CERTIFICACIÓN DE APLICACIONES
INDICE del Anexo OTROS OP-023-01
a la Norma OP023

SE02C-01	Validación de datos y parámetros
SE02C-02	Autenticación y Autorización
SE02C-03	Gestión de Sesiones
SE02C-04	Protección de Información
SE02C-05	Control de errores y excepciones
SE02C-06	Registro y auditoría
SE02C-07	Entorno de la Aplicación

MODELO DETALLADO - PRUEBAS

QA11P Verificación Ejecución de Pruebas

QA11P-01	Existencia de documentos asociados:
QA11P-02	Ejecución de Pruebas, resultados esperados y obtenidos:
QA11P-03	Todas las incidencias / defectos (sobre pruebas técnicas) deben ser registrados y controlados conforme al nivel de detalle marcado por la instalación:
QA11P-04	Cumplimiento de los niveles de servicio (tiempo de respuesta medio de la aplicación end-to-end) establecidos y aprobados en la documentación de especificación de r
QA11P-05	Infraestructura de servidor mainframe: cumplimiento de los requisitos técnicos de operación establecidos para la plataforma y tecnología correspondiente:
QA11P-06	Infraestructura de servidor distribuido: cumplimiento de los requisitos técnicos de operación establecidos para la plataforma y tecnología correspondiente:
QA11P-07	Infraestructura de comunicaciones: cumplimiento de los requisitos técnicos de operación establecidos:
QA11P-08	Infraestructura de cliente: cumplimiento de los requisitos técnicos de operación establecidos para la plataforma y tecnología correspondiente:

GLOSARIO DE RIESGO

INSTRUCCIONES DPL PARA CICS

IMPACTO INTEGRIDAD REFERENCIAL POR GESTOR

CODIGO CONTROL		PUNTO DE CONTROL/VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Nivel de Riesgo	Automático 1=sí, 0=no
QA01A	**Verificación Simbología de Modelos de Datos**						
	QA01A-01	**La simbología de los Modelos de Datos debe contemplar (Sólo aplicable a diseños con PowerDesigner):**					
	QA01A-01.001	Representación de objetos en MD	En los niveles lógico y físico, todos los objetos del modelo de datos deben aparecer en algún diagrama de representación.	P087, C310		ALTO	1
	QA01A-01.002	Representación según funcionalidad	Se considera foco del diagrama a los objetos que motivan la aparición del diagrama. En el caso de estructuras de información, todas deben estar asignadas como estructuras foco a, al menos, un diagrama.	P087, C310		ALTO	0
	QA01A-01.003	Representación de objetos relacionados	Para dar coherencia a los diagramas, en cada diagrama deben aparecer las estructuras foco asignadas, todas las estructuras directamente relacionadas con ellas y todas las relaciones existentes entre estructuras de información presentes en el	P087, C310		MODERADO	0
QA02AD	**Verificación Modelo Lógico de Datos**						
	QA02A-01	**El diseño del modelo lógico de datos debe contemplar:**					
	QA02A-01.001	Documentación del MLD	Un Modelo Lógico de datos se considera 'completamente documentado' cuando se aporta la información que se detalla en los estándares QA03D-01.002, QA03D-01.003, QA03D-01.004, QA03D-01.005, QA03D-01.006, QA03D-01.007y QA03D-01.008, cumpliendo los estándares de modelado lógico que se detallan en este documento y los estándares de nomenclatura	P087		ALTO	0
	QA02A-01.002	Documentación del MLD	Todo modelo de datos debe tener informado su Nombre, Código y Descripción.	P087		ALTO	1
	QA02A-01.003	Documentación de diagramas	Todo diagrama debe tener informado su Nombre, Código y Descripción.	P087		ALTO	1
	QA02A-01.004	Documentación de entidades	Toda entidad debe tener informado: Nombre, Código, Descripción, Número máximo de ocurrencias estimadas, si genera Estructura Física de Información en el modelo físico de Datos y Aplicación Propietaria.	P087		ALTO	1
	QA02A-01.005	Documentación de relaciones	Toda relación debe tener informado: Entidad padre, Entidad hija, Nombre (verbo), Código, Cardinalidad, Opcionalidad y Tipo	P087		ALTO	1
	QA02A-01.006	Documentación de dominios	En el caso de tener que crear un nuevo dominio debe tener informado: Nombre, Código, Descripción, Prefijo de Código de atributo y, dependiendo del SGBD, Formato, Longitud y Precisión.	P087		ALTO	0
	QA02A-01.007	Documentación de atributos	Todo atributo debe tener informado: Nombre, Código, Descripción, Dominio, Opcionalidad, Participación en Clave Primaria y Participación en Claves Candidatas.	P087		ALTO	1
	QA02A-01.008	Documentación de claves	Toda clave del Modelo Lógico de Datos debe tener documentado: Nombre, Código, Atributos, Orden de los Atributos y si es Clave Primaria.	P087		ALTO	1
	QA02A-01.009	Documentación de entidades	El nombre de la entidad debe representar el concepto manejado de manera que pueda ser reconocido en cualquier contexto (Ej. Si una entidad recoge los clientes que intervienen en un contrato, no es válido el nombre "INTERVINIENTES" y sí lo es "CLIENTE INTERVINIENTE EN EL CONTRATO"). Debe ir en singular.	P087		ALTO	0
	QA02A-01.010	Documentación descripciones	En la definición de una entidad o atributo no se debe utilizar el término a definir o describir. Las definiciones de entidades y atributos deben ser entendidas por cualquier persona que consulte el modelo sin necesidad de consultas adicionales.	P087		MODERADO	0
	QA02A-01.011	Documentación descripciones	El nombre del atributo y dominio debe ser descriptivo del dato que contiene y debe entenderse su contenido en cualquier contexto. Los atributos se nombran en singular, empleando lenguaje natural y evitando palabras redundantes con propiedades del atributo (Ej. Si "CODIGO UNICO DE BANCO" es clave primaria en la entidad "BANCO", la palabra "ÚNICO" es redundante con la propiedad "Clave Primaria").	P087		ALTO	0
	QA02A-01.012	Nombre de relación	El nombre de toda relación debe ser un verbo, no ambiguo, que junto con su forma pasiva deben indicar cómo esta relación influye en las dos entidades implicadas.	P087		MODERADO	0
	QA02A-01.013	Nombre de especialización	El nombre de la especialización debe definir el criterio utilizado para distinguir entre los distintos subtipos a los que da origen. Por ejemplo, "Tipo de Participante" sería el nombre de la especialización que da origen a los subtipos "Persona" y "Organización" y "Tipo de Dirección" sería el nombre la especialización que da origen a los subtipos "Dirección Postal" y	P087		ALTO	0
	QA02A-01.014	Nombre de claves primarias	El nombre de las claves primarias debe coincidir con su código.	P087		ALTO	1
	QA02A-01.015	Normalización de MLD	El Modelo Lógico de Datos debe normalizarse al menos hasta Tercera Forma Normal (3FN).	P087		ALTO	0
	QA02A-01.016	Relaciones redundantes	En el Modelo Lógico de Datos no deben definirse relaciones redundantes.	P087		ALTO	0
	QA02A-01.017	Integración del MLD	Toda nueva entidad del Modelo Lógico de Datos debe estar integrada y ser coherente con el resto de entidades del Modelo Corporativo de Datos de BBVA.	P087		ALTO	0
	QA02A-01.018	Información física en MLD	En el Modelo Lógico de Datos no deben aparecer entidades o atributos que representen: seguridad física, configuración, información de control (parámetros, contadores, control de concurrencia, indicadores de proceso,...), información de rendimiento (facilidades de búsqueda, agregados, estadísticas, ...).	P087		ALTO	0
	QA02A-01.019	Alcance del MLD	El Modelo Lógico, en general, debe presentar una visión estática de la realidad, abstrayendo la dimensión temporal. Por tanto, no deben incorporarse elementos a la clave con el fin de mantener la evolución en el tiempo de una ocurrencia de la entidad.	P087		ALTO	0
	QA02A-01.020	Entidades de restricción	Se permite la definición de entidades cuya única finalidad sea garantizar por modelo ciertas restricciones (tales como limitar las posibles combinaciones entre valores de más de un atributo) que no pueden implementarse en forma de relaciones o	P087		MODERADO	0
	QA02A-01.021	Restricciones de datos	En un Modelo Lógico de Datos no deben incluirse restricciones sobre los datos y sus relaciones distintas a las de su propia naturaleza, como decisiones de implementación o limitaciones de los sistemas, que deben incorporarse en el Modelo Físico	P087		ALTO	0
	QA02A-01.022	Reglas de negocio	El Modelo Lógico de Datos debe recoger lo más completamente posible las reglas de negocio establecidas sobre los datos.	P087		ALTO	0
	QA02A-01.023	Especializaciones	Se admite el uso de especializaciones y generalizaciones con el fin de integrar en el modelo ciertas reglas de negocio, de acuerdo a lo indicado en el estándar QA03D-01.020.	P087		MODERADO	0
	QA02A-01.024	Relaciones N:M	No se admiten relaciones de cardinalidad muchos a muchos (n:m) entre entidades. La solución más inmediata para resolver este conflicto es su transformación en entidades asociativas.	P087		ALTO	1
	QA02A-01.025	Relaciones binarias	Todas las relaciones definidas en el Modelo Lógico de Datos deben ser binarias, esto es, establecidas entre dos entidades que pueden ser del mismo o diferente tipo.	P087		ALTO	0
	QA02A-01.026	Atributos en relaciones	No se permite la definición de relaciones con atributos. Cuando se detecte este caso, podrá resolverse transformando la relación en entidad o transfiriendo los atributos a una de las dos entidades asociadas mediante la relación.	P087		ALTO	0
	QA02A-01.027	Atributos en entidades	Toda entidad debe poseer al menos un atributo o participar en dos relaciones dependientes.	P087		ALTO	1
	QA02A-01.028	Solapamiento en subtipos	Sobre una misma entidad supertipo es posible definir distintos 'subconjuntos' en función de diferentes atributos clasificadores. En la especialización resultante, no debe existir solapamiento entre los subtipos, debe de ser una especialización exclusiva.	P087		ALTO	0
	QA02A-01.029	Entidades con igual clave	La existencia de varias entidades con una misma clave primaria sólo es posible cuando formen parte de una estructura tipo / subtipo. En las especializaciones / generalizaciones, las claves primarias del supertipo y los subtipos deben ser idénticas.	P087		ALTO	0
	QA02A-01.030	Relación transferible	Las entidades que participan en una relación 1:1 no pueden fundirse en una cuando dicha relación sea 'transferible' o	P087		ALTO	0
	QA02A-01.031	Entidades relacionadas	En el modelo Lógico todas las entidades (no SGAT) deben participar al menos en una relación.	P087		ALTO	1
	QA02A-01.032	Restricciones del MLD	En el Modelo Lógico no se admiten atributos derivados ni diseñados, excepto fechas, horas e instantes en el tiempo.	P087		ALTO	0
	QA02A-01.033	Definición de claves	Para toda entidad del modelo deben haberse identificado todas las 'claves candidatas' y seleccionado una 'clave primaria' entre ellas. Toda clave candidata, incluida la primaria, debe cumplir las siguientes propiedades: - Estar formada por un conjunto mínimo de atributos. - Existir para cualquier ocurrencia de la entidad. -Identificar de forma única toda ocurrencia de la entidad.	P087		ALTO	0
	QA02A-01.034	Estabilidad de clave primaria	La clave primaria seleccionada para una entidad debe ser 'estable' a lo largo del tiempo.	P087		ALTO	0
	QA02A-01.035	Entidades dependientes	Debe evitarse la definición de entidades dependientes en identificación cuando la relación establecida entre las dos entidades involucradas sea transferible.	P087		ALTO	0
	QA02A-01.036	Entidades dependientes con opcionalidad	No es posible definir entidades dependientes en identificación cuando pueda existir 'opcionalidad' en la existencia del padre.	P087		ALTO	1
	QA02A-01.037	Asignación de dominios	Los atributos deben asignarse a dominios existentes en la instalación (disponibles en la herramienta de modelado PowerDesigner). No se admite la creación de ningún nuevo dominio. Aplicable sólo para diseños en PowerDesigner.	P087		MODERADO	1
	QA02A-01.038	Dominios dinámicos	No se permite la asignación de atributos a dominios discretos dinámicos, que pueden cambiar con el tiempo. En estos casos el atributo siempre debe originar una nueva entidad.	P087		ALTO	0
	QA02A-01.039	Dominios discretos estáticos	Si el dominio de un atributo es de tipo discreto estático, se asigna al atributo el dominio no discreto que defina los valores que puede contener, haciendo notar en la definición del elemento que es un Dominio Discreto Estático y se incluye entre su documentación la lista de valores que puede tomar (Ej. Sexo se asignaría al dominio Tipo y aparecerían los valores Varón y Mujer). En este caso no es necesario crear una entidad asociada.	P087		ALTO	0
	QA02A-01.040	Unicidad de atributos en el modelo corporativo	No se admite que un atributo posea diferentes definiciones (con tipos de datos diferentes) en BBVA. Sólo se admite distinta codificación para un atributo en el caso de que ese atributo tenga que aparecer más de una vez en una entidad, pero incluso así debe mantener el resto de propiedades (tipo de dato, dominio,...)	P087		ALTO	1

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
QA05D	**Verificación Entidad/tabla SGAT**								
	QA05D-01	**Una Entidad/tabla SGAT debe contemplar:**							
	QA05D-01.001		Definición de entidad/tabla SGAT	Las tablas que pueden solicitarse en SGAT son del tipo código-descripción	Modelo lógico de datos		E	ALTO	0
	QA05D-01.002		Definición de entidad/tabla SGAT - I.R	No pueden tener integridad referencial con otras tablas del modelo corporativo. No se permiten tablas SGAT de conceptos ya definidos en otras tablas del banco.	Modelo lógico de datos		E	ALTO	0
	QA05D-01.003		Definición de entidad/tabla SGAT - datos estables	Los datos deben ser estables, con poca volatilidad (número reducido de bajas y modificaciones)	Modelo lógico de datos		E	ALTO	0
	QA05D-01.004		Definición de entidad/tabla SGAT - número de filas	El total de filas que contenga no debe ser excesivo para evitar problemas de rendimiento en las consultas on-line (una estimación entre 100 y 200 filas aproximadamente).	Modelo lógico de datos		E	ALTO	0
	QA05D-01.005		Definición de entidad/tabla SGAT - multiidioma	Sólo el campo de descripción tiene la capacidad multiidioma	Modelo lógico de datos		E	ALTO	0
	QA05D-01.006		Definición de entidad/tabla SGAT - atributos adicionales	Además de los campos de código y descripción se permiten otros atributos hasta un máximo de dos. Estos atributos no pueden contener texto susceptible de tratamiento multiidioma.	Modelo lógico de datos		E	MODERADO	0
	QA05D-01.007		Definición de entidad/tabla SGAT - longitud del código	La longitud del campo código está establecida entre **1** y **10** caracteres. Adaptando la longitud al número de filas	Modelo lógico de datos		E	ALTO	0
	QA05D-01.008		Definición de entidad/tabla SGAT - longitud de la decsripción.	La descripción tiene una longitud fija de **50** caracteres.	Modelo lógico de datos		E	ALTO	0
QA06D	**Verificación Modelo Físico de Datos**								
	QA06D-01	**El diseño físico modelo de datos debe contemplar:**							
	QA06D-01.002		Redundancias e incrongruencias.	Se revisará el modelo en su conjunto buscando redundancias y/o incongruencias.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.003		Tablas numeradoras.	Deben evitarse. En caso de ser necesarias no deben contener más de un numerador por tabla, se dispondrán en fila por página y se actualizará mediante SELECT FOR UPDATE. Se pueden evitar utilizando campos IDENTITY, opción SEQUENCE (sólo para instalaciones con V8 de DB2) y con combinaciones de columnas, por ejemplo: DOCUMENTO + ULTIMOS DÍGITOS DE CCC PRODUCTO + SUB-PRODUCCTO + BCO. + OFICINA + FECHA BCO. + OFICINA + ULTIMOS DÍGITOS DE CCC + SECUENCIAL (por oficina)	Modelo lógico de datos		E	BAJO	0
	QA06D-01.004		Tablas semáforo	Se entiende por tablas semáforo aquellas estructuras de datos, almacendas en el gestor de bd, cuya información es utilizada para gestionar procesos, tanto on-line como batch. Un proceso activa un campo y otro está esperando un valor concreto en dicho campo para realizar su funcionalidad. Este tipo de tablas no deben utilizarse, se deben sustituir por otros recursos del sistema: MQ, CICS, OPC, etc.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.005		Tablas verticales	Se entiende por tabla vertical aquella que almacena los atributos del objeto concreto en filas y no en columnas. Ejemplo tipo de estas tablas son aquellas que llevan una clave, un campo que indica que atributo se almacena en la siguiente columna, y la columna con el valor del atributo. Este tipo de tablas no deben utilizarse puesto que para recuperar todos los atributos de un elemento deben leer más de una fila, no cumplen la primera forma normal puesto que todas las filas no tienen porqué tener informados todos los atributos. En caso de necesitarse deberán ser autorizadas por el departamento técnico.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.006		Tablas de información calculada	Se entiende por tablas con información calculada a aquellas tablas formadas por la desnormalización de tablas, o aquellas que, para cada clave, contienen atributos que son el resultado de operar sobre otros existentes en las tablas normalizadas. Se recomienda que sólo se usen cuando se justifique por rendimiento y, siempre y cuando, los datos no sean volátiles. Las tablas con información calculada son un recurso de Data Warehouse. Ejemplo de este tipo de tablas: valores medios de saldos por contrato, valores liquidativos de fondos, estructuras organizativas "aplanadas" para falicitar el acceso, catálogo de productos desnormalizado para mejorar el proceso de contratación, etc.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.007		Tablas de información diaria (logs, diarios de operaciones, etc.) críticas en el proceso on-line	Las tablas que se usan como logs diarios y que se vacian al final del día deberán plantearse como flip/flop. Se cargarán preformateadas con ficheros "dummy". Deben conocerlas los departamentos técnicos. En instalaciones con V8 de DB2 se analizará la viabilidad de declararlas con la opción VOLATILE.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.008		Tablas de paso o de datos volátiles	No deben utilizarse. En caso de necesitarse deberán ser autorizadas por el departamento técnico. En instalaciones con V8 de DB2 se analizará la viabilidad de declararlas con la opción VOLATILE.	Modelo lógico de datos		R	BAJO	0
	QA06D-01.009		Tablas aisladas, sin relación con el resto del modelo.	Deben evitarse salvo en el caso de tablas históricas o de log. En caso de ser necesarias deberán ser autorizadas por el departamento técnico.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.010		Información histórica	No se deben mantener los datos históricos y los activos en las mismas tablas. - Deberá enviarse a histórico un dato una vez que ya no esté vivo (sea susceptible de modificación) y las consultas sobre éste sean esporádicas o puntuales => Deberá indicarse la periodicidad del envío a históricos (semanal, mensual, anual, quinquenal, etc.).	Modelo lógico de datos		E	MODERADO	0

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA06D-01.011		Tablas de históricos	Sólo se crearán como tabla si van a ser consultados habitualmente por procesos en línea, en caso contrario se almacenarán en ficheros. - La estructura del histórico deberá ser igual que la original si va ha ser accedida por los mismos procesos o preparada ad-hoc sólo con la información que necesita guardarse. Sobre las tablas de históricos se definirán los índices necesarios para las consultas siempre y cuando no sean tablas con volúmenes elevados, ya que la creación de índices no particionados sobre tablas muy grandes, genera severos problemas en la administración. La generación de índices sobre estas tablas deberá ser supervisada por el departamento técnico correspondiente.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.012		Normalización del Modelo	El modelo físico debe estar en 3era forma normal, cualquier desnormalización debe pasar por la aprobación de los grupos técnicos.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.013		Clave única.	Todas las tablas del Modelo Físico de datos deben tener identificadas los columnas que forman su clave primaria, independientemente de si se implementa o no físicamente el índice asociado a la misma. Generalmente se creará un índice único sobre la misma.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.014		Revisión de la definición de claves e índices para cada una de las tablas.	Se tendrá en cuenta el modelo global así como el uso de cada entidad, para balancear el beneficio obtenido en los accesos frente al coste de actualización por cada índice. - Se recomienda no definir más de **2** índices por entidad => ajustar los accesos, en fase de diseño y construcción, para no requerir un número superior de índices. - Se debe prestar atención a la composición de los índices, para asegurar que no se omiten columnas que aporten cardinalidad, o que se incluyen columnas que no aportan ningún beneficio. - No deben definirse índices redundantes, es decir no puede existir un índice con todas sus columnas incluidas en otro. - No deben omitirse columnas que lógicamente deben formar parte del índice. - No deberán incluirse en el índice campos del tipo timestamp, o similar. Sí son admisibles las fechas, y para las horas deberá analizarse su utilidad en cada caso.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.015		Diseño de índice clúster.	El índice clúster de cada tabla se decidirá en base a su utilidad en procesos de selección y tratamiento masivos, tanto en on-line como en batch.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.016		Análisis de particionamiento de tablas	Se analizarán las tablas candidatas a ser particionadas, bien por el volumen de datos, según estándar de cada instalación, bien por el tipo de actividad que se realizará sobre ellas, es decir, necesidades de especial concurrencia (esto deberá ser indicado por DyD).	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.017		Tablas particionadas: consideraciones de partición.	-En versiones de DB2 anteriores a V8, la clave de partición es el índice Cluster . - El reparto de datos entre las particiones debe ser homogéneo. - Los accesos a los datos deben ser homogéneos entre las particiones, en caso contrario plantear el envío a tablas o ficheros históricos en lugar del particionamiento. - Criterios alfabéticos o numéricos pueden dar un reparto homogéneo (p.e.: nombre apellido, DNI, núm recibo, etc.) - Criterios de negocio pueden dar problemas para obtener un reparto homogeneo (p.e.: producto subproducto, divisa, etc.) - Otros criterios para decir el particionamiento no pueden detectarse en este momento del ciclo de vida del desarrollo. P.e. más de 4 niveles en índices.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.018		Formatos de columnas congruentes	Las columnas que se incluyen en más de una tabla deberán tener el mismo formato y longitud. A igualdad de campo, igualdad de formato.	Modelo lógico de datos		E	MODERADO	0
	QA06D-01.019		Valores nulos en columnas.	Para DB2 Host no se usarán definiciones de columnas con valor "null", por defecto se utilizará: "not null with default" y se podrán definir columnas con obligatoriedad de información: "not null".	Modelo lógico de datos		R	ALTO	0
	QA06D-01.020		Columnas de longitud variable `(VARCHAR)`.	Las columnas de longitud variable sólo se permitirán cuando se justifique su uso.	Modelo lógico de datos		R	BAJO	0
	QA06D-01.021		Columnas con valores constantes.	No está permitido definir columnas que almacenen un valor constante en todas las filas/renglones de la tabla.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.022		Columnas de fecha y hora.	Se deben definir todas las columnas que son fecha como DATE, los de hora como TIME y las combinaciones de fecha y hora como TIMESTAMP.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.023		Naturaleza de las columnas	Se deben definir las columnas según la naturaleza de los datos que almacenan. Sólo serán columnas numéricas aquellas que almacenen datos sobre los que se realizan operaciones matemáticas. El resto deberán ser de tipo carácter (salvo las de fecha y hora).	Modelo lógico de datos		E	BAJO	0
	QA06D-01.024		Columnas numéricas	Para datos numéricos, usar decimales cuando en verdad se requieran usar decimales, de otra manera definir los datos tipo: - `INTEGER` (-147483648 hasta 2147483647) o - `SMALLINT` (de –32768 hasta 32767) o - cuando vayan a guardar valores tan grandes que no puedan ser soportados por los tipos de datos enteros, el valor máximo que soporta un decimal es 10(31) – 1.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.025		Columnas de tipo imagen, video, sonido, etc.	Para columnas que albergan atributos de tipo imagen, sonido, video, etc. deberán utilizarse columnas tipo `LOB`, `BLOB` o `CLOB`, nunca `VARCHAR`.	Modelo lógico de datos		E	BAJO	0
	QA06D-01.027		Orden de las columnas en las tablas	El orden de las columnas en las tablas debe ser tal que las columnas más actualizadas estén al final de la fila y, dentro de las columnas actualizables, las de tipo `varchar` al final.	Modelo lógico de datos		R	BAJO	0
	QA06D-01.028		Integridad referencial	Se podrá utilizar el manejo de Integridad Referencial por Gestor siempre y cuando se analicen las implicaciones dependiendo del gestor utilizado y las necesidades de la aplicación. OCTA realizará este análisis y validación del manejo de la Integridad Referencial.	Modelo lógico de datos		R	ALTO	0

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA06D-01.029		Uso de vistas `(view)`	Se sugiere el uso de vistas para simplificar el mantenimiento de programas cuando se adicionan columnas a las tablas, es decir, cuando la tabla es creada se crea una vista con todas las columnas, cuando se adicionen columnas a la tabla, se generará una nueva vista que incluya las columnas anteriores más las columnas nuevas, así solo se tendrán que modificar los programas que hagan referencia a las columnas nuevas. La vistas serán generadas con una proyección vertical de la tabla. En caso de necesitar hacer uso de WHERE o JOIN entre varias tablas, tendrá que ser revisado y autorizado por el departamento técnico de Bases de Datos. Es importante mencionar, que al crear las vistas no se debe usar la sentencia SELECT *, sino que se debe indicar puntualmente el codigo de cada una de las columnas.	Modelo lógico de datos			BAJO	0
	QA06D-01.030		Uso de validaciones por gestor (constraints, etc.)	Para hacer uso de validaciones de campos manejada por el Gestor es necesario que el departamento técnico de Bases de Datos lo autorice.	Modelo lógico de datos			BAJO	0
	QA06D-01.031		Definición detallada de los ficheros maestros (organización, formato de registro, tipo de acceso), definiendo las características físicas del fichero (nombre, tamaño, ubicación, etc.).	Se revisará el diseño físico de los ficheros. Es necesario verificar: - La definición de la organización, el formato y el tipo de acceso son coherentes, es decir deberán detectarse casos como el uso de un fichero indexado con acceso secuencial. (para aquellos VSAM que se admitan). - La idoneidad del uso de ficheros en lugar de entidades de bbdd, en los casos que pueda ser aplicable, dependerá, entre otras cosas, de la necesidad de acceso por clave al fichero, lecturas repetidas de un mismo fichero buscando un registro determinado, etc.	Definición de Copy Definición de Fichero / archivo		E	BAJO	0
	QA06D-01.032		Definición de la estructura del fichero (copys, headers, etc.)	Debe verificarse que el diseño de los ficheros se corresponde con el modelo de datos, y con el resto de elementos de definición (copy, headers, etc.). Revisar también la estructura interna de los ficheros, verificando que no existen incongruencias o incumplimiento de normativas o estándares de la instalación, en cuanto al uso de ficheros y tipos/formatos de campos.	Definición de Copy Definición de Fichero / archivo		R	BAJO	0
	QA06D-01.033		Cumplimiento de los estándares de nomenclatura de los componentes	Se revisarán las definiciones de los elementos del modelo de datos, verificando que se ajustan, en nomenclatura y formato, a los estándandares de la instalación.	Definición de Copy Definición de Fichero / archivo Modelo físico de datos		E	BAJO	0
	QA06D-01.034			Está prohibido el uso de `DELETE CASCADE`. El borrado de una fila padre y todas sus relacionadas debe hacerse por programación para evitar problemas de escalado de bloqueos y generación de indisponibilidad de las tablas implicadas así como la degradación del gestor.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.035			Está prohibido el uso de `UPDATE CASCADE`. La propagación de cambios en las claves primarias debe hacerse por programación para evitar problemas de escalado de bloqueos y generación de indisponibilidad de las tablas implicadas así como la degradación del gestor.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.036			El establecimiento de integridad referencial por gestor se hace a nivel de aplicación. Si una aplicación está en producción y no tiene definida Integridad Referencial, no se podrá habilitar esta opción ni para tablas viejas ni para tablas nuevas. Cualquier aplicación que no ha llegado a producción puede elegir utilizar la opción Integridad Referencial. por gestor. Se entiende por aplicación el conjunto de objeto definidos bajo un código `PAAA` según nomenclatura global.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.037			Cuando se utiliza Integridad Referencial mantenida por gestor, el número máximo de niveles de anidamiento "padre-hijo" es "3".	Modelo lógico de datos		E	ALTO	0
	QA06D-01.038			Queda prohibido el uso de Integridad Referencial por gestor en modelos informacionales.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.039			Está prohibido establecer relaciones de Integridad Referencial mantenida por gestor entre tablas de diferentes aplicaciones o entre tablas de diferentes Bases de Datos. Se entiende por aplicación el conjunto de objeto definidos bajo un código `PAAA` según nomenclatura global.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.040			Para reducir el impacto en la indisponibilidad de las tablas (y por tanto en el servicio), no se podrá recuperar sobre sí misma una tabla a un momento anterior en el tiempo. Se recuperará la tabla sobre un fichero para `DB2 HOST` y se decidirá cómo arreglar el problema que origina la recuperación. Para el resto de gestores se estudiará en cada caso.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.041			Está prohibido utilizar mecanismos de Integridad Referencial recursivos sobre la misma tabla. Estas validaciones se tienen que hacer por programa.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.042			Está prohibido realizar `LOAD` con `REPLACE` a una tabla padre. Para alimentar este tipo de tablas se utilizarán programas de aplicación.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.043		IR. Procesos de carga	Los procesos de carga vía utilidad batch de tablas hijas tienen que realizarse verificando la integridad de los datos en tiempo de carga. Esto significa: EN `DB2`: Está prohibido realizar `LOAD` con `REPLACE` a una tabla hija con el parámetro `ENFORCE = NO`. Se tienen que utilizar siempre `ENFORCE = YES`. EN `ORACLE`: Está prohibido realizar `LOAD` con `REPLACE` en una tabla hija con la opción `DIRECT = YES` EN `DB2 UDB`: Es necesario realizar tras el proceso de `LOAD`, un proceso diferenciado de `SET INTEGRITY`	Modelo lógico de datos		E	ALTO	0
	QA06D-01.044		Inclusión de columnas en tablas DB2 Host	La inclusión de columnas en tablas que están en producción sólo se puede hacer si se añaden al final de la misma. Estas tendrán formato "`NOT NULL WITH DEFAULT`"	Modelo lógico de datos		E	ALTO	0
	QA06D-01.045		Aplicable a Modelos Informacionales: Objetivo del modelo de datos	Expresar el objetivo del modelo de datos propuesto. Este no debería ser la mera duplicación de un grupo de tablas del entorno origen.	Modelo lógico de datos		R	BAJO	0

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA06D-01.046		Aplicable a Modelos Informacionales: Expresar cuatro o cinco preguntas de negocio	Para modelos informacionaes, expresar cuatro o cinco preguntas de negocio que van ser resueltas mediante el area de datos implementada.	Modelo lógico de datos		R	BAJO	0
	QA06D-01.047		Valores nulos en columnas (Para Oracle, DB2 UDB y Postgre SQL)	Para Oracle, DB2 UDB y Postgre SQL, no se usarán definiciones de columnas con valor "null" cuando formen parte de la clave primaria.	Modelo lógico de datos		R	MODERADO	0
	QA06D-01.048		Particionamiento de tablas en Oracle	A partir de 15 millones de registros, se debe evaluar la conveniencia de particionar la tabla. A partir de 25 millones, es obligatorio particionar.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.049		Inclusión de columnas en tablas Oracle, DB2 UDB y Postgre SQL	La inclusión de columnas en tablas Oracle, DB2 UDB y Postgre SQL que están en producción sólo se puede hacer si se añaden al final de la misma.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.050		Documentación del MFD	Un Modelo Físico de datos se considera 'completamente documentado' cuando se aporta la información que se detalla en los estándares QA04D-01.001, QA04D-01.002, QA04D-01.004, QA04D-01.005, QA04D-01.006 y QA04D-01.007, cumpliendo los estándares de modelado físico que se detallan en este documento y los estándares de nomenclatura descritos en la norma TC-002.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.051		Documentación del MFD	Todo **modelo de datos** debe tener informado su Nombre, Código y Descripción.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.052		Documentación de diagramas del MFD	Todo **diagrama** debe tener informado su Nombre, Código y Descripción.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.053		Documentación de tablas	Toda tabla del Modelo Físico de Datos debe tener documentado: Nombre, Código, Descripción, Clave Primaria, Índices, Filas en explotación (máximo estimado teniendo en cuenta el ciclo de vida del dato). Adicionalmente dependiendo del Sistema de Gestión de Base de Datos (SGBD) debe documentarse : DB2: Índice Cluster, Aplicación Propietaria, Tablespace, Tipo actualización, Filas en explotación, Frecuencia de actualización, Filas otros entornos, Gestor, Equipo, Base de Datos, Propietario, si se aplica Backup y Medidas de seguridad aplicables. Oracle: Owner, Particionamiento (tablas e índices). Teradata: Compresión, Particionamiento.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.054		Documentación de columnas	Toda columna del Modelo Físico de Datos debe tener documentado: Nombre, Código, Descripción, Opción de Nulos, Dominio y Tipo de Dato en el gestor (Formato y Longitud acordes con los permitidos para el dominio seleccionado, expuestos en la norma AR-001).	Modelo lógico de datos		E	ALTO	1
	QA06D-01.055		Documentación de relaciones del MFD	Toda relación del Modelo Físico de Datos debe tener documentado: tabla padre, tabla hija, Nombre (verbo), Código, Cardinalidad y Opcionalidad.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.056		Documentación de índices	Todo **índice** del Modelo Físico de Datos debe tener documentado: Nombre, Código, Columnas, Orden de las Columnas, Orden de los Datos de cada Columna (ascendente o descendente), si es Único y si es Cluster.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.057		Documentaciónde claves del MFD	Toda **clave** del Modelo Físico de Datos debe tener documentado: Nombre, Código, Columnas, Orden de las columnas y si es Clave Primaria.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.058		Dominios del MFD	Los dominios del Modelo Físico de Datos se corresponden con los dominios del Modelo Lógico de Datos, siendo aplicables los estándares QA03D-01.006, QA03D-01.037, QA03D-01.038 y QA03D-01.039. Los columnas representan el papel de los atributos para estas normas	Modelo lógico de datos		E	ALTO	1
	QA06D-01.059		Documentación de descripciones	En la definición o descripción detallada de las columnas y tablas no se debe utilizar el término a definir o describir. Las definiciones de columnas y tablasdeben ser entendidas al consultar el modelo sin necesidad de consultas adicionales.	Modelo lógico de datos		R	MODERADO	0
	QA06D-01.060		Documentación de descripciones	El nombre de la tabla debe representar el concepto manejado de manera que pueda ser reconocido en cualquier contexto (Ej. Si una tabla recoge los clientes que intervienen en un contrato, no es válido el nombre "INTERVINIENTES" y sí lo es "CLIENTE INTERVINIENTE EN EL CONTRATO"). Debe ir en singular.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.061		Nombre de relación del MFD	El nombre de toda relación debe ser un verbo, no ambiguo, que junto con su forma pasiva deben indicar cómo esta relación influye en las dos tablas implicadas.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.062		Nombre de columnas y dominios	El nombre del columna y dominio debe ser descriptivo del dato que contiene y debe entenderse su contenido en cualquier contexto. Las columnas se nombran en singular, empleando lenguaje natural y evitando palabras redundantes con propiedades de la columna (Ej. Si "CODIGO UNICO DE BANCO" es clave primaria en la entidad "BANCO", la palabra "ÚNICO" es redundante con la propiedad "Clave Primaria").	Modelo lógico de datos		E	ALTO	0
	QA06D-01.063		Nombre de índices y claves	El **nombre de los índices y las claves primarias** debe coincidir con su código.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.064		Foreign Keys completas	Al establecer una relación entre dos estructuras de datos debe cumplirse que todos las columnas de la clave **primaria** de la estructura que participa como padre en la relación, figuran como columnas en la estructura que participa como hijo de la relación.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.065		Relaciones redundantes	No está permitido el establecimiento de relaciones redundantes entre estructuras físicas de información.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.066		Coherencia entre MLD y MFD	Debe mantenerse la coherencia entre el MLD y el MFD. Toda tabla del MFD que almacene información del negocio debe proceder de una o más entidades del MLD. Como excepción, las tablas de histórico, parámetros o de log no tienen correspondencia en el MLD	Modelo lógico de datos		E	ALTO	0

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA06D-01.067		Obtención de MFD	La forma natural de obtención del Modelo Físico de Datos a partir del Modelo Lógico de Datos es: .- Cada entidad se convierte en una tabla, cuyos columnas son los atributos de la entidad. La clave primaria de la tabla es el conjunto de columnas correspondientes a los atributos que forman la clave primaria de la entidad. .- De cada relación entre dos entidades del Modelo Lógico se obtiene la relación física entre las tablas correspondientes. En general debe procurarse que el MFD se encuentre normalizado hasta Tercera Forma Normal. Cualquier excepción a estas reglas de transformación, se considera desnormalización y la mejora esperada con la transformación debe ser justificada en la descripción del objeto que la origina (Integración con otras aplicaciones, Rendimiento de procesos críticos, Esquema de distribución, Ahorro en almacenamiento, Facilidades para el desarrollo, Restricciones del Sistema de Gestión de Base de Datos (SGBD), Diferenciación de históricos, Requerimientos especiales: Disponibilidad, Confidencialidad...)	Modelo lógico de datos		E	ALTO	0
	QA06D-01.068		Tablas con misma PK	En general, deben combinarse en una única tabla las entidades con la misma clave primaria y que mantengan entre sí asociaciones 1:1. Razones que justifican mantener la diferenciación de tablas son: .- Que posean un significado muy distinto para el negocio. Con seguridad, esto supone una marcada diferenciación en las asociaciones y en las funciones de negocio. .- Necesidad de distribución de datos en distintas plataformas. .- Existencia de distintos requerimientos de seguridad lógica. .- Restricciones del Sistema de Gestión de Base de Datos (SGBD). .- Consideraciones de rendimiento. .- Transferibilidad de la asociación en el MLD de procedencia.	Modelo lógico de datos		R	MODERADO	0
	QA06D-01.069		réplicas con actualización concurrente	No se deben plantear soluciones de distribución de datos mediante réplicas que contemplen la actualización concurrente del mismo conjunto de datos.	Modelo lógico de datos		R	MODERADO	0
	QA06D-01.070		tamaño de claves de acceso	Debe evitarse la definición de claves de acceso que contengan columnas de tipo alfanumérico de longitud superior a 30 caracteres. Como alternativa, podría estudiarse la posibilidad de definir nuevos columnas derivados que aporten una funcionalidad semejante.	Modelo lógico de datos		R	MODERADO	1
	QA06D-01.071		Definición de columnas	Para cada columna, se elegirá el tipo de dato más apropiado. Cuando existan varios tipos de datos que satisfagan correctamente las necesidades de un determinado atributo, se elegirá aquél que menos espacio físico ocupe.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.072		columnas multipropósito	No se admite la utilización de columnas multipropósito, esto es, que se pueda guardar información de distinta naturaleza en función de determinados criterios.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.073		concatenación de atributos	No se admite la concatenación de varios atributos en una sola columna.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.074		procedencia de columnas	Todo columna que almacene información de negocio debe proceder de algún atributo del MLD, No tendrán correspondencia aquellos que no almacenen información de negocio (campos de auditoría, campos para multiidioma)	Modelo lógico de datos		E	ALTO	0
	QA06D-01.075		naturaleza de códigos	Los columnas de naturaleza código deben tener formato alfanumérico. Se utilizará formato numérico sólo en el caso de utilizar objeto sequence para asignarle valores o se incremente de manera secuencial.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.076		Reglas de Integridad	Todo columna que forme parte de una clave candidata y claves ajenas surgidas de una asociación con obligatoriedad del padre, debe implementarse de forma que sea el Sistema de Gestión de Base de Datos (SGBD) quien garantice que no admita valores nulos (IR por gestor).	Modelo lógico de datos		E	ALTO	1
	QA06D-01.077		Documentación de modelo de vistas	Las vistas están restringidas salvo aceptación de los grupos técnicos. Para el gestor Teradata es obligatorio el acceso por vistas. Cuando las vistas están aceptadas, se considera 'completamente documentado' cuando se aporta la información que se detalla en los estándares QA06D-01.078, QA06D-01.079 y QA06D-01.080, cumpliendo los estándares de modelado físico que se detallan en este documento y los estándares de nomenclatura descritos en la norma TC-002.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.078		Documentación de modelo de vistas	Las vistas debe tener informado su Nombre, Código y Descripción.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.079		Documentación de diagrama de modelo de vistas	Todo **diagrama** debe tener informado su Nombre, Código y Descripción.	Modelo lógico de datos		E	ALTO	1
	QA06D-01.080		Documentación de vistas	Todas las Vistas tienen que heredar la documentación del Modelo Físico de Datos (tablas y columnas de las que se alimenta).	Modelo lógico de datos		E	ALTO	0
	QA06D-01.081		Origen de las vistas	Todas las Vistas tienen que crearse a partir de tablas del Modelo Físico de Datos. No se admiten vistas que accedan a otras vistas.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.082		Vistas fijas	No está recomendada la creación de lógica dentro de las Vistas. Solo se deben crear vistas fijas de las tabla del MFD o uniones simples de las tablas. Cualquier condicional creado para la vista (where) debe ser justificado.	Modelo lógico de datos		E	ALTO	0
	QA06D-01.083		Forma en Modelo de Vistas	La forma natural de obtención de las Vistas es a partir del Modelo Físico de Datos	Modelo lógico de datos		E	ALTO	0

CODIGO CONTROL		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
		QA06D-01.084		Definición de columnas en DB2	Se debe evitar en lo posible el uso de columnas con tamaño superior a 254 caracteres pues tienen algunas restricciones en DB2: .- No pueden formar parte de índices .- No admiten predicados distintos de LIKE .- No pueden usarse en SUBSELECT .- No pueden aparecer en ORDER BY ni en DISTINCT	Modelo lógico de datos		E	ALTO	0
		QA06D-01.085		Definición de índices en DB2 - orden de columnas	En ausencia de otros criterios, en la definición del índice se coloca delante los columnas con mayor número de valores distintos.	Modelo lógico de datos		E	ALTO	0
		QA06D-01.086		Definición de índices en DB2 - columnas de longitud variable	No se permite (salvo excepciones admitidas por grupos técnicos) la definición de índices con columnas de longitud variable.	Modelo lógico de datos		E	ALTO	0
		QA06D-01.087		Definición de índices en DB2 - columnas muy actualizadas	Evitar la definición de índices secundarios con columnas frecuentemente actualizadas.	Modelo lógico de datos		E	ALTO	0
		QA06D-01.088		Definición de índices en DB2 - nulos	No pueden crearse índices únicos sobre columnas que admiten nulos.	Modelo lógico de datos		E	ALTO	1
		QA06D-01.089		Definición de columnas en Oracle	Para datos alfanuméricos se utilizará siempre el tipo VARCHAR2.	Modelo lógico de datos		E	ALTO	1
		QA06D-01.090		Definición de índices en Oracle	En ausencia de otros criterios, en la definición del índice se coloca delante los columnas con mayor número de valores distintos.	Modelo lógico de datos		R	MODERADO	0
		QA06D-01.091		Definición de I.R. en Oracle	La implementación de las reglas de Integridad de Tabla **(Clave Primaria)** se efectúa siempre a través del Sistema de Gestión de Bases de Datos (SGBD).	Modelo lógico de datos		E	ALTO	1
		QA06D-01.092		Definición de columnas en Teradata - compress	Es recomendable que se haga un estudio de las columnas candidatas a compresión, con el fin de reducir el espacio de almacenamiento en base de datos	Modelo lógico de datos		E	ALTO	1
		QA06D-01.093		Definición de columnas en Teradata -varchar	Las columnas definidas con formato CHAR de gran longitud y contenido variable deben definirse como VARCHAR.	Modelo lógico de datos		E	ALTO	0
		QA06D-01.094		Definición de índices en Teradata - distribución de datos	En ausencia de otros criterios, en la definición del índice se debe seleccionar la combinación de columnas que permitan una buena distribución de los datos.	Modelo lógico de datos		E	ALTO	1
		QA06D-01.095		Definición de vistas en Teradata	Se debe definir una vista asociada a cada tabla que sea una imagen especular de esta.	Modelo lógico de datos		E	ALTO	1
		QA06D-01.096		Definición de particionamiento en Teradata	Se debe considerar particionar una tabla cuando su tamaño sea muy grande.	Modelo lógico de datos		E	ALTO	0
		QA06D-01.097		Unicidad de campos en el modelo corporativo	No se admite que un campo posea diferentes definiciones (con tipos de datos diferentes) en BBVA. Sólo se admite distinta codificación para un campo en el caso de que ese campo tenga que aparecer más de una vez en una tabla, pero incluso así debe mantener el resto de propiedades (tipo de dato, dominio,...) Por ejemplo para el código de oficina, con code : COD_OFIALFA. Siempre que se haga referencia a este concepto se debe de utilizar este code. Solo se admite codificar con otro code si una tabla, por ejemplo la de Contrato, ya tiene está columna en la tabla y necesita una segunda oficina, podrá utilizar otro code para el concepto oficina.	Modelo lógico de datos		E	ALTO	1
		QA06D-01.98		Integridad Referencial por clave primaria	Como norma general la IR se realizará utilizando la clave primaria, sólo a petición de la aplicación con su debida justificación y aceptada por el departamento técnico correspondiente se propagará una clave secundaria o alternativa.	Modelo lógico de datos		E	ALTO	0
		QA06D-01.99		Integridad Referencial por clave.	Entre dos tablas (tabla padre y tabla hija) sólo se permite Integridad Referencial por una de las claves (primaria o secundaria)	Modelo lógico de datos		E	ALTO	0
QA06D	QA06D-02	**El diseño físico modelo de datos de Teradata debe contemplar:**								
		QA06D-02.001		Tablas semáforo	En Teradata, actualmente no existen este tipo de tablas, en caso de que en un futuro, fueran solicitadas por parte de DYD, tendrian que ser validadas, por el departamento de Adm Teradata.	Modelo lógico de datos		R	BAJO	0
		QA06D-02.002		Tablas de información diaria (logs, diarios de operaciones, etc.) críticas en el proceso on-line.	En Teradata, no aplica. En caso de que en un futuro DYD nos comunicara la inclusion en el modelo de cargas de este tipo de tablas, habra que estudiar su funcionamiento.	Modelo lógico de datos		R	BAJO	0
		QA06D-02.003		Tablas de paso o de datos volátiles .	**No deben utilizarse.** En caso de necesitarse deberán ser autorizadas por el departamento técnico. La implicación de este tipo de tablas se encuentra en la limitación de spool de usuario.El usuario puede crearlas sin control.	Modelo lógico de datos		E	BAJO	0
		QA06D-02.004		Clave única.	El índice primario de la tabla debe ser seleccionado teniendo en cuenta la distribución y el acceso a datos que se va a realizar sobre la misma. Teniendo en cuenta esto, este índice puede ser único o no único. El uso de los índices en Teradata solo se realiza en las sentencias si se indican todas las columnas del índice, no se puede hacer un uso del índice parcial como en otros gestores. La creación de un índice único secuencial y la selección de esta como índice primario SOLO da una buena distribución, pero normalmente nunca será utilizado en el acceso.	Modelo lógico de datos		E	MODERADO	0
		QA06D-02.005		Análisis de particionamiento de tablas	Se analizarán las tablas candidatas a ser particionadas, bien por el volumen de datos, según estándar de cada instalación, bien por el tipo de actividad que se realizará sobre ellas, es decir, necesidades de especial concurrencia (esto deberá ser indicado	Modelo lógico de datos		E	BAJO	0
		QA06D-02.006		Columnas de longitud variable `(VARCHAR)`.	Para aquellos atributos de tipo carácter y de gran longitud y que no tengan una longitud fija en todas sus filas podrán utilizarse columnas como VARCHAR.	Modelo lógico de datos		E	BAJO	0
		QA06D-02.007		Columnas de fecha y hora.	Se deben definir todas las columnas que son fecha **completa** como `DATE`, los de hora **completa** como `TIME` y las combinaciones de fecha y hora como `TIMESTAMP`.	Modelo lógico de datos		E	BAJO	0

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA06D-02.008		Datos numéricos	Para datos numéricos, usar decimales cuando en verdad se requieran usar decimales, de otra manera definir los datos tipo: - BITEINT (-128..127) o - SMALLINT (de –32768 hasta 32767) o - INTEGER (-2147483647..2147483647) o - BIGINT (-9223372036854775808.. 9223372036854775808) - cuando vayan a guardar valores tan grandes que no puedan ser soportados por los tipos de datos enteros, el valor máximo que soporta un decimal es 10(38) – 1.	Modelo lógico de datos		E	BAJO	0
	QA06D-02.009		Uso de vistas (view)	Se sugiere el uso de vistas para simplificar el mantenimiento de programas cuando se adicionan columnas a las tablas, es decir, cuando la tabla es creada se crea una vista con todas las columnas, cuando se adicionen columnas a la tabla, se generará una nueva vista que incluya las columnas anteriores más las columnas nuevas, así solo se tendrán que modificar los programas que hagan referencia a las columnas nuevas. La vistas serán generadas con una proyección vertical de la tabla. En caso de necesitar hacer uso de WHERE o JOIN entre varias tablas, tendrá que ser revisado y autorizado por el departamento técnico de Bases de Datos. Es importante mencionar, que al crear las vistas no se debe usar la sentencia SELECT *, sino que se debe indicar puntualmente el nombre de cada una de las columnas. En Teradata no se admiten las vistas con logica. Estas, en caso de ser necesarias, deberan de ser estudiadas por el departamento de Adm Teradata.	Modelo lógico de datos		E	MODERADO	0
	QA06D-02.010		Uso de validaciones por gestor (constraints, etc.)	Actualmente la normativa BBVA no permite la existencia de integridad referencial dentro de Teradata. Es una buena práctica la implementación de integridad referencial débil con el objetivo de asegurar al sistema la existencia de dicha integridad referencial. El gestor no valida dicha información, sino que la asume sacando provecho en la explotación de la tabla relacionada.	Modelo lógico de datos		E	MODERADO	0
	QA06D-02.011		Vistas en TERADATA	Todas las vistas definidas en TERADATA deben establecer por defecto, bloqueo a nivel de fila con opción "locking row for access mode nowait".	Modelo lógico de datos		B	ALTO	0
	QA06D-02.012		Compresión de campos en Teradata	Es conveniente comprimir los valores altamente repetidos en un campo (con un máximo de 255 valores por campo) siempre y cuando el campo no forme parte de un índice o los valores sean susceptibles de ser cambiados o sustituídos en el futuro.	Modelo lógico de datos		E	MODERADO	0
	QA06D-02.013		Palabras reservadas en la definición de objetos TERADATA	Los valores BEFORE JOURNAL , AFTER JOURNAL , FALLBACK y CHECKSUM no deben ser indicados al generar el PDM ya que vienen por defecto. Si se quisieran habilitar especificamente para una tabla, habrá de ser aprobado por el equipo técnico de TERADATA .	Modelo lógico de datos		E	MODERADO	0
	QA06D-02.014		Campos de indices en TERADATA	Teradata permite VARCHAR en los PI, campos comprimidos dentro del PI como ya se ha comentado no los soporta el gestor.	Modelo lógico de datos		B	ALTO	0
	QA06D-02.015		Construir sql de alto nivel que resuelvan las queries de una manera aproximada	Construir sqls de alto nivel que resuelvan las preguntas de negocio de una manera aproximada. Prestar especial atención a las relaciones entre entidades y sus accesos posibles y/o probablemente más frecuentes (join entre tablas o vistas, con sus columnas relacionadas y las posibles cláusulas 'where')	Modelo lógico de datos		E	BAJO	
	QA06D-02.016		Expresar el objetivo de rendimiento asociado.	Expresar el objetivo de rendimiento asociado en los distintos aspectos del modelo: carga , uso en otros procesos de carga y distintos tipos de explotación (query libre, MSTR etc.). Debe determinarse este objetivo relacionado con el perfil de usuario (carga, carga crítica, explotación táctica, etc.)	Modelo lógico de datos		E	BAJO	
	QA06D-02.017		Primary Key - Alternate Key	Documentar o tener acceso directo a la documentación de cada PK o AK de los objetos origen de información, área de stage incluida.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.018		Indices	Documentar o tener acceso directo a la documentación de cada índice de los objetos origen de información. Especificar el Primary Index de las tablas Teradata usadas. Área de stage incluida.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.019		Relaciones entre tablas (Foreign Keys, etc)	Documentar o tener acceso directo a la documentación de cada FK de los objetos origen de información.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.020		Refresco del dato en origen y dependencias.	Documentar o tener acceso directo a la documentación de la carga de cada objeto origen (alta, baja, baja lógica, modificación). Documentar su periodicidad de carga y dependencias funcionales, con su planificación de carga asociada.	Modelo lógico de datos		E	BAJO	
	QA06D-02.021		Definición de tablas SET versus MULTISET	Definir cada tabla SET o MULTISET dependiendo de las características de la entidad cargada. Es preferible definir la tabla MULTISET si el hecho de que puedan almacenarse registros iguales duplicados no es un requerimiento. No definir una tabla como 'SET' si se puede prever que va a ser voluminosa y no se requiere que no tenga registros estrictamente duplicados.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.022		Nivel de bloqueo en la lectura	Gestionar apropiadamente el nivel de bloqueo en la lectura de una tabla mediante vistas, teniendo en cuenta que no siempre se requiere lectura sucia. En tiempo de carga puede ser necesaria la lectura con bloqueo para asegurar la integridad del dato, supuesto que las dependencias y ventanas determinadas en planificación de la carga estén correctamente implementadas.	Modelo lógico de datos		E	BAJO	
	QA06D-02.023		Definir el PK (y AK) de cada tabla, aunque no se implemente	Documentar el PK (y AK's) de cada tabla. Documentar si se implementan como índice primario único (UPI) o secundario único(USI) o no se implementan físicamente.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.024		Uso por defecto de PK o AK	No usar por defecto el PK o AK de una tabla como UPI o USI. Evaluar cada caso particular puesto que en algunas ocasiones no se consigue así la mejor opción de acceso a esa tabla.	Modelo lógico de datos		E	BAJO	
	QA06D-02.025		Columnas nulables en el primary index	Especificar como not null si el atributo no va a contener valores nulos (el modelo origen o la transformación en proceso de carga permite asegurar que el atributo no va a contener valores nulos)	Modelo lógico de datos		E	BAJO	
	QA06D-02.026		Tipo de dato correcto, reducido y consistente a lo largo del modelo en el primary index	Se debe mantener el tipo de dato correcto, reducido y consistente a lo largo del modelo y en relación con el mismo atributo ya presente en otras entidades relacionadas.	Modelo lógico de datos		B	ALTO	

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA06D-02.027		Atributos concatenados en el primary index	No definir columnas compuestas por concatenación de atributos si los items individuales aparecen definidos o usados separadamente en otras entidades relacionadas.	Modelo lógico de datos		B	ALTO	
	QA06D-02.028		Definición del Primary Index por criterios de acceso y distribución	No definir la composición del Primary Index por criterios que no sean los de acceso y distribución. En lo posible se deben considerar conjuntamente.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.029		Distribución equilibrada por primary index	El Primary Index debe definirse teniendo en cuenta que es deseable una distribución uniforme. Una buena distribución proporciona buenos tiempos de carga relativos y de explotación. La distribución desequilibrada de una tabla produce un empeoramiento global del skew de la base de datos a la que pertenece.	Modelo lógico de datos		E	BAJO	
	QA06D-02.030		Acceso a la tabla por primary index en tablas de tipo 'hecho'	Para tablas de tipo 'hecho', en esquemas de explotación, la Primary Key no es usada como patrón de acceso. Por tanto, en este esquema, no es correcta la elección del primary index igual a primary key. El Primary Index podría componerse de tal manera que cubra algún join crítico (muy usado, muy voluminoso, etc.)	Modelo lógico de datos		E	MODERADO	
	QA06D-02.031		Manipulación en el acceso por columnas que componen el Primary Index	No acceder a la tabla mediante transformaciones a las columnas que componen el Primary Index (conacatenación, substring, operadores, desigualdad, etc). No construir vistas con estas características.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.032		Determinación del Primary Index en función de los accesos	Todo acceso debe tenerse en cuenta para la determinación del Primary Index, incluyendo los procesos de carga, forma de backup, etc.	Modelo lógico de datos		R	BAJO	
	QA06D-02.033		Primary Index común a un grupo de tablas	Es recomendable la definición del mismo Primary Index para un determinado grupo de tablas relacionadas, cuya explotación se repite de manera frecuente usando alguna columna común a todas ellas. Esto debe compaginarse con la obtención de un porcentaje de skew aceptable en las tablas relacionadas.	Modelo lógico de datos		R	BAJO	
	QA06D-02.034		Elección de la composición del Primary Index	Es recomendable especificar las razones para la elección de la composición del Primary Index (distribución, acceso join, igualdad)	Modelo lógico de datos		R	BAJO	
	QA06D-02.035		Compresión fuera del Primary Index	Como no se pueden comprimir las columnas que pertenecen al primary index, la elección de una composición reducida del primary index, que cumpla en el mismo grado con los mismos criterios de acceso y distribución que otra composición con mayor número de columnas, permite la compresión de las restantes columnas. Esto puede suponer un ahorro alto en almacenamiento.	Modelo lógico de datos		R	BAJO	
	QA06D-02.036		Uso de claves subrogadas en el primary index	Usar claves subrogadas para manejo de atributos parte de claves primarias con tipo de dato de longitud muy grande. Pueden usarse también en situaciones de claves primarias con muchos atributos.	Modelo lógico de datos		R	BAJO	
	QA06D-02.037		Definición de columnas grandes en el primary index.	No definir columnas de longitud grande como parte del Primary Index.	Modelo lógico de datos		R	BAJO	
	QA06D-02.038		Atributos integrantes del Primary Index	En el Primary Index usar atributos que no van a ser objeto de modificación masiva frecuente ('update').	Modelo lógico de datos		E	MODERADO	
	QA06D-02.039		Número de columnas que componen el primary index (entre 6 y 10)	A la hora de determinar las columnas que conformarán el índice primario de una tabla, se deberá seleccionar el mínimo número de campos posible que garanticen el mayor número de accesos y una distribución homogénea de los datos. Recomendándose, siempre que sea posible, un número de columnas no superior a 5. Si el número de columnas que componen el primary index está entre 6 y 10, el nivel de riesgo es Crítico.	Modelo lógico de datos		E	BAJO	
	QA06D-02.040		Número de columnas que componen el primary index (entre 11 y 15)	A la hora de determinar las columnas que conformarán el índice primario de una tabla se deberá seleccionar el mínimo número de campos posible que garanticen el mayor número de accesos y una distribución homogénea de los datos. Recomendándose, siempre que sea posible, un número de columnas no superior a 5. Si el número de columnas que componen el primary index está entre 11 y 15, el nivel de riesgo es Alto.	Modelo lógico de datos		E	MODERADO	
	QA06D-02.041		Número de columnas que componen el primary index (más de 15)	A la hora de determinar las columnas que conformarán el índice primario de una tabla se deberá seleccionar el mínimo número de campos posible que garanticen el mayor número de accesos y una distribución homogénea de los datos. Recomendándose, siempre que sea posible, un número de columnas no superior a 5. Si el número de columnas que componen el primary index es mayor de 15, el nivel de riesgo es Muy Alto.	Modelo lógico de datos		B	ALTO	
	QA06D-02.042		En caso de producirse una inserción (insert select) desde área de stage hacia las bases de datos del modelo final.	Si es posible, por conocer la estructura de la tabla del modelo final, usar la misma composición para el primary index en ambas tablas. Si no es posible, en un primer momento se plantea un mecanismo simple de definición del primary index de stage, usando el primary key de la aplicación de origen, cuando exista y se conozca.	Modelo lógico de datos		E	BAJO	
	QA06D-02.043		Condiciones para particionar tablas	No se debe particionar por defecto. Antes hay que observar que las tablas objeto de partición han de ser grandes, la selectividad de lo accedido debe ser alta con respecto al total, el acceso por partición va a ocurrir en un porcentaje cercano al total y la implementación de la forma de acceso permite su uso. Igualmente debe demostrarse que se efectúa acceso por partición.	Modelo lógico de datos		E	BAJO	
	QA06D-02.044		Definición de partición multinivel	Para mejorar la selectividad de lo accedido puede implementarse una estructura de partición multinivel, existiendo una limitación del número total de combinaciones de los valores de las particiones definidas en cada columna. No definir partición multininivel sin la seguridad que algunos de los niveles son usados en cada sentencia, y que cada nivel es usado en alguna sentencia.	Modelo lógico de datos		R	BAJO	
	QA06D-02.045		Composición del Primary Index Particionado	Especificar las razones para la elección de la composición del P.P.I. (distribución, acceso join, igualdad)	Modelo lógico de datos		R	BAJO	

CODIGO CONTROL		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
		QA06D-02.046		Cuando considerar la creación de Secondary Index y de Particionamiento	En un primer momento del ciclo de desarrollo no se deben considerar la creación de índices secundarios si no está justificado por uso posterior (producción controlada). De la misma manera, la definición de particiones en tablas debe hacerse en la fase de producción controlada.	Modelo lógico de datos		E	BAJO	
		QA06D-02.047		Estadísticas	Al construir el modelo físico definir columnas, índices o tuplas a coleccionar. Una primera aproximación puede ser la toma de estadísticas de las tuplas y elementos individuales que componen PK, FK, AK, PI y SI.	Modelo lógico de datos		E	BAJO	
		QA06D-02.048		Elección de la composición del Secondary Index	Especificar las razones para la elección de la composición del Secondary Index. (Uso transversal, acceso join, igualdad, aseguramiento de unicidad)	Modelo lógico de datos		R	BAJO	
		QA06D-02.049		Selectividad prevista del Secondary Index	Documentar en lo posible el tamaño total previsto y su selectividad. El índice secundario no debería medir más que un tercio del tamaño total de la tabla.	Modelo lógico de datos		R	BAJO	
		QA06D-02.050		Composición del Join Index	Especificar las razones para la elección de la composición del Join Index. (Uso transversal, igualdad, agregación, cambio en distribución por nuevo primary index)	Modelo lógico de datos		R	BAJO	
		QA06D-02.051		Compresión específica del join index.	No se puede hacer compresión de las columnas individuales existentes en un join index. Por tanto, implementar la compresión específica del join index en su definición.	Modelo lógico de datos		E	BAJO	
		QA06D-02.052		Buenas prácticas con el Join Index	Consideraciones de análisis similares a las del Primary Index, sin considerar la compresión multivalor.	Modelo lógico de datos		E	BAJO	
QA06D	**QA06D-03**	**Restriciones del Diseño para MQWorkflow**								
		QA06D-03.001		No utilizar campos estándar de MQWorkflow para introducir información de múltiples propósitos con el objetivo de utilizarlos en consultas con wildcards	No se permite utilizar campos de MQWorkflow para codificar información diversa en los campos estándar de MQWorkflow, como por ejemplo "DESCRIPTION" o "NAME" y luego establecer filtros con comodines sobre los mismos. Para ello crear un conjunto limitado de campos sobre una tabla nueva e indexarlos, siguiendo las recomendaciones de uso del "Global Data Container" establecido en las "best practices" del support pack WA0B de IBM en http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24006852&loc=en_US&cs=utf-8&lang=en El campo DESCRIPTION puede ser usuado para mostrar campos de la "Global Data Container", pero no para referenciarlos.	N/A		B	ALTO	0
QA06D	**QA06D-04**	**Restriciones del Diseño para tablas replicadas**								
		QA06D-04.001		Procesos de LOAD	Las tablas que tengan procesos periódicos de LOAD en el HOST no podrán ser replicadas en Oracle. En caso de que se necesite replicar, se estudiará la sustitución del proceso de LOAD con el departamento técnico correspondiente.	N/A		B	ALTO	0
		QA06D-04.002		Procesos de LOAD	No está permitido crear JCLs de carguen datos en tablas DB2 Host replicadas a Oracle a través de utilidades LOAD, salvo autorización por el grupo técnico correspondiente.	N/A		B	ALTO	0
		QA06D-04.003		Actualización de tablas	No está permitida la actualización de datos de tablas replicadas directamente en Oracle. Todas las actualizaciones se realizarán en DB2 Host y por el proceso de replicado automático serán actualizadas en Oracle.	N/A		B	ALTO	0
		QA06D-04.004		Procedimiento de emergencia	En tablas replicadas, se prohíbe la recuperación de los datos de tablas Host realizada a través de una utilidad RECOVER directamente sobre la tabla. Las acciones de recuperación se realizarán directamente el grupo técnico a partir del fichero de datos, logs y fecha de recuperación y modificando los datos de la tabla a través de programa con sentencias SQL. De esta forma, la restauración de los datos en ORACLE será transparente pues la propagará el CDC (mecanismo automático de replicación).	N/A		B	ALTO	0
		QA06D-04.005		Naturaleza de los atributos	No se admiten tablas replicadas con campos VARCHAR con longitud superior a 4000 en DB2 Host debido a una limitación de ORACLE para versiones inferiores a 12c.	N/A		B	ALTO	1
		QA06D-04.006		Autorización para el uso de tablas replicadas	Antes de replicar una tabla se justificará su necesidad y uso para el cual se replica.	N/A		B	ALTO	0
QA07D	**Verificación Diseño de Sistemas OnLine**									
	QA07D-01	**El diseño de los sistemas online debe contemplar:**								
		QA07D-01.001		Definición de funcionalidades en servidor de aplicaciones	Se revisarán las funcionalidades que según el diseño de los sistemas online se alojarán en el servidor de aplicaciones (host o ssdd), verificando la idoneidad de esta ubicación en los casos que puedan plantear dudas: - Validación de datos en memoria. - Validación de datos introducidos por pantalla (fechas, importes, NIF,etc.), en referencia a formatos y obligatoriedad de campos. - Validación de datos o información de contexto (fecha, terminal, etc). * Se debe aplicar la recomendación de acercar al máximo la funcionalidad al terminal, descargando así las comunicaciones y los servidores de aplicaciones	Detalle de la funcionalidad Inventario de componentes		E	BAJO	0

CODIGO CONTROL		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
		QA07D-01.002		Definición de funcionalidades como componente gráfico reutilizable	Se revisarán las funcionalidades (identificación en diseño de elementos asimilables a componentes GUI de arquitectura ya paquetizados) que según el diseño de los sistemas online se definen como componentes gráficos reutilizables, verificando la idoneidad de esta ubicación en los casos que puedan plantear dudas, por requerir datos o funciones del servidor de aplicaciones: - Validación de datos introducidos por pantalla (fechas, importes, NIF,etc.), entendidas como validaciones del lado del cliente, que sin ser directamente visuales, pueden afectar al comportamiento de los componentes distribuidos o en ciertos casos estar encapsuladas en elementos visuales - Validación de datos o información de contexto (fecha, terminal, etc). * Se debe aplicar la recomendación de acercar al máximo la funcionalidad al terminal, descargando así las comunicaciones y los servidores de aplicaciones	Detalle de la funcionalidad Inventario de componentes	SÓLO CPD	E	BAJO	0
		QA07D-01.003		Diseño del sistema cliente	Se deben minimizar los accesos a los servidores finales de aplicaciones tipo host. Cada navegación, entendiéndose como tal, los procesos transcurridos entre que un usuario realiza un "click" y se visualizan los datos, debe contemplar un número máximo de ejecución de transacciones, siendo recomendable sólo una.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	0
		QA07D-01.004		Acceso a datos en procesos On-line	Todos los accesos a datos deben ser óptimos, en cuanto a su número y al acceso en sí, revisar la utilización de índices.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	0
		QA07D-01.005		Número de tablas accedidas en procesos On-line.	Se revisarán aquellos procesos que accedan a más de 5 tablas operativas.	Detalle de la funcionalidad Inventario de componentes		E	BAJO	0
		QA07D-01.006		Accesos únicos a los datos	No se realizará más de un acceso en consulta sobre la misma tabla, todas las verificaciones sobre una tabla deben realizarse en una única consulta por ejecución del proceso.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	0
		QA07D-01.007		Modificación de los datos	Las modificaciones, se realizan de una sola vez, al finalizar la transacción, facilitando la marcha atrás, y minimizando los tiempos de bloqueo de datos.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	0
		QA07D-01.008		Puntos de sincronismo	En procesos on-line, cada uno es una unidad lógica de proceso y sólo una. Es decir, una transacción sólo debe llevar un punto de sincronismo al final de la misma. Las transacciones se deberán diseñar bajo el concepto de "unidad lógica de trabajo" y deberán ser lo más cortas posibles.	Detalle de la funcionalidad Inventario de componentes		E	ALTO	0
		QA07D-01.009		Transacciones pseudoconversacionales	Todas las transacciones deberán ser pseudoconversacionales, esto es, cada vez que se emita un mensaje, se despliegue un mapa, etc., deberá finalizar la transacción y ejecutarse un punto de sincronismo.	Detalle de la funcionalidad Inventario de componentes		E	ALTO	
		QA07D-01.010		Catálogo de errores	Es conveniente generar un catálogo de errores para que en caso de fallo de una transacción, el mensaje que se le envíe al usuario esté asociado a una descripción de error que sea más clara y específica tanto para el usuario como para las áreas técnicas.	Detalle de la funcionalidad Inventario de componentes		R	BAJO	
		QA07D-01.011		Diseño de listas	Para conseguir un tiempo de respuesta adecuado en las consultas por listas es necesario acotar los límites de búsqueda, bien por pantalla, bien mediante la construcción de accesos concretos a cada posibilidad de selección. Evitar las consultas abiertas solucionadas mediante sentencias abiertas de consulta a la base de datos.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	
		QA07D-01.012		Valores por defecto a los campos de entrada	Se recomienda que se establezcan valores por defecto en los campos de entrada que condicionan las búsquedas para evitar que se produzcan cvonsultas muy abiertas. También sirve validar que se rellenen todos los campos necearios para la ejecución.	Detalle de la funcionalidad Inventario de componentes		R	BAJO	
		QA07D-01.013		Monitorización y automatización	Verificar que el diseño del sistema online considere los procedimientos operativos y controles suficientes de monitoreo y automatización	N/A	SÓLO CCR	E	BAJO	
		QA07D-01.014		Uso de recursos MQ-SERIES	Verificar que los objetos de `MQ-SERIES` requeridos por la aplicación son los necesarios para su funcionamiento óptimo (colas y canales).	Detalle de la funcionalidad Inventario de componentes		R	BAJO	
		QA07D-01.015		Servicios MQ-SERIES	Los servicios aplicativos propios de comunicación coneste producto `MQ-SERIES` deben ser proporcionados por los desarrollos del área de Arquitectura Aplicativa CDR, es responsabilidad de ellos en base a las necesidades de la aplicación proponer las características de los objetos a definir en `MQ SERIES`.	N/A	SÓLO CCR	E	BAJO	
		QA07D-01.016		Comunicación con aplicaciones No Host	Se deberá utilizar `MQ SERIES` para realizar la interfaz con aplicaciones no host (servidores internos).	Detalle de la funcionalidad Inventario de componentes		R	BAJO	
		QA07D-01.017		Afinidad a los sistemas	Las transacciones serán diseñadas sin afinidades a los sistemas que las ejecutan a efecto de poderlas ejecutar en cualquier servidor de la instalación. Esto aplica tanto a `CICS` como a `IMS`.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	
QA08D	**Verifcación Diseño de Sistemas Batch**									
	QA08D-01	**Deben existir los documentos marcados por la Metodología BBVA para el diseño de sistemas batch:**								
		QA08D-01.001		Definición de Cadena Batch	Verificar la existencia del documento en la carpeta de red correspondiente. Verificar que el contenido hace referencia al desarrollo en curso.	Definición de cadena batch		E	BAJO	
	QA08D-02	**La documentación generada, según la metodología BBVA, debe ser completa e integra, es decir debe existir el correspondiente documento de definición para cualquier elemento referenciado:**								

CODIGO CONTROL	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
	QA08D-02.001		Definición de Cadena Batch	Verificar que todos los elementos mantienen la integridad referencial entre los documentos de la definición de sistemas batch, es decir que todo documento/elemento referenciado existe y que todo documento/elemento que existe está referenciado por algún otro documento/elemento.	Definición de cadena batch		E	MODERADO	
QA08D-03	**El diseño de los sistemas batch debe ser claro y completo, conteniendo la siguiente información:**								
	QA08D-03.001		Definición de las características de la la cadena, es decir: equipo, periodicidad, día y hora de ejecución, criticidad, modo de actuación ante fallos (rearranques), interacción con el online.	Se revisará el diseño de los sistemas batch, verificando que se relacionan los parámetros básicos para su definición, y que éstos son acordes con los Requerimientos de Nivel de Servicio: - Periodicidad - Dia y Hora de ejecución (relación con eventos externos) - Criticidad - Actuación en caso de fallos, puntos de rearranque. - Interacción con el online	Definición de cadena batch		E	MODERADO	0
	QA08D-03.002		Inventario de procesos, indicando sus relaciones de predecesores y sucesores.	Se revisarán las dependencias entre los procesos, verificando que las relaciones de predecesores y sucesores son coherentes con los Requerimientos de Nivel de Servicio, y la definición de procesos	Definición de cadena batch		E	MODERADO	0
	QA08D-03.003		Diagrama de flujo de la cadena	Se revisarán la definición global de la cadena, a partir del diagra de flujo, verificando la corrección de todas las definiciones y parámetros, así como su coherénciа con el resto de definiciones de la aplicación	Definición de cadena batch		E	BAJO	0
QA08D-04	**Verificación de estándares y buenas prácticas de diseño de Sistemas Batch:**								
	QA08D-04.001		Modelo de procesos Batch.	Analizar el diagrama estudiando si la agrupación de funcionalidades y servicios es coherente. Se pueden plantear agrupaciones o desagrupaciones en función de las necesiades de cada aplicativo.	Detalle de la funcionalidad Inventario de componentes		E	MODERADO	0
	QA08D-04.002		Dependencias de procesos.	No deben existir condiciones redundantes o condiciones bloqueantes (bucle de condiciones).	Definición de cadena batch		E	MODERADO	0
	QA08D-04.003		Camino crítico independiente	Los procesos que pertenecen a los caminos críticos de la aplicación deben contener sólo las actividades imprescindibles codificando en otros procesos actividades de generación de listados y similares.	Detalle de la funcionalidad Inventario de componentes		E	ALTO	0
	QA08D-04.004		Información disponible más de un día	Se deben utilizar ficheros generacionales para gestionar información que debe permanecer más de un día en el sistema.	Definición de cadena batch		E	BAJO	0
	QA08D-04.005		Paralelización de procesos	Se analizará la posibilidad de paralelización de procesos en función de las acciones que se realicen.	Definición de cadena batch		E	BAJO	0
	QA08D-04.006		Técnicas de reducción de tiempo de proceso en entrada/salida	Se debe considerar el uso de técnicas de procesamiento "batch pipe" para reducir el tiempo de duración entre dos procesos en los cuales uno genera un fichero y el otro debe tratarlo. Esto permite paralelizar procesos que comparten ficheros.	Definición de cadena batch		E	BAJO	0
	QA08D-04.007		Revisión y detección de puntos de rearranque en caso de error.	Revisar la definición de actuación en caso de fallo y los puntos de rearranque, asegurando que es completa y que no se producen inconsistencias o riesgos de carácter técnico.	Definición de cadena batch		E	MODERADO	0
	QA08D-04.008		Tamaño de los procesos (jcl) host	No deberan tener más de 20 pasos o 450 líneas.	Definición de cadena batch		R	BAJO	0
	QA08D-04.009		Procesos homogeneos	Diseñar el JCL como una unidad de tratamiento homogénea. Evitar la concatenación de tratamientos dispares.	Definición de cadena batch		E	BAJO	0
	QA08D-04.010		Uso de utilidades para tratamientos complejos	Utilidades como `STARBAT` o `SORT` deben ser utilizados con precaución, sin sobrecarga de parámetros. Ante selecciones completas es necesario plantearse el desarrollo de un programa.	Definición de cadena batch		E	BAJO	0
	QA08D-04.011		Planificación dinámica	No se permite la creación y ejecución de procesos de forma dinámica, todo debe estar definido y controlado por el planificador automático.	Definición de cadena batch		E	ALTO	0
	QA08D-04.012		Generación de informes en QMF	No está permitido el uso de `QMF` para generar informes, deben construirse por programa.	Definición de cadena batch		E	ALTO	0
	QA08D-04.013		Evitar el uso de utilidades o lenguajes interpretados	Procesos realizados en `EASYTRIEVE` o `REXX`, en caso de utilizarse deben ser compilados antes de su puesta en producción.	Definición de cadena batch		E	MODERADO	0
	QA08D-04.014		Uso de CONTROL-D	No se permite el uso de las facilidades de `CONTROL-D` para ordenar, formatear o separar informes/reportes.	Definición de cadena batch		E	MODERADO	0
	QA08D-04.015		Horario de planificación	Los procesos batch serán planificados en horiario no "on-line", cualquier necesidad de planificación en horario de oficina debe ser aceptado por los responsables de producción.	Definición de cadena batch Descripción detallada del servicio		E	MODERADO	0
	QA08D-04.016		Rutinas Batch/Online	No se deben utilizar en batch rutinas diseñadas para uso online	Definición de cadena batch		E	MODERADO	0
QA08D-05	**Verificación de la Gestión de Datos en el diseño de Sistemas Batch:**								
	QA08D-05.001		Análisis y revisión de los accesos y tratamientos de datos	En general se analizarán todos los procesos de accesos a datos, validaciones, inserciones y modificaciones.	Definición de cadena batch		R	BAJO	0
	QA08D-05.002		Actualización masiva de datos	Para actualizaciones masivas de datos se debe analizar la viabilidad de utilizar utilidades de descarga, tratamiento y carga (cuidado implica parada de la tabla a tratar durante todo el proceso).	Definición de cadena batch		E	BAJO	0
	QA08D-05.003		Inserciones masivas de datos	Para inserciones masivas de datos, superiores al 10% del total de registros de una tabla se debe utilizar LOAD RESUME YES, **siempre y cuando la tabla no tenga actividad línea en el momento de la carga.**	N/A	SÓLO CCR	E	BAJO	0
	QA08D-05.004		Borrados masivos de datos	Para borrados masivos de datos, depuración de tablas analizar la viabilidad de utilizar la utilidad `REORG` con opción `DELETE`. Mientras se realiza el `REORG` para borrar no pueden haber procesos de actualización sobre filas que estén afectadas por el `WHERE` del `DELETE`. **En CPD Europa:** el REORG se considera utilidad de Administración de Datos que DYD no debe utilizar en procesos de aplicación: DYD no tiene forma de solicitar incluir un REORG como parte de su cadena de aplicación	Definición de cadena batch	SÓLO CCR	E	BAJO	0
	QA08D-05.005		Accesos únicos a los datos	Se deben agrupar los accesos a los datos bajo un mismo proceso siempre y cuando sea posible.	Definición de cadena batch		R	BAJO	0

CODIGO CONTROL		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
		QA08D-05.006		Unificar pasos que tratan archivos de entrada comunes	El objetivo es evitar lecturas redundantes del mismo fichero. Esto podría hacerse mediante un programa que seleccionara los tipos de registros que serán utilizados posteriormente. Es necesario evitar que un fichero (sobretodo si es grande) sea leido reiteradas veces con diferentes objetivos, una vez para seleccionar un subconjunto de datos, otra para seleccionar o tratar otro, etc., en estos casos se recomienda analizar la viabilidad de aunar en un proceso de lectura único con salidas múltiples para procesos posteriores. Sólo si es posible.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.007		Procesos de validación de datos	Las validaciones contra la misma tabla deben hacerse de una vez.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.008		Procesos de selección mediante ordenación	Evitar el uso de varios procesos de ordenación sobre el mismo fichero para extraer ficheros con diferentes conjuntos de datos, analizar la posibilidad de que con un único proceso de ordenación se generen todos los ficheros de salida necesarios utilizando las facilidades `OUTFILE` del proceso de `SORT`.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.009		Evitar descargas masivas de bases de datos ya descargadas.	Revisar el diseño de los procesos batch, asegurando que no se realizan descargas de datos que ya se encuentran disponibles. - Si se realizan más de 3 descargas en la misma cadena, verificar si son necesarias o pueden sustituirse por accesos directos (si se va ha utilizar menos del 30% de los registros descargados) o por una descarga con el mínimo común de datos y un proceso de ordenación con separación en varios ficheros si no se puede utilizar la descarga común como tal.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.010		Tamaño de las descargas de datos	Revisar los procesos de descarga, una vez determinada su necesidad, verificar que los atributos que se descargan son el mínimo común a todos los procesos que los van a utilizar. Mantener la máxima que es mejor una descarga con todo lo necesario que varias descargas con subconjuntos de datos y columnas.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.011		Unificación de procesos de ordenación	Evitar la reordenación de ficheros ya ordenados en diferentes cadenas. Se deben evitar el uso de más de una ordenación por fichero, eliminando las ordenaciones redundantes, se puede utilizar las opciones del proceso de `SORT` si se necesitan diferentes claves de ordenación para el mismo fichero.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.012	QAC-0179	Acceso a datos de otra aplicación	Cada aplicación solo podrá tener accesos mediante SQL o utilidades DB2 a tablas de su entorno funcional (UUAA), de requerir manipular tablas ajenas deberá solicitarlo al responsable de la aplicación proveedora.	Definición de cadena batch		E	BAJO	0
		QA08D-05.013		Uso de utilidades de Bases de datos	Se deben utilizar siempre las utilidades de BMC salvo que el departamento técnico responsable recomiende otra cosa.	Definición de cadena batch		E	BAJO	0
		QA08D-05.014		Procesos de backup de Bases de datos	Se utilizarán los estándares de cada instalación para el resguardo de información de Bases de Datos. El proceso de bakcup de información es diferente en cada instalación debido a los requisitos legales y políticas particulares. Este control lo único que indica es que se contemplen los procesos de backup según los estándares de cada instalación.	Definición de cadena batch		E	BAJO	0
		QA08D-05.015		Procesos de actualización de datos y concurrencia	Cualquier proceso que actualice datos debe contar lógica de `COMMITS` y rearranque según se especifica en los controles de programación y que cumple con la siguientes condiciones: - la frecuencia de commit debe ser un parámetro del programa que pueda ser modificado sin necesidad de cambiar el programa - el proceso cuenta con lógica de rearranque y reposicionamiento en la última uinidad lógica de proceso realizada antes de un abend. - los valores de la frecuencia de commit se darán en función del horario de ejecución del proceso, concurrencia con procesos on-line y necesidades de disponibilidad de las tablas actualizadas, se deberá poder ajustar por los departamentos de producción. - si existen rutinas estándares en la instalación, éstas deberán ser utilizadas obligatoriamente.	Definición de cadena batch		E	ALTO	0
		QA08D-05.016		Procesos de lectura de larga duración	Contemplar si es posible opciones de commit y rearranque para procesos de lectura de larga duración evitando reprocesos ante problemas.	Definición de cadena batch		E	BAJO	0
		QA08D-05.017		Actualización / Consulta masiva de datos desde fichero	Para la actualización o consulta de datos de una tabla a partir de un fichero, deberán ordenarse los datos según el índice cluster de la tabla (si lo hubiese), tanto si la actualización o consulta es mediante utilidades (unload/load) o desde programa, según convenga a la relación entre el volumen del fichero y el de la tabla.	Definición de cadena batch		E	MODERADO	0
		QA08D-05.018		Particionamiento de tablas - volumen	Las tablas cuya estimación en volumen supere los 2GB deben definirse como 'Particionadas'. Para definir el criterio de partición de estas tablas se deben utilizar rangos de claves.	Definición de cadena batch		E	ALTO	0
		QA08D-05.019		Particionamiento de tablas - campos de particionamiento	Los campos por los que se particiona una tabla deben estar incluidos en el Índice Cluster y ser los primeros campos de dicho índice.	Definición de cadena batch		E	ALTO	0
		QA08D-05.020		Indice Cluster	Las tablas que contengan índices deben de tener alguno de ellos definido como CLUSTER.	Definición de cadena batch		E	ALTO	0
		QA08D-05.021		Particionamiento de tablas - indices secundarios	No deben definirse índices secundarios particionados cuyo criterio de particionamiento no coincida con el criterio de particionamiento de la tabla (Índices DPSI).	Definición de cadena batch		E	ALTO	0
QA09D	**Verificación Diseño y Requerimientos de Entornos e Infraestructuras**									
	QA09D-01	**Deben existir entornos de prueba adaptados en capacidad y datos, a las necesidades de las pruebas (conforme a lo especificado en el plan de pruebas):**								
		QA09D-01.001		Definición de las necesidades y requerimientos físicos de los entornos (HW, SW, comunicaciones, etc.)	Verificar que se han realizado las solicitudes necesarias para la configuración de los entornos de acuerdo a las definiciones y requerimientos realizados en el plan de pruebas. Estos requerimientos se deben realizar siguiendo los procedimientos establecidos, y con el margen de tiempo suficiente para permitir la correcta configuración y carga de los entornos	Procs. Explot.		E	BAJO	0

CODIGO CONTROL		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto Metodología	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no
		QA09D-01.002		Definición y solicitud de datos de prueba	Verificar que se han realizado las solicitudes necesarias para la creación de los juegos de datos, para la carga de los entornos de prueba de acuerdo a las definiciones y requerimientos realizados en el plan de pruebas. Estos requerimientos se deben realizar siguiendo los procedimientos establecidos, y con el margen de tiempo suficiente para permitir la correcta carga de los entornos.	Procs. Explot.		E	BAJO	0

CODIGO CONTROL /VERIFICACIÓN	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
										COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
QA10C	**Verificación Codificación de Elementos SW**															
	QA10C-01	**HOST-COBOL: Definición y operaciones con ficheros**														
	QA10C-01.001	QAC-0001	Sentencia READ	No usar en la sentencia `READ` el registro `01` de la `FD` o una variable `WORKING` con distinta longitud. No se puede utilizar nunca la definición de registro de la `FILE SECTION` para leer, hay que utilizar un registro definido en `WORKING` con	Código del componente		E	MODERADO	1		4					
	QA10C-01.002	QAC-0002	Sentencia WRITE	No debe usarse en `WRITE` el registro 01 de la `FD`. Las sentencias de escritura `(WRITE)` de ficheros, se deben codificar utilizando variables definidas en la `WORKING` con idéntica longitud (se admite menor), no trabajar nunca con	Código del componente		E	MODERADO	1		4					
	QA10C-01.003	QAC-0003	Número de ficheros usados en un programa	Se recomienda utilizar menos de 10 ficheros en un programa. Por mantenibilidad, se recomienda no diseñar	Código del componente		R	BAJO	1		2					
	QA10C-01.004	QAC-0004	Uso de REWRITE	No está permitido el uso de `REWRITE` en ficheros secuenciales. No es óptimo utilizar la sentencia `REWRITE` para reemplazar registros existentes en ficheros	Código del componente		E	MODERADO	1		4			4		
	QA10C-01.005	QAC-0005	Uso de FILE STATUS	Definición de fichero sin `FILE STATUS` o `FILE STATUS` mal definido. Es obligatorio controlar cualquier acceso a fichero mediante el `FILE-STATUS`. Debe ser declarado y tratado en `PROCEDURE DIVISION`. El control correcto del	Código del componente		E	BAJO	1		3			3		
	QA10C-01.006	QAC-0006	Cláusula OPTIONAL	No permitida la cláusula `OPTIONAL` en la `FILE CONTROL`. Un fichero tratado por un programa debe existir en el	Código del componente		E	BAJO	1					3		
	QA10C-01.007	QAC-0007	Control de FILE STATUS	Es obligatorio controlar la variable de `FILE STATUS` después de cada sentencia de acceso a ficheros con el objetivo de minimizar posibles abends del programa. No	Código del componente		E	BAJO	1		3			3		
	QA10C-01.008	QAC-0008	Cláusula NOREWIND	No utilizar la cláusula `NOREWIND` en ficheros secuenciales. Si un programa utiliza entrada mediante cinta, y ésta no puede ser cargada a disco previamente, no se debe utilizar	Código del componente		E	BAJO	1					3		
	QA10C-01.009	QAC-0009	Uso de AFTER o BEFORE en WRITE	No debe usarse `AFTER` o `BEFORE` en `WRITE`. En su lugar utilizar el carácter `ASA` escribiendo en ficheros definidos	Código del componente		E	BAJO	1					3		
	QA10C-01.010	QAC-0010	Codificación de OPEN, READ y WRITE.	Se debe codificar `OPEN`, `READ`, `WRITE` una sola vez. La apertura `(OPEN)` y cierre `(CLOSE)` de ficheros se codificarán, siempre que sea posible, una sola vez en el programa, llamándolas vía `PERFORM` cada vez que se utilicen. Lo mismo con sentencias `READ` y `WRITE`. Se	Código del componente		R	BAJO	1		2			2		
	QA10C-01.011	QAC-0011	Cláusula LINAGE	No se permite la cláusula `LINAGE` en la definición de ficheros para especificar el tamaño de una página lógica, ya que no es un uso óptimo. El salto de página debe	Código del componente		E	BAJO						3		
	QA10C-01.012	QAC-0012	Agrupar OPEN y CLOSE	Se deben agrupar la apertura y cierra de todos los ficheros en un mismo `OPEN` y `CLOSE`.	Código del componente		R	BAJO	1		2			2		
	QA10C-01.013	QAC-0013	Repetición de OPEN y CLOSE	Evitar repeticiones de operaciones `Open / Close` para un fichero a lo largo del proceso. Las operaciones de apertura y cierre de ficheros deben hacerse desde el programa principal. Se ha de comprobar si el tratamiento	Código del componente		E	MODERADO	0	4				4		
	QA10C-01.014	QAC-0014	Cierre de todos los ficheros abiertos	Para evitar errores se recomienda cerrar explícitamente todos los ficheros abiertos. Como medida preventiva hacia la aparición de errores, se deben cerrar de forma explícita	Código del componente		E	BAJO			3			3		
	QA10C-02	**HOST-COBOL: Accesos y operaciones con tablas DB2**														
	QA10C-02.001	QAC-0015	Sentencia LOCK TABLE	No está permitido el uso de la sentencia `LOCK TABLE`, ya que el nivel de bloqueo, en este caso, lo decide el programa; es decir, el ámbito y duración del bloqueo quedan definidos por el propio programa, pudiendo	Código del componente		B	ALTO	1			5	5	5		
	QA10C-02.002	QAC-0016	Uso de SELECT con baja discriminación	Se debe evitar el uso de sentencias `SELECT <función>` con baja discriminación en las condiciones de la cláusula `WHERE` que seleccionen un elevado número de filas.	Código del componente		E	MODERADO	1	4		4				
	QA10C-02.003	QAC-0017	Cláusulas UNION y UNION ALL	No se permite el uso de la cláusula `UNION` ni `UNION ALL`. Modificaciones en el modelo de datos pueden evitar	Código del componente		E	MODERADO	1	4		4	5			
	QA10C-02.004	QAC-0018	Uso de vistas con más de una tabla	No están permitidas vistas que en su definición incluyan más de una tabla. Por rendimiento, la utilización de este	Código del componente		B	ALTO	1	5		5	5			
	QA10C-02.005	QAC-0019	Acceso a tablas grandes sin campos del índice informados en el WHERE	Prohibido acceso sobre una tabla grande (nº páginas >= 10.000), en el que no hay campos del índice informados en el `WHERE`. Siempre que se pueda se deben utilizar los campos del índice para el acceso a datos. DB2, si puede utilizar los índices, los utiliza facilitando así el camino de acceso a datos. Dependiendo del tamaño de la tabla, del `WHERE` de la sentencia y de la frecuencia de ejecución del programa, podría ser conveniente la creación de un índice sobre la tabla.	Código del componente		B	ALTO	1	5		5	5			
	QA10C-02.006	QAC-0020	Acceso a tablas grandes sin informar los primeros campos del índice	Prohibido acceso sobre una tabla grande (nº páginas >= 10.000) en el que no se informan los primeros campos de índice. Siempre que las tablas `DB2` tengan índices definidos, las sentencias `SQL` deben acceder por campos que estén incluidos en estos índices. Además, se debe informar el mayor número posible de campos del índice por el que se accede, informando siempre los primeros campos para evitar que el acceso a `DB2` sea costoso.	Código del componente		B	ALTO	1	5		5	5			
	QA10C-02.007	QAC-0021	Acceso a tablas grandes con más de un índice para resolver el acceso	Evitar una sentencia sobre una tabla grande (nº páginas >= 10.000), que usa más de un índice para resolver el acceso. No se deben construir sentencias SQL que accedan a las tablas utilizando más de un índice definido sobre ellas. Dependiendo de cada caso en particular y de lo costoso del acceso, las posibles alternativas serán: - Modificaciones funcionales que permitan cambiar el `WHERE` de la sentencia. - Recuperación / calculo de la información necesaria para codificar en el `WHERE` y utilizar un sólo índice. - Rediseño de los índices existentes. - Diseño de nuevos índices.	Código del componente		E	MODERADO	1	4		4	4			
	QA10C-02.008	QAC-0022	Acceso a tablas medianas sin campos del índice informados en el WHERE	Evitar un acceso sobre una tabla mediana (7 < nº páginas < 10.000), en el que no hay campos del índice informados en el `WHERE`. Siempre que se pueda se deben utilizar los campos del índice para el acceso a datos. `DB2`, si puede utilizar los índices, los utiliza facilitando así el camino de	Código del componente		E	MODERADO		4		4	4			
	QA10C-02.009	QAC-0023	Acceso a tablas medianas sin informar los primeros campos del índice	Evitar un acceso a una tabla mediana (7 < nº páginas < 10.000) en el que no se informan los primeros campos de índice. Siempre que las tablas `DB2` tengan índices definidos, las sentencias `SQL` deben acceder por campos que estén incluidos en estos índices. Además, se debe informar el mayor número posible de campos del índice por el que se accede, informando siempre los primeros campos para evitar que el acceso a `DB2` sea costoso.	Código del componente		E	MODERADO		4		4	4			
	QA10C-02.010	QAC-0024	Acceso a tablas medianas con más de un índice para resolver el acceso	Evitar una sentencia sobre una tabla mediana (7 < nº páginas < 10.000), que usa más de un índice para resolver el acceso. No se deben construir sentencias `SQL` que accedan a las tablas utilizando más de un índice definido sobre ellas. Dependiendo de cada caso en particular y de lo costoso del acceso, las posibles alternativas serán: - Modificaciones funcionales que permitan cambiar el `WHERE` de la sentencia. - Recuperación / calculo de la información necesaria para codificar en el `WHERE` y utilizar un sólo índice. - Rediseño de los índices existentes. - Diseño de nuevos índices.	Código del componente		E	MODERADO		4		4	4			
	QA10C-02.011	QAC-0025	Acceso a tablas pequeñas sin informar los primeros campos del índice, pero se accede a través del índice	Evitar un acceso a una tabla pequeña (nº páginas <= 7) en el que no se informan los primeros campos de índice pero está accediendo a través del índice. En este caso es mejor un acceso secuencial (ningún campo del índice informado o ningún índice en la tabla), o un acceso por los primeros campos del índice. Siempre que las tablas `DB2` tengan índices definidos, las sentencias `SQL` deben acceder por campos que estén incluidos en estos índices. Además, se debe informar el mayor número posible de campos del índice por el que se accede, informando siempre los primeros campos para evitar que el acceso a `DB2` sea costoso. Para tablas pequeñas el acceso sin índice también es correcto.	Código del componente		E	BAJO	1	3		3	3			

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-02.012	QAC-0026	Acceso a tablas pequeñas con más de un índice para resolver el acceso	Evitar una sentencia sobre una tabla pequeña (nº páginas <= 7), que usa más de un índice para resolver el acceso. No se deben construir sentencias SQL que accedan a las tablas utilizando más de un índice definido sobre ellas. Para el caso de tablas pequeñas es mejor el acceso secuencial que el acceso por múltiples índices.	Código del componente		E	BAJO		3		3	3			
		QA10C-02.013	QAC-0027	Acceso a tablas medianas en un bucle sin campos del índice informados en el WHERE	Evitar un acceso en un bucle sobre una tabla mediana (7 < nº páginas < 10.000), en el que no hay campos del índice informados en el WHERE. Siempre que se pueda se deben utilizar los campos del índice para el acceso a datos. DB2, si puede utilizar los índices, los utiliza facilitando así el camino de acceso a datos. Dependiendo del tamaño de la tabla, del WHERE de la sentencia y de la frecuencia de ejecución del programa, podría ser conveniente la creación de un índice sobre la tabla.	Código del componente		E	MODERADO	1	4		4	4			
		QA10C-02.014	QAC-0028	Acceso a tablas medianas en un bucle con más de un índice para resolver el acceso	Evitar una sentencia dentro de un bucle sobre una tabla mediana (nº páginas >= 7), que usa más de un índice para resolver el acceso. No se deben construir sentencias SQL que accedan a las tablas utilizando más de un índice definido sobre ellas. Dependiendo de cada caso en particular y de lo costoso del acceso, las posibles alternativas serán: - Modificaciones funcionales que permitan cambiar el WHERE de la sentencia. - Recuperación / cálculo de la información necesaria para	Código del componente		E	MODERADO	1	4		4	4			
		QA10C-02.015	QAC-0029	Uso de JOIN de dos tablas	No está permitida la utilización de JOIN de dos tablas. No se debe utilizar JOIN de tablas salvo en casos puntuales y estrictamente necesarios que serán validados por el departamento técnico correspondiente. Esta exigencia debe preverse y evitarse durante la fase de diseño del modelo de datos.	Código del componente		E	MODERADO	1	4	4	4	4			
		QA10C-02.016	QAC-0030	Uso de JOIN con accesos costosos	No está permitida la utilización de JOIN que contengan accesos costosos (R0, I0, MX). Si es indispensable el uso de JOIN, no se debe utilizar accesos costosos a alguna de las tablas que intervienen. Deben evitarse especialmente accesos de tipo R0 o SCANS. Estos	Código del componente		B	ALTO	1	5	5	5	5			
		QA10C-02.017	QAC-0031	Uso de sentencias SQL con tablas temporales	No está permitido el uso de sentencias SQL que incluyan tablas temporales definidas mediante la cláusula 'AS' .	Código del componente		B	ALTO	1	5	5	5	5			
		QA10C-02.018	QAC-0032	Uso de funciones sobre variables HOST en la cláusula WHERE	No está permitido el uso de funciones sobre variables HOST en la cláusula WHERE de las sentencias SQL.	Código del componente		E	MODERADO	1	4		4				
		QA10C-02.019	QAC-0033	Uso de LIKE sin operador con valor concreto	Se debe evitar el uso de LIKE '%'y LIKE '_'. Evitar la utilización de operadores LIKE donde la expresión a evaluar no empiece por un valor concreto, como por ejemplo: 1. LIKE '%dato' 2. LIKE '_dato'.	Código del componente		E	MODERADO	1	4		4				
		QA10C-02.020	QAC-0034	Uso de SUBSELECT	No está permitida la utilización de SUBSELECT salvo en casos puntuales y estrictamente necesarios que serán validados por el departamenteo técnico correspondiente.	Código del componente		E	MODERADO	1	4		4				
		QA10C-02.021	QAC-0035	Unico acceso a tablas para obtener datos	Los datos necesitados de una tabla deben obtenerse mediante un único acceso a ésta. Evitar reiteradas consultas sobre una misma tabla para recuperar diferentes atributos de un registro, los datos necesitados de una tabla	Código del componente		E	MODERADO	0	4		4				
		QA10C-02.022	QAC-0036	Uso de tablas de baja volumetría	Las tablas DB2 de bajo volumen y muy accedidas deben copiarse en WORKING al principio de la ejecución del programa si éste el batch. Para el caso de online, sólo en caso de necesitarse el 90% de las filas de la tabla en cada ejecución, en caso contrario, acceder sólo a las filas necesarias. Para CCR y en caso de CICS, se podrá cargar la tabla en una Temporary Storage si se comparte con más transacciones o ejecuciones de una misma transacción.	Código del componente		E	MODERADO		2		2				
		QA10C-02.023	QAC-0037	Uso de funciones sobre columnas en la cláusula WHERE	No está permitido el uso de funciones sobre columnas en la cláusula WHERE de las sentencias SQL. Estos tipos de predicado no son indexables y por tanto si se aplican a	Código del componente		E	MODERADO		4		4				
		QA10C-02.024	QAC-0038	Ejecución de SORT en query DB2	Se debe evitar el proceso de SORT por parte de DB2. Si existe una necesidad funcional, deben buscarse soluciones de diseño identificando el origen de la necesidad del SORT con el fin de evitarlo. Posibles soluciones al SORT DB2, dependiendo de cada caso en particular y de la gravedad del SORT podrían ser : - Pasos previos de SORT en JCL - Cambios en el orden de tablas del modelo de datos - Diseño de índices alternativos.	Código del componente		R	BAJO		2		2				
		QA10C-02.025	QAC-0039	Columnas a incluir en la sentencia FETCH	La sentencia FETCH debe incluir las mismas columnas y en el mismo orden que aparecen en la declaración del cursor	Código del componente		E	MODERADO			4					
		QA10C-02.026	QAC-0040	Condición >= en la cláusula WHERE	Se debe cambiar en la cláusula WHERE la condición C>DATO OR C=DATO por C>=DATO. Aunque las dos sentencias recuperan los mismos datos, DB2 realiza un proceso de búsqueda más sencillo en el segundo caso que	Código del componente		E	MODERADO		4		4				
		QA10C-02.027	QAC-0041	Uso de expresiones NOT> y NOT<	No usar las expresiones: NOT> y NOT<, en su lugar utilizar: <= y >= respectivamente. Evitar todos aquellos casos en los que DB2 no utiliza índices (predicados no indexables) como: Predicados con '<>' o 'Not =', o con 'Not >' y 'Not <'	Código del componente		E	MODERADO		4		4				
		QA10C-02.028	QAC-0042	Uso de SELECT *	No está permitido el uso de SELECT *. Su uso no independiza al módulo de posibles alteraciones de la tabla como añadir columnas, y por tanto no se justifica frente al coste de recuperación de todo el registro.	Código del componente		B	ALTO					5			
		QA10C-02.029	QAC-0043	Uso de SELECT COUNT(*)	No se debe usar la sentencia SELECT COUNT(*) si sólo se pretende verificar la existencia o no de filas, en su lugar codificar una sentecia Select y controlar los códigos de retorno 0, 100 y -811 para conocer si existe una fila, ninguna o más de una. Si se precisase conocer el número	Código del componente		E	MODERADO		4		4				
		QA10C-02.030	QAC-0044	Actualizaciones masivas mediante SQL	No está permitido realizar actualizaciones masivas mediante una sentencia SQL. Dependiendo de la discriminación del WHERE pueden provocar un número elevado de bloqueos, perjudicando la concurrencia de las aplicaciones. Debe ser revisado por el departamento técnico correspondiente.	Código del componente		E	MODERADO		4	4	4	4	4		
		QA10C-02.031	QAC-0045	Uso de cursor FOR UPDATE	Evitar cursor declarado FOR UPDATE sin DELETE o UPDATE WHERE CURRENT posterior. Los cursores declarados con la cláusula FOR UPDATE deben tener una sentencia UPDATE/DELETE WHERE CURRENT OF CURSOR asociada en el programa Se reduce el impacto por el nivel de bloqueo que estas sentencias establecen.	Código del componente		E	MODERADO		4	4	4	4			
		QA10C-02.032	QAC-0046	Uso de HAVING	Evitar el uso de HAVING. Por rendimiento, no es recomendable la utilización de la cláusula de condición HAVING para GROUP BY. Se deben realizar diseños de	Código del componente		E	MODERADO		4		4				
		QA10C-02.033	QAC-0047	Recuperación de columnas en sentencias FETCH o SELECT	En sentencias FETCH o SELECT recuperar sólo las columnas que necesita el programa/proceso.	Código del componente		E	BAJO		3		3				
		QA10C-02.034	QAC-0048	Acceso a tablas dentro de un bucle sin informar los primeros campos de índice.	Evitar un acceso dentro de un bucle a una tabla (nº páginas >= 7) en el que no se informan los primeros campos de índice. Siempre que las tablas DB2 tengan índices definidos, las sentencias SQL deben acceder por campos que estén incluidos en estos índices. Además, se debe informar el mayor número posible de campos del índice por el que se accede, informando siempre los primeros campos para evitar que el acceso a DB2 sea costoso.	Código del componente		E	MODERADO	1	4		4				

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-02.035	QAC-0049	Uso de INITIALIZE para DCLGEN	No utilizar `INITIALIZE` para `DCLGEN a nivel 01`. Se debe utilizar sólo cuando la sentencia a ejecutar sea INSERT y no se conozca el valor de todas las columnas.	Código del componente		R	BAJO		2		2				
		QA10C-02.036	QAC-0050	Uso de indicador de nulos en SELECT MAX, MIN, AVG y SUM	Se debe usar el indicador de nulos en `SELECT MAX, MIN, AVG` y `SUM`. Las sentencias `SQL` del tipo `SELECT MAX(),MIN(),AVG()` y `SUM()` deben contemplar el indicador de nulos en una variable `HOST`. No se puede	Código del componente		E	BAJO		3	3	3				
		QA10C-02.037	QAC-0051	Uso de indicador de nulos en SELECT COUNT	No se debe utilizar el indicador de nulos en `SELECT COUNT`, solamente debe controlarse el `SQLCODE`.	Código del componente		E	BAJO			3					
		QA10C-02.038	QAC-0052	Uso de SELECT CURRENT TIMESTAMP	Evitar el uso de `SELECT CURRENT TIMESTAMP`. La fecha de proceso debería ser un atributo más del modelo de datos de la aplicación. Si la fecha del sistema se utiliza como fecha de proceso, hay que tener en cuenta los problemas que se podrían ocasionar si se tuviera que	Código del componente		R	BAJO		2	2	2		2		
		QA10C-02.039	QAC-0053	Cierre de cursores.	Cerrar los cursores durante la ejecución del programa. Es obligatorio cerrar de forma explícita todos los cursores abiertos durante un proceso. Sólo en caso de error serán cerrados por terminación anormal.	Código del componente		E	BAJO			3		3	3		
		QA10C-02.040	QAC-0054	Uso de cursores en recuperaciones de una sola fila	No utilizar cursores si sólo se recuperará una fila, en su lugar utilizar `SELECT`. Los cursores deben ser utilizados sólo cuando el `WHERE` de la sentencia devuelva más de una fila y además todas las filas devueltas necesiten ser tratadas por el programa. En caso contrario, se deben utilizar sentencias `SELECT`, controlando el `SQLCODE -811` en caso de que la sentencia devuelva más de una fila	Código del componente		E	BAJO			3		3				
		QA10C-02.041	QAC-0055	Existencia de una fila	Si se necesita saber si existe o no una fila, buscarla seleccionando un solo campo y que sea de índice. Si es necesario verificar sólo la existencia o no de una fila y, no es necesario conocer ningún valor de la misma, seleccionar sólo una columna del índice (aunque se condicione por igual). Esto provoca que `DB2` sólo acceda al fichero de índices no teniendo necesidad de acceder al fichero de datos.	Código del componente		E	BAJO			3		3				
		QA10C-02.043	QAC-0056	Sentencia INSERT	La sentencia `INSERT` debe codificarse completa y con las variables `HOST` en el mismo orden que los valores que toman. Las variables `HOST` deben estar en el mismo orden en que se han especificado los atributos. Esto no significa que se especifiquen todas las columnas de una tabla, sino aquéllas que se necesiten, ya que el resto tomarán los valores por defecto (dependiendo de la definición de la columna, si admite o no valores nulos): - Espacios para atributos alfanuméricos. - Ceros para atributos numéricos. - Fecha y/o Hora actual en campos de fecha y hora.	Código del componente		E	BAJO		3	3	3				
		QA10C-02.044	QAC-0057	Uso de cursor FOR UPDATE	Se debe usar `CURSOR FOR UPDATE` en lugar de realizar `SELECT` y `UPDATE/DELETE`, en actualizaciones que requieren el conocimiento del contenido de las filas antes de la sentencia de actualización. Si es necesario conocer previamente el valor de la fila a modificar ó borrar: utilizar un cursor declarado `FOR UPDATE` en lugar de realizar primero una sentencia `SELECT` y luego la sentencia de actualización. La razón es la forma de adquirir el bloqueo, ya que de la forma recomendada se evitan posibles `DEADLOCKS` y `TIMEOUTS`. Esto es completamete obligatorio en el caso de actualización de tablas numeradoras.	Código del componente		E	MODERADO		4	4	4	4	4		
		QA10C-02.045	QAC-0058	Columnas en sentencias UPDATE	En las sentencias `UPDATE` no se deben poner columnas cuyo valor no ha sido modificado. No deben incluirse en la cláusula `SET` de un `UPDATE` columnas cuyo valor no hay	Código del componente		E	BAJO		3	3	3				
		QA10C-02.046	QAC-0059	Uso de SELECT/FETCH para verificar existencia de filas	No deben usarse `SELECT/FETCH` para verificar la existencia de una fila para su posterior lectura o actualización. Se deben controlar directamente los valores de retorno del `SQLCODE` en la sentencia de actualización para comprobar si la fila existe.	Código del componente		E	BAJO		3		3				
		QA10C-02.047	QAC-0060	Utilización de INCLUDE de tablas no necesarias	No realizar `INCLUDE` de tablas no utilizadas en el programa. Hacen el `DBRM` más grande de lo necesario.	Código del componente		R	BAJO					2	2		
		QA10C-02.048	QAC-0061	Duplicidad de accesos SQL	No duplicar accesos `SQL`. Las sentencias SQL idénticas deben codificarse en un párrafo aparte y llamarlas mediante `PERFORM`, en lugar de escribirlas varias veces en	Código del componente		R	BAJO					2	2		
		QA10C-02.049	QAC-0062	Codificación de sentencias SQL no utilizadas	Evitar la codificación de sentencias `SQL` que no se ejecutan. No se pueden codificar sentencias `SQL` que no son ejecutadas durante el proceso. Esto hace que tanto el `DBRM` como el `PLAN` sean más grandes de lo necesario, aumentando el tiempo de carga y el espacio en el `EDM`	Código del componente		R	BAJO					2	2		
		QA10C-02.050	QAC-0063	Columnas actualizadas en sentencia UPDATE WHERE CURRENT	Las columnas actualizadas en una sentencia `UPDATE WHERE CURRENT OF CURSOR` deben ser las mismas que las declaradas en la cláusula `FOR UPDATE`.	Código del componente		R	BAJO						2		
		QA10C-02.051	QAC-0064	Columnas en cursores para borrar filas	En cursores para borrar filas con `DELETE WHERE CURRENT OF`, en la cláusula `FOR UPDATE` debe tener una	Código del componente		R	BAJO						1		
		QA10C-02.052	QAC-0065	Cursor FOR UPDATE sin informar todas las columnas a modificar	Evitar un cursor `FOR UPDATE` que no informa en el `SET` del `WHERE CURRENT OF` todas las columnas que van a ser modificadas.	Código del componente		R	BAJO						2		
		QA10C-02.053	QAC-0066	Recuperación de campos conocidos en el WHERE	No recuperar campos que se condicionan por igual en el `WHERE`, el valor del campo es conocido y por lo tanto no es necesario aumentar el bloque de E/S. Sólo en caso de	Código del componente		R	BAJO		2		2				
		QA10C-02.054	QAC-0067	Declaración de cursores	Todos los cursores en `PROCEDURE` deben estar declarados. Los `CURSORES` referenciados a través de sentencias `SQL OPEN`, `CLOSE` y `FETCH` deben estar declarados en el programa en la zona declarativa `Working`.	Código del componente		E	BAJO						3		
		QA10C-02.055	QAC-0068	Uso de COMMIT	El uso de COMMIT debe cumplir: - La frecuencia de `COMMIT` en el programa dependerá de factores como la hora de ejecución, el tipo de tablas a actualizar, la concurrencia con otras aplicaciones, el tipo de proceso, el bloqueo de recursos etc. La sentencia `COMMIT` es costosa por lo que la elección correcta de la frecuencia de `COMMIT` en el programa es importante. - El valor de la frecuencia de `COMMIT` no debe ser codificado en el programa sino que será una variable de entrada bien en un fichero, en un parámetro de jcl o en una tabla. Si existen rutinas de tratamiento general de `COMMIT` y rearranque deberán ser utilizadas. - Deberán contar con lógica de reinicios. - La frecuencia de COMMITS deberá ser tal que se evite provocar el escalamiento de locks (lock scalation). - Cuando la actualización sea sobre tablas línea, se deberá indicar una frecuencia de COMMIT muy alta (inclusive actualización por registro) para evitar conflictos con los servicios online. Considerar que entre más alta la frecuencia de COMMITS mayor será la duración del proceso.	Código del componente		R	BAJO		1	1	1	1	1		
		QA10C-02.056	QAC-0069	Uso de cursores en Online	En online los cursores no deben recuperar un número elevado de filas, no deben provocar `LIST PREFECH`. En online, los cursores que tienen como finalidad construir listas, no deberían recuperar muchas filas. Si debido a las condiciones de las claúsulas `WHERE`, los cursores recuperaran muchas filas, sería necesario aplicar técnicas de lista-reposicionamiento para realizar recuperaciones parciales de filas y optimizar los accesos.	Código del componente		E	BAJO		3		3				

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-02.057	QAC-0070	Uso de GROUP BY en acceso SQL	Evitar el uso de `GROUP BY` en un acceso `SQL`. Por rendimiento, evitar el uso de la cláusulas `GROUP BY` en sentencias que provoquen `SORT DB2`.	Código del componente		R	BAJO		1		1				
		QA10C-02.058	QAC-0071	Codificación de ROLLBACK	Codificar `ROLLBACK` para delimitar las unidad lógicas de trabajo, principalmente en los programas batch. Nunca debería utilizarse en rutinas que pueden ser utilizadas por más de un programa	Código del componente		R	BAJO			1		1	1		
		QA10C-02.059	QAC-0072	Uso de DISTINCT	Evitar el uso de `DISTINCT`. Por rendimiento no es recomendable la utilización de la cláusula `DISTINCT`. Se deben realizar diseños de aplicación que eviten la utilización de este tipo de cláusulas en fase de programación.	Código del componente		E	BAJO		3		3				
		QA10C-02.060	QAC-0073	Uso de INCLUDE en DCLGEN de las tablas utilizadas	Se recomienda el uso de `INCLUDE` de la `DCLGEN` de las tablas `DB2` utilizadas. En cualquier sentencia `SQL`, las variables utilizadas deben ser las definidas en la `DCLGEN` ó, si no se puede, las definidas en `WORKING` con el mismo formato que el generado para la columna afectada en la `DCLGEN`. Cuando las variables utilizadas no tienen el mismo formato, `DB2` se ve forzado a realizar conversiones innecesarias. Si además la columna forma parte de un índice, puede que `DB2` no utilice dicho índice en el acceso; este es el caso de comparaciones numéricas de distinta precisión, así como alfanuméricas donde la variable `HOST` es más larga que el atributo a comparar.	Código del componente		R	BAJO		1	1	1		1		
		QA10C-02.061	QAC-0074	Uso de expresiones aritméticas en cláusula WHERE	Evitar el uso de expresiones aritméticas en la parte derecha de la cláusula WHERE en sentencias SQL; en su lugar, realizar la operación aritmética, mover el resultado a una variable HOST y comparar con dicha variable. Esto se	Código del componente		E	MODERADO	0	4		4				
		QA10C-02.062	QAC-0075	Uso de SQL dinámico	Está completamente prohibido el uso de `SQL dinámico` en	Código del componente		B	ALTO		5	5	5	5	5		
		QA10C-02.063	QAC-0076	Actualización de columnas incluidas en el índice de partición	Para versiones de `DB2` anteriores a `V8`, no se deben actualizar las columnas que pertenecen al índice de partición en tablas particionadas.	Código del componente		E	MODERADO	0							
		QA10C-02.064	QAC-0077	Actualización de campos incluidos en la clave primaria	No está permitido la actualización de campos que formen parte de la clave primaria de una tabla. En su lugar se debe	Código del componente		E	BAJO	0							
		QA10C-02.065	QAC-0078	Uso obligatorio de PACKAGES	Todas las aplicaciones deben hacer uso de `PACKAGES` en lugar de la opción `Plan/DBRM`.	Código del componente		B	ALTO	0				5	5		
		QA10C-02.066	QAC-0079	Uso de opción CURSOR WITH HOLD	Se debe utilizar la opción `CURSOR WITH HOLD` en procesos batch que hagan uso de de proceso de commit y rearranque (obligatorio cuando se actualiza `DB2`), para evitar tener que reposicionarse en los datos después de la	Código del componente		E	BAJO	0	3		3				
		QA10C-02.072	QAC-0080	Uso de opción FOR FETCH n ROW en programas online para paginación o reposicionamiento del cursor	Utilizar la opción `FOR FETCH n ROW` para programas online en donde sea necesaria la paginación o reposicionamiento del cursor, donde `n` es el número de filas a recuperar y mostrar en pantalla o enviar al terminal. Esta	Código del componente		E	BAJO	0	3		3				
		QA10C-02.074	QAC-0081	Evitar accesos, inserciones y/o actualizaciones directas en tablas de otras aplicaciones	Se deben evitar accesos, inserciones y/o actualizaciones directas sobre tablas críticas que no sean de la propia aplicación, en su lugar utilizar las rutinas propias de cada aplicación para realizar la función deseada.	Código del componente	SÓLO CCR	B	ALTO	0				5	5		
		QA10C-02.075	QAC-0082	Uso de utilidades para depuraciones y reorganizaciones	Las depuraciones y reorganizaciones deben realizarse mediante utilidades/utilierías salvo en el caso de que se requiera validación de información de otra tabla o la tabla requiera estar 100% disponible.	Código del componente		E	BAJO	0	3		3		3		
		QA10C-02.076	QAC-0083	Uso de la cláusula CASE	No está permitido el uso de la claúsula `CASE` en sentencias `SQL`.	Código del componente		B	ALTO	1	5		5				
		QA10C-02.077	QAC-0084	Uso de CURRENT DATE	Evitar el uso de `CURRENT DATE`, para evitar desfases entre la fecha real y la de proceso. No se debe utilizar la fecha del sistema como fecha de proceso, hay que tener en cuenta los problemas que se podrían ocasionar si se tuviera que repetir un proceso con fecha del día anterior.	Código del componente		R	BAJO	1	2		2		2		
		QA10C-02.078	QAC-0085	Uso de CURRENT TIME	Evitar el uso de `CURRENT TIME`, para evitar desfases entre la hora real y la de proceso. Si la hora del sistema se utiliza como hora de proceso puede ser peligroso por los posibles desfases.	Código del componente		R	BAJO	1	2		2		2		
		QA10C-02.079	QAC-0086	Uso de sentencias que desencadenen SORT en DB2 con elevado numero de registros	Se detectarán todos las sentecias que desencaden un proceso costoso de `SORT` por parte del `DB2` debido al elevado número de filas seleccionas. Si existe una necesidad funcional, deben buscarse soluciones de diseño identificando el origen y la necsidad del SORT con el fin de evitarlo. Posibles soluciones al `SORT DB2` dependiendo de cada caso en particular y de la gravedad del `SORT` podría ser: - Pasos previos de `SORT` en `JCL` - Cambios en el orden de tablas del modelo de datos - Diseño de índices alternativos - Modificación de índices existentes - Ajustes en la sentencia, `WHERE, GROUP BY, DISTINCT`, etc - Etc. dependiendo de cada caso	Código del componente		E	MODERADO	0	4		4				
		QA10C-02.080	QAC-0087	Accesos mínimos a tablas	Para recuperar un dato, se debe realizar el menor número	Código del componente		E	MODERADO	0	4		4				
		QA10C-02.081	QAC-0088	No permitido tamaños de ROWSET > 200	En sentencias que usen la opción MULTIROW, el tamaño del ROWSET nunca debe ser superior a 200. Por encima der de 200, la mejora en el rendimiento de las sentencias se ve considerablemente mermado.	Código del componente		E	MODERADO	1	4		4				
		QA10C-02.082	QAC-0089	N en FOR N ROWS <= OCCURS tabla ROWSET	En sentencias que usen la opción MULTIROW, el valor del parámetro "N" de la opción FOR "N" ROWS ha de ser <= al OCCURS de la tabla definida para recibir el ROWSET. Se debe garantizar que todas las filas seleccionadas mediante la opción MULTIROW, tienen el espacio necesario definido en la tabla que recibirá el ROWSET.	Código del componente		B	ALTO	1	5		5				
		QA10C-02.083	QAC-0090	MULTIROW concordante CURSOR y FETCH	Si un cursor se define WHIT ROWSET POSITIONING, el FETCH correspondiente a ese cursor, debe ser definido con la cláusula NETX ROWSET y viceversa. La definición del cursor y el tratamiento de las filas seleccionadas por el mismo, ha de ser concordante.	Código del componente		E	MODERADO	1	4		4				
		QA10C-02.084	QAC-0091	FETCH MULTIROW control SQLCODE=100	Si una sentencia está definida con la cláusula NEXT ROWSET, se debe realizar un control y tratamiento adecuados del SQLCODE +100. Se ha de comprobar si el ROWSET devuelto junto con el SQLCODE +100 ha devuelto alguna fila y realizar el	Código del componente		B	ALTO	1	5		5				
		QA10C-02.085	QAC-0092	Cursores MULTIROW usar todas las columnas	Todas las columnas declaradas en la SELECT de un cursor definido WITH ROWSET POSITIONING deben ser utilizadas posteriormente en el programa. Se debe garantizar que todas la filas seleccionadas son necesarias puesto que son usadas posteriormente en el	Código del componente		E	BAJO	1	3		3				
		QA10C-02.086	QAC-0093	INSERT MULTIROW prohibido NOT ATOMIC	En sentencias INSERT definidas con opción MULTIROW, no se permite usar la cláusula NOT ATOMIC CONTINUE ON SQLEXCEPCION. Se debe garantizar que si el INSERT termina con error, no se ha insertado ninguna fila y todo el ROWSET ha sido rechazado. No se permiten inserciones parciales por la dificultad de control.	Código del componente		B	ALTO	1	5		5				
		QA10C-02.087	QAC-0094	El COMMIT se fija según número de filas afectado	En sentencias que usen la opción MULTIROW, la frecuencia del COMMIT, se debe definir en función del número de filas afectado por cada sentencia, no del número de sentencias ejecutado. Cada sentencia afecta a un número de filas que viene determinado por el tamaño del ROWSET, luego se tendrá en cuenta este dato para calcular la frecuencia del COMMIT, no solo el número de sentencias ejecutado.	Código del componente		E	MODERADO	1	4		4				

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-02.088	QAC-0095	: No permitido FOR ROW "N" OF ROWSET	En sentencias UPDATE ó DELETE definidas con opción MULTIROW, no está permitido usar la cláusula FOR ROW "N" para actualizar ó borrar solo la N'ésima fila del ROWSET. Se considera una práctica de riesgo y de difícil control realizar una sentencia que solo afecte a una fila de un ROWSET que contiene varias.	Código del componente		B	ALTO	1	5		5				
		QA10C-02.089	QAC-0096	WHERE CURRENT OF concordante con FETCH	En sentencias UPDATE o DELETE que utilicen la opción MULTIROW con la cláusula WHERE CURRENT OF cursorXX, la definición del CURSOR y de la FETCH deben concordar en cuanto a las opciones MULTIROW. Afecta a las sentencias UPDATE y DELETE que utilicen la opción MULTIROW con la cláusula WHERE CURRENT OF cursorX. Comprobar que el cursorX está definido como MULTIROW si la FETCH correspondiente está definida como MULTIROW	Código del componente		B	ALTO	1	5		5				
		QA10C-02.090	QAC-0097	Controlar los SQLCODE -359 y -845	En el tratamiento de OBJETOS SEQUENCE mediante cláusulas NEXT VALUE y PREVIOUS VALUE, se debe controlar respectivamente, el SQLCODE -359 y -845. El control adecuado de estos códigos, garantiza la fiabilidad del valor de la secuencia obtenida y permite tomar las acciones oportunas cuando se producen.	Código del componente		B	ALTO	1	5		5				
		QA10C-02.091	QAC-0098	Uso de cursores SCROLLABLES prohibido	El uso de cursores SCROLLABLES está estrictamente prohibido en la instalación. Se considera una práctica de riesgo el uso de este tipo de cursores y no se permite su uso.	Código del componente		B	ALTO	1	5		5				
		QA10C-02.092	QAC-0099	Uso obligatorio de COPY para variables de uso EXTERNAL	Es obligatorio definir mediante el uso de una COPY cualquier variable de uso EXTERNAL. El control adecuado de este tipo de variables y su uso, requiere el uso de COPY para evitar duplicidades y uso indebido de las mismas.	Código del componente		B	ALTO	1		5					
		QA10C-02.093	QAC-0100	Detectada llamada a rutina obsoleta	Se debe evitar el uso de llamar a rutinas obsoletas por el riesto que conlleva. El listado de funciones a evitar es el siguiente: AEATM990, AEATX010, AEGTX000, CEE3ABD, CFECHE, **CIC1CCM,** FINABEND, FIPHMCAN, FSIFM503, GDFSM100, GSFED000, ILBOABN0, IMPRIMIR, KAELMR11, KGICR089, KGICR107, KGICR202, KGICR251, KGICR371, KGICR373, KGICR374, KGICR375, KGICR808, KGICR808, KGICR810, KGICR812, KGICR813, KGICR817, KGICR821, KGICR823, KGICR824, KGICR827, KGICR840, KGICR842, KGICRE07, KGICRE08.	Código del componente		B	CPD Europa : MODERADO CCR Méjico: ALTO					4		
		QA10C-02.094	QAC-0101	Validación de existencia de registro mediante SELECT	Cuando se lee una tabla a través de una instrucción SELECT directa y solo se pretende utilizar el primer registro devuelto o validar la existencia de una clave (el SQLCODE -811 se da por bueno), se debe utilizar la cláusula FETCH FIRST 1 ROW ONLY, evitando de esta forma el SQLCODE -811 y optimizando el acceso a datos.	Código del componente		E	MODERADO	0	4		4				
	QA10C-03	**HOST-COBOL: Normas de codificación para la definición variables, tablas internas, etc.**															
		QA10C-03.001	QAC-0102	Se ha detectado la definición de una tabla LINKAGE o WORKING con demasiados elementos o con tamaño total grande.	No definir tablas en LINKAGE ni en WORKING con más de 5.000 elementos. Por rendimiento, no utilizar en los programas tablas que tengan definido un número elevado de ocurrencias. No se deben definir tablas que ocupen mucha zona de memoria. Los datos contenidos en las tablas LINKAGE y WORKING grandes deberían almacenarse en otro tipo de dispositivos, como ficheros o tablas DB2, a los que se puede acceder de una forma más eficiente de cara al consumo de recursos.	Código del componente		E	MODERADO		4		4				
		QA10C-03.003	QAC-0104	Cláusula BLOCK CONTAINS	La cláusula `BLOCK CONTAINS` debe estar definida como `BLOCK CONTAINS 0 RECORDS` para dejar al `JCL` la gestión del bloque óptimo según el tipo de dispositivo.	Código del componente		B	ALTO						5		
		QA10C-03.004	QAC-0105	Uso de COMP-2	No está permitido el uso de COMP-2 por su elevado consumo, sólo se admitirán en casos excepcionales.	Código del componente		E	BAJO		3						
		QA10C-03.005	QAC-0106	Uso de FILLER a nivel 01	Se recomienda no usar `FILLER` a nivel `01`.	Código del componente		R	BAJO		2						
		QA10C-03.006	QAC-0107	Índice de tablas WORKING	El índice de una tabla `WORKING` debe ser `COMP` por razones de rendimiento.	Código del componente		E	MODERADO		4						
		QA10C-03.007	QAC-0108	Uso de cláusula INITIALIZE	Se recomienda el uso de INITIALIZE para variables y tablas internas de nivel 01, para hacer más eficiente el uso de CPU de esta instrucción.	Código del componente		E	BAJO		3						
		QA10C-03.008	QAC-0109	Uso de SQLCA	No incluir `SQLCA` si no hay accesos `DB2` en el programa.	Código del componente		E	BAJO						3		
		QA10C-03.009	QAC-0110	Uso de DCLGEN a nivel 01	No está permitido el uso de la `DCLGEN` a nivel `01`.	Código del componente		E	BAJO		3	3					
		QA10C-03.010	QAC-0111	Campo AUTHOR	El campo `AUTHOR` debe ir informado.	Código del componente		R	BAJO						2		
		QA10C-03.011	QAC-0112	Campo PROGRAM-ID	El `PROGRAM-ID` es obligatorio y debe ser igual al miembro que lo almacena.	Código del componente		B	ALTO						5		
		QA10C-03.012	QAC-0113	Definición de tablas.	Las tablas deben definirse al final de la WORKING y antes de la declaración de cursores con el fin de evitar que cualquier desbordamiento provoque errores en otras	Código del componente		E	BAJO			3			3		
		QA10C-03.013	QAC-0114	Definición de cursores.	Los cursores deben definirse al final de la `WORKING` y nunca en `PROCEDURE DIVISION`.	Código del componente		E	BAJO			3			3		
		QA10C-03.014	QAC-0115	Definición de indicador de nulos en sentencias SQL.	Se debe definir correctamente el indicador de nulos para acceso en sentencias `SQL`, debe ser: `PIC S9(4) COMP.`	Código del componente		E	BAJO		3				3		
	QA10C-04	**HOST-COBOL: Normas de codificación para el uso de sentencias COBOL**															
		QA10C-04.001	QAC-0116	Uso de SORT interno	Está prohibido el uso de `SORT` interno, para ello utilizar la	Código del componente		B	ALTO		5		5		5		
		QA10C-04.002	QAC-0117	Sentencia GO TO	La utilización de la sentencia `GO TO` está prohibida.	Código del componente		E	MODERADO						4		
		QA10C-04.003	QAC-0118	Sentencia ALTER	No está permitida la sentencia Cobol `ALTER`.	Código del componente		B	ALTO						5		
		QA10C-04.004	QAC-0119	Uso de ON SIZE ERROR	Se debe eliminar el uso de `ON SIZE ERROR`, usando técnicas como comprobación de que el denominador no está a cero antes de ejecutar la instrucción.	Código del componente		E	MODERADO		4						
		QA10C-04.005	QAC-0120	Niveles de anidamiento en sentencia IF	En la instrucción `IF` no se permiten más de 10 niveles de anidación por razones de mantenibilidad y rendimiento.	Código del componente		R	BAJO						2		
		QA10C-04.006	QAC-0121	Búsquedas en tablas Working de más de 50 elementos	Para búsquedas en tablas `WORKING` de más de 50 elementos usar `SEARCH ALL`. Los elementos de la tabla deben estar ordenados conforme la clave de búsqueda. CCR consideraque está duplicado con el 24, por lo que se sugiere eliminar éste y dejar el 24.	Código del componente		E	MODERADO		4		4				
		QA10C-04.007	QAC-0122	Uso de cláusula AT END en sentencia SEARCH	En `SEARCH` debe controlarse la cláusula `AT END`.	Código del componente		E	BAJO						3		
		QA10C-04.008	QAC-0123	Sentencia CANCEL	No está permitida la instrucción `CANCEL`. Su utilización es debida normalmente a la aparición de un número elevado de niveles de llamadas entre programas como consecuencia de una excesiva modularización. La reducción del número de llamadas a programas dentro de la ejecución de un programa elimina la necesidad de utilizar esta sentencia así como las opciones de compilación/linkedición `AMODE (31)`.	Código del componente		E	MODERADO		4		4		4		
		QA10C-04.009	QAC-0124	Uso de WHEN OTHER en cláusula EVALUATE	Es obligatorio el control de `WHEN OTHER` en la cláusula `EVALUATE` para completar todas las posibles condiciones.	Código del componente		E	BAJO			3			3		
		QA10C-04.010	QAC-0125	Uso de PERFORM VARYING	El uso de `PERFORM VARYING` sólo es aceptado para tratar tablas `WORKING`. Es debido a rendimiento y fiabilidad. Debe utilizarse para bucles que procesan tablas de datos secuencialmente. La variable de control debe ser índice de la tabla tratada.	Código del componente		E	BAJO		3	3					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-04.011	QAC-0126	Uso de PERFORM N TIMES	El uso de PERFORM N TIMES sólo es aceptado para tratar tablas en memoria. Es debido a rendimiento y fiabilidad. Debe utilizarse para bucles que procesan tablas de datos secuencialmente. El valor de N debe ser menor o	Código del componente		E	BAJO		3	3					
		QA10C-04.012	QAC-0127	Búsquedas en tablas Working de menos de 50 elementos	Para búsquedas en tablas `WORKING` de menos de 50 elementos usar `SEARCH`. Esta instrucción es más rápida que `SEARCH ALL` para tablas pequeñas.	Código del componente		E	BAJO		3		3				
		QA10C-04.013	QAC-0128	Uso de DISPLAY UPON CONSOLE	No está permitido el uso de `DISPLAY [variable/literal] UPON CONSOLE`. Está completamente prohibido enviar datos a consola, esto afecta a la explotabilidad de la instalación y de la aplicación.	Código del componente		E	BAJO		3				3		
		QA10C-04.014	QAC-0129	Uso de ACCEPT FROM CONSOLE	No está permitido el uso de `ACCEPT [variable/literal] FROM CONSOLE`. Está completamente prohibido recibir datos de consola, esto afecta a la explotabilidad de la instalación y de la aplicación.	Código del componente		E	BAJO		3				3		
		QA10C-04.015	QAC-0130	Uso de INCLUDE en PROCEDURE DIVISION	No está permitido el uso de `INCLUDE` en la `PROCEDURE DIVISION`.	Código del componente		E	BAJO			3			3		
		QA10C-04.016	QAC-0131	Uso de COPY en PROCEDURE DIVISION	No está permitido el uso de `COPY` en `PROCEDURE DIVISION`.	Código del componente		E	BAJO			3			3		
		QA10C-04.017	QAC-0132	Definición de campos que intervienen en operaciones aritméticas	La PICTURE de los campos que intervienen en una operación aritmética debe ser la misma (todos COMP o todos COMP-3) y tener la misma longitud y signo. Cualquier otro formato requiere conversión interna.	Código del componente		E	BAJO		3		3				
		QA10C-04.018	QAC-0133	Divisiones entre 0	Se deben evitar las divisiones entre `0`. Revisión de las operaciones que multiplican o dividen `0` ó `1`. Estas operaciones consumen mucho tiempo de CPU, incluso cuando los resultados no son de interés.	Código del componente		E	BAJO		3		3				
		QA10C-04.019	QAC-0134	Uso de IF NUMERIC e IF ALPHABETIC	Se debe evitar el uso de `IF NUMERIC` e `IF ALPHABETIC`. Estas instrucciones utilizan la instrucción máquina `TRT` que es muy potente y a la vez muy costosa, por lo que se recomienda que se utilice los menos posible. Para evitar la utilización de estas instrucciones hay que definir los datos de origen en fase de diseño en el formato correcto.	Código del componente		E	BAJO		3		3				
		QA10C-04.020	QAC-0135	Tamaño de los programas Cobol	Se recomiendan programas pequeños, con menos de `5.000` líneas en fuente expandido. Se recomienda no diseñar programas con un número elevado de líneas de código con el fin de facilitar la depuración y el mantenimiento.	Código del componente		R	BAJO						2		
		QA10C-04.021	QAC-0136	Uso de comentarios dentro de los programas	Se deben poner comentarios al inicio de los párrafos que expliquen el proceso de los mismos.	Código del componente		R	BAJO						2		
		QA10C-04.022	QAC-0137	Párrafos, sentencias y variables no utilizadas	Todos los párrafos, sentencias y variables no utilizados deben ser eliminadas de los programas.	Código del componente		R	BAJO						2		
		QA10C-04.023	QAC-0138	Uso de sentencia PERFORM THRU	La codificación de la instrucción `PERFORM THRU` debe tener su correspondiente párrafo `EXIT`. La codificación de la etiqueta de la cláusula `THRU` sólo debe usarse para garantizar la legibilidad del punto de finalización del `PERFORM`; por esa razón, esta sentencia abarcará un máximo de dos párrafos o etiquetas (por ejemplo, `PROCESO-XXX` y `PROCESO-XXX-FIN`), y el contenido del segundo párrafo será una única sentencia `EXIT`. No debe existir ningún otro párrafo entre medias.	Código del componente		R	BAJO						2		
		QA10C-04.024	QAC-0139	Codificación de IF con END-IF	Se debe finalizar un `IF` con su `END-IF` correspondiente.	Código del componente		R	BAJO						2		
		QA10C-04.025	QAC-0140	Codificación de READ con END-READ	Se debe finalizar un `READ` con su `END-READ` correspondiente.	Código del componente		R	BAJO						2		
		QA10C-04.026	QAC-0141	Codificación de EVALUATE con END-EVALUATE.	Se debe finalizar un `EVALUATE` con su `END-EVALUATE` correspondiente.	Código del componente		R	BAJO						2		
		QA10C-04.027	QAC-0142	Codificación de SEARCH con END-SEARCH	Se debe finalizar un `SEARCH` con su `END-SEARCH` correspondiente.	Código del componente		R	BAJO						2		
		QA10C-04.028	QAC-0143	Uso del punto al final de párrafos y sentencias	Se recomienda poner punto sólo al final de párrafo. Evitar poner un punto al final de las sentencias excepto en: 1. Fin de párrafo 2.Sentencias `IF` que incluyan la sentencia `NEXT SENTENCE`; éstas no sólo deberán ir cerradas con `END-IF`, sino que además llevarán un punto detrás del `END-IF`. Esto es debido a que `NEXT SENTENCE` ignora el `END-IF`. En el caso que se cumpla la condición asociada al `NEXT SENTENCE`, la siguiente instrucción que se ejecutará será la que se encuentre inmediatamente después del punto más próximo. Por esta razón, no es recomendable el uso de la sentencia `NEXT SENTENCE` en su lugar se puede utilizar `CONTINUE`.	Código del componente		R	BAJO			2			2		
		QA10C-04.029	QAC-0144	Uso de ACCEPT FROM DATE, TIME o DAY	Se debe evitar el uso del `ACCEPT FROM DATE`, `TIME` o `DAY`. No se deben utilizar como fechas de proceso. La fecha de proceso debería ser un atributo más del modelo de datos de la aplicación.	Código del componente		E	BAJO			3			3		
		QA10C-04.030	QAC-0145	Uso de niveles 88	Se recomienda el uso de niveles `88` para manejo de valores constantes y booleanos en las variables gestionadas por la sentencia `IF`.	Código del componente		R	BAJO		2						
		QA10C-04.031	QAC-0146	Uso de XML en Cobol	No está permitida la lectura e interpretación de XML en Cobol. Sentencia PARSE prohibida.	Código del componente		B	ALTO		5				5		
		QA10C-04.032	QAC-0147	Generación de XML en Cobol	No está permitida la generación de XML en Cobol. Sentencia GENERATE prohibida.	Código del componente		B	ALTO		5				5		
		QA10C-04.034	QAC-0149	Detectado cursor definido numérico	Para prevenir la migración a Cobol Microfocus por el proyecto de Downsizing, no está permitido declarar	Código del componente		B	ALTO						5		
		QA10C-04.035	QAC-0150	No permitido CURRENT SERVER	Para prevenir la migracion a Cobol Microfocus por el proyecto de Downsizing, no esta permitido el uso de la sentencia CURRENT SERVER en sentencias SQL ya	Código del componente		B	ALTO						5		
		QA10C-04.036	QAC-0151	Detectada variable con ID repetido	Para prevenir la migración a Cobol Microfocus por el proyecto de Downsizing, no está permitido definir una variable con un nombre ya utilizado en el programa y no calificada o calificada incorrectamente.	Código del componente		B	ALTO						5		
		QA10C-04.037	QAC-0152	Detectado operador de concatenación	Para prevenir la migración a Cobol Microfocus por el proyecto de Downsizing, no está permitido utilizar el	Código del componente		B	ALTO						5		
		QA10C-04.038	QAC-0930	Detectado harcode local en programa paquetizable	No se deben codificar harcodes en programas paquetizables por su incoherencia con este tipo de programas y los riesgos que conllevan.	Código del componente		B	ALTO						5		
		QA10C-04.039	QAC-0931	Detectado valor hardcode o literal	Se debe evitar codificar valores harcodes y / o literales en los programas.	Código del componente		B	MODERADO						3		
		QA10C-04.040	QAC-0180	Prohibido el uso de la sentencia SQL SUBSTR	Prohibido el uso de la sentencia SQL SUBSTR	Código del componente		B	ALTO						3		
	QA10C-05	**HOST-COBOL: Gestión de errores, normas de codificación para el tratamiento de códigos de retorno**															
		QA10C-05.001	QAC-0153	Uso de variable RETURN-CODE	No está permitido el uso de la variable `RETURN-CODE`. EL flujo de ejecución debe ser condicionado por el planificador y no por el programa. Para condicionar ejecuciones posteriores se deben utilizar las condiciones de disparo del planificador.	Código del componente	SÓLO CPD	E	MODERADO						4		
		QA10C-05.002	QAC-0154	Valores no estandar de SQLCODE	Cualquier valor no estandar de `SQLCODE` debe tratarse como error. Se consideran valores estándar: 0 : ejecución correcta. +100: registro no encontrado / fin de cursor -803: registro a insertar duplicado según definición de índices únicos. -811: existe más de un registro que cumple la condición de búsqueda	Código del componente		B	ALTO			5			5		
		QA10C-05.003	QAC-0155	Uso de sentencia WHENEVER	Evitar el uso de la sentencia `WHENEVER`, controlar el `SQLCODE` de forma explícita. Es muy costoso, se	Código del componente		E	MODERADO			4			4		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-05.004	QAC-0156	Uso de DISPLAY	El uso de `DISPLAY` sólo se permite por fin de programa o por `ABEND`. No se puede utilizar la sentencia `DISPLAY` excepto en los casos en que informa de un `ABEND` controlado (errores `DB2` y errores del programa) ó para sacar estadísticas al final de la ejecución del programa. Durante la fase de construcción se puede utilizar la sentencia cuando sea necesario, pero cuando el módulo se pasa al entorno de Producción deben desaparecer todas las sentencias `DISPLAY` de pruebas de programas. La sentencia `DISPLAY` no se puede ejecutar en entornos `CICS`.	Código del componente		E	CPD Europa : BAJO CCR Méjico: ALTO		3		3		3		
		QA10C-05.006	QAC-0157	Devolución del área de comunicaciones con DB2	Las rutinas con sentencias `DB2` deben devolver todo el área de comunicaciones con el `DB2`, `SQLCA`, al programa principal. En procesos batch se deberá mostrar los campos de la `SQLCA` ante abends para reducir el tiempo de resolución de la incidencia.	Código del componente		E	BAJO			3			3		
		QA10C-05.007	QAC-0158	Mezcla de SQLCODE con variables del programa en sentencia IF	No mezclar `SQLCODE` con variables del programa en `IF`. Es obligatorio controlar el `SQLCODE` inmediatamente después de la sentencia `SQL`. No está permitida ninguna línea de código entre la sentencia `SQL` y el control de su `SQLCODE`. Además, la línea de control debe ser exclusiva para el `SQLCODE`, es decir, no se deben mezclar la condiciones del control de `SQLCODE` con variables `HOST` del programa.	Código del componente		E	BAJO			3			3		
		QA10C-05.008	QAC-0159	Control de WHEN OTHER/ELSE del SQL	Se debe controlar el `WHEN OTHER/ELSE` del `SQL` para que lleve a la cancelación controlada del programa.Los valores no estándar de `SQLCODE` que no se controlan explícitamente en el programa, deben controlarse de forma genérica con una cláusula `WHEN OTHER/ELSE` y provocar la cancelación del programa.	Código del componente		E	BAJO			3			3		
		QA10C-05.009	QAC-0160	Emplazamiento del control del SQLCODE	El control del `SQLCODE` debe hacerse inmediatamente después de cada acceso `SQL`, no está permitida ninguna línea de código que se pueda ejecutar entre la finalización de la sentencia `SQL` y el control del `SQLCODE`.	Código del componente		E	MODERADO			4			4		
		QA10C-05.010	QAC-0161	Valores a controlar con el SQLCODE	Deben controlarse explícitamente los valores del `SQLCODE`. - En sentencias `SELECT` sobre tabla con índice único e informando todos los campos del índice por igual: `SQLCODE = +0` y `+100` - En sentencias `SELECT` que puedan recuperar más de una fila: `SQLCODE = +0`,`+100` y `-811` - En sentencias `INSERT` sobre tablas con índice único: `SQLCODE = +0` y `-803` - En sentencias `INSERT` sobre tablas que no tienen índice único: `SQLCODE = +0` - En sentencias `DELETE`: `SQLCODE = +0` y `+100` - En sentencias `UPDATE`: `SQLCODE = +0`, `+100` y `-803`	Código del componente		B	ALTO			5			5		
	QA10C-06	**HOST-COBOL: Accesos y operaciones con rutinas.**																
		QA10C-06.001	QAC-0162	Número de llamadas a rutinas	No realizar más llamadas a rutinas de las permitidas (máximo: 10).Se recomienda diseñar los procesos evitando la excesiva modularización de programas. Este tipo de procesos implican un elevado consumo de	Código del componente		E	MODERADO		4		4		4		
		QA10C-06.002	QAC-0163	Repetición de llamadas a rutinas	No realizar diferentes llamadas a una misma rutina.Si es necesario recibir información diferente proporcionada por una misma rutina, se recuperá toda desde una única llamada y se utilizará a lo largo del programa.	Código del componente		E	BAJO		4		4		4		
		QA10C-06.003	QAC-0164	Forma de realizar llamadas a rutinas	Las llamadas a rutinas en `procedure` deben hacerse en forma dinámica, es decir, definiendo una variable en `WORKING` alfanumérica, a la cual en la `procedure` se le hará el `MOVE` del nombre de la rutina y posterior el llamado a dicha variable.	Código del componente	SÓLO CCR	E	BAJO						3		
		QA10C-06.004	QAC-0165	Realización de rutinas a medida	Se recomienda realizar rutinas a medida de las necesidades de las aplicaciones, con el fin de ahorrar accesos a tablas, realizar más de una llamada a rutinas, etc.	Código del componente		R	BAJO		2		2	2	2		
		QA10C-06.005	QAC-0166	Agrupar lógica de actualizaciones a BBDD en rutinas	Se recomienda realizar rutinas que aglutinen la lógica de actualizaciones a BB.DD., con ello se minimizan los deadlocks en procesos.	Código del componente		E	MODERADO		4	4	4	4	4		
		QA10C-06.006	QAC-0167	Rutinas para acceso a tablas propias	No deben crearse rutinas para el acceso a las tablas propias de la aplicación. Dicho acceso se hará directamente desde programa.	Código del componente		E	BAJO		3		3				
		QA10C-06.007	QAC-0168	Reutilización en Batch de la rutinas de Online	Se recomienda no reutilizar en Batch los servicios y rutinas diseñadas para uso Online. Si la funcionalidad es la misma, se pueden adaptar a este uso (o viceversa)	Código del componente		E	BAJO		3		3		3		
	QA10C-07	**HOST-ONLINE Especial para CICS.**																
		QA10C-07.001	QAC-0169	Uso de comandos CICS en invocación DPL	En caso de invocación `DPL (CAA-88-DPL)`, se prohíbe el uso de los siguientes comandos `CICS`: - Comandos de control de terminales referidos a su Principal Facility, - Comandos que establezcan o pregunten por atributos del terminal, - Comandos `BMS`, - Comandos `SIGNON` y `SIGNOFF` - Comandos de intercambio de datos con el Batch - Comandos de direccionamiento de la `TCTUA`	Código del componente			BAJO								
		QA10C-07.002	QAC-0170	Uso de instrucciones CICS en DPL	Las aplicaciones no deberán utilizar las instrucciones `CICS` prohibidas en `DPL`.	Código del componente			ALTO								
		QA10C-07.003	QAC-0171	Llamada a la rutina QG6CTUT para recuperar valores	Los programas de aplicación en on-line que no tengan acceso a la commarea de Arquitectura (`QGECCAA`) y necesiten recuperar el terminal lógico, la entidad, el centro contable, el terminal contable, el usuario que ejecuta la transacción, el `STARTCODE`, el idioma del terminal o el	Código del componente			ALTO								
		QA10C-07.004	QAC-0172	Intercomunicación entre aplicaciones vía LINK	En caso de que las aplicaciones requieran intercomunicarse entre sí vía `LINK` de programas y residan en `AORs` independientes, deberán hacer uso de un `TOR` como paso asignando una `TRANSID` para identificar las aplicaciones involucradas, está última deberá ser una copia de la `CSMI`	Código del componente			ALTO								
		QA10C-07.005	QAC-0173	Uso de comando ENQ	Si se utiliza el comando `ENQ` para controlar el acceso a algún recurso y garantizar la integridad de la información, se deberá emitir el comando `DEQ` lo más rápido posible	Código del componente			ALTO								
		QA10C-07.006	QAC-0174	Uso del nombre del CICS	Las aplicaciones que requieran hacer uso del nombre del `CICS` (applid, sysid) deberán hacerlo mediante el uso de variables o a través del comando `ASSIGN`, siempre y	Código del componente			BAJO								
		QA10C-07.007	QAC-0175	Verificación del código de retorno de las sentencias CICS	Siempre verificar el código de retorno de las sentencias `CICS` para evitar `ABENDs` en las transacciones.	Código del componente			BAJO								
		QA10C-07.008	QAC-0176	Uso de áreas de memoria	Las aplicaciones NO deberán utilizar áreas de memoria de manera indiscriminada, por ejemplo el empleo de colas `TS`	Código del componente			ALTO								
		QA10C-07.009	QAC-0177	Uso de colas TS	Se podrá aprovechar el uso de colas `TS` para cargar en memoria tablas `DB2` pequeñas. El número máximo de registros aconsejable es dependiente de la disponibilidad de memoria y el número de accesos a estas tablas. Se	Código del componente			ALTO								
		QA10C-07.010	QAC-0178	Prohibido el uso del comando NOSUSPEND para programas línea	Prohibido el uso del comando NOSUSPEND para programas línea	Código del componente			ALTO								
		QA10C-07.011	QAC-0181	Prohibido el uso del comando SEND TEXT	Prohibido el uso del comando SEND TEXT	Código del componente			ALTO								

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-07.012	QAC-0182	Prohibido el uso del comando TRACEID	Prohibido el uso del comando TRACEID	Código del componente			ALTO								
		QA10C-07.013	QAC-0185	Prohibido el uso de la sentencia SQL DROP	Prohibido el uso de la sentencia SQL DROP	Código del componente			ALTO								
		QA10C-07.014	QAC-0186	Prohibido el uso de la sentencia SQL CREATE	Prohibido el uso de la sentencia SQL CREATE	Código del componente			ALTO								
		QA10C-07.015	QAC-0953	Prohibido el uso del comando FORCEPURGE	Prohibido el uso del comando FORCEPURGE, se debe cambiar por el comando PURGE.	Código del componente			ALTO								
	QA10C-08	**HOST-JCL:Verificaciones que rigen la construcción de jcl's y cadenas.**															
		QA10C-08.001		Nombres de los pasos en los JCLS	Los nombres de los pasos deben corresponderse con el pgm o proc ejecutado. Siguiendo los estándares de la instalación, los nombres de los pasos o unidades de trabajo deberán corresponderse con el programa o procedimiento que se ejecuta.	Código del componente		R	BAJO								
		QA10C-08.002		Descripción de los pasos	Cada paso debe estar descrito mediante comentarios.	Código del componente		R	BAJO						1		
		QA10C-08.003		Nomenclatura de jcls, pasos, procedimientos, utilidades, ficheros, etc	La nomenclatura de jcl's, pasos, procedimientos, utilidades, ficheros, etc. se ajustará a cada instalación según los estándares de las misma. *CCR: http://intranet.ccr.igrupobbva/default.html* *CPD: ESPACIO APLICACIONES GLOBALES: NOMENCLATURA*	Código del componente		B	ALTO						5		
		QA10C-08.004		Uso de procedimientos estándares	Deberán utilizarse los procedimientos estándares de la instalación, desarrollados por las diferentes áreas y administrados por los técnicos correspondientes. Para asegurar la estabilidad de la producción, principalmente ante el cambio de versiones y upgrade de productos, deberán utilizarse los procedimientos estándares de Producción, desarrollados por las diferentes áreas técnicas.	Código del componente		E	BAJO								
		QA10C-08.005		Cláusula BLKSIZE	El `BLKSIZE` de la `DCB` en la definición de los ficheros deberá informarse a cero (o), para que el sistema determine el blocaje optimo. Esto es aplicable a la práctica totalidad de ficheros, a excepción de los ficheros con formato `U` y de los ficheros `VSAM`.	Código del componente		B	ALTO						5		
		QA10C-08.006		Nomenclatura de ficheros	Los ficheros deben cumplir con los estándares de nomenclatura de la instalación.	Código del componente		B	ALTO								
		QA10C-08.007		Realización de SORT	Evitar la realización de `SORT` repetitivos sobre el mismo fichero. Deberá analizarse la definición de procesos y las funcionalidades requeridas, para reutilizar ordenaciones previas del mismo fichero.	Código del componente		E	MODERADO		4		4		4		
		QA10C-08.008		Tamaño de las SORTWORK	Determinar los `SORTWORK` en base a los estándares de cada instalación.	Código del componente		R	BAJO						2		
		QA10C-08.009		Uso de procedimientos de operación sobre datos	Utilizar los procedimientos de operación sobre datos, desarrollados por cada instalación. Se utilizarán para el manejo de ficheros, creación, borrado, copia, etc. los procedimientos estándares de cada instalación.	Código del componente		E	BAJO		3		3		3		
		QA10C-08.010		Realización de descargas de la misma tabla	Evitar la realización de descargas reiterativas de la misma tabla. Deberá analizarse la definición de procesos y las funcionalidades requeridas, para reutilizar descargas previas de la misma tabla.	Código del componente		E	MODERADO		4	4	4	4			
		QA10C-08.011		Realización de cargas de la misma tabla	Evitar la realización de cargas reiterativas de la misma tabla.Deberá analizarse la definición de procesos y las funcionalidades requeridas, para realizar las cargas sobre una tabla en un único punto de la cadena.	Código del componente		E	MODERADO		4	4	4	4			
		QA10C-08.012		Uso de procedimientos sobre BBDD	Utilizar los procedimientos de operación sobre bbdd, desarrollados por BBDD. Para realizar cualquier tipo de acción sobre las bbdd `DB2` (descargas, cargas, borrados, modificaciones, etc.), se utilizarán los procedimientos estándares de la instalación. Si se utilizan otros procedimientos distintos se pueden producir problemas al realizar cambios de versión, upgrade de productos, etc.	Código del componente		E	BAJO		3		3		3		
		QA10C-08.013		Creación de ficheros en pasos previos	Los ficheros utilizados en pasos que ejecutan DB2 deberán existir, es recomendable por tanto que los ficheros se	Código del componente		R	BAJO						2		
		QA10C-08.014		Uso de SyncSort en pasos de ordenación	SyncSort debe ser utilizado como software de ordenación de ficheros en batch, excepto en los siguientes casos, para los cuales SYNCSORT actualmente no dispone de funcionalidad: 1. Ordenaciones sobre campos no informados en ficheros de longitud variables 2. Ordenaciones con parametrizaciones numéricas sobre ficheros alfanuméricos.	Código del componente		E	MODERADO		3						
		QA10C-08.015		Uso de la opción de compresión en ficheros.	En el momento de creación de ficheros (excepto ficheros VSAM), estos deben tener activa o no la opción de compresión dependiendo de los siguientes parámetros: - Los ficheros menores de 10GB deberían crearse descomprimidos. - Los ficheros mayores de 10GB deberian comprimirse cuando sea necesario que estén en disco o enviar a cartucho, según criterios funcionales.	Código del componente	SOLO para CPD Europa	R	BAJO	0							
	QA10C-09	**Código SS.DD.: Accesos a Bases de Datos y tratamiento de los datos en programación.**															
		QA10C-09.001	QAC-0190		Hacer uso de la clase `PreparedStatement`, en lugar de la clase `Statement`.	Código del componente		E	BAJO	1							
		QA10C-09.002	QAC-0191		Para aplicaciones `J2EE`, se debe hacer uso de objetos `DataSource` para acceder a los pooles de conexiones a	Código del componente		E	BAJO	1							
		QA10C-09.003	QAC-0192		Es necesario cerrar los recursos (Statement,PreparedStatement,ResultSet y Connection) después de usarlos. Nota: Es necesario cerrar las conexiones en orden inverso	Código del componente		E	ALTO	1							
		QA10C-09.004	QAC-0193		Transformación de sentencias `SQL`: Evitar el uso del operador `OR` en el caso de consultas en las cuales todas las condiciones implicadas disponen de un acceso vía índice. Para estos casos se recomienda la utilización de `UNION_ALL`. Ej.: `SELECT * FROM EMP WHERE DEPTNO = 20 OR EMPNO > 7746;` `SELECT * FROM EMP WHERE DETNO = 20 UNION ALL SELECT * FROM EMP WHERE EMPNO > 7746 AND DEPTNO !=20;`	Código del componente		E	BAJO								
		QA10C-09.005	QAC-0194		Reducir el tráfico en la Base de Datos. No se permite recuperar todos los campos de una tabla determinada mediante la instrucción: `SELECT * FROM tabla`	Código del componente		E	MODERADO								
		QA10C-09.006	QAC-0195		Se recomienda el uso de `DECODE` (Oracle) o `CASE` (DB2) en sentencias `ORDER BY` y `GROUP BY`, siempre y cuando el número de filas a recuperar sea pequeño.	Código del componente		R	BAJO								
		QA10C-09.007	QAC-0196		Evitar el uso de la cláusula `HAVING`, sustituir por la cláusula `WHERE`.	Código del componente		E	BAJO								
		QA10C-09.008	QAC-0197		Contar filas de Tablas. Se recomienda para contar filas de tablas el uso de `COUNT (INDEX_COLUMN)` en lugar de	Código del componente		E	BAJO								
		QA10C-09.009	QAC-0198		Uso de `Joins`. No deben realizarse `JOINS` de más de tres tablas dentro de una misma sentencia "`SELECT`".	Código del componente		E	BAJO								
		QA10C-09.010	QAC-0199		Usar la cláusula `NOT EXISTS` en vez de `NOT IN`, y preferiblemente haciendo referencia a una columna	Código del componente		R	BAJO								
		QA10C-09.011	QAC-0200		Uso del operador "`OR`". Usar el operador `UNION` en lugar de `OR`. Utilizar cláusulas `IN` y/o `UNION` en vez de la	Código del componente		E	BAJO								

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-09.012	QAC-0201		Evitar el uso de funciones en las cláusulas `WHERE` sobre campos que forman parte de algún índice.	Código del componente		E	MODERADO								
		QA10C-09.013	QAC-0202		Procedimientos almacenados. Se debe utilizar `Stored Procedures` para implementar el acceso a BDs desde aplicaciones Java. Para programas batch (o ante la necesidad de utilización) existe la posibilidad de utilizar la	Código del componente		E	BAJO								
		QA10C-09.014	QAC-0203		Los accesos por programa a `DB2-UDB`, las sentencias se tienen que codificar utilizando variables en lugar de constantes. El programa moverá los valores pertinentes a dichas variables antes de la ejecución de la sentencia. El	Código del componente		B	ALTO								
		QA10C-09.015	QAC-0204		Se impide la concatenacion de "String" con el operador "+" o con el método "concat" cuando uno de los argumentos es una cadena de un único caracter y este dentro de un bucle.	Código del componente		B	MODERADO								
	QA10C-10	**Código SS.DD. JAVA: Cálculos innecesarios.**															
		QA10C-10.001	QAC-0205		Minimizar el uso de cálculos repetidos innecesarios, utilizando variables temporales para almacenar valores.	Código del componente		E	BAJO	1							
		QA10C-10.002	QAC-0206		Llamadas a funciones en la condición de finalización de un bucle.	Código del componente		E	MODERADO	1							
		QA10C-10.003	QAC-0207		Evitar el uso de lógica condicional excesiva en los métodos, con un máximo nivel de anidación igual a 4.	Código del componente		R	BAJO	1							
		QA10C-10.004	QAC-0208		Evitar las transformaciones de datos al llamar a métodos o en los datos de salida. Recomendable definir métodos en `xsl:xsl` con los parámetros de entrada y salida adecuados	Código del componente		R	BAJO	0							
		QA10C-10.005	QAC-0209		Evitar el uso de mecanismos de recursividad; sustituir por un proceso iterativo.	Código del componente		E	BAJO	1							
		QA10C-10.006	QAC-0210		Evitar la creación innecesaria de objetos mediante `new`; los objetos deben crearse inmediatamente antes de su primer uso (pasando los parámetros de entrada a un método)	Código del componente		E	BAJO	1							
		QA10C-10.007	QAC-0211		Evitar la importación de los paquetes `javax.resource.cci` o `java.sql`, salvo excepciones de módulos explícitamente realizados para el acceso a	Código del componente		R	BAJO	0							
	QA10C-11	**Código SS.DD. JAVA: Entrada/Salida**															
		QA10C-11.001	QAC-0212		Detectar flujos de `E/S (sockets`, colas mq, ficheros, etc) que no se han cerrado en su lugar, ni en el `finally`.	Código del componente		E	MODERADO	1							
		QA10C-11.002	QAC-0213		Evitar llamadas a `System.out.println` y `System.err.println`, debido al coste de las mismas.	Código del componente		R	BAJO	1							
		QA10C-11.003	QAC-0214		Accesos a ficheros. Detectar accesos a ficheros con apertura/lectura/cierre de forma reiterativa	Código del componente		E	BAJO	0							
		QA10C-11.004	QAC-0215		Flujos con Buffer. Utilización en accesos de lectura y escritura a ficheros flujos con buffer como son (`BufferedInputStream`, `BufferedReader`, `BufferedOutputStream` y `BufferedWriter`), para un rendimiento eficiente.	Código del componente		E	BAJO	0							
	QA10C-12	**Código SS.DD.: Gestión de cadenas de texto y colecciones.**															
		QA10C-12.001	QAC-0216		Evitar la creación innecesaria de `strings`.	Código del componente		E	BAJO	1							
		QA10C-12.002	QAC-0217		Evitar el cálculo de longitud de `stringbuffer` mediante `string`.	Código del componente		E	BAJO	1							
		QA10C-12.003	QAC-0218		Comparación Cadenas: Detectar la comparación de caracteres mediante cadenas	Código del componente		E	BAJO	1							
		QA10C-12.004	QAC-0219		No debe usarse el operador '+' para la concatenación de Strings. En su defecto usar el método 'append' con StringBuffer o StringBuilder.NOTA: La concatenación con el operador '+' entre literales puros, no se considera violación.	Código del componente		E	MODERADO	1							
		QA10C-12.005	QAC-0220		Uso de `StringBuffer`: Dimensionar los `StringBuffer` de forma adecuada para que no sea necesaria su apliación .	Código del componente		E	BAJO	1							
		QA10C-12.006	QAC-0221		Evitar el uso de `StringTokenizer`.	Código del componente		R	BAJO	1							
		QA10C-12.007	QAC-0222		Evitar situaciones donde es recomendable el uso de `intern()` porque se efectúa la comparación de un `String` con un número alto de literales.	Código del componente		E	MODERADO	1							
		QA10C-12.008	QAC-0223		Se imposibilita la declaración de variables de tipos que son implementaciones de las interfaces: 'java.util.List', 'java.util.Set' y 'java.util.SortedSet' Estas son: - para 'java.util.List': AbstractList, LinkedList, Vector, y ArrayList. - para 'java.util.Set': AbstractSet, HashSet y para	Código del componente		R	BAJO	1							
		QA10C-12.009	QAC-0224		Minimizar el uso de colecciones sincronizadas (`Vector`, `HashTable`). Hacer uso de `ArrayList` y `HashMap`, que no son sincronizadas. Evitar el uso de `Collections.synchronizedCollection()`.	Código del componente		E	BAJO	1							
		QA10C-12.010	QAC-0225		En el caso de `HashMap`, comprobar el factor de carga respecto al tamaño dado al `HashMap`.	Código del componente		E	BAJO	1							
		QA10C-12.011	QAC-0226		Utilizar siempre `System.arrayCopy()` para realizar copias de arrays.	Código del componente		E	BAJO	1							
	QA10C-13	**Código SS.DD.: Sincronización, directrices de uso, multithreading.**															
		QA10C-13.001	QAC-0227		Evitar la sincronización de procesos de más de 15 líneas.	Código del componente		E	MODERADO	1							
		QA10C-13.002	QAC-0228		Evitar la lectura de recursos dentro de un bloque de código	Código del componente		R	BAJO	1							
		QA10C-13.003	QAC-0229		Sincronizar las operaciones de escritura a ficheros, en	Código del componente		E	BAJO	0							
		QA10C-13.004	QAC-0230		Evitar la existencia de atributos estáticos en los `servlets`.	Código del componente		R	BAJO	1							
	QA10C-14	**SS.DD. - Java: Creación, cacheo y casting de objetos**															
		QA10C-14.001	QAC-0231		Detectar la creación de `java.util.Date` o `java.util.GregorianCalendar`, y recomendar el	Código del componente		R	BAJO	1							
		QA10C-14.002	QAC-0232		Detectar llamadas al método `newTransformer()` de objetos de tipo `TransformerFactory`	Código del componente		E	BAJO	1							
		QA10C-14.003	QAC-0233		Detectar la creación y no cacheo de objetos costosos, como SimpleDateFormat (mediante una la lista	Código del componente		E	BAJO	1							
		QA10C-14.004	QAC-0234		No realizar más de una llamada al método `LookUp` para	Código del componente		E	BAJO	0							
		QA10C-14.005	QAC-0235		Para evitar el Casting innecesario de objetos, es recomendable crear un objeto temporal y hacer un único	Código del componente		E	BAJO	1							
	QA10C-15	**SS.DD. - Java: Definición de métodos**															
		QA10C-15.001	QAC-0236		Evitar definir como no estáticos atributos que sólo se acceden desde métodos estáticos y que sean comunes a todas las instancias de la clase.	Código del componente		R	BAJO	1							
		QA10C-15.002	QAC-0237		Evitar definir como no estáticos métodos que sólo usan	Código del componente		R	BAJO	1							
		QA10C-15.003	QAC-0238		Detectar aquellos métodos cuyo número de argumentos supera un máximo de 5.	Código del componente		E	BAJO	1							
	QA10C-16	**SS.DD. - Java: Otros**															
		QA10C-16.001	QAC-0239		En la gestión de errores, utilizar las clases de excepciones proporcionadas por el `J2EE`, con bloques `Catch` no	Código del componente		R	BAJO	1							
		QA10C-16.002	QAC-0240		Para la transformación `XML` con `XSL, hacer uso de la` especificación `JAXP`.	Código del componente		R	BAJO	0							
		QA10C-16.003	QAC-0241		Utilizar parsers `SAX` en lugar de parsers `DOM`, salvo en el caso de que se precise almacenar toda la información en	Código del componente		R	BAJO	0							
		QA10C-16.004	QAC-0242		Revisión de la serialización ya que es costosa, y debe evitarse. Recomendar el uso de `transient`.	Código del componente		R	BAJO	1							
		QA10C-16.005	QAC-0243		Evitar llamadas a `system.gc`.	Código del componente		R	ALTO	1	1						
		QA10C-16.006	QAC-0244		Debe evitarse el uso de reflexión. No utilizar la clase java.lang.Class ni las incluidas en el paquete java.lang.reflect (Constructor, Field, Method, ...).	Código del componente		R	BAJO	1							
		QA10C-16.007	QAC-0245		Antipatrón `Blob`:Detectar para un paquete, `JAR`, u otro tipo de agrupación de clases, si alguna de ellas cumple tiene un número excesivo de atributos y/o métodos, o carencia de `getters` y `setters`.	Código del componente		E	BAJO	1							
		QA10C-16.008	QAC-0992	Evite asignar valores a variables dentro de la condición de los bucles for.	Evite toda asignación en la parte de condición del bucle for.	Código del componente			ALTO	1		1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-16.009	QAC-0993	Evite reutilizar las variables de control en los bucles anidados.	La reutilización de una variable de control de bucle como inicializador y control de un bucle anidado es peligrosa, ya que el valor de la variable se modificará. Esto tendrá efectos negativos en los otros bucles que utilicen esta variable como control.	Código del componente			ALTO	1		1					
		QA10C-16.010	QAC-0994	Evite llamar a toString() en una matriz.	La invocación del método toString() en una matriz dará como resultado una serie de caracteres aleatorios irreconocibles. En su lugar, utilice el método Arrays.string().	Código del componente			MODERADO	1		1					
		QA10C-16.011	QAC-0995	Evite utilizar == y != para comprobar si un valor float/double es NaN.	NaN no representa ningún valor, por lo que no hay valor igual a el. Como resultado, para comprobar si un valor float o double es NaN, debe utilizarse el método isNaN().	Código del componente			MODERADO	1		1					
		QA10C-16.012	QAC-0996	Evite lanzar NullPointerExceptions.	No se recomienda lanzar NullPointerExceptions.	Código del componente			MODERADO	1		1					
		QA10C-16.013	QAC-0997	Evite utilizar el método await() de la clase java.util.concurrent.locks.Condition fuera de un bucle while.	La llamada a await() suspende la ejecución del hilo hasta que otro hilo notifique que se ha alcanzado el estado previsto del programa. Se recomienda que la llamada a await() se realice dentro del bucle while para que la condición de parada del bucle pueda verificar que el estado de la aplicación es el esperado. Imagine que podría haber un tercer hilo que podría notificar al primero y en este caso la aplicación no se encontraría en el estado esperado.	Código del componente			ALTO	1		1					
		QA10C-16.014	QAC-0998	Evite llamar a wait() en un objeto java.util.concurrent.locks.Condition.	La espera de un objeto java.util.concurrent.locks.Condition no debe realizarse utilizando el método wait() de la clase java.lang.Object. La espera debe realizarse utilizando el método await() de la clase java.util.concurrent.locks.Condition.	Código del componente			ALTO	1		1					
		QA10C-16.015	QAC-0999	Utilice siempre 'synchronize' para los métodos run().	Toda clase que implemente la interfaz "Runnable' debe tener un método 'run()' y este método run debe tener el modificador 'synchronize'.	Código del componente			ALTO	1		1					
		QA10C-16.016	QAC-1000	Evitar llamar a java.lang.System.loadLibrary() o java.lang.Runtime.loadLibrary().	Puesto que el código nativo se ejecuta directamente en el sistema operativo subyacente, se salta todas las restricciones de seguridad Java y podría robar información privada o modificar el sistema operativo.	Código del componente			ALTO	1		1					
		QA10C-16.017	QAC-1001	Evite llamar a java.lang.System.setSecurityManager().	El gestor de seguridad ya está definido en el Contenedor J2EE. De forma predeterminada, evite los cambios personalizados. Minimice los cambios en el comportamiento predeterminado del sistema. En su lugar cambie la política con el SecurityManager existente.	Código del componente			ALTO	1					1		
		QA10C-16.018	QAC-1002	Evitar métodos nativos protegidos.	Por definición, los métodos nativos están fuera del sistema de seguridad de Java. Ni el gestor de seguridad ni ningún otro mecanismo de seguridad de Java está diseñado para controlar el comportamiento del código nativo. Por lo tanto, los errores o los agujeros en la seguridad del código nativo pueden crear problemas graves. Examine los parámetros que toman y los valores que devuelven los métodos nativos. Específicamente, si un método nativo se salta las comprobaciones de seguridad de Java, debe definir cuidadosamente la modalidad de acceso al método. Debe examinar las posibles consecuencias del método y decidir si debería ser privado.	Código del componente			ALTO	1					1		
		QA10C-16.019	QAC-1003	Evitar métodos nativos públicos.	Por definición, los métodos nativos están fuera del sistema de seguridad de Java. Ni el gestor de seguridad ni ningún otro mecanismo de seguridad de Java está diseñado para controlar el comportamiento del código nativo. Por lo tanto, los errores o los agujeros en la seguridad del código nativo pueden crear problemas graves. Examine los parámetros que toman y los valores que devuelven los métodos nativos.Específicamente, si un método nativo se salta las comprobaciones de seguridad de Java, debe definir cuidadosamente la modalidad de acceso al método. Debe examinar las posibles consecuencias del método y decidir si debería ser privado.	Código del componente			ALTO	1					1		
		QA10C-16.020	QAC-1004	Evite construir querys con parámetros dados por el usuario	Construir sentencias SQL que concatenen código SQL con datos bajo el control de los usuarios es lo mismo que abrir un conexión a la base de datos para los usuarios del sistema. Esta es la vía más común para los ataques de inyección de SQL (conocido problema de seguridad).	Código del componente			ALTO	1					1		
		QA10C-16.021	QAC-1005	JDBC.SETPS: Cargar los valores de los parámetros del PreparedStatement antes de su consulta.	Se debe indicar los parámetros de la consulta antes de ejecutar la misma. Es decir, invocar a los métodos setters de los campos correspondientes.	Código del componente			BAJO	1	1						
		QA10C-16.022	QAC-1006	Evitar llamar a finalize() excepto en el bloque finally del método finalize().	El método finalize() se ha diseñado para que lo invoque la JVM en los objetos que se van a recoger como basura. A veces, cuando el programador define explícitamente el método finalize(), el método libera recursos adquiridos por el objeto. En aplicaciones J2EE, es recomendable no utilizar finalize() y en su lugar definir un mecanismo explícito para adquirir y liberar recursos. Independientemente de si se sobrescribe o no el método finalize, no es recomendable invocarlo explícitamente. Tales invocaciones pueden provocar problemas de memoria y pueden llevar a problemas de sincronización en aplicaciones con hebras.	Código del componente			ALTO	1	1						
		QA10C-16.023	QAC-1008	Evitar declarar finalize() public.	El método finalize() se ha diseñado para que lo invoque la JVM en objetos que se van a recoger como basura. Cuando el método de finalización se define explícitamente, el método libera recursos adquiridos por el objeto. En aplicaciones J2EE, es recomendable no utilizar finalize y en su lugar definir un mecanismo explícito para adquirir o liberar recursos. Al alterar temporalmente finalize(), mantenga protegido el modificador de acceso. Al cambiarlo a public se permite que un código externo llame explícitamente a finalize(), lo que no es recomendable.	Código del componente			ALTO	1	1						

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-16.024	QAC-1009	Posible confusión entre asignación y comparación en una expresión condicional.	La regla detecta asignaciones de variables (usos del operador '=') en la expresión de una estructura condicional 'if'. Es posible que en realidad el programador quisiera realizar una comparación (uso del operador '==').	Código del componente			MODERADO	1		1				
		QA10C-16.025	QAC-1010	Evite llamar a java.applet.* desde el método finalize() de un servlet.	Ninguna operación de CPU intensiva o que tenga posibilidad de bloqueo debe ser llamada desde el método finalize(). El método finalize() es ejecutado por el recolector de basura y por esta razón debe ejecutar tareas complejas.	Código del componente			ALTO	1	1						
		QA10C-16.026	QAC-1011	Evite llamar a java.lang.ClassLoader desde el método finalize() de un servlet.	Ninguna operación de CPU intensiva o que tenga posibilidad de bloqueo debe ser llamada desde el método finalize(). El método finalize() es ejecutado por el recolector de basura y por esta razón debe ejecutar tareas complejas.	Código del componente			ALTO	1	1						
		QA10C-16.027	QAC-1012	Evite llamar a java.lang.Compiler desde cualquier servlet.	Utilizar el compilador es una operación que consume mucha CPU y que limita mucho el rendimiento y la escalabilidad de una aplicación J2EE. Si debe compilarse el código, debe compilarse antes de utilizar la aplicación J2EE.	Código del componente			ALTO	1	1						
		QA10C-16.028	QAC-1013	Evite llamar a java.lang.reflect.* desde cualquier servlet.	Utilizar el reflejo dificulta la compresión y la depuración del código de la aplicación y origina problemas de rendimiento. Por estos motivos, el reflejo debería evitarse en la medida de lo posible. En su lugar, utilice interfaces con métodos bien conocidos.	Código del componente			MODERADO	1	1						
		QA10C-16.029	QAC-1014	Evita utilizar métodos nativos definidos por el usuario.	Se impide métodos nativos definidos por el usuario.	Código del componente			ALTO	1							1
		QA10C-16.030	QAC-1015	No utilizar rutas absolutas.	No utilizar rutas absolutas en los constructores de las clases definidas. La regla comprobara tanto si en el constructor se indica directamente una ruta absoluta, como si se utilza una variable String a la que previamente se le ha asignado una ruta absoluta. La lista separada por comas de constructores de clases es parametrizable.	Código del componente			MODERADO	1							1
		QA10C-16.031	QAC-1016	Evita utilizar directamente interfaces java.awt.peer.*.	Se impide utilizar directamente las interfaces definidas en el paquete "java.awt.peer".	Código del componente			ALTO	1							1
		QA10C-16.032	QAC-1017	Hacer el método clone() final por seguridad.	Se requiere que las clases que implementan la "interface" 'Cloneable' tengan un método 'clone ()' de caracter "final".	Código del componente			ALTO	1					1		
		QA10C-16.033	QAC-1018	Impide la comparacion de objetos de clase con el método getName().	Se impide la comparacion de objetos de clase utilizando el método 'getName ()'.	Código del componente			ALTO	1					1		
		QA10C-16.034	QAC-1019	Campos 'transient' en clases 'Serializable'.	Si hace sus clases 'Serializable', asegurese de que cada campo es 'transient' (los campos 'transient' no serán serializados). Se desaconseja, en la medida de lo posible, la utilización de clases que extiendan la interfaz 'java.io.Serializable'. En el caso de que sea necesaria la extensión, haga todos los campos 'transient'.	Código del componente			ALTO	1					1		
		QA10C-16.035	QAC-1020	Evite invocar System.exit().	System.exit() es una forma peligrosa de salir de una clase. En su lugar, lance una RuntimeException.	Código del componente			ALTO	1		1					
		QA10C-16.036	QAC-1021	Evite invocar java.lang.System.runFinalizers OnExit().	java.lang.System.runFinalizersOnExit() es una forma peligrosa de salir de una clase	Código del componente			ALTO	1		1					
		QA10C-16.037	QAC-1022	Evita el uso de variables de tipo java.lang.ThreadGroup.	Se impide el uso de variables 'java.lang.ThreadGroup'.	Código del componente			ALTO	1					1		
		QA10C-16.038	QAC-1023	Evita el uso de Thread.yield.	Se impide el uso de 'Thread.yield()' y 'Thread.currentThread().yield()'	Código del componente			ALTO	1	-						1
		QA10C-16.039	QAC-1024	Impide llamar de un método sincronizado a otro.	Se impide realizar llamadas desde un método "sincronizado" a otro.	Código del componente			ALTO	1		1					
		QA10C-16.040	QAC-1025	Utiliza charAt()en vez de startsWith()para las comparaciones de un caracter.	Se impide el uso de 'String.startsWith("string constant")', cuando la constante es una cadena de texto de un único caracter.	Código del componente			BAJO	1	1						
		QA10C-16.041	QAC-1026	Evite los métodos finalize() vacíos.	Los métodos finalize vacíos utilizan recursos de forma innecesaria.	Código del componente			MODERADO	1	1						
		QA10C-16.042	QAC-1027	Compruebe que no se escribe una direccon IP en el código	Se requiere referenciar al nombre de la maquina en cuestion y no a la direccion IP. La regla no mostrará violación sobre la IP local (127.0.0.1) ni sobre máscaras de red.	Código del componente			MODERADO	1							1
		QA10C-16.043	QAC-1028	Impide la creación de variables locales DataSource en métodos	Se impide la creación de variables locales de tipo DataSource en métodos.	Código del componente			ALTO	1	1						
		QA10C-16.044	QAC-1029	Evite llamar a java.util.zip.* desde cualquier servlet.	Utilizar el programa de utilidad de Zip es una operación que consume mucha CPU y que limita mucho el rendimiento y la escalabilidad de una aplicación J2EE.	Código del componente			ALTO	1	1						
		QA10C-16.045	QAC-1030	No devolver objetos nuevos en el return de un método.	Esta regla identifica los métodos que devuelven un objeto nuevo en lugar de devolver parámetros. Normalmente el método crea un nuevo objeto que podria ser rapidamente descartado, esto afecta a la memoria del dispositivo. Para evitar ese comportamiento indeseado, el método puede devolver un parámetro que modifica y asi estamos reutilizando memoria.	Código del componente			MODERADO	1	1						
	QA10C-17	**SS.DD. - Nácar**															
		QA10C-17.001	QAC-0246		Se debe utilizar `Stored Procedures` para implementar el acceso a BDs desde aplicaciones Nacar al DB2 en host.	Código del componente		E	BAJO	0							
		QA10C-17.002	QAC-0247		Avisar de que los pasos de flujo para la navegación externa son "sin vuelta", cuando éstos parten de un servicio de	Código del componente		E	BAJO	0							
		QA10C-17.003	QAC-0248		Comprobar que cada flujo llama como máximo a una sola ventana, realizándose las llamadas a ventanas auxiliares	Código del componente		R	BAJO	0							
		QA10C-17.004	QAC-0249		Antipatrón `Blob`: Detectar para un flujo, el patrón en el que una ventana es el centro de toda la funcionalidad, y a partir de ella se invocan otros flujos o servicios que terminan volviendo a esta macro ventana. Permitir definir unos	Código del componente		E	BAJO	0							
		QA10C-17.005	QAC-0250		Se recomienda en las aplicaciones el uso de librerías de log para trazas de aplicación como por ejemplo log4j. Al utilizarlas no se debe hacer uso de los niveles FATAL, ERROR y WARN. Las trazas de aplicación deben escribirse con INFO o DEBUG (solo para depuración y no para entornos productivos) dejando el resto de niveles para la gestión de errores.	Código del componente		E	BAJO	0							
		QA10C-17.006	QAC-0251		Revisar tamaños del `XML` de las ventanas no superior a `2KB y con no` más de 30 campos.	Código del componente		E	BAJO	1							
	QA10C-18	**SS.DD. - JSPs**															
		QA10C-18.001	QAC-0252		Antipatrón "`Too much code`". Controlar que el porcentaje de código `JAVA` que hay en la página `JSP` no supere el	Código del componente		E	BAJO	1							
		QA10C-18.002	QAC-0253		Antipatrón "`Ignoring reality`". Comprobar que cada página contiene la directiva necesaria para mostrar una	Código del componente		E	BAJO	1							
		QA10C-18.003	QAC-0254		En aquellas `JSP` que se correspondan con un `XML`, comprobar que el número de campos a enviar en el	Código del componente		E	BAJO	1							
		QA10C-18.004	QAC-0255		Para aquellos `XSL` que se correspondan con un `JSP`, comprobar que todos los valores establecidos en el	Código del componente		R	BAJO	1							

CODIGO CONTROL /VERIFICACIÓN	Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
										COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
QA10C-19	**SS.DD. - HTML**															
	QA10C-19.001	QAC-0256		Evitar la definición de estilos mediante el tag `Style` (recomendar las hojas de estilos).	Código del componente		E	BAJO	1							
	QA10C-19.002	QAC-0257		Evitar el uso del uso del tag `font` (recomendar las hojas de estilos).	Código del componente		E	BAJO	1							
	QA10C-19.003	QAC-0258		Evitar el uso de comentarios en `HTML` en entornos distintos al de Desarrollo.	Código del componente		R	BAJO	1							
	QA10C-19.004	QAC-0259		Evitar el uso de comentarios en JavaScript en entornos distintos al de Desarrollo.	Código del componente		R	BAJO	1							
	QA10C-19.005	QAC-0260		Evitar la inicialización de arrays mediante una secuencia de asignaciones para cada uno de los elementos que lo	Código del componente		E	BAJO	1							
	QA10C-20	**SS.DD. - Portlets**														
	QA10C-20.001	QAC-0261		**Utilizar paso de parámetros a métodos.** Los portlets son instancias únicas dentro de la JVM. Una misma instancia del portlet sirve todas las peticiones y debe por tanto ser thread-safe para permitir invocaciones concurrentes sin conflictos entre ellas. Cada petición deberá disponer de su propia información independientemente de la concurrencia de peticiones. Se debe evitar el uso de variables de instancia (variables no estáticas) y el paso de información a los métodos debe hacerse por parámetro.	Código del componente		E	BAJO	1							
	QA10C-20.002	QAC-0262		**Delimitación de contenido HTML entre página y portlet**. El portlet devuelve una salida HTML que se renderizará en el navegador. Esa salida se encuentra dentro del marco de una página con más contenido HTML. Es importante distinguir qué fragmentos de la página devuelve cada elemento del servidor. La salida del portlet se encontrará en una celda de una tabla (<TD>…</TD>) y por tanto no deberá contener etiquetas <HTML>, <BODY> ó <HEAD>. La generación de estas etiquetas corresponde a la página que contiene el portlet. Éste sólo deberá generar el contenido para el que se crea mientras que el script o página continente se encargará de generar el layout y los elementos de apertura y cierre de documento HTML.	Código del componente		R	BAJO	1							
	QA10C-20.003	QAC-0263		**Consideraciones de dimensionamiento de de portlets**. Los portlets generan de forma dinámica el contenido de una parte de la página mostrada en el navegador. Por este motivo es importante considerar las dimensiones de los mismos ya que si no se dimensionan correctamente para la porción de página disponible aparecerán elementos de scroll en el navegador. Se debe evitar dimensionar el portlet con una cantidad exacta de pixeles. El tamaño del portlet debe adaptarse a diferentes tamaños de pantalla y resoluciones e interesa que su tamaño sea relativo a la ventana en la que se ubica.	Código del componente		R	BAJO	1							
	QA10C-20.004	QAC-0264		**Utilizar clases de estilo en lugar de estilos individuales.** Los portlets pueden utilizar los estilos definidos en las CSS del portal. Al igual que ocurre con otros elementos es muy recomendable utilizar hojas de estilos en lugar de estilos individuales para cada portlet. De esta forma se facilita la configuración del aspecto del portlet y se agiliza su mantenimiento.	Código del componente		E	BAJO	0							
	QA10C-20.005	QAC-0265		**Codificación de nombres por namespace.** Todos los nombres de los elementos randerizados por los portlets de una misma página deben ser distintos. Se recomienda que los nombres de estos elementos HTML referencien al portlet que los genera de forma que no exista repetición de nombres entre elementos similares de portlets distintos. Para facilitar esta tarea se utiliza el tag <portletAPI:encodeNamespace/> que se encarga de añadir el nombre del portlet como prefijo de la nomenclatura del elemento HTML. A continuación se muestra un ejemplo,	Código del componente		R	BAJO	1							
	QA10C-20.006	QAC-0266		**Minimizar el uso de JavaScript.** El portlet debe depender lo menos posible de la utilización de JavaScript. Puesto que la implementación que los diferentes navegadores hacen de Javascript es diferente cuanto menos se utilice mayor será la portabilidad del portlet entre navegadores. Al mismo tiempo, cuanto menos código JavaScript exista en la página mayor será la portabilidad a otros lenguajes de marcas diferentes a HTML.	Código del componente		R	BAJO	0							
	QA10C-20.007	QAC-0267		**Limitar el uso de ventanas emergentes.** La interacción en el portal está basada en estados. Para ello el portal guarda información de navegación a través de páginas y portlets. La generación de una ventana emergente que contenga portlets puede generar una pérdida de esta información. Para evitar los popups se puede abrir un nuevo navegador con el contenido deseado en lugar de una ventana emergente. En caso de utilizar ventanas emergentes es conveniente cumplir las siguientes recomendaciones generales: • El popup debe cerrarse si se cierra la página que lo genera • Si se pincha sobre el enlace que muestra el popup y éste ya existe, el foco se debe situar en la ventana emergente existente sin generar otra, independientemente de que esta esté o no mostrada en background	Código del componente		R	BAJO	1							
	QA10C-20.008	QAC-0268		**Utilización de taglibs.** La utilización de taglibs para encapsular código Java permite, aparte de la reutilización de código, que la codificación del JSP sea más limpia y se aprecie mejor el layout de la página. La claridad del contenido del JSP simplifica su mantenimiento, más aún considerando que dentro de la estructura de la página se incrustan los portlets, que a su vez verterán también contenido HTML una vez renderizada la página.	Código del componente		R	BAJO	0							
	QA10C-20.009	QAC-0269		**Consideraciones del uso de IFRAMEs.** Es posible utilizar un IFRAME para incluir contenido de una URL externa dentro de un portlet. No obstante es conveniente considerar tres aspectos a tener en cuenta antes de utilizar este recurso: • La URL incluida debe estar accesible desde el servidor • No todos los navegadores soportan los IFRAMEs • Si el contenido incluido es mayor que la región del portlet se mostrarán barras de scroll. Este mismo contenido externo, proveniente de otra URL, puede tener también barras de scroll con lo que se tendrían dos scrolls verticales y otros dos horizontales, situación claramente a evitar.	Código del componente		R	BAJO	0							
	QA10C-20.010	QAC-0270		**Asignación de TARGET a los fornularios.** Es importante revisar que los formularios generados por los portlets tienen todos informados su target. De lo contrario cuando se submitan el contenido devuelto reemplaza toda la página y no sólo el área del portlet correspondiente.	Código del componente		R	BAJO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-20.011	QAC-0271		**Soporte de varios navegadores.**Para visualizar correctamente el contenido de la página en varios navegadores hay que evitar utilizar la detección del tipo de navegador para generar las etiquetas correctas para ese tipo de navegador. Si la adecuación al navegador se realiza de esta forma se corre el riesgo de que se muestre un contenido cacheado que no esté adaptado al navegador que accede en ese momento. Para evitar esto se recomienda incluir las distintas etiquetas HTML válidas para cada navegador. Esta redundancia de etiquetas no produce ningún error de renderización puesto que el navegador no las muestra al no reconocerlas. De esta forma se asegura que, aunque el contenido se cachee, estará disponible la etiqueta para cada navegador.	Código del componente		R	BAJO	1							
		QA10C-20.012	QAC-0272		**Propagación de identidad (single sing-on).** Si un usuario requiere autenticarse para acceder al portal, los portlets en cuestión deben poder propagar la información de autenticación a otras aplicaciones o bien convertir dicha información en información válida de autenticación para esas aplicaciones. Esto se puede llevar a cabo mediante el	Código del componente		R	BAJO	0							
		QA10C-20.013	QAC-0273		**Uso del repositorio Credential Vault con información sensible.**El repositorio Credential Vault permite almacenar información sensible (identidades, certificados, claves privadas, contraseñas, etc.) de forma segura. En el repositorio existen dos tipos de credenciales: pasivas y activas. Las pasivas permiten al portlet recuperar la información almacenada en el repositorio. Las activas se	Código del componente		R	BAJO	0							
		QA10C-20.014	QAC-0274		**SSL con autentificador básico.** En caso de utilizar autenticación básica en el portal el usuario y la contraseña se envían en la cabecera HTTP con una codificación débil. Puesto que la identidad del usuario se propagará por todo el portal es conveniente que se utilice SSL para que estos datos viajen cifrados.	Código del componente		E	MODERADO	0							
		QA10C-20.015	QAC-0275		**Utilizar PortletRequest para los datos entre vista y controlador.**Los datos que se muestran en la vista provenientes del controlador deben pasarse utilizando el objeto PortletRequest. Una vez completada la petición este objeto se liberará y se recolectará convenientemente. El uso del objeto PortletRequest simplifica el manejo de datos a la utilización de funciones get() y set() con los nombres de los datos correspondientes.	Código del componente		E	BAJO	1							
		QA10C-20.016	QAC-0276		**Publicar funciones comunes a varios portlets.** Cuando existen funciones similares o idénticas en diferentes portlets es conveniente agruparlas y tenerlas accesibles de forma externa para todos los portlets. Una opción para modularizar y publicar esta funcionalidad común es crear un JAR con las clases comunes y depositarlo en el directorio de librerías compartidas del servidor de aplicaciones.	Código del componente		E	BAJO	1							
		QA10C-20.017	QAC-0277		**Agrupar funcionalidad similar en el mismo portlet.**Si existen varios portlets con funciones similares se recomienda crear un único portlet que se comporte de forma diferente en función de sus settings. Cada una de las funciones realizadas por el portlet tendrá un setting unívoco que le configurará para hacer esa función concreta. Dichos settings pueden definirse en el portlet.xml y configurarse posteriormente en el método doConfigure().	Código del componente		E	BAJO	1							
		QA10C-20.018	QAC-0278		**Utilizar ficheros properties para internacionalización.** Si el contenido del portlet debe generarse para varios idiomas se recomienda incluir todo el contenido a internacionalizar en ficheros .properties. Esta recomendación es idéntica a otro tipo de aplicaciones.	Código del componente		E	BAJO	1							
		QA10C-20.019	QAC-0279		**Utizar comentarios en Java.** Los comentarios en Java son preferibles a los comentarios en HTML puesto que estos últimos se renderizan en la página ocupando tiempo y ancho de banda. Si los comentarios del JSP se realizan en la parte Java el servidor se encarga de eliminarnos y no se propagan hasta el navegador cliente. Para hacer un comentario en Java basta con abrir y cerrar scriptlets y realizar dentro el comentario.	Código del componente		R	BAJO	1							
		QA10C-20.020	QAC-0280		**Limitar hilos de ejecución.**Se debe limitar el uso de Threads en tiempo de ejecución. Si fuera necesario utilizarlos su duración debe ser lo más limitada posible. Puesto que los portlets son ya de por sí multihilo es importante evitar la generación de más hilos para evitar problemas de sincronización.	Código del componente		E	MODERADO	1							
		QA10C-20.021	QAC-0281		**Evitar la creación o asignación de variables y Objetos temporales dentro de un bucle** ya que puede producir costes de rendimiento.	Código del componente		R	BAJO	0							
		QA10C-20.022	QAC-0282		**Evitar métodos sincronizados.** La sincronización es un cuello de botella en el portlet puesto que hay un punto donde se debe esperar a la ejecución de un solo hilo para continuar con el resto. Este factor hace que aumente también el tiempo medio de respuesta del portlet y la probabilidad de bloqueo del mismo.	Código del componente		E	MODERADO	0							
		QA10C-20.023	QAC-0283		**Evitar bucles de larga duración.** Se deben evitar bucles de larga duración y en general cualquier operación de duración considerable. Estos procesos hacen que la página espere a la ejecución del portlet en cuestión para renderizar el contenido, aumentando así el tiempo total de respuesta. Si fuera necesario incluir alguna operación de este tipo en un portlet se debe estudiar si existe algún camino alternativo para fraccionar este coste temporal en varios pasos paralelizables más pequeños, teniendo en cuenta la restricción de no consumir un número grande de hilos para realizar esta tarea.	Código del componente		E	MODERADO	0							
		QA10C-20.024	QAC-0284		**Emplear JSP en lugar de XML/XSLT**. Para mostrar una misma información en el navegador, es en general más eficiente utilizar JSP que aplicar una plantilla XSL sobre un documento XML. El JSP se compila en un servlet que se ejecuta mucho más rápidamente que la transformación XSLT. Esta transformación genera múltiples objetos String temporales que son necesarios generar y destruir en un período de tiempo breve con el consiguiente gasto de CPU y memoria asociado. En caso de requerirse esta transformación se debe reducir en la medida de lo posible tanto el parseado XML como la transformación XSLT.	Código del componente		R	BAJO	1							
		QA10C-20.025	QAC-0285		**Configurar cacheado de los portlets.** Si la salida del portlet es de naturaleza estática o se mantiene constante durantes períodos considerables de tiempo es conveniente cachear este contenido para evitar ejecuciones innecesarias que devolverían idéntica información. Para cachear este contenido dentro del fichero portlet.xml debemos utilizar la etiqueta <expiration-cache>...</expiration-cache> que contendrá la duración en segundos del cacheado de contenido. El portlet puede implementar el método getLastModified() para invalidar el contenido de la caché de forma que en la siguiente petición se llame al método doView() para generar la salida de nuevo.	Código del componente		E	MODERADO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-20.026	QAC-0286		**Asignación y calculo de todas las propiedades de portles de forma paralela antes de mostrarlo en pantalla.** Cuando un determinado portlet tiene un tiempo de carga elevado es conveniente que se renderice en paralelo con otros portlets para que se reduzca el tiempo total de renderizado de la página. Configurar la renderización paralela de portlets depende del servidor de aplicaciones donde se sirva el contenido. Es importante fijar correctamente el parámetro timeout del renderizado ya que transcurrido el tiempo máximo fijado se parará el proceso y se lanzará una excepción.	Código del componente		R	MODERADO	1							
		QA10C-20.027	QAC-0287		**Recomendaciones en el manejo de sesiones.** En un portlet es posible manejar sesiones de usuario a través del objeto PortletSession. Sin embargo, no se recomienda utilizar sesiones en caso de que el portlet se acceda de forma anónima, sin autenticación. En este caso, si fuera necesario mantener algún tipo de información, se puede recurrir a su almacenamiento en campos hidden o en cookies en lugar de crear una sesión para almacenar los datos. En el caso de que se utilizaran sesiones de usuario, se debe emplear el objeto PortletSession para almacenar únicamente información global de usuario que sea necesaria para todas las peticiones. Esta consideración se basa en el hecho de que el manejo de sesiones implica un cierto consumo de CPU y de memoria dado que se almacena información de forma temporal que no se liberará hasta que no se cierre o se termine la sesión.	Código del componente		R	MODERADO	1							
		QA10C-20.028	QAC-0288		**Servicios de comunicación entre Portlets.** Si es necesario comunicar información entre portlets éstos pueden implementar la interfaz PortletMessage para poder enviarse información entre sí. La difusión de mensajes puede hacerse portlet a portlet o a todos los portlets de la página. Los portlets pueden prepararse para responder a eventos de recepción de mensajes implementando los métodos de la interfaz MessageListener para ejecutar la acción correspondiente según el mensaje recibido.	Código del componente		R	BAJO	1							
		QA10C-20.029	QAC-0289		**Métodos para compartir datos entre aplicaciones.** El objeto PortletSession almacena datos de sesión en el ámbito del portlet, al igual que ocurre con el objeto HttpSession, si se quiere replicar la sesión en otros ámbitos es necesario recurrir a los servicios de replicación de sesión del servidor de aplicaciones. Existen formas alternativas para compartir datos de sesión entre diferentes aplicaciones en función del ámbito de los datos: • Si un servlet y un portlet se encuentran en el mismo [illegible]	Código del componente		R	BAJO	1							
		QA10C-20.030	QAC-0948		Se requiere que 'StringBuffer', 'StringBuilder', 'ArrayList', 'HashMap', 'HashSet', 'Hashtable', 'Vector' y 'WeakHashMap' se instancien con una capacidad inicial específica.	Código del componente		R	BAJO	0							
	QA10C-21	**SS.DD. - TERADATA**																
		QA10C-21.001		Vistas	Vistas. No están permitidas vistas que en su definición incluyan más de una tabla. Por rendimiento, la utilización de este tipo de vistas es muy costosa. Las Vistas en	Código del componente		E	BAJO	1							
		QA10C-21.002		Generación de DDLs	**Generación de DDLs.** Para un mejor control de la correcta sintaxis de los ficheros SQL se solicita la creación de DDL separados de COMMENT, DROP, CREATE y/o	Código del componente		R	BAJO	0							
		QA10C-21.004		Campos con TITLE	No están permitidos los campos con TITLE en la generación de la ddl .	Código del componente		E	BAJO	0							
		QA10C-21.006		Accesos únicos a tabla	Accesos únicos a tabla. Los datos necesitados de una tabla deben obtenerse mediante un único acceso a ésta. Evitar reiteradas consultas sobre una misma tabla para recuperar diferentes atributos de un registro, los datos	Código del componente		R	BAJO	0							
		QA10C-21.007		Tablas temporales	**Tablas temporales.** No está permitido el uso de sentencias SQL que incluyan tablas temporales definidas mediante la cláusula 'AS' . Su uso se limita a la valoración	Código del componente		E	MODERADO	1							
		QA10C-21.010		Uso de SELECT *	No está permitido el **uso de `SELECT *`**. Su uso no independiza al módulo de posibles alteraciones de la tabla como añadir columnas, y por tanto no se justifica frente al	Código del componente		E	BAJO	1							
		QA10C-21.012		Uso de UPDATES	Los updates, **por defecto,** no están permitidos y en caso de necesitar ejecutarse, será una vez Adm. Teradata haya estudiado cada caso concreto y lo haya autorizado	Código del componente		E	MODERADO	1							
		QA10C-21.018			Evitar la manipulación de cadenas y el uso de expresiones aritméticas en la parte derecha de la claúsula WHERE en	Código del componente		E	MODERADO	0							
		QA10C-21.019		Uso de SAMPLE	En la medida en que no se quiera seleccionar todos los registros de la tabla, sino tan solo tomar una muestra aleatoria de los datos se debe de utilizar SAMPLE (o top para muestra no aleatoria)	Código del componente		E	MODERADO	1							
		QA10C-21.022		Uso de NOT EXISTIS	Usar preferentemente la cláusula NOT EXISTS en lugar de NOT IN.	Código del componente		R	BAJO	0							
		QA10C-21.023		Uso de Procedimientos Almacenados	En TERADATA no está permitido el uso de procedimientos almacenados. Únicamente cuando sea necesario el úso de una funcionalidad del gestor que no pueda cubrirse por otro	Código del componente		E	MODERADO	0							
		QA10C-21.024		Factor SKEW	Se recomienda que el factor de Skew (factor de distribución de los datos por índice) no supere en lo posible el 20% si el tamaño de la tabla supera el Gb. Entre 200 Mb. y 1 Gb. no debe superar el 30%.			E	BAJO								
		QA10C-21.025		Actualización de campos incluidos en el índice primario	No está permitido la actualización de campos que formen parte del índice primario de una tabla. En su lugar se debe	Código del componente		E	MODERADO	0							
		QA10C-21.027		Compresión de campos	No comprimir aquellos valores susceptibles de ser cambiados o substituidos en el futuro, es decir, que si en un campo, tenemos los códigos de algún producto, que en el momento de la compresión de datos eran en su gran mayoría XXXXX pero que en unos meses ese código pueda cambiar a YYYYY , el compress no solo queda obsoleto, sino que además, ocupa mas espacio (Teradata se reserva bits por cada compress para indexar los datos comprimidos de modo que si comprimes un campo que no existe en vez de reducir espacio utilizas estos bits) Si se comprimen esta clase de datos, habréis de modificar la definición de la tabla en función del tipo de dato insertado.	Código del componente		E	BAJO	0							
		QA10C-21.028		Compresión de fechas	No comprimir fechas a menos que sean genéricas o por defecto (como podría ser una tipo 31-12-2999) y que estas se den en su gran mayoría, ya que la fecha es un campo que se modifica continuamente y no podemos tomar como bueno, el valor que en el momento de la compresión apareciera. En un futuro este valor podría desaparecer al ser substituido por las fechas venideras.	Código del componente		E	BAJO	0							
		QA10C-21.029		Compresión de campos poco repetidos	No comprimir aquellos campos que no tengan un porcentaje alto respecto a los registros totales de la tabla. Es decir, no es optimo comprimir los valores del 1 al 120 de un campo, si por ejemplo en una tabla de 20.000 registros el valor 3 aparece 15.000 veces y el resto de 100 a 500 veces. Comprimir aquellos valores que tengan una repetición grande.	Código del componente		E	BAJO	0							
		QA10C-21.031		Insercione y/o actualizaciones directas	Se prohiben inserciones y/o actualizaciones directas sobre tablas que no sean de la propia aplicación, en su lugar utilizar las rutinas propias de cada aplicación para realizar	Código del componente		B	ALTO	0							
		QA10C-21.036		Sesiones abiertas	No se debe dejar una sesión abierta en el sistema con un proceso en ejecución	Código del componente		E	MODERADO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-21.038		Uso de técnica de PUSHDOWN	Se recomienda utilizar la técnica de Pushdown para no tener que bajarse todos los datos de la base de datos y tratarlo con Powercenter, sino que se obtenga el resultado deseado mediante querys a la misma, con ello se aumenta la eficiencia en el proceso y se reduce el tiempo de ejecución.	Código del componente		R	BAJO								
		QA10C-21.039		No abortar procesos que provoquen rollback	Se debe evitar abortar un proceso que pueda provocar rollback en el sistema.	Código del componente		E	BAJO								
		QA10C-21.040		Ejecución de nuevas sentencias	Siempre que se ejecute una nueva sentencia deberá analizarse el plan de ejecución (explain) para evaluar el posible impacto sobre el sistema.	Código del componente		E	MODERADO								
		QA10C-21.041		Considerar un límite máximo de pasos en el explain. **VALOR máximo 100 pasos**	El plan de ejecución de una sentencia SQL devuelto por el EXPLAIN permite evaluar a alto nivel la existencia de estadísticas apropiadas (high confidence) , la corrección de los cruces entre tablas y eventualmente su skew. No es admisible un plan de ejecución que anticipe miles de	Código del componente		E	MODERADO								
		QA10C-21.042		Sentencia de QueryBand	Toda sentencia ejecutada en un entorno productivo que provenga de un producto debe llevar asociada una sentencia de QueryBand.	Código del componente		E	MODERADO								
		QA10C-21.043		Opción "In Database"	Siempre que se trabaje a través de un producto y sea posible debería emplearse la opción 'In Database' para mejorar la eficiencia de los procesos.	Código del componente		E	BAJO								
		QA10C-21.044		Comprimir tablas grandes	En lo posible comprimir toda tabla superior a 1 Gb. Es recomendable la compresión del mayor porcentaje de tablas posible y una revisión periódica.	Código del componente		E	BAJO								
		QA10C-21.045		Estadísticas de la Partición	Tomar estadísticas de la columna (partition)	Código del componente		E	MODERADO								
		QA10C-21.046		Definición de la historicidad y revisión de la partición	Revisión periódica de la validez de la definición de la partición, teniendo en cuenta la historicidad almacenada en la tabla. Inclusión de 'NO RANGE', UNKNOWN' para prever la inserción de valores fuera de partición definida (durante un intervalo reducido de tiempo). Ajustar la implementación de la partición histórica (EACH INTERVAL '1' DAY, EACH INTERVAL '1' MONTH) al tipo de tabla (diaria, mensual) si aplica.	Código del componente		B	ALTO								
		QA10C-21.047		Mantenimiento/Backup de la tabla	Si es posible, utilizar la capacidad de inserción y borrado de filas basada en las particiones.	Código del componente		E	BAJO								
		QA10C-21.048		Toma de estadísticas	En un primer momento debe considerarse la toma de estadísticas de todas las tuplas presentes en los join del modelo y de sus componentes individuales (columnas). Estudiar idoneidad de toma en índices, tuplas usadas en join, columnas con valores 'where' , mal skew o alta repetición.	Código del componente		E	MODERADO								
		QA10C-21.049		Recolección de estadísticas	Cursar el mecanismo interno de recolección de estadísticas antes de la validación del modelo físico u optimización de la explotación.	Código del componente		B	ALTO								
		QA10C-21.050		Utilización del Secondary Index	Debe demostrarse la utilización del secondary index propuesto y mostrarse el uso de recursos para su creación.	Código del componente		E	BAJO								
		QA10C-21.051		Utilización del Join Index	Debe demostrarse la utilización del join index propuesto.	Código del componente		E	MODERADO								
		QA10C-21.052		Dependencias en el mantenimiento de Join Index	Tener en cuenta los aspectos de mantenimiento provocados por los mantenimientos de las tablas base de tal manera que se gestione correctamente el proceso de carga.	Código del componente		R	BAJO								
		SS.DD. - TERADATA															
	QA10C-22	**Arquitectura SPRING - CORE**															
		QA10C-22.001	QAC-0290	Uso de Ids y una nomenclatura correcta en la definición de los beans	Para detectar los incumplimientos se recorrerá dentro del ApplicationContext.xml, todos los elementos del tipo bean y generará una violación si se cumple alguno de los siguientes casos: - ID del bean = null, - ID del bean no se llama igual que la CLASS del Bean (comparación se realiza sobre la clase sin paquete y con él). También hay que tener en cuenta que en la comparación no se distingue entre minúsculas-mayúsculas. Ejemplo: El ID de un Bean de una instancia de la clase de InvasorDAO deberá ser invasorDAO: <bean id="invasorDAO" class="es.sadiel.plantilla.dao.InvasorDAO"> <property name="dataSource" ref="dataSource" /> </bean> Beneficios: Se fomenta el usar convenciones de nomenclatura, ya que usar una nomenclatura clara, descriptiva y consistente en todo el proyecto es de gran ayuda para que los			E	BAJO	1	1				1		
		QA10C-22.002	QAC-0291	Ficheros de configuración del contexto divididos por ambito.	No debemos tener un número de beans definidos en el archivo xml demasiado grande ya que el archivo se haría inmanejable. Se recomienda que los distintos beans estén configurados en archivos distintos según el ámbito al que aplican. Esto se puede hacer de una manara clasica mediante imports. <import resource="classpath:spring/beans-transactional.xml" /> <import resource="classpath:spring/beans-mails.xml" /> <import resource="classpath:spring/beans-i18n.xml" /> O mediante un patron de configuración en el web.xml. <context-param> <param-name>contextConfigLocation</param-name> <param-value>classpath*:META-INF/spring/applicationContext*.xml </param-value> </context-param>			E	BAJO	1					1		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-22.003	QAC-0292	No abusar de la inyección de dependencias.	No se debe usar la inyección de dependencias en exceso: - Los beans definidos no deben contener muchas dependencias, se establece un valor máximo por defecto de 10 dependencias por bean. Ejemplo: Saltará la violación si algún bean contiene más de 10 dependencias hacia otros beans. Identificamos cada dependencia si <bean id="invasorDAO" class="es.sadiel.plantilla.dao.InvasorDAO"> <property name="dataSource" ref="dataSource" /> … <property name="dataSource" ref="dataSource" /> </bean> Beneficios: El ApplicationContext de Spring puede crear los objetos java que sean necesarios, pero no todos los objetos deberían ser creados a través del motor de inyección de dependencias. Los archivos de configuración pueden sobrecargarse mucho si no se controla bien el crecimiento de los beans definidos. Sobreutilizar la inyección de dependencias puede hacer el archivo de configuración muy complicado y sobrecargado. Cómo solucionarlo: Reducir las dependencias de cada uno de los beans que contiene el ApplicationContext.xml a un número menor del			E	BAJO	1	1				1		
		QA10C-22.004	QAC-0293	Usar nomenclatura abreviada en la definición de beans	Comprobar que se están usando formas no abreviadas a la hora de definir los beans. <property name="nombreInvasor"> <value>Zim</value> </property> ó <constructor-arg> <ref bean="invasorDAO"> </constructor-arg> Ejemplo: A continuación se muestra la forma no abreviada: <bean id="invasorService" class="com.dosideas.spring.InvasorService"> <property name="nombreInvasor"> <value>Zim</value> </property> <constructor-arg> <ref bean="invasorDAO"> </constructor-arg> </bean> Puede ser reescrito de forma abreviada: <bean id="invasorService" class="com.dosideas.spring.InvasorService"> <property name="nombreInvasor" value="Zim"/> <constructor-arg ref="invasorDAO"/> </bean> Beneficios: Dentro del archivo de configuración, las formas abreviadas son más fáciles de leer, dado que trasforma el valor de los			R	BAJO	1	1						
		QA10C-22.005	QAC-0294	No usar la propiedad autowire en la definición de los beans	No usar la propiedad autowire dentro de los beans definidos en el applicationContext.xml Ejemplo: A continuación se muestra un ejemplo de uso de autowiring, donde los nombres de las propiedades de la clase InvasorService son usadas para buscar instancias de beans en el contenedor: <bean id="invasorService" class="com.dosideas.spring.InvasorService" autowire="byName"/> Beneficios: El uso de Autowiring nos salva de tipear mucho código. De todas formas, no es recomendado su uso, nunca deberíamos utilizar Autowiring en una aplicación real porque perjudicamos la legibilidad de nuestro archivo de			E	BAJO	1	1						
		QA10C-22.006	QAC-0295	Usar type en vez de index para los argumentos del constructor en los beans.	Spring permite utilizar índices (que comienzan en cero) para resolver el problema de ambigüedad cuando un constructor tiene mas de un argumento del mismo tipo, o cuando se utiliza el atributo value para asignar el valor. La regla es no utilizar index para los argumentos del constructor Ejemplo: Uso de index: <bean id="InvasorService" class="com.dosideas.spring.InvazorService"> <constructor-arg index="0" value="Zim"/> <constructor-arg index="1" value="100"/> </bean> Beneficios: Utilizar index es mas sencillo, ya que tipeamos menos, pero es mas propenso a errores y mucho mas difícil de leer que si utilizamos type. Se debería utilizar index solamente cuando se presenta una ambigüedad en los argumentos del constructor. Cómo solucionarlo: Utilizando type en vez de index, como se muestra a continuación se solucionaría el ejemplo anterior: <bean id="invasorService" class="com.dosideas.spring.InvasorService"> <constructor-arg type="java.lang.String" value="Zim"/> <constructor-arg type="int" value="100"/> </bean>			E	BAJO	1	1	1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-22.007	QAC-0296	Definir y Reutilizar AbstractBeans	Una definición de un bean hijo puede heredar la información de configuración de sus padres, los cuales sirven como un "template" para los beans hijos Todo lo que se necesita hacer es especificar abstract=true en el Bean padre y referenciar a dicho Bean desde los Beans hijos. En el aplicationContext se recomienda que un gran número de beans (valor mínimo del 50 % de beans), hereden de clases abstractas. Hay que pensar que la mayoría de beans definidos en una aplicación JEE son Factorias, DAOs y Servicios que fácilmente pueden tener su clase padre. Ejemplo: <bean id="abstractInvasorService" abstract="true" class="com.dosideas.spring.AbstractInvasorService"> <property name="nombreInvasor" value="Zim"/> </bean> <bean id="InvasorService" parent="abstractInvasorService" class="com.dosideas.spring.InvasorService"> <property name="robotAsignado" value="Gir"/> </bean> El bean InvasorService hereda el valor Zim para la propiedad nombreInvasor del bean abstractInvasorService. Una aclaración: Si no se especifica el nombre de una clase en un factory method en la definición de un bean, este bean es implícitamente abstracto. Beneficios:			E	BAJO	1	1						
		QA10C-22.008	QAC-0297	Unica Descripcion	Agregar una sola descripción en archivo de configuración, y que sea en el encabezado Es preferible usar ids y nombres descriptivos en vez de utilizar comentarios en los archivos de configuración. Ejemplo: <beans> <description> Este archivo define los Invasores que atacaran los disintos planetas. Depende de RobotsServices.xml, el cual provee los servicios de los robots que cada invasor tiene asignado.... </description> ... </beans> Beneficios: Con esta práctica conseguiremos mejor mantenibilidad al documentar mejor nuestros archivos de configuración. Cómo solucionarlo:			R	BAJO	1	1						
		QA10C-22.009	QAC-0298	Usar Dependency Check en Desarrollo	Usar dependency-check durante la fase de desarrollo. Se puede asignar al atributo dependency-check en la definición de un bean a un valor distinto del default none, como puede ser: simple, objects, o all. Hay que comprobar que no pase a producción este atributo. Ejemplo: <bean id="invasorService" class="com.dosideas.spring.InvasorService" dependency-check="objects"> <property name="nombreInvasor" value="Zim"/> <constructor-arg ref="invasorDAO"/> </bean> En este ejemplo, el contendedor se asegurará que las propiedades de InvasorService que no sean primitivos o colecciones estén asignadas. Se puede configurar el chequeo de dependencias para que controle que todas las propiedades del bean estén asignadas, pero esto es necesario en raras ocasiones, ya que puede que haya propiedades que no necesiten ser asignadas. Beneficios: Con esto el contenedor puede realizar validaciones de dependencias. Esto es útil cuando todas las propiedades (o cierto grupo de propiedades) de un bean deben ser asignadas explícitamente. También nos aseguremos que estos atributos no pasen a producción, para no perder eficiencia. Cómo solucionarlo: En fase de desarrollo, poner la propiedad dependency-			E	BAJO	1	1	1						
		QA10C-22.010	QAC-0299	No Usar Dependency Check en producción	No se debe pasar a producción con el atributo dependency-check habilitado ya que esto provocaría un aumento del tiempo de despliegue de la aplicación Pese a que los beneficios del atributo dependency-check en desarrollo son claros, se debe eliminar de producción			E	MODERADO	1	1		1		1		
		QA10C-22.011	QAC-0300	No abusar del ApplicationContext y del BeanFactory o no usarlos.	Intentar no usar o no abusar de acceder a los beans por medio del ApplicationContext o del BeanFactory. Spring por medio del IOC te permite inyectar los distintos beans dentro del applicationContext.xml donde sea necesario, salvo en casos muy concretos y justificados			E	BAJO	1	1	1			1		
		QA10C-22.012	QAC-0301	Usar ApplicationContext en vez de BeanFactory para acceder a los beans.	Utilizar ApplicationContext en vez de BeanFactory a la hora de cargar y enviar las definiciones de beans Ejemplo: Uso de BeanFactory (no recomendado): public static void main(String[] args) { BeanFactory beanFactory = new XmlBeanFactory(new ClassPathResource("applicationContext.xml")); ServicioRemoto servicio = beanFactory.getBean("servicioRemoto", ServicioRemoto.class); System.out.println("El valor es " + servicio.consultaDato()); } Beneficios: La principal diferencia entre ApplicationContext y BeanFactory, es la manera en la que cargan los beans. Hay que decir que por defecto todos los beans manejados por Spring son Singletons, solo existe una instancia de ellos por aplicación. BeanFactory crea los beans al momento que estos son pedidos (con la invocación de getBean). ApplicationContext trabaja de una forma mejor, ya que carga todos los singletons cuando se inicia el contexto, y de esta forma se asegura que estarán listos en el momento en el que sean solicitados. Además ApplicationContext ofrece más cosas: • Proporciona un medio para resolver mensajes de texto, incluyendo soporte para internacionalización (I18N) de estos mensajes.			E	BAJO	1	1	1						

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-22.013	QAC-0302	Fomentar Inyeccion dependencias setter antes que por constructor	Spring provee tres tipos de inyección de dependencias: inyección por constructor, por setter y por método. Se suelen usar las dos primeras: Spring nos permite además mezclar ambas formas. Se recomienda usar la inyección por setter a las demás. Si se usa la inyección por constructor, se debe revisar que no se supere un número máximo de parámetros (5). Ejemplo: En este ejemplo, el bean invasorService usa inyección por constructor, mientras que el bean robotService usa inyección por setter. <bean id="invasorService" class="com.dosideas.spring.InvasorService"> <constructor-arg ref="invasorDAO"/> </bean> <bean id="robotService" class="com.dosideas.spring.RobotService"> <property name="robotDAO" ref="robotDAO"> </bean> Beneficios: El equipo desarrollador de Spring sugiere usar inyección por setter, porque tener constructores con un número grande de argumentos puede ser engorroso, especialmente si algunos de los argumentos son opcionales. Los métodos setter también hacen que los objetos de esas clases sean más fáciles de reconfigurar o re-inyectar después. Se podría usar inyección de dependencias basada en constructor para las dependencias obligatorias y la inyección de dependencias basada en setter para las			E	BAJO	1	1	1					
		QA10C-22.014	QAC-0303	Usar Terracotta o Infinispan como cache Distribuida	Dentro de las librerías de cache distribuidas que existen, se recomienda por su gran rendimiento e integración con Spring el uso de terracotta e infinispan (JBOSS). NOTA: Esto es importante sobre todo si nos encontramos en una arquitectura enfocada a las librerías opensource, si nos encontramos en un cliente en el que el soporte es lo mas importante, podríamos optar también por Coherence (ORACLE) Una buena forma de integrarse sería una combinación con Ehcache. Ejemplo de configuración ehcache + terracotta. Fácilmente detectable buscando la etiqueta <terracotta> en el archivo ehcache.xml: <cache name="sampleTerracottaCache" maxElementsInMemory="1000" eternal="false" timeToIdleSeconds="3600" timeToLiveSeconds="1800" overflowToDisk="false"> <terracotta/> </cache> Infinispan se integraría con Spring por medio del Infinispan Cache Manager. Podemos validarlo en desarrollo buscando la dependencia en el pom.xml			E	BAJO	1			1	1			
		QA10C-22.015	QAC-0304	Utilizar algún mecanismo de cacheo	Comprobar que se utiliza algún servicio de cacheo de Spring. Por ejemplo para verificar que se utiliza Coherence buscaríamos: <coherence:caching a cada bean definido de tipo DAO. Un ejemplo de uso de cache es: <bean id="customerDaoTarget" class="org.springmodules.cache.samples.dao.CustomerDao" /> </bean> <coherence:proxy id="customerDao" refId="customerDaoTarget"> <coherence:caching methodName="load" cacheName="customerCache" /> <coherence:flushing methodName="update" cacheNames="customerCache"/> </coherence:proxy> Otro ejemplo para ehcache, se crea el fichero ehcache.xml en classpath.: <ehcache> <defaultCache maxElementsInMemory="500" eternal="true" overflowToDisk="false" memoryStoreEvictionPolicy="LFU"/> <cache name="contactCache"			E	MODERADO	1			1	1			
		QA10C-22.016	QAC-0305	Usar patrón Singleton para la definición de beans	Por norma general, no crear beans distintos de Singleton. Ejemplo: Por defecto todos los beans definidos en Spring son singleton. Sin embargo se pueden crear distintos de singleton poniendo la propiedad "singleton" a false. El inconveniente es que esto crea cada vez una instancia al llamar al getBean, no es recomendable usar una configuración distinta de singleton y con más motivo si se usa una base de datos o conexiones a red. Cómo solucionarlo:			E	MODERADO	1	1	1					
		QA10C-22.017	QAC-0306	Evitar usar Interfaces de Spring	Recorrer código revisando que no se usen interfaces de Spring: NameAware, BeanFactoryAware y ApplicationContextAware Ejemplo: Cualquier clase que implemente estas clases, o método que utilice estas interfaces. Beneficios: Se recomienda siempre que sea posible no usar las interfaces de Spring, para no vincular el código al fabricante (principal ventaja de Spring).			E	BAJO	1	1						

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-22.018	QAC-0307	Separar detalles despliegue del application context	Intentar separar los detalles del despliegue fuera del application context en otro xml o fichero de properties. Se pueden especificar los detalles específicos en un fichero de properties usando PropertyPlaceHolderConfigurer o a través de JNDI. Ejemplo: A continuación se muestran los 2 ejemplos de buen uso: 1) Ejemplo fichero de properties: <bean id="propertyConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"> <property name="location"><value>datasource.properties</value></property> </bean> <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"> <property name="url"><value>${database.url}</value></property> <property name="driverClassName"><value>${database.driver}</value></property> <property name="username"><value>${database.user}</value></property> <property name="password"><value>${database.password}</value></property> </bean>			E	BAJO	1	1	1	1	1	1		
		QA10C-22.019	QAC-0308	No duplicar librerias	Es importante que no se dupliquen librerías en nuestro WEB-INF/lib porque podría ocasionar conflictos a la hora del despliegue o un comportamiento defectuoso del aplicativo. Por ejemplo cuando desarrollamos una aplicación con maven, al declarar algunas librerías no solo nos incluye las librerías que le indicamos sino otras de las que depende, lo que puede provocar que nos encontremos con librerías repetidas de igual o distinta versión.			E	MODERADO	1	1	1			1		
		QA10C-22.020	QAC-0309	Servicios heredan de clase comun	En una aplicación JEE con un diseño tipico MVC, la capa de servicios es muy importante y supone gran parte del código de nuestra aplicación. Normalmente estos servicios contienen recursos que pueden ser inicializados y configurados en una clase padre que declararemos como abstracta en su definición en nuestro applicationContext.xml. Ejemplo: public abstract AbstractServicio { …// métodos comunes a todos los servicios } public Servicio extends AbstractServicio { … }			E	BAJO	1	1	1					
	QA10C-23	**Arquitectura SPRING - AOP**															
		QA10C-23.001	QAC-0310	Usar Spring AOP para aspectos que solo se encuentran en Beans de Spring	Spring AOP es más simple que el uso de AspectJ completo ya que no hay necesidad de introducir el compilador de AspectJ en el desarrollo y construcción de procesos. Si sólo necesitamos incluir aspecto en beans bajo el contexto de spring, Spring AOP es la elección correcta. Esta regla será opcional para desarrollos en los que se conozca que solo beans de Spring van a estar afectados.			E	BAJO	1			1				
		QA10C-23.002	QAC-0311	Usar AspectJ_Annotation	Habilitar la implementación de Spring AOP con AspectJ annotations, Beneficio: La implementación de Spring AOP mediante XML no responde al principio de encapsulamiento que indica que cada componente de nuestro sistema debe encontrarse centralizado en una única entidad. Además, provee funcionalidad más limitada que annotations, por ejemplo en la declaración de pointcuts, en donde en xml solamente está soportado el modelo de instanciación de aspecto "singleton". Ejemplo En este ejemplo se observa la diferencia entre la definición de pointcuts en el estilo @AspectJ y en el estilo XML. En el estilo @AspectJ sería: @Pointcut(execution(* get*())) public void propertyAccess() {} @Pointcut(execution(org.xyz.Account+ *(..)) public void operationReturningAnAccount() {} @Pointcut(propertyAccess() && operationReturningAnAccount()) public void accountPropertyAccess() {} En el estilo XML solo se podría crear los dos primeros, dada sus limitaciones:			E	BAJO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-23.003	QAC-0312	Usar expresiones regulares en los pointcuts	En lugar de utilizar la definición completa de los métodos pointcuts, es recomendable el uso de expresiones regulares en la definición de interceptores. Esta regla será opcional ya que en algunos casos se podría necesitar la definición completa de los métodos. Spring soporta 3 tipos de metodos regex: • org.springframework.aop.support.JdkRegexpMethodPointcut : expresión regular java 1.4. • org.springframework.aop.support.Perl5RegexpMethodPointcut : expresión regular Perl5 • org.springframework.aop.support.RegexpMethodPointcutAdvisor : clase por defecto para métodos regexp pointcuts que tienen advices. Por defecto, JdkRegexpMethodPointcut será usado con JDK 1.4 o superior, utilizando Perl5RegexpMethodPointcut en JDK 1.3. Beneficio No es necesario especificar el nombre del método completo, se puede utilizar expresiones regulares para decir que se intercepten los métodos que cumplan un determinado patrón. Ejemplo <bean id="crudInterceptor" class="com.mycompany.CrudInterceptor"/>			E	BAJO	1					1		
		QA10C-23.004	QAC-0313	Usar Anotaciones con Java 5 o superior	La implementación de AOP en Spring puede realizarse de dos formas diferentes: • mediante la ultilización de ApectJ annotations. • mediante el archivo de configuración de Spring (XML). Si la versión JDK es anterior a la 1.5, la opción recomendada para la implementación de AOP es mediante un fichero XML. En cualquier otro caso, se debe utilizar AspectJ Annotations. Beneficio: La configuración mediante XML es recomendada si se utiliza AOP para implementar la configuración del sistema, debido a que para modificar la configuración solamente se modifica el archivo XML. En cualquier otro caso es conveniente usar annotations, ya que la utilización de AOP mediante XML no responde al principio de encapsulamiento que indica que cada componente de nuestro sistema debe encontrarse centralizado en una única entidad. Ejemplo: Ejemplo mediante la utilización de fichero XML: <bean id="crudInterceptor" class="com.mycompany.CrudInterceptor"/> <bean id="crud" class="org.springframework.aop.support.RegexpMethodPointcutAdvisor"> <property name="advice">			E	BAJO	1							
		QA10C-23.005	QAC-0314	Usar pointcuts estáticos	Se recomienda la utilización de pointcut estáticos sobre los dinámicos. Se puede validar que las clases definidas como pointcuts hereden de StaticMethodMatcherPointcut Beneficio Los pointcut estáticos pueden ser evaluados cuando un proxy es creado, no pueden ser modificados, los dinámicos dependen de la información en tiempo de ejecución, los dinámicos son más lentos que los estáticos y permiten menos potencial de optimización. Los pointcut dinámicos son más costosos de evaluar que los estáticos. Toman en cuenta los argumentos del método, así como información estática. Esto significa que deben ser evaluados cada vez que se invoca un metodo, el resultado no puede ser cacheado, los argumentos varían. Cómo solucionarlo StaticMethodMatcherPointcut es la clase base para poder crear pointcuts estáticos. Ejemplo: class TestStaticPointcut extends			E	MODERADO	1			1		1		
	QA10C-24	**Arquitectura SPRING - ACCESO A DATOS y ORM**																
		QA10C-24.001	QAC-0315	Manejar Excepciones DataAccessException	Se debe manejar la excepción DataAccessException Ejemplo: En DAO: } catch (DataAccessException e) { throw MyAppException... } Beneficios: Spring da una consistente jerarquía de excepciones en su nivel DAO. Todas las SQL tanto como excepciones DAO están bajo DataAccessException. Manejando está excepción se puede facilmente obtener un log de errores. Para ayudar a manejar las excepciones también se puede usar la clase de Spring org.springframework.web.servlet.handler.SimpleMappingExceptionResolver			E	BAJO	1		1	1		1		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-24.002	QAC-0316	Usar Transaccionalidad Declarativa	Para manejar la transacción de forma declarativa por Spring, debemos englobar nuestros servicios con un <aop:pointcut y asociarle un <tx:advice Ejemplo: < aop:config > < aop:pointcut id="fooServiceOperation" expression="execution(* org.mipaquete.core.domain.logic.Facade.*(..))"/ > < aop:advisor advice-ref="txAdvice" pointcut-ref="facade"/ > < /aop:config> < tx:advice id="txAdvice" transaction-manager="txManager" > < tx:attributes > < tx:method name="get*" propagation="PROPAGATION_REQUIRED" read-only="true"/ > < tx:method name="*" propagation="PROPAGATION_REQUIRED"/ > < /tx:attributes > < /tx:advice> Beneficios: Spring proporciona transacción declarativa y programática, la potencia que tiene la declarativa gracias a su configuracion en el Application Context hace recomendable el uso de esta. Cómo solucionarlo:			E	BAJO	1		1					
		QA10C-24.003	QAC-0317	Configurar los atributos de la transaccionabilidad	Se deben definir, dentro de las transacciones declarativas, las propiedades propagation e isolation. De todas maneras es importante indicar que esta regla se puede desactivar en el caso de que estemos usando la configuración por defecto de transacccionabilidad porque es la que necesitamos. Mas información en: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html Ejemplo: <tx:advice id="defaultTxAdvice"> <tx:attributes> <tx:method name="get*" read-only="true" propagation="REQUIRED" isolation="DEFAULT" /> <tx:method name="*"/> </tx:attributes> </tx:advice> <tx:advice id="noTxAdvice"> <tx:attributes> <tx:method name="*" propagation="NEVER"/> </tx:attributes> </tx:advice> Beneficios: Spring tiene diferentes niveles de aislamiento y			E	BAJO	1		1					
		QA10C-24.004	QAC-0318	Usar JtaTransactionManager para definir la transaccionabilidad de nuestros servicios.	Se deben definir las propiedades propagation e isolation en la declaración de nuestra transaccionalidad, ya se por medio de la configuración de nuestro xml o por Anotaciones. Más información en: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html			E	BAJO	1		1					
		QA10C-24.005	QAC-0319	Usar un pool de conexiones como por ejemplo el de Apache Commons	Se debe buscar que se usa algún pool de conexiones conocido como el de apache commons en la configuración del datasource. Se puede enviar a esta regla como parámetro la lista de datasources validos para la configuración. Ejemplo de un datasource con el de apache commons. <data-source type="org.apache.commons.dbcp.BasicDataSource"> <set-property property="driverClassname" value="com.mysql.jdbc.Driver"/> <set-property property="url" value="jdbc:mysql://localhost:8080/sksoftwares"/> <set-property property="username" value="root"/> <set-property property="password" value="admin"/> </data-source> Beneficios: En la configuración del DataSource de Spring se suele usar la clase: org.springframework.jdbc.datasource.DriverManagerDataSource. Es recomendado usar un pool de conexiones, lo cual se			E	MODERADO	1	1	1	1	1			
		QA10C-24.006	QAC-0320	Usar Hibernate como ORM	Se recomienda el uso de hibernate como ORM ya que es el mas afianzado por la comunidad de programadores, da un rendimiento optimo y facilita bastante el desarrollo de aplicaciones con Spring. Para validar esta regla bastara con buscar en el applicationContext.xml alguna clase de hibernate como LocalSessionFactoryBean o HibernateTemplate. <bean id="sessionFactory" class="org.springframework.orm.hibernate3.LocalSessionF			E	BAJO	1	1	1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-24.007	QAC-0321	DAOs hereden de clase Comun	Todos los DAOs deben usar una clase común. Muy recomendable usar patrón de diseño Generic DAO. Ejemplo: public class GenericDaoHibernateImpl <T, PK extends Serializable> implements GenericDao<T, PK>, FinderExecutor { private Class<T> type; public GenericDaoHibernateImpl(Class<T> type) { this.type = type; } public PK create(T o) { return (PK) getSession().save(o); } public T read(PK id) { return (T) getSession().get(type, id); } public void update(T o) { getSession().update(o); } public void delete(T o) { getSession().delete(o); } // Not showing implementations of getSession() and setSessionFactory() } En applicationContext.xml: <bean id="personDao" class="genericdao.impl.GenericDaoHibernateImpl"> <constructor-arg> <value>genericdaotest.domain.Person</value>			E	MODERADO	1	1	1					
		QA10C-24.008	QAC-0322	Usar Patrón Dao o Repository para el acceso a BBDD	Usar para el acceso a BBDD alguno de los dos patrones reconocidos para este tipo de acceso. Todas las operaciones que requieren interactuar con la BBDD deben estar aisladas en clases que implementen alguno de estos dos Patrones. Es importante indicar que los patrones Dao y Repository no son completamente iguales, el patrón Repository nació para dar soporte al DDD (Domain Driven Designer) de manera que los objetos de Dominio no tuvieran que implementar la capa de persistencia.			E	MODERADO	1	1	1					
		QA10C-24.009	QAC-0323	Un datasource por cada tipo de conexión a la BBDD	Comprobar que existe un bean de tipo datasource para cada una de las diferentes conexiones a BBDD. Beneficio: Puede ser reutilizable. Ejemplo: <!-- DataSource Definition --> <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close"> <property name="driverClassName"> <value>org.postgresql.Driver</value> </property> <property name="url"> <value>jdbc:postgresql://ip_server:port/portal</value> </property> <property name="username"> <value>username</value> </property> <property name="password"> <value>password</value> </property> </bean>			B	ALTO	1	1	1		1	1		
		QA10C-24.010	QAC-0324	Controlar excepciones de Base de Datos	Descripción del incumplimiento: Utilización de un componente "jdbcExceptionTranslator", para capturar todas las excepciones que se producen en el acceso a la fuente de datos. Beneficio: Traduce las excepciones a un lenguaje común, con independencia de la BD utilizada. Ejemplo: <!-- Spring Data Access Exception Translator Defintion --> <bean id="jdbcExceptionTranslator" class="org.springframework.jdbc.support.SQLErrorCodeSQLExceptionTranslator">			E	BAJO	1		1					
		QA10C-24.011	QAC-0325	Usar Spring template para el acceso a BBDD	Utilizar Templates de spring para el acceso a BD, utilizar el apropiado a cada tipo de tecnologia: hibernate, jpa, ibatis, jdo… Ejemplo: <!-- Hibernate Template Definition --> <bean id="hibernateTemplate" class="org.springframework.orm.hibernate3.HibernateTemplate"> <property name="sessionFactory"> <ref bean="sessionFactory" /> </property> <property name="jdbcExceptionTranslator"> <ref bean="jdbcExceptionTranslator" />			E	MODERADO	1		1		1	1		
		QA10C-24.012	QAC-0326	Datasources configurados en el servidor de aplicaciones	Es importante definir los datasources dentro del servidor de aplicaciones y acceder a los mismos por jndi, de esta manera cualquier cambio en el acceso a bd, puertos, pool de conexiones, etc … no nos obliga a modificar nuestro war y queda en manos de los administradores del servidor de aplicaciones y los DBAs. Ejemplo:			E	MODERADO	1	1	1			1		
		QA10C-24.013	QAC-0327	Separar Capas de acceso a datos de la capa de servicios	Es recomendable realizar la separación de Dao y servicio. Beneficio: Separar la capa de acceso a datos de los servicios de negocio. Ejemplo: <bean id="conjuntoUmbralesDao" class="es.bull.model.dao.impl.ConjuntoUmbralesDaoHibernateImpl"> <property name="hibernateTemplate"> <ref bean="hibernateTemplate" /> </property> </bean> <bean id="portalService" class="es.bull.service.impl.PortalServiceImpl"> <property name="atributoDao" ref="atributoDao" /> </bean>			E	ALTO	1	1	1		1	1		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-24.014	QAC-0328	HibernateTemplate inyectado en el DAO o el Repositorio	Es recomendable que los Dao y los repositorios utilicen los templates definidos para cada una de las tecnologías, para ello hay que comprobar que en la definición del Dao se le inyecta la template correspondiente en la property. También hay que tener en cuenta que los Daos o repositorios podrían estar heredando de una clase padre que fuera la que tenga inyectado el template. Ejemplo: <bean id="contactoDao" class="es.bull.model.dao.impl.ContactoDaoHibernateImpl"> <property name="hibernateTemplate" ref="hibernateTemplate" /> </bean> <!-- Hibernate Template Definition --> <bean id="hibernateTemplate" class="org.springframework.orm.hibernate3.HibernateTemplate"> <property name="sessionFactory">			E	MODERADO	1				1	1		
		QA10C-24.015	QAC-0329	Daos o Repositorios inyectados en los Servicios	Comprobar que los Bean con "id = service" tengan inyectados los DAOs o los repositorios que utilizan. Ejemplo: <!-- User Service Defintion --> <bean id="portalService" class="es.bull.service.impl.PortalQAServiceImpl"> <property name="atributoDao" ref="atributoDao" /> <property name="certificacionDao" ref="certificacionDao" /> <property name="conjuntoUmbralesDao" ref="conjuntoUmbralesDao" /> <property name="contactoDao" ref="contactoDao" />			E	MODERADO	1		1		1			
		QA10C-24.016	QAC-0330	Daos o Repositorios inyectados.	Comprobar que no existen DAOs o Repositorios sin inyectar por algún servicio. En esta regla buscaríamos en todos los beans de Spring que cumplan el patrón asignado para un DAO o un repositorio, y todos aquellos que no estén declarados como abstractos deberán estar inyectados en algún Servicio.			E	BAJO	1		1		1	1		
		QA10C-24.017	QAC-0331	Definir un transactionManager	Comprobar que los servicios están envueltos en una capa transaccional. Para ello debe existir un bean TransactionManager (encargado del manejo de transacciones) definido. Existiendo las siguientes posibilidades: DataSourceTransactionManager, HibernateTransactionManager, JdoTransactionManager, JmsTransactionManager, JtaTransactionManager, PersistenceBrokerTransactionManager Ejemplo:			E	BAJO	1		1	1	1	1		
		QA10C-24.018	QAC-0332	Configurar JPA con transaccionalidad	JPA (Java Persistence Api) es el api mas conocido dentro del mundo JEE para interactuar con BBDD. Spring tiene una implementación muy eficiente de este api. Con JPA es muy facil incluir la transaccionalidad mediante la Anotación @Transactional Podemos configurarlo de la siguiente manera: <bean class="org.springframework.orm.jpa.JpaTransactionManager" id="transactionManager"> <property name="entityManagerFactory" ref="entityManagerFactory"/> </bean> <bean class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean" id="entityManagerFactory"> <property name="persistenceUnitName" value="persistenceUnit"/> <property name="dataSource" ref="dataSource"/> </bean> Se puede integrar a su vez muy fácil con hibernate configurando persistenteUnit de la siguiente manera: <?xml version="1.0" encoding="UTF-8" standalone="no"?> <persistence xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"			E	BAJO	1		1		1	1		
		QA10C-24.019	QAC-0333	Configurar Transaccionalidad con JTA	JTA es api que proporciona JEE para el manejo de transacciones en aplicaciones distribuidas, se configurara de la siguiente manera: <bean id="myJtaTransactionManager" class="org.springframework.beans.factory.config.MethodInvokingFactoryBean"> <property name="targetObject" ref="myTxManager" /> <property name="targetMethod" value="getTransactionManager" /> </bean> Donde myTxManager sera: <bean id="myTxManager" class="org.springframework.transaction.jta.JtaTransaction			E	BAJO	1		1		1	1		
		QA10C-24.020	QAC-0334	Misma implementación de ORM	Comprobar que se utiliza la misma implementación de ORM en los siguientes componentes: 1.TransactionManager 2.sessionFactory 3.Templates spring Ejemplo: <!-- Hibernate Transaction Manager Definition --> <bean id="transactionManager" class="org.springframework.orm.hibernate3.HibernateTransactionManager"> <!-- Hibernate session factory --> <bean id="sessionFactory"			E	BAJO	1		1	1				

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-24.021	QAC-0335	Usar la cache de hibernate para optimizar las consultas.	Usar la cache de hibernate para mejorar la velocidad de los accesos para entidades de solo lectura. Hibernate nos proporciona por defecto una cache que permite configurar el modo de acceso a las entidades que tiene mapeadas y acelerar las consultas. De esta manera podemos indicar, cual de estas entidades solo tienen acceso en modo lectura. <property name="entityCacheStrategies"> <props> <prop key="Projecto">read-only</prop> <prop key="Requisito">read-only</prop> <prop key="Tarea">read-write</prop>			E	BAJO	1			1				
		QA10C-24.022	QAC-0336	Utilizar hibernatetools para la creación de la base de datos, basandonos en mapeos o en anotaciones.	Utilizar hibernatetools para la creación de la base de datos, basándonos en mapeos o en anotaciones. Hibernate te proporciona varios tasks para lanzar con ant que te permiten crear la base de datos a partir de los mapeos de las entidades. Tiene la ventaja de que dependiendo del dialecto elegido, nos creara el schema para la BBDD que estemos usando. Se puede hacer con: <taskdef name="hibernatetool" classname="org.hibernate.tool.ant.HibernateToolTask"> <classpath> <fileset dir="${installDir}/resources/lib"> <include name="**/*.jar" /> </fileset> <path path="${installDir}/resources/classes" /> <path path="${classesDir}" /> </classpath> </taskdef> <hibernatetool destdir="${installDir}/generated"> <hbm2ddl export="true" update="false" drop="true" create="true" haltonerror="false" outputfilename="filename.ddl" /> </hibernatetool> O:			E	BAJO	1							
		QA10C-24.023	QAC-0337	Cargar Base de Datos con dbunit	Cargar Base de datos con dbunit, Dbunit proporciona una tarea ant, que nos permite cargar nuestra base de datos. Esta carga se realiza mediante ficheros xml, de manera muy sencilla y multi BBDD. <taskdef name="dbunit" classname="org.dbunit.ant.DbUnitTask"> <classpath> <fileset dir="${installDir}/resources/lib"> <include name="**/*.jar" /> </fileset> </classpath> </taskdef> <target name="insert-master-data" description="Inserts into the database the master information the system."> <dbunit driver="${hibernate.connection.driver_class}" url="${hibernate.connection.url}" userid="${hibernate.connection.username}" password="${hibernate.connection.password}"> <operation type="CLEAN_INSERT"			E	BAJO	1							
	QA10C-25	**Arquitectura SPRING - MVC**																
		QA10C-25.001	QAC-0338	Limitar el número de Front Controllers	Demasiados Controllers hacen la aplicación difícil de mantener y reusar. No usar páginas JSP como controladores. Como solucionarlo: Se validará que solo se haga uso de un FrontController, llamado DispatcherServlet, que es el servlet que actua como un gateway común a todas las peticiones web según el modelo MVC de Spring. Beneficios: * El DispatcherServlet interceptará todas las peticiones web de los diferentes clientes y los redirecciona a las controladores apropiados de cada página. * Solo habrá un solo punto de entrada en toda la aplicación (Control centralizado). * Diseño simplificado * Se promueve la reusabilidad			E	MODERADO	1	1	1	1				
		QA10C-25.002	QAC-0339	Application Controller	Dejar que el FrontController se encargue de tareas de Spring tales como invocar componentes de negocio o identificar y redireccionar las acciones a sus correspondientes vistas. Como solucionarlo: Se deberá usar un ApplicationController para desacoplar del FrontController la funcionalidad de Spring de la gestión de acciones y vistas. El FrontController deberá ser genérico y ligero. Beneficios: * Se mejorará la reusabilidad, cohesión y modularidad del código de la aplicación.			E	MODERADO	1	1	1			1		
		QA10C-25.003	QAC-0340	No abusar de controladores que solo redirijan a una JSP.	No abusar de controladores que solo redirijan a una JSP. Evitar que controladores basados en páginas JSP sean los responsables de invocar a beans de sesión que lleven a cabo alguna función de negocio. Como solucionarlo: Usar un controlador de página (PageController) por cada acción que se realice desde el cliente. Beneficios: * Estos controladores podrán ser usados por distintas peticiones URL, por lo que aporta mayor reusabilidad del			E	BAJO	1		1					
		QA10C-25.004	QAC-0341	Usar ContextObject	Usar objetos altamente acoplados al protocolo HTTP o al API del servlet, como por ejemplo objetos HttpServletRequest y HttpServletResponse. Como solucionarlo: Para no depender de ningún protocolo a la hora de compartir y encapsular datos se deberá hacer uso de objetos de contexto (ContextObject). Beneficios: * Podrán soportarse en distintos entornos y clientes. * Mayor reusabilidad de los objetos en cuestión.			E	BAJO	1	1	1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-25.005	QAC-0342	Código en las JSPs	Existencia de lógica de negocio y programación en la capa de presentación, por ejemplo, en páginas JSP. Como solucionarlo: Uso de ViewHelper para adaptar el modelo de datos con los componentes en la capa de presentación. Beneficios: * Separación de roles: Ayuda a diferenciar la capa de presentación de datos (autoría web) y la lógica de negocio (programación en Java). * Mayor sencillez de mantenimiento y ahorro de tiempo de desarrollo. Ejemplo válido: Uso de librerías JSTL en las páginas JSP, como por ejemplo: <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %> <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"			B	ALTO	1	1	1					
		QA10C-25.006	QAC-0343	Usar <context:component-scan> para la simplificar la configuración xml.	Usar <context:component-scan> para la simplificar la configuración xml. Si nuestra aplicación es muy grande, tener configurados en un xml todos los controladores podría hacer inmanejable el mismo. Si usamos el stereotype Controller y creamos nuestras clases con la anotación @Controller, será mucho mas fácil y limpia la configuración. <!-- The controllers are auto detected POJOs labeled with the @Controller annotation. -->			E	BAJO	1	1						
		QA10C-25.007	QAC-0344	Composite View en las páginas JSPs	Usar el mismo código de cabecera, pie de página, menús, etc. en todas las páginas JSP. Como solucionarlo: Usar CompositeView para agrupar una serie de componentes que se apliquen en sus correspondientes capas. También se puede solucionar usando un Framework de templates como tiles. Beneficio: * Mayor flexibilidad, ya que las nuevas páginas se cargan en contenedores embebidos en una página principal. * Fácilidad a la hora de modificar el diseño general de la página. * Se mejora la reutilizacón del código. Ejemplo válido:			E	BAJO	1							
		QA10C-25.008	QAC-0345	Usar patrón Service to Worker o Dispatcher View	Para coordinar las peticiones del flujo de datos invocando componentes desde diferentes capas se deberá usar el modelo Service to Worker, Dispatcher View o Front Controller. En los dos Patrones son los Controladores los que se encargan de llamar a la capa de servicios. Una vez obtenido el resultado será el mismo Controlador (en el caso del patrón Front Controller) o el Dispatcher el que se encargue de redireccionar el resultado a la vista.			E	MODERADO	1		1					
		QA10C-25.009	QAC-0346	Usar Apache tiles como Framework de templates	Usar Apache tiles como Framework de templates Apache tiles nos ofrece la posibilidad de configurar la resolución de nuestras vistas por medio de templates fácilmente configurables y reutilizables. Se podrá validar buscando en la definición de beans las siguientes clases: org.springframework.web.servlet.view.tiles2.TilesConfigurer org.springframework.web.servlet.view.tiles2.TilesView Ejemplo de configuración: <bean id="tilesConfigurer" class="org.springframework.web.servlet.view.tiles2.TilesConfigurer"> <property name="definitions"> <list> <value>/WEB-INF/tiles-defs/templates.xml</value> </list> </property> </bean> <bean id="tilesViewResolver" class="org.springframework.web.servlet.view.UrlBasedViewResolver">			E	BAJO	1							
		QA10C-25.010	QAC-0347	Usar Apache velocity o freemarker como Framework de templates	Usar Apache velocity o freemarker como Framework de templates Apache velocity y freemarker son dos de los frameworks de templates mas conocidos y potentes existentes dentro del mundo JEE, además al igual que apache tiles se integran muy fácilmente con Spring. Se podrá validar buscando en la definición de beans las siguientes clases: org.springframework.web.servlet.view.velocity.VelocityConfigurer org.springframework.web.servlet.view.velocity.VelocityViewResolver org.springframework.web.servlet.view.freemarker.FreeMarkerConfigurer org.springframework.web.servlet.view.freemarker.FreeMarkerViewResolver Ejemplo de configuración: <bean id="velocityConfig" class="org.springframework.web.servlet.view.velocity.VelocityConfigurer"> <property name="resourceLoaderPath"> <value>/WEB-INF/mods/core/sample/velocity/</value> </property> </bean>			E	BAJO	1							
	QA10C-26	**Arquitectura SPRING - JEE**																

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-26.001	QAC-0348	Service Locator	Las llamadas servidor de nombres de JDNI se hace de maneras distintas, dependen del entorno del cliente o se usan indiscriminadamente. Como solucionarlo: Se deberá usar ServiceLocator para encapsular el complejo proceso de realizar el lookup mediante JDNI. Esto se logra mediante la configuración del Spring Framework. Beneficios: * Abstrae por completo el acceso al lookup de JNDI * Facil de externalizar la configuración del ServiceLocator. Ejemplo válido En el servlet.xml existente se definirá una sóla vez el ServiceLocator: <bean name="beanName" class="org.springframework.jndi.JndiObjectFactoryBean">			E	BAJO	1	1	1						
		QA10C-26.002	QAC-0349	Business Delegate	El PageController invoca directamente a los componentes de negocio. Las capas de presentación y negocio están altamente acopladas. Esta regla solo estará activada si se usan EJBs, ya que es muy parecida a la regla de Application Controller Como solucionarlo: BusinessDelegate es el adaptador que se deberá usar para invocar objetos de negocio desde la capa de presentación Beneficios: * Se minimiza el acoplamiento entre ambas capas * Se ocultan los detalles de la arquitectura a los servicios			E	MODERADO	1		1		1				
		QA10C-26.003	QAC-0350	Session Facade	Se accede a multiples métodos remotos en respuesta a acciones del usuario. Esto crea dificultad a la hora de responder a esas peticiones remotas. Esta regla solo estará activada si se usan EJBs, ya que es muy parecida a la regla de Limitar el número de Front Controllers Como solucionarlo: Se usará SessionFacade para el acceso remoto, el cual oculta la lógica de negocio al cliente. Beneficios: * Mejora las transacciones entre los servicios de las			E	MODERADO	1		1						
		QA10C-26.004	QAC-0351	Application Service	Toda la lógica de negocio está implementada en la SessionFacade, por lo que la gestión de la aplicación se hace complicada. Esta regla solo estará activada si se usan EJBs Como solucionarlo: Mediante ApplicationService se podrá encapsular toda la lógica de negocio en clases y componentes POJO. Beneficios: * Los componetes POJO acceden a los servicios del EJB como si estuviesen invocados en la SessionFacade. * Los componentes POJO hacen a la aplicación mas fácil de testear, y puede ser ejecutada fuera del contendedor EJB. Ejemplo válido: En la configuración de los beans de cada EJB deberá especificarse el ApplicationService: <bean id="myAppService"			E	MODERADO	1		1						
		QA10C-26.005	QAC-0352	Business Interface	La interfaz remota de una SessionFacade define los métodos que son accesibles para el cliente. La clase del bean ofrece la implementación de los métodos, pero no están directamente relacionados, lo cual conlleva errores en tiempo de despliegue. Como solucionarlo: Se deberá implementar una interfaz común que contenga los métodos de la lógica de negocio. Beneficios: * Se previene de anomalías entre la interfaz remota y la implementación del Bean.			E	MODERADO	1		1		1				
		QA10C-26.006	QAC-0353	Comprimir recursos Javascript para produccion	Dentro de la distribución web comprimir los recursos javascript y css con alguna librería como yuicompressor. Se realizara la compresión a la hora de hacer el war o el ear. Si la construcción se hace mediante maven, podemos hacerlo de la siguiente manera: <plugin> <groupId>net.alchim31.maven</groupId> <artifactId>yuicompressor-maven-plugin</artifactId> <version>1.1</version> <executions> <execution> <goals> <goal>compress</goal> </goals> </execution> </executions> <configuration> <nosuffix>true</nosuffix> <force>true</force> <encoding>UTF-8</encoding> <skip>${yuicompressor.skip}</skip> </configuration> </plugin>			E	MODERADO	1			1	1				

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-26.007	QAC-0354	Distribuir los distintos mensajes de la aplicación en ficheros para facilitar su mantenimiento.	Distribuir los distintos mensajes de la aplicación en ficheros para facilitar su mantenimiento. El habitual encontrar en una aplicación JEE, que todos los mensajes (de aplicación y errores) y textos de la misma son configurados en ficheros .properties y encima para distintos idiomas. Esto hace que los ficheros de propiedades crezcan hasta convertirlos en inmanejables. Por este motivo se recomienda dividir tantos los mensajes como los distintos textos en ficheros distribuidos por módulos. Para ello se recomienda usar ResourceBundleMessageSource de spring y se configuraría de la siguiente manera. <bean id="messageSource" class="org.springframework.context.support.ResourceBundleMessageSource"> <property name="basenames"> <list> <!-- Default literals --> <value>i18n/login</value> <value>i18n/administration</value> <value>i18n/applications</value> <value>i18n/common</value> <value>i18n/customer</value> <value>i18n/menu</value> <value>i18n/user</value> <value>i18n/movements</value> <value>i18n/analysis</value> <value>i18n/reports</value>			E	BAJO	1	1				1		
		QA10C-26.008	QAC-0355	Usar slf4j con la implementación log4j para el data logging	Usar la librería slf4j para desligarnos de la implementación de trazas que queramos utilizar, adicionalmente se recomienda el uso de log4j que es la librería de data logging mas utilizada y fiable de la comunidad open source. Con maven lo importaremos de la siguiente manera: <dependency> <groupId>log4j</groupId> <artifactId>log4j</artifactId> <version>1.2.16</version> </dependency> <dependency> <groupId>org.slf4j</groupId> <artifactId>com.springsource.slf4j.log4j</artifactId> <version>1.5.6</version> <exclusions> <exclusion> <artifactId>com.springsource.org.apache.log4j</artifactId> <groupId>org.apache.log4j</groupId> </exclusion> </exclusions> </dependency> <dependency> <groupId>org.slf4j</groupId> <artifactId>com.springsource.slf4j.api</artifactId> <version>1.5.6</version> </dependency> Nota: Como se puede observar, estamos utilizando la			E	MODERADO	1	1	1					
		QA10C-26.009	QAC-0356	Colocar implementación de log4j en el lib del appserver para evitar memory leak	Se recomienda colocar la librería jcl-over-slf4j.jar en el lib del Servidor de aplicaciones y no tener una copia en el WEB-INF/lib de cada aplicación (en nuestro caso en la librería compartida) ver http://www.slf4j.org/codes.html#release			E	MODERADO	1	1	1				1		
		QA10C-26.010	QAC-0357	Configuración de mail fuera de nuestra aplicación.	Configuración de mail fuera de nuestra aplicación. Se recomienda al igual que con los DataSources que la configuración de mail quede en manos de los administradores del servidor de aplicaciones. Podemos acceder a la misma vía JNDI de la siguiente manera:			E	BAJO	1	1					1		
	QA10C-27	**Arquitectura SPRING - TEST**																
		QA10C-27.001	QAC-0358	Usar Anotaciones para los Test Unitarios y de Integracion	Realizar de forma continua pruebas unitarias y de integración. Spring Framework 2.5.x funciona con JUnit 4.4; Spring Framework 3.0 ya trae soporte para JUnit 4.5 y superior. Ejemplo: @RunWith(SpringJUnit4ClassRunner.class) @ContextConfiguration(locations = { "classpath:application-db.xml", "classpath:application-bo.xml" }) public class InvasorBoTest { /** La anotacion Autowired inyecta automáticamente el bean de acuerdo a su tipo */ @Autowired private InvasorBo invasorBo; /** Busca un invasor por su identificador unico */ @Test public void buscar() { System.out.println("testBuscar"); String idInvasor = 100; Invasor invasor = invasorBo.buscar(idInvasor); assertNotNull(invasor); assertEquals(idInvasor, invasor.getId()); } } Beneficios: Mejorar la calidad de la aplicación. Si se utiliza JUnit 4.x la configuración para inyección es mucho más simple, haciéndose a través de anotaciones.			E	BAJO	1	1	1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-27.002	QAC-0359	Hacer Test de base de datos con transaccionalidad	Es recomendable manejar las transacciones en los tests, de manera de hacer un rollback automático de las operaciones realizadas.. Spring Framework 2.5.x funciona con JUnit 4.4; Spring Framework 3.0 ya trae soporte para JUnit 4.5 y superior. Ejemplo: @RunWith(SpringJUnit4ClassRunner.class) @ContextConfiguration(locations = { ""classpath:application-db.xml"", ""classpath:application-bo.xml"" }) @TransactionConfiguration(transactionManager=""transactionManager"") @Transactional public class InvasorBoTest { /** La anotacion Autowired inyecta automáticamente el bean de acuerdo a su tipo */ @Autowired private InvasorBo invasorBo; /** Este test tendrá un rollback automático */ @Test public void buscar() { ... } } Beneficios: Si necesitamos que los datos se restablezcan después de			E	BAJO	1	1	1					
		QA10C-27.003	QAC-0360	Test de BBDD contra base de datos distinta al desarrollo.	Es fundamental para que los tests sean completamente independientes e útiles que utilicen otro schema de BBDD al que se utiliza en el desarrollo. Mediante la configuración de los tests se le indicara cual es el datasource a utilizar durante las pruebas.			E	MODERADO	1	1	1					
		QA10C-27.004	QAC-0361	Realizar Tests en la capa de Servicios	Comprobar que se realizan pruebas unitarias y de integración para los servicios Spring (capa negocio). Para validar esta regla se puede buscar que para el paquete de servicios tengan clases de test que contengan la anotación @RunWith(SpringJUnit4ClassRunner.class) para los test de integración. Para los Tests unitarios bastara con que tengan clases que contengan la anotación @Test o tengan métodos que empiecen por test. También se puede buscar que importen y usen clases del paquete org.junit.Assert. Beneficios: La capa Service es una parte importante que hay que testear, es la que realiza el negocio y cualquier fallo en ella provocará el fallo a nivel de la aplicación. Además es muy			E	BAJO	1	1	1					
		QA10C-27.005	QAC-0362	Tests en la capa de acceso a datos.	Comprobar que se realizan pruebas unitarias y de integración para la capa de Acceso a datos (DAOs o Repositorios) de Spring. Para validar esta regla se puede buscar que para el paquete de servicios tengan clases de test que contengan la anotación @RunWith(SpringJUnit4ClassRunner.class) para los test de integración. Otro punto a validar es que se esten usando clases de dbunit como org.dbunit.DatabaseTestCase Para los Tests unitarios bastara con que tengan clases que contengan la anotación @Test o tengan métodos que empiecen por test. También se puede buscar que importen y usen clases del paquete org.junit.Assert. Beneficios: La capa DAO es una parte importante que hay que testear,			E	BAJO	1		1					
		QA10C-27.006	QAC-0363	Realizar Pruebas de Integración de la Base de Datos con dbunit.	DBUNIT nos proporciona un api muy flexible para la ejecución de pruebas contra Base de Datos. Mediante este api podemos hacer que cada test o grupo de tests sean totalmente independientes proporcionando una lista de ficheros xml con los datos de prueba a usar y permitiéndonos borrar esos datos al finalizar las pruebas. Esta regla puede ser opcional, en caso de que queramos validar que efectivamente se usa dbunit buscaremos que los tests utilicen clases como: import org.dbunit.DatabaseTestCase; import org.dbunit.database.DatabaseConnection; import org.dbunit.database.IDatabaseConnection; import org.dbunit.dataset.IDataSet;			E	BAJO	1	1	1					
	QA10C-28	**Arquitectura SPRING - SECURITY**															
		QA10C-28.001	QAC-0364	Separar applicationContext-Security	Usar fichero aparte para la configuración de la seguridad de Spring: applicationContext-security.xml Beneficios: Al tener la configuración de seguridad separada del archivo de configuración de Spring, tenemos un sistema más mantenible. Es importante tener en cuenta que la sintaxis del fichero de configuración de spring security es distinta a la declaración típica de beans de una aplicación. Cómo solucionarlo: Separar los módulos de Spring Security en otro xml, e importar este xml desde el applicationContext.xml principal.			E	BAJO	1		1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-28.002	QAC-0365	Usar SpringSecurity	Si no se utiliza Spring Security saltará la regla. Para detectar que se utiliza debe aparecer un bloque http y otro authentication-manager dentro del applicationContext-security.xml, además en el web.xml comprobar que se añade el fichero xml que contiene al applicationContext-security.xml, además de definir el filtro de org.springframework.web.filter.DelegatingFilterProxy. Ejemplo: En el bloque http estamos diciendo que las expresiones para el control de acceso estén activadas y que usaremos un login basado en un formulario. En el intercept-url estamos diciendo que todo lo que cumpla el patron /** (el raiz y todos sus subdirectorios) es accesible a todo el mundo "permitAll". En el bloque del authentication-manager definimos que servicio va a resolver la autenticación, en nuestro caso consultado la base de datos mediante un repositorio. En una aplicación real se deben utilizar usuarios y roles de una base de datos, de un servidor LDAP o integrarse con sistemas de single sign-on (como por ejemplo OpenID). <sec:http auto-config="true"> <sec:intercept-url pattern="/**" access="IS_AUTHENTICATED_ANONYMOUSLY" /> <sec:form-login login-page="/web/common/login" default-target-url="/web/main/main" authentication-failure-handler-ref="authenticationFailureHandler" always-use-default-			E	MODERADO	1		1						
		QA10C-28.003	QAC-0366	Mantenernos actualizados a las últimas versiones de Spring Security	Mantenernos actualizados a las últimas versiones de Spring Security. Es importante actualizarse a las últimas versiones de este módulo de seguridad. La seguridad es uno de los puntos mas críticos en cualquier aplicación, y las nuevas versiones que van apareciendo nos prevén entre otras cosas de posibles vulnerabilidades. Para ello podemos buscar en nuestro fichero de configuración de seguridad applicationContext-Security.xml que versión se esta utilizando mediante el xsd: http://www.springframework.org/schema/security/spring-security-3.0.3.xsd			E	MODERADO	1		1						
		QA10C-28.004	QAC-0367	Filtrar recursos accedidos	Comprobar que dentro del apartado <http> dentro del applicationContext-security.xml aparecen varias <intercept-url> y no aparece solamente: <sec:intercept-url pattern="/web/public/**/**" access="IS_AUTHENTICATED_ANONYMOUSLY" /> Ejemplo: <sec:intercept-url pattern="/web/public/**/**" access="IS_AUTHENTICATED_ANONYMOUSLY" /> <sec:intercept-url pattern="/web/main/main" access="ROLE_USER" /> <sec:intercept-url pattern="/web/sysadmin/**/**" access="ROLE_SYS_ADMIN" /> <sec:intercept-url pattern="/web/common/login" access="IS_AUTHENTICATED_ANONYMOUSLY" /> <sec:intercept-url pattern="/resources/**/**" access="IS_AUTHENTICATED_ANONYMOUSLY, ROLE_USER" /> Con esta configuración cualquiera podrá acceder al contenido de /web/public, a los recursos de /resources y a la página de login, los usuarios autenticados podrán acceder a la carpeta /web/common/login, los usuarios con el rol "SYS_ADMIN" podrán acceder a /web/sysadmin y sus subdirectorios. Beneficios: Suele ser una buena práctica negar el acceso por defecto			E	MODERADO	1		1						
		QA10C-28.005	QAC-0368	No seguridad en recursos estaticos.	Si no tenemos otro bloque http con secured=false para los recursos estáticos, aparecerá una violación. Ejemplo: Con este patrón haremos que no se aplique la seguridad (definida en el siguiente bloque http que tengamos en el AppContext-security.xml) a los recursos dentro de la carpeta static y sus subdirectorios. <http pattern="/static/**" secured="false"> </http> Beneficios: Puede interesar que recursos estáticos como las hojas de estilo no se vean afectadas por los filtros de Spring Security. Cómo solucionarlo: Se pueden añadir bloques adicionales de http que sólo se			E	BAJO	1		1						
		QA10C-28.006	QAC-0369	Organizar todos los recursos publicos a partir de unas rutas claramente definidas para facilitar la configuración.	Organizar todos los recursos públicos a partir de unas rutas claramente definidas para facilitar la configuración. Por ejemplo si organizamos toda la zona pública de nuestra aplicación a partir de la ruta /web/public la configuración de acceso anónima seria tan fácil como esto: <sec:intercept-url pattern="/web/public/**/**"			E	BAJO	1		1						
		QA10C-28.007	QAC-0370	LogOut Debe Habilitarse	Comprobad que dentro de los bloques <http> dentro del applicationContext-security.xml, nos encontramos con la etiqueta <logout>. Con esto tendríamos habilitada la posibilidad de desconectarnos. Ejemplo: <sec:logout logout-success-url="/web/common/login" invalidate-session="true" /> Beneficios: Si se intenta usar el link de logout para desconectarse de una aplicación con Spring Security y dice que el acceso esta denegado. Eso es debido a que no está habilitada la pantalla de logout, gracias a esta regla se comprueba que este habilitada. Es importante que un usuario tenga la posibilidad de desconectarse, por motivos de seguridad.			E	MODERADO	1		1						

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-28.008	QAC-0371	Usar Encriptacion Password	Comprobar que dentro del authentication manager hay asociado un <password-encoder ref="encoder" />, también tendrá que estar definido el bean encoder con alguna clase que implemente org.springframework.security.authentication.encoding.PasswordEncoder. Ejemplo: Beneficios: La encriptación de contraseñas es una practica altamente recomendable y que con Spring Security es bastante sencillo de usar. Podemos utilizar por ejemplo la implementación org.springframework.security.authentication.encoding.ShaPasswordEncoder. ShaPasswordEncoder aplica 1024 iteraciones del algoritmo SHA que le indiquemos. Además se puede utilizar una clave para toda la aplicación que se combinará también con la password y el salt autogenerado. Esto hace que las passwords sean menos vulnerables a un ataque por fuerza bruta. Cómo solucionarlo: Hay que modificar la parte correspondiente al authentication manager y añadir el bean que se encargara de realizar el cifrado de la contraseña.			E	MODERADO	1		1					
		QA10C-28.009	QAC-0372	Usar un algoritmo de encriptación SHA-2 (SHA-224, SHA-256, SHA-384, y SHA-512)	La encriptación de contraseñas es una practica altamente recomendable y que con Spring Security es bastante sencillo de usar. Podemos utilizar por ejemplo la implementación org.springframework.security.authentication.encoding.ShaPasswordEncoder. ShaPasswordEncoder aplica 1024 iteraciones del algoritmo SHA que le indiquemos. Es conveniente usar como mínimo SHA-224 para la encriptación <sec:authentication-manager> <sec:authentication-provider user-service-ref="userDetailsServiceRepositoryBasedOn"> <sec:password-encoder ref="passwordEncoder"> <sec:salt-source ref="saltSource"/> </sec:password-encoder> </sec:authentication-provider> </sec:authentication-manager>			E	MODERADO	1		1					
		QA10C-28.010	QAC-0373	Usar un mecanismo de SALT para la configuración de seguridad.	La autenticación SALT, consiste en agregar un texto adicional a la contraseña, convirtiéndose en una Buena practica ya que proporciona una dificultad adicional a la contraseña escogida. Es muy fácil de usar si partimos de la implementación ReflectionSaltSource de spring que nos permite coger de cada usuario la SALT que le ha sido asignada. <sec:authentication-manager> <sec:authentication-provider user-service-ref="userDetailsServiceRepositoryBasedOn"> <sec:password-encoder ref="passwordEncoder"> <sec:salt-source ref="saltSource"/> </sec:password-encoder> </sec:authentication-provider> </sec:authentication-manager> <bean id="passwordEncoder" class="org.springframework.security.authentication.encoding.ShaPasswordEncoder"> <constructor-arg value="256" /> </bean>			E	BAJO	1		1					
		QA10C-28.011	QAC-0374	Usuarios ni en xml ni en properties	Comprobar que dentro de la etiqueta <security:authentication-manager> no se encuentren definidos los <security:user>, ni este definida la etiqueta <security:user-service properties="xxxx.properties">, la cual apunta a un archivo properties con los valores para los usuarios/pass. Ejemplo: Para obtener los usuarios desde XML: <authentication-manager> <authentication-provider> <password-encoder hash="sha"/> <user-service> <user name="jimi" password="d7e6351eaa13189a5a3641bab846c8e8c69ba39f" authorities="ROLE_USER, ROLE_ADMIN" /> <user name="bob" password="4e7421b1b8765d8f9406d87e7cc6aa784c4ab97f" authorities="ROLE_USER" /> </user-service> </authentication-provider> </authentication-manager> Para obtener los usuarios desde un archivo de propiedades, para eso el XML anterior se reduce a: <user-service id="userDetailsService"			B	ALTO	1		1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10C-28.012	QAC-0375	Usuarios en LDAP	Comprobar que dentro de la etiqueta <authentication-manager> se esta usando la implementación de LDAP. LdapAuthenticationProvider Ejemplo: <bean id="authenticationManager" class="org.springframework.security.authentication.ProviderManager"> <property name="providers"> <list> <ref local="daoAuthenticationProvider"/> <ref local="anonymousAuthenticationProvider"/> <ref local="ldapAuthenticationProvider"/> </list> </property> </bean> <bean id="contextSource" class="org.springframework.security.ldap.DefaultSpringSecurityContextSource"> <constructor-arg value="ldap://monkeymachine:389/dc=springframework,dc=org"/> <property name="userDn" value="cn=manager,dc=springframework,dc=org"/> <property name="password" value="password"/> </bean>			E	BAJO	1		1					
		QA10C-28.013	QAC-0376	Usuarios en Base de Datos	La práctica más habitual para guardar los usuarios de un sistema es en Base de datos, hay que asegurarse que tengamos definido un servicio que utilice una estrategia de acceso a datos mediante un DAO o Repositorio inyectado. Ejemplo: <sec:authentication-manager> <sec:authentication-provider user-service-ref="userDetailsServiceRepositoryBasedOn"> <sec:password-encoder ref="passwordEncoder"> <sec:salt-source ref="saltSource"/> </sec:password-encoder> </sec:authentication-provider> </sec:authentication-manager> <bean id="userDetailsServiceRepositoryBasedOn" class="com..security.model.services.implementations.UserDetailsServiceRepositoryBasedOn"> <property name="securityRepository" ref="securityRepository" /> </bean> Como podemos observar en el ejemplo el servicio de			E	BAJO	1		1					
		QA10C-28.014	QAC-0377	Usar proveedor de usuarios	La violación se lanzara si no tenemos definido un servicio que actúe como proveedor de usuarios, eso se comprueba si nos encontramos el atributo security:authentication-provider user-service-ref=" "> dentro de la etiqueta <security:authentication-manager> Ejemplo: <sec:authentication-manager> <sec:authentication-provider user-service-ref="userDetailsServiceRepositoryBasedOn"> <sec:password-encoder ref="passwordEncoder"> <sec:salt-source ref="saltSource"/> </sec:password-encoder> </sec:authentication-provider> </sec:authentication-manager> <bean id="userDetailsServiceRepositoryBasedOn" class="com..security.model.services.implementations.UserDetailsServiceRepositoryBasedOn"> <property name="securityRepository" ref="securityRepository" /> </bean> Como podemos observar en el ejemplo el servicio de autenticación utiliza un repositorio que es el que utilizara para validar el acceso. Este servicio tendrá que sobrescribir el método			E	MODERADO	1		1					
		QA10C-28.015	QAC-0378	Encapsular la obtención del Usuario	Comprobar que en el código no existen llamadas a: SecurityContextHolder.getContext().getAuthentication().getPrincipal(); Solo debería existir una llamada en una clase que fuera usada por las demás (encapsulación). Ejemplo: Para obtener el nombre de un usuario se hace de esta forma: Object obj = SecurityContextHolder.getContext().getAuthentication().getPrincipal(); String username=null; if (obj instanceof UserDetails) { username= ((UserDetails)obj).getUsername(); } else { username = obj.toString(); } Beneficios: Gracias a la encapsulación al poner todas las llamadas iguales en un mismo método, reutilizamos código y hacemos que este sea más mantenible, ya que si quisiéramos cambiar el mecanismo de seguridad solo lo tendríamos que hacer en un sitio. Cómo solucionarlo: Podríamos usar alguna clase que encapsulara la obtención del usuario y que estuviera disponible desde el contexto de Spring y con el patrón Singleton.			E	BAJO	1		1					
		Arquitectura SPRING - SECURITY																
	QA10C-29	**Verificación Codificación de Aplicaciones MQWorkflow**																
		QA10D-29.001		No utilizar filtros que comiencen con wildcards sobre campos estándar de MQWorkflow	No se permite utilizar filtros LIKE que comiencen por wildcards sobre campos comunes de MQWorkflow. además siempre se ha de buscar el filtro más restrictivo para acotar previamente las tareas o instancias de proceso sobre las que buscar. Seguir las recomendaciones expuestas las "best practices" del support pack WA0B de IBM en http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24006852&loc=en_US&cs=utf-8&lang=en			B	ALTO	0	5		5				
	QA10C-30	**Planificador Host: Contol-M**																

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automatico 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10D-30.001	QAC-0954	No utilizar recursos de Control-M: TANTIA TANTIAVIP TANTIA_BLADES	No se permite el uso de los recursos de Control-M: TANTIA TANTIAVIP TANTIA_BLADES			B	ALTO	1		1		1			
		QA10D-30.002		NOMBRE DE GGRUPO DEBE SER DE 8 POSICIONES.	VALIDA QUE EL NOMBRE DEL GRUPO EN LAS MALLAS DE CONTROL-M, DEBERÁ DE SER DE OCHO POSICIONES.			B	ALTO	1		1			1		
		QA10D-30.003		POSICIONES 1-8 DE GGRUPO DEBE SER IGUAL A GMEMNAME.	VALIDA QUE EL NOMBRE DEL GRUPO EN LAS MALLAS DE CONTROL-M, SEA IGUAL AL GMEMNAME			B	ALTO	1		1					
		QA10D-30.004		GMEMNAME DEBE SER IGUAL GTABLE.	VALIDA EL NOMBRE DE LA TABLA DE CONTROL-M, SEA IGUAL AL NOMBRE DEL GMEMNAME			B	ALTO	1		1					
		QA10D-30.005		GTABLE POSICIONES 1-2 DEBE SER CVE PAIS A 2 POS. VALIDO.	VALIDA QUE LAS PRIMERAS 2 POSICIONES DE LA TABLA DE CONTROL-M, SEAN IGUAL AL PREFIJO DEL PAIS.			B	ALTO	1		1			1		
		QA10D-30.006		GTABLE POSICIONES 3-4 DEBE SER CVE DE APLICACION VALIDO.	VALIDA QUE LAS POSICIONES 3 y 4, DE LAS TABLAS DE CONTROL-M, SEAN IGUAL PREFIJO DE LA APLICACIÓN.			B	ALTO	1		1			1		
		QA10D-30.007		GTABLE POSICIONES 5 DEBE SER NEGOCIO VALIDO.	VALIDA QUE LA QUINTA POSICION, DEL NOMBRE DE LA TABLA TIPO GRUPO, SEAN IGUAL AL PREFIJO DEL NEGOCIO			B	ALTO	1		1			1		
		QA10D-30.008		GMEMNAME DEBE SER DE 8 POSICIONES.	VALIDA, QUE MEMNAME DE UNA MALLA TIPO GRUPO, SEA DE OCHO POSICIONES.			B	ALTO	1		1			1		
		QA10D-30.009		GOWNER CODIFICADO DEBE SER VALIDO.	VALIDA, QUE EL OWNER CODIFICADO EN EL PROCESO SEA VALIDO.			B	ALTO	1		1					
		QA10D-30.010		DEBE CODIFICAR EL CAMPO GDESC.	DENTRO DE UNA MALLA DE CONTROL-M TIPO GRUPO, VALIDA QUE TENGA UNA DESCRIPCION, HASTA DE 50 POSICIONES.			B	ALTO	1		1			1		
		QA10D-30.011		CAMPO ADJUST CONDITIONS DEBE SER VALIDO.	VALIDA, QUE PARA EL CAMPO "ADJUST CONDITIONS", DEBERA SER EL CORRECTO.			B	ALTO	1		1					
		QA10D-30.012		MAXWAIT DEBE SER EL PERMITIDO.	VALIDA QUE EL GMAXWAIT A NIVEL GRUPO SEA IGUAL O MENOR AL MAXIMO MAXWAIT CODIFICADO A NIVEL TAG			B	ALTO	1		1		1			
		QA10D-30.013		NO DEBEN EXISTIR VARIABLES ASIGNADAS.	VALIDA QUE A NIVEL GRUPO, NO VENGAN ASIGNADAS VARIABLES DE CONTROL-M			B	ALTO	1		1					
		QA10D-30.014		VALIDA QUE EL GDOCMEN SEA IGUAL GMEMNAME.	ESTA REGLA VALIDA QUE EL CAMPO MEMNAME Y DOCMEM, TENGAN EL MISMO VALOR			B	ALTO	1		1					
		QA10D-30.015		VALIDA QUE LA BIBLIOTECA GDOCLIB SEA VALIDA.	VALIDA, QUE SEA CORRECTA LA BIBLIOTECA "DOCLIB".			B	ALTO	1		1					
		QA10D-30.016		PROHIBIDO EL USO DE JOBS TIPO GRUPO	VALIDA QUE NO EXISTAN CODIFICADOS JOBS TIPO GRUPO PARA UN PERFIL DIFERENTE A ALNOVA			B	ALTO	1		1					
		QA10D-30.017		NO DEBEN EXISTIR CONDICIONES DE ENTRADA.	PARA PROCESOS TIPO GRUPO, NO DEBERAN DE EXISTIR CONDICIONES DE ENTRADA			B	ALTO	1		1					
		QA10D-30.018		NO SE PERMITE CONFIRM IGUAL A YES.	PARA PROCESOS TIPO GRUPO, NO ESTA PERMITIDO INGRESAR "CONFIRM"			B	ALTO	1		1	1				
		QA10D-30.019		NO DEBEN EXISTIR CONDICIONES DE SALIDA.	PARA MALLAS TIPO GRUPO, NO DEBERAN DE EXISTIR CONDICIONES DE SALIDA			B	ALTO	1		1					
		QA10D-30.020		TRANSFERENCIAS CODIFICAN N EN CUARTA POSICION DE JOB	VALIDA QUE SI SE CODIFICA UNA CONDICION GLOBAL LA CUARTA POSICION DEL MEMNAME CORRESPONDA A TRANSFERENCIA			B	ALTO	1		1			1		
		QA10D-30.021		TRANSFERENCIAS CODIFICAN "F" EN QUINTA POSICION DE JOB	VALIDA QUE SI SE CODIFICA UNA CONDICION GLOBAL LA QUINTA POSICION DEL MEMNAME TRAIGA UNA LETRA "F" Ó UNA LETRA "K" y CORRESPONDA A UNA TRANSFERENCIA			B	ALTO	1		1			1		
		QA10D-30.022		CALENDARIO GDCAL A NIVEL TAG DEBE SER VALIDO.	VALIDA, QUE EN EL CALENDARIO "GDCAL", SEA EL VALIDO, A NIVEL "TAG"			B	ALTO	1		1			1		
		QA10D-30.023		CALENDARIO GWCAL A NIVEL TAG DEBE SER VALIDO.	VALIDA, QUE EN EL CALENDARIO "GWCAL", SEA EL VALIDO A NIVEL "TAG"			B	ALTO	1		1			1		
		QA10D-30.024		CALENDARIO GCONFCAL A NIVEL TAG DEBE SER VALIDO.	VALIDA QUE EL CALENDARIO "GCONFCAL" SEA VALIDAO A NIVEL TAG.			B	ALTO	1		1			1		
		QA10D-30.025		GMAXMAIT A NIVEL TAG DEBE SER VALIDO.	VALIDA, QUE PARA PROCESOS TIPO GRUPO, EL PARAMETRO "GMAXWAIT", SEA VALIDO.			B	ALTO	1		1		1			
		QA10D-30.026		BIBLIOTECA DE DOCUMENTACION DEBE SER VALIDA.	VALIDA QUE DE NO SER UN SEPARADOR, LA DOCLIB SEA LA CORRECTA Y CODIFICADA A NIVEL JOB, DE HABER OMITIDO LA MISMA, VALIDARÁ QUE EXISTA CODIFICADA A NIVEL GRUPO			B	ALTO	1		1					
		QA10D-30.027		DEBE EXISTIR AL MENOS UNA CONDICION DE ENTRADA.	VALIDA QUE SI NO ES UN SEPARADOR, DEBERA DE EXISTIR AL MENOS UNA CONDICION DE ENTRADA, DE LO CONTRARIO EMITIRA MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.028		POSICIONES 1-8 DE CONDICION DE SALIDA DEBE SER EL GRUPO.	PARA PROCESOS CONTROL-M, VALIDA QUE LAS PRIMERAS OCHO POSICIONES DE LA CONDICION DE SALIDA SEAN IGUALES AL GRUPO			B	ALTO	1		1					
		QA10D-30.029		POSICION 9 DE CONDICION DE SALIDA DEBER SER "_" .	VALIDA QUE LA NOVENA POSICION DE LA CONDICION DE SALIDA SEA UN GUION BAJO "_"			B	ALTO	1		1			1		
		QA10D-30.030		POSICIONES 10-17 DE CONDICION DE SALIDA DEBE SER MEMNAME.	VALIDA QUE LAS POSICIONES 10 A 17 DE LA CONDICION DE SALIDA, SEAN IGUALES AL MEMNAME.			B	ALTO	1		1			1		
		QA10D-30.031		JOBS CON TAREA CICLICA DEBEN CODIFICAR MAXWAIT A NIVEL JOB	SI LA TAREA ES UN JOB CICLICO MAXWAIT DEBE CODIFICARSE A NIVEL JOB.			B	ALTO	1		1		1			
		QA10D-30.032		DEBE CODIFICAR CALENDARIO A NIVEL JOB (COMO DCAL O TAG)	VALIDA SI EXISTE CODIFICADO EL CALENDARIO "DCAL" A NIVEL JOB EN CASO DE NO EXISTIR, VERIFICA SI EL TAG QUE TIENE ASIGNADO ES DIFERENTE A ESPACIOS O GENERAL, Y BUSCA EN EL TAG ASIGNADO SI EXISTE CODIFICADO UN CALENDARIO "DCAL".			B	ALTO	1		1			1		
		QA10D-30.033		OWNER DEBE SER VALIDO.	VALIDA SI EXISTE CODIFICADO EL OWNER A NIVEL JOB Y EN CASO DE NO ESTAR CODIFICADO, VALIDARÁ QUE EXISTA UN OWNER A NIVEL GRUPO.			B	ALTO	1		1					
		QA10D-30.034		VALIDA QUE MAXWAIT SEA VALIDO.	VALIDA A NIVEL JOB SI EL MEMNAME NO ES UN SEPARADOR Y NO ES UNA TAREA CICLICA, SI EL MAXWAIT DEBE SER VALIDO			B	ALTO	1		1		1			
		QA10D-30.035		PROHIBIDO EL USO DE ESTA BIBLIOTECA A NIVEL MEMLIB.	VALIDA QUE SI NO ES UN SEPARADOR, QUEDARÁ PROHIBIDO EL USO DE ESTA BILIOTECA A NIVEL "MEMLIB"			B	ALTO	1		1					
		QA10D-30.036		DAYS CON FECHAS NEGATIVAS Y SIN CALENDARIO ASOCIADO	IMPLEMENTA REGLA, SI CALENDARIO ES NULO Y DAYS CONTIENE '-' ENVIA ERROR			B	ALTO	1		1			1		
		QA10D-30.037		WDAYS CON FECHAS NEGATIVAS Y SIN CALENDARIO ASOCIADO	IMPLEMENTA REGLA, SI CALENDARIO ES NULO Y WDAYS CONTIENE '-' ENVIA ERROR			B	ALTO	1		1			1		
		QA10D-30.038		EL CAMPO GROUP DEBE CONTENER INFORMACION DE LA POSICION 1 A 8	VALIDA, QUE SI NO ES UN SEPARADOR, EL CAMPO GROUP DEBERÁ DE CONTENER INFORMACION			B	ALTO	1		1					
		QA10D-30.039		LA POSICION NUMERO 7 DE LA CONDICION GLOBAL DEBE SER UN GUION.	PARA UNA CONDICION GLOBAL VALIDA LA SEPTIMA POSICION DE LA CONDICION DE SALIDA, Y ESTA DEBERÁ DE SER UN GUION.			B	ALTO	1		1			1		
		QA10D-30.040		POSICIONES DE LA 8 A LA 15 DE UNA CONDICION GLOBAL DEBE SER IGUAL AL MEMNAME.	PARA UNA CONDICION GLOBAL VALIDA DE LA POSICION OCHO A LA POSICION 15 SEA IGUAL AL MEMNAME.			B	ALTO	1		1			1		
		QA10D-30.041		POSICIONES 16 Y 17 DE UNA CONDICION GLOBAL DEBE SER IGUAL A "00".	PARA UNA CONDICION GLOBAL VALIDA LAS POSICIONES DE LA 16 A LA 17 DE LA CONDICION DE SALIDA, DEBERAN DE SER "00".			B	ALTO	1		1			1		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automatico 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10D-30.042		POSICION 18 DE CONDICION GLOBAL DEBE SER UN GUION BAJO.	PARA UNA CONDICION GLOBAL VALIDA LA POSICION 18 DE LA CONDICION DE SALIDA, DEBERA DE SER UN GUION BAJO " _ ".			B	ALTO	1		1			1		
		QA10D-30.043		POSICIONES 19 Y 20 DE UNA CONDICION GLOBAL DEBE SER VALIDA.	PARA UNA CONDICION GLOBAL VALIDA LAS POSICIONES 19 Y 20 DE LA CONDICION DE SALIDA, DEBERAN DE SER "PO"			B	ALTO	1		1			1		
		QA10D-30.044	QAC-0737	NOMBRE DE TABLA DEBE SER DE 8 POSICIONES	VALIDA, QUE DE NO COMENZAR EL NOMBRE DE LA TABLA CON "EV", PARA MALLAS EVENTUALES, ENTONCES LOS NOMBRES DE LAS "MALLAS, SEAN DE 8 POSICIONES.			B	ALTO	1		1			1		
		QA10D-30.045	QAC-0738	NOMBRE DE TABLA, POSICIONES 1 Y 2 DEBE CODIFICAR CLAVE DEL PAIS A 2 POSICIONES	VALIDA, QUE DE NO COMENZAR EL NOMBRE DE LA TABLA CON "EV", PARA MALLAS EVENTUALES, ENTONCES DEBERÁ DE VALIDAR LOS NOMBRES DE LAS TABLAS, QUE COMIENCEN CON LOS 2 PRIMEROS CALIFICADORES, DEL PAÍS			B	ALTO	1		1			1		
		QA10D-30.046	QAC-0739	NOMBRE DE TABLA POSICIONES 3-4 DEBE CODIFICAR APLICACION VALIDA	VALIDA, QUE DE NO COMENZAR EL NOMBRE DE LA TABLA CON "EV", PARA MALLAS EVENTUALES, LOS NOMBRES DE LAS TABLAS, DEBERÁN COMENZAR, EN LAS POSICIONES 3 y 4, LA APLICACIÓN.			B	ALTO	1		1			1		
		QA10D-30.047	QAC-0740	NOMBRE DE TABLA POSICION 5 DEBE CODIFICAR CLAVE DE NEGOCIO VALIDO	VALIDA, QUE DE NO COMENZAR EL NOMBRE DE LA TABLA CON "EV", PARA MALLAS EVENTUALES, LOS NOMBRES DE LAS TABLAS, DEBERÁN CONTENER, EN LA POSICION 5, EL "NEGOCIO" EN 1 POSICION,			B	ALTO	1		1			1		
		QA10D-30.048	QAC-0741	NOMBRE DE TABLA POSICIONES 6-8 DEBE CODIFICAR PERIODICIDAD VALIDA	SI EL NOMBRE DE LA TABLA, POSICIONES 1-2, NO ES EVENTUAL O POSICIONES 3-4 NO ES "HA", O NO ES "RAR", DEBERÁ CODIFICAR LA PERIODICIDAD CORRECTA EN 3 POSICIONES			B	ALTO	1		1		1	1		
		QA10D-30.049	QAC-0742	NOMBRE DE GRUPO POSICION 9 DEBE SER GUION BAJO	VALIDA, QUE DE NO SER UN SEPARADOR, ENTONCES VALIDE LA ETIQUETA "GROUP", Y DEBERÁ TRAER UN GUION BAJO "_", EN LA POSICION 9.			B	ALTO	1		1			1		
		QA10D-30.050	QAC-0743	NOMBRE DE GRUPO POSICIONES 10-14 DEBE SER TIPO PROCESO VALIDO.	VALIDA, QUE DE NO SER UN SEPARADOR, ENTOCES EL NOMBRE DEL GRUPO SE VERIFICARA EN LAS POSICIONES 10-14, Y DEBE SER VALIDO			B	ALTO	1		1			1		
		QA10D-30.051	QAC-0744	NOMBRE DE GRUPO POSICIONES 1-8 DEBE SER IGUAL A RESPALDO O NOMBRE DE LA TABLA-	VALIDA, QUE DE NO SER UN SEPARADOR, Y AL VERIFICAR LA ETQUETA "GROUP", NO ES "RESPALDO", DEBERÁ DE VALIDAR EL NOMBRE DE LA TABLA Y EL "GROUP", DEBERAN SER IGUALES			B	ALTO	1		1					
		QA10D-30.052	QAC-0745	FECHA SIMBOLICA DE CONDICION DE ENTRADA DEBE SER DISTINTA DE ****	SI AL VALIDAR EL NOMBRE DE LA MALLA, NO ES "DS", ENTONCES, VALIDARÁ LA FECHA SIMBOLICA DE LAS CONDICIONES DE ENTRADA, DEBERÁ DE SER DISTINTA DE (****).			B	ALTO	1		1					
		QA10D-30.053	QAC-0746	PREVENT-NCT2 DEBE SER IGUAL "Y"	VALIDA, QUE DE NO SER UN SEPARADOR, ENTONCES VALIDARÁ LA ETIQUETA "PREVENT_NCT2", Y DEBERÁ SER IGUAL A "Y"			B	ALTO	1		1			1		
		QA10D-30.054	QAC-0747	BIBLIOTECA MEMLIB DEBE SER VALIDA	VALIDA, QUE DE NO SER UN SEPARADOR,ENTONCES VALIDARÁ QUE LA BIBLIOTECA "MEMLIB" SEA VALIDA.			B	ALTO	1		1					
		QA10D-30.055	QAC-0748	OWNER DEBE SER VALIDO.	VALIDA, QUE DE NO SER UN SEPARADOR, Y SI EL "MEMNAME", NO ES CORRECTO, ENTONCES VALIDARÁ LA ETIQUETA "TABLE" , Y DE SER DIFERENTE, VALIDARÁ QUE EL "OWNER" SEA VALIDO			B	ALTO	1		1					
		QA10D-30.056	QAC-0749	MAXWAIT DEBE SER VALIDO.	VALIDA, QUE EL VALOR PARA EL PARÁMETRO "MAXWAIT" SEA VALIDO.			B	ALTO	1		1		1			
		QA10D-30.057	QAC-0750	D-CAT DEBE SER VALIDO.	VALIDA, QUE DE NO SER UN SEPARADOR, Y AL VERIFICAR LA ETQUETA "D_CAT", DEBERÁ DE SER VALIDA.			B	BAJO	1		1					
		QA10D-30.058	QAC-0752	CUANDO JOBNAME ES UN SEPARADOR MEMLIB DEBE SER DUMMY	ESTA REGLA VALIDA, QUE CUANDO SEA UN SEPARADOR, "MEMLIB" DEBERÁ SER "DUMMY"			B	ALTO	1		1					
		QA10D-30.059	QAC-0753	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE VARIABLES CONTROL-M	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE VARIABLES CONTROL-M			B	ALTO	1		1					
		QA10D-30.060	QAC-0754	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE CONDICIONES DE ENTRADA	VALIDA UN JOBNAME Y SI SE TRATA, DE UN SEPARADOR, NO DEBERÁ TENER CONDICIONES DE ENTRADA DECLARADAS			B	ALTO	1		1					
		QA10D-30.061	QAC-0755	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE CONDICIONES DE SALIDA	VALIDA UN JOBNAME Y SI SE TRATA, DE UN SEPARADOR, NO DEBERÁ TENER CONDICIONES DE SALIDA DECLARADAS			B	ALTO	1		1					
		QA10D-30.062	QAC-0756	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE CONTROLES	VALIDA UN JOBNAME Y SI SE TRATA, DE UN SEPARADOR, NO DEBERÁN CODIFICARSE CONTROLES			B	ALTO	1		1					
		QA10D-30.063	QAC-0757	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE RECURSOS DE CONTROL	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE RECURSOS DE CONTROL			B	ALTO	1		1					
		QA10D-30.064	QAC-0758	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE STEP RANGE	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE STEP-RANGE			B	ALTO	1		1					
		QA10D-30.065	QAC-0759	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE ACCIONES ON PGMST	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE "ON PGMST"			B	ALTO	1		1					
		QA10D-30.066	QAC-0760	CUANDO JOBNAME ES UN SEPARADOR NO DEBEN CODIFICARSE MENSAJES	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE "DO SHOUT MSG", "MENSAJES"			B	ALTO	1		1					
		QA10D-30.067	QAC-0761	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE PRIORITY	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE PRIORIDADES			B	ALTO	1		1					
		QA10D-30.068	QAC-0762	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE CALENDARIO DCAL	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE CALENDARIOS			B	ALTO	1		1			1		
		QA10D-30.069	QAC-0763	NO DEBE CODIFICARSE CALENDARIO DE CONFIRMACION (CONFCAL)	VALIDA UN JOBNAME Y SI SE TRATA DE UN SEPARADOR, NO DEBERAN CODIFICARSE "CONFCAL" CONFIRMACION DE CALENDARIO.			B	ALTO	1		1			1		
		QA10D-30.070	QAC-0765	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE CONDICION	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR EN EL CAMPO AND/OR.			B	ALTO	1		1					
		QA10D-30.071	QAC-0766	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE CALENDARIO SEMANAL	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR CALENDARIO SEMANAL			B	ALTO	1		1			1		
		QA10D-30.072	QAC-0767	CUANDO JOBNAME ES UN SEPARADOR TIME FROM NO DEBE CODIFICARSE	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR "TIME FROM"			B	ALTO	1		1			1		
		QA10D-30.073	QAC-0768	CUANDO JOBNAME ES UN SEPARADOR, EL PARAMETRO UNTIL NO DEBE CODIFICARSE	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR EL PARAMETRO "UNTIL"			B	ALTO	1		1			1		
		QA10D-30.074	QAC-0769	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE CONFIRM	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR EL PARAMETRO "CONFIRM"			B	ALTO	1		1	1				
		QA10D-30.075	QAC-0770	CUANDO JOBNAME ES UN SEPARADOR, EL PARAMETRO DUE OUT NO DEBE CODIFICARSE	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR EL PARAMETRO "DUE OUT"			B	ALTO	1		1					
		QA10D-30.076	QAC-0771	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE SAC	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR EL PARAMETRO "SAC"			B	ALTO	1		1					
		QA10D-30.077	QAC-0772	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE D-CAT	CALIFICA, QUE SI ES UN SEPARADOR, NO DEBERA CODIFICAR EL PARAMETRO "D-CAT"			B	ALTO	1		1			1		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automatico 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10D-30.078	QAC-0773	DEBE CODIFICARSE DESCRIPCION DEL OBJETIVO DEL JOB	VALIDARÁ, QUE SE HAYA CODICADO ALGUN TEXTO EN EL CAMPO "DESCRIPCION"			B	ALTO	1		1					
		QA10D-30.079	QAC-0774	NO DEBE CODIFICARSE BIBLIOTECA OVERLIB	VALIDARÁ, QUE NO SE HAYA CODICADO ALGUN TEXTO EN EL CAMPO "OVERLIB"			B	ALTO	1		1					
		QA10D-30.080	QAC-0775	MIEMBRO DE DOCUMENTACION DEBE SER IGUAL AL JOBNAME	DE NO SER UN SEPARADOR, VALIDARÁ QUE EL MIEMBRO DE LA DOCUMENTACION SEA IGUAL AL JOBNAME			B	ALTO	1		1			1		
		QA10D-30.081	QAC-0776	BIBLIOTECA DE DOCUMENTACION DEBE SER VALIDA.	VALIDA, QUE DE NO SER UN SEPARADOR, Y SI LA ETIQUETA "MEMNAME", NO ESTA CONTENIDA EN LA LISTA "LG6IOAPR", Y SI LA ETIQUETA "DOCLIB", NO SE ENCUENTRA EN LA LISTA "L63281", ENTONCES, EMITA UN MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.082	QAC-0777	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE MAXDAYS	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "MAXDAYS", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1			1		
		QA10D-30.083	QAC-0778	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE MAXRUNS	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "MAXRUNS", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1		1	1		
		QA10D-30.084	QAC-0779	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE PREVENT-NCT2	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "PREVENT_NCT2", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1			1		
		QA10D-30.085	QAC-0780	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE MAXRERUN	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "MAXRERUN", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1		1	1		
		QA10D-30.086	QAC-0781	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE RERUN MEMBER	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "RERUNMEM", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1					
		QA10D-30.087	QAC-0784	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE SYSOUT OP	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "SYSOUT_OP", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1					
		QA10D-30.088	QAC-0784	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE FROM CLASS DE SYSOUT OP	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "SYSOUT_FROM", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1					
		QA10D-30.089	QAC-0787	EN JOBS SEPARADORES NO CODIFICAR RETENTION OF DAYS TO KEEP	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "RETENTION_#_OF_DAYS_TO_KEEP", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1		1	1		
		QA10D-30.090	QAC-0788	CUANDO JOBNAME ES UN SEPARADOR, MAXWAIT DEBE SER IGUAL A "00".	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "MAXWAIT", DEBERÁ SER "00"			B	ALTO	1		1		1			
		QA10D-30.091	QAC-0789	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE DAYS	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "CTM.DAYS", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1		1			
		QA10D-30.092	QAC-0790	CUANDO JOBNAME ES UN SEPARADOR NO DEBE CODIFICARSE WDAYS	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "WDAYS", NO DEBERÁDE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1		1			
		QA10D-30.093	QAC-0791	CALENDARIO DCAL DEBE SER VALIDO.	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "DCAL", DEBERÁ CONTENER UN VALOR CORRECTO			B	ALTO	1		1		1	1		
		QA10D-30.094	QAC-0792	CALENDARIO WCAL DEBE SER VALIDO	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "WCAL", DEBERÁ CONTENER UN VALOR CORRECTO			B	ALTO	1		1		1	1		
		QA10D-30.095	QAC-0793	CALENDARIO CONFCAL DEBE SER VALIDO	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "CONFCAL", DEBERÁ CONTENER UN VALOR CORRECTO			B	ALTO	1		1		1	1		
		QA10D-30.096	QAC-0794	NO SE ACEPTAN CODES>4 EN PRODUCCION	VALIDA, QUE QUE EL CODIGO DE TERMINACION, NO SEA MAYOR DE "0004", DE LO CONTRARIO EMITIRA UN MENSAJE DE ERROR			B	ALTO	1		1			1		
		QA10D-30.097	QAC-0795	BIBLIOTECA INVALIDA EN FORCEJOB	VALIDA QUE EL CAMPO "DO_FORCEJOB_LIBRARY", CONTENGA LA BIBLIOTECA CORRECTA			B	ALTO	1		1					
		QA10D-30.098	QAC-0796	MEMNAME(1A1) DEBE CODIFICAR PAIS VALIDO	VALIDA, QUE DE NO SER UN SEPARADOR, REVISE EL CAMPO "MEMNAME" Y LA PRIMERA POSICION DEL "MEMNAME", DEBERÁ SER EL PAÍS			B	ALTO	1		1			1		
		QA10D-30.099	QAC-0797	MEMNAME(2A3) DEBE CODIFICAR APLICACION VALIDA	VALIDA QUE DE NO SER SEPARADOR, ENTONCES AL CAMPO "MEMNAME", SE LE HAGA UN "SUBSTRING", A 2 POSICIONES QUE DEBERÁ DE SER LA APLICACIÓN			B	ALTO	1		1			1		
		QA10D-30.100	QAC-0798	MEMNAME(4A4) DEBE CODIFICAR ENTORNO VALIDO	VALIDA QUE DE NO SER SEPARADOR, ENTONCES DEL CAMPO "MEMNAME", SE LE HAGA UN "SUBSTRING", DE LA CUARTA POSICION , EL CUAL DEBERÁ SER EL ENTORNO			B	ALTO	1		1			1		
		QA10D-30.101	QAC-0799	MEMNAME(5A5) DEBE CODIFICAR PERIODICIDAD VALIDA	DE NO SER UN SEPARADOR, VALIDA QUE LA QUINTA POSICION DEL "MEMNAME", SEA LA PERIODICIDAD			B	ALTO	1		1			1		
		QA10D-30.102	QAC-0800	MEMNAME DEBE SER DE 8 POSICIONES	VALIDA, QUE DE NO SER DE 8 POSICIONES EL CAMPO "MEMNAME", EMITA UN MENSAJE DE ERROR.			B	ALTO	1		1			1		
		QA10D-30.103	QAC-0801	NO ESTA PERMITIDO EL USO DEL CAMPO DATES	SI EL CAMPO "DATES", TRAE CODIFICADA ALGUN TIPO DE INFORMACION, DEBERÁ DE EMITIR UN MENSAJE DE ERROR			B	MEDIO	1		1			1		
		QA10D-30.104	QAC-0803	EN PROCESOS DIARIOS MAXWAIT=03 ES OBLIGATORIO	VALIDA QUE DE NO SER SEPARADOR, ENTONCES SI EL PROCESO ES DIARIO, SERÁ OBLIGATORIO EL "MAXWAIT=03"			B	ALTO	1		1		1	1		
		QA10D-30.105	QAC-0804	EN PROCESOS DIFERENTES A EVENTUAL/DIARIO MAXWAIT=03 ES OBLIGATORIO	PARA PROCESOS DIFERENTES A EVENTUALES O DIARIO, SERÁ OBLIGATORIO CODIFICAR "MAXWAIT=03"			B	ALTO	1		1		1	1		
		QA10D-30.106	QAC-0805	NO ESTA PERMITIDO CODIFICAR CONFIRM=Y	ESTA PROHIBIDO CODIFICAR "CONFIRM=Y"			B	ALTO	1		1	1				
		QA10D-30.107	QAC-0807	EL CAMPO TIME ZONE NO DEBE CODIFICAR INFORMACION	ÚNICAMENTE VALIDA QUE EL CAMPO "TIME ZONE", NO CONTENGA INFORMACION, DE LO CONTRARIO, EMITIRÁ UN MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.108	QAC-0808	OWNER PERMITIDO SOLO PARA APLICACIONES CASA DE BOLSA	VALIDA EL "OWNER", EL CUAL SERÁ PERMITIDO SOLO PARA LAS APLICACIONES CASA DE BOLSA			B	ALTO	1		1			1		
		QA10D-30.109	QAC-0809	PROHIBIDO EL USO DEL DO SHOUT WHEN	VALIDA DE DE EXISTIR UN "DO SHOUT WHEN" CODIFICADO, EMITA UN MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.110	QAC-0810	POSICION 1 A LA 3 DE LA CONDICION DE SALIDA DEBE SER IGUAL A LAS PRIMERAS 3 POSICIONES DEL JOBNAME	POSICION 1 A LA 3 DE LA CONDICION DE SALIDA DEBE SER IGUAL A LAS PRIMERAS 3 POSICIONES DEL JOBNAME			B	ALTO	1		1			1		
		QA10D-30.111	QAC-0811	SE DEBE CODIFICAR AL MENOS UNA VEZ INITNOLINEA EN EL RESOURCE	SI NO ES UN SEPARADOR, VALIDA QUE SE DEBE CODIFICAR AL MENOS UNA VEZ INITNOLINEA EN EL RESOURCE			B	ALTO	1		1			1		
		QA10D-30.112	QAC-0812	CODIFICAR AL MENOS UN CALENDARIO EN DCAL, WCAL O CONFCAL	VALIDA LAS ETIQUETAS "DCAL", "WCAL", y "CONFCAL", y DEBERÁN DE TRAER CODIFICADOS POR LO MENOS UN CALENDARIO, DE LO CONTRARIO EMITIRÁ UN MENSAJE DE ERROR			B	ALTO	1		1		1	1		
		QA10D-30.113	QAC-0813	NO ESTA PERMITIDO EL USO DE VARIABLES TIPO CONTROL-M	SI NO ES SEPARADOR, NO ESTA PERMITIDO EL USO DE VARIABLES TIPO CONTROL-M			B	ALTO	1		1					
		QA10D-30.114	QAC-0814	JOB CICLICOS DEBEN CODIFICAR ONPGM=ANYSTEP, CODE=NOTOK, STOPCYC	VALIDA, QUE SI EL JOB ES CICLICO, DEBERAN DE CODIFICAR ONPGM=ANYSTEP, CODE=NOTOK, STOPCYCL			B	ALTO	1		1					
		QA10D-30.115	QAC-0815	EN EVENTUALES, LA CONDICION DE SALIDA POSICIONES DE 1 A 4 DEBE SER EVE	SI NO ES SEPARADOR, DEBERA DE CODIFICARSE "EVE_", EN LAS CONDICIONES DE SALIDA, DE LO CONTRARIO EMITIRÁ UN MENSAJE DE ERROR			B	ALTO	1		1			1		
		QA10D-30.116	QAC-0816	EN EVENTUALES, PRIMER CARÁCTER DE NOMBRE DE JOB DEBE SER LA CLAVE DEL PAIS A UNA POSICION	VALIDA QUE SI SE CODIFICA UNA BIBLIOTECA EVENTUAL, Y NO ES SEPARADOR O APLICAION "RAR", DEBERA IR EN PRIMERA POSICION DEL "MEMNAME", EL CARÁCTER DEL PAÍS			B	ALTO	1		1			1		

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automatico 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10D-30.117	QAC-0817	PARA EVENTUALES POSICIONES 2 Y 3 DE JOB CORRESPODEN A LA APLICACION	VALIDA QUE SI SE CODIFICA UNA BIBLIOTECA EVENTUAL, Y NO ES SEPARADOR LAS POSICIONES 2-3, CORRESPONDERAN A LA APLICACIÓN			B	ALTO	1		1			1		
		QA10D-30.118	QAC-0818	PARA EVENTUALES LA CUARTA POSICION DEL JOB ES P	VALIDA QUE SI SE CODIFICA UNA BIBLIOTECA EVENTUAL, Y NO ES SEPARADOR LA CUARTA POSICION SEA UNA "P"			B	ALTO	1		1			1		
		QA10D-30.119	QAC-0819	CODIFICAR BIBLIOTECA DOCLIB VALIDA PARA PROCESOS EVENTUALES.	VALIDA QUE PARA PROCESOS EVENTUALES SE CODIFIQUE BILBIOTECA "DOCLIB" CORRECTA			B	ALTO	1		1			1		
		QA10D-30.120	QAC-0820	OWNER INVALIDO	REGLA QUE VALIDA, QUE DE NO SER UN SEPARADOR O UN APLICATIVO "RAR", SE CODIFIQUE UN "OWNER" CORRECTO			B	ALTO	1		1			1		
		QA10D-30.121	QAC-0821	PARA EVENTUALES EL GRUPO DEBER EMPEZAR CON "EVE_".	SI NO ES UNA APLICACIÓN "RAR", EL GRUPO DEBERÁ DE COMENZAR CON "EVE_"			B	ALTO	1		1			1		
		QA10D-30.122	QAC-0822	EN TABLAS EVENTUALES CONSIDERAR CVE DE PAIS A POSICIONES 1 A 2	REGLA QUE VALIDA LA CLAVE DEL PAÍS EN LAS PRIMERAS 2 POSICIONES, PARA TABLAS EVENTUALES			B	ALTO	1		1			1		
		QA10D-30.123	QAC-0823	EN TABLAS EVENTUALES CONSIDERAR CLAVE DE APLICACION EN POSICIONES 3 Y 4	EN TABLAS EVENTUALES, VALIDA QUE DE NO SER APLICACIÓN PLATAFORMA GRUPO, PREGUNTA SI ES "RAR", EN POSICIONES 3-4, DE NO SER ASI PREGUNTA POR UNA APLICACIÓN VALIDA O EMITIRÁ UN MSG. DE ERROR			B	ALTO	1		1			1		
		QA10D-30.124	QAC-0824	EN TABLAS EVENTUALES CONSIDERAR CLAVE DE NEGOCIO EN POSICION 5.	VALIDA QUEN DE NO SER UNA APLICACIÓN TIPO GRUPO, PREGUNTA EL TIPO DE NEGOCIO, EN LA QUINTA POSICION, DE NO SER EL CORRECTO EMITIRÁ MENSAJE DE ERROR, PARA TABLAS EVENTUALES			B	ALTO	1		1			1		
		QA10D-30.125	QAC-0825	PROCESOS EVENTUALES NO DEBEN LLEVAR CALENDARIO	REGLA QUE POR PERTENECER AL PERFIL DE EVENTUALES, NO DEBERÁ DE CODIFICARSE CALENDARIO ALGUNO			B	ALTO	1		1		1	1		
		QA10D-30.126	QAC-0826	EN EVENTUALES, CODIFICAR "N" EN TODOS LOS CAMPOS MONTH	PERFIL EVENTUAL, DE NO SER UN SEPARADOR SE VALIDARÁ QUE TRAIGA CODIFICADO UNA "N" EN EL CAMPO "MONTH"			B	ALTO	1		1		1	1		
		QA10D-30.127	QAC-0827	EN TABLAS EVENTUALES LOS JOBS CODIFICAN 9 EN LA QUINTA POSICION	PERFIL EVENTUAL, VALIDA LA BIBLIOTECA EN EL CAMPO "MEMLIB", Y DEBERÁ DE CODIFICAR UN "9" EN LA QUINTA POSICION			B	ALTO	1		1			1		
		QA10D-30.128	QAC-0828	TABLAS EVENTUALES CODIFICAN "EV" EN POSICIONES 6 Y 7 DE LA TABLA	VALIDA LA ETIQUETA "TABLE", EN POSICIONE 6-7, DEBERA DE CODIFICAR "EV", PARA PROCESOS EVENTUALES			B	ALTO	1		1			1		
		QA10D-30.129	QAC-0829	EN EVENTUALES, EL CAMPO TIME DEBE SER MAYOR A 17:00 Y MENOR A LAS 05:00	EN MALLAS EVENTUALES, VALIDA EL CAMPO "TIME_FROM", NO DEBERÁ DE SER MAYOR A 17:00 Y MENOR A LAS 05:00, EN CASO CONTRARIO EMITIRÁ MSG. ERROR			B	ALTO	1		1		1	1		
		QA10D-30.130	QAC-0830	EN EVENTUALES SOLO SE CODIFICA TIME O CONDICION DE ENTRADA	SI NO ES SEPARADOR, NI ES APLICACIÓN "RAR", VERIFICA QUE SE DEBERÁ DE CODIFICAR O "TIME" O CONDICION DE ENTRADA "IN", DE LO CONTRARIO EMITIRÁ MENSAJE DE ERROR			B	ALTO	1		1		1	1		
		QA10D-30.131	QAC-0831	EN EVENTUALES AL MENOS DEBE EXISTIR TIME O CONDICION DE ENTRADA	SI NO ES SEPARADOR, VERIFICA QUE SE DEBERÁ DE CODIFICAR O "TIME" O CONDICION DE ENTRADA "IN", DE LO CONTRARIO EMITIRÁ MENSAJE DE ERROR			B	ALTO	1		1		1	1		
		QA10D-30.132	QAC-0832	PARA PROCESOS CICLICOS ES NECESARIO CODIFICAR AL MENOS UNA VEZ EL PARAMETRO CODES=NOTOK	VALIDA, QUE DE NO SER UN SEPARADOR, CALIFIQUE EL CAMPO "TASKTYPE", Y DE SER CICLICO, VALIDARÁ EL CAMPO "ON_CODES", Y SE CODIFICARÁ EL PARÁMETRO CODES=NOTOK			B	ALTO	1		1			1		
		QA10D-30.133	QAC-0837	NOMBRE DE TABLA, LA PRIMERA POSICION DEBE SER LA CLAVE DEL PAIS A 1 CARACTER	VALIDA QUE DE SER UNA APLICACION PLATAFORMA GRUPO, EL NOMBRE DE LA TABLA DEBERÁ DE COMENZAR EN LA PRIMERA POSICION, CON LA CLAVE DEL PAÍS.			B	ALTO	1		1			1		
		QA10D-30.134	QAC-0838	NOMBRE DE TABLA POSICION 8 NEGOCIO VALIDO PARA PLATAFORMA GRUPO	REGLA QUE VALIDA QUE LA POSICION 8 DEL NOMBRE SEA EL NEGOCIO, Y SOLO PARA PLATAFORMA GRUPO			B	ALTO	1		1			1		
		QA10D-30.135	QAC-0839	NOMBRE DE LA TABLA POSICIONES DE LA 2 A LA 4 CORRESPONDE A UN PREFIJO DE APLICACIÓN VALIDA	DE NO SER EVENTUAL, VERIFICA LAS POSICIONES 2-4, DEL NOMBRE DE LA TABLA, DE SER DIFERENTE EMITIRÁ MENSAJE DE ERROR			B	ALTO	1		1			1		
		QA10D-30.136	QAC-0840	POSICIONES 2-4 DEBE CORRESPONDER A UN PREFIJO DE APLICACION PERMITIDO	VALIDA, QUE DE NO SER UN SEPARADOR, CALIFICA QUE SEA CORRECTA LA INF. CODIFICADA EN LAS POSICIONES 2-4, DEL "MEMNAME", DE LO CONTRARIO EMITA MENSAJE DE ERROR			B	ALTO	1		1			1		
		QA10D-30.137	QAC-0841	MEMNAME(5A5) DEBE CODIFICAR UN TIPO DE JOB VALIDO	VALIDA QUE SI NO SE TRATA DE UN SEPARADOR, CALIFIQUE EN LA POSICION 5, DEL "MEMNAME", SEA UN TIPO DE JOB VALIDO			B	ALTO	1		1			1		
		QA10D-30.138	QAC-0842	VALIDA POSICIONES 1-4 DE CONDICION DE SALIDA CON POSICIONES 1-4 DE NOMBRE DE JOB	SI NO SE TRATA DE UN SEPARADOR VALIDARÁ QUE LAS POSICIONES 1-4, TANTO DE LAS CONDICIONES DE SALIDA, COMO DEL NOMBRE DEL QUE SEAN IGUALES			B	ALTO	1		1			1		
		QA10D-30.139	QAC-0843	NOMBRE DE TABLAS POSICIONES 6 Y 7 DEBE SER UNA PERIODICIDAD VALIDA	SI EL NOMBRE DE LA TABLA, POSICIONES 1-2, NO ES EVENTUAL O POSICIONES 3-4 NO ES "HA", O NO ES "RAR", Y SI EN POSICIONES 6-7, NO ES LA PERIODICIDAD CORRECTA, ENTONCES EMITIRÁ UN MSG. DE ERROR			B	ALTO	1		1		1	1		
		QA10D-30.140	QAC-0844	JOB CICLICOS QUE NO SON DIARIOS MAXWAIT=02 O 03 ES OBLIGATORIO	SI NO ES SEPARADOR PERO ES UN JOB CICLICO, ENTONCES VALIDARA EL CAMPO "MAXWAIT", Y LOS VALORES (02 o 03), SERÁN OBLIGATORIOS			B	ALTO	1		1		1	1		
		QA10D-30.141	QAC-0845	SE DEBE CODIFICAR AL MENOS UNA VEZ EL RESOURCE INITNOLINEA	DE NO SER UN SEPARADOR CALIFICARA AL PROCESO, EL CUAL DEBERÁ DE TRAER CODIFICADO AL MENOS UNA VEZ EN EL "RESOURCE" INITNOLINEA			B	ALTO	1		1			1		
		QA10D-30.142	QAC-0846	SE DEBE CODIFICAR AL MENOS UNA VEZ EL RESOURCE "INICIADOR"	DE NO SER SEPARADOR, VALIDA QUE EN EL CAMPO "RESOURCE", ESTE CODIFICADO CORRECTAMENTE, DE ACUERDO AL CONTENIDO DE LA LISTA "LG6INICI", EN CASO CONTRARIO EMITIRÁ MENSAJE DE ERROR			B	ALTO	1		1			1		
		QA10D-30.143	QAC-0847	MAXWAIT NO PERMITIDO PARA MALLAS MENSUALES DE LA APLICACIÓN "GY"	VALIDA QUE SI ES UNA APLICACIÓN "GYM", MAXWAIT NO PERMITIDO PARA MALLAS MENSUALES			B	ALTO	1		1		1	1		
		QA10D-30.144	QAC-0848	EL JCL NO CORRESPONDE AL PREFIJO DE APLICACION DE LA MALLA	SI NO ES UN SEPARADOR, VALIDA QUE LAS PRIMERAAS 4 POSICIONES, TANTO DE LA TABLA COMO DEL JOB, SEAN IGUALES, DE LO CONTRARIO EMITIRÁ UN MENSAJE DE ERROR			B	ALTO	1		1			1		
		QA10D-30.145	QAC-0851	EN PROCESOS EVENTUALES MAXWAIT=03 ES OBLIGATORIO	DE NO SER SEPARADOR, ENTONCES VALIDA SI LA QUINTA POSICION ES UN "9", SE TRATARÁ DE UN PROCESO EVENTUAL QUE REQUERIRÁ DE UN "MAXWAIT=03"			B	ALTO	1		1		1	1		
		QA10D-30.146	QAC-0905	CUANDO JOBNAME ES UN SEPARADOR CODIFICAR N EN LOS CAMPOS MONTH	VALIDA QUE DE SER UN SEPARADOR, LA ETIQUETA "MONTHS", DEBERÁ CONTENER UNA "N"			B	ALTO	1		1		1	1		
		QA10D-30.147	QAC-0906	NO CODIFICAR OPER** EN EL CAMPO DO SHOUT WHEN TO	PROHIBIDO CODIFICAR OPER** EN EL CAMPO "SHOUT_WHEN_TO"			B	ALTO	1		1			1		
		QA10D-30.148	QAC-0907	INSTREAM-JCL DEBE SER IGUAL "N"	VALIDA, QUE DE NO SER UN SEPARADOR, EL CAMPO "INSTREAM_JCL", DEBERÁ DE SER "N"			B	ALTO	1		1					
		QA10D-30.149	QAC-0908	EL GRUPO ES UN CAMPO REQUERIDO	VALIDA QUE DE NO SER UN SEPARADOR, ES REQUERIDO QUE SE CODIFQUE EL CAMPO "GROUP"			B	ALTO	1		1					
		QA10D-30.150	QAC-0909	EN CONDICION DE ENTRADA NO SE PERMITE CODIFICAR "+"	VALIDA, QUE PARA LAS CONDICIONES DE ENTRADA, VENGA CODIFICADO EL SIGNO "+"			B	ALTO	1		1					
		QA10D-30.151	QAC-0910	TABLE DE FORCEJOB NO DEBE SER IGUAL AL TABLE NAME	VALIDA, QUE EL NOMBRE DE LA "FORCEJOB LIBRARY" NO SEA IGUAL A LA "FORCEJOB TABLE", DE LO CONTRARIO EMITIRA MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.152	QAC-0911	ODATE NO PERMITIDO EN CONDICION DE ENTRADA	VALIDA EL PARAMETRO "DATES", NO ESTA PERMITIDO EN CONDICIONES DE ENTRADA			B	ALTO	1		1					
		QA10D-30.153	QAC-0912	NO ESTA PERMITIDO CODIFICAR INFORMACION EN CAMPO SYSTEM-ID	NO ESTA PERMITIDO CODIFICAR INFORMACION EN CAMPO SYSTEM-ID			B	ALTO	1		1					

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		QA10D-30.154	QAC-0913	DEBE CODIFICAR CALENDARIO FERIADOS,	REGLA EXCLUSIVA CASA DE BOLSA VALIDA QUE EL JOB CBUDND51 SE ASIGNE UNICAMENTE CALENDARIO FESTIVO			B	ALTO	1		1		1	1		
		QA10D-30.155	QAC-0914	NO ESTA PERMITIDO EL USO DEL CALENDARIO "FERIADOS" DCAL, WCAL.CONFCAL	REGLA QUE VALIDA QUE EL CALENDARIO FERIADOS LO OCUPE UNICAMENTE EL JOB CBUDND51			B	ALTO	1		1		1	1		
		QA10D-30.156	QAC-0935	EN PROCESOS CYCLICOS DIARIOS MAXWAIT=00 ES OBLIGATORIO	VALIDA QUE SI NO ES UN SEPARADOR, PARA PROCESOS CICLICOS EL PARÁMETRO "MAXWAIT=00", SERÁ OBLIGATORIO			B	ALTO	1		1		1	1		
		QA10D-30.157	QAC-0936	MAXWAIT NO PERMITIDO PARA LA APLICACIÓN "GY"	VALIDA QUE LA CODIFICACIÓN EN POSICIONES 2-4, QUE SEA UNA APLICACIÓN "GYM", Y SEA UNA MAXWAIT PERMITIDO			B	ALTO	1		1		1	1		
		QA10D-30.158	QAC-0937	EN FORCE_JOB EL PRIMER CARACTER DEL NOMBRE DE JOB DEBE SER VALIDO.	VALIDA QUE EL PRIMER CARÁCTER DEL NOMBRE DEL "FORCEJOB_JOB", SEA EL DEL PAIS.			B	ALTO	1		1			1		
		QA10D-30.159	QAC-0938	EN FORCE_JOB EL CUARTO CARACTER DEL NOMBRE DE JOB DEBE SER VALIDO.	VALIDA, QUE EL CUARTO CARÁCTER DEL NOMBRE DEL "FORCEJOB", SEA UNA "P"			B	ALTO	1		1			1		
		QA10D-30.160	QAC-0939	PG POSICIONES 1-1 DEBE CODIFICAR PAIS VALIDO	PERFIL PLATAFORMA GRUPO, DE NO SER SEPARADOR, VERIFICA EL PAÍS, EN LA PRIMERA POSICION DEL "MEMNAME", DE NO SER CORRECTO EMITIRÁ MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.161	QAC-0940	VALIDA POSICIONES 1-3 DE CONDICION DE SALIDA CON POSICIONES 1-3 DE NOMBRE DE JOB	LAS PRIMERAS 3 POSICIONES DE LAS CONDICIONES DE SALIDA DEBERAN DE SER IGUALES A LAS 3 PRIMERAS POSICIONES DEL "JOBNAME", DE LO CONTRARIO EMITIRÁ MENSAJE DE ERROR			B	ALTO	1		1					
		QA10D-30.162	QAC-0941	EN JOBS SEPARADORES NO CODIFICAR OF GENERATIONS TO KEEP	VALIDA QUE DE SER UN SEPARADOR, EN LA ETIQUETA "RETENTION_#_OF_GENERATIONS_TO_KEEP", NO DEBERÁ DE VENIR CODIFICADO TEXTO ALGUNO.			B	ALTO	1		1					
		QA10D-30.163	QAC-0942	CAMPO STAT_CAL NO DEBE CONTENER INFORMACION	VALIDA QUE EL CAMPO STAT CAL NO CONTENGA INFORMACION			B	ALTO	1		1		1			
		QA10D-30.164	QAC-0943	CAMPO DEFINITION_ACTIVE NO DEBE CONTENER INFORMACION	VALIDA QUE EL CAMPO "DEFINITION_ACTIVE_FROM", NO TENGA CODIFICADA INFORMACION, DE LO CONTRARIO EMITIRA MENSAJES DE ERROR			B	ALTO	1		1					
		QA10D-30.165	QAC-0944	EN DO SHOUT EL CAMPO TO DEBE EMPEZAR CON H,M, X, T, O	VALIDA QUE EL CAMPO "DO_SHOUT_TO", COMIENCE CON LA LETRA "H, M, X, T, O"			B	ALTO	1		1					
		QA10D-30.166	QAC-0945	CONTROL NO DEBE HACER USO DEL SUBPARAMETRO K	VALIDA EL ESTATUS OK/NOOK DEL CONTROL (NO USAR K)			B	ALTO	1		1					
		QA10D-30.167	QAC-0946	PROHIBIDO CODIFICAR INFORMACION EN EL CAMPO FROM TIME DAYS	VALIDA QUE EL CAMPO FROM_DAYS NO CONTENGA INFORMACION			B	ALTO	1		1					
		QA10D-30.168	QAC-0947	RESOURCE NO DEBE HACER USO DEL SUBPARAMETRO D	VALIDA EL ESTATUS OK/NOOK DEL RESOURCE(NO USAR D)			B	ALTO	1		1					
		Arquitectura APX															
		QA10D-31.001	QAC-0974	Uso throws en librerías APX	Queda prohibido utilizar la clausula throws en los métodos execute expuestos en las librerías APX			B	ALTO	1					1		
		QA10D-31.002	QAC-0975	Bloque try - catch en invocación a librería APX	Queda prohibido incluir dentro de un bloque try - catch la invocación a una librería APX			B	ALTO	1					1		
		QA10D-31.003	QAC-0976	Bloque try - catch para excepciones no controlables	No se pueden capturar en código aplicativo aquellas excepciones que están indicadas como solo controlables por la Arquitectura			B	ALTO	1					1		
		QA10D-31.004	QAC-0977	Librerías APX sin estado	Queda prohibido tener variables miembro no finales en las librerías.			B	ALTO	1					1		
		QA10D-31.005	QAC-0978	Transacciones APX sin estado	Queda prohibido tener variables miembro no finales en las transacciones.			B	ALTO	1					1		
		QA10D-31.006	QAC-0979	Métodos deprecados en transacciones	Queda prohibido el uso de métodos deprecados de la clase AbstractTransaction			B	ALTO	1					1		
		QA10D-31.007	QAC-0980	Métodos deprecados en librerías APX	Queda prohibido el uso de métodos deprecados de la clase AbstractLibrary			B	ALTO	1					1		
		QA10D-31.008	QAC-0981	Instanciación de librerías APX	No se deben instanciar las librerías mediante el operador "new" de JAVA			B	ALTO	1					1		
		QA10D-31.009	QAC-0982	Uso de caché JPA	Queda prohibido el uso de la caché JPA en APX			B	ALTO	1					1		
		QA10D-31.010	QAC-0983	Acceso a MongoDB	No debe accederse a Mongo utilizando las clases MongoClient, DB y DBCollection propias de MongoDB			B	ALTO	1					1		
		QA10D-31.011	QAC-0984	Obtención de bundles	No debe obtenerse bundle alguno a través del método getBundle de la clase BundleContext			B	ALTO	1					1		
		QA10D-31.012	QAC-0985	Exportación de paquetes en librerías APX (Implementación)	No debe exportarse ningún paquete en el pom.xml del proyecto de implementación de librerías			B	ALTO	1					1		
		QA10D-31.013	QAC-0986	Exportación de paquetes en librerías APX (Interfaz)	El proyecto de interfaz de librerías sólo debe contener el paquete de la interfaz y el de dtos			B	ALTO	1					1		
		QA10D-31.014	QAC-0987	Importación de paquetes de Arquitectura en librerías APX y transacciones	No se permite la importación de las siguientes clases / librerías: - org.drools - org.drools.core			B	ALTO	1					1		
		QA10D-31.015	QAC-0988	Paquetería de componentes aplicativos	Las clases de un aplicativo siempre deben pertenecer a un paquete que cumpla el patrón "com.bbva.<uuaa>"			B	ALTO	1					1		
		QA10D-31.016	QAC-0990	Nomenclatura métodos interfaz	El proyecto de la interfaz de una librería APX solo debe contener en la definición de la interfaz métodos que comiencen por la palabra execute y tengan un máximo de 48 caracteres de longitud			B	ALTO	1					1		
		QA10D-31.017	QAC-0991	Dependencia de librerías APX	La dependencia con una librería APX a la que invocar debe ser declarada en el pom.xml para solicitar siempre la última versión de la misma			B	ALTO	1					1		
SE02C	**Verificación Codificación de Elementos SW en lo relativo a Seguridad de Aplicaciones**																
	SE02C-01	**Validación de datos y parámetros**															
		SE02C-01.001			XSS (Cross Site Scripting) público : Problema de seguridad que ocurre por el envío de datos no validados al navegador del cliente pudiendo ejecutar código malicioso en el mismo. La forma más común de este problema se produce cuando la aplicación usa la entrada de datos de un usuario para generar la salida y esta no se valida. Se encuentra en la parte pública de aplicaciones accesibles desde internet. Referencias: http://www.owasp.org/index.php/Testing_for_Cross_site_scripting http://www.fortify.com/vulncat/en/vulncat/cfml/cross_site_scripting.html http://es.wikipedia.org/wiki/Cross-site_scripting	Código del componente		B	ALTO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		SE02C-01.002			XSS (Cross Site Scripting) privado: Problema de seguridad que ocurre por el envío de datos no validados al navegador del cliente pudiendo ejecutar código malicioso en el mismo. La forma más común de este problema se produce cuando la aplicación usa la entrada de datos de un usuario para generar la salida y esta no se valida. Se encuentra en aplicaciones internas o en la parte privada de aplicaciones accesibles desde internet. Referencias: http://www.owasp.org/index.php/Testing_for_Cross_site_scripting http://www.fortify.com/vulncat/en/vulncat/cfml/cross_site_scripting.html http://es.wikipedia.org/wiki/Cross-site_scripting	Código del componente		E	MODERADO	1							
		SE02C-01.003			Manipulacion de cabeceras: Cookies. Pueden darse diversos problemas de seguridad al incluir datos no validados en las cabeceras HTTP. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/header_manipulation.html http://www.owasp.org/index.php/Testing_for_cookies_attributes_(OWASP-SM-002) http://es.wikipedia.org/wiki/Cookies#Inconvenientes_de_las_cookies	Código del componente		E	BAJO	1							
		SE02C-01.004			SQL Inyection: El uso de datos de entrada para la construcción dinámica de determinadas sentencias SQL puede permitir la ejecución de otras sentencias o comandos SQL no deseados. Validar correctamente la entrada de datos para la construcción de estas sentencias. Se encuentra en la parte pública de aplicaciones accesibles desde internet. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/sql_injection.html http://www.owasp.org/index.php/Testing_for_SQL_Injection_(OWASP-DV-005) http://es.wikipedia.org/wiki/SQL_injection	Código del componente		B	ALTO	1							
		SE02C-01.005			SQL Inyection: El uso de datos de entrada para la construcción dinámica de determinadas sentencias SQL puede permitir la ejecución de otras sentencias o comandos SQL no deseados. Validar correctamente la entrada de datos para la construcción de estas sentencias. Se encuentra en aplicaciones internas o en la parte privada de aplicaciones accesibles desde internet. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/sql_injection.html http://www.owasp.org/index.php/Testing_for_SQL_Injection_(OWASP-DV-005) http://es.wikipedia.org/wiki/SQL_injection	Código del componente		E	MODERADO	1							
		SE02C-01.006			Ejecución de comandos: Determinadas ejecuciones desde orígenes o entornos inseguros puede causar que las aplicaciones ejecuten comandos maliciosos, por suplantación del comando original o del entorno en el que éste se ejecuta. Referencias: http://www.owasp.org/index.php/Process_Control http://www.fortify.com/vulncat/en/vulncat/cfml/process_control.html	Código del componente		E	ALTO	1							
		SE02C-01.007			Desbordamiento de buffer: Se puede forzar a la aplicación para que escriba fuera de lo límites de memoria reservada pudiendo causar errores o comportameitnos inesperados. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/buffer_overflow.html http://www.owasp.org/index.php/Buffer_Overflow http://es.wikipedia.org/wiki/Desbordamiento_de_búfer	Código del componente		E	MODERADO	1							
		SE02C-01.008			Denegación de Servicio (DoS): La aplicación es vulnerable a ataques de denegación de servicio, puede provocar un error que deja indisponible la aplicación a los usuarios. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/denial_of_service.html http://www.owasp.org/index.php/Application_Denial_of_Service http://es.wikipedia.org/wiki/Denegación_de_servicio	Código del componente		B	ALTO	1							
		SE02C-01.009			LDAP Injection: Construir sentencias LDAP dinámicas con los datos de entrada de los usuarios puede permitir que se modifiquen las sentencias y conseguir resultados no permitidos. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/ldap_injection.html http://www.owasp.org/index.php/LDAP_injection	Código del componente		E	MODERADO	1							
		SE02C-01.010			Path Manipulation: El control sobre las rutas (path) en operaciones sobre filesystems permite accesos o modificaciones sobre recursos protegidos del sistema. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/path_manipulation.html http://www.owasp.org/index.php/Path_Manipulation	Código del componente		E	MODERADO	1							
		SE02C-01.011			Incorrecta validación XML (Missing XML Validation): Una incorrecta validación XML puede permitir que se produzcan entradas maliciosas e inesperadas. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/missing_xml_validation.html http://www.owasp.org/index.php/Missing_XML_Validation	Código del componente		E	MODERADO	1							
		SE02C-01.012			Subida de ficheros: En la aplicación se permite la subida de ficheros, lo que puede provocar inyección de contenido o ejecución de código malicioso en el servidor. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/often_misused_file_upload.html http://www.owasp.org/index.php/Unrestricted_File_Upload	Código del componente		B	ALTO	1							
		SE02C-01.013			Subida de ficheros: En la aplicación se permite la subida de ficheros, lo que puede provocar inyección de contenido o ejecución de código malicioso en el servidor. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/often_misused_file_upload.html http://www.owasp.org/index.php/Unrestricted_File_Upload	Código del componente		E	BAJO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		SE02C-01.014			Ficheros o código en desuso: Se deben controlar los ficheros incluídos dinámicamente en un JSP, ficheros antigûos o en desuso, una mala gestión de este problema puede provocar la ejecución de código malicioso. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/dangerous_file_inclusion.html http://www.owasp.org/index.php/Testing_for_Old,_Backup_and_Unreferenced_Files_(OWASP-CM-006)	Código del componente		R	BAJO	1							
		SE02C-01.015			Control de variables (Value Shadowing): La aplicación accede a las variables de manera ambigüa (variables diferentes con el mismo nombre por ejemplo), lo que puede provocar comportamientos y accesos no deseados. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/value_shadowing.html	Código del componente		R	BAJO	1							
		SE02C-01.016			Inyección XML: Escribir datos no validados en un documento XML puede permitir que se hagan cambios en la estructura y contenidos del XML. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/xml_injection.html http://www.owasp.org/index.php/Testing_for_XML_Injection_(OWASP-DV-008)	Código del componente		E	MODERADO	1							
		SE02C-01.017			Inyección Xpath: Construir una sentencia Xpath dinámica con la entrada de los usuarios puede permitir que se modifique la sentencia y devuelva resultados inesperados. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/xpath_injection.html http://www.owasp.org/index.php/XPATH_Injection	Código del componente		E	MODERADO	1							
		SE02C-01.018			Inyección de código: Determinadas faltas de validación en los datos puede puede permitir que se inyecte y se ejecute código malicioso. Referencias: http://www.owasp.org/index.php/Code_Injection http://www.fortify.com/vulncat/en/vulncat/java/dynamic_code_evaluation_code_injection.html	Código del componente		B	ALTO	1							
		SE02C-01.019			Open Redirect: Se deben validar las entradas URL usadas en las redirecciones para evitar ataques de phishing (redirecciones a páginas que no corresponden a BBVA en las que se intenta conseguir información confidencial). Se encuentra en la parte pública de aplicaciones accesibles desde internet. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/open_redirect.html http://www.owasp.org/index.php/Open_redirect	Código del componente		B	ALTO	1							
		SE02C-01.020			Open Redirect: Se deben validar las entradas URL usadas en las redirecciones para evitar ataques de phishing (redirecciones a páginas que no corresponden a BBVA en las que se intenta conseguir información confidencial). Se encuentra en aplicaciones internas o en la parte privada de aplicaciones accesibles desde internet. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/open_redirect.html http://www.owasp.org/index.php/Open_redirect	Código del componente		E	MODERADO	1							
		SE02C-01.021	QAC-0962	Comprobar la entrada de usuario utilizada en consultas DL/I (IMS)	Control de acceso inadecuado, se ejecuta un comando EXEC DLI que permite a un usuario controlar una clave primaria que permite al atacante ver registros no autorizados de DLI. Los errores de acceso DL/I se producen cuando: 1. Los datos de entrada a un programa son de una fuente no fiable. 2. Los datos se utilizan para especificar un argumento para	Código del componente		E	ALTO	1							
		SE02C-01.022	QAC-0964	Evite inyección de código SQL formado a partir de entradas externas no controladas	El software construye parte de un comando SQL concatenando entradas externas, y no neutraliza apropiadamente dichas entradas antes de enviar el SQL resultante a la base de datos. En Cobol las sentencias EXEC SQL suelen estar preparadas ("bind") de modo que las entradas externas se interpretan como datos y no como parte de la sentencia SQL, pero existen ciertos comandos SQL dinámicos (PREPARE, EXECUTE IMMEDIATE), para los cuales la consulta sí se interpreta como literal, y la entrada externa forma parte del SQL que va a lanzarse contra la base de datos. El uso de SQL dinámico es algo que debe evitarse, pero la regla considera violación sólo si el SQL usado incluye entradas externas no neutralizadas. Nota: El uso de procedimientos almacenados NO previene inyección SQL si, por ejemplo, el SQL que se envía a base de datos para la ejecución del procedimiento almacenado se compone concatenando entrada externa no neutralizada. Las formas de evitar inyección SQL son: 1) usar SQL parametrizado ("bound"), de modo que el código SQL y los parámetros de entrada no se mezclan (las entradas no cambian la semántica de la sentencia SQL); 2) Si el código SQL debe depender de entrada externa (e.g. para construir cláusulas WHERE 'dinamicas'), todas las	Código del componente		E	ALTO	1							
		SE02C-01.023	QAC-0965	No permitir que una entrada de usuario pueda controlar campos del descriptor MQSeries	Permitir al usuario controlar campos del descriptor de objetos MQSeries podría habilitar a un atacante el acceso o modificación de recursos MQSeries que de otra forma estarían protegidos. En general, no debe permitirse que datos introducidos por el usuario o datos no confiables controlen valores sensibles. Argumentos sensibles como credenciales o nombres de colas no deben estar controlados por entradas del usuario. Incluso los mensajes no deben contener entradas del usuario que no se hayan validados apropiadamente por llamadas explícitas a rutinas de	Código del componente	Solo para CPD Europa	E	ALTO	1							
		SE02C-01.024	QAC-0968	Evite entradas externas no neutralizadas como parte de una ruta (fichero o directorio) usada en operaciones de E/S	El software usa una entrada externa para construir una ruta que identifica un fichero o directorio bajo una directorio restringido, pero el software no neutraliza la entrada, de forma que un atacante puede lograr que la ruta se resuelva a un directorio fuera del directorio restringido. Bajo Cobol este problema no es frecuente, pero ciertas	Código del componente	Solo para CPD Europa	E	ALTO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		SE02C-01.025	QAC-0973	Buscar acceso a los recursos del sistema donde se accede al recurso podría ser controlado por la entrada de usuario.	Se buscan accesos a recursos del sistema que puedan estar controlados por una entrada del usuario. Un problema de inyección de recurso ocurre cuando se satisfacen las siguientes dos condiciones: 1. Un atacante puede controlar el identificador utilizado para acceder a un recurso del sistema. Por ejemplo, un atacante podría especificar el nombre de una cola CICS, un fichero de datos, etc. 2. Al especificar el recurso, el atacante gana una capacidad que de otra forma no se permitiría. La regla comprueba si el nombre de un recurso del sistema	Código del componente		E	ALTO	1							
	SE02C-02	Autenticación y Autorización															
		SE02C-02.001			Manipulación de parámetros (Unsafe Reflection): La manipulación del valor de los parametros de la aplicación podría modificar el flujo correcto, esto podría permitir saltarse las validaciones de acceso. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/unsafe_reflection.html http://www.owasp.org/index.php/Unsafe_Reflection	Código del componente		E	MODERADO	1							
		SE02C-02.002			Manipulación de recursos: Los parametros introducidos para la identificación de recursos (nombres de fichero, números de puertos, entre otros) debe validarse correctamente. Esto podría permitir el acceso o manipulación de recursos que no fueran correctamente protegidos. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/resource_injection.html http://www.owasp.org/index.php/Resource_Injection	Código del componente		E	MODERADO	1							
		SE02C-02.003			Gestión de credenciales: Los nombres de usuario nunca se deben almacenar en texto plano en los ficheros de configuración y tablas de bases e datos. Esta información puede facilitar que se consigan usuarios para acceso a la aplicación. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/credential_management.html	Código del componente		E	BAJO	1							
		SE02C-02.004			Gestión de credenciales: En el código fuente de la aplicación no deben aparecer nombres de usuario ni claves de acceso, ya que si se consigue acceso a este se facilita que se conozcan usuarios para acceso a la aplicación. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/credential_management_hardcoded_username.html http://www.owasp.org/index.php?title=Hard-Coded_Password&setlang=es	Código del componente		E	BAJO	1							
		SE02C-02.005			Gestión de Contraseñas: Las contraseñas o claves de los usuarios nunca deben almacenarse en claro en los ficheros de configuración y tablas en bases de datos. El acceso a esta información puede comprometer el acceso a el sistema. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/password_management.html http://www.owasp.org/index.php/Password_Plaintext_Storage	Código del componente		B	ALTO	1							
		SE02C-02.006			Gestión de Contraseñas: En el código fuente de la aplicación no deben aparecer contraseñas ni claves de los usuarios, el acceso a este podría facilitar que se obtengan contraseñas que el permitan accesos a la aplicación. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/password_management_hardcoded_password.html http://www.owasp.org/index.php/Use_of_hard-coded_password	Código del componente		B	ALTO	1							
		SE02C-02.007			Gestión de Contraseñas: Los comentarios en el código no deben contener contraseñas o claves de usuario. Esta información puede facilitar el acceso a la aplicación. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/password_management_password_in_comment.html http://www.owasp.org/index.php?title=Hard-Coded_Password&setlang=es	Código del componente		B	ALTO	1							
		SE02C-02.008			Gestión de Contraseñas - Cifrado débil: Los métodos de cifrado de contraseñas deben ser seguros y robustos, con el objetivo de que no se puedan conocer facilmente. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/password_management_weak_cryptography.html http://www.owasp.org/index.php/Password_Management:_Weak_Cryptography	Código del componente		E	BAJO	1							
		SE02C-02.009			Gestión de Contraseñas : Las contraseñas no deben ser enviadas como parte de las peticiones HTTP al hacer redirecciones. Esto puede causar que las contraseñas sean capturadas o visualizadas por terceros en sistemas intermedios. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/password_management_redirect.html	Código del componente		B	ALTO	1							
		SE02C-02.010			Control de Acceso - Consultas anónimas en LDAP: Un control de acceso que no esté correctamente implemtentado puede permitir que se realicen consultas anonimas en el LDAP. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/access_control_anonymous_ldap_bind.html	Código del componente		E	BAJO	1							
		SE02C-02.011			Control de Acceso - Base de Datos: Debe realizarse correctamente el control de acceso para no permitir accesos a registros no autorizados de las bases de datos. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/access_control_database.html	Código del componente		E	BAJO	1							
		SE02C-02.012			Control de Acceso - LDAP: Debe realizarse correctamente el control de acceso un para que no se permita obtener accesos no autorizados y ejecución de comandos contra el LDAP. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/access_control_ldap.html	Código del componente		E	BAJO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=si, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		SE02C-02.013			Violacion de privilegios: La aplicación debe obtener permisos de administrador únicamente para ejecutar comandos que no puedan ser lanzados con otros permisos. Para los demás comandos la aplicación debe ejecutarse con los permisos mínimos necesarios. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/least_privilege_violation.html http://www.owasp.org/index.php/Least_Privilege_Violation	Código del componente		E	BAJO	1							
		SE02C-02.014	QAC-0966	Evitar ocultar una contraseña con una codificación trivial	Ocultar una contraseña u otra información sensible con una codificación trivial no protege la contraseña. Los problemas en la gestión de contraseñas ocurren cuando una contraseña se almancena en texto plano (por ejemplo, en un fichero de configuración). Un programador podría intentar solucionar el problema de gestión de contraseñas ocultando la contraseña con una función de codificación, como base 64. Pero este esfuerzo no protege adecuadamente la contraseña.	Código del componente		E	MODERADO	1							
		SE02C-02.015	QAC-0969	Evitar colocar contraseñas y otra información sensible en los comentarios del código	La existencia de contraseñas en los comentarios del código puede otorgar acceso inintencionado a credenciales sensibles al personal con acceso al código fuente. Esta regla simplemente busca palabras contraseñas específicas en los comentarios del código. Estas palabras se pueden personalizar en la propiedad "pattern".	Código del componente		E	BAJO	1							
		SE02C-02.016	QAC-0971	Las contraseñas escritas en código pueden comprometer la seguridad de un sistema de forma difícilmente remediable	Nunca es una buena idea escribir una contraseña en el código. No sólo permite al resto de los desarrolladores del proyecto verla, también hace que sea muy difícil resolver el problema. Una vez que el código está en producción, la contraseña no puede cambiarse sin parchear el software. Si la cuenta protegida por la contraseña está	Código del componente		E	ALTO	1							
	SE02C-03	**Gestión de Sesiones**															
		SE02C-03.001			Seguridad de las cookies - Cookie no enviada bajo SSL: Las cookies deben enviarse bajo protocolo seguro ya que en caso contrario la información podría ser capturada o visualizada por terceros.	Código del componente		E	BAJO	1							
		SE02C-03.002			Seguridad de las Cookies - "HTTPOnly" no informado: Debe establecerse esta propiedad al crear la cookie sino puede viajar en claro por la red, lo que permitiría que su valor fuera capturado o visualizado. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/cookie_security_httponly_not_set.html http://www.owasp.org/index.php/HTTPOnly	Código del componente		E	BAJO	1							
		SE02C-03.003			Seguridad de las Cookies - Overly Broad Domain: Una mala limitación en el dominio de las cookies puede permitir el acceso a los datos de las cookies desde dominios externos. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/cookie_security_overly_broad_domain.html	Código del componente		R	BAJO	1							
		SE02C-03.004			Seguridad de las Cookies (Overly Broad Path): Una mala configuración en la limitación del path en las cookies puede permitir el acceso a datos de las mismas desde otras aplicaciones alojadas en el mismo dominio. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/cookie_security_overly_broad_path.html	Código del componente		R	BAJO	1							
		SE02C-03.005			Seguridad de las Cookies - Persistencia de las cookie: Se debe realizar una buena configuración del tiempo de expiración de las cookies, sino se facilita el robo de información almacenada en las mismas. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/cookie_security_persistent_cookie.html	Código del componente		E	BAJO	1							
	SE02C-04	**Protección de Información**															
		SE02C-04.001			Violación de privacidad: Se debe realizar una buena gestión de la información privada de los usuarios (números de cuenta, contraseñas, nombres de usuarios, entre otros) para no facilitar que puedan obtenerse. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/privacy_violation.html	Código del componente		R	MODERADO	1						1	
		SE02C-04.002			Cifrado debil: Los metodos de cifrado deben ser robustos para evitar el acceso a información protegida. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/weak_encryption.html http://www.owasp.org/index.php/Top_10_2007-Insecure_Cryptographic_Storage	Código del componente		E	MODERADO	1							
		SE02C-04.003			JavaScript Hijacking (FrameWork vulnerable): La utilización de Microsoft AJAX.NET permite el acceso a los datos enviados mediante JSON entre el cliente y servidor. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/javascript_hijacking_vulnerable_framework.html	Código del componente		R	BAJO	1							
		SE02C-04.004			JavaScript Hijacking (Ad Hoc Ajax): Utilizar javascript para el transporte de datos puede permitir que estos sean capturados o visualizados. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/javascript_hijacking_ad_hoc_ajax.html	Código del componente		R	BAJO	1							
		SE02C-04.005			Campos ocultos: No se debe de almacenar información crítica en los campos ocultos de los formularios. Existen técnicas sencillas para detectar y visualizar esta información. Referencias: http://www.fortify.com/vulncat/en/vulncat/html/hidden_field.html	Código del componente		E	BAJO	1							
		SE02C-04.006	QAC-0967	Evitar escribir información del sistema (típicamente para debug) en código para producción	Revelar datos del sistema o información de debug ayuda al adversario a conocer el sistema y crear un plan de ataque. Por ejemplo, con CICS, evite el comando DUMP TRANSACTION (que escribe todas las areas de almacenamiento relacionadas con tareas, la tabla de control de la terminal y el área de datos especificada). Dependiendo de la configuración del sistema, esta información puede escribirse en la consola, en un fichero de log, o ser expuesta a un usuario remoto. En algunos	Código del componente		E	MODERADO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		SE02C-04.007	QAC-0972	Información expuesta mediante depuración	El programa contiene código bajo depuración, que puede exponer información sensible a usuarios no autorizados. Emite una violación cuando debug está activo en un programa Cobol: Cláusual WITH DEBUGGING MODE en párrafo SOURCE COMPUTER bajo CONFIGURATION	Código del componente		E	MODERADO	1							
	SE02C-05	**Control de errores y excepciones**															
		SE02C-05.001			Control de excepciones: La utilización de la función EnterCriticalSection() puede causar un error en condiciones de poca memoria (suele darse en sistemas operativos anteriores a Windows 2000) Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/often_misused_exception_handling.html http://www.owasp.org/index.php/Top_10_2007-Information_Leakage_and_Improper_Error_Handling	Código del componente		R	BAJO	1							
		SE02C-05.002			Incorrecto control de errores - Excepción no controlada: Un error al capturar una excepción puede causar que la aplicación realice operaciones no controladas. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/unhandled_exception.html http://www.owasp.org/index.php/Top_10_2007-Information_Leakage_and_Improper_Error_Handling	Código del componente		R	BAJO	1							
		SE02C-05.003			Incorrecto control de errores (bloque "catch" vacío): Ignorar las excepciones puede causar que se generen estados o condiciones inesperadas. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/poor_error_handling_empty_catch_block.html http://www.owasp.org/index.php/Top_10_2007-Information_Leakage_and_Improper_Error_Handling	Código del componente		R	BAJO	1							
		SE02C-05.004			Incorrecto control de errores - (bloques "cach" demasiado genéricos): El uso de bloques "catch" que controlen excepciones muy genéricas podría hacer que la aplicación controle errores de forma incorrecta y que deban ser tratados en otro punto. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/poor_error_handling_overly_broad_catch_block.html http://www.owasp.org/index.php/Top_10_2007-A6	Código del componente		R	BAJO	1							
		SE02C-05.005			Incorrecto control de errores (captura de excepción NullReferenceException): No en todos los casos es una buena practica la captura de esta excepción. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/poor_error_handling_program_catches_nullreferenceexception.html http://www.owasp.org/index.php/Top_10_2007-A6	Código del componente		R	BAJO	1							
		SE02C-05.006			Incorrecto control de errores (excepción demasiado genérica): Las excepciones que lanza un método no deben ser demasiado genéricas ya que esto deficulta que se pueda realizar una correcta gestión de errores. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/poor_error_handling_overly_broad_throws.html http://www.owasp.org/index.php/Top_10_2007-A6	Código del componente		R	BAJO	1							
		SE02C-05.007			Incorrecto control de errores (excepción NullPointerException): No en todos los casos es una buena practica la captura de esta excepción. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/poor_error_handling_program_catches_nullpointerexception.html http://www.owasp.org/index.php/Top_10_2007-A6	Código del componente		R	BAJO	1							
		SE02C-05.008			Incorrecto control de errores (Return en bloques Finally): La sentencia "return" en un bloque "finally" puede provocar que no se lancen algunas excepciones. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/poor_error_handling_return_inside_finally.html http://www.owasp.org/index.php/Top_10_2007-A6	Código del componente		R	BAJO	1							
		SE02C-05.009			Incorrecto control de errores (Excepción SSL no controlada): El tratamiento incorrecto de las excepciones SSL puede causar errores inesperado en la aplicación. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/poor_error_handling_unhandled_ssl_exception.html http://www.owasp.org/index.php/Top_10_2007-A6	Código del componente		R	BAJO	1							
		SE02C-05.010	QAC-0963	Evitar aritmética de punteros en Cobol	La mayoría de los dialectos de Cobol no proporcionan soporte directo para la aritmética de punteros, pero puede realizarse mediante alias de puntero (una variable numérica asociada a un puntero mediante REDEFINES), sobre el que se realiza una operación aritmética. Cuando el puntero se dereferencia, se accede a una zona de memoria cuya dirección se ha obtenido mediante una expresión aritmética, que puede dar lugar a la terminación no esperada (""abend"") del programa o dar lugar a otro tipo de condiciones, en especial si los operandos de la operación aritmética pueden ser controlados desde entradas externas. La aritmética de punteros expone detalles de plataforma (como el tamaño en bytes del puntero o de zonas de memoria) que pueden hacer que el programa no pueda portarse a otra plataforma. La regla registra violación en cualquier operación aritmetica sobre un alias de puntero, si posteriormente el puntero se dereferencia (mediante sentencia SET).	Código del componente		R	ALTO	1							
		SE02C-05.011	QAC-0970	Evitar el uso de ALTER	La utilización de la instrucción ALTER no está permitida. La sentencia ALTER fomenta el uso de prácticas de programación no estructuradas; la sentencia EVALUATE ofrece la misma funcionalidad que la sentencia ALTER pero ayuda a asegurar que un programa esté bien estructurado.	Código del componente		R	ALTO	1							

CODIGO CONTROL /VERIFICACIÓN		Código OP-023	Código OP-023 Checking	PUNTO DE CONTROL VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Bloq./Exig./ Rec. B/E/R	Nivel de Riesgo	Automático 1=sí, 0=no	AFECTA A:						
											COSTE	FIABILIDAD	Tiempo de respuesta	Disponibilidad	Explotabilidad	Seguridad	Portabilidad
		SE02C-05.012		No proporcionar información confidencial como parte de los mensajes de error.	No está permitido el proporcionar información sensible o confidencial como parte de los mensajes que una aplicación devuelve en la descripción de un error para facilitar el seguimiento y /o resolución del mismo. Por ejemplo, el consultar los campos (COD_PAN, NUM_REF_PAN, NUM_CRD) y mostrar su contenido en campos de manejo de Errores referidos como CAA-ERR-VARI.	Código del componente		R	BAJO								1	
	SE02C-06	**Registro y auditoría**															
		SE02C-06.001			Log Forging: Se debe validar el acceso de los usuarios antes de resgistrarlo en el log, en caso contrario se podría facilitar que se falsifiquen datos o inyectar contenido malicioso. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/log_forging.html http://www.owasp.org/index.php/Top_10_2007-Injection_Flaws	Código del componente		R	BAJO	1								
		SE02C-06.002			Incorrecta gestión de logs (uso de System Output Stream): No se debe hacer uso de la salida estandar del sistema para el registro de logs. Referencias: http://www.fortify.com/vulncat/en/vulncat/dotnet/poor_logging_practice_use_of_a_system_output_stream.html http://www.owasp.org/index.php/Top_10_2007-Information_Leakage_and_Improper_Error_Handling	Código del componente		R	BAJO	1								
		SE02C-06.003			Incorrecta gestión de logs (Logger no declarado como Static Final): Es recomendable declarar la clase encargada de la gestión de logs como statica y final para que se comparta un único objeto entre todas las instancias. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/poor_logging_practice_logger_is_not_declared_static_final.html	Código del componente		R	BAJO	1								
		SE02C-06.004			Incorrecta gestión de logs (Multiples Loggers): Es una buena práctica utilizar diferentes niveles de logging en lugar de utilizar multiples clases de loggers. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/poor_logging_practice_multiple_loggers.html	Código del componente		R	BAJO	1								
	SE02C-07	**Entorno de la Aplicación**															
		SE02C-07.001			Manipulación de la configuración: Permitir el acceso a la configuración desde el exterior puede facilitar que esta sea modificada. Referencias: http://www.fortify.com/vulncat/en/vulncat/cpp/setting_manipulation.html	Código del componente		E	BAJO	1								
		SE02C-07.002			Manipulación de los permisos de ficheros: Permitir que la entrada del usuario modifique directamente los permisos de los ficheros puede posibilitar que el usuario obtenenga permisos no deseados. Referencias: http://www.fortify.com/vulncat/en/vulncat/php/file_permission_manipulation.html	Código del componente		E	BAJO	1								
		SE02C-07.003			Acceso a la aplicación mediante HTTP y HTTPS (Acegi Misconfiguration): Permitir que los usuarios cambien entre los protocolos HTTP y HTTPS facilita que la aplicación sea vulnerable a la manipulación de sesiones. Referencias: http://www.fortify.com/vulncat/en/vulncat/java/acegi_misconfiguration_insecure_channel_mixing.html http://www.owasp.org/index.php/Top_10_2007-Broken_Authentication_and_Session_Management	Código del componente		E	BAJO	1								
		SE02C-07.004			Uso inseguro de archivos temporales: No es una buena práctica el uso de ficheros temporales. Referencia: http://www.fortify.com/vulncat/en/vulncat/cpp/insecure_temporary_file.html	Código del componente		R	BAJO	1								
		SE02C-07.005			Proporcionar información del sistema: Es una buena práctica no porporcionar información del sistema ni de la arquitectura propia de la aplicación. Referencias: http://www.fortify.com/vulncat/en/vulncat/cfml/system_infor	Código del componente		E	BAJO	1								

CODIGO CONTROL / VERIFICACIÓN			PUNTO DE CONTROL/VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Exig./	Nivel de Riesgo	Automático 1=sí,
QA11P	**Verificación Ejecución de Pruebas**								
	QA11P-01	**Existencia de documentos asociados:**							
		QA11P-01.001	Casos de Prueba	Verificar la existencia del documento en la carpeta de red correspondiente. Verificar que el contenido hace referencia al desarrollo en curso	Casos de prueba		E	MODERADO	0
		QA11P-01.002	Relación de Incidencias	Verificar la existencia del documento en la carpeta de red correspondiente. Verificar que el contenido hace referencia al desarrollo en curso	Casos de prueba		E	MODERADO	0
	QA11P-02	**Ejecución de Pruebas, resultados esperados y obtenidos:**							
		QA11P-02.001	Ejecución de pruebas técnicas (concurrencia, rendimiento y gestión de errores)	Se revisará el documento de Casos de pruebas, verificando que se han ejecuctado los casos de prueba relativos a las pruebas técnicas: - Concurrencia - Rendimiento - Gestión de errores Se comprobará que los resultado esperados se corresponden con los resultados obtenidos, y en caso contrario que existe la correspondiente incidencia, con sus acciones a ejecutar	Casos de prueba		E	MODERADO	0
		QA11P-02.002	Ejecución de pruebas de volumen	Se revisará el documento de Casos de pruebas, verificando que se han ejecuctado los casos de prueba relativos a las pruebas de volumen. Se comprobará que los resultado esperados se corresponden con los resultados obtenidos, y en caso contrario que existe la correspondiente incidencia, con sus acciones a ejecutar	Casos de prueba		E	BAJO	0
		QA11P-02.003	Ejecución de pruebas de estrés	Se revisará el documento de Casos de pruebas, verificando que se han ejecuctado los casos de prueba relativos a las pruebas de estrés. Se comprobará que los resultado esperados se corresponden con los resultados obtenidos, y en caso contrario que existe	Casos de prueba		E	MODERADO	0
	QA11P-03	**Todas las incidencias / defectos (sobre pruebas técnicas) deben ser registrados y controlados conforme al nivel de detalle marcado por la instalación:**							
		QA11P-03.001	Todo registro debe ofrecer trazabilidad respecto del caso de prueba afectado, garantizando la relación con los requisitos afectados y las repetibilidad de la prueba	Para aquellos casos de prueba (tecnicas, de volumen, o estrés) en los que no se han obtenido los resultados esperados se debe contar con su correspondiente registro de incidencia. Éste debe aportar toda la información necesaria para: - Realizar el seguimiento de su resolución - Identificar claramente la funcionalidad afectada - Definir si es necesario o no realizar nuevamente la ejecución del caso de prueba	Casos de prueba		E	MODERADO	0
		QA11P-03.002	Las incidencias / defectos deben detallar objetivamente el problema detectado, indicando en cada caso la transcripción literal de los mensajes de error producidos (pop-ups, logs, trazas, etc.), las especificaciones técnicas del problema (en base a conceptos técnicos como abends, excepciones, interbloqueos, uso indebido de recursos, etc.)	Revisar la documentación, verificando que para las indicencias / defectos detectados en las pruebas técnicas se identifica, con la suficiente claridad, el problema o error. Debe recogerse toda la información que pueda ayudar en la comprensión tanto del problema en sí, como de su origen. Si es necesario se adjuntarán copias de las pantallas de error, logs, etc	Casos de prueba		E	MODERADO	0
		QA11P-03.003	Todas aquellas incidencias / defectos resueltos deben contener información detallada sobre la solución propuesta y su aceptación explícita	Revisar la documentación, verificando que para las indicencias / defectos detectados en las pruebas técnicas se identifican las diferentes alternativas para resolver el defecto / incidencia, si existen diferentes niveles de resolución (p.e. "work arround", definitiva, etc.) deberá especificarse de que tipo es cada alternativa	Casos de prueba		E	MODERADO	0
		QA11P-03.004	Todas aquellas incidencias / defectos resueltos deben contener información detallada sobre la solución finalmente implementada	Como parte del detalle de cada una de las incidencias / defectos detectados en las pruebas técnicas, debe figurar la solución implantada, con el detalle suficiente	Casos de prueba		E	MODERADO	0
	QA11P-04	**Cumplimiento de los niveles de servicio (tiempo de respuesta medio de la aplicación *end-to-end*) establecidos y aprobados en la documentación de especificación de requisitos:**							
		QA11P-04.001	Niveles de servicio en situaciones de baja, media y alta carga de usuarios concurrentes	Se verificará que el nivel de servicio, tiempo de respuesta, que se ha definido como esperado en el Catálogo de Requisitos, se cumple(*) bajo diferentes condiciones de carga: - Con un nivel bajo, número de usuarios inferior al estimado como normal (del 25 al 50% de usuarios concurrentes estimados). - Con el nivel medio de carga, número de usuarios estimado como normal (del 85 al 105% de usuarios concurrentes estimados). - Con un nivel de carga límite, número de usuarios por encima de lo estimado como normal (superior al 110% de usuarios concurrentes estimados). * Se considerará admisible una variación sobre el tiempo de respuesta esperado de: 5%, 10% y 15% respectivamente	Dosier del usuario Inventario de funcionalidades Sistemas		E	MODERADO	0
		QA11P-04.002	Umbrales de degradación (plazo de operación en situación de normalidad antes de presentarse caidas perceptibles - técnica y operativamente en el nivel de servicio) en situaciones de baja, media y alta carga de usuarios concurrentes	Se verificarán los umbrales(*) en los que el nivel de servicio se comienza a degradar, es decir se presentan problemas / empeoramiento en los tiempos de respuesta o en el servicio en general, bajo diferentes circunstancias de carga - Con un nivel bajo, número de usuarios inferior al estimado como normal (del 25 al 50% de usuarios concurrentes estimados). - Con el nivel medio de carga, número de usuarios estimado como normal (del 85 al 105% de usuarios concurrentes estimados). - Con un nivel de carga límite, número de usuarios por encima de lo estimado como normal (superior al 110% de usuarios concurrentes estimados). * Se considerará superado el umbral si se altera el tiempo de respuesta esperado en: - 5% independientemente del tiempo para carga baja - 15% tras 2 horas de servicio ininterrumpido con carga normal - 20% tras 1 hora de servicio ininterrumpido con carga límite	Dosier del usuario Inventario de funcionalidades Sistemas		E	MODERADO	0
	QA11P-05	**Infraestructura de servidor mainframe: cumplimiento de los requisitos técnicos de operación establecidos para la plataforma y tecnología correspondiente:**							
		QA11P-05.001	Consumo de CPU central / MIPS (servidor host) por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el consumo de CPU, en MIPS, de los procesos batch se ajusta a los estándares de la instalación. Deberán analizarse aquellos procesos que ya en esta fase de pruebas presenten desviaciones en estos consumos: - para una carga de hasta el 40% de los datos, el consumo debe ser inferior al 50% - para una carga superior al 75% de los datos (prueba de volumen), el consumo debe ser inferior al 100% del límite	Sistemas		E	MODERADO	0
		QA11P-05.002	Consumo de CPU central / MIPS (servidor host) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el consumo de CPU, en MIPS, de los procesos online se ajusta a los estándares de cada instalación. Para Transacciones IMS los valores son: - Hasta 100,000 ejec/mes 0,0050 MIPs - De 100,000 a 2,000,000 ejec/mes 0,0028 MIPS - Más de 2,000,000 ejec/mes 0,0014 MIPS Para Transacciones CICS los valores son: . 0,0098 MIPS Deberán analizarse aquellos procesos que ya en esta fase de pruebas presenten desviaciones en estos consumos	Sistemas		E	MODERADO	0
		QA11P-05.003	Consumo de memoria central (servidor host) por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará el consumo de memoria por los diferentes procesos batch. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora	Sistemas		E	MODERADO	0
		QA11P-05.004	Consumo de memoria central (servidor host) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará el consumo de memoria por las diferentes transacciones. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora	Sistemas		E	MODERADO	0
		QA11P-05.005	Consumo de disco central (servidor host) por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará la utilización de disco por los diferentes procesos batch. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora. Deberá prestarse atención al cumplimiento de las políticas de generación de datos temporales y su mantenimiento en disco tras la finalización correcta del proceso Cuando hablamos de consumos de disco, nos referimos fundamentalmente a dos cosas: volumetría de consumo en espacio (derivada por ejemplo de una generación anormal de logs, trazas, etc.) que pueden comprometer la dispobilidad de un filesystem, o latencias de I/O, como comentáis (asociadas a volumetría en accesos derivados de un hit-ratio en caché de disco muy pobre u otros factores - aunque esto último puede ser complicado de determinar dada la influencia de variables como la latencia de red, o el cambio en los patrones de petición desde un cliente).	Sistemas		E	MODERADO	0
		QA11P-05.006	Consumo de disco central (servidor host) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará la utilización de disco por los diferentes procesos online. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora	Sistemas		E	MODERADO	0

CODIGO CONTROL / VERIFICACIÓN			PUNTO DE CONTROL/VERIFICACIÓN	DESCRIPCIÓN DE LA VERIFICACIÓN	Producto	EXCEPCIÓN	Tipo de Verif. Exig./	Nivel de Riesgo	Automático 1=sí,
		QA11P-05.007	Tiempo de respuesta (tiempo de process) (servidor host) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el tiempo de respuesta (tiempo de process), en segundos, de los procesos online se ajusta a los estándares de cada instalación. Para Transacciones IMS el valor es que el tiempo de respuesta es mayor de 0,250 seg (independientemente del número de ejecuciones). Para Transacciones CICS los valores son: - 0,10 segundos. Deberán analizarse aquellos procesos que ya en esta fase de pruebas presenten desviaciones en estos consumos	Sistemas		E	MODERADO	0
	QA11P-06	**Infraestructura de servidor distribuido: cumplimiento de los requisitos técnicos de operación establecidos para la plataforma y tecnología correspondiente:**							
		QA11P-06.001	Consumo de CPU central / (servidor aplicaciones) por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el consumo de CPU de los procesos batch se ajusta a los estándares de la instalación. Se efecturan analisis de aquellos métodos que superen un 15%-20% de consumo. También se debe de tener en cuenta el % de consumo de un método en proporción con el resto (que se están evaluando) y número de llamadas que realiza en el programa. Para los métodos de arquitectura se considera elevado, debiendo ser analizado, si el método supera el 35% de consumo de CPU	Sistemas		E	MODERADO	0
		QA11P-06.002	Consumo de CPU central / (servidor aplicaciones) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el consumo de CPU de los procesos online se ajusta a los estándares de la instalación. En el analisis se realizara por métodos, paquetes o clases quedanto todo englobado en métodos. Se efecturan analisis de aquellos métodos que superen un 15%-20% de consumo. También se debe de tener en cuenta el % de consumo de un método en proporción con el resto (que se están evaluando) y número de llamadas que realiza en el programa. Para los métodos de arquitectura se considera elevado, debiendo ser analizado, si supera el 35% de consumo de CPU.	Sistemas		E	MODERADO	0
		QA11P-06.003	Consumo de memoria central (servidor aplicaciones) por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el consumo de memoria de los procesos online se ajusta a los estándares de la instalación. Deberán analizarse aquellos procesos que superen un 20% de consumo de memoria del servidor. El consumo de memoria debe de ser lineal, analizando todos aquellos en los que los picos de consumo, aún siendo por debajo del porcentaje indicado, no sean constantes . Ningún proceso debe superar el 50% de consumo de memoría (de forma líneal)	Sistemas		E	MODERADO	0
		QA11P-06.004	Consumo de memoria central (servidor aplicaciones) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se debe verificar que el consumo de memoria de los procesos online se ajusta a los estándares de la instalación. Deberán analizarse aquellos procesos que superen un 35% de consumo de memoria del servidor. El consumo de memoria debe de ser lineal, analizando todos aquellos en los que los picos de consumo, aún siendo por debajo del porcentaje indicado, no sean constantes . Ningún proceso debe superar el 50% de consumo de memoría (de forma líneal)	Sistemas		E	MODERADO	0
		QA11P-06.005	Consumo de disco central (servidor aplicaciones) por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará la utilización de disco por los diferentes procesos batch. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora. Deberá prestarse atención al cumplimiento de las políticas de generación de datos temporales y su mantenimiento en disco tras la finalización correcta del proceso.	Sistemas		E	MODERADO	0
		QA11P-06.006	Consumo de disco central (servidor aplicaciones) por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará la utilización de disco por los diferentes procesos online. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora	Sistemas		E	MODERADO	0
	QA11P-07	**Infraestructura de comunicaciones: cumplimiento de los requisitos técnicos de operación establecidos:**							
		QA11P-07.001	Consumo de I/O - recursos de comunicaciones por procesos *batch* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará el consumo y utilización de los recursos de I/O por los procesos batch. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y las posibilidades de mejora.	Sistemas		E	MODERADO	0
		QA11P-07.002	Consumo de I/O - recursos de comunicaciones por procesos *online* conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará el consumo y utilización de los recursos de I/O por los procesos online. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo	Sistemas		E	MODERADO	0
	QA11P-08	**Infraestructura de cliente: cumplimiento de los requisitos técnicos de operación establecidos para la plataforma y tecnología correspondiente:**							
		QA11P-08.001	Consumo de CPU cliente conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará el consumo de CPU de los procesos en la plataforma cliente. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y estudiar las acciones de mejora	Sistemas		E	MODERADO	0
		QA11P-08.002	Consumo de memoria cliente conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará la utilización de la memoria por los procesos en la plataforma cliente. Aquellos procesos que presenten consumos anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este consumo y estu	Sistemas		E	MODERADO	0
		QA11P-08.003	Consumo de disco en plataforma cliente conforme al estándar de la instalación para la plataforma y tecnología base del aplicativo	Se verificará el uso y las necesidades de disco por los procesos en la plataforma cliente. Aquellos procesos que presenten consumos o necesidades anormalmente elevados, por encima de los estándares, deberán analizarse en mayor profundidad para estudiar las causas de este co	Sistemas		E	MODERADO	0

Consideraciones para determinar la ponderación del *RIESGO*

Magnitud del Impacto	Ponderación de Riesgo
Deben considerarse las dimensiones que podrían comprender la materialización del riesgo identificado	
Aceptado: La consecuencia del riesgo representaría una situación de magnitudes imperceptibles o pequeñas.	**1**
Aceptado con Riesgo: La consecuencia del riesgo representaría una situación que se puede mantener bajo cierto control y que puede llegar a ser severa en CPD-CCR.	**2**
Rechazado: La consecuencia del riesgo representaría unas situación de extrema importancia para la Producción (CCR - CPD).	**3**

Comandos Restringidos
CONNECT PROCESS
CONVERSE
EXTRACT ATTRIBUTES
EXTRACT PROCESS
FREE
HANDLE AID
PURGE MESSAGE
RECEIVE
ROUTE
SIGNOFF
SIGNON
WAIT TERMINAL

Comandos con opciones restringidas

Comando	Opción
ISSUE	ABEND
	CONFIRMATION
	ERROR
	PREPARE
	SIGNAL
	PRINT
	ABORT
	ADD
	END
	ERASE
	NOTE
	QUERY
	RECEIVE
	REPLACE
	SEND
	WAIT

Comando	Opción
SEND	CONTROL
	MAP
	PARTNSET
	TEXT
	TEXT(MAPPED)
	TEXT(NOEDIT)
	PAGE

Comando	Opción
ASSIGN	ALTSCRNHT
	ALTSCRNWD
	APLKYBD
	APLTEXT
	BTRANS
	COLOR
	DEFSCRNHT
	DEFSCRNWD
	DELIMITER
	DESTCOUNT
	DESTID
	DESTIDLENG
	DS3270
	DSSCS
	EWASUPP
	EXTDS
	FACILITY
	FCI
	GCHARS
	GCODES

Comando	Opción
XCTL	INPUTMSG
	INPUTMSGLEN

Opción	Opción
GMMI	PARTNS
HILIGHT	PARTNSET
INPARTN	PS
KATAKANA	QNAME
LDCMNEM	SCRNHT
LDCNUM	SCRNWD
MAPCOLUMN	SIGDATA
MAPHEIGHT	SOSI
MAPLINE	STATIONID
MAPWIDTH	TCTUALENG
MSRCONTROL	TELLERID
NATLANGINUS	TERMCODE
NEXTTRANSID	TERMPRIORITY
NUMTAB	TEXTKYBD
OPCLASS	TEXTPRINT
OPSECURITY	UNATTEND
OUTLINE	USERNAME
PAGENUM	USERPRIORITY
PARTNPAGE	VALIDATION

CERTIFICACIÓN DE APLICACIONES
Integridad Referencial

DEFINICIÓN

La integridad referencial es un sistema de reglas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental producien

Tipos de Relaciones de Integridad

Relación Uno a Uno: Cuando un registro de una tabla sólo puede estar relacionado con un único registro de la otra tabla y viceversa.
Relación Uno a Varios: Cuando un registro de una tabla (tabla secundaria) sólo puede estar relacionado con un único registro de la otra tabla (tabla principal) y un registro de la tabla principal puede tener más de un registro relacionado en la tabla secundaria, en este caso se suele hacer referencia a la tabla principal como tabla 'padre' y a la tabla secundaria como tabla 'hijo', entonces la regla se convierte en 'un padre puede tener varios hijos pero un hijo solo tiene un padre. Esta relación se representa incluyendo en la tabla 'hijo' una columna que se corresponde con la clave principal de la tabla 'padre', esta columna es lo que se denomina clave foránea (o clave ajena o clave externa).
Una clave foránea es pues un campo de una tabla que contiene una referencia a un registro de otra tabla.
Relación Varios a Varios: Cuando un registro de una tabla puede estar relacionado con más de un registro de la otra tabla y viceversa. En este caso las dos tablas no pueden estar relacionadas directamente, se tiene que añadir una tabla entre las dos que incluya los pares de valores relacionados entre sí.

NOTA: No se hace referencia a TERADATA pues está prohibido el uso de I:R. por gestor en un modelo Informacional puro.

D= DB2 HOST; **U=** DB2-UDB; **O=** ORACLE; **S=**SQLSERVER

D	O	U	S	DESCRIPCIÓN	DESCRIPCIÓN	EFECTO - IMPACTO POTENCIAL
X	X	X	X	Codificación de programas	El gestor realiza las validaciones, que le han sido indicadas en la definición de las reglas de I.R., al crear o modificar una tabla evitando tener que hacerlo por programa. El programa debe contemplar los códigos de retorno en el acceso a BB.DD. que informen del estado de la actualización. Un cambio en la regla no obliga a un cambio en programación salvo que existan validaciones de campos previas a la sentencia de actualización.	Reducción tiempo de desarrollo.
X	X	X	X	Limitaciones en I.R.	Las únicas reglas de integridad referencial que se permiten implementar para ser validadas por el gestor son las que hacen referencia a Clave Primaria- Clave ajena. Es decir, está prohibido el uso de `CONSTRAINTS` salvo en los casos: - `NOT NULL` - `NOT NULL WITH DEFAULT` - `UNIQUE INDEX` Está prohibido el uso de `TRIGGERS`.	Cualquier validación de integridad de la información que no sea la verificación de clave primaria-clave ajena debe ser codificada en los programas.
X	X	X	X	Independencia del método de acceso.	Las reglas de I.R. definidas al gestor son a nivel de tabla por lo que se garantiza su cumplimiento independientemente del tipo de acceso a la tabla: programa, herramienta, etc.	Evita errores de consistencia de datos en cuanto a la relación clave primaria-clave ajena. Puede prevenir errores en cambios "a mano" de los datos en producción.
X	X	X	X	**Utilidad batch LOAD**. Tratamiento de registros rechazados por violación de I.R.	Cuando se utiliza I.R. por gestor y se realizan procesos de carga, si se encuentran registros que no cumplen con las reglas definidas, el proceso se para y se debe intervenir antes de rearrancar el proceso. En el gestor Oracle estará prohibida la opción de `DIRECT = Y` en la utilización del programa `SQLLOADER` ya que esta opción carga por bloques la información y no comprueba la IR que este implementada.	Probable aumento de indisponibilidad ante errores. Requiere intervención manual antes de rearranque.
X	X	X	X	**Procesos de recuperación**: disponibilidad de las tablas.	Cuando se aplica una recuperación a una tabla con I.R. a un momento anterior en el tiempos se produce una indisponibilidad sobre todo el árbol de dependencias hacia abajo definidas al gestor pudiendo llegar a afectar a la base de datos completa. Debido al impacto en la disponibilidad de las aplicaciones que conlleva el hecho de incorporar la Integridad por gestor en los procesos de recuperación a un momento anterior en el tiempo, no se va a permitir la recuperación sobre la propia tabla. .- Para `DB2 HOST` la alternativa propuesta será recuperar la tabla a un momento anterior en el tiempo sobre un fichero secuencial para que sea la aplicación la que decida qué hacer dependiendo de los datos del fichero. .- Para el resto de gestores no se puede recuperar sobre fichero secuencial, por lo que se estudiará en cada caso.	Aumento del ámbito y duración de la indisponibilidad ante una recuperación.
X	X	X	X	**Procesos Online:** rendimiento	Para los procesos online el uso de la IR puede hacer disminuir el consumo de la `CPU`, al realizar el gestor las validaciones en una sola llamada al mismo. Para que esto ocurra, los procesos on-line deben ser diseñados teniendo en cuenta que la IR la mantiene el Gestor, es decir, las aplicaciones no deben realizar las validaciones de integridad que hará automáticamente el gestor.	La eficiencia en el proceso de validación de datos por parte del gestor (para los casos previstos) es mayor o igual que si se hiciesen por programa, nunca menor.
X	X	X	X	**Actualización mediante herramientas de acceso directo:** SPUFI, etc.	Actualmente en los entornos productivos se realizan ejecuciones de sentencias `SQL` dinámico de actualización lanzadas de forma controlada utilizando lo que se denomina comúnmente "`SPUFI batch`" o "`Boletas`". En general la utilización de esta herramienta por parte de DYD es como consecuencia de correctivos a nivel de datos, es decir, problemas en los datos generados por los propios programas. La forma de resolver estos errores en los datos es ejecutar sentencias `SQL: INSERT, UPDATE` y `DELETE` vía "`SPUFI batch`" o "`Boleta`".	El definir I.R. por gestor podría reducir, en alguna medida, la necesidad de ejecutar este tipo de actualizaciones. Sólo afecta a los "arreglos" que hubiera que hacer por errores en las reglas que se pueden definir manejadas por el gestor. El resto no los
X	X	X	X	**Procesos batch:** El orden de los procesos sí importa.	Cuando se utiliza la facilidad de mantener la I.R. por gestor hay que tener en cuenta que el orden en el que se ejecutan los procesos batch sí que importa puesto que puede afectar al funcionamiento adecuado de la aplicación, principalmente en procesios masivos. Las tablas "padre" deben contener la infomración necesaria antes del proceso de carga (por programa o utilidad) de las tablas "hijas".	Simplemente es necesario conocer las dependencias definidas al gestor en el momento de la planificación de los procesos.
X	X	X	X	**Procesos Batch:** rendimiento.	Para el caso de los procesos batch, el impacto en el rendimiento es mayor, principalmente en actualizaciones masivas, dado el volumen de datos que hay que validar para mantener la integridad de los mismos, obligada por las relaciones entre las tablas. .- En procesos batch que se ejecutan sobre tablas sin IR, se tiene la libertad de realizar las validaciones, mediante descargas, sort, cruces de ficheros, etc, es decir, por fuera del gestor y con utilidades de tratamiento masivo de datos. .- En procesos batch que se ejecutan sobre tablas con IR, es el gestor el que realiza de forma obligada las validaciones y por tanto siempre hay que trabajar por "dentro" del mismo, aunque el volumen de datos a tratar pueda aconsejar, para mejorar el rendimiento, el uso de esas utilidades de tratamiento masivo de datos. Además hay que considerar que todas las validaciones de integridad referencial que actualmente las aplicaciones realizan a posteriori en procesos Batch para optimizar el tiempo de respuesta de las transacciones, ahora se van a realizar en tiempo Online. Esto puede suponer un incremento en el tiempo de `ELAPSED` y `CPU` de las transacciones dependiendo del nivel de implantación de la integridad referencial en los modelos	Aumento significativo en el tiempo de respuesta y consumo de cpu de los procesos batch.
X	X	X	X	**Procesos Batch:** Paralelización de procesos.	Es necesario tener en cuenta las relaciones establecidas entre tablas por I.R. mantenida por gestor y la distribución de paralelismo de procesos. No se podrían estar haciendo procesos de borrado masivo de registros, en paralelo, de tablas "padre" y tablas "hijas", primero habría que borrar en las hijas y luego en las padre. Sin I.R. por gestor no hay que tener en cuenta esta restricción. Esto afecta por igual a inserciones y actualizaciones masivas. Para el proceso de insert se debería ejecutar primero en las tablas padre y terminado este proceso se podría ejecutar en paralelo el insert en la tablas hijas con problemas de rendimiento tipo "cuello de botella" en el índice primario de las tablas de cuentas padre, al tener todos los procesos que verificar que existe el contrato correspondiente en dichas tablas.	El paralelismo de procesos puede quedar reducido derivado de las reglas de integridad referencial definidas al gestor. Un menor nivel de paralelismo puede conllevar mayor tiempo de respuesta del proceso.
	X	X	X	**Administración de BB.DD.**: Conocimientos	El gestor de base de datos requiere el acceso a información del modelo entidad-relación, como elemento de apoyo en su trabajo, sin que este conocimiento suponga responsabilidad alguna sobre el modelo lógico de datos, ni obligación alguna de comprobar que el modelo físico existente/creado se ajuste la modelo lógico definido.	Traspaso de mayor información entre DyD y TyE
X	X			**Administración de BB.DD.**: Disponibilidad	Incremento de el tiempo de mantenimiento y modificación de estructuras de tablas. Los procesos de alta y modificación de tablas afectan directamente en la disponibilidad de las aplicaciones puesto que afectan a la tabla a modificarse y sus relacionadas mediante I.R. por gestor. Para gestores distribuidos (`ORACLE, DB2-UDB`) también se ve afectada la disponibilidad ante utilidades de `REORG`.	Potencial aumento de indisponibilidad ante cambios en las estructuras de las tablas.