Health Check: Business Automation Workflow



Octubre 2024

Contact Details Cliente:		
Contact Details IBM:	Sebastian Sturla SebastianSturla@ibm.com	Juan Felipe Cabezas jcabezas@co.ibm.com

Copyright © 2024 by International Business Machines Corporation (IBM). No part of this document may be reproduced or transmitted in any form without written permission from IBM.

DETALLE DEL VERSIONAMIENTO			
Versión	**Fecha Revisión**	**Autor**	**Descripción de la versión**
1.0	24/10/2024	Sebastian Sturla	Primera versión
2.0	30/10/2024	Sebastian Sturla	Segunda versión
3.0	31/10/2024	Sebastian Sturla	Tercera versión

Contenido

1. INTRODUCCIÓN 4
   1.1. Objetivo del Documento 4
   1.2. Alcance 4
2. INFORME 5
   2.1. Diseño de la arquitectura y sus patrones 5
   2.2. Revisión de código - Flujos 6
   2.3. Revisión de código - Diseño de los servicios 9
   2.4. Revisión de código - Diseño de los servicios “clasificación de documentos” 13
   2.5. Revisión de código - Diseño de las excepciones 14
   2.6. Diseño de las pantallas desarrolladas 14
   2.7. Análisis de los logs 16
   2.8. Mantenimiento y configuración del ambiente 17
3. RECOMENDACIONES 21
   3.1. Process App – Gestión de Comunicaciones Oficiales 21
   3.2. Process App – Módulo de Administración 27
4. CONCLUSIÓN 29

1. INTRODUCCIÓN

1.1. Objetivo del Documento

En este informe se procedió a realizar una revisión del código de la Process App “Gestión de Comunicaciones Oficiales” y de la Process App “Modulo de Administración” con sus respectivos Client-Side Human Services que figuran en el alcance del contrato. Se busca encontrar tendencias en su comportamiento durante la revisión del código de manera de ofrecer finalmente una serie de hallazgos y conclusiones con sugerencias enfocadas en implementar y reforzar las buenas prácticas, buscando por objetivo final la calidad del producto.

1.2. Alcance

El presente informe está dirigido al personal de la UGPP y especialistas de IBM involucrados en el Health Check de Expert Labs.

2. INFORME

El presente documento incluye una breve descripción, un listado que corresponde a los hallazgos obtenidos durante la revisión del código de las Process App y un detalle de los puntos que fueron descubiertos en base a los estándares de IBM sobre las buenas prácticas a utilizar.

2.1. Diseño de la arquitectura y sus patrones

Dentro de este subtitulo, se abordará la metodología, la lógica utilizada al momento de diseñar la arquitectura de los procesos y los patrones implementados.

La arquitectura implementada en los procesos que abarca el alcance de este documento cuenta con una correcta gestión de estos donde la distribución de los procesos y su segmentación cumple en una buena medida con las buenas prácticas. El caso contrario a lo mencionando, es la aglomeración de los procesos o Client-Side abarcando diferentes actividades a tal punto que sean diseños muy grandes ocasionando consecuencias negativas para su mantención y operabilidad.

Al momento de revisar los patrones, se pudo detectar que no siempre se utiliza el mismo para el desarrollo y eso puede provocar que al momento de realizar alguna modificación ocasione demoras en la comprensión previa que se necesita para realizar dicha modificación del código. Aunque esta mención será abordada con más detalle en los próximos subtítulos.

La utilización de RPA para agilizar o completar determinadas actividades dentro de los procesos, no se considera una mala practica debido a que estos se les tiene que enseñar que hacer para realizar la actividad. También a estos se les puede aumentar la velocidad de su ejecución para completar las tareas con más rapidez, aunque se debe tener precaución, ya que, el Bot puede realizar las tareas más rápido pero las actividades que el Bot va a utilizar deben estar optimizadas y con buenos tiempos de respuesta. Entonces, se recomienda evaluar los tiempos de ejecución de las actividades que el Bot utiliza, así puede finalizar con las tareas para lo que fue creado o reducir la velocidad del Bot para la realización y finalización de sus tareas.

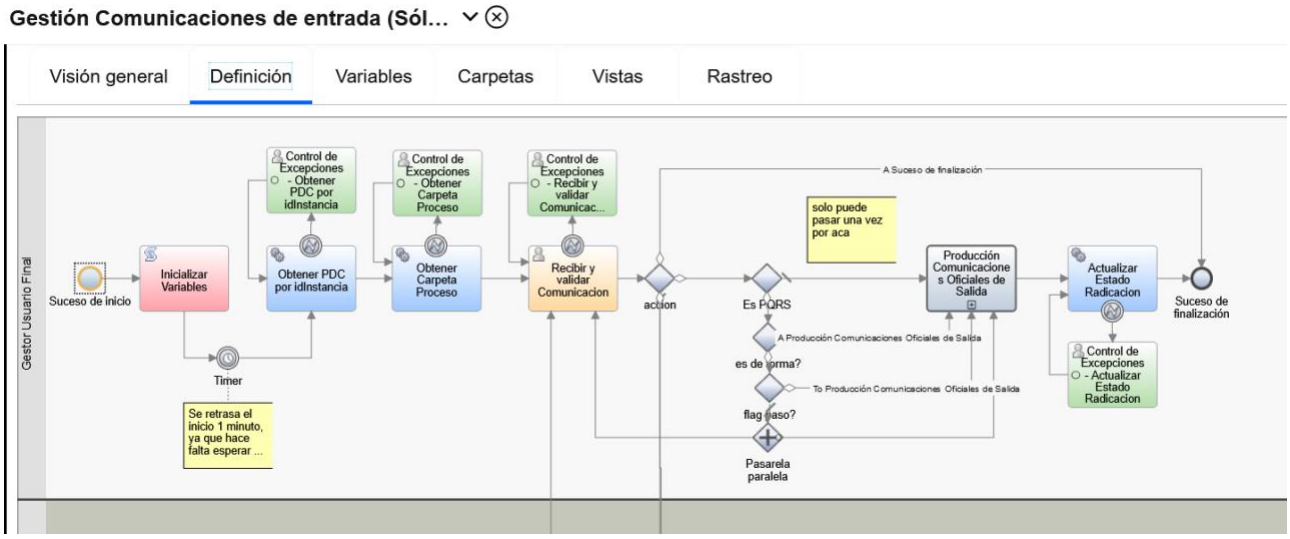
Para concluir con este subtitulo, se debe someter a evaluación los tiempos de respuesta de las actividades que realiza el Bot de RPA, ya que este actúa en consecuencia a lo que se le programo. Una vez completada la evaluación, se debe optimizar las actividades o reducir la velocidad de ejecución del Bot hasta la optimización de las actividades en la que está involucrado.

2.2. Revisión de código - Flujos

Dentro de este subtitulo, se somete a análisis la composición de los flujos de los procesos a alto nivel que se encuentran de cara al usuario y si nos adentramos, nos encontramos con los flujos de los Client-Side, que contienen la lógica funcional del negocio y son las actividades que funcionan por debajo de la ejecución de las pantallas. En estos se debe tener otro tipo de análisis debido a que contienen la lógica funcional del negocio y para cualquier cambio se debe evaluar el costo beneficio además de los plazos que requieren la reimplementación. No obstante, hay ciertas pautas que se deben remarcar para lograr una mejoría en el performance de los ambientes que se van a ir mencionando a lo largo del subtitulo.

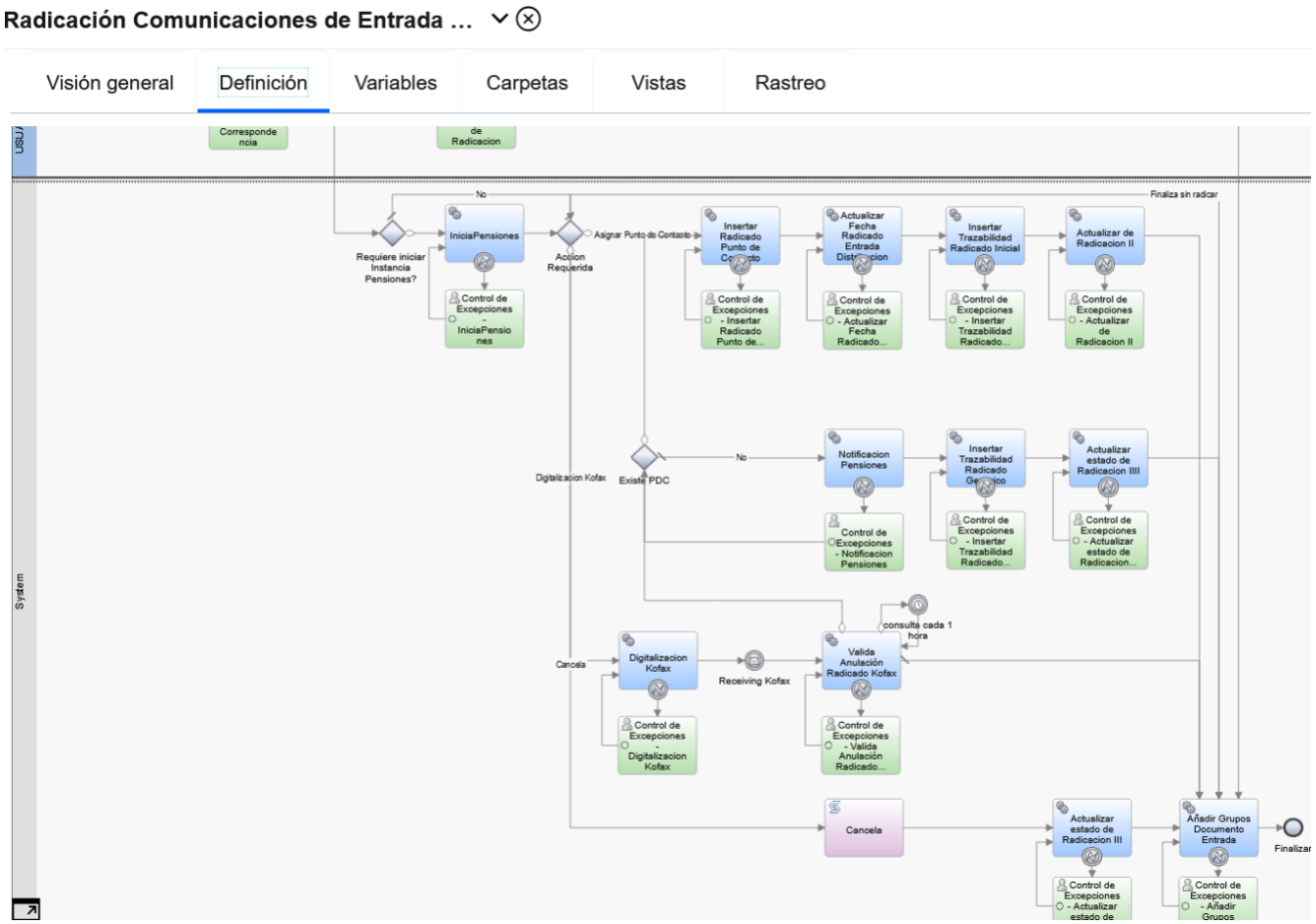
A continuación, se procede a listar los descubrimientos para luego brindar un detalle de cada punto.

- Utilización reiterativa del componente “Gateway”



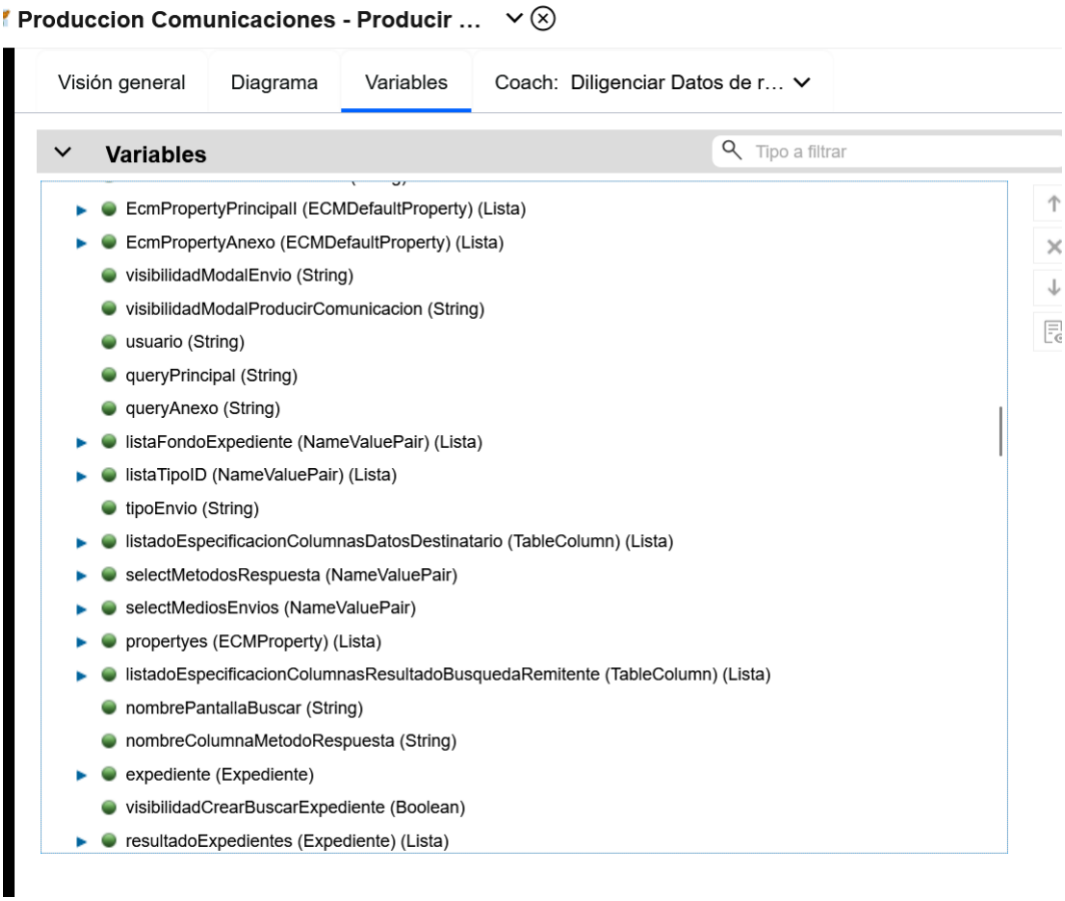
(Imagen 1)

- Cantidad elevada de componentes según los criterios de diseño



(Imagen 2)

- Limpieza de variables privadas y auxiliares



(Imagen 3)

Estos fueron los descubrimientos que se consideran como malas prácticas en el diseño de los flujos de los procesos y los Client-Side con sus respectivas observaciones.

Como primer punto, se observó una elevada utilización del componente “Gateway” anidados en los procesos donde por norma de diseño no es lo deseable. En este caso, por la cantidad de alternativas que poseen los flujos y con el desconocimiento en detalle de los requerimientos funcionales del negocio dicha norma debe evaluar su costo beneficio para su modificación, aunque al no ser una práctica critica se convierte en un llamado de atención.

Como segundo punto, que está muy alineado al primero. Este abarca a todos los componentes de cada uno de los procesos, deben contar con la característica de alto nivel al estar frente a los usuarios de negocio porque no debe ser relevante el cómo está diseñado sino en cumplir con los objetivos y tareas que tienen asignados. Se pudo observar que, en ciertas ocasiones, hay una sobre utilización de servicios en esta etapa de alto nivel, aunque esto no repercute en una disminución del performance. En cambio, para los Client-Side se debe modificar este razonamiento

porque estos van a tener implementado toda la lógica del negocio. Entonces no se detectó malas prácticas críticas que afecten al performance del ambiente, para definir una situación, lo recomendable es encapsular todos o la mayor cantidad de servicios previos a la aparición de la pantalla para una mejor compresión del código y sobre la reutilización de servicios. Además analizando estos servicios, no son consultas complejas por lo tanto, desde la revisión del código no se encuentran motivos suficientes para determinar que este es el causante de la perdida de performance.

Como último punto, no se logró identificar el correcto tratamiento sobre la limpieza de variables de los procesos o de los Client-Side, donde cada uno de estos poseen una muy alta cantidad de variables privadas y auxiliares. Se puede visualizar en la evidencia a modo de ejemplo que pertenece al Client-Side “Producir Comunicaciones – Producir Documento” de la ProcessApp “Gestión de Comunicaciones Oficiales” que es una lista muy larga de variables que no es lo recomendable y significa, recursos que no se están gestionando de la manera correcta. Lo que sucede en estos casos es que por cada instancia creada y se adiciona el incremento natural por el funcionamiento del negocio, se va a almacenar en memoria todos esos datos provocando una gran cantidad de información sin tratamiento y en consecuencia terminar provocando una disminución del performance del ambiente. Este tratamiento consiste en la inicialización de las variables que ya no cumple una tarea dentro del flujo o Client-Side. Es decir, si una variable se utiliza hasta la mitad del flujo, pasado la mitad se debe inicializar para no consumir y liberar los recursos. Este es uno de los puntos que más relevancia tiene en este subtitulo, por el tiempo que puede llegar a consumir realizar dicha revisión y modificación, tanto para los procesos como en los Client-Side por las altas dimensiones que poseen, y que cada uno de ellos debe contar con sus respectivas pruebas para validar que todo siga funcionando como es debido.

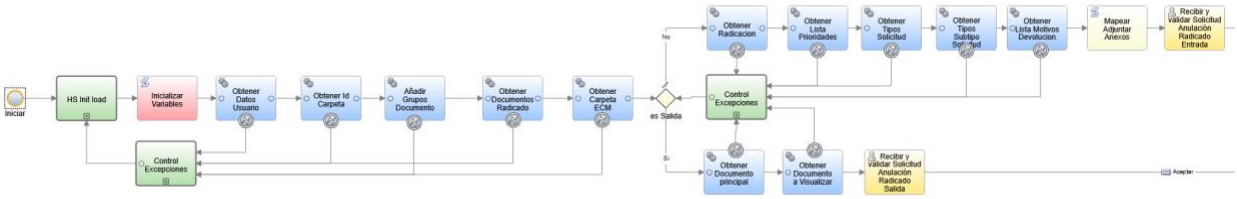
Con toda la información analizada, se puede determinar que la prioridad en este subtitulo es el tratamiento de la limpieza de las variables privadas y auxiliares al ser un factor que si afecta al performance del ambiente por la mala gestión que se tiene actualmente. Dicho tratamiento debe ser sometido a pruebas para corroborar que no haya afectado al funcionamiento de flujo.

2.3. Revisión de código - Diseño de los servicios

Dentro de este subtitulo, se evalúa el cómo están desarrollados los servicios y sus respectivas configuraciones, el cómo están implementados en los procesos y en los Client-Side.

A continuación, se procede a listar los descubrimientos para luego brindar un detalle de cada punto.

- Unificación de los servicios hijo a un servicio padre



(Imagen 4)

- Múltiples criterios de desarrollo
- Falta de definición en las consultas
- Limpieza de las variables privadas y auxiliares
- Configuración de los servicios



(Imagen 5)

Estos fueron los descubrimientos que se consideran como malas prácticas de desarrollo y observaciones que son capaces de optimizar el performance de los ambientes.

Como primer punto, se observó que en ciertos Client-Side hay una cadena de servicios con el objetivo de obtener información de las respectivas fuentes de información que funcionalmente es lo correcto, aunque por normas de diseño es mejor aglomerar estas cadenas de servicios, en un servicio padre en los casos que sea posible implementarlo, para optimizar el mantenimiento y sea más legible el código.

Como segundo punto, se detectó diferentes formas de realizar los desarrollos de los servicios que impactan a la base de datos. Donde este, se considera como una mala práctica porque al contar con múltiples maneras de desarrollo en los servicios significa que no se tiene un standard al momento de desarrollar. Y llegado el caso de

suceder un incidente en algún servicio, el desarrollador primero debe analizar el código para entender de que se trata y luego debe empezar con la resolución, pero si es una de las formas en que no suele desarrollar esto termina repercutiendo e impactando en el tiempo de resolución del incidente al no tener dicho standard de desarrollo.

A continuación, se listará los distintos métodos encontrados a modo de referencia y luego se indicará cuáles son las maneras correctas e incorrectas:

- La utilización de los params (Servicio: Insertar Persona Directorio)



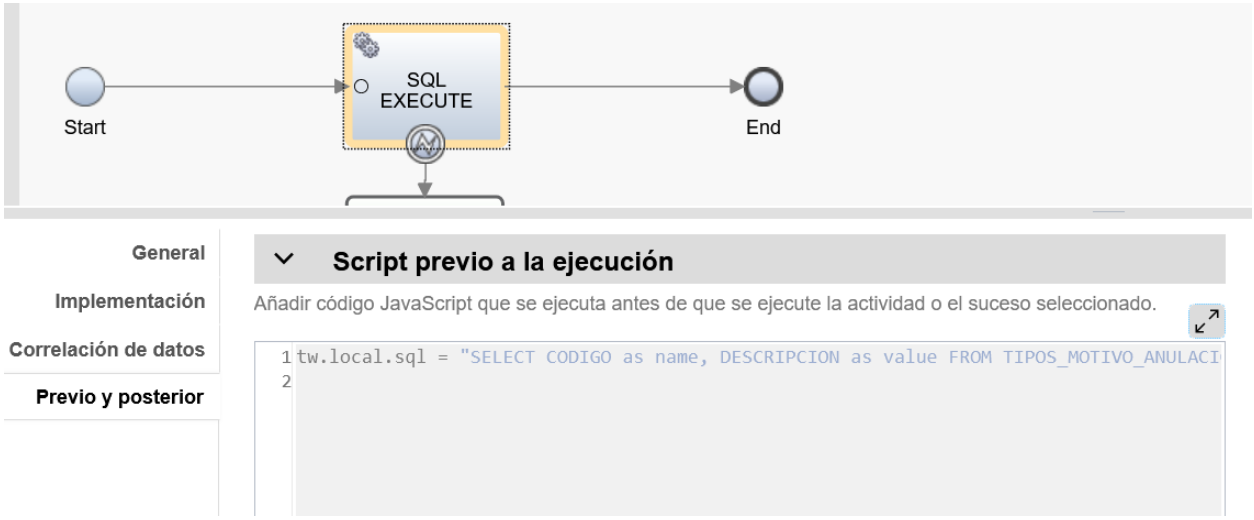
(Imagen 6)

- La utilización de la modalidad scriplet (Servicio: Buscar Remitentes Directorio)



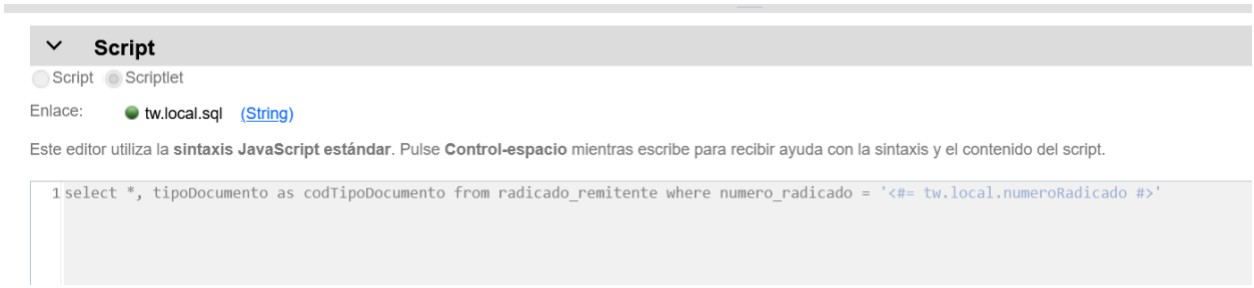
(Imagen 7)

- La utilización del pre/post (Obtener Lista Motivos Anulación)



(Imagen 8)

- La utilización del “SELECT *” (Servicio: Obtener Radicado Remitente)



(Imagen 9)

La utilización de los 2 primeros ítems, son los métodos tradicionales y son los que están bien para realizar cualquier tipo de consulta hacia las diferentes fuentes de información.

En cambio, los siguientes 2 ítems son los que están mal para realizar cualquier tipo de consulta. Uno consiste en utilizar los pre/post de los componentes ocasionando un aumento de los tiempos de la resolución de incidente al no estar sujeto al standard o el otro que provoca un impacto en el performance de los servicios sobre la utilización del asterisco (*) en las consultas hacia la base de datos por el esfuerzo que conlleva realizar la acción con esa modalidad. Este se encontró en un 2% de todos los servicios, aun siendo bajo el porcentaje no se debe hacer caso omiso al tema porque este porcentaje puede ir incrementando muy de a poco hasta llegar a un nivel alarmante. Por esa razón se debe estar pendiente.

Como tercer punto, esta estrictamente vinculado a uno de los criterios de desarrollo, sobre la utilización del asterisco (*) que ya fue mencionado en el subpunto anterior, al ser considerado como un mal hábito y práctica.

Como cuarto punto, comparte la misma lógica del subtítulo anterior “2.2. Revisión de código – Flujos” donde en este caso, la aplicación de esta limpieza de variables tiene una implementación menos compleja debido a que estos tipos de servicios contienen pocas variables y son más pequeños a diferencia de un proceso o un Client-Side como tal.

Como último punto, hay una opción “Sincronizar automáticamente objetos de negocio compartidos” que se encuentra activada por defecto en la vista general de los servicios, que no está cumpliendo con su función en este caso y lo que realiza es una sincronización de los “BO Shared” que puede repercutir en el performance durante la ejecución de los procesos por querer realizar dicha acción. Y, hay otra opción “Habilitar almacenamiento en memoria caché de resultados de servicio” que permite almacenar el resultado de las consultas en la memoria cache del navegador y evita que estos servicios, se vuelvan a ejecutar logrando una optimización de los tiempos durante la ejecución de los Client-Side. Lo que se debe tener en cuenta sobre esta opción es, que se debe habilitar en los casos donde el resultado de las consultas sea siempre el mismo.

Con toda la información analizada, se puede determinar que la prioridad en este subtitulo sigue siendo el tratamiento de limpieza de las variables privadas y auxiliares. Y a su vez, la definición de un standard de desarrollo con el fin de optimizar los tiempos de mantenimiento del código dentro de los servicios. Sigue siendo implícito que cualquier modificación se debe someter a las pruebas correspondientes para validar el correcto funcionamiento del servicio en este caso.

2.4. Revisión de código - Diseño de los servicios “clasificación de documentos”

Por las dimensiones actuales de estos servicios, no se detecta con mala salud desde su estructura y tampoco se debe perder el enfoque sobre la limpieza de variables que fue mencionada con anterioridad. En cambio, se detectó un mal hábito que se va a detallar a continuación.

Ante los inconvenientes que se presentaron en la reunión de entendimiento sobre la problemática de la clasificación de los documentos mediante la integración de Filenet en el BAW, se dio a conocer que no se están asignando los permisos adecuados de los usuarios para la visualización de los documentos/expedientes, y en consecuencia, no cumple con el requerimiento informado.

Se descubrió que en el código se está utilizando la misma variable global (es de gran utilidad durante el desarrollo porque se define una sola vez y se puede utilizar en cualquier lugar de la Process App) para la creación o interacción con el servicio de integración de Filenet, excepto en el ambiente de producción que utiliza otro valor.

Tras haber revisado los servicios, debuggearlos y analizar las variables que se utilizan durante esa etapa, se concluye que no se encuentra una lógica en el código que permita realizar un filtro para la asignación especifica de los usuarios o grupos hacia los documentos/expedientes, además se está utilizando la variable global de manera incorrecta ocasionando este comportamiento incorrecto. Y que también se debe analizar la composición de los grupos y de la composición de estos por si cuenta con los usuarios correctos debido a que no se está cumpliendo el requerimiento solicitado.

2.5. Revisión de código - Diseño de las excepciones

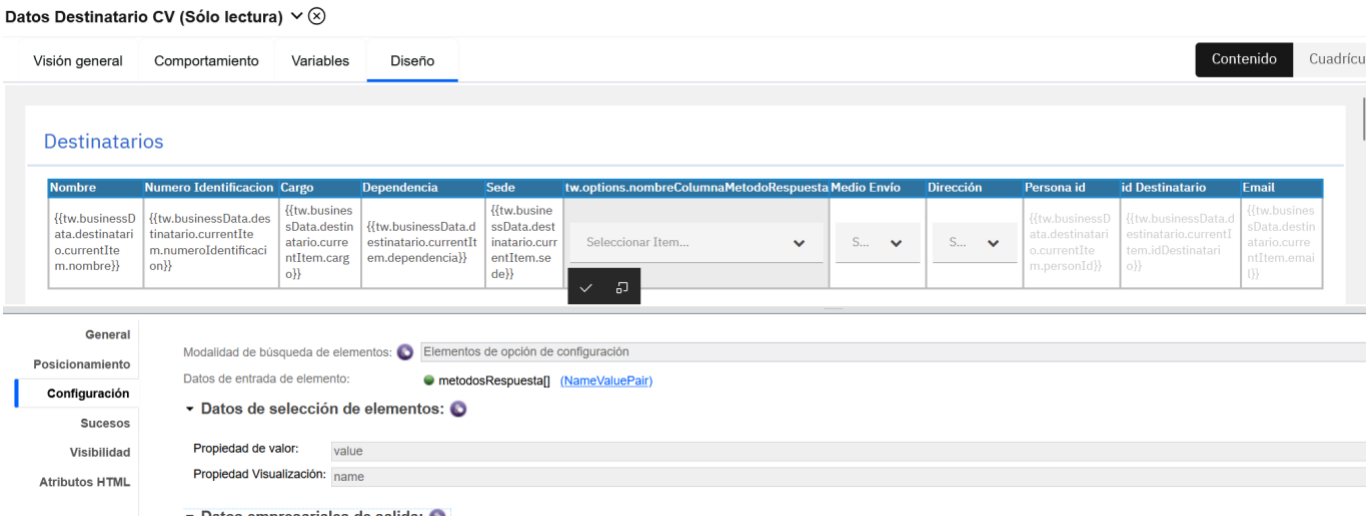
Dentro de este subtitulo, no se detectó ninguna mala práctica o habito sobre el diseño de las excepciones. Al momento de diseñar los procesos, las excepciones nos evitan la aparición de un error que irrumpa o detenga la instancia, en un estado inicial o avanzado dentro del flujo.

El diseño elegido para las excepciones es simple y funcional para las partes (usuario final y el equipo de desarrollo). Al usuario final le permite rápidamente enviar el error que le impide avanzar con su instancia y al equipo de desarrollo, le permite visualizar donde este sucediendo el error con su sección de detalle y la descripción del error en general. Además, que este, esta implementado en todos los servicios de los procesos considerándolo como una muy buena práctica y modalidad de trabajo.

2.6. Diseño de las pantallas desarrolladas

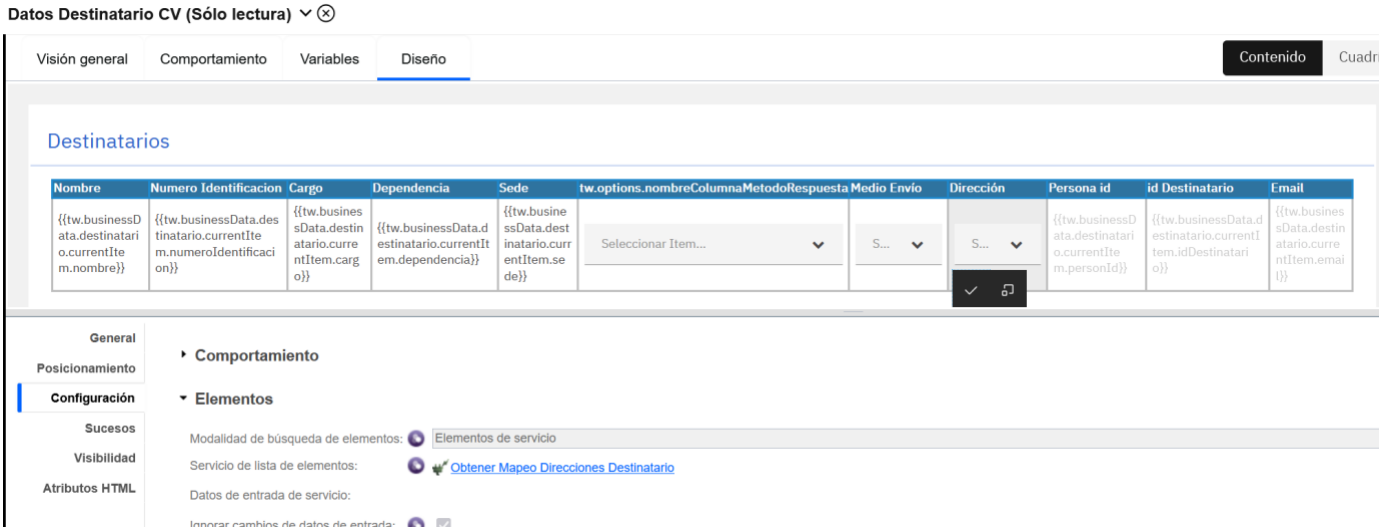
Dentro de este subtitulo, lo que se analizó fue la composición de las pantallas. Las mismas no presentan sobre saturación de componentes, ya que esta es una de las causas principales de problemas de performance para la carga inicial y durante la interacción de las pantallas. Y que en esa sobresaturación mencionada, se vincula estrictamente a la asignación de variables que genera aún más demora en el performance del ambiente pero esto no es lo que está sucediendo en los ambientes.

Desde el enfoque de desarrollo, se detectó sobre el componente diseñado por el producto, el “Single Select” o selector que muestra los datos en un formato de fila que se encuentra ubicado en múltiples pantallas. Donde los servicios que permiten completar o llenar a este componente, están ubicados en el diagrama del Client-Side y eso repercute en un servicio más que se ejecuta dentro del Client-Side aumentando los tiempos de ejecución de los servicios previos a la visualización de las pantallas. Como figura en la evidencia desde la vista del diseño de la pantalla, a continuación:



(imagen 10)

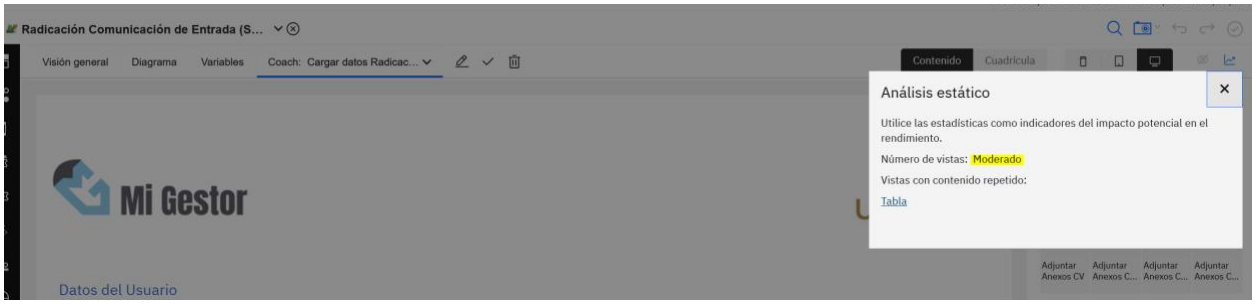
Esta acción de llenado es una alternativa viable y demuestra que funciona de manera correcta, aunque hay una manera más eficiente al transmitir este esfuerzo y tiempo de ejecución al navegador que se esté utilizando. Y consiste en la utilización de ese servicio creado, y reubicarlo fácilmente dentro de la configuración de este componente. Como si figura en la evidencia desde la vista del diseño de la pantalla, a continuación:



(Imagen 11)

Otro punto que sirve como parámetro para determinar el estado de las pantallas durante su diseño consiste en la utilización de un ícono del web designer, que entrega un rating o score en base a la carga de los componentes que posee la pantalla y como resultado, se observó en múltiples pantallas el score moderado (solo existen 3 niveles), este es el nivel intermedio provocado por el uso de las tablas. Este factor no

se debe utilizar como parámetro de evaluación en estos casos debido a que no se presenta en el último nivel y además que la utilización de una tabla para cualquier tipo de diseño aumenta el score. Entonces solo se debe tener en cuenta cuando llega al último nivel y no es lo que sucede.



(Imagen 12)

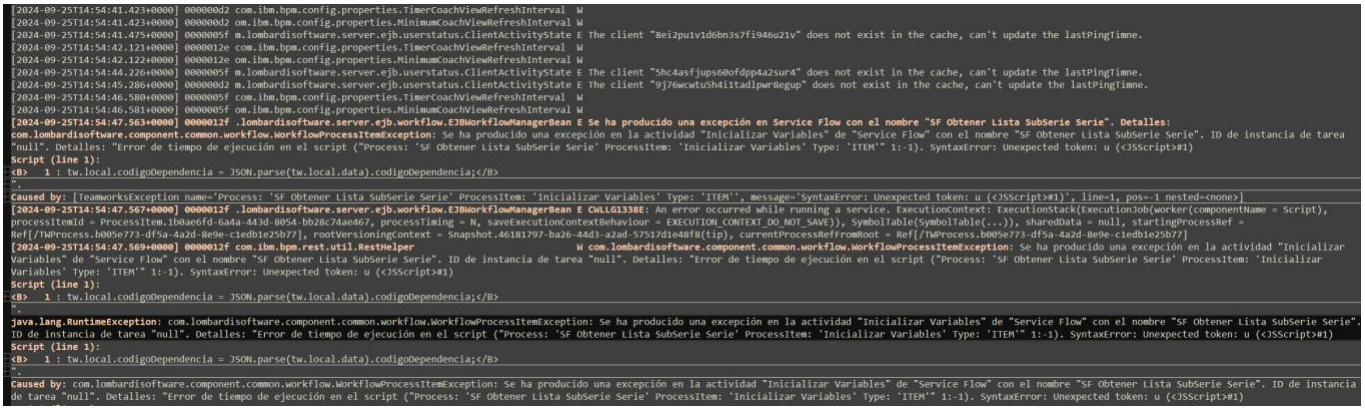
Para concluir con este subtitulo sobre los diseños de las pantallas, se debe revisar los componentes “Single Select” de los todos los coach de todos los Client-Side con el fin de determinar cuál es el método que se está utilizando, reemplazarlo por el más eficiente como se mencionó y que figura en la “imagen 11”.

2.7. Análisis de los logs

Dentro de este subtitulo, se solicitó un set de logs del ambiente de QA debido a que en el ambiente de producción puede contener información delicada sobre algún cliente.

Luego de su análisis, no se detectaron cantidades alarmantes ni altas cantidades de códigos de error que pueden ocasionar consecuencias en caso de no aplicarse el correcto tratamiento previo al deploy del Snapshot en el ambiente productivo. Ni tampoco se detectó, la utilización de la función “console.log” que permite mostrar en los logs la información de las variables o lo que se le indique, considerándolo una muy mala práctica para un ambiente productivo por “ensuciar” el log e impidiendo una rápida detección del hipotético incidente.

Como se puede evidenciar, la siguiente captura de pantalla del log del ambiente de QA del día 25 de septiembre del presente año. Se puede observar, una excepción con su grado de detalle y estas situaciones se pueden presentar durante el funcionamiento de este ambiente. Dejando en claro el funcionamiento de las excepciones implementadas y lo que debe corregir o agregar a la implementación por parte del equipo de desarrollo.



(Imagen 12)

En conclusión, los logs del ambiente de QA se encuentran bien y lo que se muestra en la evidencia es un comportamiento esperable por el tipo de ambiente. Los códigos de error que pueden llegar a aparecer durante la operatoria, se les debe aplicar el correcto tratamiento para evitar que estos lleguen a ocasionar incidentes en el ambiente productivo. Se puede determinar que las probabilidades de que se ocasionen incidentes en el ambiente productivo son bajas por el panorama analizado en los logs de QA y en caso de que sucedan, los tiempos de resolución de estos deberían ser bajos dependiendo del tipo error en este supuesto.

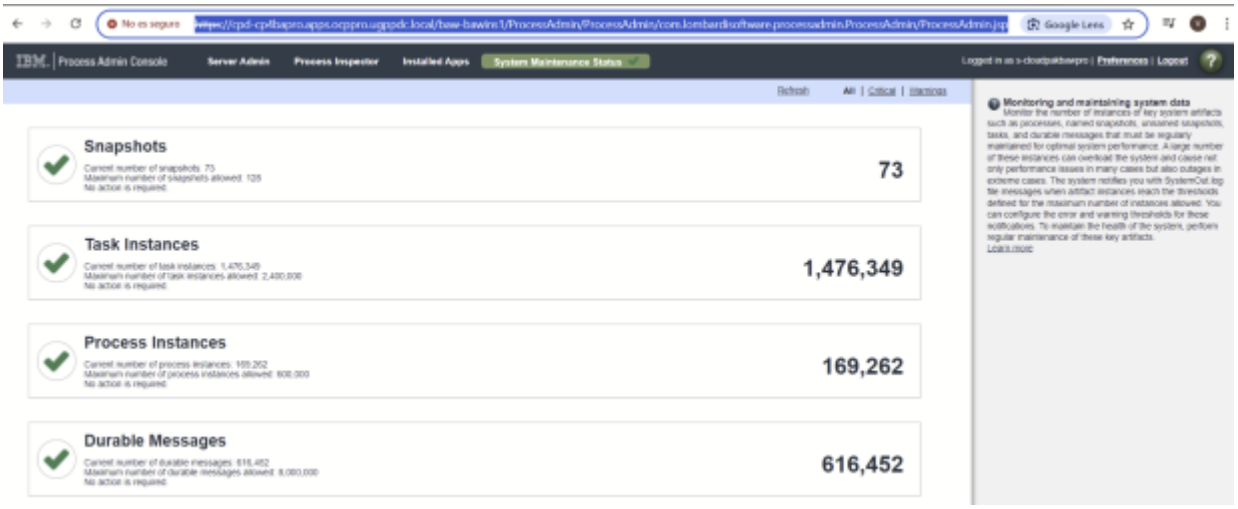
2.8. Mantenimiento y configuración del ambiente

Los temas que serán tratados dentro de este subtitulo serán observaciones debido a que el enfoque principal está sobre la revisión del código y no se cuenta con el historial adecuado del ambiente y no se realizó una estrategia de tuning/estabilización sobre el producto.

Por lo pronto, luego de haber ingresado al Process Admin Console de producción se detectó que no se tiene definido un porcentaje de seguridad por la alta cantidad de Task Instances. Es decir, para mantener un ambiente saludable, se debe establecer un umbral de seguridad y en caso de superarlo, como se puede evidenciar donde el límite máximo que figura es de 2.400.000 y se observa que se tiene 1.476.349 que representa un 61.51%. Se determina que no se tiene establecido un umbral y esto termina repercutiendo en el performance del ambiente ocasionando demoras. Lo que se debe hacer es establecer el umbral, que como recomendación debe estar entre el 30% a 40% para realizar el mantenimiento. Esto se debe realizar por fuera del horario laboral debido a que repercute en el performance del ambiente, luego la periodicidad debe estar establecida en base al incremento o carga que se genera en el día a día. Dicho esto y dado la situación, se recomienda establecerlo semanalmente hasta alcanzar el umbral mencionado y una vez estabilizado se debe reevaluar la periodicidad de los mantenimientos.

Lo mismo se aplica a las Process Intances, Durable Messages y Snapshots, estos pueden tener un umbral de seguridad más permisivo dado los límites máximos que tienen donde se puede establecer en un 40% a 50% que sería lo saludable.

La metodología de limpieza utilizada que proporciona el portal es un proceso standard de mantenimiento y es correcto. Aun así, con lo ya mencionado se debe readaptar y posiblemente redefinir el mantenimiento que se realiza ante los hallazgos mencionados para alcanzar un estado saludable.

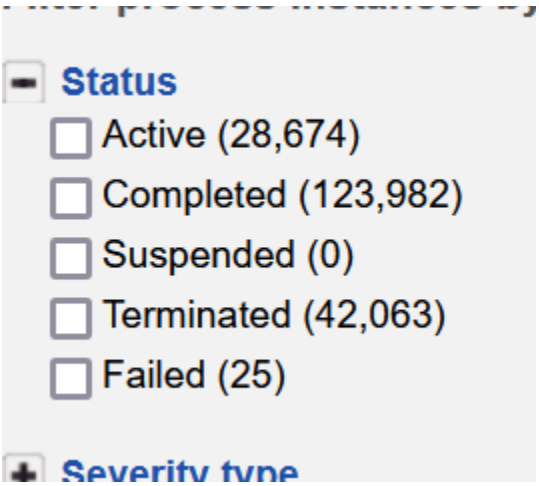


(Imagen 13)

Por el Process Inspector del ambiente de produccion, se puede observar un porcentaje menor de instancias activas en comparación a la suma de instancias completadas y terminadas.

Dada la situación de la alta cantidad de instancias completadas, lo que puede estar sucediendo es que se estén almacenando en la base de datos del producto y lo que debería suceder es que se almacene en una base de datos del negocio (es el nombre habitual que se establece). Todos los datos de las instancias completadas se pueden migrar a esta otra base y este es un proceso posible de realizar aunque va a requerir tiempo por la densidad de la información y por la alta cantidad que se observa.

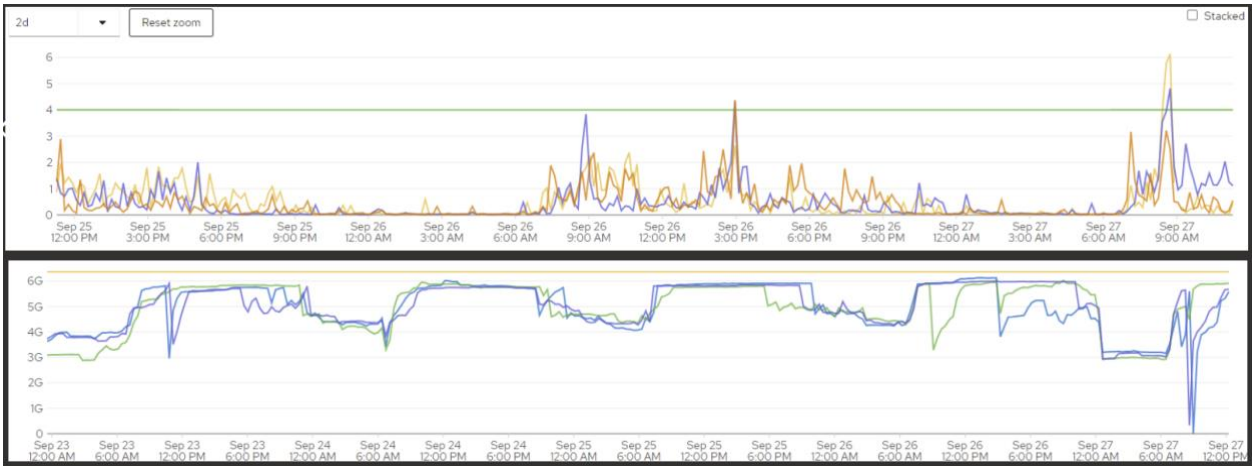
Se recomienda evaluar si las instancias terminadas tienen propósitos de auditoria o no para en tal caso, realizar una limpieza de estas y liberar la carga en memoria pudiendo así mejorar el performance hasta que se decida realizar una migración a otra tabla como se mencionó.



(Imagen 14)

Otro de los puntos sobre el Process Inspector, es sobre los supuestos accesos no autorizados al usuario con privilegios de administrador, que cuentan con el permiso para terminar cualquier instancia considerándola como una mala práctica e incumpliendo una norma de seguridad. Por ende, lo recomendable es establecer desde políticas para la terminación de instancias, o el cambio de contraseña de este usuario y así solo otorgarles el acceso a los responsables a cargo de esta actividad, o generar contraseñas temporales para poder acceder a ese usuario con privilegios o la creación de nuevos usuarios con el rol Admin y que estos solo deben ser entregados a determinadas personas con la autoridad de poder realizar dicha acción ganando una mejoría en la trazabilidad de los accesos. Estas alternativas son debido a que no hay manera de establecer una traza sobre quiénes son los que provocan esta terminación de instancias utilizando el usuario administrador, ya que en el Process Inspector solo figura como el administrador que realizó la acción por este acceso no autorizado al usuario con privilegios.

Con respecto a la configuración, primero se revisó la asignación de los recursos de CPU y memoria RAM en Openshift, para los respectivos Pods de “workflow-server” de BAW, no se detectó irregularidades en la asignación de recursos utilizando la documentación que determina los valores mínimos recomendados. Estos mismos valores, es común que aplicado a cada cliente pueda variar y se deba realizar un tuning dedicado por las necesidades de cada proceso que poseen. En el título de “Recomendaciones”, se dejará una recomendación en base a picos de consumo de CPU (Ítem E) y momentos donde la utilización de memoria RAM (Ítem F) supera un límite de seguridad del 90% pudiendo probar que cuenta con poca memoria libre en caso de suceder un pico de consumo excesivo ocasionado por alguna situación desconocida en lo inmediato.



(Imagen 15)

Como segundo punto en lo referido a la configuración, el tema de los reinicios en los pods en el ambiente productivo no es una practica habitual o comportamiento habitual. En el caso de que sea un reinicio programado o por alguna modificacion que se haya realizado en la configuracion, lo recomendable es notificar al negocio sobre dicho reinicio asi no afecta a la operación del negocio o le da la herramienta al negocio de poder gestionar este corte de servicio y para ejemplificar, en determinado horario en determinado día no se va a poder utilizar la plataforma por alguna razón. Esto lo debe realizar el equipo de infraestructura al tener el control del ambiente. Pero en el caso de que no sea programado, se recomienda la utilizacion de un producto de monitoreo de la familia de Kubernetes para que genere las alarmas a las áreas correspondientes y puedan revisar lo que está sucediendo. Se debe utilizar esa alternativa debido a que BAW no posee o no puede generar ese tipo de alertas.

3. RECOMENDACIONES

Para dar comienzo, se comparte un link a la documentación sobre los buenos usos y prácticas que se deben tener a la hora de desarrollar, se debe prestar atención a los capítulos 2 y 3 que abordan temas de desarrollo, estos se encuentran dentro del denominado IBM RedBooks - IBM Business Process Manager V8.5 Performance Tuning and Best Practices. Se debe considerar que a pesar de ser una versión antigua a la que cuentan actualmente, los lineamientos base se siguen respetando. Link: https://www.ibm.com/docs/en/baw/22.x?topic=management-tuning

A continuación un resumen de los subtítulos que se deben tener en consideración para el área de Sistemas de Información de la UGPP:

- 2.2.1 Common best practices: Choose the appropriate granularity for a process
- 2.2.2 Process Designer Arquitecture best practices: Minimize use of Service Tasks
- 2.2.2 Process Designer Arquitecture best practices: Manage variable usage
- 2.2.2 Process Designer Arquitecture best practices: Develop efficient coaches
- 2.2.2 Process Designer Arquitecture best practices: Minimize use of large javascript scripts
- 2.3.4 Evaluate service providers and external interfaces
- 2.4.2 Use a high-performance browser
- 2.4.3 Enable browser caching
- 2.5 Large business objects
- 3.1 BPMN Business process authoring best practices
- 3.1.1 Clear variables in exposed human services
- 3.1.3 Use conditional joins only when necessary
- 3.1.5 Use sequential system lane activities efficiently
- 3.4 Browser environment considerations

Todas las sugerencias o recomendaciones deberán ser sometidas a pruebas para corroborar su rendimiento y efectividad.

3.1. Process App – Gestión de Comunicaciones Oficiales

A. Se debe establecer un porcentaje de entre 30% a 40% como umbral de seguridad sobre las Task Instances que figura en el Process Admin Console y si supera el porcentaje establecido, se debe realizar las correspondientes tareas de mantenimiento. La periodicidad va a depender de que tan rápido supere el umbral establecido y se debe realizar por fuera del horario laboral por el impacto en el

performance que puede provocar. Se recomienda que hasta no alcanzar el umbral, se debe realizar el mantenimiento semanalmente y por fuera del horario laboral.

B. Se propone evaluar una limpieza de las instancias terminadas si no cumplen un rol de auditoría, se debe hacer semanalmente y por fuera del horario laboral para no generar impacto en el performance del ambiente. Pero en el caso de cumplir un rol de auditoría, se debe evaluar la migración de las instancias completadas y terminadas a otra base de negocio como se mencionó dado la alta cantidad que se puede visualizar en el Process Inspector.

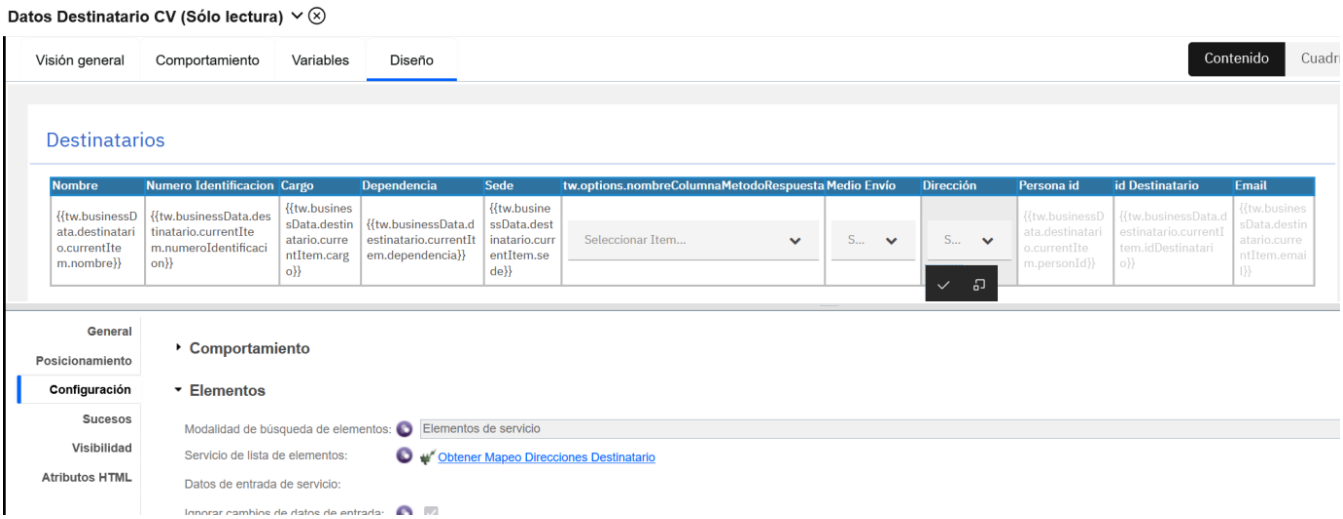
C. Se propone aplicar el tratamiento indicado sobre la limpieza de las variables en todos los flujos, en los Client-Side y en los servicios con el objetivo de inicializar las variables privadas y auxiliares que ya hayan cumplido su vida útil dentro del flujo, Client-Side y servicios como fue mencionado en el subtítulo “2.2” y “2.3”. Esto consiste en la creación de unos scripts ubicados a criterio del desarrollador dentro de los diagramas para inicializar las variables y dejarlas vacías. Logrando así, una mejora en la gestión de los recursos por los datos en la operatoria del producto.

D. Para Filenet, el grupo de sistemas de información de la UGPP debe analizar la composición de los grupos y de los usuarios asignados para validar si las asignaciones son las correctas. Y luego, se debe implementar una lógica de desarrollo mediante el diseño de filtros para la asignación de permisos hacia los usuarios. Logrando así, que puedan visualizar e interactuar con los documentos según los requerimientos.

E. Como sugerencia, se propone realizar un incremento en la asignación de CPU de 1 core más a cada pod del “workflow-server”, permitiéndole tener una holgura a la plataforma en las horas y momentos pico del ambiente.

F. Como sugerencia, se propone realizar un incremento en la asignación de memoria RAM de al menos 1GiB a cada pod del “workflow-server”, permitiéndole tener una holgura en las horas y momentos pico del ambiente.

G. Se debe cambiar el método de llenado del componente “Single Select” por el método más eficiente mencionado en el subtítulo “2.6”. Se deja la imagen 16 a modo de ejemplo.



(Imagen 16)

H. Se propone utilizar uno o dos criterios de desarrollo en los servicios mencionados, según el grado de dificultad de la consulta. El objetivo es implementar un standard para que todos los servicios sean similares y mejorar los tiempos de resolución de incidentes y/o realizar cambios en el código.

I. Al momento de desarrollar la query en los servicios, no se debe utilizar el asterisco (*) en el SELECT. Siempre se debe definir los campos a consultar.

J. Se propone deshabilitar la opción de “Sincronizar automáticamente objetos de negocio compartidos” que figura, en la vista general de cada uno de los servicios, ya que no está cumpliendo ninguna función ni el propósito de esta. Se utiliza este servicio de ejemplo.



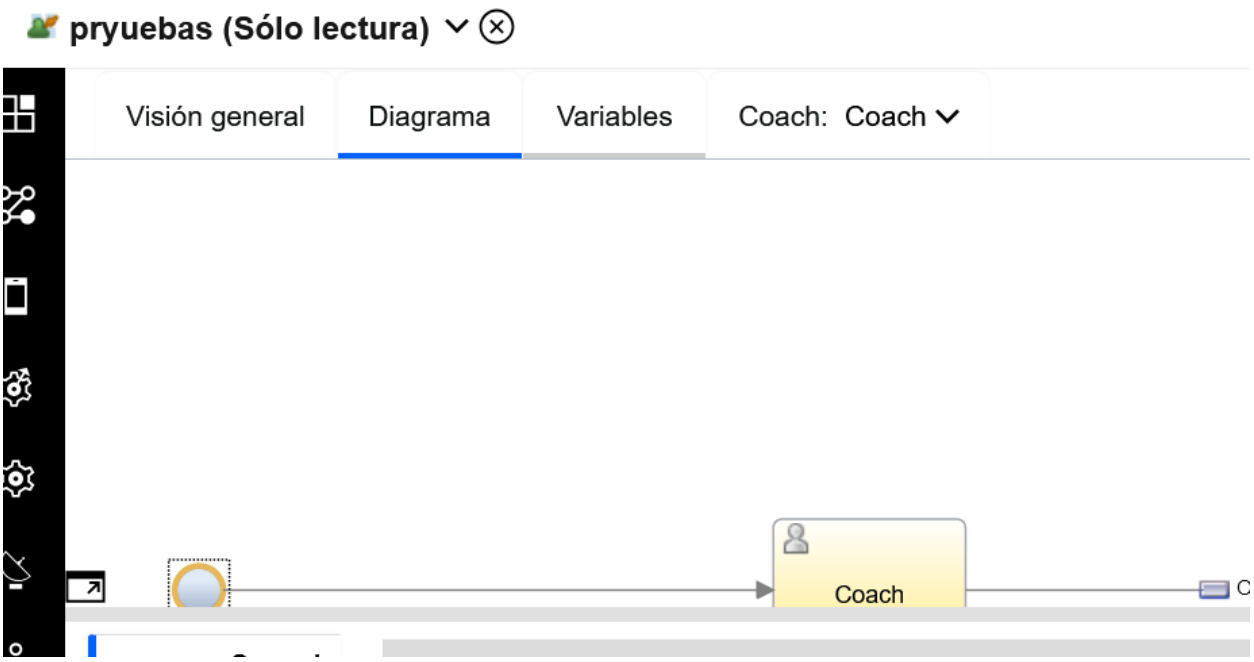
(Imagen 17)

K. Se propone habilitar la opción “Habilitar almacenamiento en memoria cache de resultados de servicio” que figura, en la vista general de cada uno de los servicios pero solo en los casos donde los datos que se obtienen de la base de datos sean los mismos o no tengan modificaciones (Por ej. los países). Se debe utilizar la imagen del ítem J como ejemplo.

L. Se propone revisar el backend de cada servicio externo para comprobar sus tiempos de respuesta.

M. Se propone aglomerar los servicios “en cadena” que se encuentran en los Client-Side, solo si es posible por criterio del desarrollador, en un solo servicio padre previo a la ejecución de las pantallas logrando una mejor gestión y otorgando la posibilidad de reutilizar este servicio padre.

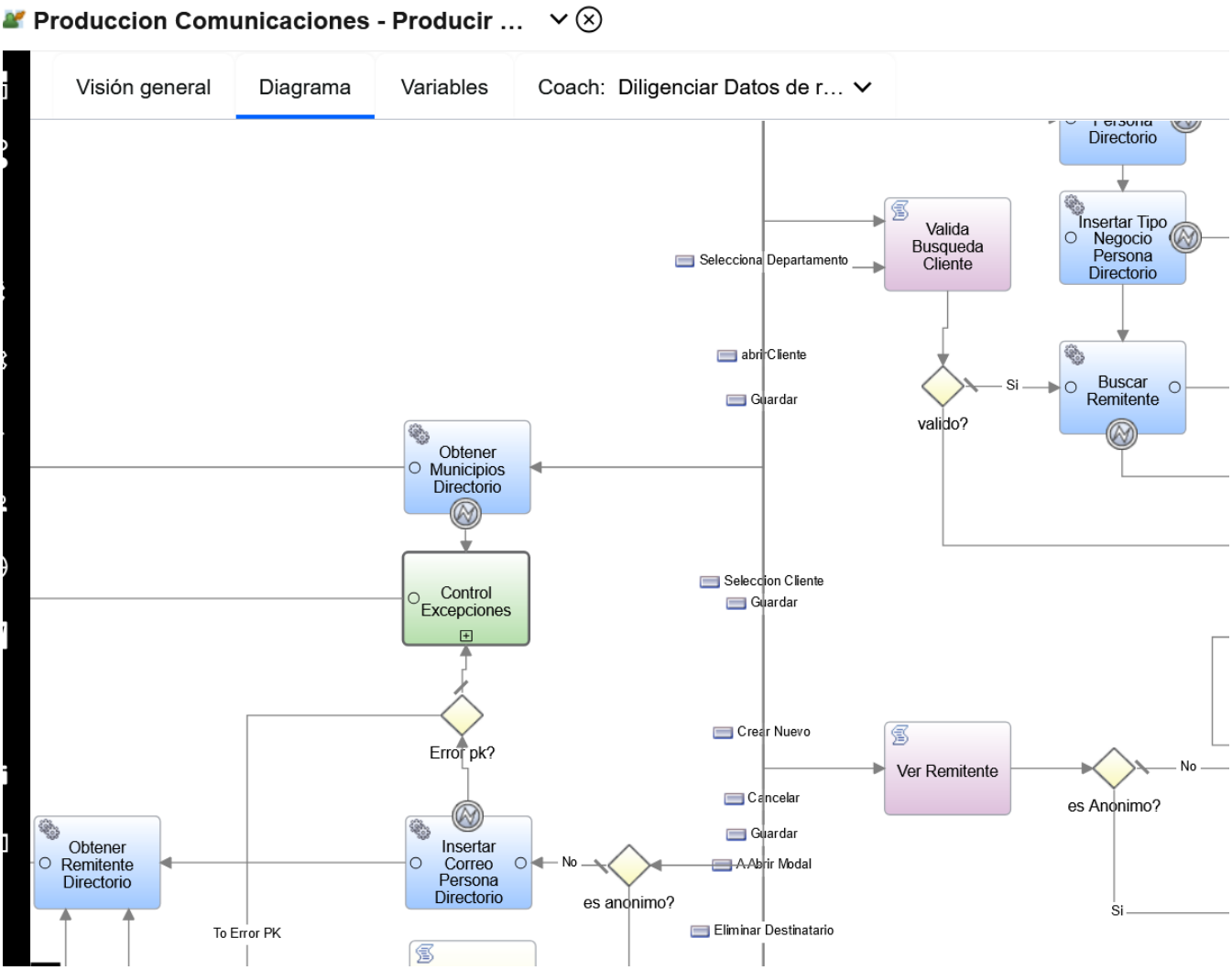
N. Se debe realizar una búsqueda y limpieza de componentes creados que no tienen un objetivo funcional. Esta es una práctica muy habitual de desarrollo que permite experimentar y probar nuevas funcionalidades de manera ágil evitando generar consecuencias en los flujos principales. Aunque una vez completado su propósito, se deben eliminar. Como figura en la siguiente evidencia de un Client-Side.



(Imagen 18)

O. Se sugiere realizar una revisión periódica de los logs de QA como una tarea proactiva, con el objetivo de aumentar la calidad del desarrollo previo al envió del ambiente productivo reduciendo las posibilidades de incidentes. Esta tarea depende del área de QA, al ser estos quienes reportan los errores y esperan la corrección de lo reportado. Se recomienda una frecuencia de una vez por mes.

P. Se propone mejorar el aspecto visual en determinados diagramas de los procesos o de los Client-Side para lograr una mejor comprensión, en casos por si el funcional requiere ver el flujo o para el desarrollador en casos de mejoras o corrección de incidentes. Se deja como evidencia un fragmento de un Client-Side donde toda esa aglomeración de botones puede llegar a generar problemas en la comprensión y entendimiento.

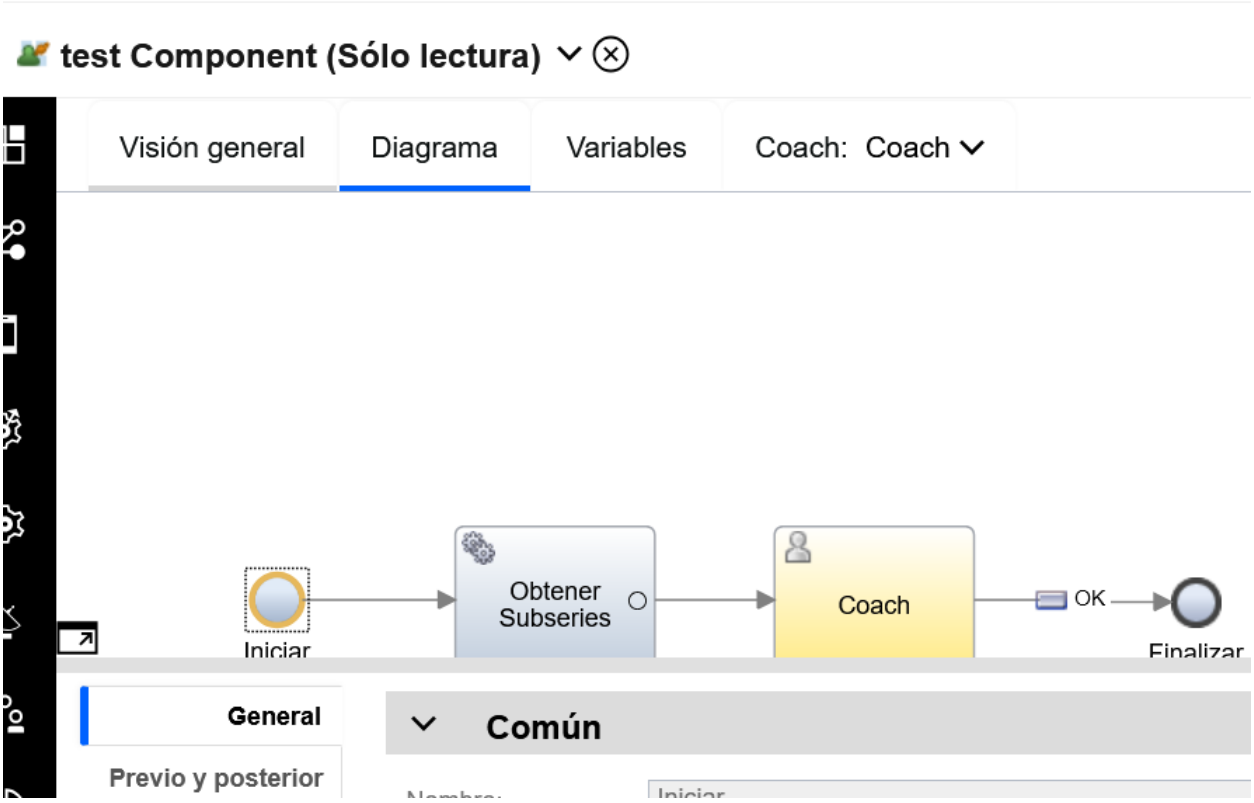

Produccion Comunicaciones - Producir ...
Visión general
Diagrama
Variables
Coach: Diligenciar Datos de r...
Directorio
Valida Busqueda Cliente
Insertar Tipo Negocio Persona Directorio
Selecciona Departamento
abrirCliente
Guardar
valido?
Si
Buscar Remitente
Obtener Municipios Directorio
Control Excepciones
Seleccion Cliente
Guardar
Error pk?
Crear Nuevo
Ver Remitente
No
es Anonimo?
Cancelar
Guardar
A Abrir Modal
Obtener Remitente Directorio
Insertar Correo Persona Directorio
No
es anonimo?
Si
To Error PK
Eliminar Destinatario

(Imagen 19)

3.2. Process App – Módulo de Administración

A. Se propone aplicar el tratamiento indicado sobre la limpieza de las variables en todos los flujos, en los Client-Side y en los servicios con el objetivo de inicializar las variables privadas y auxiliares que ya hayan cumplido su vida útil dentro del flujo, Client-Side y servicios como fue mencionado en el subtítulo “2.2” y “2.3”. Esto consiste en la creación de unos scripts ubicados a criterio del desarrollador dentro de los diagramas para inicializar las variables y dejarlas vacías. Logrando así, una mejora en la gestión de los recursos por los datos en la operatoria del producto.

B. Se debe realizar una búsqueda y limpieza de componentes creados que no tienen un objetivo funcional. Esta es una práctica muy habitual de desarrollo que permite experimentar y probar nuevas funcionalidades de manera ágil evitando generar consecuencias en los flujos principales. Aunque una vez completado su propósito, se deben eliminar. Como figura en la siguiente evidencia de un Client-Side.



(Imagen 20)

C. Se propone deshabilitar la opción de “Sincronizar automáticamente objetos de negocio compartidos” que figura, en la vista general de cada uno de los servicios, ya que no está cumpliendo ninguna función ni el propósito de esta. Se utiliza este servicio de ejemplo.



(Imagen 21)

D. Se propone habilitar la opción “Habilitar almacenamiento en memoria cache de resultados de servicio” que figura, en la vista general de cada uno de los servicios, pero solo en los casos donde los datos que se obtienen de la base de datos sean los mismos o no tengan modificaciones (Por ej. los países). Se debe utilizar la imagen 21 como ejemplo.

E. Se sugiere mejorar el aspecto visual del diagrama del Client-Side “Administración Instrumentos Archivísticos” para lograr un mejor entendimiento de este en caso de que otro desarrollador esté asignado.

4. CONCLUSIÓN

Tras la revisión del código en ambas Process Apps con sus distintos procesos y Client-Side del ambiente de QA y de Producción, se determina que hay falencias en distintos aspectos del código que ya fueron mencionados en detalle y que se deben tomar las medidas necesarias para su corrección. Además de los múltiples hallazgos en las integraciones con Filenet, que se deben tratar por la dependencia que el negocio tiene con esta funcionalidad.

Tras la revisión proactiva del mantenimiento del ambiente, se determina que no se está realizando el tratamiento adecuado y que está generando la mayoría de las consecuencias en el performance del ambiente como ya fue detallado en su respectivo subtitulo. Y por esa razón, se debería tener prioridad sobre el mantenimiento que es una actividad fundamental para alcanzar un buen estado de salud del ambiente.

Como uno de los temas tratados, la utilización de productos externos para evaluar el funcionamiento del portal no permite determinar la confiabilidad de este. Y esto es debido a que no hay documentación al respecto sobre su funcionalidad contra BAW, los criterios internos de evaluación que utiliza esa herramienta y la confiabilidad de los resultados que arroja son volátiles para este tipo de producto.