos durante el tiempo de ejecución. Es importante enlazar la información grabada con los detalles del proceso, de manera de obtener todos los eventos una vez que la aplicación de monitoreo esté realizando el proceso de recolección. Cada servicio que implementa las actividades del proceso debería ser responsable de guardar con los datos del negocio alguna información acerca de la ejecución del mismo (identificación de instancias, marcas de tiempo e identificación de actividades) en el almacén de eventos, de manera de ser capaz de recuperar todos los objetos OLC durante la fase de monitoreo tal como sea necesario. Esta es la manera más tradicional de integrar la lógica del proceso con los datos del negocio: usando persistencia de eventos como una manera de recuperar a posteriori la información asociada con el proceso ejecutado [6] [13] [18].

Uso de objetos OLC en el componente de monitoreo

Tomando en consideración la arquitectura previamente presentada en [8], el componente de monitoreo se forma esencialmente con dos definiciones de servicios web: *getInstanceService* y *getInstanceActivityService.* El primero,

está a cargo de recolectar información sobre instancias de un proceso determinado desde cada nodo en la arquitectura

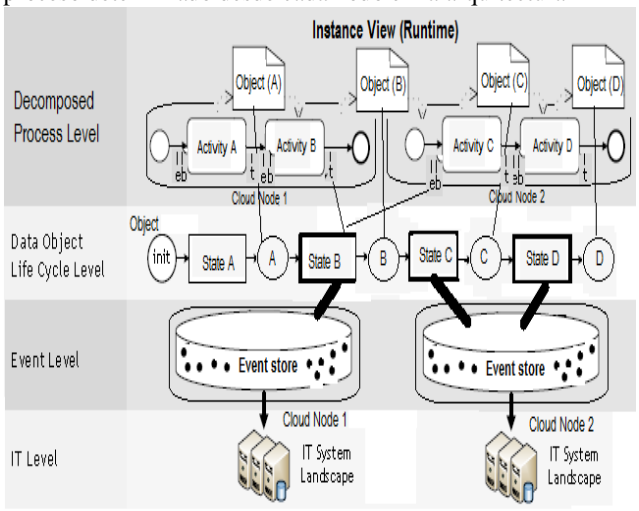

Fig.2. Vista modelo descompuesto en tiempo de ejecución [6]

híbrida, mientras que el segundo actúa una vez que el primero fue ejecutado, de manera de obtener detalles acerca de cada actividad asociada a la instancia seleccionada. Estos servicios en términos generales mantienen su comportamiento original, salvo que deberán ampliar su definición para considerar los objetos de datos OLC asociados al proceso. Esto significa, no solamente buscar información en las bases de datos del motor de procesos, sino acceder a las distintas fuentes en el almacén de eventos que hayan sido configuradas. Todos estos mecanismos deberían ser duplicados en cada nodo distribuido.

Un problema importante a ser considerado en este contexto es la visualización de los objetos de datos OLC, de manera de desplegarlos al usuario en forma útil. Los objetos OLC son realmente relevantes para la medición del proceso y el progreso del mismo debido a que están profundamente relacionados con la lógica del negocio y sus reglas. Analizar estos objetos provee una visión interesante acerca de la ejecución del proceso, su performance y progreso, pero dependen severamente del contexto de la organización. Así, la aplicación de monitoreo debería considerar una manera estándar de mostrarlos para desacoplar el mecanismo de visualización y la lógica de negocio, tal como lo han hecho las disciplinas de BAM (*Business Activity Monitoring*) y BI (*Business Intelligence*) [7] [14] [17] [19].

VII. Ejemplo de ejecución y monitoreo de procesos distribuidos utilizando eventos y objetos OLC

En el presente ejemplo se describirán las características de ejecución y monitoreo que se han tenido en cuenta para poder implementar un modelo de proceso distribuido en un ambiente híbrido, así como la posterior consideración de eventos y objetos OLC de manera tal de capturar información intermedia del flujo del proceso que pudiera resultar relevante para el monitoreo y la medición de indicadores. El proceso a considerar es un ejemplo desarrollado con fines de investigación y medición. El mismo ejecuta una Solicitud de Compra (correspondiente a una organización de mediano a gran porte, con varias etapas intermedias de elaboración y aprobación).

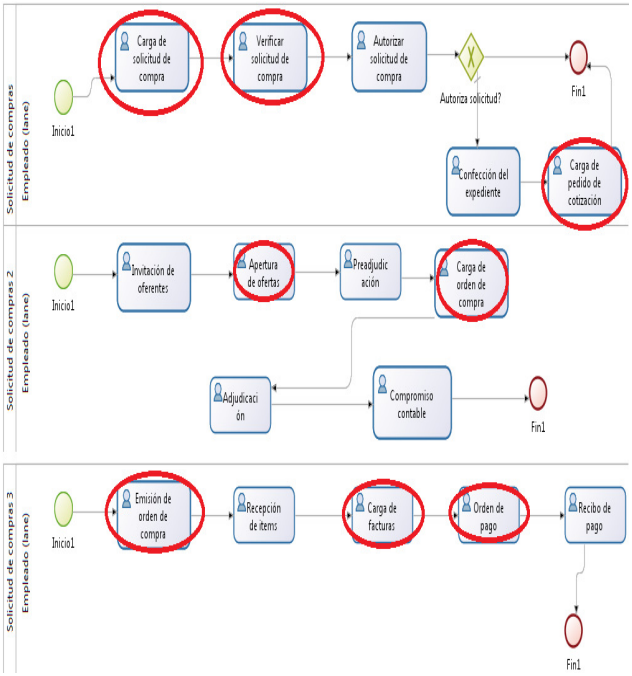

Fig.3. Estados observables en las particiones del proceso

Se decidió colocar dicho proceso en un entorno híbrido por las siguientes razones técnicas: en primer lugar, existían aplicaciones previas necesarias de ser integradas y ante la imposibilidad de migrar todas ellas a un único nodo de ejecución por motivos de portabilidad, se decidió separar el sistema en nodos de un *cloud* privado (donde no existen problemas de seguridad e integridad sobre los datos sensibles). De esta manera, al contar con tres nodos de tipo *cloud* se optó por separar el proceso en tres partes y sincronizar su ejecución y monitoreo a través de conectores.

Es necesario observar la importancia de ciertos eventos dentro de este proceso. El mismo atraviesa etapas, como la Confección de Órdenes de Pago o de los Recibos de pago, donde se generan eventos internos dentro de las actividades que pueden tener una separación cronológica previa a la compleción de las mismas. Esta característica hace que dichos eventos deban ser marcados como observables, y que los datos arrojados por dichas transiciones deban ser considerados por la aplicación de monitoreo.

Se observan en la Fig. 3 las porciones consideradas como observables en los diagramas de proceso descompuestos. Dichas porciones se encuentran contenidas en las particiones del modelo de descomposición, las cuales serán conectadas en tiempo de ejecución por medio de los conectores programados en Bonita, bajo la arquitectura de ejecución descripta en secciones anteriores. Durante la ejecución del conector se guardan los datos necesarios para poder recuperar las instancias durante la etapa de recolección. De esta manera, mediante el acceso a un servidor es posible recuperar toda la

traza de ejecución de las instancias distribuidas, y luego concatenar dichos datos con fines de monitoreo y mejora.

A su vez, como en este caso, se observa que muchos datos asociados al proceso se obtienen mediante la ejecución de los servicios, por lo que los motores de proceso no necesariamente poseen información de dichos eventos relacionados. De esta manera se nota la necesidad de guardar en los almacenes de eventos y datos corporativos referencias a las instancias de proceso, de modo tal de poder asociar dicha información con aquella buscada por los servicios web de recolección referente a la ejecución propia de actividades y compuertas. En términos de implementación, el proceso de Solicitud de Compra fue dividido en las siguientes tres particiones:

1) Desde la carga de Solicitud de compra hasta la Carga del pedido de cotización inclusive. Esta etapa significa recibir el pedido de los ítems que se desea comprar, observar la viabilidad del pedido en términos presupuestarios, creación del expediente interno y finalmente la formalización de los pedidos de cotización a los posibles proveedores oferentes.
2) Desde la Invitación a oferentes hasta el Compromiso contable inclusive. Esta segunda etapa implica la invitación a los diferentes proveedores que quieran participar de la licitación, la apertura de la misma, la selección del proveedor final, su carga en la Orden de compra (documento que formaliza la operación), la adjudicación de dicho proveedor y finalmente el compromiso contable de la operación económica (con una posible modificación presupuestaria).
3) La tercera y última etapa cubre desde la Emisión de la Orden de compra hasta la recepción de los ítems. En esta fase se reciben los productos, se los ingresa a stock y se emiten comprobantes como la Orden de Pago y el Recibo de pago correspondiente.

Reiteramos que tal como se enunció previamente, la razón por la que se ha dividido el proceso en estas tres etapas se debe a que los servicios necesarios de ser invocados corresponden a aplicaciones previamente construidas y localizadas en nodos de un *cloud* privado. Así, el proceso se pudo diseñar teniendo en cuenta dichas ubicaciones y haciendo uso óptimo de la distribución. Por otro lado, es posible notar que en diversos servicios dentro de cada partición del diseño existen transiciones importantes de ser consideradas como observables. Es posible ver que las mismas contienen información acerca de eventos manejados por los servicios, y que estos no necesariamente se guardan en el motor de procesos. Por esto existen datos del negocio que pueden resultar útiles para fines de monitoreo, pero que deben ser rastreados desde el almacén de eventos en cada uno de los nodos distribuidos.

La lista de actividades por partición que contienen transiciones observables son:

1) En la primer partición, la Carga de la solicitud de compra, la Autorización de la solicitud y la Carga del Pedido de cotización concentran estados de importancia para el monitoreo. La solicitud de compra puede atravesar los estados Creado, Aprobado, Anulado; luego puede ser Autorizada o Rechazada en la actividad de Autorización de Solicitud de Compra. Finalmente el Pedido de Cotización puede atravesar los estados Creado, Aprobado y Anulado.
2) En la segunda partición los estados observables son los correspondientes a las actividades de Apertura de la licitación (Creada, Aprobada, Anulada) y Carga de Orden de compra (Creada, Aprobada, Anulada).
3) Por último, en la tercera partición encontramos como estados observables aquellos correspondientes a las actividades Emisión de Orden de Compra, Carga de Factura y Carga de Orden de Pago (Todos con estados Creada, Aprobada, Rechazada).

De esta manera, para poder efectuar el seguimiento de los estados marcados como observables es necesario, en cada ítem de datos correspondiente (OLC), guardar una referencia a la instancia de proceso correspondiente. En nuestra implementación, los servicios web de recolección de datos (*getInstanceService* y *getActivityInstanceService*) fueron complementados con servicios distribuidos encargados de rastrear los datos correspondientes a los almacenes de eventos configurados. Así, al definir en la configuración un almacén de eventos sobre el cual observar datos, la aplicación de monitoreo ejecuta un servicio web propio encargado de conectarse a dicho almacén y luego durante la fase de recolección extraer los datos correspondientes a las tablas y columnas que se indicaron. De esta manera la aplicación de monitoreo se encarga de recolectar la información propia de los motores de proceso distribuidos en los servidores de la arquitectura, tomando una definición de proceso como parámetro. Para esto ejecuta el servicio *getInstanceService,* el cual se encarga de recuperar todas las instancias distribuidas correspondientes a los distintos servidores. Una vez recuperadas las mismas, es posible realizar una operación de desglose sobre las actividades internas utilizando el servicio web *getActivityInstanceService,* recuperando la información correspondiente a cada actividad definida en las particiones del proceso. Así, al desglosar una instancia se pueden recuperar los objetos de datos marcados como observables y presentarlos en indicadores útiles para el usuario. El enfoque de visualización de los indicadores debe ser lo suficientemente personalizado como para arrojar resultados trascendentes destinados a los usuarios y analistas del proceso. En el presente caso se han definido como necesarios los siguientes indicadores de monitoreo (con intervalos temporales acotados):

- Solicitudes de compra aprobadas para una instancia dada (pueden ser más de una), por un monto mayor a una suma parametrizada (para lo cual se hace necesario monitorear la tabla de Solicitudes de Compra en el nodo 1 de la arquitectura, en sus columnas de monto y estado).
- Pedidos de cotización aprobados para solicitudes de compra en una instancia dada, durante un período de tiempo determinado (observando la tabla de Pedidos de cotización en sus columnas fecha y estado).

- Detalle de estados atravesados por una orden de compra para una instancia dada en un intervalo de tiempo determinado.
- Modificaciones presupuestarias por un monto mayor a una suma parametrizada
- Facturas y órdenes de pago aprobadas correspondientes a Órdenes de compra superiores a un monto determinado en un período de tiempo determinado.

En el conjunto de indicadores anteriores es posible notar que algunos de ellos son propios del monitoreo de las instancias y surgen de su relación con los datos asociados, así como existen otros que tienen naturaleza de reporte sobre los datos de ejecución de las instancias finalizadas. En este caso es necesario observar que existen procesos como el anteriormente citado que guardan una estrecha relación con los datos que generan a su paso. Así, monitorear el Proceso de Compras no significa únicamente observar la fecha de inicio y finalización de sus actividades internas, así como su actividad y participante actual, sino que puede significar observar cuál fue el último comprobante generado en dicho proceso, en qué estado se encuentra el mismo actualmente, y si además supera algún parámetro (monto, fecha) que hubiera sido de interés de seguimiento para la organización.

Incorporando marcas del proceso a los datos corporativos (es decir, guardando identificadores de instancias de proceso en las tablas corporativas) se posibilita la recolección de los mismos desde la aplicación de monitoreo, así como se incrementa la capacidad de someterlos a trabajos de análisis que permitan obtener marcas de ejecución e indicadores relevantes para los usuarios y analistas del negocio.

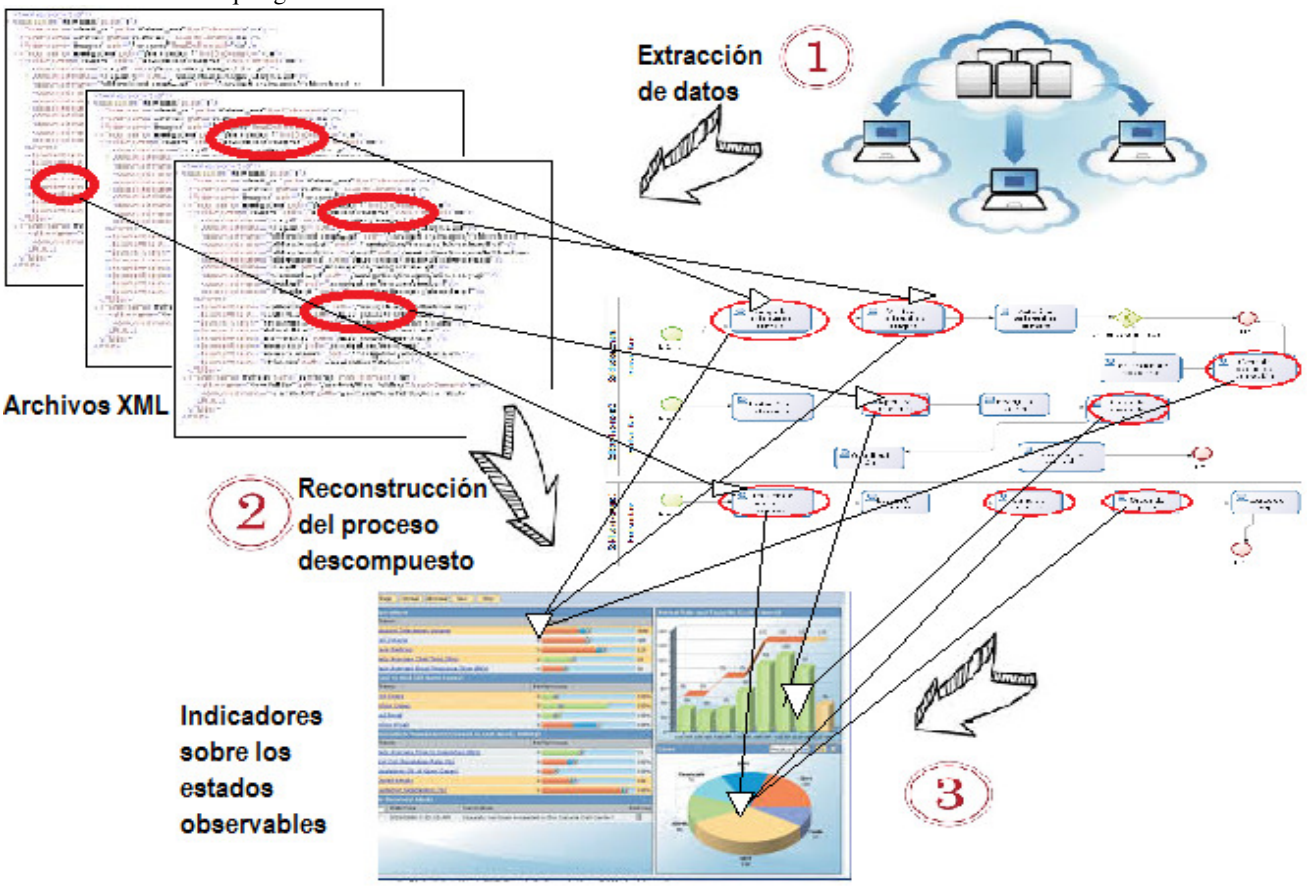

Fig.4. Reconstrucción y monitoreo a partir de la extracción de datos del sistema híbrido

En la Fig. 4 se ilustra el proceso de reconstrucción y monitoreo de un proceso descompuesto en el *cloud*. A partir de la aplicación de técnicas de monitoreo se extraen datos de los servidores distribuidos. Dicha extracción es realizada por los servicios web antes descriptos, y arroja como resultado una serie de archivos XML necesarios de concatenar. Una vez concatenados los mismos, es posible reconstruir el modelo de proceso original y desplegarlo al usuario. Es posible notar como en la figura citada aparecen resaltados los estados observables dentro de los archivos recuperados durante el período de extracción, y cómo los mismos se correlacionan con los estados marcados dentro del modelo de proceso. De esta manera, en los archivos XML figurarán los rastros recuperados que corresponden a los objetos OLC marcados como observables.

Dichos objetos son utilizados luego para construir las tablas y gráficos correspondientes a los reportes e indicadores definidos. Así vemos como por ejemplo, para una solicitud de compra es posible monitorear la actividad "Solicitud de compra" del proceso, y además refinar el monitoreo de la misma mediante el despliegue del indicador de Solicitudes de Compra aprobadas para una instancia dada y con una suma parametrizada (para observar por ejemplo el crecimiento o decrecimiento de las mismas en comparación a períodos de tiempo anteriores). De esta manera, la actividad del proceso se podrá reconstruir con los datos recuperados directamente desde los motores distribuidos mediante los servicios web de recolección, mientras que los datos de la Solicitud de Compra que hayan sido marcados como observables se tomarán desde los distintos objetos OLC indicados, luego concatenados y finalmente desplegados en forma de gráfico para su análisis y mejora. En adición, los indicadores pueden mostrar los estados que ha atravesado cada objeto OLC (aún dentro de una misma actividad de proceso) discriminados por fecha, y observar las transiciones de los mismos dentro de la actividad.

VIII. CONCLUSIONES

En el presente trabajo se ha elaborado una recopilación de los principales conceptos que se presentan en la actualidad en relación a la descomposición de procesos de negocio para su ejecución en un ambiente híbrido, así como el monitoreo de los mismos y el uso complementario de objetos de datos OLC para la obtención de indicadores útiles en la medición y mejora de procesos. La cualidad de marcar datos y transiciones como observables permite a las herramientas de monitoreo ejecutar el proceso de extracción tendiente a obtener y concatenar información necesaria para la medición y mejora. Esta información recuperada debe haber sido enlazada previamente mediante algún mecanismo con los datos del proceso en cuestión.

En términos del problema que se ha abarcado en el presente trabajo, se presentan una serie de interrogantes que intentarán ser contestados en futuras publicaciones, a saber:

1) ¿Es posible modificar la notación de diseño de procesos (BPMN) de manera tal de indicar en forma automática el criterio de descomposición para los procesos, evitando así la interacción humana en el despliegue de los mismos sobre la arquitectura distribuida?
2) ¿Es posible indicar los datos y transiciones observables a través de elementos del diagrama de proceso para así evitar las configuraciones posteriores necesarias en la fase de recolección de dichos elementos?
3) ¿Es posible generalizar el mecanismo de ejecución y monitoreo de procesos descompuestos de manera tal de utilizar solo lenguajes estándar y no circunscribir esta metodología a determinados BPMS en el mercado?

De esta manera, mediante la resolución de los interrogantes arriba citados se indican los criterios necesarios para la ejecución y monitoreo de procesos (incluido el uso de OLC) durante el tiempo de diseño. Así, además de estandarizar los criterios utilizados, se facilita la creación de modelos cohesivos que contengan toda la información necesaria para la ejecución distribuida y el monitoreo de los mismos.

IX. REFERENCIAS

[1] Zhenyu Fang, Changqing Yin, "Intelligent Information Management", 2010, 2, 329-333 doi:10.4236/iim.2010.25039 Published Online (http://www.SciRP.org/jou rnal/iim) Copyright © 2010 SciRes. IIM "BPM Architecture Design Based on Cloud Computing". May 2010.

[2] Gregor Srdić, Matjaž B. Jurič. Proceedings of the 1st International Conference on CLoud Assisted ServiceS Bled. "BPM and iBPMS in the Cloud Aleš Frece". 25 October 2012.

[3] Luis Ferreira Pires, Luiz O. Bonino da Silva Santos. 2012 IEEE 16th International Enterprise Distributed Object Computing Conference Workshops. "Towards a BPM Cloud Architecture with Data and Activity Distribution". Evert F. Duipmans. May 2012.

[4] Cristina Cabanillas, Anne Baumgrass, Jan Mendling, Patricia Rogetzer, and Bruno Bellovoda "Towards the Enhancement of Business Process Monitoring for Complex Logistics Chains". Vienna University of Economic and Business. July 2013.

[5] Nico Herzberg and Andreas Meyer. "Improving Process Monitoring and Progress Prediction with Data State Transition Events". Plattner Institute at Potsdam University. May 2013.

[6] Nico Herzberg, Andreas Meyer, Oleh Khovalko, and Mathias Weske. "Improving Business Process Intelligence with Object State Transition Events". Plattner Institute at Potsdam University. July 2013.

[7] José Martinez Garro, Patricia Bazán. "Monitoreo de procesos en el cloud. Una propuesta arquitectónica". JCC Universidad de Temuco. Chile. November 2013.

[8] José Martinez Garro, Patricia Bazán. "Constructing and monitoring processes in BPM using hybrid architectures". IJACSA Journal. London. February 2014.

[9] M Mevius, R. Stephan, P. Wiedmann, "Innovative Approach for Agile BPM", eKNOW 2013: The Fifth International Conference on Information, Process, and Knowledge Management, 2013.

[10] H Sakai, K Amasaka. "Creating a Business Process Monitoring System "A-IOMS" for Software Development". Chinese Business Review, ISSN 1537-1506 June 2012, Vol. 11, No. 6, 588-595.

[11] M. Gerhards, V. Sander, A. Belloum, "About the flexible Migration of Workflow Tasks to Clouds -Combining on and off premise Executions of Applications", CLOUD COMPUTING 2012: The Third International Conference on Cloud Computing, GRIDs, and Virtualization, 2012.

[12] E Duipmans, Dr. L Ferreira Pires, "Business Process Management in the cloud: Business Process as a Service (BPaaS)", University of Twente, April, 2012.

[13] JP Friedenstab, C Janieschy, M Matzner, O Mullerz. "Extending BPMN for Business Activity Monitoring". University of Liechtenstein, Hilti Chair of Business Process Management, Vaduz, Liechtenstein. September 2011.

[14] M Reichert, J Kolb, R Bobrik, T Bauer. "Enabling Personalized Visualization of Large Business Processes through Parameterizable Views". Hochschule Neu-Ulm, Neu-Ulm, Germany. November 2011.

[15] J Kolar, T Pitner, "Agile BPM in the age of Cloud technologies", Scalable Computing: Practice and Experience, 2012.

[16] A Lehmann and D Fahland , "Information Flow Security for Business Process Models - just one click away", University of Rostock, Germany, 2012.

[17] R Accorsi, T Stocker, G Müller, "On the Exploitation of Process Mining for Security Audits: The Process Discovery Case", Department of Telematics, University of Freiburg, Germany, 2012.

[18] A Frece, G Srdić, MB. Jurič, "BPM and iBPMS in the Cloud", Proceedings of the 1st International Conference on Cloud Assisted ServiceS, Bled, 25 Octubre 2012

[19] S Zugal, J Pinggera and B Weber. "Toward enhanced life-cycle support for declarative processes". JOURNAL OF SOFTWARE: EVOLUTION March 2012

[20] Bonita Open Solution http://es.bonitasoft.com/. August, 2014.