# Versionado de Soluciones webMethods

3/3/2017 – v 1.1

El presente documento es propiedad de MAPFRE y es exclusivamente para uso interno o de cualquiera de las Entidades del Grupo MAPFRE (listado completo en la página [www.mapfre.com](http://www.mapfre.com)). No podrá ser reproducido total o parcialmente, ni procederse a su transmisión de ninguna forma, ya sea electrónica, mecánica, por fotocopia, grabación, reproducción u otros medios sin autorización expresa al efecto.

Control de cambios de la plantilla

**NOTA: Esta sección no puede ser modificada. Los cambios de versión del documento se recogen en la siguiente tabla de control de cambios.wm\_mws stopwm\_mws stop**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Fecha** | **Versión MpM** | **Resumen de los cambios** |
| 1.0 | 29/02/2016 |  | Creación del documento |
|  |  |  |  |
|  |  |  |  |

|  |
| --- |
| **Comentarios** |
|  |

Control de cambios del documento

|  |  |
| --- | --- |
| **Información del producto** | |
|  | **Nombre** |
| **Proyecto** |  |
| **Módulo** |  |
| **Aplicación** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Histórico de Versiones** | | | |
| **Versión** | **Fecha** | **Resumen de los cambios** | **Autor** |
| 1.0 | 29/2/2016 | Versión Inicial | Área Integración - BPM |
| 1.1 | 3/3/2017 | Consideración de uso de webMethods Business Rules | Área Responsable BPM |

|  |
| --- |
| **Comentarios** |
|  |

## Índice

[Versionado de Soluciones webMethods 1](#_Toc475701110)

[1 Índice 2](#_Toc475701111)

[2 Descripción del documento 3](#_Toc475701112)

[2.1 Propósito 3](#_Toc475701113)

[2.2 Audiencia 3](#_Toc475701114)

[3 Visión General 4](#_Toc475701115)

[3.1 Aspectos generales a considerar 4](#_Toc475701116)

[3.2 Modelo de componentes de las soluciones 4](#_Toc475701117)

[3.3 Portlets de Aplicación 5](#_Toc475701118)

[3.4 Interfaz de usuario de las tareas (portlets) 6](#_Toc475701119)

[3.5 Servicios de aplicación invocados desde los portlets 7](#_Toc475701120)

[3.6 Servicios invocados como tareas automáticas del proceso 8](#_Toc475701121)

[3.7 Procesos y subprocesos 9](#_Toc475701122)

[3.8 Organización del proyecto y nomenclatura 13](#_Toc475701123)

[3.8.1 Procesos 14](#_Toc475701124)

[3.8.2 Procesos asociados a los callactivity 14](#_Toc475701125)

[3.8.3 Servicios 14](#_Toc475701126)

[3.8.4 Reglas de Negocio 15](#_Toc475701127)

[3.8.5 Portlets 15](#_Toc475701128)

[3.8.6 Tareas 15](#_Toc475701129)

## Descripción del documento

### Propósito

El presente documento describe los aspectos a considerar en el versionado de los componentes de una solución de negocio de MAPFRE basada en la tecnología webMethods.

### Audiencia

Este es un documento de trabajo destinado a los equipos que van a construir cambios sobre soluciones BPM en MAPFRE.

## Visión General

Como norma general, los cambios que se aborden en las soluciones, deben ser retro-compatibles, esto es, que no sea necesario tener al mismo tiempo dos versiones del mismo componente en ejecución.

Sólo en el caso en el que sea estrictamente necesario, y por requisito expreso de negocio, se abordará el versionado de componentes con ese fin. En este sentido, cobra especial relevancia el componente de tipo proceso, ya que, dada su latencia, es el más susceptible de necesitar diferentes versiones activas al mismo tiempo en ejecución. Y, como resultado de esto, se deriva también la necesidad de mantener diferentes versiones en ejecución de los componentes del modelo que interactúan con los procesos.

Como es lógico, la identificación de los componentes de las soluciones que podrían verse afectados en un versionado, dependerá de la naturaleza del cambio implementado.

A continuación se describen los aspectos generales a tener en cuenta en la gestión de cambios en las soluciones, así como el modelo de componentes de las mismas, a partir del cual se identifican las piezas componentes susceptibles de ser versionadas.

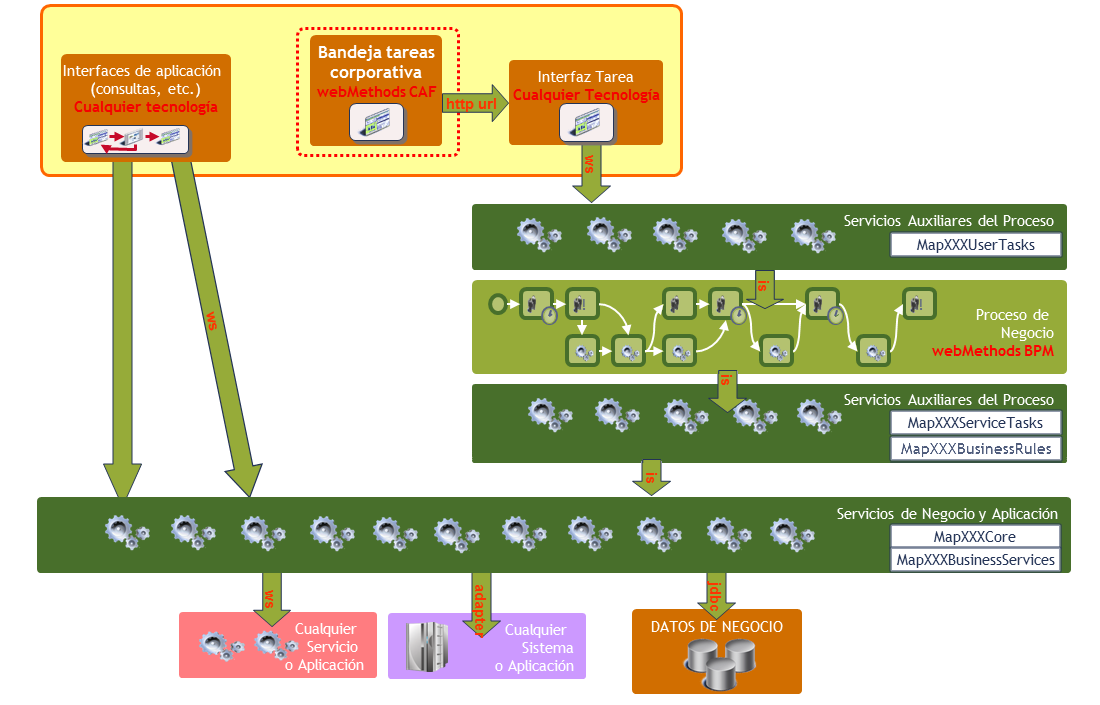
### Aspectos generales a considerar

Evidentemente, lo más sencillo es aplicar las modificaciones necesarias sobre el componente existente y no versionarlo. Todo dependerá de los requisitos concretos de negocio que se tengan que abordar.

Para analizar el cambio a realizar sobre la solución, y antes de determinar la estrategia a seguir, se deben evaluar aspectos como la incompatibilidad hacia atrás de los cambios incorporados y la complejidad del resultado final tras el cambio, considerando la latencia de las instancias vivas de versiones anteriores del proceso y la facilidad para la supresión de funcionalidades obsoletas.

### Modelo de componentes de las soluciones

En el siguiente diagrama se muestra el modelo típico de componentes de una solución de negocio basada en el BPM webMethods:



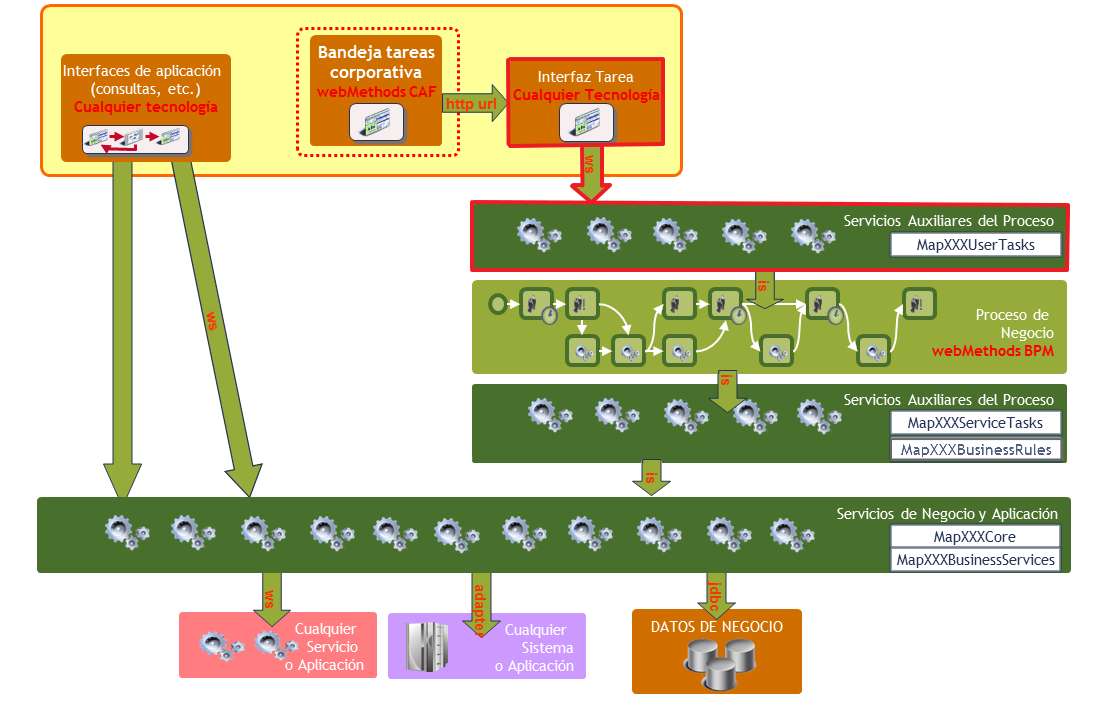
Como resultado de la gestión del cambio en una solución, podría determinarse el versionado de cualquiera de los componentes presentados en el diagrama anterior, en cuyo caso se deberían tener en cuenta determinados aspectos, en función del tipo de componente.

### Portlets de Aplicación

A los portlets de aplicación se accede normalmente desde el menú situado en la parte izquierda del Escritorio de Procesos, que se forma dinámicamente a partir de la información proporcionada por la propia solución en el documento XML que contiene los metadatos de dicha solución necesarios para su funcionamiento en el Escritorio.

Al no interactuar directamente con los procesos, no parece necesario mantener dos versiones diferentes de este tipo de componentes al mismo tiempo en ejecución.

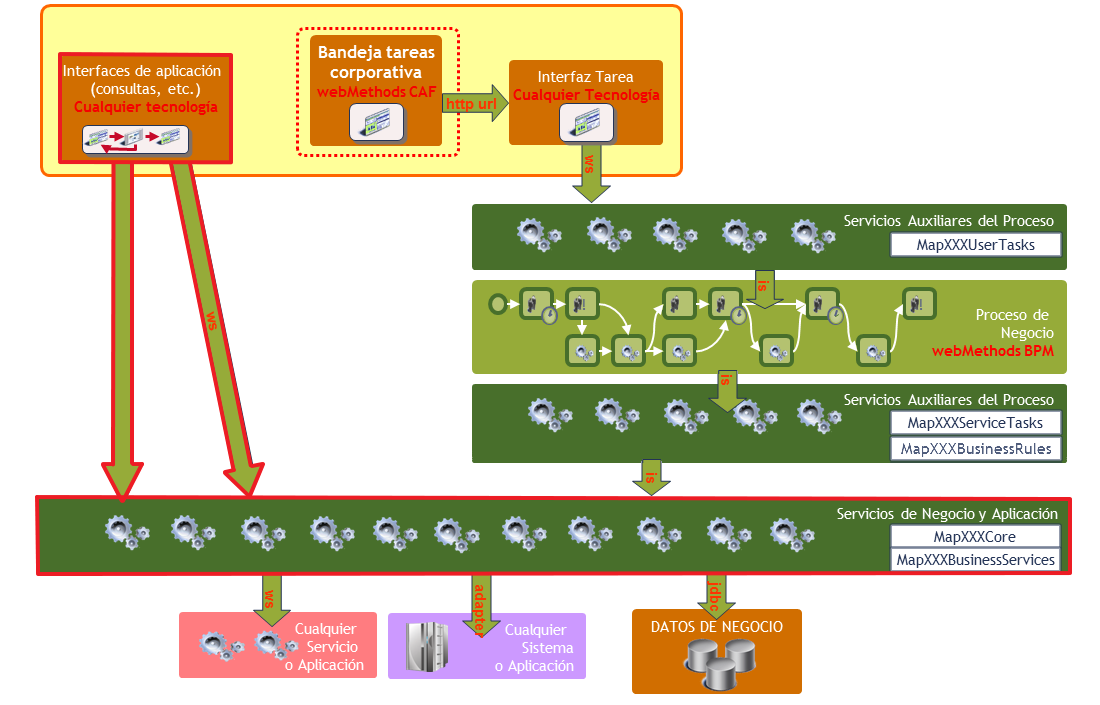
### Interfaz de usuario de las tareas (portlets)

Si, como resultado de una nueva versión de proceso, se cambian los datos de entrada/salida de alguna de sus tareas manuales, la interfaz de usuario de las mismas (portlets CAF, frontal ArqOS, etc.) se verá significativamente afectada.

La gestión de dos versiones o vistas diferentes de la interfaz puede tener que ser soportada mediante la implementación de un nuevo portlet o componente de interfaz para la nueva versión, que permita tener ambos en ejecución al mismo tiempo, y enlazando con ellos desde la Bandeja de Tareas Corporativa mediante la URL contenida como atributo en la propia tarea.

Como quiera que ese atributo se define en el encolado de la tarea por parte del proceso, será en ese momento cuando se deba determinar la URL en función de la versión activa del proceso que la genera (*processModelVersion*). Para diferenciar un portlet de otro ver apartado de *Organización del proyecto y nomenclatura*.

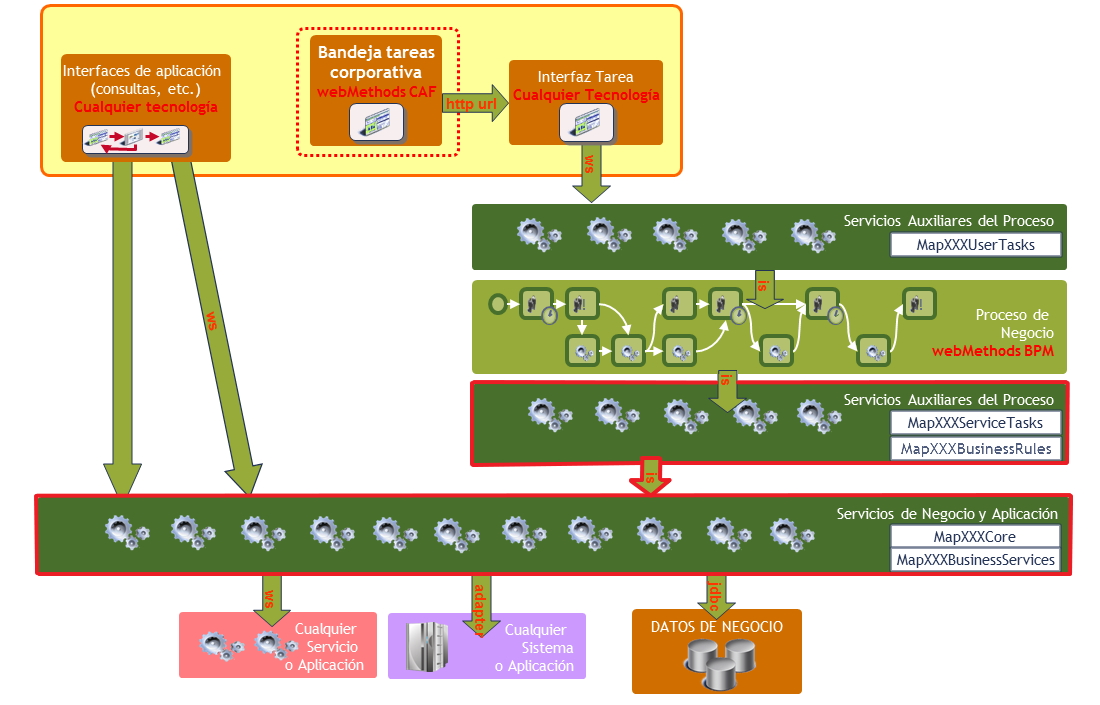
### Servicios de aplicación invocados desde los portlets



Para los servicios de aplicación invocados desde los componentes de interfaz de usuario (tanto de aplicación, como de tareas) no debería ser necesario mantener diferentes versiones funcionando al mismo tiempo.

Siempre que fuera posible y no se pusiera en peligro el buen diseño de los componentes, los nuevos servicios deberían ser compatibles con sus consumidores existentes. Si no fuera posible y existiera incompatibilidad, se deberán abordar conjuntamente los cambios en los servicios y en los componentes consumidores que se vieran impactados.

### Servicios invocados como tareas automáticas del proceso



Los Servicios IS de aplicación o de negocio, que dan soporte a las tareas automáticas del proceso, así como las Tablas de Decisión de las Reglas de Negocio, invocadas directamente por los pasos de un modelo de proceso, también pueden verse afectados por la existencia de diferentes versiones del proceso conviviendo en ejecución.

Tal como se indica en el apartado de *Organización del proyecto y nomenclatura* estos servicios se deben distinguir con diferentes namespaces(carpetas) dentro del mismo paquete, siempre que se trate del mismo proceso.

La existencia en las soluciones de un paquete llamado *MapXXXServicesTasks*, donde se ubican los servicios que se corresponden con los pasos automáticos del proceso, sirve, entre otras cosas, para el desacoplamiento operativo entre los procesos desplegados en el motor BPM y los servicios de aplicación/de negocio. En consecuencia, es la ubicación ideal para gestionar la invocación a diferentes versiones de dichos servicios en función de la versión de la instancia del proceso de negocio que los invoca.

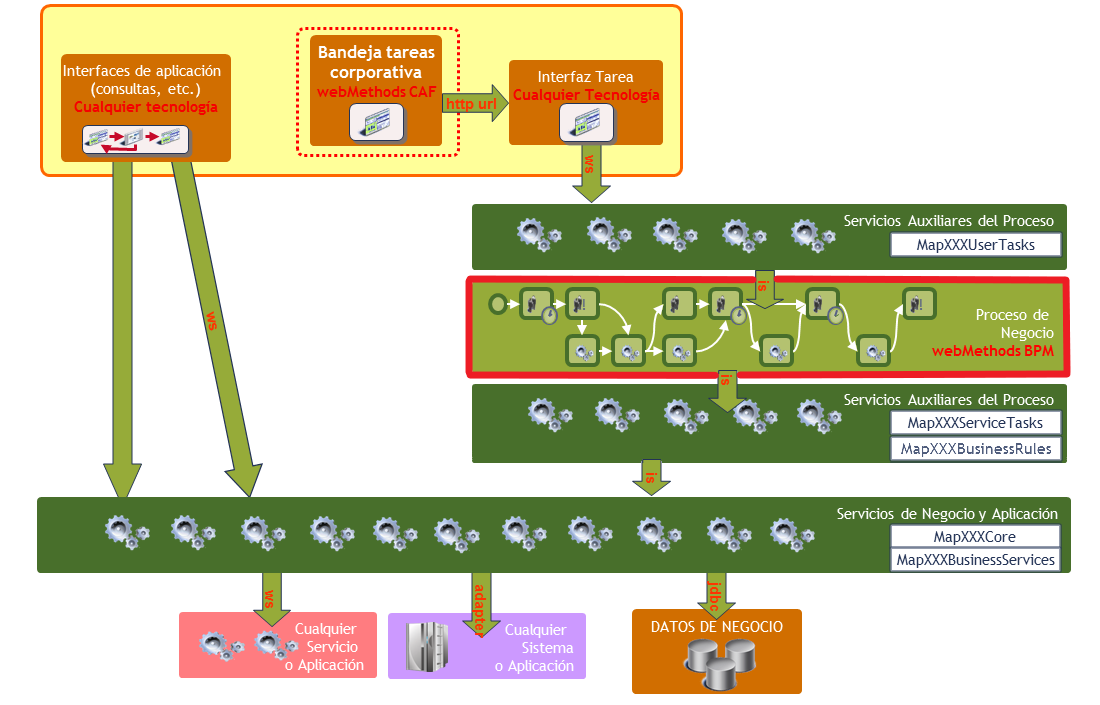
Esta versión se puede obtener vía *ProcessData (*processModelVersion*)*, ya que los servicios contenidos en ese paquete, invocados directamente desde el proceso, pueden acceder a la información de contexto del mismo.

Si se decide la existencia de servicios diferentes para cada versión, se deberán crear los nuevos de tal manera que su namespace sea diferente (ver apartado de *Organización del proyecto y nomenclatura*), e incluir en los servicios que se corresponden con las tareas automáticas del proceso la lógica de bifurcación necesaria para invocar a unos o a otros.

Para las Reglas de Negocio, los Modelos de Datos de las Tablas de Decisión pueden ser sincronizados con sus Document Types correspondientes, en caso de que sufran algún cambio que sea *compatible* con el modelo de datos existente.

En el caso de ser necesario versionar una Regla de Negocio, debido a que la naturaleza de los cambios sean incompatibles dentro de las Tablas de Decisión existentes, se recomienda crear, en el mismo Proyecto de Reglas, nuevas Tablas de Decisión y/o Modelos de Datos para las nuevas versiones, de manera análoga a los servicios IS, quedando *no recomendado* implementar decisiones en base al valor de processModelVersion, ya que las Tablas de Decisión son mantenidas directamente por Negocio, y el campo de versión de modelo de proceso es de carácter técnico y no debería de estar accesible en las Reglas de Negocio.

### Procesos y subprocesos



Hay que tener en cuenta, que, cuando se deban realizar modificaciones sobre un proceso, se pueden dar dos situaciones finales diferentes respecto al modelo de proceso existente en producción, a la hora de desplegarlo:

* **Nueva versión del proceso (versionado)**: en este caso se desarrollarán los cambios necesarios en el proceso y se desplegará en el motor de procesos, de forma que pueden llegar a convivir ambas versiones desplegadas.
* **Modificaciones en un proceso (evolución)**: En este caso se desarrollarán los cambios en el mismo proceso y se desplegarán con la misma versión, de forma que el nuevo proceso sustituye al anterior. En éste caso, todas las instancias que se encuentren vivas funcionarán conforme a las modificaciones que se hayan realizado.

Dependiendo de la naturaleza del cambio, de la compatibilidad hacia atrás del mismo, y de que pudiera existir la necesidad de seguir creando instancias de una versión anterior del modelo, se podrá decidir el tipo de cambio a realizar.

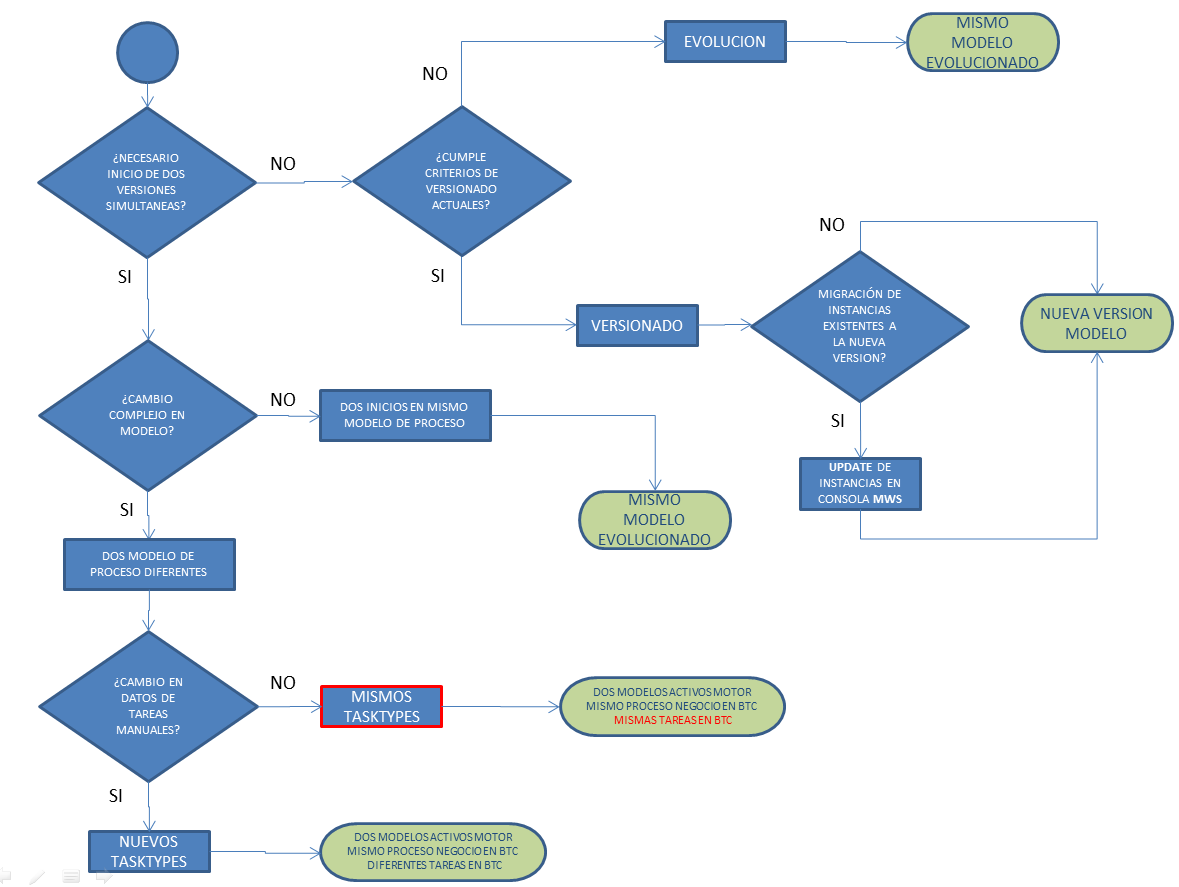
El hecho del tipo de modificación a realizar en el proceso, debe plantearse teniendo en cuenta también los cambios necesarios en el resto de elementos que son utilizados en el proceso. Es decir, hay que plantearse también que tipo de cambio se requiere en los elementos de tipo Tareas Humanas, Portlet de las tareas, servicios a los que se invoca desde las tareas automáticas, documentos que se tratan a lo largo del proceso, y en especial los subprocesos a los que se invoque cuando lo que se quiere versionar es un proceso padre.

El tipo de cambio respecto al modelo de proceso, puede desencadenar finalmente en:

* Nueva versión del modelo
* Nueva forma de iniciar el mismo modelo, soportada por la lógica del mismo, de tal manera que en el mismo proceso se contemple la funcionalidad nueva y la anterior si hiciera falta.
* Nuevo modelo independiente, esto es, dos procesos diferentes
  + Con este enfoque habría que reflexionar ante la posibilidad de que una misma tarea manual pudiera existir en ambos procesos, y la forma en la que esto debería ser soportado por la BTC. Inicialmente se debería tratar de otro tipo de tarea nuevo.

En el caso de que existan procesos hijos, el cambio a realizar abarcará también la decisión de si necesita modificar el padre o el hijo o ambos.

Inicialmente se podría tener como apoyo, el siguiente flujo de decisión, a la hora de decidir los tipos de cambios a realizar en los elementos afectados del proceso y de ahí decidir el tipo de cambio en el proceso.



El detalle de las condiciones a tener a en cuenta, paraun mejor entendimiento del diagrama, es:

|  |  |
| --- | --- |
| **Condición a chequear** | **Descripción** |
| Necesario inicio de dos versiones simultáneas | Si se necesitan en producción a la vez dos “versiones” funcionando, porque se deban tener que iniciar instancias con el modelo de negocio anterior y otras con el nuevo. |
| Cambio complejo del modelo | Se pueden chequear los cambios requeridos en el modelo, se puede verificar si cambian datos de entrada/salida a tratar, decisiones muy diferentes a lo largo del proceso que impliquen cambios en las bifurcaciones que compliquen el modelo en exceso, etc..  Puede tenerse la alternativa de dos inicios para un mimos proceso para tratar en cada caso los datos de entrada si son diferentes |
| Cambio en datos de tareas manuales | Se pueden chequear los tipos de cambios que se necesitan sobre las tareas manuales existentes y si son incompatibles. |
| Cumple criterios de versionado actuales | Hay algunas reglas básicas, a tener en cuenta, que pueden hacer necesario versionar, de cara a que las instancias que pueda haber en producción anteriores al cambio, no se queden incongruentes (que no sean capaces de transitar por sí mismas) o por el contario es mejor evolucionar el proceso actual.  Algunas importantes que pueden hacer obligatorio versionar son:   * Desaparece una tarea de usuario del proceso * Desaparece una tarea automática o un camino del proceso, en el que podría haber instancias paradas   El resto de cambios inicialmente hacen posible que se realiza la evolución. |
| Migración de instancias existentes a la nueva | Se debe chequear si es posible, que las instancias actuales de producción puedan ser migradas a la versión nueva o los cambios son incompatibles y se deben conservar con la versión anterior, para decidir la acción a realizar (migrar instancias o no) |

El detalle de las posibles acciones resultantes a realizar es:

|  |  |
| --- | --- |
| **Acción a realizar** | **Descripción** |
| Mismo modelo evolucionado | Se realizan los cambios en el mismo modelo y se despliega |
| Nueva versión modelo | Se realizan los cambios en la nueva versión del proceso y se despliega, decidiendo si se migran instancias vivas o no. |
| Mismo modelo con dos inicios | Dos inicios diferentes en el modelo actual bifurcando los caminos según las necesidades de cambio |
| Dos modelos activos en el motor mismo proceso de negocio en BTC, mismas tareas | Se crea un nuevo modelo de proceso para recoger la nueva funcionalidad requerida y convivirá en producción con la actual. En algún punto se decide qué modelo instanciar. Este caso y el siguiente, podrían ser los casos para “versionar” subprocesos, en el caso de tener que invocar dos “subprocesos” hijos diferentes dependiendo de los datos de negocio del padre. |
| Dos modelos activos en el motor mismo proceso de negocio en BTC, diferentes tareas | Mismo caso anterior pero creando tareas nuevas para recoger la nueva funcionalidad al tener cambios incompatibles en las tareas |

Hay que tener en cuenta, que se trata de tener una base para la toma de decisiones a la hora de abordar los cambios en un modelo de proceso, pero los cambios a analizar dependen, en gran medida, de los requisitos de negocio por los que se cambia un proceso.

### Organización del proyecto y nomenclatura

Si se deben crear nuevos componentes, derivados de la decisión que se haya adoptado al versionar la solución, se debe tener clara la organización del proyecto y la nomenclatura especifica para los nuevos componentes, al tratarse realmente elementos semejantes a los existentes pero “versionados”.

De cara a que el impacto en nomenclatura sea el mínimo posible, se aconseja utilizar el término <función> que consta y da flexibilidad a la nomenclatura actual en varios tipos de componentes, y así extender su “filosofía” al versionado de componentes.

La utilización de ésta <función> en estos casos, tendría como objetivo, identificar la “versión del componente”, ya sea con terminología de negocio si tienen sentido o con el propio número secuencial de versión que genera el cambio. Esta decisión se deja abierta a la decisión de la cada Solución, ya que se considera que puede tener dependencia del cambio a realizar.

A continuación se revisa nomenclatura por cada tipo de componente a versionar.

#### Procesos

Se utilizará el versionado propio del proceso por lo que no se necesita reorganizar el proyecto ni aplicar nomenclatura especifica. El nombre del proceso no varía.

#### Procesos asociados a los callactivity

Si al final se ha debido crear un nuevo modelo de proceso (para el caso en concreto de un callactivity), por ser la solución a su versionado, la nomenclatura asociada al nuevo modelo de proceso se aplicará la misma inicialmente

XXX\_YYY\_#\_#\_***<NombreProceso>***

donde el NombreProceso cuando se trata de callactivity suele contener el nombre de la tarea que implementa, por lo que en el nuevo proceso, su propio nombre debe indicar la diferencia funcional existente entre un modelo y otro, y si la causa fuese meramente técnica se podrá incorporar el nombre de la versión del proceso padre, por ejemplo v1, v2, etc..., esto se incluiría al final. Es decir seria:

XXX\_YYY\_#\_#\_***<NombreProceso><función>***

El concepto de función se utiliza ya en la nomenclatura actual para englobar funcionalmente los servicios.

#### Servicios

Dentro de cada paquete por nomenclatura siempre se ha podido utilizar el término <funcion> (*ver nomenclatura oficial publicada*), para separar las funcionalidades o lo que se considerase oportuno. Si se ha tenido que versionar un servicio se debe utilizar esta <funcion> para diferenciar el servicio, por lo que el nuevo tendrá siempre un namespace diferente. El servicio puede permanecer de esta manera en el mismo paquete, para todo tipo de servicios.

Dependiendo del motivo del versionado, se puede sustituir función por algún término con significado de negocio o simplemente asociarlo al número de versión del proceso. Lo aconsejable es realizar de la misma forma para todos los elementos, de forma uniforme en todo el proyecto.

En la nomenclatura actual ya se menciona esta división de la siguiente manera:

***com.mapfre.<Id-Sol>.<funcion>***

Esto sería aplicable independientemente del paquete en el que se encuentre el servicio a versionar.

#### Reglas de Negocio

Como norma general se puede mantener un único Proyecto de Reglas por aplicación, ya que su contenido es el que será versionado, principalmente Modelos de Datos y Tablas de Decisión, siendo de aplicación los mismos criterios utilizados para la nomenclatura de servicios IS, esto es, dependiendo del motivo del versionado, sustituir <función> por algún término con significado de negocio o simplemente asociarlo al número de versión del proceso, manteniendo un criterio uniforme para todo el proyecto.

#### Portlets

Si la solución pasa por “versionar” los portlets, y siempre que correspondan con la misma tarea, es decir, el nombre de la tarea que se identifica también en el nombre del portlet, es el mismo, se debe poder distinguir un portlet de otro.

Los nuevos puede ubicarse en un proyecto de portlets independiente similar a:

Map***<Id-Sol><NombrePortlet><función>***

Donde función sigue la misma filosofía que en otros componentes, concepto funcional o versión del proceso por la que se realiza el nuevo portlet.

De esta manera, la url a formar del portlet se distingue entre:

***Portlet/<Id-Sol><<NombrePortlet>\_\_<NombreTarea>***

***Portlet/<Id-Sol><<NombrePortlet>< función >\_\_<NombreTarea>***

#### Tareas

Dependiendo de la magnitud de los cambios, si ha sido necesario crear nuevas tareas pero asociadas a la misma tarea ARIS. Estarán contenidas dentro del proyecto del proyecto de tareas correspondiente Map***<Id-Sol>***ProcessesTasks.

Las tareas normalmente denominadas:

XXX\_YYY\_#\_#\_SecuencialN ***<NombreTarea>***

Podrán ser divididas en:

XXX\_YYY\_#\_#\_SecuencialN\_[a-Z]\_***<NombreTarea>***

De la misma manera que se realiza con las tareas de servicios cuando se dividen por motivos técnicos.