UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Factoria de Proves de WebServices sobre arquitectura SOAP

Treball Fi de Grau

Grau en Enginyeria Informàtica

Autor: Carrió Àlvarez, Jorge

Tutor: Busquets Mataix, José Vicente

Juliol 2014

Resum

Aquest treball és el resultat d'un estudi sobre una factoria de *software* i, concretament, dins de la factoria la part encarregada de les proves. En ell el que es pretén és dur a terme una millora de la qualitat del codi desenvolupat aplicant la tècnica del *testing*. S'analitzen i s'estudien les ferramentes de prova, la documentació i la gestió que es deu realitzar en una factoria de proves dins d'un context determinat. Així mateix, s'aborda d'una manera més específica el procés de proves d'un cicle de vida d'un *WebService*. D'aquesta manera s'analitzen els tipus de prova amb més detall i amb un exemple real. Per finalitzar, s'extreuen una sèrie de conclusions que pretenen generar una reflexió sobre la millora que suposa haver dut a terme proves en les fases de desenvolupament de codi, en concret amb el manteniment futur i amb la funcionalitat a l'hora d'integrar tot el conjunt de *WebServices*.

Paraules clau: factoria de software ,qualitat de codi, *testing*, cicle de vida, *WebService*.

Resumen

Este trabajo es el resultado de un estudio sobre una factoría de software y, concretamente, dentro de la factoría la parte encargada de las pruebas. En él se pretende llevar a cabo una mejora de la calidad del código desarrollado aplicando la técnica del testing. Se analizan y se estudian las herramientas de prueba, la documentación y la gestión que se debe realizar en una factoría de pruebas dentro de un contexto determinado. Asimismo, se aborda de una manera más específica el proceso de pruebas de un ciclo de vida aplicado a un WebService. De esta manera se analizan los tipos de prueba con más detalle y con un ejemplo real. Para finalizar, se extraen una serie de conclusiones que permiten generar una reflexión sobre la mejora que supone haber llevado a cabo pruebas en las fases de desarrollo de código, particularmente con el mantenimiento futuro y con la funcionalidad a la hora de integrar todo el conjunto de WebService.

Palabras clave: factoría de software, calidad de código, *testing*, ciclo de vida, *WebService*.

Abstract

This work is the result of a study about a software factory, specifically within the department responsible for factory testing. Its purpose is to conduct a quality improvement achieved by applying the code testing technique. We analyze and test tools, documentation and management processes to be performed in a test factory within a given context. It also addresses more specifically the test process of a WebService life cycle. Therefore we analyzed this in more detail and with a real example. Finally, we can summarize all of this with conclusions that show us the improvement achieved by having conducted tests at the stages of code development, in particular regarding maintenance and functionality when integrating all the WebServices.

Keywords: software factory, code quality, testing, life cycle, WebService.

“El testing puede probar la presencia de errores pero no la ausencia de ellos”, E. W. Dijkstra.

Índex

1. Introducció

1.1. Missió, objectius i abast

La missió de la factoria de proves és assegurar que les peticions entregades pel client, a nivell de component, estan dissenyades amb la màxima qualitat respecte a les característiques (ISO 9126) de funcionalitat, eficiència i manteniment, amb el propòsit de generar un alt nivell de confiança en els components proporcionats.

Els objectius principals que es deriven de la missió són:

La realització de proves exhaustives independents a nivell de component per aconseguir una qualitat de 10. Una qualitat 10 significa certificar que un *WebService* està lliure d'errors respecte a requisits i l'enfocament de les proves és acceptat. No s'ha de certificar la qualitat, sinó assegurar que es construeix amb qualitat, que s'apliquen les bones pràctiques de la indústria (disseny, construcció, cohesió funcional...) i marcar aquesta qualitat 10 en un nombre finit de proves executables.

Des de factoria es proposa establir bones pràctiques de disseny i construcció assegurant el procés per obtindre qualitat 10, proporcionar el *WebService* de qualitat requerida i taules d'estimació amb diferents graus en funció de l'abast. També es proporciona profunditat de les proves, és a dir, per a cada petició determinar la seua criticitat i establir la qualitat requerida en funció de la relació qualitat/preu.

Cal considerar que no es tindrà una visió de qualitat global a nivell de factoria de proves, sinó unitària i que la qualitat també va en funció del risc.

L'altre objectiu que es deriva de la missió és la detecció i eliminació de defectes de manera primerenca. Les proves estan plantejades per a que els defectes puguen trobar-se en les etapes més primerenques de les peticions. A banda de millorar la qualitat també s'assegura que el manteniment d'aquests *WebService* serà més fàcil de dur a terme.

L'abast de les proves funcionals, estàtiques i no funcionals es limita a la realització de proves a nivell de component per als elements de tipus *webservice*, comprovant la seua integració amb elements genèrics del sistema com són el *framework* i els magatzems de dades.

Queden expressament fora de l'abast les proves d'integració dels components amb altres elements interns o externs al sistema.

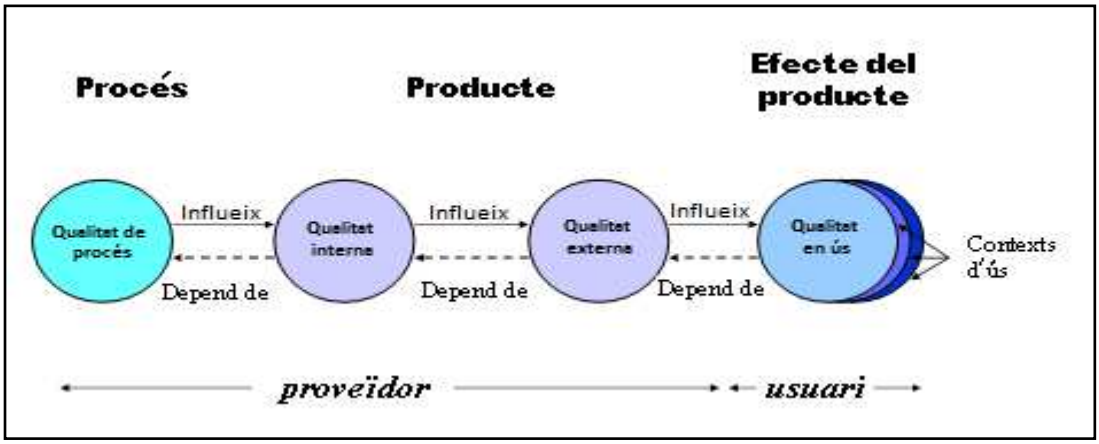
Dins de l'abast del projecte també està contemplada l'organització d'una factoria de proves fictícia i la interacció amb la factoria de desenvolupament i el client.

En aquest document explicarem la Factoria de Proves. El servei donat per la FdP és el manteniment i testeig de les aplicacions. A continuació es presenten els distints serveis i tipus de proves:

- Gestió del servei
  - Gestió de peticions
  - Generació i presentació d'informes
  - Gestió d'entorns
  - Millora contínua (avaluació periòdica del servei respecte als models de referència, amb l'objectiu d'identificar àrees de millora i implantació d'eixes millores)
- Proves estàtiques
  - Revisió de documentació
  - Revisió de l'entrega
  - Revisió del compliment d'estàndards i qualitat del codi
  - Verificació de proves de caixa blanca
- Proves funcionals
  - Funcionalitat
- Proves no funcionals
  - Rendiment (escalabilitat, càrrega, estrés i estabilitat)
  - Seguretat (estàtica i dinàmica)

2. Normativa de qualitat

L'objectiu no és necessariament arribar a una qualitat perfecta, sinó la suficient per a cada context d'ús. És necessari comprendre les necessitats reals dels usuaris (requeriments) per oferir la qualitat necessària.



Apareixen aquestes normes sobre la qualitat del producte *software* per a controlar la qualitat del producte, millorar les característiques, assegurar als clients un nivell mínim de qualitat, comparar productes amb la competència, posicionar el producte en el mercat, augmentar les ventes del producte i minimitzar errades en producció. També ajuda alhora d'adquirir *software* per conèixer la qualitat del producte que es compra, comprar entre diverses alternatives, reduir costos.

Una cosa molt important a tindre en compte en el següent conjunt de normes és que no estableixen els nivells de qualitat desitjable per a cada projecte. És a dir, no esmenen el llindar de qualitat que deu acomplir una determinada mètrica, sinó que ajuden en paràmetres per a determinar la qualitat del producte *software*, com per exemple el seu manteniment al llarg del temps.

Aquest fet és raonable, tenint en compte la multitud de situacions que es poden presentar, ja siguen software, empreses i demés. Seria irreal fixar un valor únic com a referència per a tota la indústria.

Així doncs aquestes normes recomanen que els requeriments de qualitat siguen proporcionals a les necessitats de les aplicacions i a la seua criticitat. Per tant, serà per a cada cas diferent i motiu d'estudi determinar el nivell de qualitat final que deu tindre el producte *software*.

2.1. ISO 25000

L'estàndard ISO 25000, coneguda també com SQuaRE (*Software Product Quality Requirements and Evaluation*), és una família de normes que tenen com a objectiu la creació d'un marc de treball comú per avaluar la qualitat del producte

software. Proporciona una guia per a l'ús dels nous estàndards internacionals, anomenats requisits i avaluació de qualitat de productes software. Està constituït per una sèrie de normes basades en la ISO 9126 i la ISO 14598 substituint aquestes dos. El seu objectiu principal és guiar el desenvolupament dels productes *software* amb l'especificació i avaluació dels requeriments de qualitat sent referent en l'avaluació de la qualitat del producte software. Estableix criteri per a l'especificació dels requeriments de qualitat de *software*, les seues mètriques i les seues avaluacions.

La norma es divideix en una sèrie de divisions o mòduls. Els més importants són:

ISO 2500n: mòdul corresponent a la gestió de la qualitat. Es defineixen tots els models comuns, termes i referències als que fan referència la resta de divisions.

- ISO 2501n: l'estàndard que conforma aquest mòdul presenta un model de qualitat al detall, incloent característiques per a la qualitat interna, externa i d'ús.
- ISO 2502n: aquesta divisió és l'encarregada d'estandarditzar les mètriques pertinents per a la qualitat del software. Presenta mètriques a nivell intern, extern i d'ús.
- ISO 2503n: aquesta divisió ajuda a l'especificació de requeriments de qualitat.
- ISO 2504n: aquesta divisió proporciona recomanacions i guies per a l'avaluació d'un producte *software*. Estan en l'àmbit tant de programadors, com compradors o avaluadors de software.

Aquesta norma recomana que els requeriments de qualitat deuen ser proporcionals a les necessitats de l'aplicació i al nivell de criticitat respecte al correcte funcionament del sistema desplegat.

2.1.1. ISO 9126

L'estàndard 9126, actualment inclosa en l'ISO 25000, internacional per a l'avaluació de la qualitat del *software*. Aquest estàndard està dividit en quatre parts: model de qualitat, mètriques externes, mètriques internes i mètriques de qualitat d'ús.

La primera part classifica la qualitat del *software* en un conjunt estructurat de característiques:

- **Funcionalitat**: conjunt d'atributs i propietats específiques de les funcions que es deuen acomplir. Adequació, exactitud, interoperabilitat i seguretat d'accés són alguns del atributs de la funcionalitat.
- **Fiabilitat**: conjunt d'atributs relacionats amb la capacitat del *software* en mantindre el seu nivell de prestacions en unes certes condicions i durant un període de temps establert.
- **Usabilitat**: capacitat per a ser entès, aprés i poder operar amb el *software* després d'haver-lo utilitzat.
- **Eficiència**: propietat relacionada amb el nivell d'acompliment sota unes condicions establertes.
- **Mantenibilitat**: propietat que representa la quantitat d'esforç requerida per conservar el funcionament normal o per restituir-lo després d'una fallada.
- **Portabilitat**: característica que posseeix un *software* per executar-se en diferents plataformes.

Hi ha diferents aspectes de qualitat segons quines siguen les mesures realitzades: interna, és aquella on es mesuren les característiques intrínseques, és a dir, el codi font; estàtiques, que no depenen de l'execució del programari i externa, són les aplicables al *software* en execució, com en una prova.

La part de les mètriques en qualitat d'ús està composta per quatre atributs principals:

- **Efectivitat**: aquest atribut medeix la capacitat del *software* en permetre als usuaris arribar als seus objectius en un context especificat.
- **Productivitat**: propietat que representa la capacitat del *software* per a permetre als usuaris utilitzar una quantitat de recursos en relació a l'efectivitat en un context específic.
- **Seguretat física**: es refereix la capacitat del producte *software* per arribar a nivells acceptables del risc de posar en perill al negoci, al *software*, a les persones o al medi ambient en un context d'ús específic.
- **Satisfacció**: capacitat del producte *software* per satisfer als usuaris en un context d'ús específic.

ISO 9126 distingeix entre fallada i no conformitat. Dins del marc de factoria fallada és quan es troba un codi mal programat, és a dir, que no acompleix els requisits prèviament definits en el DTAN. Mentre que la no conformitat és aquell cas en el qual si que s'acompleixen els requisits del DTAN però a criteri de FdP no és correcte.

2.1.2. ISO 14598

Aquest estàndard aplica al marc de treball per avaluar la qualitat de tot tipus de producte *software*. Proporciona unes indicacions per als requeriments per als mètodes i el procés d'avaluació. Les característiques d'aquest estàndard són la capacitat del programari en ser repetible, reproductible, imparcial i objectiu.

Aquesta norma es divideix en etapes:

- **Visió general**: estableix un resum de les següents parts de la norma. Explica la relació entre l'avaluació del producte *software* i el model de qualitat.
- **Planificació i control**: guies de suport per a la planificació i la gestió del producte *software*. Preparació de polítiques, definició d'objectius, identificació de la tecnologia i assignació de responsabilitats són algunes de les activitats que entren dins d'aquesta part.
- **Procés de desenvolupadors**: aquesta etapa és utilitzada majorment per les empreses que es dediquen a desenvolupar productes *software*. Dins de les activitats pròpies de les quals destaquen l'organització, les especificacions i el disseny.
- **Procés de comparadors**: utilitzat per les organitzacions que pretenen comparar productes *software* existents, ja siguen requeriments, especificacions o dissenys.
- **Procés d'avaluadors**: aquest procés s'encarrega d'avaluar, proveir de requeriments i guies per a l'avaluació del producte *software*.
- **Mòdul d'avaluació**: s'especifiquen les mesures que es van a prendre sobre les característiques de qualitat que s'han definit en l'etapa anterior.

Aquesta norma proporciona un marc de treball per avaluar la qualitat del *software*, indicant els requeriments a medir i analitzar-los durant el procés. Dins del marc de FdP les dos últimes etapes són les més enfocades al *testing* perquè estan més enfocats a l'avaluació de la qualitat pròpia del *software* i al posterior anàlisi del resultats. Les idees utilitzades per a desenvolupar en els punts posteriors d'aquest document provenen d'aquests apartats.

Aplicar els estàndards ajuda a millorar la qualitat del *software* i a mitigar els possibles errors que es poden presentar quan aquest està en execució.

2.2. IEEE 829

L'estàndard IEEE 829, també conegut com l'estàndard per a les proves i la documentació del *testing*, especifica el format del conjunt de documents del procés de

proves però no inclou cap criteri sobre el contingut adequat per a aquests documents. Aquests són una qüestió de judici fora de l'àmbit de la norma.

Els documents que defineix la norma són:

- **Pla mestre de proves**: el propòsit del qual és proporcionar un document global de planificació de les proves i gestió de proves per a diversos nivells.
- **Pla de proves de nivell**: per a cada nivell cal descriure l'abast, l'enfocament, recursos i calendari de les activitats específiques de la prova.
- **Disseny de proves de nivell**: detall de casos de prova i els resultats esperats, així com els criteris d'acceptació de les proves per a cada nivell.
- **Cas de prova**: especificació de les dades de prova per al seu ús en la gestió dels casos de prova assenyalats en el disseny de proves de nivell.
- **Procediment de prova de nivell**: detall de com executar cada prova, incloent condicions prèvies de configuració i els passos que s'han de seguir.
- **Informe de defecte**: per documentar qualsevol esdeveniment que es produeix durant el procés de proves que requereix investigació. Es pot anomenar incident de prova, defecte, anomalia, fallada, etc. L'informe consta de tots els detalls de l'incident a raó d'una discrepància entre els resultats esperats i reals. L'informe també inclourà a ser possible una avaluació de l'impacte de l'incident.
- **Informe mestre de prova**: aquest informe conté un resum del resultat de les proves. Es tracta d'un document de gestió per proporcionar informació i que inclou l'avaluació de la qualitat de les proves, l'entorn, el temps que es va prendre en realitzar la prova i el que es va realitzar a fi de millorar qualsevol planificació futura de proves.

La norma forma part del programa de formació dels certificats ISTQB ja que es prengué la IEEE 829 com a referència per a l'estàndard i la documentació del *testing*.

2.3. Normativa ISTQB

La norma internacional ISTQB (*International Software Testing Qualification Board*) és en l'actualitat la norma superior que certifica la qualitat dels professionals que treballen amb les proves de *software* que permeten verificar i avaluar la qualitat d'un producte *software*, el *testing*.

El comité internacional de qualificació de proves de *software*, ISTQB, és una organització creada l'any 2002 amb la finalitat de suportar i definir un esquema de certificació internacional. Aquest comité subministra el pla d'estudis on es defineixen els estàndards internacionals per nivell i estableix les guies d'acreditació i avaluació dels professionals del *testing*.

Les proves de *software* s'integren dins les diferents fases del cicle del *software* per determinar la qualitat del mateix. Es deuen efectuar diferents proves que permeten comprovar el grau de compliment respecte a les especificacions del sistema.

Les proves de *software*, *testing*, són un procés utilitzat per a identificar possibles errades d'implementació, de qualitat o usabilitat d'un programari.

Bàsicament és una fase del cicle del *software* que consisteix en provar les aplicacions construïdes.

Hi ha molts plantejaments alhora d'aplicar el procés de proves de *software*, però, per a verificar productes complexos de forma efectiva es requereix d'un procés d'investigació més que d'un procediment al peu de la lletra.

En general, els informàtics distingeixen entre error de programació (o *bugs*) i defectes de forma. En un defecte de forma, el programa no realitza allò que l'usuari espera. Mentre que un error de programació pot descriure's com una fallada en la semàntica del programa.

Una pràctica comú és que el procés de proves es realitze per un grup independent de "*testers*" encara que també es pot fer a nivell intern, però un grup extern sempre és més crític a l'hora de realitzar les proves i trobar defectes.

A l'hora de realitzar les proves de *software* també hi ha dos tendències: quan ja s'ha acabat de desenvolupar o en el mateix moment que està desenvolupant-se el codi. En aquest cas, es tracta de la segon tendència.

3. Procés

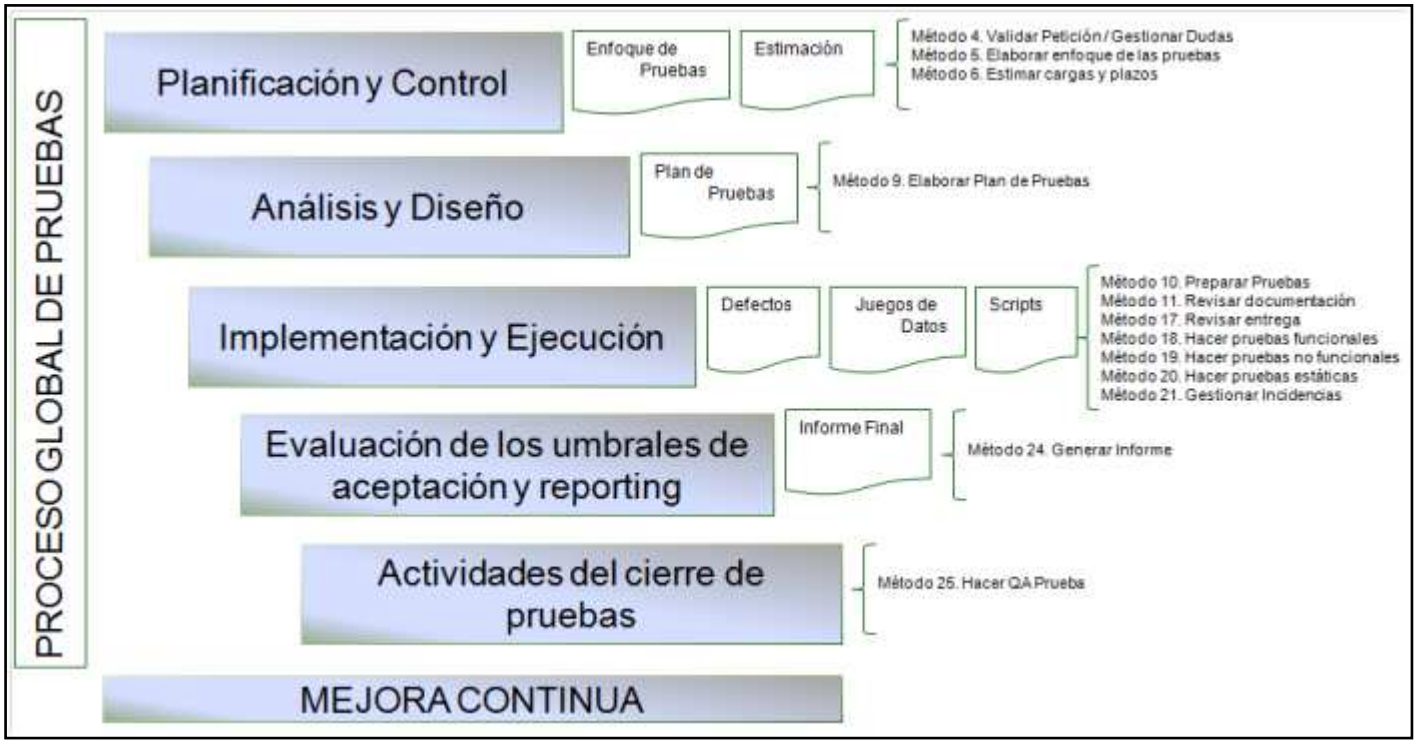
3.1. El procés de proves

3.1.1. Context

En l'àmbit del procés global d'Organitzar la Cadena de muntatge' del client, les activitats de factoria es troben englobades en el procés.

Hi ha tres tipus de peticions identificades: nou desenvolupament, evolutiu i correctiu.

Independentment del tipus de petició, l'execució de les activitats de la FdP es basa en el següent model on s'han reflectit, per cada subprocés, els entregables associats i els mètodes corresponents del procés global de factoria:



Encara que els processos són seqüencials, els subprocessos poden solapar-se o tindre lloc de manera concurrent o iterativa. Els objectius de cada subprocés poden resumir-se amb els mètodes del procés global de factoria:

- Planificació i control
  - Confirmar la missió i els objectius de les proves
  - Especificar en l'enfocament de proves les activitats per acomplir-les
  - Elaborar l'estimació de les activitats

    - Monitoritzar el procés durant tot el cicle de vida de la petició
    - Medir i analitzar els resultats
- Anàlisi i disseny
    - Identificar les condicions de proves i les dades en que es basen
    - Redactar el PLP
- Implementació i execució
    - Crear els jocs de dades de proves
    - Elaborar *scripts* de proves
    - Verificar que les ferramentes i l'entorn estan preparats
    - Executar el PLP d'acord a la seqüència planificada
    - Registrar les eixides i arxivar els productes de suport a les proves
    - Comparar els resultats obtinguts amb els esperats i reportar incidències en forma de defectes si hi ha
    - En cada cicle de proves, repetir les activitats d'execució que corresponguen
- Avaluació dels llindars d'acceptació i *reporting*
    - Acarar els registres de proves contra els llindars d'acceptació i els criteris de finalització establerts
    - Elaborar l'informe final de les proves
- Activitats de tancament de proves
    - Finalitzar i arxivar els productes de suport de proves, entorn i infraestructura
    - Entregar els productes de suport de proves
    - Analitzar les lliçons apresses, millorant la maduresa de les proves

3.1.2. Cicle de proves

Amb cada nova entrega realitzada per FdD es posen en funcionament les activitats de FdP. Aquest procés s'anomena cicle de proves.

A priori es planifiquen 3 cicles de prova sobre el *WebService*. L'objectiu consisteix a realitzar el màxim tipus de proves en cada un d'ells. Un cicle es considera finalitzat quan:

- Es troba un defecte bloquejant (aquells que no permeten progressar amb l'execució del Pla de Proves).
- S'executa el pla de proves en la seua totalitat.

En funció de la qualitat de l'entrega rebuda, pot haver variacions sobre el nombre de cicles executats.

En el primer cicle de proves, si no es troben defectes bloquejants, es realitzaran totes les proves. En cicles posteriors, es realitzaran en primer lloc les proves necessàries per confirmar la resolució de defectes reportats en el cicle anterior. A continuació, si es confirma la correcció de defectes, s'executaran la resta de proves. Si no s'han resolt correctament els defectes, es passa al següent cicle sense executar la resta de tipologia de proves.

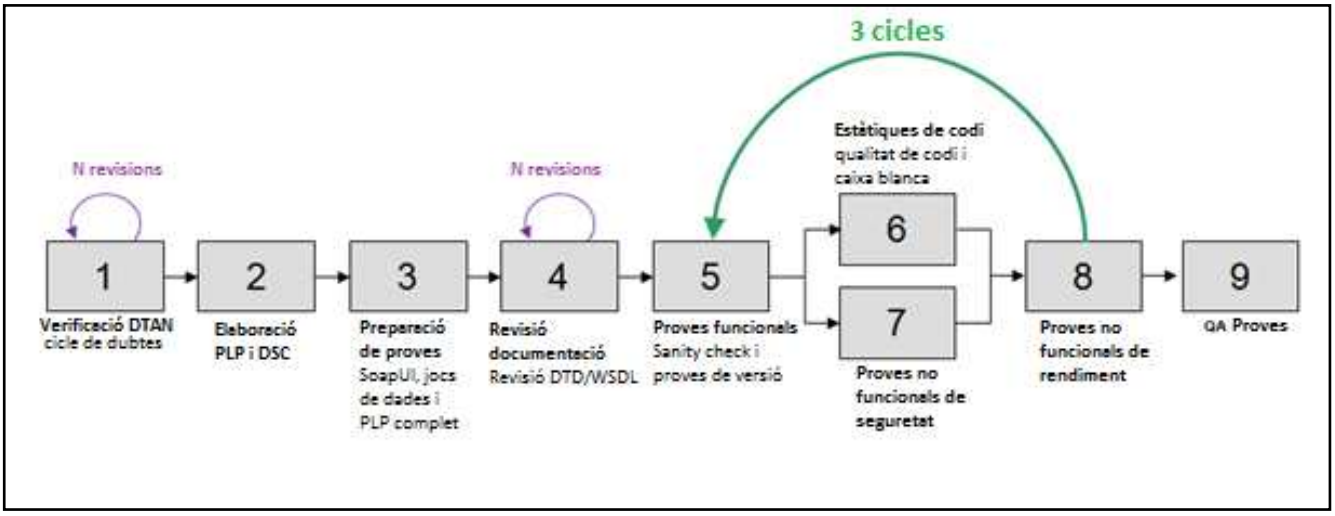
En funció de les característiques del *WebService* objecte de proves, és possible que un defecte bloquejant paralitze la resta de proves d'una determinada operació però no totes les del *WebService*. Aquestes situacions s'estudiaran i comunicaran en cada cas.

Amb la finalitat de gestionar la demanda d'un manera adequada s'han fixat quatre setmanes per dur-la a terme. En aquest període es realitzarà tota la fase de proves i resolució d'incidències. A continuació s'indiquen els temps detallats:

Fites petició (cicle normal)	S1							S2							S3							S4							S5						
	dl	dt	dc	dj	dv	ds	dg	dl	dt	dc	dj	dv	ds	dg	dl	dt	dc	dj	dv	ds	dg	dl	dt	dc	dj	dv	ds	dg	dl	dt	dc	dj	dv	ds	dg
Gestió de dubtes i validació de la petició																																			
Enviament d'estimació a FdD																																			
Enviament d'estimació a FdP																																			
Aceptació d'estimacions																																			
Enviament de DTD i WSDL																																			
Enviament del Plan de Pruebas Sanity Check																																			
Enviament del PLP genèric																																			
Entrega de codi a FdP																																			
Fi del cicle 1																																			
Resolució del cicle 1																																			
Fi del cicle 2																																			
Resolució del cicle 2																																			
Entrega Petició al client																																			
Tancament de la petició																																			

Cicle de gestió de dubtes	
Actor FdD	
Actor FdP	
Actor client	

La seqüència de les proves i les fites important i intermèdies del cicle de vida d'una petició estan resumits en la següent imatge:



3.1.3. Inventari de proves

El propòsit general que s'aplica a totes les proves és el de la màxima anticipació, és a dir, el propòsit és la detecció primerenca de defectes de cada tipologia de proves. Encara que amb aquest enfocament és possible que no es realitzen exclusivament les proves estrictament necessàries per a cada tipus de petició i que algunes proves s'hagen de repetir en un cicle posterior, el gran avantatge consisteix en la possibilitat de reportar i solucionar defectes el més prompte possible, amb la conseqüent reducció del nombre de cicles de prova.

En la següent taula s'especifiquen les activitats d'execució de proves que es van a realitzar en la FdP depenent de la fase i del cicle de proves.

WebService		**Proves estàtiques**				**Proves funcionals**			**Proves no funcionals**	
Tipologia	**Cicle**	Revisió documentació	Revisió entrega	Compliment estàndards i qualitat de codi	Caixa blanca	Sanity Check	Versió	Regresió	Seguretat	Rendiment
Nova petició	1	Si	Si, si no hi ha defectes bloquejants	Si, si no hi ha defectes bloquejants en funcionals	Si, si no hi ha defectes bloquejants en funcionals	Si, si no hi ha defectes bloquejants en revisió de documentació	Si, si no hi ha defectes bloquejants en Sanity Check	No aplica	Si, si no hi ha defectes bloquejants en les proves funcionals	Si, si no hi ha defectes bloquejants en las proves funcionals, estàtiques i seguretat. Estrés i estabilitat si no hi ha defectes bloquejants en las proves de càrrega.
	+1	Si, si s'entrega una nova versió del document	Si, si no hi ha defectes bloquejants	Si, si no hi ha defectes bloquejants en funcionals	Si, si no hi ha defectes bloquejants en funcionals	Si, si no hi ha defectes bloquejants en revisió de documentació	Si, si no hi ha defectes bloquejants en Sanity Check	Si, si no hi ha defectes bloquejants en Sanity Check. En funció de les modificacions poden ser parcials o totals	Si, si no hi ha defectes bloquejants en les proves funcionals	Si, si no hi ha defectes bloquejants en las proves funcionals, estàtiques i seguretat. Estrés i estabilitat si no hi ha defectes bloquejants en las proves de càrrega.

3.1.4. Entorn de prova

A l'inici del projecte les proves de FdP es realitzaven en els entorns ubicats en les instal·lacions del Client. Hi havia dos entorns principals per a provar: integració, que era un entorn amb menor volum de dades que el real, i preproducció, que és una còpia de l'entorn real del Client. Però degut a que aquests entorns eren compartits per moltes altres empreses dins de la Factoria i les dates d'entrega es podien veure retardades degut a les esperes per poder tindre accés als entorns del Client i realitzar les proves pertinents, FdP decideix emprar els seus propis entorns en les seues instal·lacions per tal d'agilitar les proves. Aquestos entorns propis de FdP són una imatge dels que s'utilitzaven abans però autogestionats i sense cues d'espera per poder accedir a ells degut a que altres factories estaven fent proves.

3.1.5. Ferramentes de prova

En la taula següent es mostren les ferramentes que seran utilitzades en cada tipus de prova:

		Factoria de proves								
		Proves estàtiques				Proves funcionals			Proves no funcionals	
Ferramentes		Revisió entrega	Revisió Documentació	Compliment estàndards i qualitat de codi	Caixa blanca	Sanity check	Versió	Regressió	Seguretat	Rendiment
FdP	Microsoft Office	☑	☑	☑	☑	☑	☑	☑	☑	☑
	Statiqueitor			☑	☑					
	ETFGen					☑	☑	☑		
	DARGenerator									☑
	MySQL			☑	☑	☑	☑	☑	☑	☑
	SonarQube			☑	☑					
	SQL Developer			☑	☑	☑	☑	☑		☑
	Filezilla					☑	☑	☑		☑
	SoapUI Pro					☑	☑	☑	☑	☑
	Yourkit java profiler									☑
	Subversion	Repositiori d'elements de factoria de proves								
	Jira	Gestió del flux de peticions i incidències.								
Client	IBM Clearcase	Repositiori d'elements de factoria								
	IBM ClearQuest	Gestió del flux de peticions i incidències.								

Estan explicades al detall en el punt 5 de la memòria.

3.1.6. Tipus de proves

En els apartats següents s'especifiquen les diferents tipologies de proves que s'inclouen en el catàleg del servei donat per la Factoria. Depenent del tipus de component objecte d'anàlisi i de les seues característiques es realitzaran unes proves o altres. Els tipus de prova de FdP són les detallades en els següents apartats.

3.1.6.1. Proves estàtiques

3.1.6.1.1. Revisió de documentació

Les proves o verificació de documentació tenen com a objectiu la detecció de defectes en el disseny, evitant arrastrar-los a la construcció del sistema. Aporta beneficis importants:

- Redueix el nombre d'errors originats per inexistència o interpretacions incorrectes de la documentació.
- Facilita el manteniment posterior de l'aplicació, a l'exigir una homogeneïtzació en la documentació durant tot el procés de desenvolupament.

Les proves de documentació consisteixen en la verificació de la documentació que es genera al llarg del cicle de vida. A l'inici del servei s'acordaran amb el client les especificacions funcionals i tècniques dels mòduls desenvolupats. Els documents objecte de verificació poden ser per exemple: disseny funcional, disseny tècnic, pla de proves, manuals...

Mitjançant l'ús de llistes de comprovacions (*checklist*), s'avalua la qualitat de la documentació. Aquestes llistes de comprovacions revisen aspectes relatius a l'existència, format i contingut de la documentació, adaptant-se a les polítiques, normatives, estàndards i processos existents en cada organització.

Els documents a revisar són el DTAN, el DTD i el WSDL.

Les mètriques utilitzades són:

- Nombre de revisions realitzades
- Nombre de *checkpoints* avaluats per a cada revisió realitzada
- Nombre de dubtes o defectes detectats per a cada revisió realitzada
- Nombre de dubtes o defectes pendents de resolució per a cada revisió realitzada

3.1.6.1.2. Revisió de l'entrega

L'objectiu d'aquestes proves és revisar que el paquet de l'entrega procedent de la FdD estiga complet i acomplisca els requeriments necessaris per a començar les proves per part de FdP.

Es comprova que tots els fitxers pertinents hagen segut entregats per. Els fitxers que deuen estar són: codi font, *JUnit* disponibles i l'índex d'entrega.

Les mètriques utilitzades són:

- Nombre de revisions realitzades
- Nombre de *checkpoints* avaluats per a cada revisió realitzada
- Nombre de dubtes o defectes detectats per a cada revisió realitzada
- Nombre de dubtes o defectes pendents de resolució per a cada revisió realitzada

3.1.6.1.3. Revisió del compliment d'estàndards i qualitat del codi

L'objectiu d'aquesta revisió és el compliment d'estàndards. Es tracta d'assegurar que els equips implicats en la construcció del codi estan aplicant les bones pràctiques de dissenys i programació vigents en el Client. L'abast d'aquestes proves inclouran el codi junt amb la documentació directament associada al mateix.

A través de l'execució de les activitats pròpies d'aquest procés es comprovarà que els valors llindars de les mètriques acordades amb el Client no excedeixen i que, a més, s'està dins dels nivells respecte als estàndards definits, tant els recomanats per a la indústria, com els que s'hagen definit a nivell intern.

En aquest tipus de prova es durà a terme un anàlisi del codi amb la ferramenta *Sonar*. Per a aquest anàlisi hi ha definides unes regles per la Factoria. Algunes mètriques de les utilitzades són el resultat d'aplicar les regles. Les mètriques són:

- Nombre de classes
- Nombre de línies de codi
- Nombre de violacions bloquejants
- Nombre de violacions crítiques
- Percentatge de regles acomplides, del total de regles de *Sonar*
- Percentatge de codi duplicat
- Nombre de mètodes amb complexitat ciclomàtica > 10, que proporciona una medició quantitativa de la complexitat lògica del programa
- Percentatge de codi mort

Es podran dur a terme auditories addicionals en aquells casos en els que es considere recomanable després d'analitzar els resultats d'altres tipus de proves, com són la qualitat intrínseca del codi.

3.1.6.1.4. Verificació de proves de caixa blanca

L'objectiu és verificar que totes les proves unitàries entregades per la FdD s'executen lliure d'errors i amb un grau de cobertura del codi suficient. Els criteris són aplicar les regles de *Sonar* per a caixa blanca definits. Com a resultat s'obté un informe de la qualitat del codi.

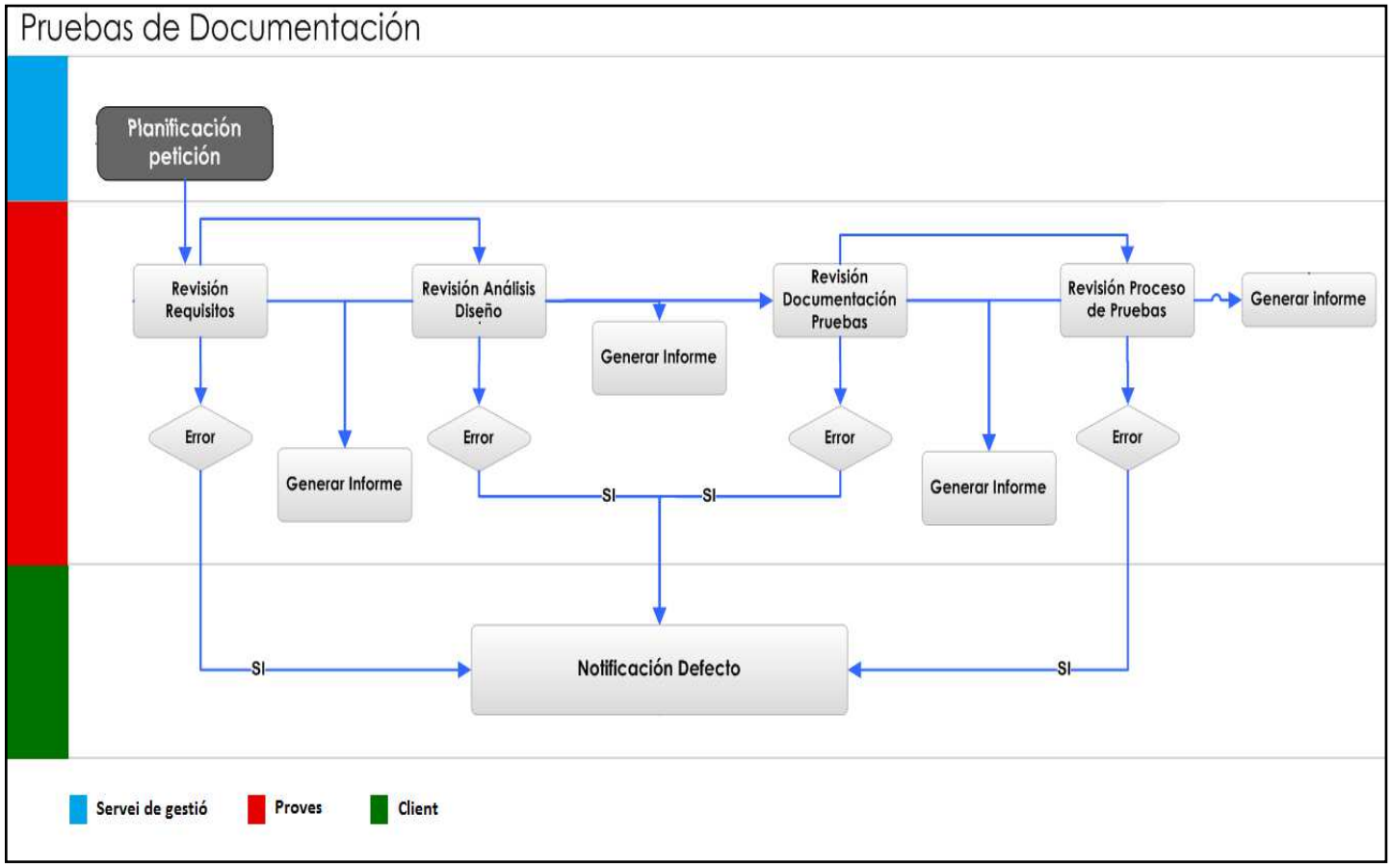
Les mètriques per determinar la qualitat del codi són:

- Percentatge de cobertura de proves unitàries, és a dir, percentatge de codi que es cobreix amb les proves unitàries *JUnit*.
- Percentatge d'èxit de les proves unitàries.

Els beneficis obtinguts com a resultat de les proves de qualitat intrínseca del codi es poden resumir en:

- Facilita el posterior manteniment del codi, reduint el temps utilitzat per al manteniment.
- Proporciona informació interna sobre la qualitat del codi que pot ser utilitzat per a millorar la qualitat del mateix.
- Ajuda a previndre defectes en projectes futurs mitjançant l'anàlisi de les lliçons apreses.
- Proporciona senyals primerenques d'avís mitjançant la detecció d'aspectes sospitosos del codi o el disseny, dependències...
- Al permetre detectar anticipadament els errors de qualitat es pot anticipar la resolució abans.

A continuació es presenta el flux de treball proposat:



3.1.6.2. Proves funcionals

Aquestes proves estan orientades a comprovar la funcionalitat del software abans que passen a producció, amb la finalitat de verificar que el sistema construït satisfà els requeriments funcionals i d'abast especificats. Les proves funcionals aporten importants beneficis:

- Confiança a l'usuari, ja que rep un software provat i robust.
- Increment de la productivitat de les proves, ja que els actius de proves que es generen, en especial els plans de prova, poden ser utilitzats en proves posteriors.
- Faciliten una posterior automatització de l'execució de les proves en futurs cicles de la mateixa petició.

El procés d'execució es cíclic fins que es rep un software que no genera fallades. En el primer cicle es duu a terme una execució completa del pla de proves, mentre que els següents es centren en les proves de les incidències solucionades pels equips de desenvolupament. Una vegada solucionades totes les incidències, s'executa un últim cicle per assegurar-se que la resolució d'un defecte no ha repercutit negativament en altres parts del software.

Dins de la Factoria de Proves aquestes es classifiquen en:

- ***Sanity Check***: consisteix en la realització de les proves necessàries per assegurar que es compleixen els requeriments corresponents a les funcionalitats principals descrites en el DTAN.
- **Versió**: consisteix a comprovar la correcta implementació de tots els requeriments de la petició descrits en el DTAN, per tant, s'executaran aquelles proves que els verifiquen.
- **Regressió**: consisteix en la realització de les proves necessàries per assegurar que després dels canvis realitzats, el *WebService* entregat satisfà tots els requeriments funcionals.

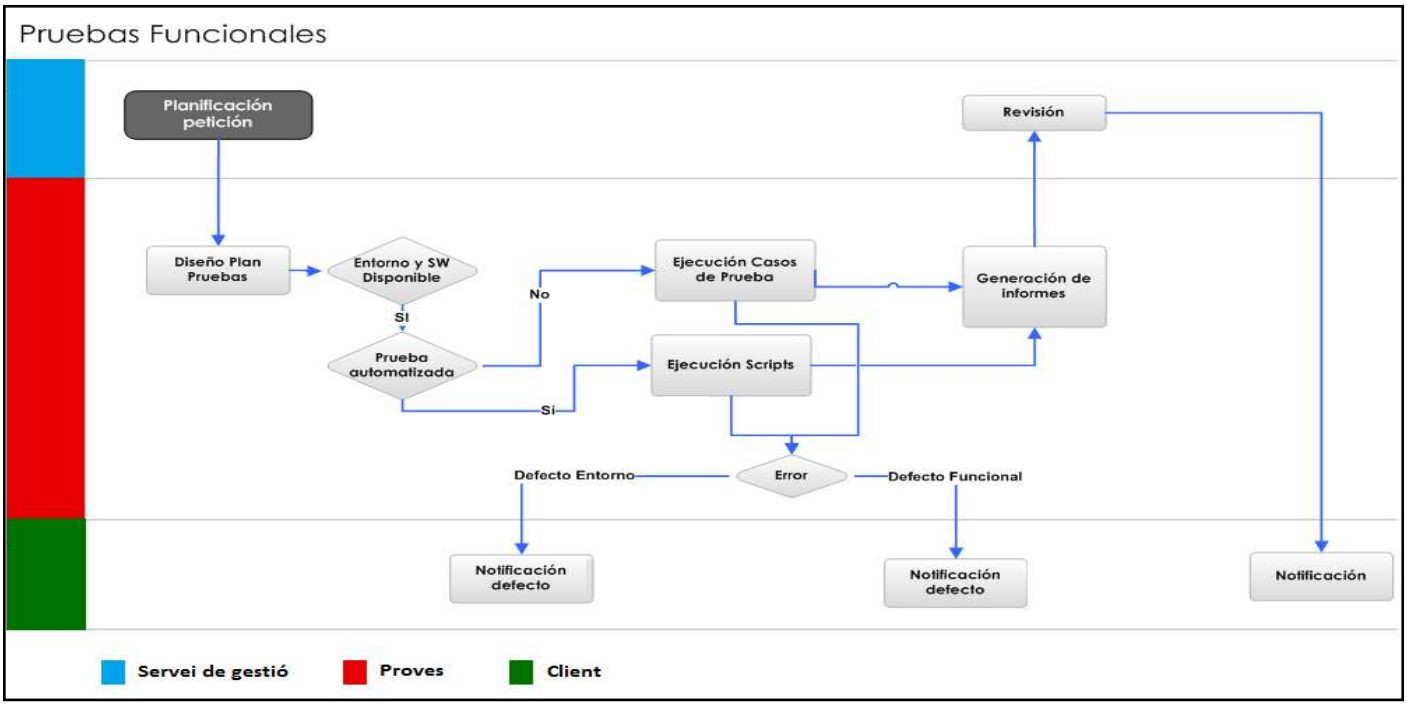
Per a dur a terme les proves funcionals és necessari el pla de proves funcionals, el joc de dades, els scripts pertinents, l'entorn i el *WebService* desplegat en l'entorn, tot preparat i disponible.

Una vegada finalitzades les proves es disposarà del PLP executat, si hi ha, els defectes reportats al client i l'informe de proves funcionals (és de caràcter intern).

Les mètriques utilitzades per determinar el resultat de les proves són:

- Nombre de cicles de proves realitzat.
- Nombre de requeriments funcionals.
- Total de casos de prova funcionals.
- Cobertura de proves:
  - Casos de prova de *Sanity Check vs* no executats per cicle.
  - Casos de prova de Versió *vs* no executats per cicle.
  - Casos de prova de Regressió *vs* no executats per cicle.
- Casos de prova amb resultat correcte/total de casos de prova.
- Defectes reportats/pendents en cada cicle de prova.

A continuació es presenta el flux de treball proposat:



3.1.6.3. Proves no funcionals

A continuació es detalla el procés de les proves no funcionals. Els diferents tipus de proves que es recullen en aquesta línia del servei es detallen en els apartats següents:

3.1.6.3.1. Proves de seguretat

L'objectiu de les proves de seguretat és verificar que les aplicacions que deuen passar a l'entorn de producció operen d'acord a les seues necessitats de seguretat i a les normes i polítiques establertes pel Client.

Aquestes proves avaluen la seguretat de les aplicacions des d'un punt de vista extern aplicant tècniques que impliquen l'execució de l'aplicació (proves dinàmiques) i des d'un punt intern mitjançant l'execució de proves estàtiques de seguretat de codi orientades a aconseguir que les aplicacions en Producció siguen segures front a potencials atacs de seguretat (verificació de que el codi font d'una aplicació siga robust i confiable).

- Proves dinàmiques

  Estan orientades a assegurar tres aspectes fonamentals:
  - **Confidencialitat**: l'accés a l'aplicació i a la informació està limitada mitjançant mecanismes eficients d'identificació, autentificació i control d'accés.
  - **Integritat**: la informació utilitzada per l'aplicació sols pot ser accedida per usuaris autoritzats.
  - **Disponibilitat**: l'accés a l'aplicació està garantida pels usuaris autoritzats.

- Proves estàtiques

  Les proves dinàmiques de seguretat es complementen amb les proves estàtiques de seguretat de codi, que anticipen possibles vulnerabilitats en el codi abans que aquest siga executat, proporcionant informació segons categoria i prioritat dels riscos, el que permet escometre la resolució d'una forma organitzada. Les activitats a realitzar en l'anàlisi estàtic de seguretat de codi es poden resumir en:
  - Configurar les regles per al projecte o petició.
  - Executar l'escaneig automatitzat del codi.
  - Analitzar els resultats obtinguts.
  - Proporcionar *feedback* dels resultats de l'anàlisi de codi en el cicle de desenvolupament per millorar la seguretat del software construït.

Els actius necessaris per a dur a terme aquestes proves són: el joc de dades, scripts de prova, l'entorn i el *WebService* desplegat en l'entorn. Tots ells disponibles i preparats.

Les mètriques utilitzades en aquesta tipologia de proves són:

- Nombre de cicles de prova realitzats.

- Nombre d'operacions del *WebService*.
- Nombre de no conformitats detectades en cada cicle de proves.
- Defectes reportats/pendents en cada cicle.

3.1.6.3.2. Proves de rendiment

Les proves de rendiment tenen com a objectiu assegurar que les aplicacions noves o modificades tindran un comportament en l'entorn de producció acord als paràmetres establerts en l'organització. Amb aquestes proves es determinen els llindars operatius i es simula l'ús real de l'aplicació, obtenint informació sobre temps de resposta, tasques de transferència, disponibilitat, utilització de recursos, capacitat del sistema...

Baix el concepte de proves de rendiment s'inclouen diferents tipologies de proves tècniques:

- **Proves d'escalabilitat**: s'utilitza per determinar si l'aplicació permet acomodar a un nombre creixent d'usuaris mitjançant l'addició de hardware, sense haver de modificar el software. Durant l'execució, no es busquen realment xifres de quants usuaris poden pujar-se a la plataforma, sinó establir un patró d'escalabilitat per al sistema. Aquesta informació permet determinar el nombre d'usuaris que poder ser suportats de forma acceptable per a un entorn definit.
- **Proves de càrrega**: s'utilitzen per a modelar una càrrega el més pareguda possible a la real i prendre mida anticipadament del rendiment que proporcionarà un sistema en l'entorn de producció. Permet identificar els temps de resposta que obtindrà un usuari típic front a condicions normals.
- **Proves d'estrès**: aquestes proves estan dissenyades per presentar el sistema de càrrega massiva, per verificar el seu funcionament en el llindar dels recursos. L'objectiu és establir els punts finals on el sistema comença a operar per baix de les seues especificacions. Aquest tipus de proves es realitzen per determinar la solidesa de l'aplicació en els moments de càrrega extrema i ajudar als administradors a determinar si l'aplicació rendirà prou en cas que la càrrega real supere a l'esperada.
- **Proves d'escalabilitat**: aquestes proes consisteixen en dissenyar una càrrega el més pareguda possible a la real i atacar amb ella l'aplicació durant llargs períodes de temps (típicament 24h) per buscar possibles deterioracions o degradacions en les prestacions de l'aplicació.

Els actius necessaris per a dur a terme aquestes proves són: el joc de dades preparat, scripts de prova preparats, l'entorn disponible i preparat i el *WebService* desplegat en l'entorn.

Les mètriques utilitzades en aquesta tipologia de proves són:

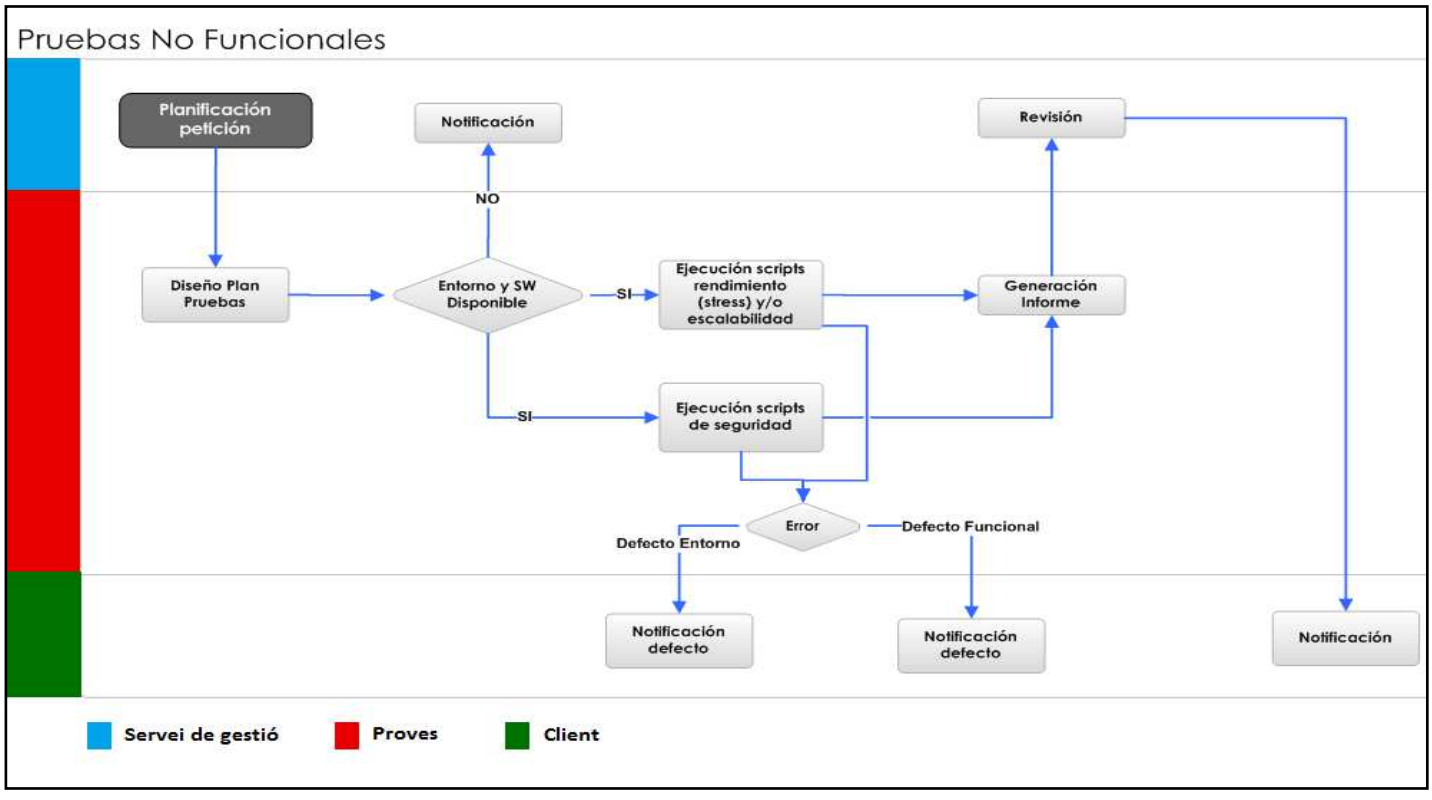
- Nombre de cicles de prova realitzats.
- Total de casos de prova de rendiment.
- Nombre d'operacions del *WebService*.
- Pool de connexions *JDBC Servidor Aplicacions*.

- Nombre de no conformitats detectades en cada cicle de proves.
- Defectes reportats/pendents en cada cicle.
- Temps mínim de resposta (mil·lisegons).
- Temps mitjà de resposta (mil·lisegons).
- Temps màxim de resposta (mil·lisegons).
- Nombre de connexions concurrents que duen a inestabilitat de la RAM.
- Nombre de connexions concurrents que duen a inestabilitat de la CPU.

La seqüència de les proves de rendiment per tal d'agilitzar al màxim la disponibilitat dels entorns és:

- 1er cicle:
  1. Proves de càrrega de tots els mètodes.
  2. Proves d'estrés de tots els mètodes que hagen superat les proves de càrrega.
  3. Si es disposa de temps, s'executaran les proves d'estabilitat d'aquells mètodes que complisquen els llindars de les proves de càrrega i estrés.
- 2on cicle:
  1. Proves de càrrega i estrés dels mètodes que han generat defecte en el 1er cicle.
  2. Proves d'estabilitat d'aquells mètodes que, per no disposar de marge de temps, no s'avaluaren en el cicle 1.
- 3er cicle:
  1. Proves d'estabilitat d'aquells mètodes que, per no disposar de marge de temps, no s'avaluaren en el cicle 1 i/o cicle 2.
  2. Si es disposa de temps, s'executaran les proves d'escalabilitat.

A continuació es presenta el flux de treball per a les proves no funcionals proposat:



3.2. Criteris d'acceptació

La classificació de defectes i de la seua severitat, així com els criteris d'acceptació del *WebService* varia depenent de la fase, de la tipologia de proves i dels llindars d'acceptació definits en cada cas.

En aquest document l'estratègia definida de proves es centra exclusivament en els defectes trobats en el desenvolupament del *WebService*, encara que dins de la Factoria les activitats de FdP inclouen les proves en tres fases diferents:

- Desenvolupament del *WebService*.

- Integració de *WebServices*.

- Manteniment del *WebService* en Producció.

En la fase de desenvolupament, s'estableix el criteri d'acceptació general per a les proves realitzades a zero defectes a resoldre. No obstant, pot adaptar-se a les necessitats i circumstàncies de les peticions, i, sempre de manera extraordinària i justificada, es pot acceptar un component amb determinats defectes pendents de resoldre.

Per a cada tipologia de proves s'han definit uns llindars d'acceptació que determinen els criteris de detecció de defectes i de gestió del mateix.

3.2.1. Proves estàtiques

3.2.1.1. Revisió de documentació

La revisió dels documents es basa sobre dos tipus de documents: el DTAN i el DTS/WSDL.

Els criteris d'acceptació de la revisió del DTAN estableixen que s'haja verificat el 100% dels requeriments del DTAN i que el 100% dels dubtes bloquejants i alts estiguen resolts.

Es reportarà un únic defecte representatiu de totes les no conformitats detectades. La severitat del defecte serà igual a la major severitat que es done en les no conformitats detectades en base als següents criteris:

Severitat No-Conformitat	Umbral de acceptació	Severitat Defecte a reportar
Bloquejant	0%	Bloquejant
Alta	0%	Alta
Baixa	15%	Alta

3.2.1.2. Revisió de l'entrega

En la revisió de l'Entrega es detecten no conformitats. Aquestes es classifiquen en base a les següents severitats:

- **Bloquejant**: no permet iniciar proves. Per exemple: error *Sanity Check*, entrega no completa.
- **Alta**: error relatiu a l'estat de la petició o a l'estructura formal de la mateixa. Obligat a arreglar abans del tancament de la petició.
- **Mitjana**: no aplica per a aquest tipus de proves.
- **Baixa**: no aplica per a aquest tipus de proves.

Es reportarà un únic defecte representatiu de totes les no conformitats detectades. La severitat del defecte serà igual a la major severitat que es done en les no conformitats detectades en base als següents criteris:

Severitat No-Conformitat	Umbral de acceptació	Severitat Defecte a reportar
Bloquejant	0%	Bloquejant
Alta	0%	Alta

3.2.1.3. Revisió del compliment d'estàndards i qualitat del codi

Les no conformitats detectades en la revisió qualitativa del codi es classifiquen en base a les regles del *Sonar* definides.

Es reportarà un únic defecte representatiu de totes les no conformitats detectades. La severitat del defecte serà igual a la major severitat de les no conformitats detectades.

La severitat de les no conformitats es defineix en funció del seu impacte en el manteniment futur del codi:

- **Bloquejant**: no aplica per a aquest tipus de proves.
- **Alta**: violacions que poden donar lloc a errors o comprometre de manera significativa el manteniment del codi.
- **Mitjana**: violacions que afecten en menor mesura al manteniment futur del codi.
- **Baixa**: actualment no aplica per a aquest tipus de proves.

Tipologia de No-conformitat	Umbral de acceptació	Severitat Defecte a reportar
Violacions Bloquejants	0	Alta
Violacions Critiques	0	Alta
% Regles Sonar incomplides	10%	Alta

Tipologia de No-conformitat	Umbral de acceptació	Severitat Defecte a reportar
% Mínim de Comentaris	30%	Media
% Codi Duplicat	5%	Media
Complexitat Ciclomàtica per mètode	10	Alta
% Codi Mort	0%	Media

3.2.1.4. Verificació de proves de caixa blanca

Les no conformitats detectades en la verificació de proves de caixa blanca es classifiquen en base als següents llindars d'acceptació associats a les regles corresponents definides en *Sonar*.

Es reportarà un únic defecte representatiu de totes les no conformitats detectades. En aquest cas es defineix exclusivament el concepte de severitat bloquejant, donat que la verificació de les proves de caixa blanca es condició indispensable per poder donar inici a les proves de caixa negra.

S'han definit els següents llindars d'acceptació:

Tipologia de No-conformitat	Umbral de acceptació	Severitat Defecte a reportar
% Cobertura Sentencies	85%	Bloquejant
% Èxit Proves Unitàries	100%	Bloquejant

3.2.2. Proves funcionals

Els defectes reportats en l'execució del pla de proves es classifiquen en base a les següents severitats:

- **Bloquejant**: defectes trobats en casos de prova del *Sanity Check*, o aquells que no permeten continuar amb l'execució del pla de proves.
- **Alta**: defectes trobats en els comportaments principals del *WebService*.
- **Mitjana**: defectes trobats en els comportaments alternatius del *WebService*.
- **Baixa**: defectes estètics trobats en qualsevol dels comportaments.

Cal diferenciar entre:

- **Error**: acció realitzada per un ésser humà que produeix un resultat incorrecte.
- **Defecte**: el resultat d'un error en el codi o en un document que pot provocar una desviació respecte al funcionament esperat.
- **Fallada**: desviació respecte al funcionament esperat.

Es reportarà un defecte per cada fallada detectat.

3.2.3. Proves no funcionals

3.2.3.1. Proves de seguretat

Les no conformitats detectades en les proves de seguretat es classifiquen en base a les següents severitats definides seguint les recomanacions del OWASP (*Open Web Application Security Project*) per a les proves de seguretat.

Es reportarà un únic defecte representatiu de totes les no conformitats detectades. La severitat del defecte serà igual a la major severitat de les no conformitats detectades:

Tipologia de proves	Tipologia	Llindar d' acceptació	Severitat defecte a reportar
Estàtiques (Sonar)	*Hardcoded constant database password*	0	Alta
	A prepared statement is generated from a nonconstant String	0	Alta
	Array is stored directly	0	Alta
	Empty database password	0	Alta
	Method returns internal array	0	Alta
	Nonconstant string passed to execute method on an SQL statement	0	Alta
Dinàmiques (*SoapUI*)	*SQL Injection*	0	Alta
	XPath Injection	0	Alta
	Cross Site Scripting	0	Alta
	XML Bomb	0	Alta
	Malformed XML	0	Alta
	Fuzzing Scan	0	Media
	Invalid Types	0	Media
	Boundary Scan	0	Media

3.2.3.2. Proves de rendiment

Les no conformitats detectades en les proves de rendiment es classifiquen en base a les següents severitats:

- **Bloquejant**: afecta al rendiment i/o a l'estabilitat del sistema i impedeix realitzar la resta de proves de rendiment.
- **Alta**: pot afectar al rendiment i/o estabilitat del sistema.
- **Mitjana**: no aplica per a aquests tipus de proves.
- **Baixa**: no aplica per a aquests tipus de proves.

Es reportarà un únic defecte representatiu de totes les no conformitats detectades. La severitat del defecte serà igual a la major de les severitats que es done en les no conformitats detectades:

Tipologia de prova	Tipologia	Llindar d'acceptació	Severitat defecte a reportar
Carga	Temps mitjà resposta (mil·lisegons)	500	Bloquejant
	Peticiones errònies (rati 1/1000)	0,001	Bloquejant
	Memòria RAM estable? (SI/NO)	NO	Bloquejant
	CPU estable? (SI/NO)	NO	Bloquejant
	Número C.C. inestabilitat RAM	0	Bloquejant
	Número C.C. inestabilitat CPU	0	Bloquejant
Estrès	Temps mitjà resposta (mil·lisegons)	600	Alta
	Peticiones errònies (rati 1/1000)	0,001	Alta
	Memòria RAM estable? (SI/NO)	NO	Alta
	CPU estable? (SI/NO)	NO	Alta
	Número C.C. inestabilitat RAM	0	Alta
	Número C.C. inestabilitat CPU	0	Alta
Estabilitat	Temps mitjà resposta (mil·lisegons)	500	Alta
	Peticiones errònies (rati 1/1000)	0,001	Alta
	Memòria RAM estable? (SI/NO)	NO	Alta
	CPU estable? (SI/NO)	NO	Alta
	Número C.C. inestabilitat RAM	0	Alta
	Número C.C. inestabilitat CPU	0	Alta

3.3. Gestió dels actius de proves

El terme 'actiu de prova' es refereix a tots els productes de suport associats a les activitats realitzades per la Factoria.

Una gestió eficient dels actius de prova permet disposar d'un mecanisme de reutilització dels mateixos. Per a la reutilització és necessari realitzar un manteniment efectiu dels actius de prova generats i aplicar de manera metòdica i sistemàtica les normes de gestió de configuració definides.

Existeixen actius de prova de caràcter intern a FdP i altres susceptibles de ser entregats al client. Els dos tipus deuen ser emmagatzemats, versionats i mantinguts en la ubicació corresponent.

A nivell intern de FdP s'ha creat un repositori associat a cada petició, que respecta l'estructura definida per a l'execució de les proves. El responsable de la petició és el rol encarregat d'assegurar que el directori de la mateixa estiga sempre actualitzat i que la informació siga consistent.

A nivell extern, s'ha de fer el *QA Proves* que s'encarrega de gestionar els actius de prova generats per FdP assegurant que s'ubica en les carpetes definides per a tal propòsit en la ferramenta *ClearCase*.

3.3.1. Enfocament de proves

L'enfocament de proves és el document on es particularitza l'estratègia definida per a un component específic.

En base al tipus de petició i a les operacions del *WebService*, es defineixen les proves que es deuen realitzar en cada cas.

És possible que en el curs de l'execució de les proves es presenten variacions respecte al plantejament descrit en l'enfocament de proves degut a motius aliens a FdP. Qualsevol comunicació en temps real als rols involucrats en el procés i serà posteriorment descrit en l'informe final de les proves.

3.3.2. Pla de proves

El Pla de Proves és el document que defineix al detall les proves que validen els requeriments definits en el DTAN, per la qual cosa inclou la definició al detall de les proves que es van a realitzar.

El PLP es dissenya sobre un document *Excel*. En la definició dels casos de prova es determina:

- Flux del DTAN que valida.
- Identificador del cas de prova.
- Objectiu de la prova, és a dir, el comportament que es valida.
- Condicions que es deuen donar per a executar el TC.
- Paràmetres d'entrada que modelen el comportament.
- Eixa esperada (bassada en l'especificació del DTAN).

3.3.3. Full de defectes

El Full de Defectes és el document que defineix el detall dels defectes reportats per FdP en l'execució de proves.

El Full de Defectes és reportat actualment i es dissenya sobre la ferramenta *ClearQuest*. Els defectes es classifiquen en base a quatre estats i en cada estat es defineixen els camps que deuen ser informats:

- **Obert**: identificació de la petició, prioritat, resum, etc.
- **Solucionat**: responsable de desenvolupament, data inici, data de resolució...
- **Rebutjat**: motiu de desestimació, data de desestimació i documentació associada.
- **Tancat**: detalls, data i responsable.

3.3.4. Guia d'execució de proves

Les proves realitzades per FdP deuen acomplir la característica de reproductibilitat, de manera que siga independent de qui les execute i que els resultats siguen els mateixos.

A la fi de l'execució de les proves FdP elabora i entrega la guia d'execució, en la que s'expliquen els passos que es deuen dur a terme per poder reproduir les proves que generen *scripts* i automatitzacions:

- Proves funcionals
  - S'indica el fitxer que es deu executar.
  - S'indiquen els passos que es deuen dur a terme per a reproduir les proves de *Sanity Check*, Versió i Regressió (projectes *SoapUI*).
- Proves de seguretat dinàmiques:
  - S'indica el projecte *SoapUI* corresponent.
- Proves de rendiment
  - S'indica el fitxer que deu ser executat (projecte *SoapUI*).

3.3.5. Joc de dades

El desenvolupament del joc de dades és un procés fonamental per a que en l'execució de les proves funcionals es garatitzen les precondicions de prova que s'han definit en el PLP.

A continuació, es descriuen els passos que es segueixen:

- En el PLP hi ha condicions que deu modelar el joc de dades de proves.
- Depenent de l'existència de registres en taules implicades en la implementació de la funcionalitat:
  - Existeixen dades: en aquest cas, es seleccionaran les dades que coincideixen amb les condicions establertes en el document PLP i es generaran els *scripts* de proves corresponents.
  - No existeixen dades: es generaran directament els *scripts* de proves.

Aquests *scripts* contenen les instruccions necessàries per emplenar la BBDD amb les dades pertinents per a cada prova a realitzar. Hi ha quatre tipus de fitxers de joc de dades:

- **DSC**: conté les instruccions necessàries per a introduir en la BBDD les dades corresponents als cassos de prova de sanity check.
- **DAT**: està compost per les instruccions per a dur a terme totes les altres proves funcionals. Ver informat a partir del sanity check.
- **DMA**: instruccions d'eliminació per a totes les insercions anteriors.
- **ETF**: aquest joc de dades està compost per les entrades del *SoapUI* per a la petició. Es disposa de ferramentes per a automatitzar aquest tipus de *script*, *ETFGen*.

3.3.6. *Scripts* de proves

Els *scripts* de proves són els projectes implementats amb *SoapUI* que permeten comprovar que el *WebService* desenvolupat funciona tal i com es modela en el DTAN. Es segueix una estratègia enfocada a l'automatització, per tal de facilitar les futures proves. Açò suposa la generació de *scripts* de proves que s'executaran sempre que siga necessari realitzar un nou cicle de proves.

A continuació s'expliquen els tipus de projectes de prova generats per FdP:

- **SCC**: projecte que implementa els casos de prova funcionals de *Sanity Check*.
- **SCV**: projecte que inclou els casos de prova funcionals que validen els requeriments del DTAN (cicle 1) o un conjunt de canvis (cicle>1).
- **SCS**: projecte que implementa els casos de prova de seguretat.
- **SCP**: projecte que implementa els casos de prova de rendiment (càrrega, estrés, estabilitat i escalabilitat).

3.3.7. Informe final de proves

Els resultats de les diferents activitats de proves es documenten en els informes de proves que s'han definit amb eixe objectiu.

Per a cada cicle i tipologia de prova que es duga a terme s'elabora un informe de proves de caràcter intern.

Una vegada finalitzats tots els cicles de prova pertinents, s'elaborarà i entregarà l'informe final de proves, que conté la informació dels resultats de les proves realitzades en cada cicle. L'informe final té com a objectiu recopilar i presentar els resultats obtinguts després de l'execució de les proves aplicades al *WebService* de la petició.

L'informe final de proves s'estructura de la següent manera:

- Anàlisi executiu
  - Quadre resum: indica de forma visual les alertes trobades.
  - Resultats per cicles d'execució de proves: indica els resultats de les proves realitzades en cada cicle i el *baseline* corresponent.
  - Nombre total de defectes per tipus de prova: gràfic que mostra els defectes detectats i pendents de resoldre per tipus de prova.
  - Defectes: un llistat dels defectes amb la seua severitat i el tipus.

  - Volumetria: dades de partida abans de processar el *WebService* i les dades després de processar-lo en FdP.

- Detall de les proves
  - Les no conformitats esperades i la seua severitat
  - Els defectes reportats.
  - Els defectes pendents de resolució.

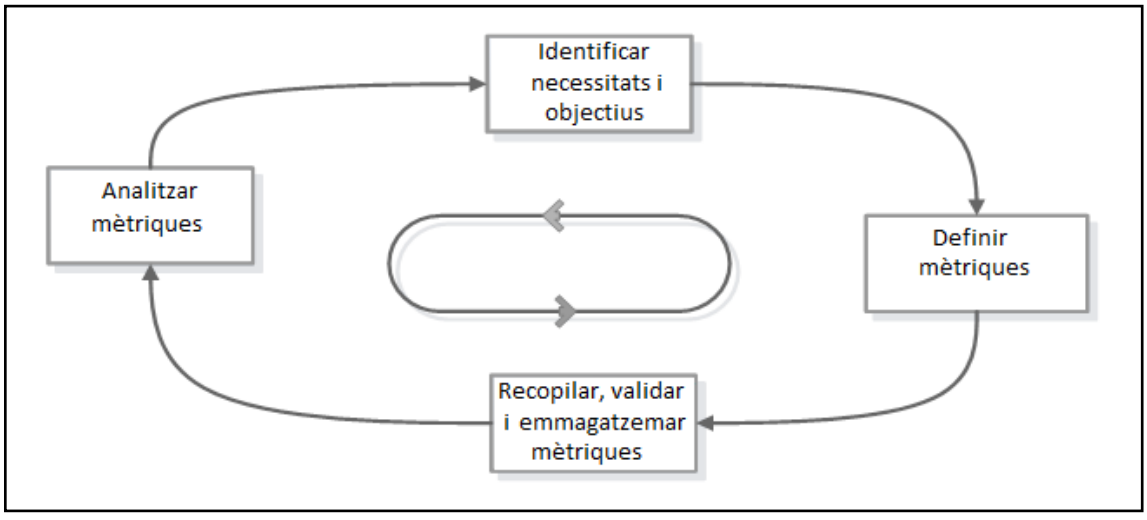
En el cas de les proves funcionals s'indica, per a cada cicle, la cobertura dels cassos de prova i la relació entre casos de prova OK/KO per a cada tipus de prova funcional (*Sanity Check*, Versió i Regressió).

3.4. Mètriques

Les mètriques són mesures quantitatives que permeten obtindre una visió global de la qualitat dels components construïts i de la eficàcia del procés global de proves. Es medeix per aconseguir:

- **Caracteritzar**: obtindre coneixement del processos i projectes per poder comparar posteriorment.
- **Avaluar**: determinar el grau de compliment de les planificacions i els objectius de qualitat.
- **Pronosticar**: poder predir i planificar.
- **Millorar**: identificar els obstacles, ineficiències i altres oportunitats de millora de la qualitat i rendiment del procés de proves.

Un procés de medició estructurat i repetible defineix les activitats de medició dels serveis. El procés de medició és el següent:



Les mètriques es poden classificar sobre els següents dos tipus de qualitat:

- **Producte**: dirigides a augmentar la qualitat dels components entregats. Avalua les característiques intrínseques del component. Es tracta de les

mètriques definides en (3.4.Mètriques) del present document per a cada tipus de prova.

- **Procés**: orientades al control i la millora del procés de proves definit. Les mètriques de procés proporcionen informació sobre el procés, el grau d'ajust de les activitats executades respecte a les definides, possibles problemes, aspectes a millorar, etc. Addicionalment i a nivell intern, FdP defineix indicadors necessaris per a mesurar l'efectivitat de cadascuna de les fases del procés global de proves. Aquests indicadors constitueixen una de les entrades principals per al procés de millora contínua (3.6.Millora contínua).

3.5. Rols i responsabilitats

A nivell de Factoria de Proves, es defineixen els següents perfils:

Perfil	Objectius y responsabilitats
Tester	- Preparar els *scripts* de proves. - Generar les dades de les proves. - Preparar els entorns de proves. - Executar les proves. - Comprovar els resultats de les proves.
Analista de proves	- Assegurar la qualitat, la cobertura total i el rigor de las proves a realitzar sobre una petició. - Actuar como responsable de la petició. - Dissenyar el pla de proves. - Donar suport als testers en l'execució de les proves. - Validar els resultats de l'execució de las proves. - Validar els defectes detectats. - Reportar els defectes.
Cap d'equip	- Col·laborar en la estimació de la petició. - Col·laborar en la planificació i assignació de recursos per a les peticions. - Garantir la qualitat de les entregues i estandardització del treball realitzat. - Detectar problemes i proposar solucions i millores. - Actuar como responsable de la petició.
Responsable millora continua	- Promoure l' estandardització de la factoria, amb objectius per al increment de la productivitat. - Definir processos i procediments: o Externs: Processos comuns FdP-FdD-client, . o Interns: procediments detallats de proves, estratègia de proves, etc. - Responsable del pla de millora contínua : o Participar en l'avaluació i anàlisis. o Definició d'accions i seguiment. - Responsable Gestió del coneixement: o Gestió del repositori de documentació. o Promoure accions de formació. - Participar en els comitès de qualitat i millora.
Cap de Projecte	- Responsable principal de la factoria de proves. o Organització del equipo o Coordinació de la definició de processos (externs i interns), ferramentes i entregues - Màxim responsable del projecte o Comercial

Perfil	Objectius y responsabilitats
	o Contracte o Econòmic - Responsable de l'aplicació dels valors y normes de la companyia en el projecte, tenint en ment els objectius del projecte tant per a la companyia com per al client. - Responsable en primera persona del seguiment operatiu diari. o Assignació d'esforços, terminis i recursos o Seguiment continu de l'avanç de les tasques i la capacitat de l'equip - Participar en els comitès operatius, de seguiment, així com en els executius i estratègics.

3.6. Millora contínua

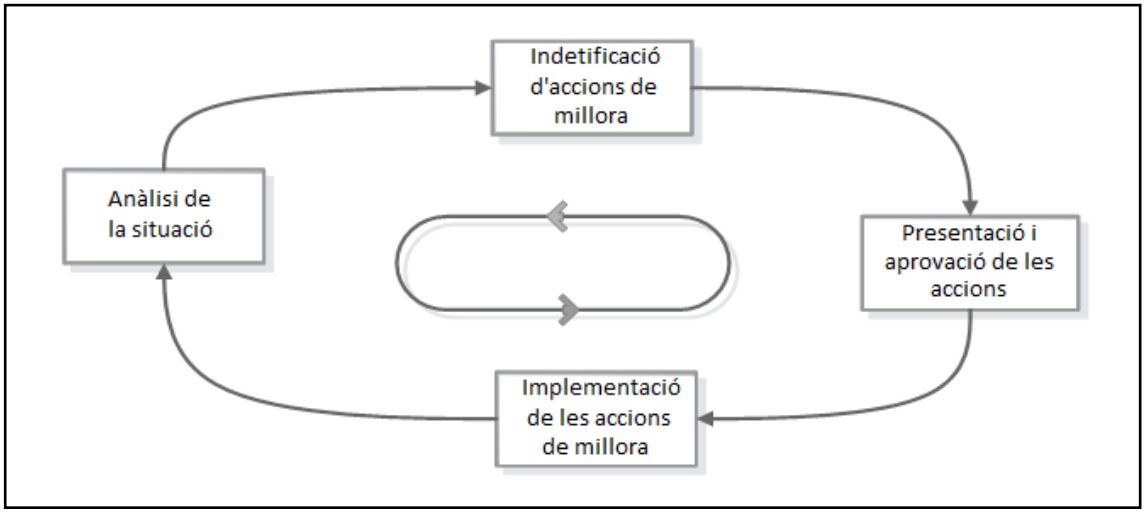
Per tal de millorar la qualitat del servei oferit s'han d'avaluar els components del servei periòdicament per aconseguir una evolució progressiva. Per a identificar les oportunitats de millora es realitzaran les següents activitats:

- Anàlisi dels resultats de l'execució del servei.
- Avaluació periòdica.

El procés de millora continua es divideix en quatre passos:

- Anàlisi de la situació actual per tal de descriure la situació actual del procés de prova.
- Identificació de les accions de millora
- Presentació i aprovació de les accions de millora per part del Comité de Qualitat.
- Implementació de les accions de millora planificades. És en aquest punt on s'apliquen les mètriques definides i s'avaluen els resultats.

El cicle de millora continua és el següent:



4. Cicle de vida

4.1. Creació nova petició

Quan arriba una petició nova a factoria el primer pas es crear l'estructura per emmagatzemar tots els documents a nivell de FdP, ja que a nivell de factoria és el client qui crea l'estructura per alliberar la petició. Aquesta creació es responsabilitat del cap d'equip. Una vegada creada l'estructura en *Jira* i SVN ja es procedeix a la validació del DTAN.

4.2. Validació DTAN

Una vegada creada l'estructura de la nova petició el següent pas és la validació del DTAN. Aquest document es valida tant per FdD com per FdP. Açò es degut a que es tracta d'un punt molt important en el cicle de vida. Ja que una vegada validat aquest document conforma els fonaments de les proves i del desenvolupament. En aquest moment si es detecta alguna incongruència s'anomena dubte, és així per política del client.

Es disposa de dos dies per a dur a terme la validació per ambdues factories. Fins que les dos no donen per bo el document aquest no es pot donar per validat. Durant aquest procés es designa un analista encarregat de dur a terme la validació del DTAN i un tester que passarà també la CH. En aquest punt es on es realitza la primera prova pròpiament dita, la revisió de documentació del DTAN. És important dur a terme aquesta prova al mateix moment que l'analista realitza la validació perquè així s'ajuda a detectar dubtes.

4.3. Estimacions

Una vegada el DTAN està validat el següent pas a seguir en el cicle de vida és l'estimació del cost de la petició per part de FdD i FdP, una per a estimar el desenvolupament i l'altra les proves. Aquestes estimacions ja es duen per separat però es deuen entregar el mateix dia. El dia corresponent a l'entrega de les estimacions al client és el següent a la validació del DTAN.

Després d'entregar les estimacions per part de les diferents factories el client disposa de dos dies per a acceptar-les. Arribat a aquest punt ja ha finalitzat la primera setmana del cicle de vida imposat pel client.

4.4. Primera entrega

El sext dia des de l'inici de cicle FdP entrega el pla de proves *sanity check* junt amb les dades corresponents. La creació del PLP *sanity check* es responsabilitat de l'analista. I com s'explica en punts posteriors d'aquest document tan sols inclou els mínim nombre de cassos per tal de comprovar la funcionalitat del *WebService*. Aquest mateix dia FdD deu entregar el WSDL i el DTD. Una vegada factoria de proves disposa

d'aquest dos documents es revisa, és a dir, es realitza la segon prova de revisió de documentació, però aquesta vegada la *check list* corresponent al DTD.

Quatre dies després de l'entrega del PLP *sanity check* es deu entregar el PLP genèric a FdD, aquesta tasca correspon al analista assignat a la petició.

Aquestes dues proves de revisió es deuen dur a terme sempre que hi haga una nova entrega de documentació.

4.5. Preparació de proves

Després de la primera entrega, la del PLP, es disposa d'una setmana per a preparar els scripts, el PLP genèric i el complet i els jocs de dades. En ordre de prioritat el PLP és el més urgent. Normalment aquesta tasca es du a terme pel analista, però de vegades si el PLP genèric ja està fet correspon al tester acabar de col·locar els valors i completar les entrades i eixides. En el punt de documents s'explica en més detall.

Una vegada el PLP està complet ja es pot començar a prepara els scripts o els jocs de dades. Aquesta preparació es du de forma independent l'una de l'altra.

El joc de dades es prepara amb l'ajuda de les dades enviades junt al PLP *sanity check*, el DSC. Aquest serveix d'exemple de les taules a informar per a la realització de les proves. Hi ha que preparar dos jocs de dades, el d'inserció i el d'eliminació de les dades.

La preparació dels scripts inclou els quatre tipus de projectes *SoapUI* (SCC, SCV, SCS i SCP) i la generació del ETF.

Per generar el ETF s'utilitza la ferramenta desenvolupada per FdP, *ETFGen*. Per a generar-lo tan sols hi ha que obrir la ferramenta, llistar el PLP corresponent a la petició, el cicle i crear el ETF.

La creació dels projectes de *SoapUI* és més laboriosa.

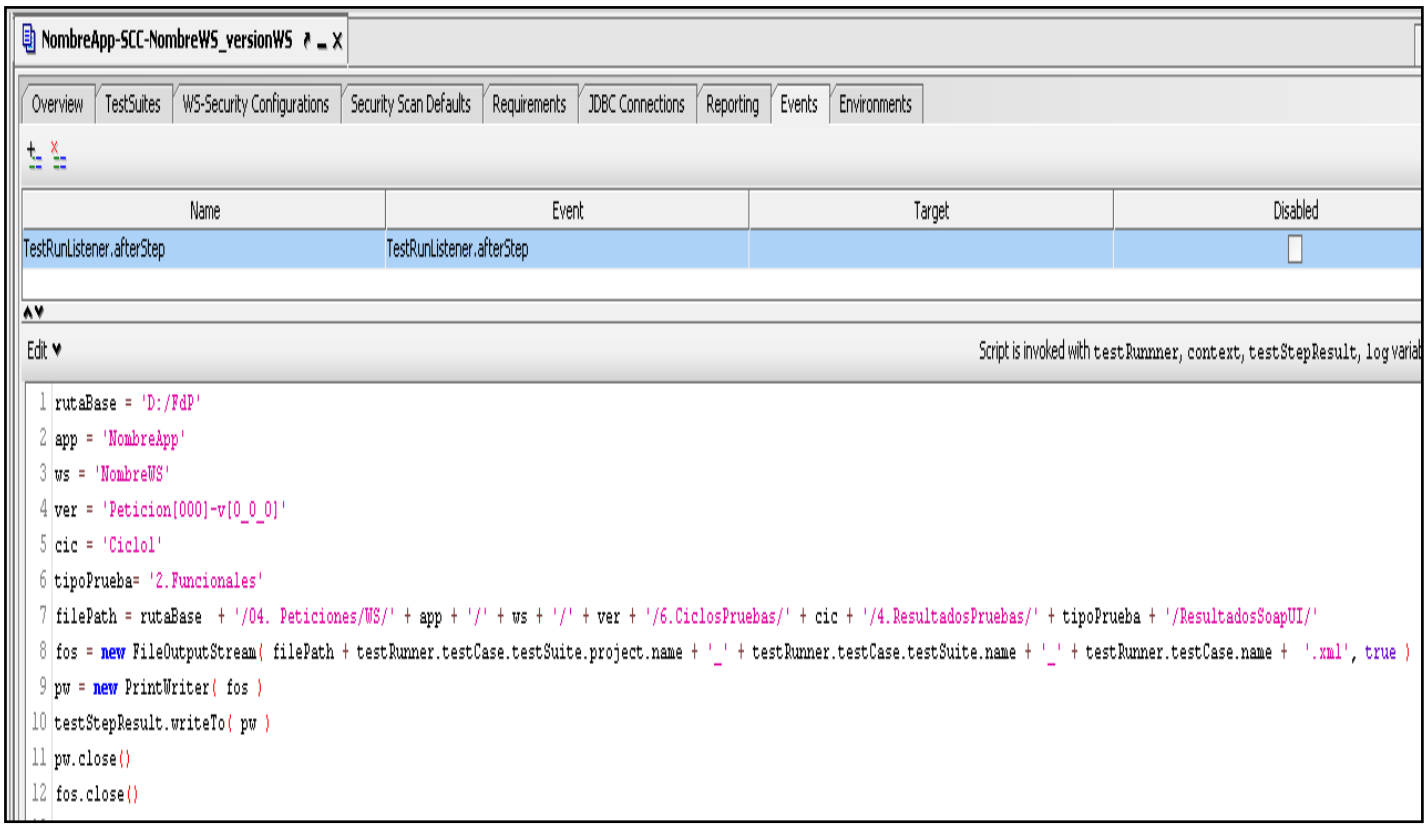
4.5.1. Proves funcionals

Les proves funcionals consisteixen en la realització de les proves necessàries per assegurar que s'acompleixen els requisits funcionals descrits en el DTAN intentant arribar a una cobertura total de funcionalitat. Pel seu caràcter repetitiu, cobra sentit que aquest tipus de proves s'automatitzen al màxim a través de *scripts* que s'elaboren i es mantenen d'acord al disseny del PLP.

Hi ha dos projectes per a les proves funcionals: el SCC que està format pels cassos de prova que conformen el PLP de *sanity check* i el SCV que conté tots els altres cassos del PLP complet.

A continuació es van a descriure els passos a seguir per poder automatitzar una prova, des de l'associació amb el WSDL fins que es disposa de la prova preparada per al llançament quan es tinga el WS. La idea es definir un conjunt de passos sobre un cas d'una petició i poder repetir-los les vegades que siguen necessàries.

Per a la preparació de les proves el primer pas a realitzar és el re-nomenament dels scripts amb el corresponent nom propi de la petició.

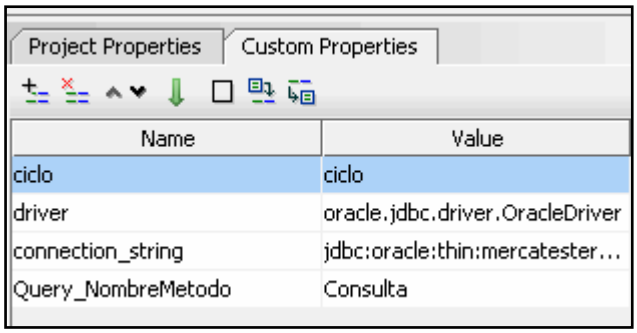


S'han d'incloure unes configuracions a nivell de projecte per a facilitar l'automatització de les proves. Per una banda, el directori on es guarden els *logs* propis de la ferramenta *SoapUI* i, per l'altra, configuració de variables globals.

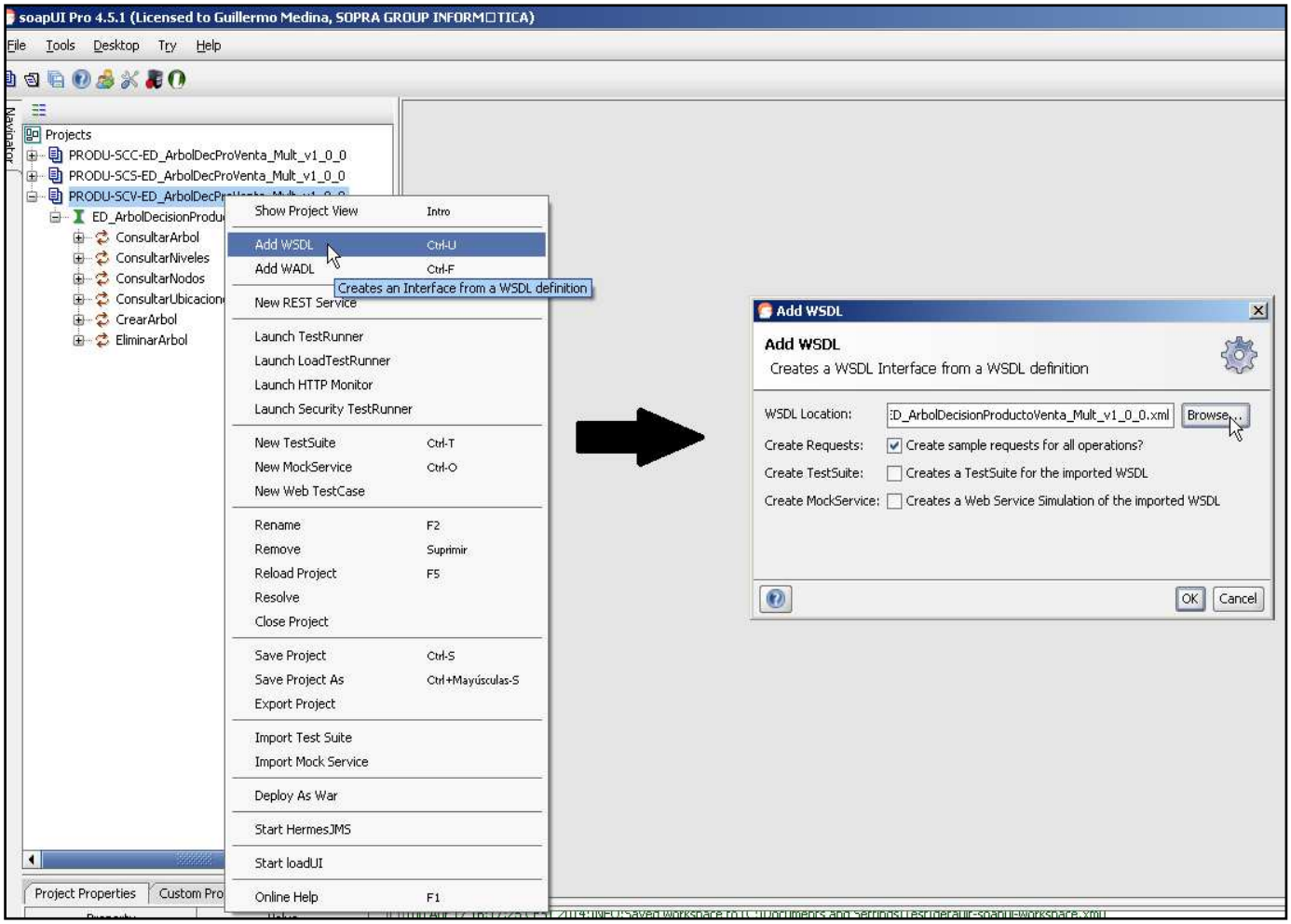
Per indicar el directori on es van a guardar els *logs* de *SoapUI* hi ha que obrir el projecte i situar-se en la pestanya "*Events*" i seleccionar tal com mostra la imatge:

Ací hi ha que realitzar alguns canvis respecte a la plantilla: nom aplicació, nom del *WebService*, codi de la petició i cicle. Amb aquests canvis quan executem les proves automàticament es guardaran els *logs* en el directori propi de la petició.

El següent pas es configurar els paràmetres globals del projecte. Per a canviar-los hi ha que seleccionar el projecte i en la part inferior esquerre de la finestra de *SoapUI* hi ha que afegir les *queries* corresponents en la pestanya "*Custom properties*". La nomenclatura de les *queries* és "Query_NomOperació" que es fica en la columna "*Name*". En la columna de "*Value*" hi ha que copiar la query que apareix al final de l'operació del ETF. En l'apartat "*selects*". La següent imatge mostra un exemple:

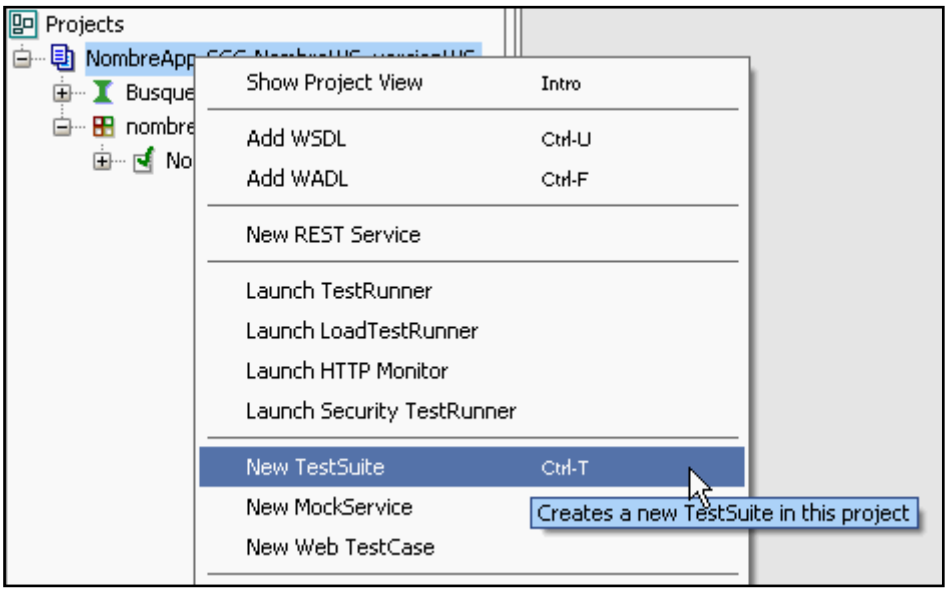


Seguidament afegim el WSDL corresponent. Per això pressionem el botó dret a nivell de projecte "*Add WSDL*" i seleccionem el directori on es troba el WSDL, recordar que hi ha que marcar la primera opció, "*Create Requests*" tal com mostra la següent imatge:



Fins ací és igual la preparació del SCC i el SCV, ara els canvis un respecte a l'altre.

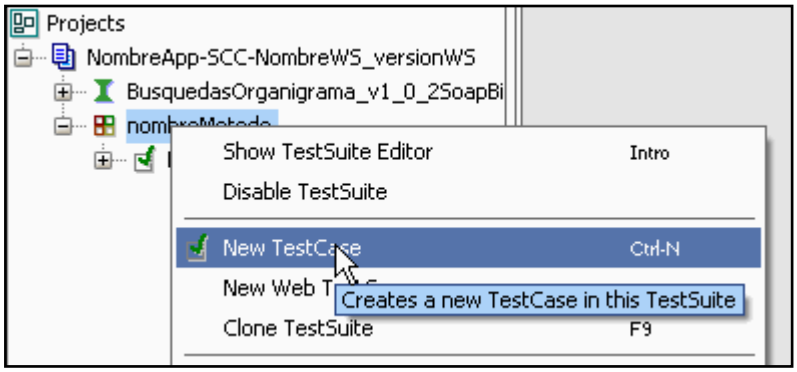
Ara si es tracta del projecte de SCC es crea un *TestSuite* per a cada operació. Per a crear un TS cal donar-li al botó dret sobre el projecte i elegir "New *TestSuite*" o la drecera de teclat Crtl+T:



La nomenclatura del TS deu ser el nom de l'operació. Després creem els cassos corresponents que conformen el SCC de l'operació (normalment es tan sols un SC per operació però no sempre).

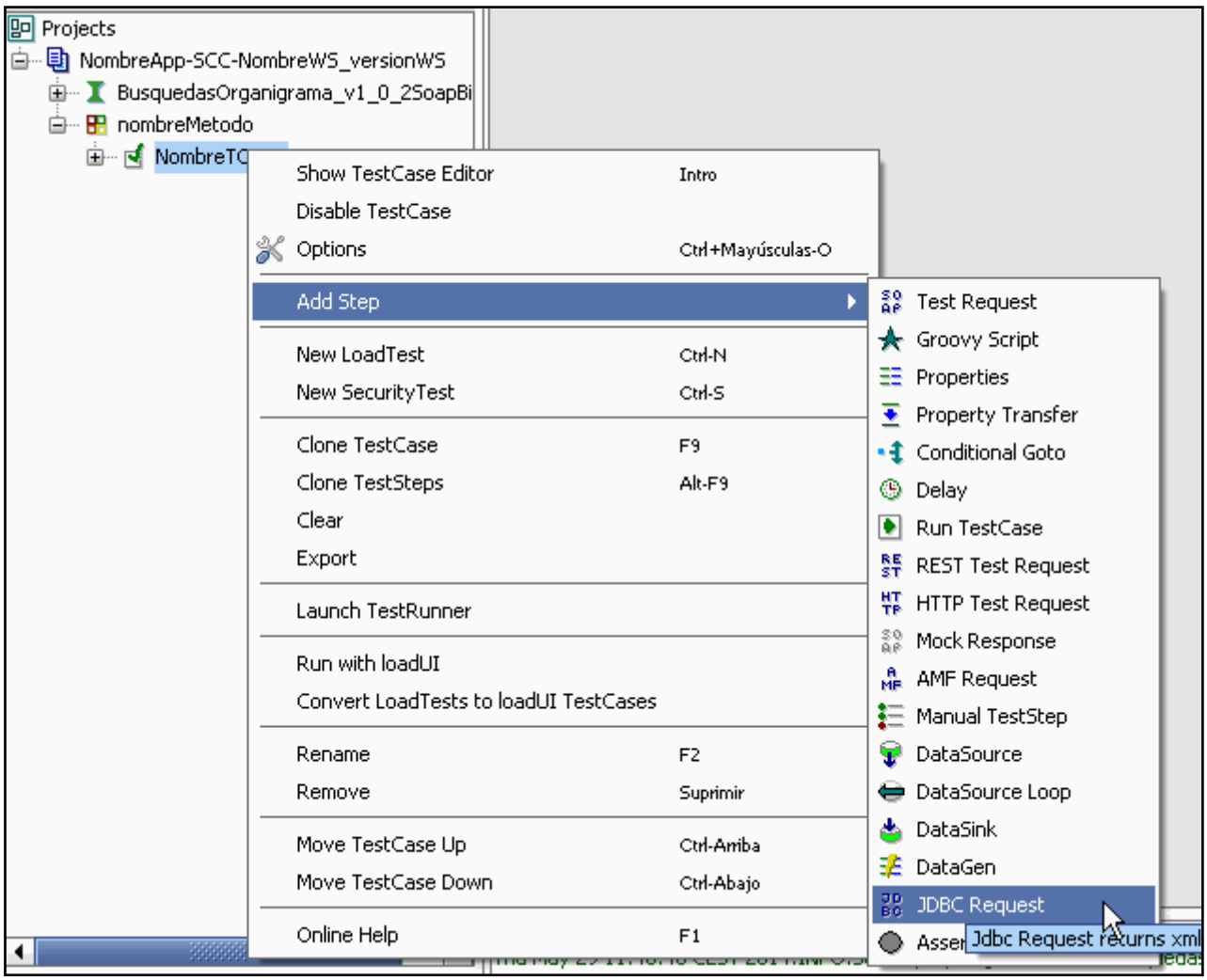
Per preparar el projecte de SCV es creen tres *TestSuit* per operació. Els tres s'anomenen igual però tenen extensió diferent: _SIM, _COM o _ERR. Açò es fa per a ordenar els cassos de prova. En el _SIM i el _COM es creen els TC corresponents a una funcionalitat amb resultat correcte però en el _SIM s'introdueixen els que tenen una eixida simple, mentre que en els altres _COM com a resultat tenen més d'un registre a l'eixida. En el _ERR van els TC que tenen una funcionalitat amb un resultat erroni.

A l'hora de crear els TC per a les proves en ambdós projectes es fa de la mateixa forma. Per crear un TC a nivell de TS "*New TestCase*" i es segueix la nomenclatura del PLP (OP_XX-F-YYY on XX és el nombre de l'operació i YYY el nombre de TC):

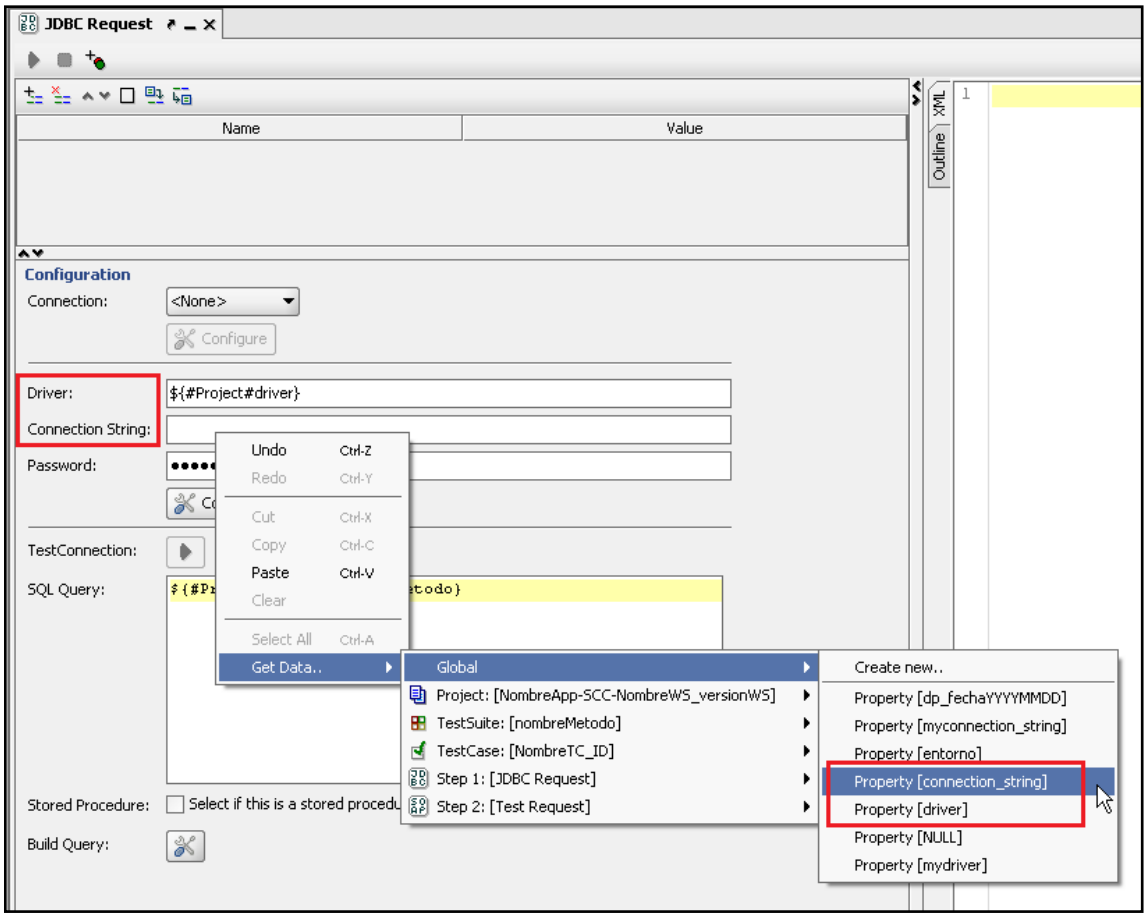


Un *TestCase* està format per dos passos o *steps*: el "*JDBC Request*" i el "*Test Request*". El primer és l'encarregat de connectar-se a la base de dades per obtindre els paràmetres d'entrada del *TestCase* a executar, mentre que el segon composa la interfície gràfica dels paràmetres que se li passen i l'eixida si en té. Es deu seguir l'ordre de creació per a que al llançar-los a l'hora d'executar-se es puga fer a nivell de *TestCase* i no de *Request*.

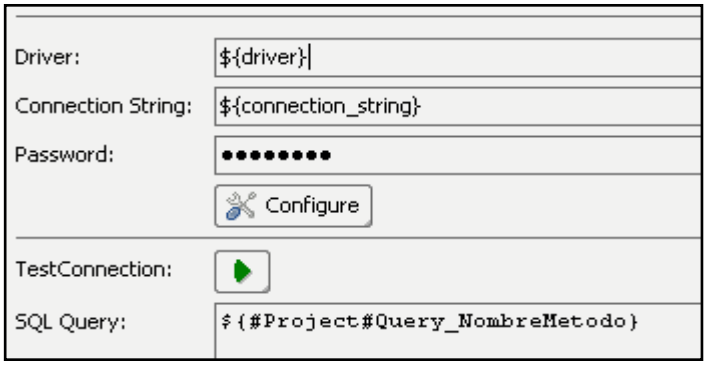
Per crear-los cal fer *click* dret del ratolí sobre el *TestCase* per afegir el *JDBC Request*, tal com es mostra en la imatge:



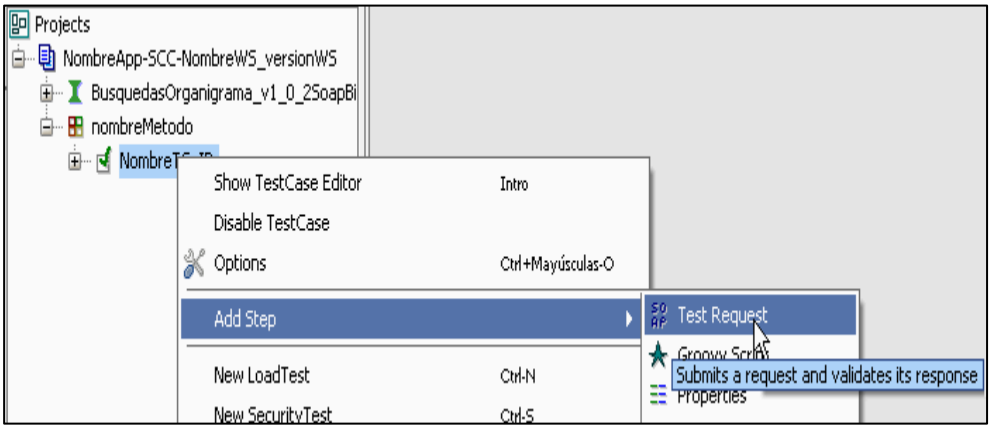
Aquest *Step* necessita d'una configuració determinada per a que funcione amb la base de dades pertinent. Es necessita el nom de la base de dades, l'usuari per a connectar-se i la seua contrasenya i la *query* que farà la petició sobre la BBDD. (afegir en les imatges que la *query* és la del ETF). Les següents imatges mostren com configurar els paràmetres d'usuari:



A continuació es veu un exemple ja complet de la configuració del *JDBC Request*:

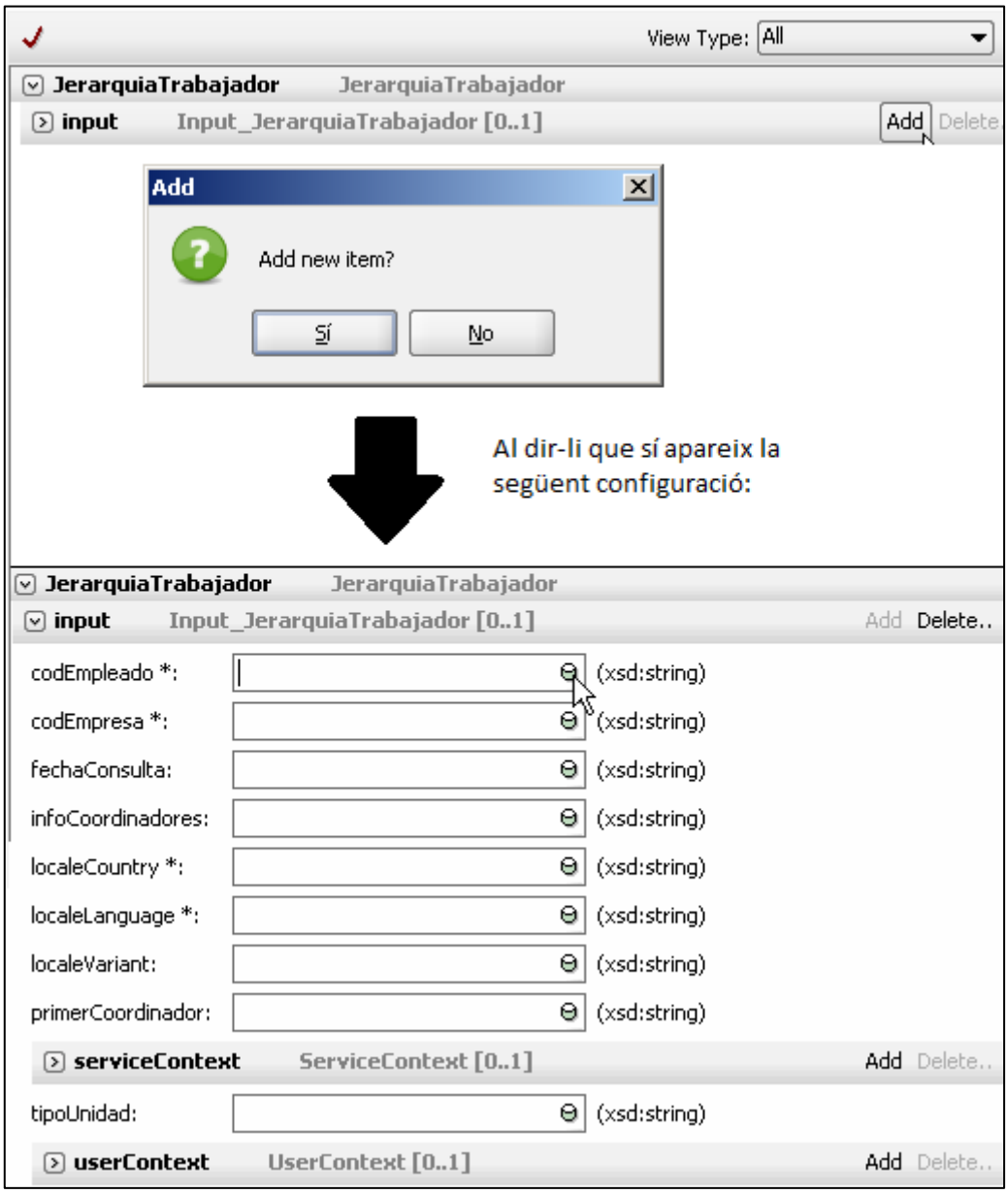


Una vegada creat el *JDBC Request* ja es crea el *Test Request*. Ho fem de la mateixa manera que l'anterior:

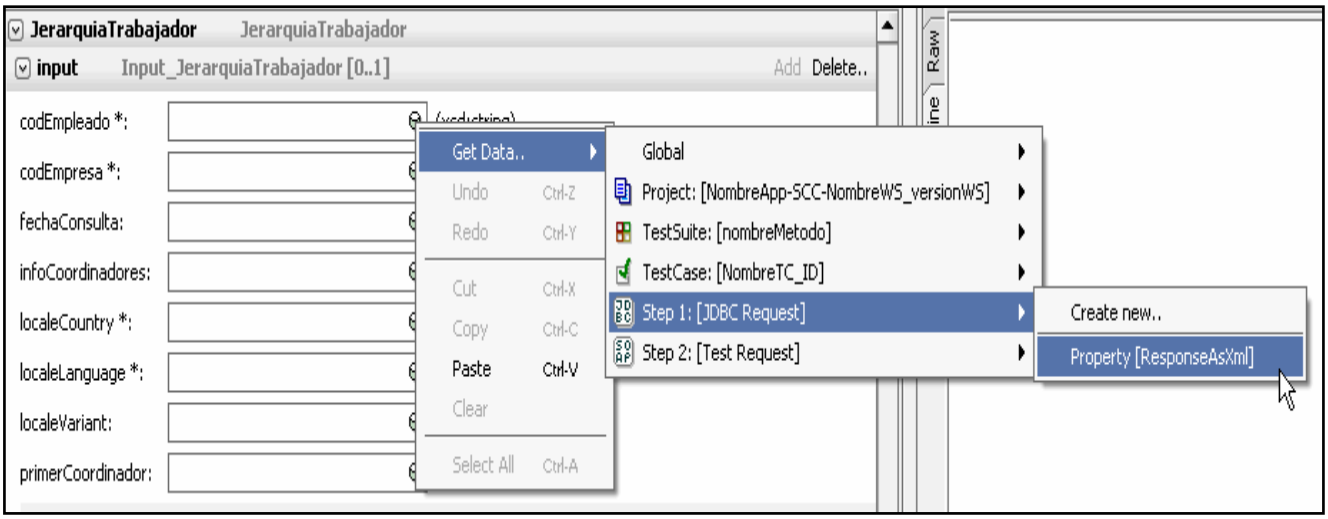


Aquest s'ha de configurar de manera que les dades que li arriben al *SoapUI* del pas anterior siguen les que se li passaran al *WebService* quan s'execute. Per tal de configurar els paràmetres amb els valors del JDBC s'han de seguir els següents passos:

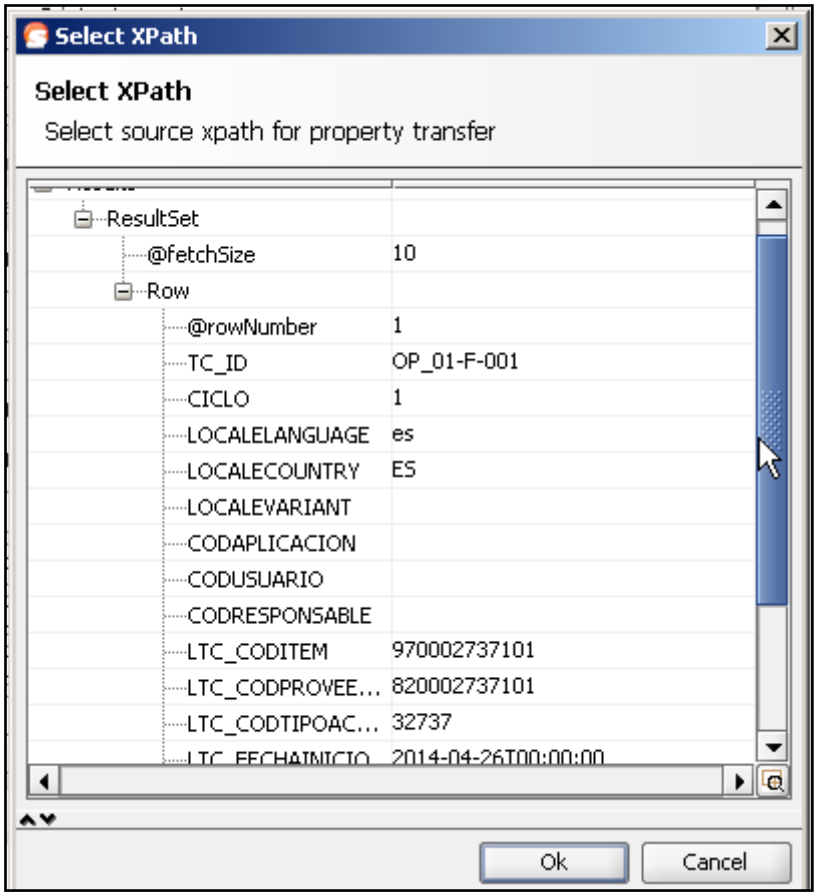
El primer pas es obrir el *Test Request* i donar-li al botó de "*Add*" de manera que es carregue la interfície gràfica dels paràmetres d'entrada:



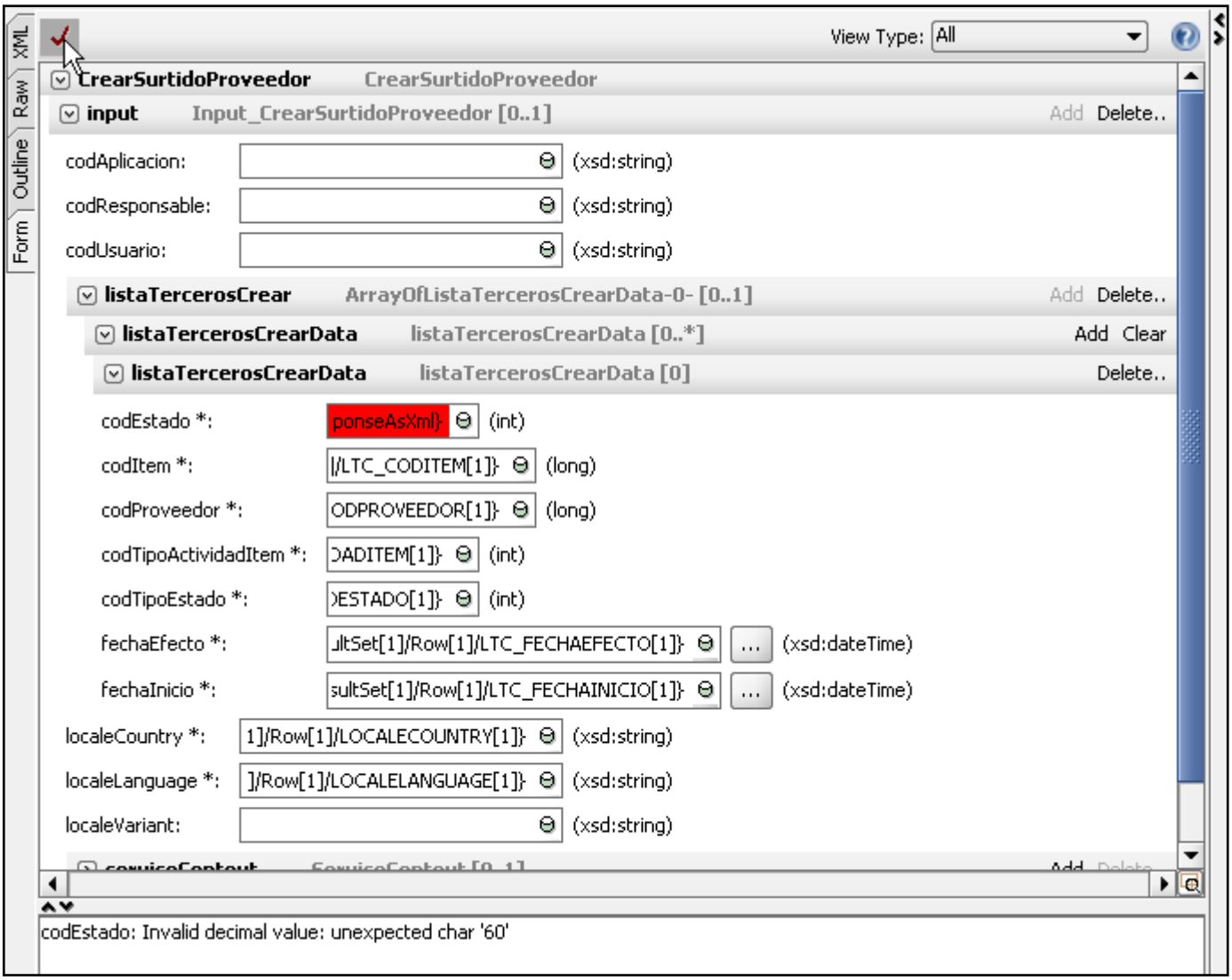
Després s'han de passar els valors del JDBC. Per configurar-ho es prem el botó redó que hi ha al final del camp i es segueix tal com en la imatge:



Aquest pas ens obri una nova finestra, d'on s'ha de seleccionar el paràmetre provinent del JDBC:

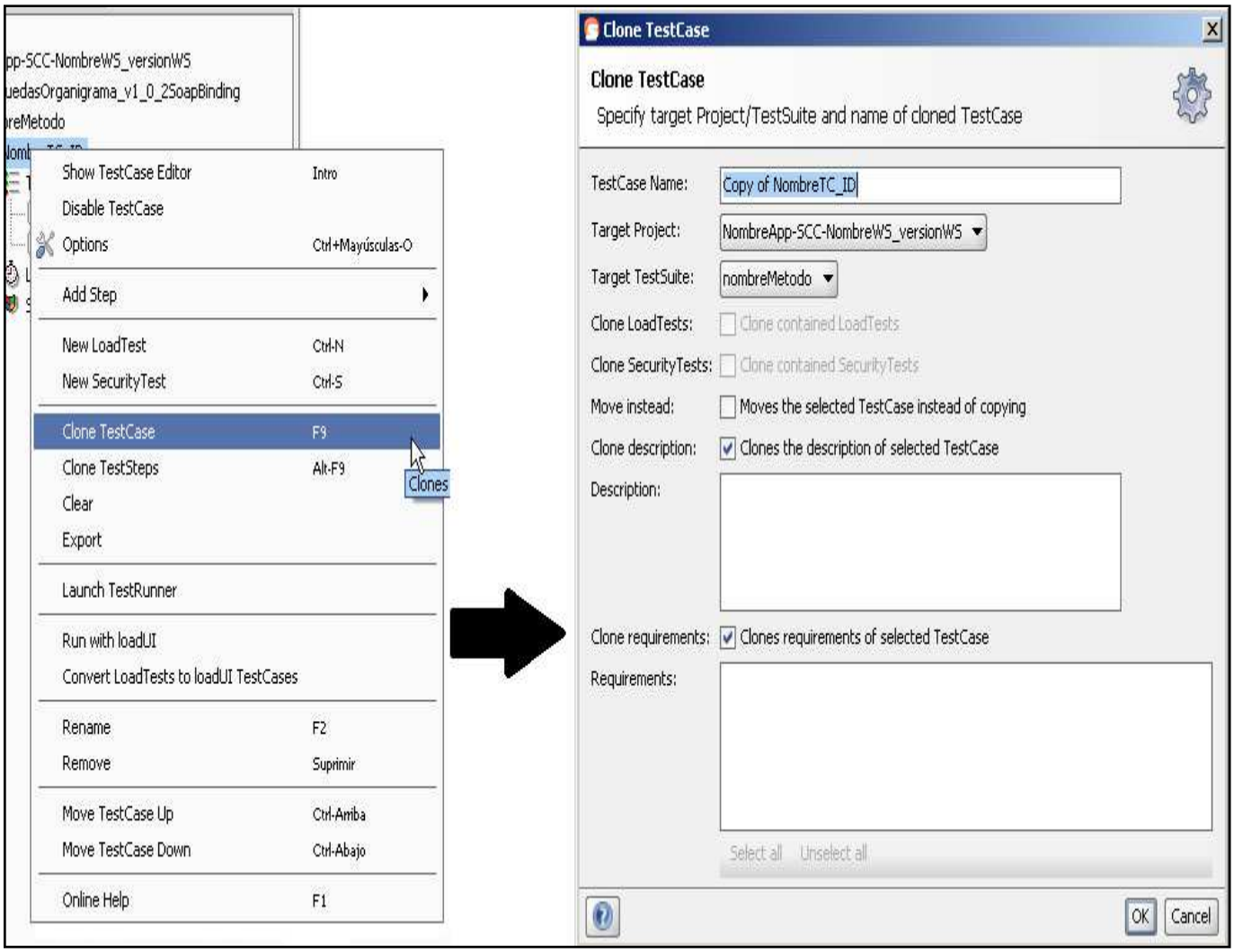


Una vegada que tots els paràmetres d'entrada pertinents al *TestCase* a provar estiguen configurats es poden validar per veure si algun d'ells no està informat de la manera adequada. Per a validar-los hi ha un botó en forma de *tick* roig en la part superior de la finestra dels paràmetres d'entrada:



Si algun dels paràmetres té un valor incorrecte, com puga ser una cadena de text en un camp numèric, aquest paràmetre es marcarà en roig i baix apareixerà la descripció de l'error.

La creació de *TestCase* es repetirà tantes vegades com *TestCase* tinga el PLP. Com es tracta d'una tasca molt repetitiva podem fer ús de l'opció de clonar del *SoapUI*. S'ha de tenir cura a l'hora de clonar *TestCase* perquè es copien totes les propietats d'un TC a un altre. És aconsellable només clonar TC de la mateixa operació ja que els paràmetres representats en el *Test Request* poden no correspondre d'una a un altra. Per a clonar un TC s'ha de seleccionar el TC a clonar amb el botó dret i pressionar F9:



Com mostra la imatge anterior al clonar un TC es pot seleccionar el destí de la còpia, tant a nivell de projecte o de *TestSuit*.

Una vegada estiguen tots els TC preparats i amb els valors corresponents ja es disposa del *script* de proves per a poder provar la funcionalitat de la petició.

S'han de pujar els *scripts* de proves funcionals al repositori de SVN.

Si s'han de preparar proves funcionals per a cicles posteriors al primer els passos que s'han de seguir són:

1. Actualitzar el *script* ETF amb els inserts necessaris per a l'execució del nou cicle, ja siguen modificacions o nous casos de prova inclosos en el PLP.
2. Clonar el projecte i canviar les variables de configuració necessàries.
3. Si hi haguera, crear els nous casos de prova indicats en el PLP.
4. Pujar els projectes i documents actualitzats al repositori de SVN.

4.5.2. Proves no funcionals

4.5.2.1. Proves de seguretat

A cada *WebService* se li associa un perfil de seguretat especificat en el DTAN depenent de la criticitat de les dades en que treballa i dels requeriments que es dessitge. Així doncs, es realitzen proves de comprovació del perfil de seguretat per comprovar que es tenen els permisos corresponents al perfil. Els perfils de seguretat s'han definit en el *framework* pel client.

El *framework* actualment va per la versió 1.5 i incorpora mecanismes de seguretat en els *WebService* que cobreixen els aspectes d'identificació i autenticació, autorització, integritat, no repudi i confidencialitat.

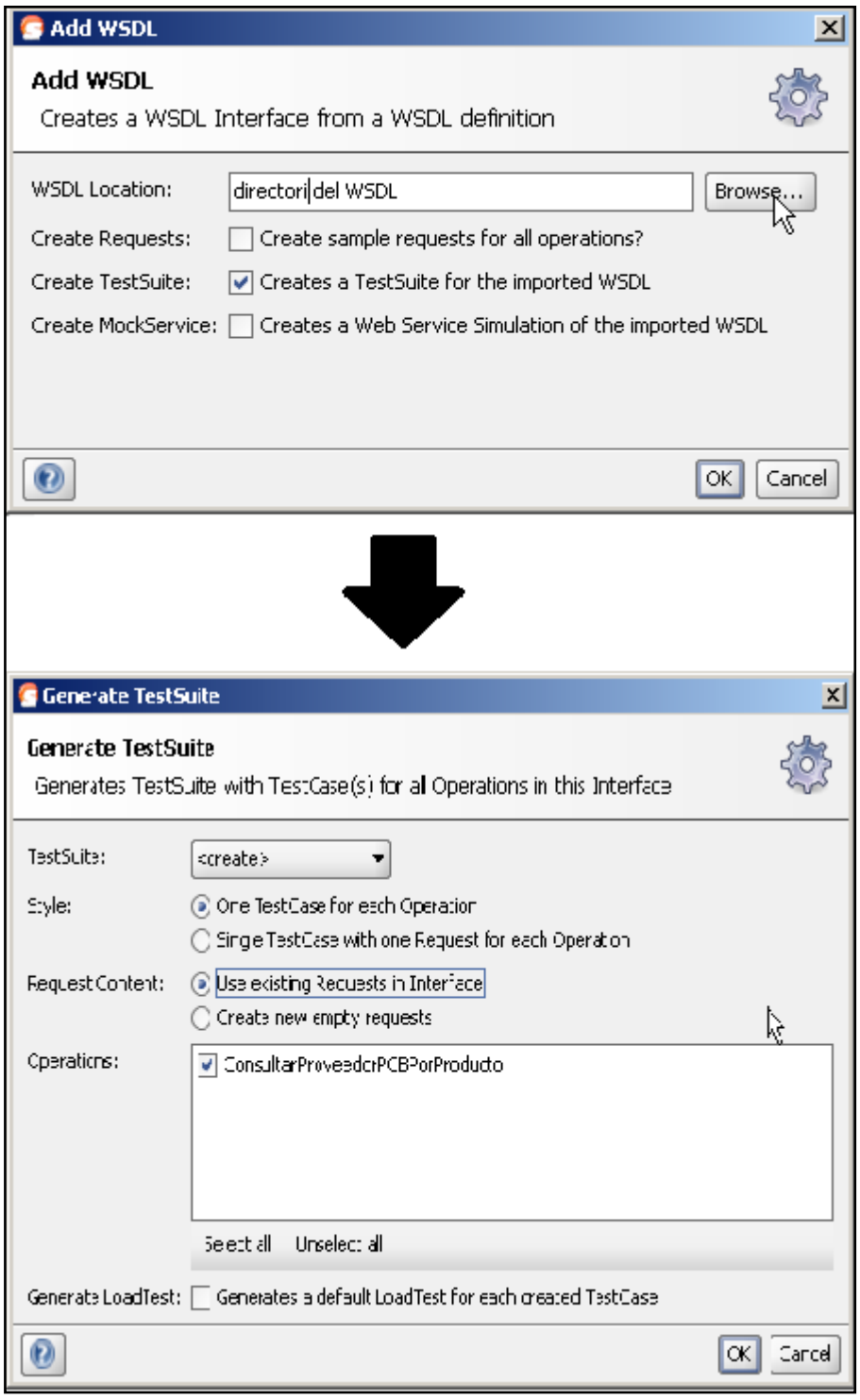
Al projecte *SoapUI* SCS es preparen les proves de vulnerabilitat conegudes dividides en dos rames (*STRING* & *ALL*):

Prova	**Descripció**	**Accions**	**Comprovacions**
Sql injection/path injection	Permet executar ordres de comandament sql per extreure informació de la BBDD	Afegir el sql personalitzat segons el DTAN	Assegurar que no es llança una resposta http 5xx i que no es recupere un *id_session*
Boundary scan	Permet obtindre informació del servidor a través dels valors anòmals en els paràmetres d'entrada.	Introduir valors fora de rang en els paràmetres d'entrada i recollir el comportament del *WebService*.	La resposta del servidor ve ben informada. No hi ha informació que mostre estructura interna del servidor, com la traça d'error.
Cross site scripting	Permet a un atacant introduir codi script a través d'una cridada al *WebService*, de manera que la aplicació web finalment execute aquest codi.	Afegir el script de client a la ferramenta.	Configuració estàndard de la ferramenta
Fuzzing scan	Es realitzen multitud de peticions on s'introdueixen valors per als	Configurar la potencia de l'atac amb el nombre de peticions i l'interval de temps.	No hi ha informació que mostra l'estructura interna del servidor, com una traça d'error i

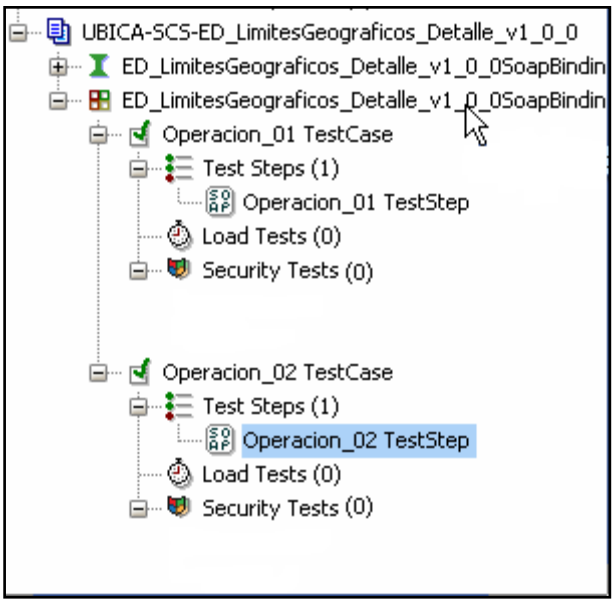
	camps d'entrada de forma aleatòria durant un temps suficientment llarg per provocar errors al servidor.		assegurar que no es llança una resposta HTTP 5xx.
Invalid types	Introdueix valors fora del tipus esperat per als paràmetres d'entrada i recollir el comportament del *WebService*	Configurar els tipus de dades i els valors que estan fora del format predefinit en el DTAN	La resposta del servidor està ben informada i no hi ha informació que mostre l'estructura interna.
Malformet XML	Enviament de XML mal format amb objecte de descobrir una vulnerabilitat en el servidor	Configurar els elements XML formats incorrectament a propòsit	La resposta del servidor està ben informada i no es mostra cap informació interna del servidor
Xml bomb	missatge enviat al servidor amb una estructura capaç de provocar una sobrecàrrega en l'analitzador de XML estructures que requereixen gran capacitat de memòria RAM i poden dur a la denegació del servei	El script de *SoapUI* ja és representatiu	El *WebService* segueix donant servei després de l'atac, la resposta del servidor està ben informada i no es mostra informació interna del servidor, com una traça d'error.

La primera tasca a realitzar per executar les proves de seguretat de la petició es situar-se en directori corresponent de SVN per modificar el nom de la plantilla pel corresponent de la petició. Una vegada reanomenat ja procedim a obrir-lo amb la ferramenta *SoapUI*.

El primer pas a realitzar quan ja hem obert el projecte amb el *SoapUI* és afegir-li el WSDL (amb l'opció "*Add*" WSDL a nivell de projecte) corresponent de la petició amb la següent configuració:

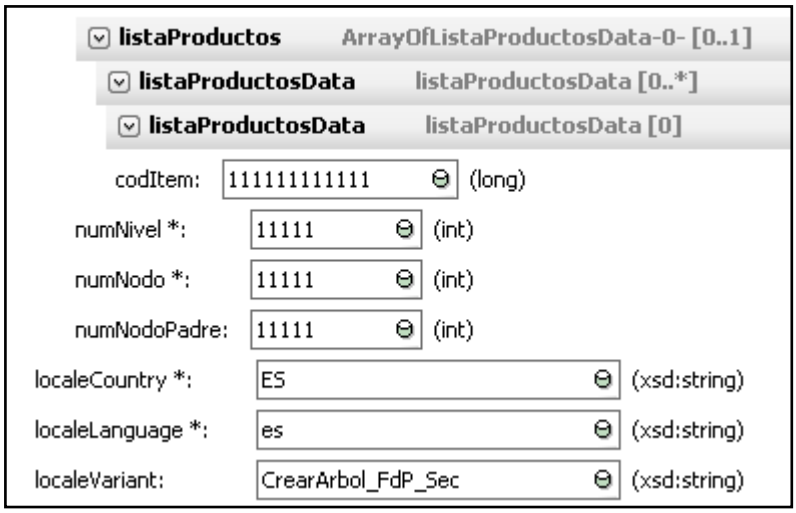


A l'afegir el WSDL amb la configuració anterior es crea un *TestSuit* amb totes les operacions del *WebService* i per cada operació es crea també un *TestCase* amb el *TestRequest* corresponent. L'estructura que presenta el projecte SCS és la que es mostra en la següent imatge:



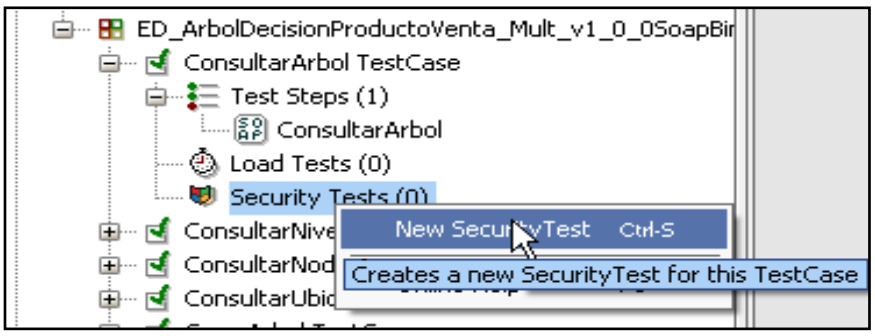
El següent pas és posar dades en els corresponents *TestRequest* de cada operació. La política que es segueix en FdP es la de posar uns en els paràmetres d'entrada numèrics. Depenent del tipus de paràmetre es posarà un nombre determinats d'uns (11111 per als enters, 111111111111 per als *flotants*...). Els camps que el seu tipus siga de *String* s'introduirà "NombreOperació_FdP_Sec", menys els camps de localització que són sempre "Es" i "es" per a aquest tipus de proves. Els camps *booleans* sempre a "*true*" i els de dates sol seleccionar-se el mateix dia que es prepara la prova.

El que volem aconseguir amb aquests valors es que quan es llancen les proves si es miren els *logs* siguen fàcil d'identificar com a proves de seguretat. La següent imatge mostra un exemple dels valors a introduir en cadascun dels paràmetres:

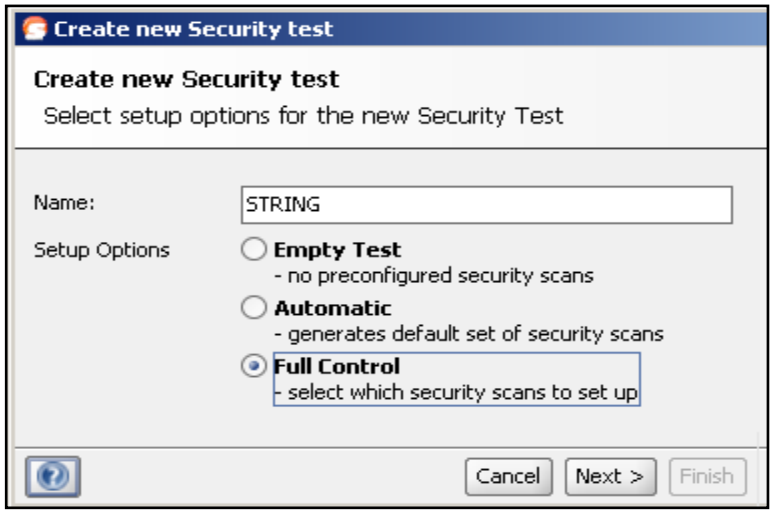


Ara s'han de crear els *Test Security* corresponents a cada operació. Es creen dos tipus de *Test Security*, un que inclou proves per als paràmetres de tipus *String* i un altre que inclou proves per a tots els tipus de paràmetres.

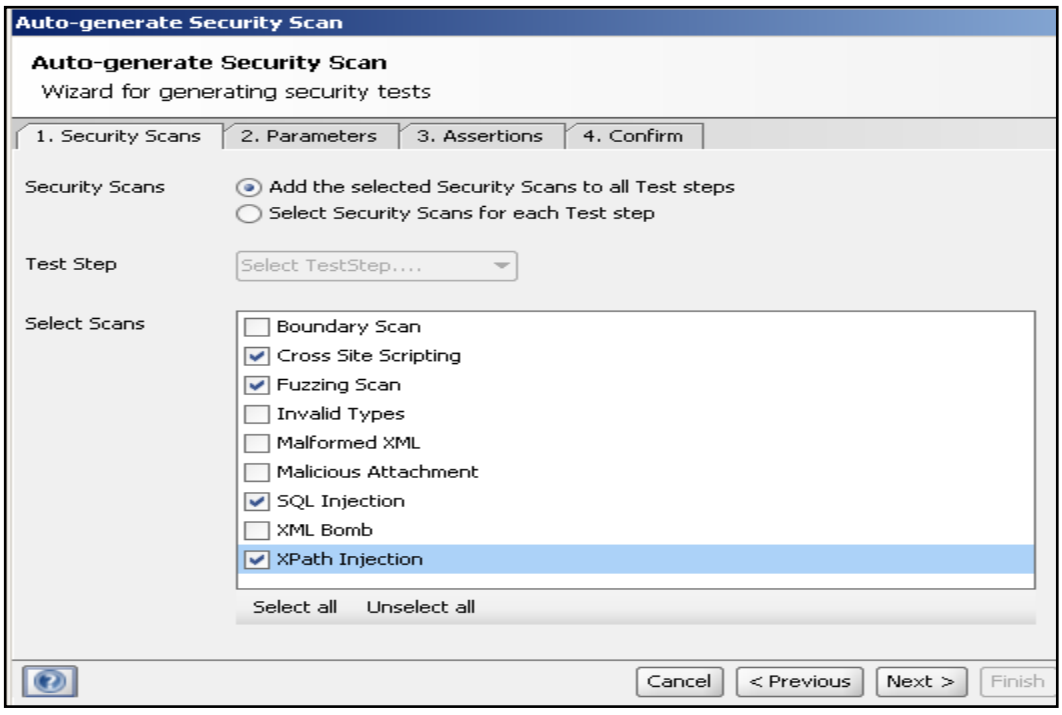
Per crear el Test de *String* polsem el botó dret del ratolí sobre "*Security Test*" i seleccionem l'opció "*New Security Test*" com mostra la imatge:



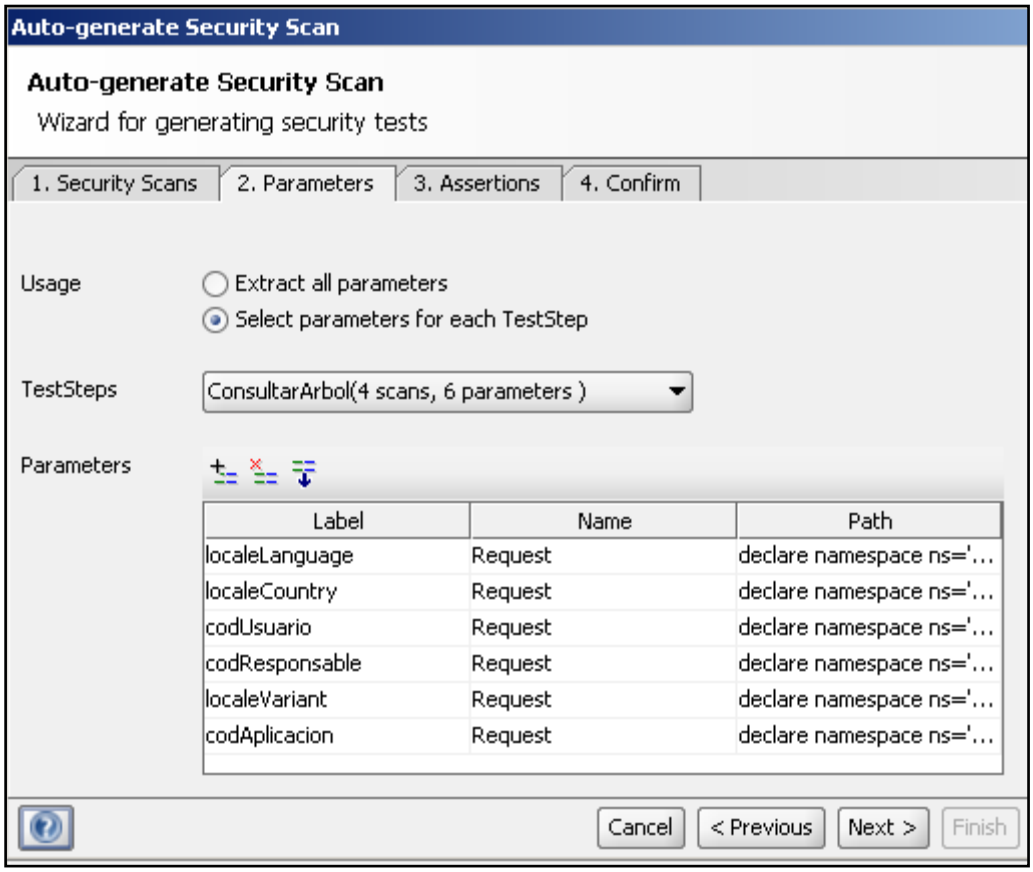
En la següent pantalla s'introdueix en nom del *Test Security*, en aquest cas el nom deu ser "*String*" ja que les proves d'aquest Test seran per als paràmetres de tipus *String*. Marquem l'opció "*Full Control*" i polsem "*Next*".



Seguidament es presenta la finestra de selecció de les proves. En aquest cas de prova cal seleccionar les proves "*Cross Site Scripting*", "*Fuzzing Scan*", "*SQL Injection*" i "*Xpath Injection*" com en la imatge i anar a la següent finestra:

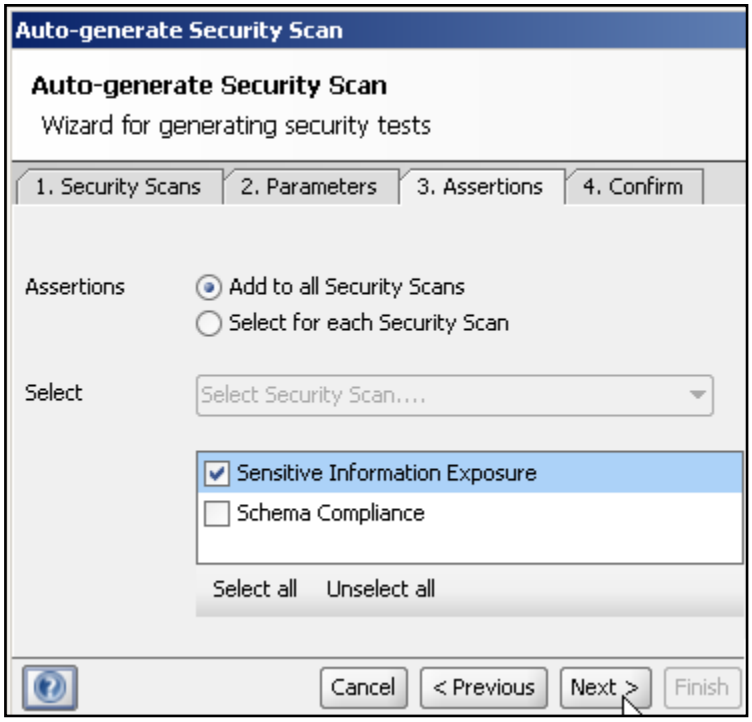


La següent pantalla de configuració és per seleccionar els paràmetres a tindre en compte durant la prova. Del primer apartat, "*Usage*", es selecciona la segon opció. De la part "*TestSteps*" marquem l'operació en que ens trobem. Una vegada seleccionada l'operació premem el botó per carregar els paràmetres corresponents a l'operació ,marcat en roig en la imatge, i d'aquesta llista llevem tots els paràmetres que no són de tipus *String*.

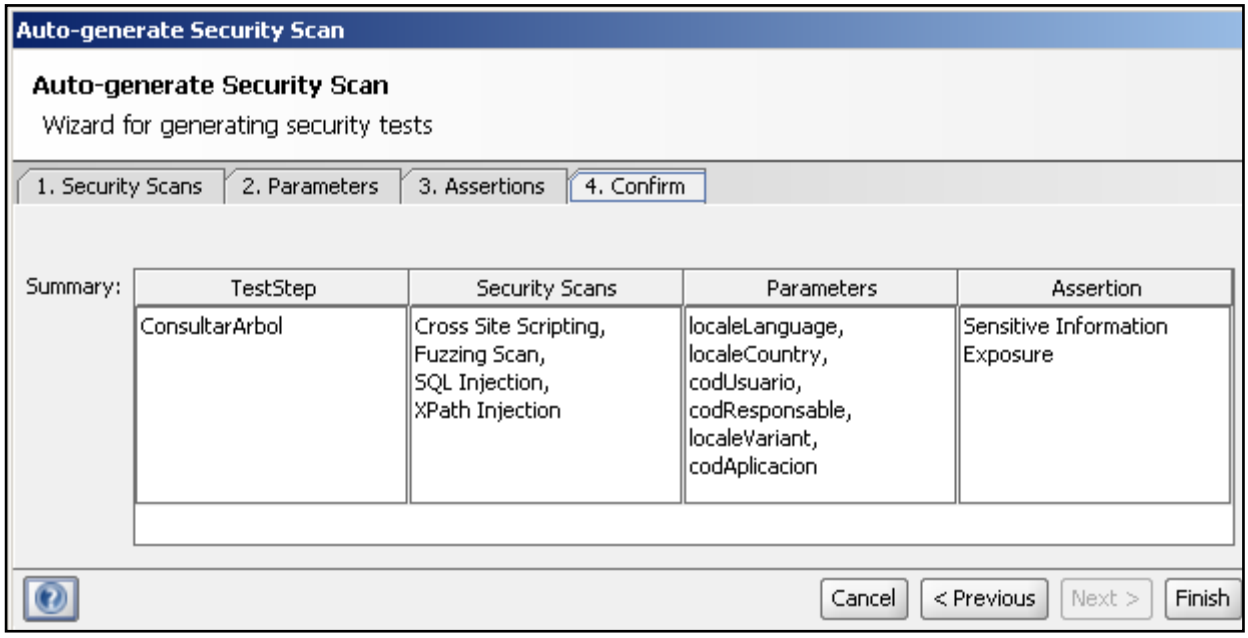


En aquest punt ens em adonat que existeix una mena de *bug* en el programa *SopaUI*. Si es pressiona alguna de les columnes de la taula de paràmetres per ordenar-los, a l'hora d'eliminar algun d'ells no s'elimina el seleccionat sinó el que anteriorment a l'ordenament ocupava eixa posició.

Al prémer "*Next*" es va a la següent pantalla, on només s'ha de seleccionar l'opció que es mostra en la imatge ("*Sensitive Information Exposure*"):

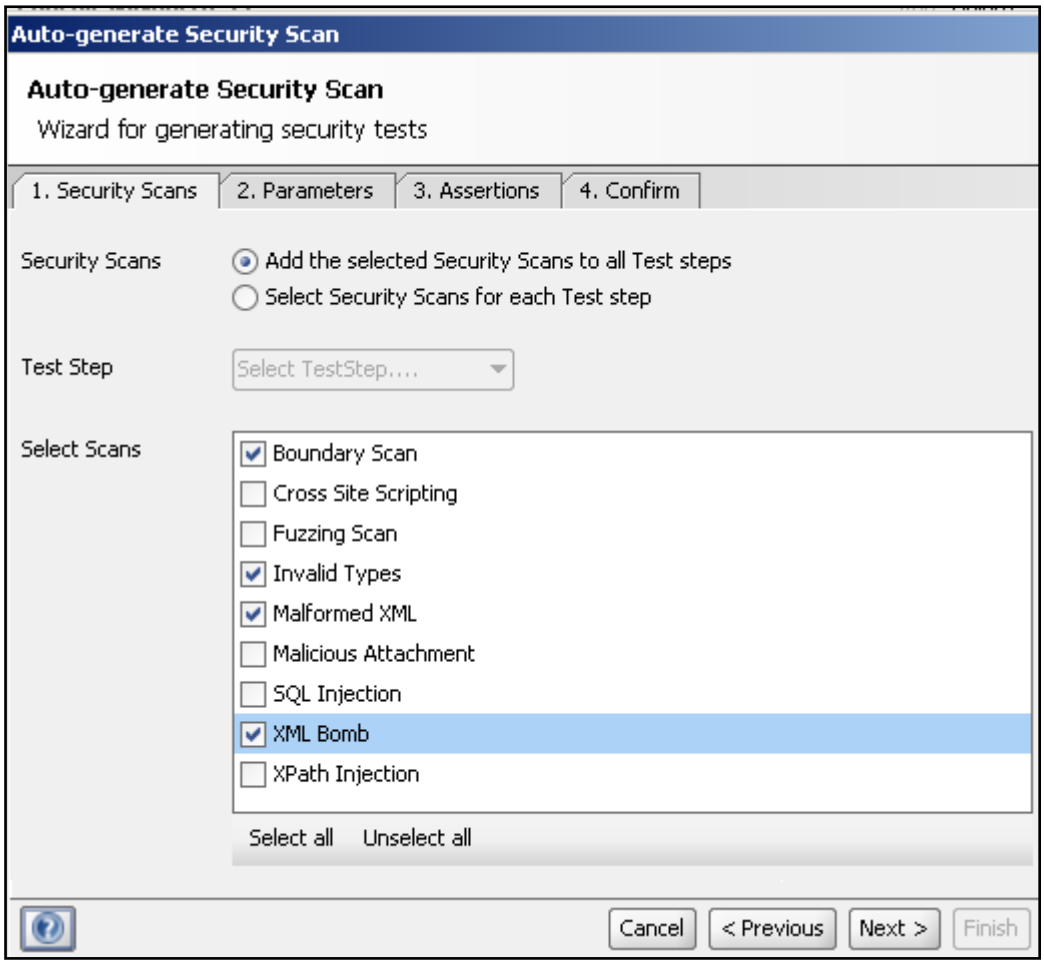


L'última finestra de configuració de la prova és un resum de totes les opcions seleccionades anteriorment:



Fins ací la configuració del *Test Security* dels paràmetres de tipus *String*. Ara queda preparar el *Test Security* que inclou tots els tipus de paràmetres, anomenat "*ALL*". La configuració és similar a l'anterior. Es crea un nou *Test Security* on la primera finestra de configuració és igual que l'anterior però en aquest cas el nom és "*ALL*".

A l'hora de seleccionar les proves que s'executaran en aquest *Test* és diferent a l'anterior preparació. En aquest cas les proves són:



En la següent finestra de configuració, que correspon a la selecció dels paràmetres, fem igual que l'anterior preparació quant als *String*, però en aquest cas no eliminem cap paràmetre. Només lleven un camp que el posa automàticament el *SoapUI*, "*AuthPath*".

La resta de la preparació és igual que l'anterior.

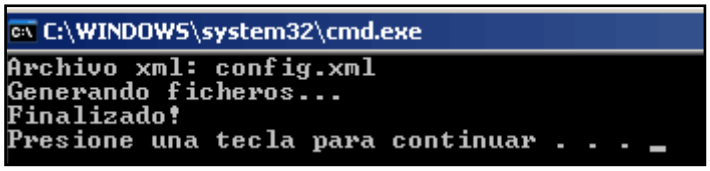
4.5.2.2. Proves de Rendiment

Les proves de rendiment consisteixen en assegurar que el comportament del *WebService* funciona d'acord amb els paràmetres establerts pel client. El propòsit és determinar els llindars operatius i obtindre informació sobre determinats factors com els temps de resposta, taxes de transferència, disponibilitat i utilització de recursos.

La primera part de la preparació de les proves de rendiment es fa mitjançant la ferramenta desenvolupada a nivell intern per FdP, *DARGenerator*. Aquesta ferramenta utilitza un fitxer anomenat "*config.xml*" per a generar automàticament les dades necessàries per a dur a terme les proves. Es prepara per operacions amb ajuda del DAT, mirant d'aquest joc de dades quines són les taules i com estan estructurades per a informar el fitxer XML i aquest generar-les. S'utilitza el DSC, però si hi ha llistes s'utilitza el TC corresponent del DAT. El "*config.xml*" presenta la següent estructura:

```
<schema>

    <name>adm_maestras</name>


    <table>
        <name>D_TIPO_LOCALIZACION</name>
        <columns>
            <column>
                <name>COD_N_TIPO_LOCALIZACION</name>
                <type>Number</type>
                <init>35732</init>
                <incremental>true</incremental>
                <frequency>1</frequency>
                <minValue>1</minValue>
                <maxValue>99999</maxValue>
            </column>
            <column>
                <name>TXT_NOMBRE</name>
                <type>varchar2</type>
                <init>ST3570 TIPO_LOCALIZACION OP_03</init>
                <incremental>false</incremental>
                <frequency>1</frequency>
                <minValue>1</minValue>
                <maxValue>1</maxValue>
            </column>
        </columns>
        <nrows>500</nrows>
        <commitFrequency></commitFrequency>
    </table>
```

L'exemple presenta una taula, però s'haurà de repetir aquesta estructura per cadascuna de les taules que s'utilitza en el DAT o DSC. Una vegada s'han llistat totes les taules i les seues propietats aquest fitxer es compila amb el *DARGenerator*:

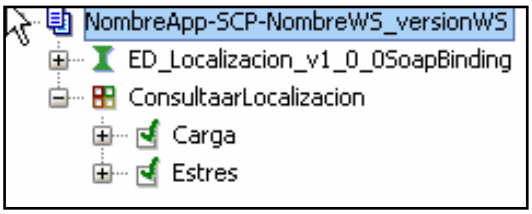


També disposa d'ajuda alhora de compilar si hi ha defectes en el fitxer "*config.xml*":

Hi ha que tindre en compte acomplir amb les volumetries de les llistes d'entrada i d'eixida especificades en el DTAN alhora d'informar els volums de les llistes en el fitxer *config.xml*. Les taules que no venen amb volumetria especificada en el DTAN però s'utilitzen per a modelar la funcionalitat del *WebService* estan especificades en un document anomenat "Model de base de dades". Aquest document explica totes característiques de les taules existents per a una aplicació donada. Hi ha que tindre compte en aquestes volumetries alhora de preparar les dades.

Prepara els *scripts* de prova de rendiment amb el *SoapUI*:

El primer pas igual que la preparació dels projectes de proves anterior és seguir la nomenclatura pròpia de factoria. En aquest tipus de proves les sigles son SCP. Des de *SoapUI* hi ha que carregar el WSDL pertinent a la petició i crear un *TestSuit* per cada operació amb el nom corresponent. El següent pas es crear un dos *TestCase* dins de cada TS, un per a càrrega i un altre per a estrés:

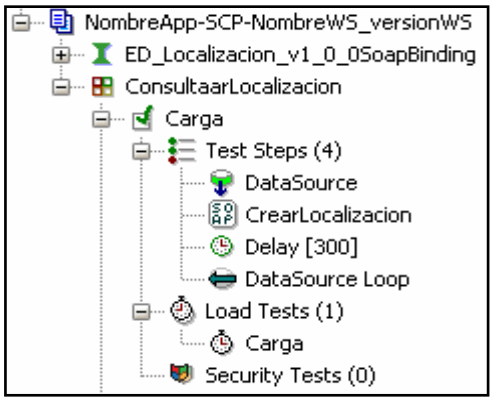


La preparació de càrrega i estrés són pràcticament iguals però hi ha uns xicotets canvis en alguna configuració. Es va a mostrar com es prepara una prova de càrrega, després es clona i es canvia el nom per estrés substituint els paràmetres de configuració que són distints.

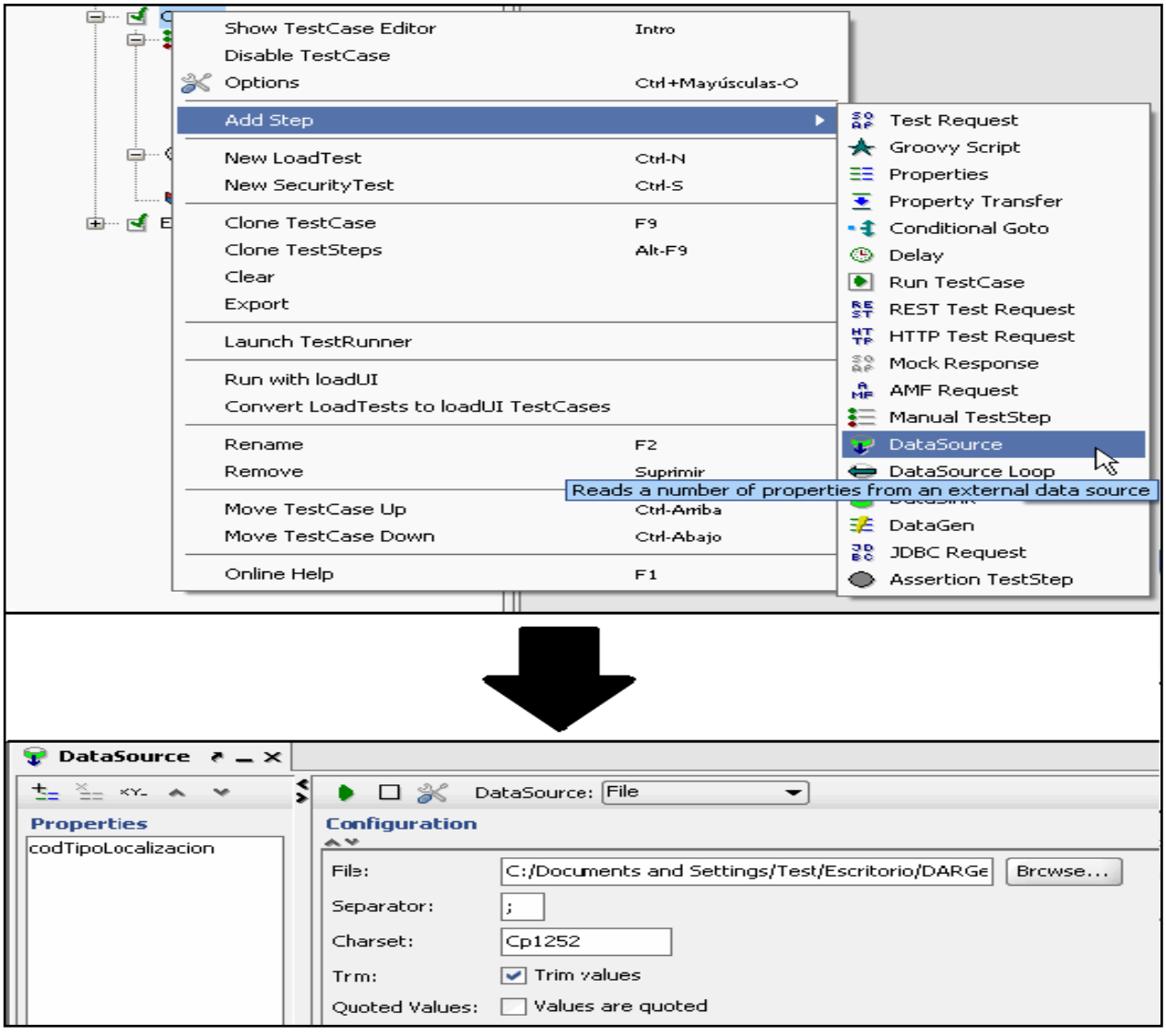
La prova de càrrega s'utilitza per a modelar una càrrega el més pareguda possible a la real i medir anticipadament el rendiment que proporcionarà un sistema en

l'entorn de producció. Permet identificar els temps de resposta que obtindrà una petició típica front a condicions normals.

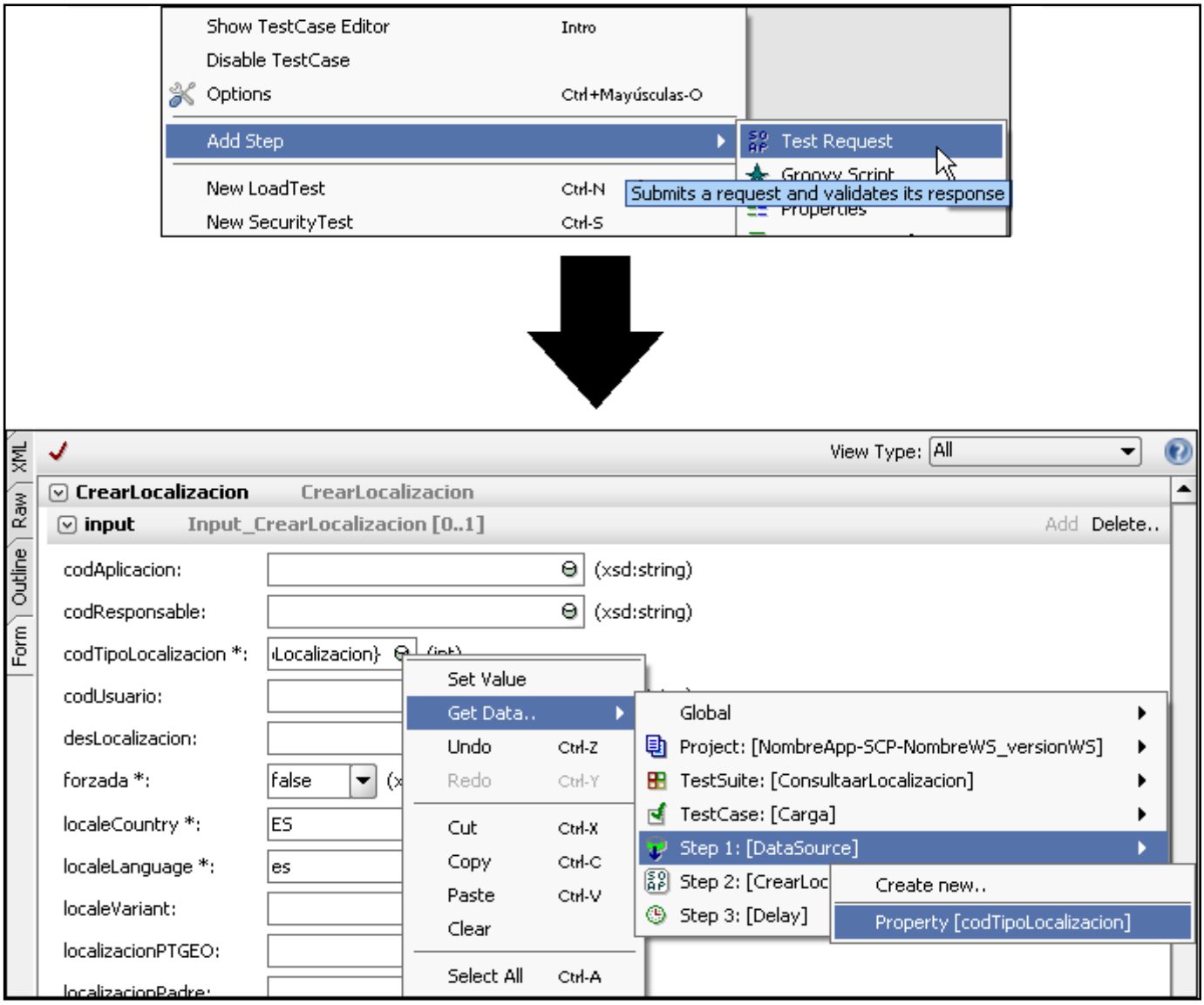
Hi ha cinc *Steps* que deu tindre un TC de càrrega: *DataSource*, *TestRequest*, *Delay* i *DataSource Loop* de *Test Steps* i un *Load Tests* anomenat "Carga". Tal com mostra la imatge:



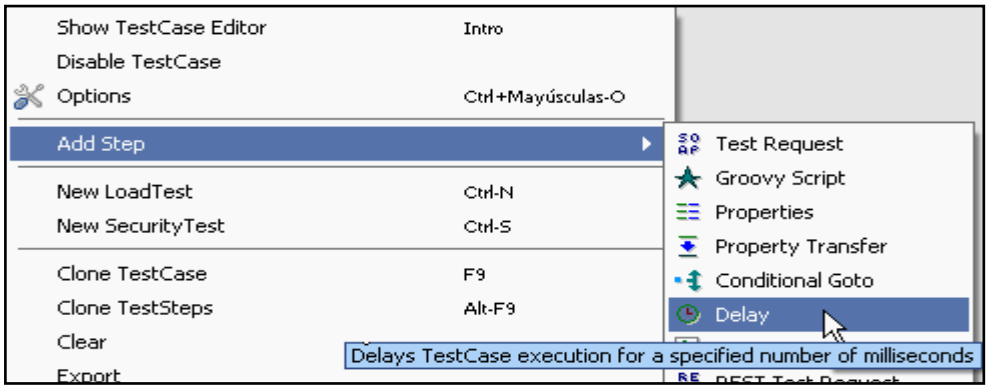
Anem a veure com es prepara cadascun dels Test *Steps* pas a pas. El *Step DataSource* s'utilitza per a agafar com a paràmetres d'entrada els fitxers .csv generats amb anterioritat pel *DARGenerator*. Es poden afegir tants paràmetres com es vulga i configurar cadascun d'ells del fitxer d'on llegir:



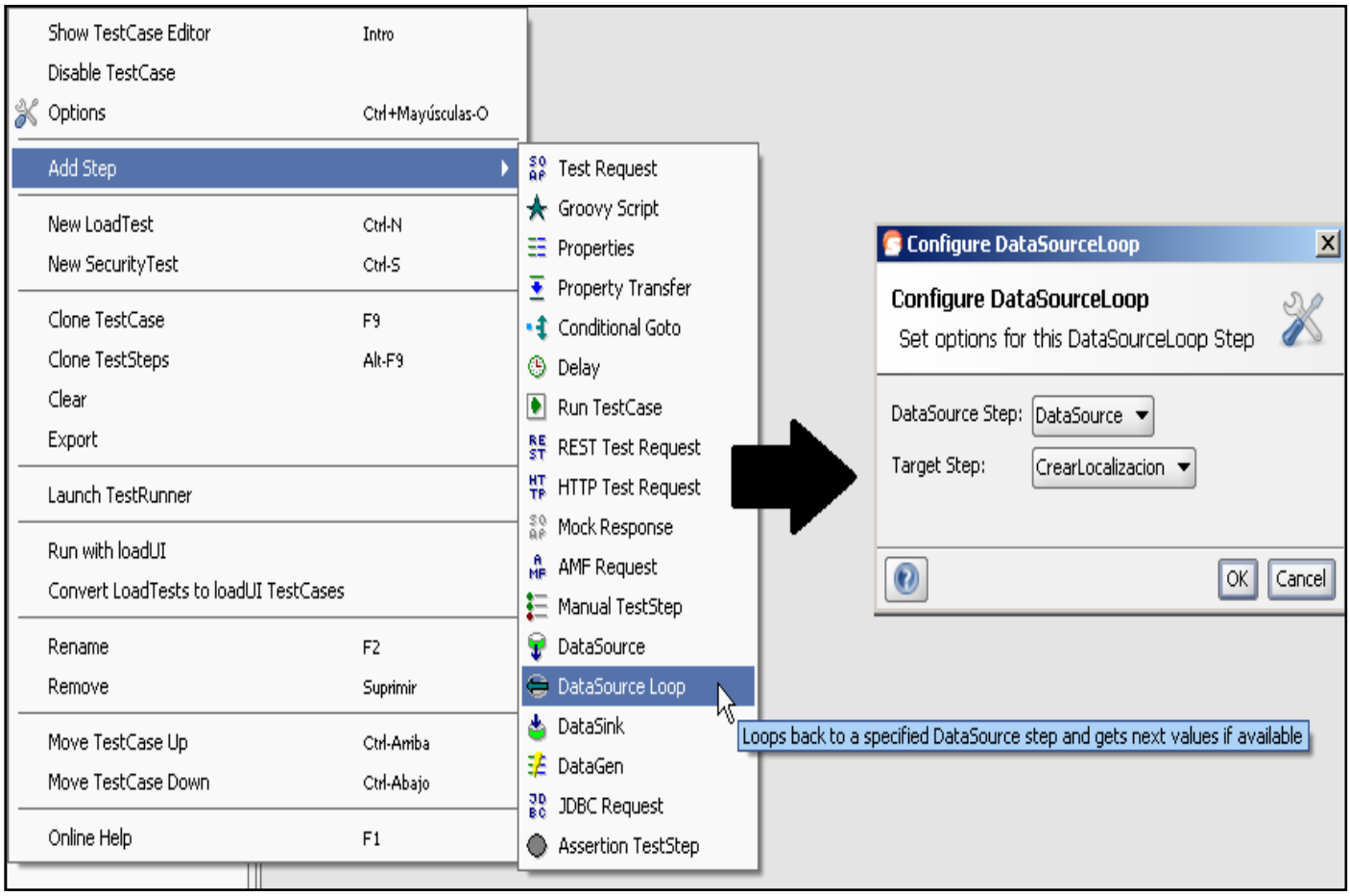
Aquest *Test Request*, a diferencia dels preparats en les proves funcionals, no utilitza per a tots els paràmetres un JDBC per extreure els valors, sinó que s'informen els valors directament en cadascun dels camps i en els paràmetres configurats en el *DataSource* s'informen per a que s'agafen les dades d'aquest *Step*:



El següent *Step* a crear és el "*Delay*" on hi ha que informar-lo amb valor de 300. Aquest indica el temps en mil·lisegons que tarda el bucle en tornar a llançar una petició, en aquest cas 300ms entre iteració.

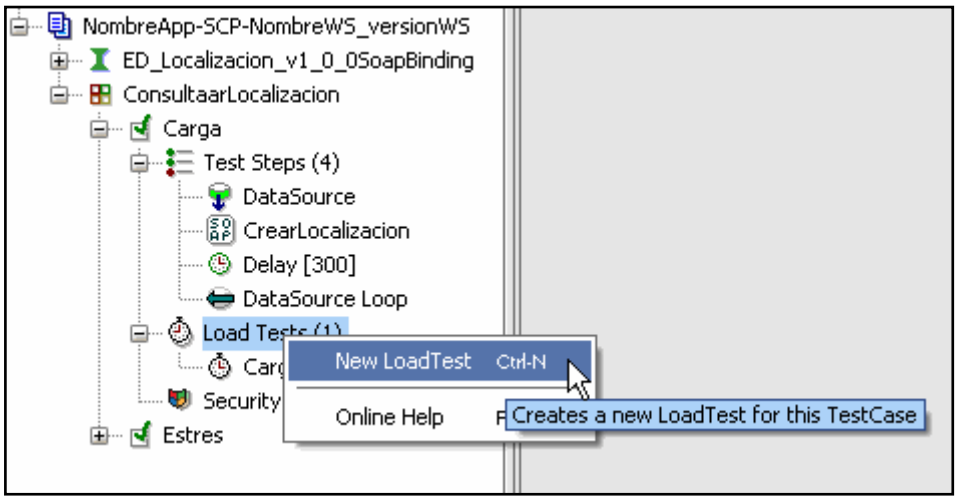


L'últim a crear és el "*DataSource Loop*" que configura les iteracions del bucle:

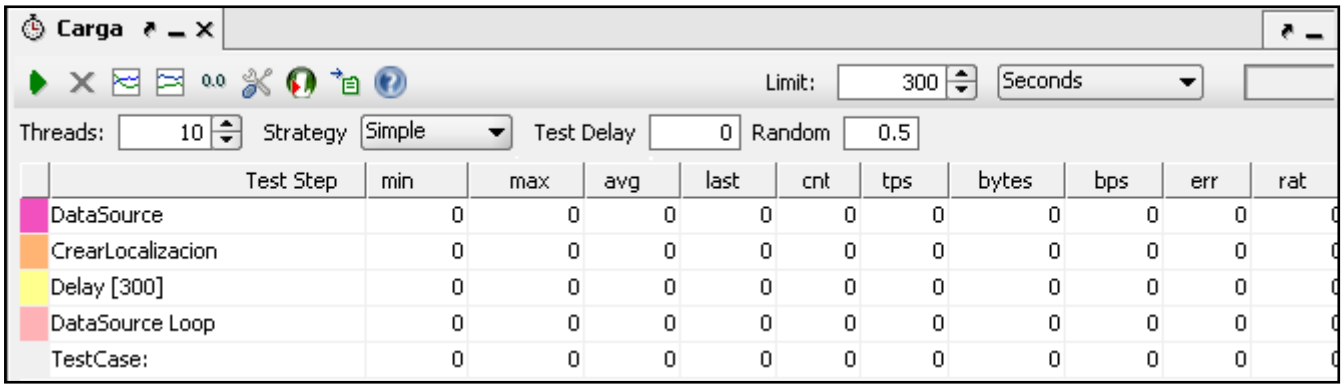


En el bucle hi ha dos paràmetres a especificar: "*DataSource Step*" (de quin *Step* s'agafen els valors) i "Target *Step*" (quin *Step* s'itera)

Falta crear el "*Load Tests*":



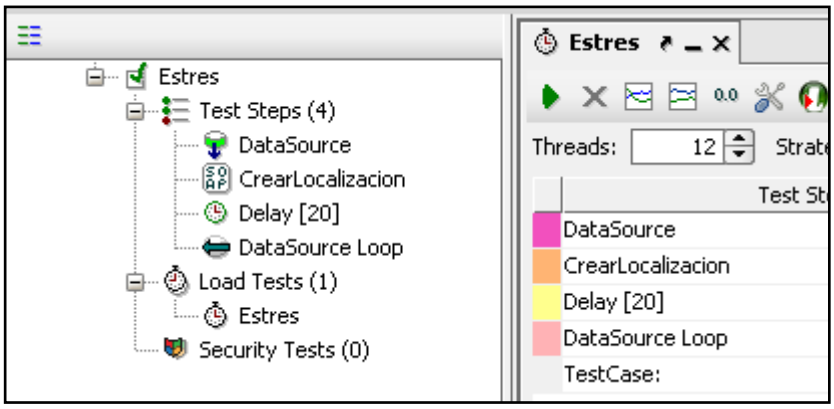
D'aquest *Step* hi ha que configurar el nombre de "*Threads*", determinar el "*Test Delay*" a 0 i el "*Limit*" que és el temps que estarà la prova en execució, que per política s'ha definit a 300 segons. La configuració queda de la següent manera:



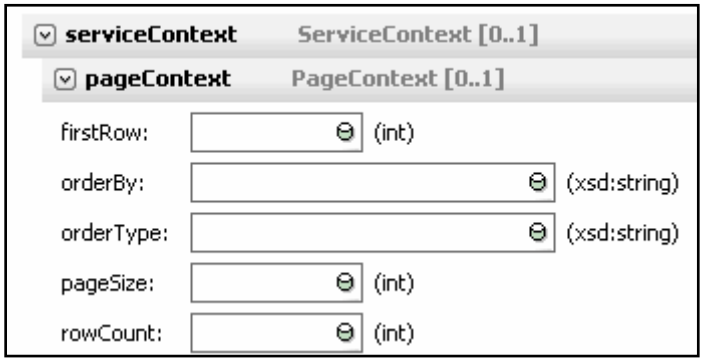
El nombre de *threads* a utilitzar ve definit en el DTAN. En aquest document ve en l'apartat connexions al *pool*. Si es tracta d'una operació de consulta o modificació el nombre de *threads* serà igual al nombre de connexions al *pool*, en canvi, si es una operació de creació o d'eliminació el nombre de *threads* serà la meitat del nombre de connexions al *pool*.

Ara hi ha que preparar les proves d'estrés. Aquestes proves estan dissenyades per verificar que el funcionament del sistema front a un llindar límit de recursos, amb el màxim nombre de peticions previstes. Aquestes proves es realitzen per determinar la solidesa de l'aplicació en els moments de càrrega extrema i ajudar a determinar si l'aplicació rendirà suficientment en cas de que la càrrega real supere l'esperada.

Hi ha que clonar el TC de "Carga" i es reanomenar-lo com a "Estrés". Els únics canvis a realitzar són el canvi del "Delay" a 20ms, canviar el nom del "Load Tests" a "Estrés" i el nombre de *threads* d'aquest últim a un 20% més que en càrrega. Un exemple dels canvis aplicats:



Alhora de preparar el *SoapUI* d'una prova de rendiment és molt important tindre en compte la configurar la paginació i ordenació especificats en el DTAN. Si l'operació es requereix paginació, es deu utilitzar els camps *firstRow* (registre a partir del qual va a mostrar) i *pageSize* (nombre de registres per pàgina) per a gestionar-la. Si requereix ordenació sútilitzen els camps *orderBy* (nombre del camp pel qual va a ordenar-se) i *orderType* (ASC o DESC).



Una vegada estiguen tots els fitxers preparats hi ha que arxivar-los en el repositori corresponent dins la petició de les proves de rendiment de SVN.

4.6.Execució de proves

La tercera setmana del cicle de vida de la petició correspon a l'execució de les proves. En aquesta setmana FdD entrega el codi a factoria de proves, a partir de l'entrega es disposa de tres dies de proves. En aquest moment s'executen les proves per ordre de prioritat: proves funcionals de *sanity check*, proves funcionals de versió, proves no funcionals de seguretat, proves estàtiques de qualitat de codi i verificació de caixa blanca i proves no funcionals de rendiment.

En l'execució de proves cada vegada que es troba d'una incongruència s'anomena defecte.

4.6.1. Proves funcionals

Els projectes implicats en l'execució de proves funcionals són el SCC i el SCV. Ambdós segueixen els mateixos passos en l'execució de les proves. A continuació s'expliquen els passos generals abans de començar l'execució pròpiament dita i l'execució d'un TC representatiu de tots els altres.

Per a passar les proves funcionals el cap d'equip és l'encarregat de demanar al departament de sistemes que desplegue la petició en els entorns locals. Aquesta demanda es fa mitjançant correu electrònic i es posa en còpia al tester encarregat en provar-la.

D'aqueta manera només està desplegat arriba un correu amb la informació per provar la petició amb la següent informació:

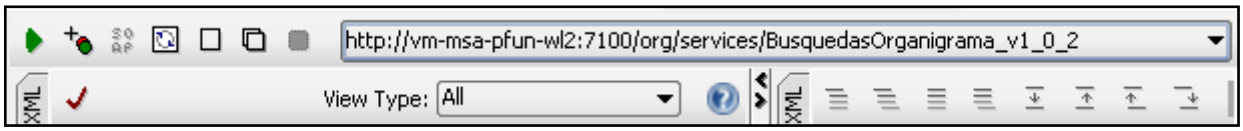
Desplegado:
http://vm-pfun-wl8:7100/produ/services
http://vm-pfun-wl8:7100/produ/.version_app.txt
Saludos,

El primer enllaç és la direcció on està desplegada la petició i el segon la *baseline* que s'ha utilitzat al desplegar.

Dels enllaços que anteriorment s'han rebut via correu electrònic s'ha de copiar la direcció del WSDL i per a cadascun dels projectes funcionals:

ED_TipoImpuesto_v1_0_0: • ConsultarTipoImpuesto	**Endpoint address:** http://vm-pfun-wl2:7100/produ/services/ED_TipoImpuesto_v1_0_0 **Target namespace:** - **Security profile:** K **Use attachments:** - **Base64 attachments:** false **WSDL:** http://vm-pfun-wl2:7100/produ/services/ED_TipoImpuesto_v1_0_0?wsdl **BusinessOperation:** http://vm-pfun-wl2:7100/produ/services/ED_TipoImpuesto_v1_0_0?operation

Amb aquesta direcció ens situem sobre el WSDL i fem un "*Refactor definition*" del preparat amb el WSDL desplegat en l'entorn de proves. Automàticament la ferramenta ens demana confirmació per a guardar *l'endpoint* associat al WSDL. Aquest *endpoint* es posa automàticament en cada TC del projecte. Es comprova que *l'endpoint* del *SoapUI* és el mateix que acompanya a la direcció del WSDL que hem copiat anteriorment:



Fins ací, la configuració a nivell de projecte per executar les proves funcionals. Els següents passos són els que s'han de realitzar per a executar un TC:

El primer pas a realitzar és carregar el fitxer ETF en BBDD per a que el *SoapUI* tinga valors en els paràmetres d'entrada. Aquest fitxer és un ".sql" i, per tant, s'utilitza la ferramenta gràfica *SQL Developer* per llançar-lo. Aquest fitxer es llança en tres parts. Primer es borra la taula per assegurar que els valors a introduir són els actuals:

```
--------------------------
-- Borrado de datos
--------------------------

DELETE FROM Taula_ETF_petició_Operació;
COMMIT;


--------------------------
-- Borrado de tabla
--------------------------

DROP TABLE Taula_ETF_petició_Operació;
COMMIT;
```

Després es crea la taula:

```
--------------------------
-- Creación de la tabla
--------------------------

CREATE TABLE Taula_ETF_petició_Operació (
  TC_ID VARCHAR2(200 BYTE) NOT NULL ENABLE,
  CICLO NUMBER NOT NULL ENABLE,
  LOCALELANGUAGE VARCHAR2(2000 BYTE),
  LOCALECOUNTRY VARCHAR2(2000 BYTE),
  LOCALEVARIANT VARCHAR2(2000 BYTE),
  CODAPLICACION VARCHAR2(2000 BYTE),
  CODUSUARIO VARCHAR2(2000 BYTE),
  CODRESPONSABLE VARCHAR2(2000 BYTE),
  CODITEM VARCHAR2(2000 BYTE),
  FECHAVALORDESDE VARCHAR2(2000 BYTE),
  FECHAVALORHASTA VARCHAR2(2000 BYTE),
  CODTIPOTERCERO VARCHAR2(2000 BYTE),
  SOLOVIGENTES VARCHAR2(2000 BYTE),
  LC_CODCENTRONEGOCIO VARCHAR2(2000 BYTE),
  ESPERADA VARCHAR2(200 BYTE) DEFAULT '[ver PLP]'
);
COMMIT;
```

Després es carreguen tots els registres d'entrada:

```
-- 0P_01-BaseEntrada-01
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-001', 1, 'es', 'ES', null, null, null, null, '970002732101', null, null, null, null, null, default);

-- 0P_01-BaseEntrada-02
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-002', 1, 'es', 'ES', null, null, null, null, '970002732102', null, null, null, null, '820002732102'||

-- 0P_01-BaseEntrada-03
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-003', 1, 'es', 'ES', null, null, null, null, '970002732103', null, null, null, null, '820002732103',

-- 0P_01-BaseEntrada-04
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-004', 1, 'en', 'UK', 'vv', 'SURVE', 'FdP', 'MSA', '970002732104', '2014-03-24T00:00:00', '2015-05-24T

-- 0P_01-BaseEntrada-05
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-005', 1, 'es', 'ES', null, null, null, null, '970002732105', null, null, null, null, null, default);

-- 0P_01-BaseEntrada-ER
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-017', 1, 'es', 'ES', null, 'aaaaaaaaaaaaaaaaaaaaa', 'aaaaaaaaaaaaaaaaaaaaa', 'aaaaaaaaaaaaaaaaaaaaa',
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-018', 1, 'es', 'ES', null, null, null, null, '970002732199', null, null, null, null, null, default);
INSERT INTO Taula_ETF_petició_Operació VALUES ('0P_01-F-019', 1, 'es', 'ES', null, null, null, null, '970002732199', null, null, null, null, null, default);
COMMIT;
```

Una vegada seguits els passos descrits ja es disposa dels valors d'entrada corresponents per a cada TC a l'hora d'executar la prova des de *SoapUI*.

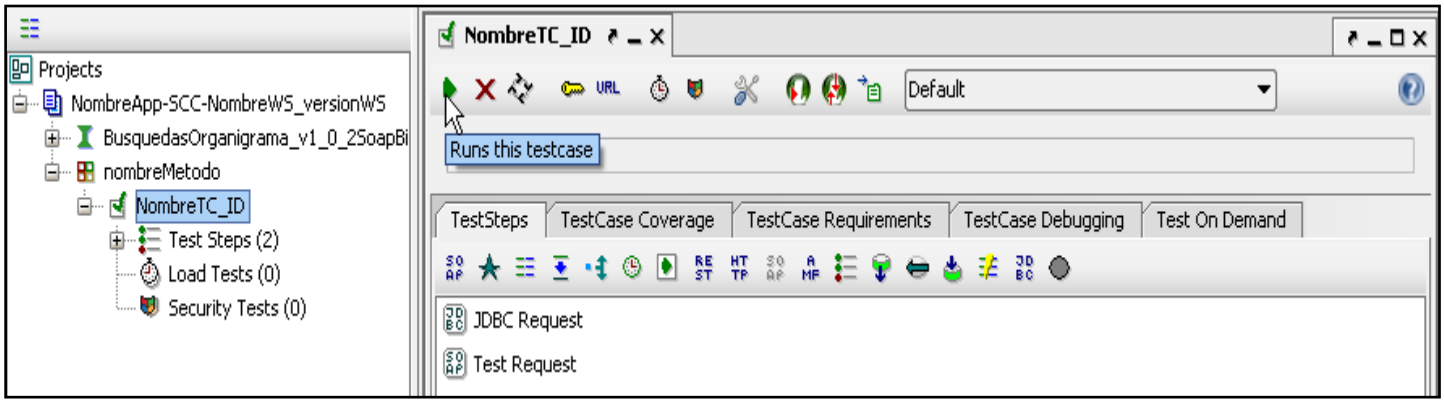
El següent a executar és el fitxer DMA. Aquest fitxer és un ".sql", per tant, es llança des del *SQL Developer*. Es fa de forma completa i aprofita per a eliminar qualsevol registre en la BBDD sobre la que anem a executar les proves. Aquest pas és important per assegurar-nos que les proves van a realitzar-se de forma independent quant als registres de bases de dades. En la següent imatge es mostra una xicoteta part del fitxer:

```
--OP-01-F-005
DELETE FROM S_ESTADO_SURTIDO_TIENDA WHERE COD_N_TIPO_ESTADO_SURTIDO = 42721 AND COD_N_CENTRO_NEGOCIO = 820002721105 AND C
COMMIT;
DELETE FROM S_SURTIDO_TIENDA WHERE COD_N_CENTRO_NEGOCIO=820002721105 AND COD_N_ITEM=970002721105 AND FEC_D_INICIO='23/04/
COMMIT;
DELETE FROM D_ITEM WHERE COD_N_ITEM=970002721105 AND COD_N_TIPO_ITEM=22721;
COMMIT;
DELETE FROM D_TIENDA WHERE COD_N_CENTRO_NEGOCIO=820002721105;
COMMIT;
DELETE FROM D_CENTRO_NEGOCIO WHERE COD_N_CENTRO_NEGOCIO=820002721105 AND COD_N_TIPO_TERCERO=72721 AND COD_N_PUBLICO=2721;
COMMIT;
DELETE FROM D_ORGANIZACION WHERE COD_N_TERCERO=820002721105 AND COD_N_TIPO_TERCERO=72721;
COMMIT;
DELETE FROM D_TERCERO WHERE COD_N_TERCERO=820002721105 AND COD_N_TIPO_TERCERO=72721;
COMMIT;


--OP-01-F-006
DELETE FROM S_ESTADO_SURTIDO_TIENDA WHERE COD_N_TIPO_ESTADO_SURTIDO = 42721 AND COD_N_CENTRO_NEGOCIO = 820002721106 AND C
COMMIT;
DELETE FROM S_SURTIDO_TIENDA WHERE COD_N_CENTRO_NEGOCIO=820002721106 AND COD_N_ITEM=970002721106 AND FEC_D_INICIO='23/04/
COMMIT;
DELETE FROM D_ITEM WHERE COD_N_ITEM=970002721106 AND COD_N_TIPO_ITEM=22721;
COMMIT;
DELETE FROM D_TIENDA WHERE COD_N_CENTRO_NEGOCIO=820002721106;
COMMIT;
DELETE FROM D_CENTRO_NEGOCIO WHERE COD_N_CENTRO_NEGOCIO=820002721106 AND COD_N_TIPO_TERCERO=72721 AND COD_N_PUBLICO=2721;
COMMIT;
DELETE FROM D_ORGANIZACION WHERE COD_N_TERCERO=820002721106 AND COD_N_TIPO_TERCERO=72721;
COMMIT;
DELETE FROM D_TERCERO WHERE COD_N_TERCERO=820002721106 AND COD_N_TIPO_TERCERO=72721;
COMMIT;
```

A continuació es carreguen els registres que corresponen al TC a executar. A l'hora de carregar-los es pot fer des de dos fitxers, el DSC en cas de que siga el TC corresponent al SCC, o des del DAT si és qualsevol altre TC. El següent exemple és sobre un DSC:

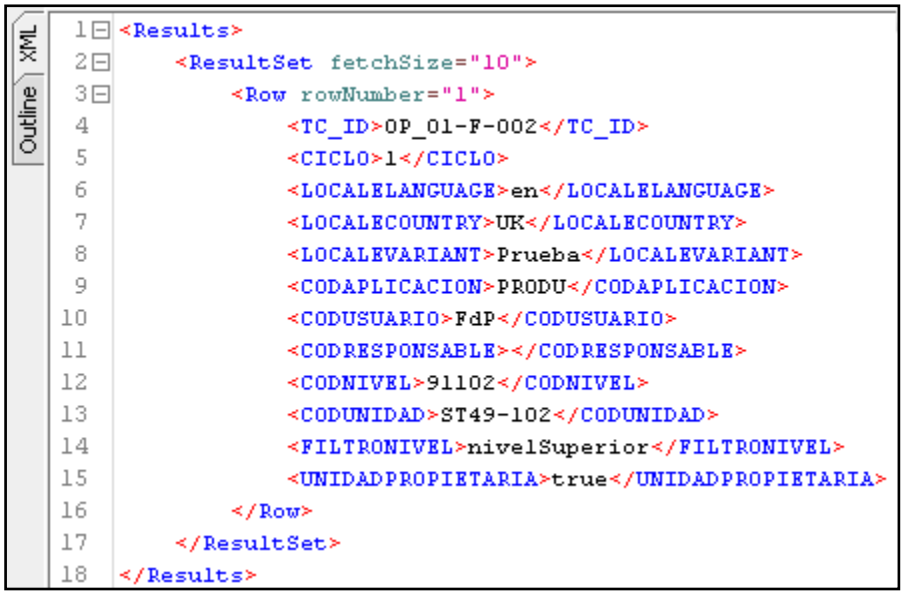
```
-- Nom_Operació

--OP-01-F-001
INSERT INTO D_TIPO_ESTADO_SURTIDO(COD_N_TIPO_ESTADO_SURTIDO, TXT_NOMBRE)
VALUES(42721, 'NombreTipoEstadoSurtido OP-01-F-SC');
COMMIT;

INSERT INTO D_ESTADO_SURTIDO (COD_N_TIPO_ESTADO_SURTIDO, COD_N_ESTADO_SURTIDO, TXT_NOMBRE)
VALUES(42721, 52721, 'Nombre EstadoSurtido OP-01-F-SC');
COMMIT;


INSERT INTO D_TIPO_TERCERO (COD_N_TIPO_TERCERO, TXT_NOMBRE)
VALUES(72721, 'NombreTipoTercero OP-01-F-SC');
COMMIT;

INSERT INTO D_TERCERO(COD_N_TERCERO, COD_N_TIPO_TERCERO)
VALUES(820002721101, 72721);
COMMIT;

INSERT INTO D_TIPO_ORGANIZACION (COD_N_TIPO_TERCERO)
VALUES(72721);
COMMIT;

INSERT INTO D_ORGANIZACION (COD_N_TERCERO, COD_N_TIPO_TERCERO)
VALUES(820002721101, 72721);
COMMIT;

INSERT INTO D_TIPO_CENTRO_NEGOCIO (COD_N_TIPO_TERCERO)
VALUES(72721);
COMMIT;

INSERT INTO D_CENTRO_NEGOCIO (COD_N_CENTRO_NEGOCIO, COD_N_TIPO_TERCERO, TXT_NOMBRE_CENTRO, COD_N_PUBLICO)
VALUES(820002721101, 72721, 'NombreCentro OP-01-F-SC', 2721);
COMMIT;

INSERT INTO D_TIENDA (COD_N_CENTRO_NEGOCIO)
VALUES(820002721101);
COMMIT;
```

Una vegada en aquest punt ja es disposa de totes les dades corresponents per a l'execució del TC. Així doncs, ja passem al TC en la ferramenta *SoapUI*. Seleccionem el TC a executar i l'obrim. Des de el botó de *play* executem el cas:



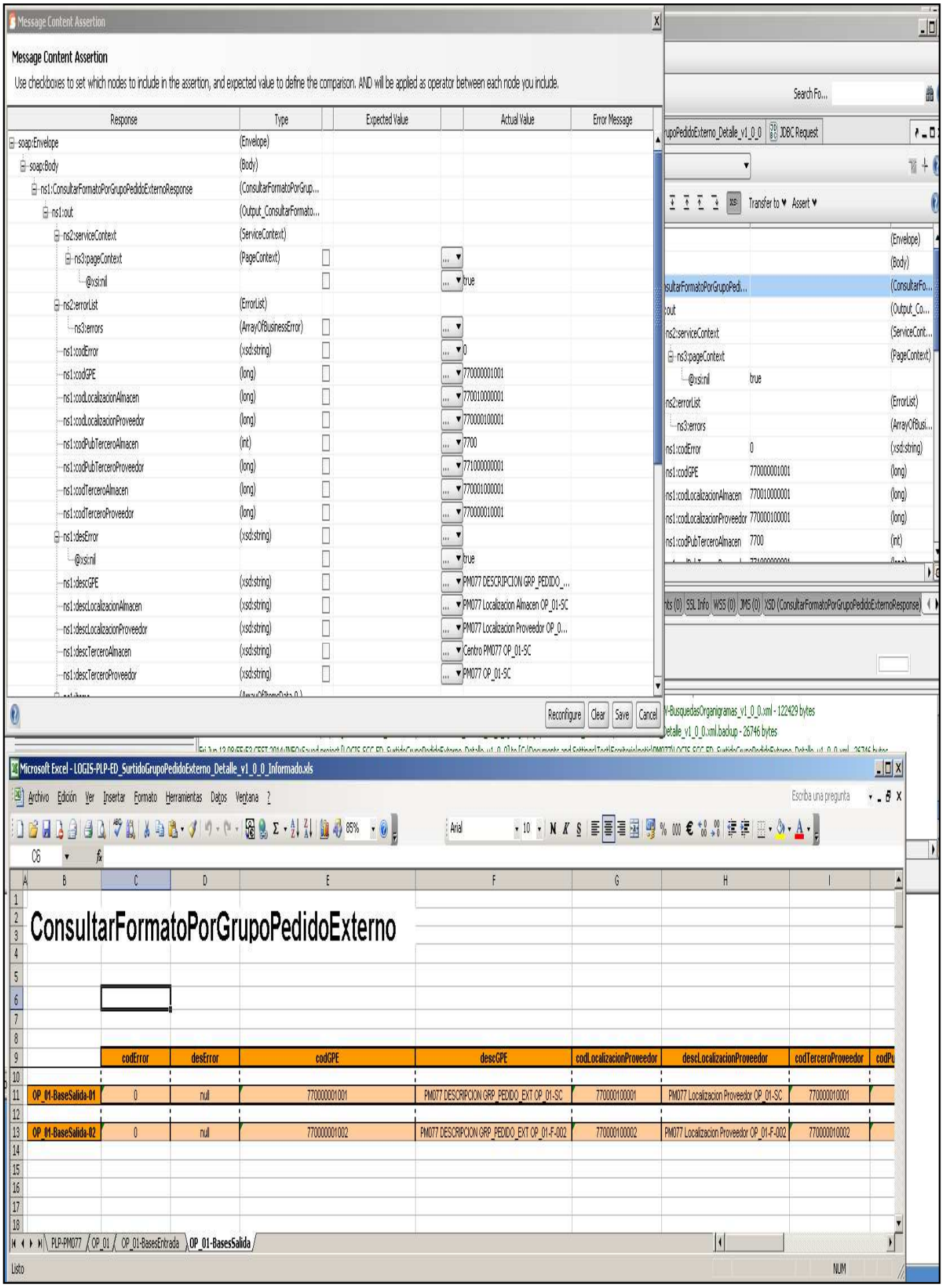
És important llançar el cas en aquest nivell i no un altre perquè la configuració dels directoris automàtics dels *logs* del *SoapUI* que hem dut a terme en passos anteriors només funciona en aquest nivell.

L'execució anterior comporta que tant el *JDBC Request* com el *Test Request* s'han executat. En el cas del primer es poden veure els resultats en la part dreta si obrim el JDBC.

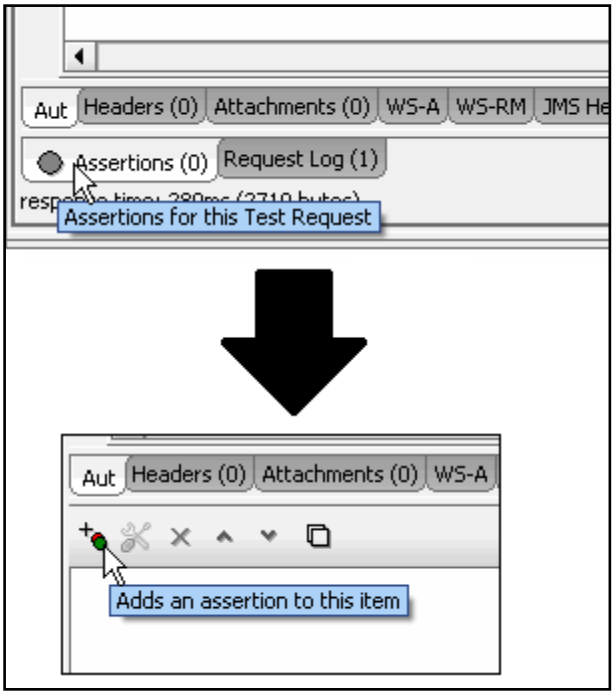


L'eixida del *WebService* es pot veure en el *Test Request*. Normalment els *WebService* tenen eixida però en alguns casos no és així. Quan no en tenen s'han de comprovar els resultats tornats del TC en els logs de les màquines, és a dir, en els logs dels entorns on està desplegada la petició. Aquests logs es miren a través de la ferramenta *Filezilla* i són importants a l'hora de veure pas a pas l'execució del *WebService*.

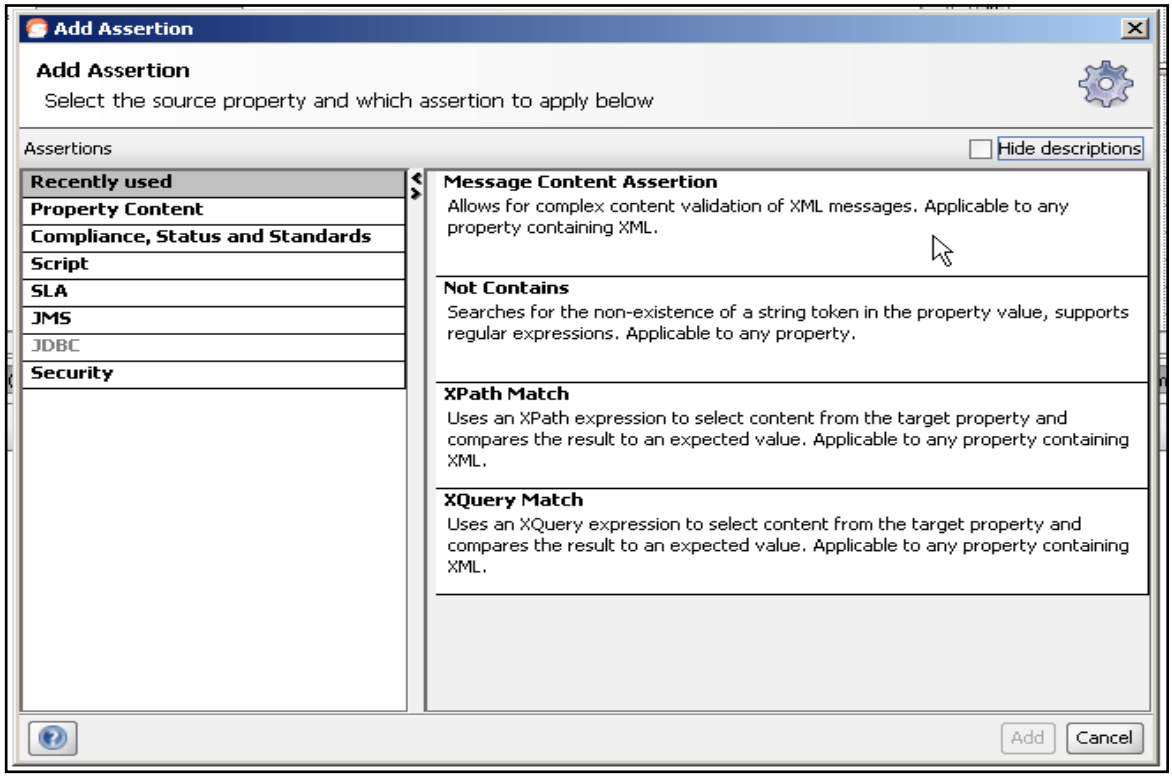
En obrir el *Test Request* veiem en la part dreta de la finestra l'eixida del TC. Aquests resultats deuen ser els mateixos que els esperats, les eixides esperades de la pestanya "salida esperada" del PLP. Es deu comprovar que són els mateixos.


Message Content Assertion
Message Content Assertion
Use checkboxes to set which nodes to include in the assertion, and expected value to define the comparison. AND will be applied as operator between each node you include.
Response
Type
Expected Value
Actual Value
Error Message
soap:Envelope
soap:Body
ns1:ConsultarFormatoPorGrupoPedidoExternoResponse
ns1:out
ns2:serviceContext
ns3:pageContext
@xsi:nil
ns2:errorList
ns3:errors
ns1:codError
ns1:codGPE
ns1:codLocalizacionAlmacen
ns1:codLocalizacionProveedor
ns1:codPubTerceroAlmacen
ns1:codPubTerceroProveedor
ns1:codTerceroAlmacen
ns1:codTerceroProveedor
ns1:desError
ns1:descGPE
ns1:descLocalizacionAlmacen
ns1:descLocalizacionProveedor
ns1:descTerceroAlmacen
ns1:descTerceroProveedor
Reconfigure
Clear
Save
Cancel
Microsoft Excel - LOGIS-PLP-ED_SurtidoGrupoPedidoExterno_Detalle_v1_0_0_Informado.xls
ConsultarFormatoPorGrupoPedidoExterno
codError
desError
codGPE
descGPE
codLocalizacionProveedor
descLocalizacionProveedor
codTerceroProveedor
OP_01-BaseSalida-01
OP_01-BaseSalida-02
PM077 DESCRIPCION GRP_PEDIDO_EXT OP_01-SC
PM077 DESCRIPCION GRP_PEDIDO_EXT OP_01-F-002
PM077 Localizacion Proveedor OP_01-SC
PM077 Localizacion Proveedor OP_01-F-002
PLP-PM077
OP_01
OP_01-BasesEntrada
OP_01-BasesSalida
Listo
NUM

Per facilitar cicles posteriors la ferramenta *SoapUI* disposa d'unes funcionalitats per a guardar les eixides en un "*assertion*". Per crear un "*assertion*" s'ha de polsar el botó "+". S'elegeix de la llista disponible el "Message Content Assertion" tal com es mostra en la següent imatge:



A continuació es comprova si el "Actual Value" és el valor esperat segons el PLP. Marcant el Check de "Expected Value", el propi SoapUI el rellena amb el valor actual, així en tots els valors que es volen guardar, tal com mostra la imatge:



Així queda guardat junt en el projecte *SoapUI* i en cicles posteriors només s'ha de veure que la bola que marca *l'assertion* està verda, si es així tot ha anat correctament. Ens estalviaríem comprovar un a un els paràmetres de l'eixida amb el PLP.

S'ha de tenir especial cura si el PLP ha canviat, ja que suposaria que *l'assertion* no és correcte i el resultat de les proves tampoc.

Tot aquest procés s'ha de repetir per a cadascun dels TC que conformen el PLP per a la petició.

Depenent de l'estat de la petició els TC a executar poden ser diferents. Si les proves funcionals a executar són les del cicle 1 s'executen tots els TC del PLP. Primer s'executa el projecte de SCC. Si hi ha algun KO en aquest punt saturen les proves, ja que els defectes trobats en SCC són defectes bloquejants.

Si a l'executar el SCC els resultats són els correctes passarem a executar els altres tipus de TC, els del projecte de proves funcionals de versió. S'executen per ordre, ja que estan creats per ordres de prioritat en el PLP.

Si les proves a realitzar són posteriors al cicle 1 els casos a executar poden variar depenent de la situació. Si per exemple, hi ha un nou cicle degut a un defecte trobat en un cicle anterior només es prova el SCC de totes les operacions més els TC corresponents als defectes. Un altre cas on no és necessari executar tots els TC de la petició és quan hi ha algun canvi en el DTAN. S'executa el SCC de totes les operacions per assegurar que els canvis en una operació en concret no afecten a la funcionalitat global del *WebService* i tots els TC de l'operació afectada.

L'objectiu d'aquestes proves una vegada ja s'ha realitzat el cicle 1 és la realització de les proves necessàries per assegurar que després d'una implementació d'un conjunt de canvis el software entregat segueix satisfent tots els requisits funcionals descrits en el DTAN, és a dir, que no s'han introduït errors sobre funcionalitats no modificades.

Al finalitzar tot el procés de proves s'ha d'informar dels resultats obtinguts en l'informe final de proves. Dins d'aquest document en la pestanya de "Funcionales" es completa amb els resultats funcionals. La següent imatge exemplifica un INP informat en l'apartat de "Funcionales":

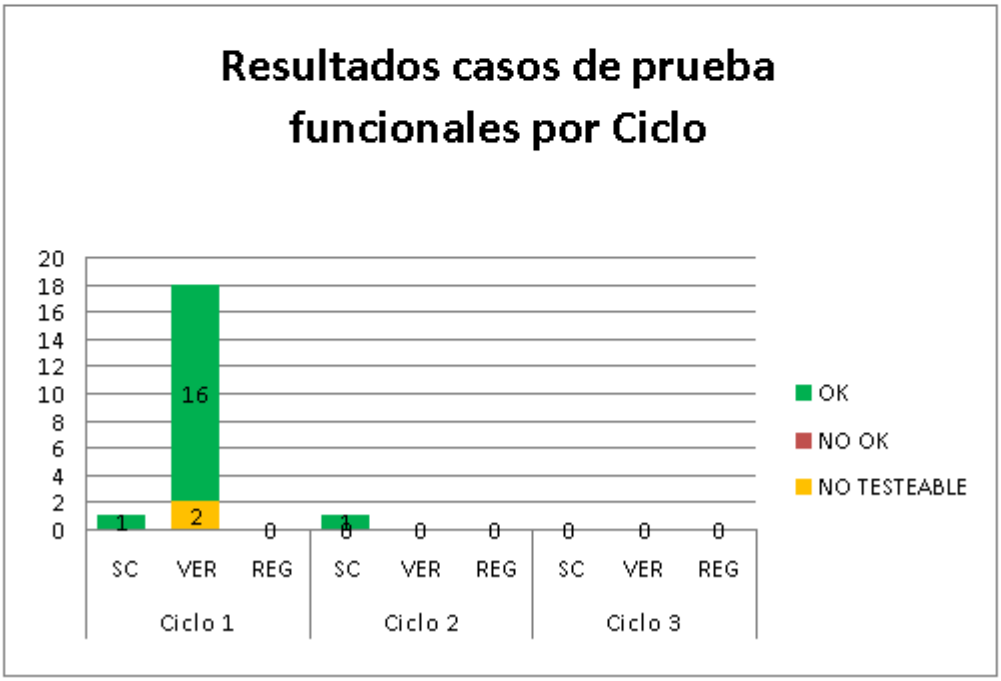
4.2. Resultado de las pruebas

		Ciclo 1			Ciclo 2			Ciclo 3		
		SC	*VER*	*REG*	*SC*	*VER*	*REG*	*SC*	*VER*	*REG*
Casos de prueba	*OK*	1	16	0	1	0	0	0	0	0
	NO OK	0	0	0	0	0	0	0	0	0
	NO TESTEABLE	0	2	0	0	0	0	0	0	0
Defectos reportados	*Bloqueante*	0	0	0	0	0	0	0	0	0
	Alta	0	0	0	0	0	0	0	0	0
	Media	0	0	0	0	0	0	0	0	0
	Baja	0	0	0	0	0	0	0	0	0

[Cuando no colorearla

Tabla 4.1 Resultados de las pruebas funcionales

En la gráfica siguiente se indican los resultados de los casos de prueba ejecutados en cada ciclo:



2. Resultados de las pruebas | 3. Pruebas estáticas | **4. Funcionales** | 5. No funcionales | Anexos | Info para métricas

També s'ha d'informar la *baseline* utilitzada durant el procés de proves. Així, un conjunt de resultats de proves va associat a una *baseline* determinada:

Resultados por ciclos de ejecución de pruebas:

			Ciclo 1	Ciclo 2	Ciclo 3
Etiquetado (BaseLine)			bl_surve_online_v0.10003.77	bl_surve_online_v0.10003.133	x.y.z
Fecha Inicio			20/05/2014	05/06/2014	DD/MM/AAAA
Fecha Fin			21/05/2014	05/06/2014	DD/MM/AAAA
Tipología pruebas ejecutadas	ST	ENT	O	No ejecutado	No ejecutado
		COD	P	No ejecutado	No ejecutado
		VCB	P	No ejecutado	No ejecutado
	FUN	SC	P	P	No ejecutado
		VER	P	No ejecutado	No ejecutado
		REG	No ejecutado	No ejecutado	No ejecutado
	NFUN	REN	P	No ejecutado	No ejecutado
		SEG	P	No ejecutado	No ejecutado
		LOC	No ejecutado	No ejecutado	No ejecutado
		COM	No ejecutado	No ejecutado	No ejecutado

Tabla 2.2 Resultados por ciclos de ejecución de pruebas

Aquesta *baseline* ve informada en el correu electrònic del desplegament i la podem saber obrint el segon enllaç del mateix.

Si ha sorgit algun defecte, es reportarà mitjançant la ferramenta CQFAC. L'encarregat d'informar els defectes és l'analista de la petició. Per a dur-ho a terme fan falta els *logs* del *SoapUI*, el joc de dades utilitzat per al cas i els paràmetres d'entrada. D'aquesta manera FdD pot reproduir el mateix error. De vegades es dóna el cas de que el mateix error es repeteix en més d'un TC. En aquestes situacions s'informa d'un únic error englobant als altres.

Una vegada tots els documents informats i actualitzats s'han de pujar al repositori de SVN per a que estiga disponible. Els documents a pujar són: logs de proves, projecte *SoapUI* de SCC i SCV amb els "*assertions*", PLP informat amb els resultats i l'INP.

4.6.2. Proves no funcionals

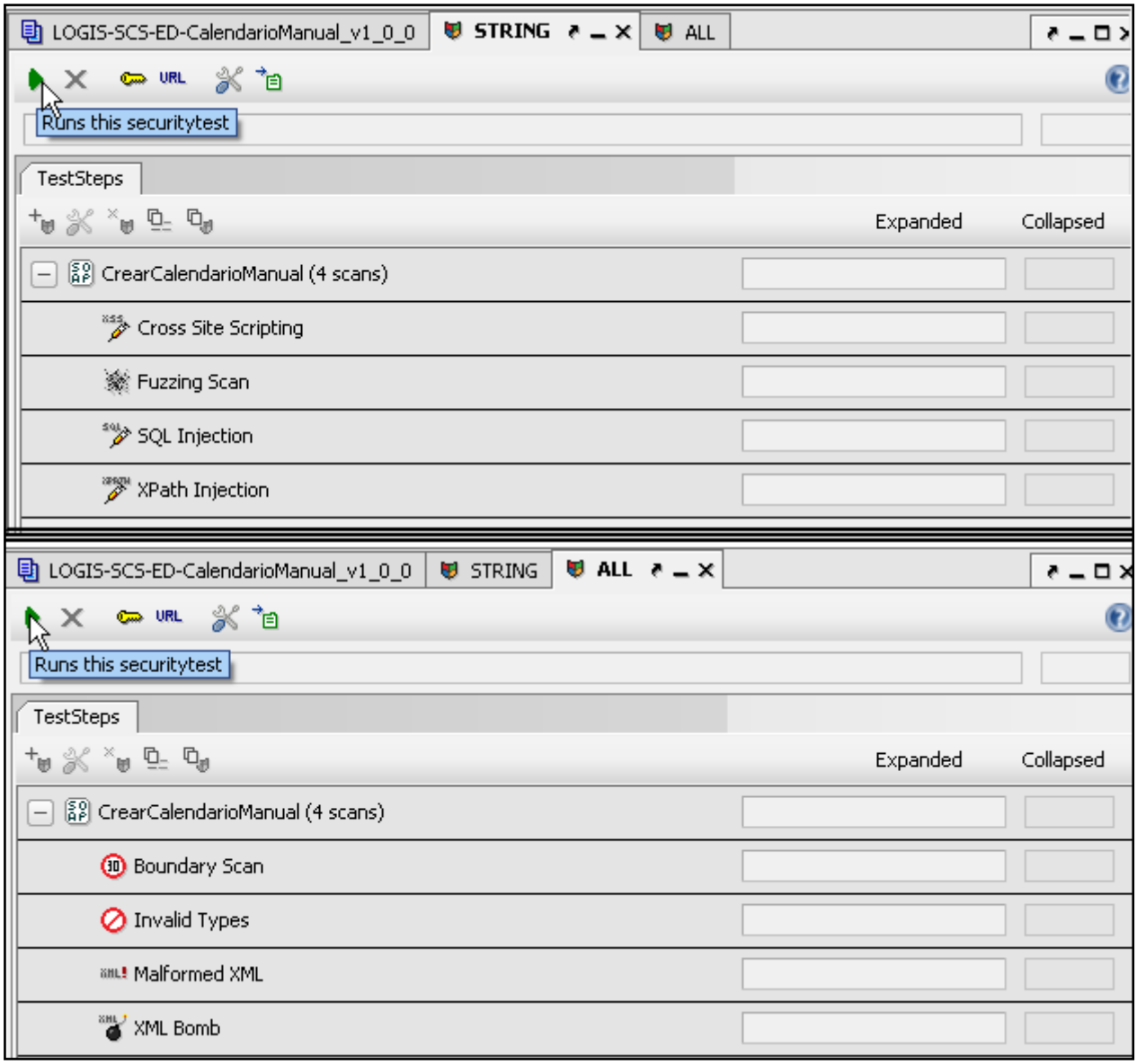
4.6.2.1.Proves de seguretat

Aquestes proves si han segut preparades correctament en els anteriors passos són molt ràpides d'executar. S'executen sobre el mateix entorn que les proves

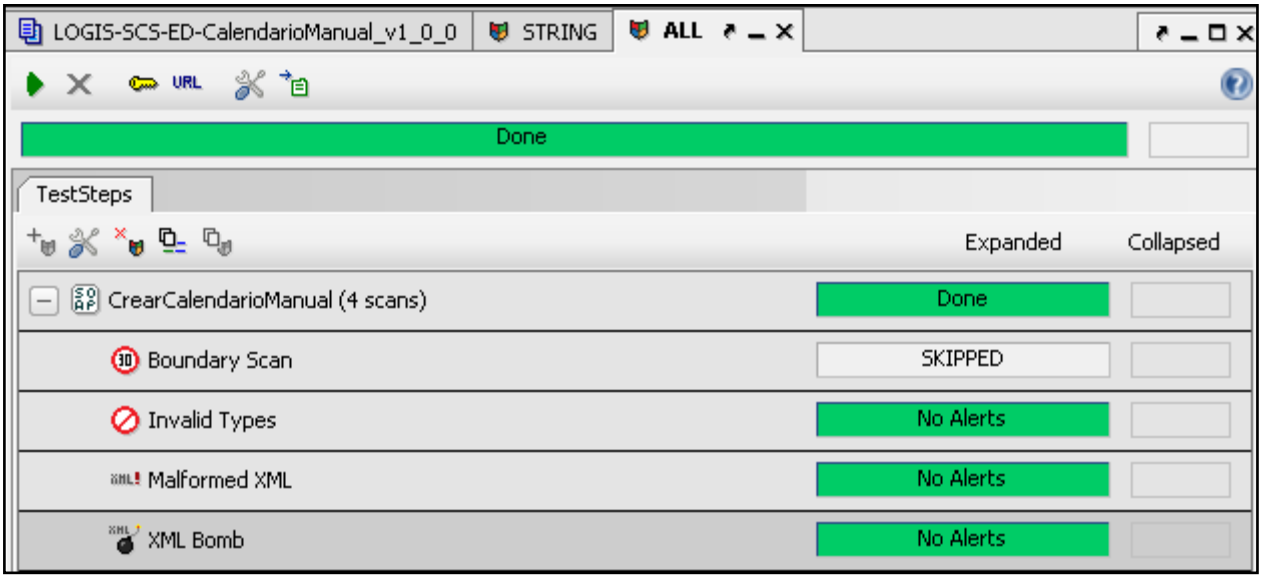
funcionals. Van després d'aquestes últimes, ja que les funcionals són més prioritàries per si es troben defectes.

Per executar-se s'ha de fer un "*Refarctor definition*" del WSDL en cada una de les peticions, només tenen un *TestCase* per operació, seleccionar *l'endpoint* que acompanya al WSDL desplegat. Una vegada fets aquests dos canvis ja es poden llançar.

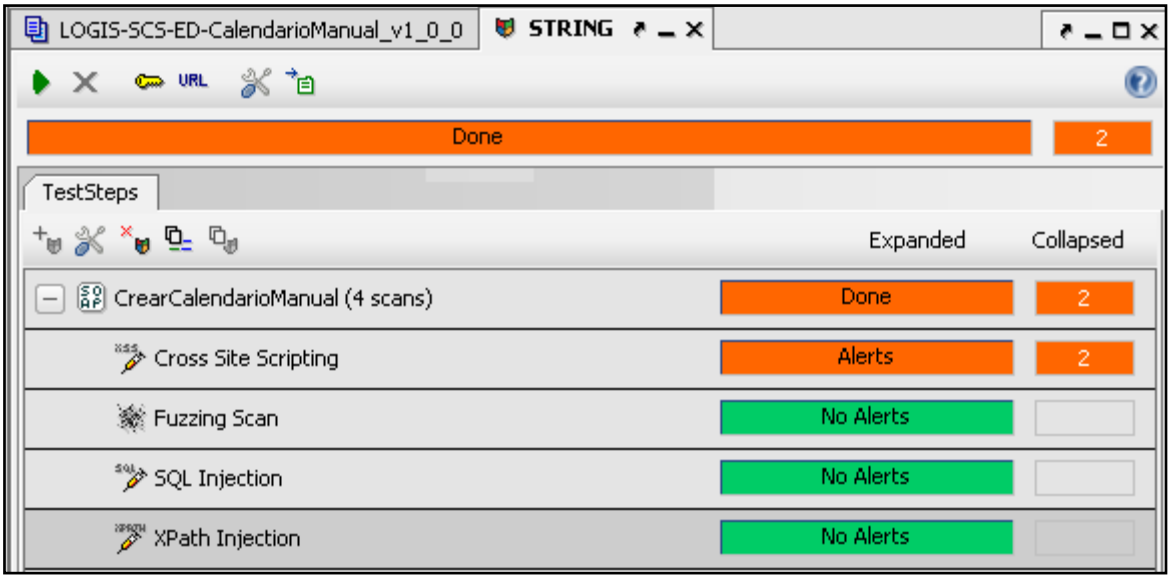
Simplement s'han d'obrir els *TestSecurity* (*String* i *All*) de l'operació que volem executar i donar-li al *play*.



Aquestes proves són molt automàtiques i només s'ha de comprovar que els resultats són correctes. Si ha anat tot bé, veurem en la interfície gràfica de la ferramenta *SoapUI* el següent:

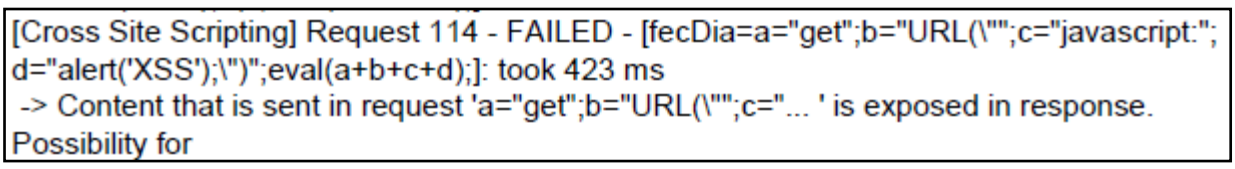


Si s'han trobat defectes apareixen les barres corresponents a cadascuna de les proves de color taronja:



Moltes vegades aquests defectes que es troben en les proves no funcionals de seguretat són falsos positius. Hi ha dos que estan clarament definits:

1. Fals positiu: XSS què és un tipus d'inseguretat informàtica típic d'aplicacions web, que permet injectar en pàgines web codi *JavaScript* o algun altre llenguatge similar.

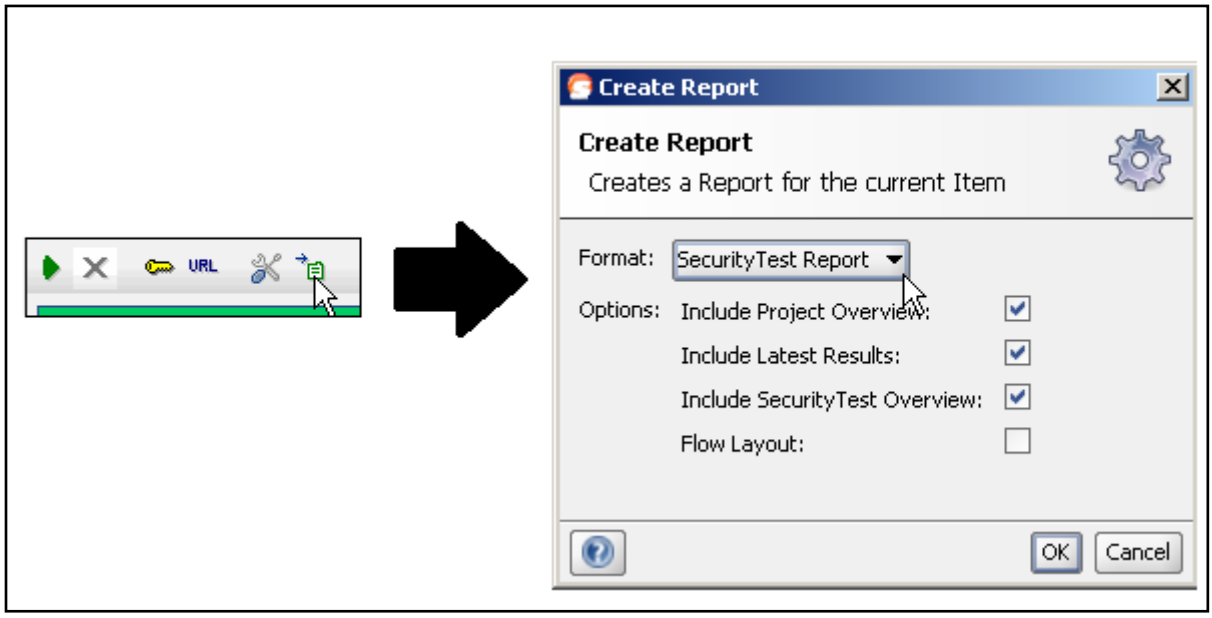


2. Fals positiu: *stacktrace* es dóna quan informació sensible pot estar exposada a terceres persones.

```
[Malformed XML] Request 73 - FAILED - [codItem=<ns:cod_item xmlns:ns="http://www.
mercadona.es/schema/ED_PCTTienda_Detalle/v1/0/0">111111111111</ns:]: took 110 ms
 -> [Stacktrace] Can give hackers information about which software or language you are
using - Token [(?
 -> [Stacktrace] Can give hackers information about which software or language you are
using - Token [(?
 -> [Stacktrace] Can give hackers information about which software or language you are
using - Token [(?
```

Si els defectes trobats en l'execució de les proves és algun d'aquests es descarta i el resultat de les proves es com si no hi havera hagut errors.

Quan s'ha executat un *TestSecurity* s'han de guardar els resultats en un PDF. Es pot fer des de *SoapUI* donant-li al botó del menú i després al "OK", tal com mostra la següent imatge:



Una vegada finalitzades les proves s'ha d'informar dels resultats en l'informe final de proves. Les proves no funcionals de seguretat s'informen en la pestanya "No funcionales" i en l'apartat "Resultados Seguridad". La següent imatge mostra un exemple de la manera i on informar dels resultats d'aquestes proves:

5. Pruebas no funcionales

5.1. Pruebas de seguridad

5.1.1. Resultado de las pruebas

A continuación se adjuntan los resultados obtenidos detallándose, seguidamente, los tipos de pruebas de seguridad ejecutados.

		Ciclo 1	Ciclo 2	Ciclo 3
No conformidades Detectadas	*Bloqueante*			
	Alta	0		
	Media	0		
	Baja			
Severidad del defecto reportado (Bloqueante/Alta/Media/Baja)		No genera defecto		

Tabla 4.1 Resultados de Seguridad

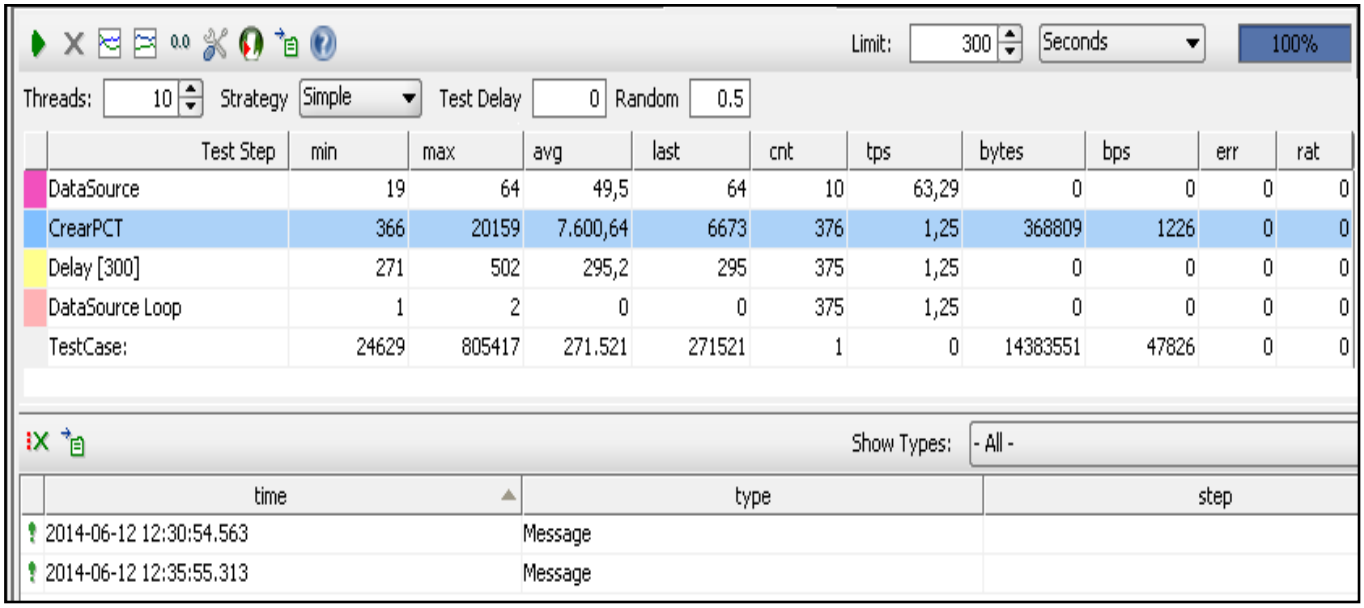
Aquest arxiu PDF, el INP i els logs propis de la ferramenta s'han de pujar al repositori de SVN una vegada les proves hagen finalitzat.

4.6.2.2.Proves de rendiment

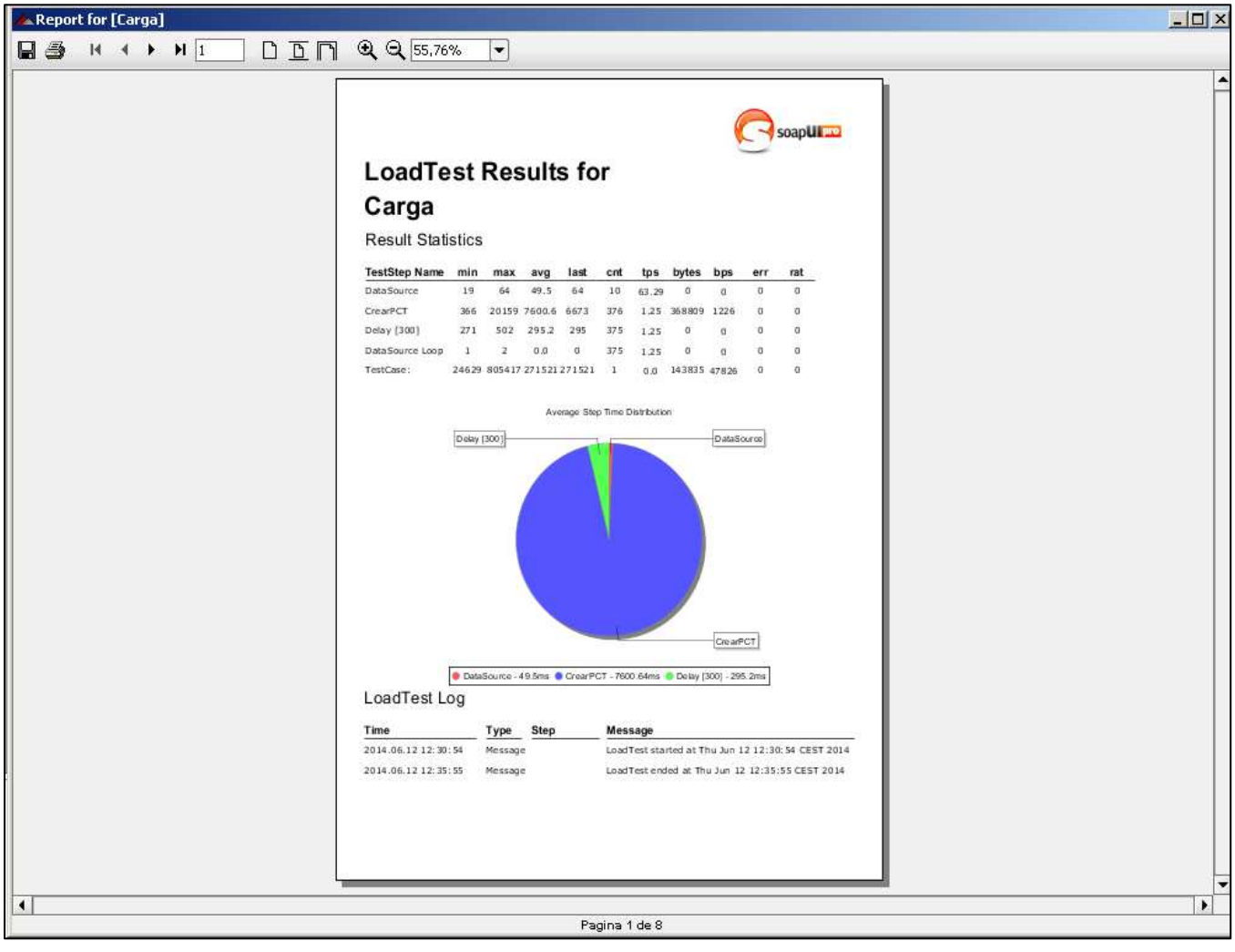
La realització de les proves de rendiment es basa en la simulació d'un determinat volum de peticions que accedeixen de forma concurrent a un component. Serveix per a detectar defectes en el desenvolupament relatiu a accés ineficient a base de dades o males implementacions dels algorismes.

Les proves es duran a terme en un entorn local. El cap d'equip és el responsable de demanar a sistemes via e-mail un entorn de rendiment, quan estiga desplegat i preparat per a funcionar se li contesta al correu. Aleshores el tester ja es capaç de començar a passar les proves de rendiment.

Hi ha que obrir el projecte amb la ferramenta *SoapUI* i fer un "*Refactor*" del WSDL com en les proves anteriors. Una vegada el WSDL ja està actualitzat el *endpoint* s'haura canviat també, per tant tan sols queda obrir el *step* inclòs en "*Load Tests*" que anteriorment em preparat i donar-li a executar des del botó de "Play".



Quan l'execució acabe hi ha que guardar un fitxer pdf que guarda l'execució i els resultats obtinguts. Per a guardar-lo hi ha que premer el botó de guardar que està situat en la part superior (igual que en les proves de seguretat). El resultat deu ser una imatge com la següent:



Una vegada acabades les proves es sol·licita un informe al departament de sistemes, aquest informe es demana en un rang de temps determinat, per això es important saber en quin moment exacte comencen i acaben les proves.

Una vegada es disposa de l'informe s'analitzen els resultats. Si en l'informe es determina que existeix alguna *query* que pot ser optimitzada o es recomana la creació d'un índex es deu incloure en el INP. Si els temps mitjà de resposta d'alguna de les operacions és major que el llindar es reportarà un defecte a FdD.

En canvi si l'informe de sistemes és tot correcte però algun dels temps de resposta segueix per dalt del llindar determinat hi ha que realitzar un nou tipus de prova amb la ferramenta *Yourkit*.

Els resultats de *SoapUI* i els del informe de sistemes es deu actualitzar en la pestanya corresponent a les proves no funcionals del INP. La següent imatge mostra un exemple de com deuen quedar els resultats informats en el INP:

5.2. Pruebas de rendimiento

5.2.1. Resultado de las pruebas

		Ciclo 1	Ciclo 2	Ciclo 3
No conformidades Detectadas	*Bloqueante*	0	0	0
	Alta	0	0	0
	Media			
	Baja			
Severidad del defecto reportado (Bloqueante/Alta/Media/Baja)		No genera defecto		

Tabla 4.4 Resultado de Rendimiento

A continuación se plasma el detalle de cada uno de los ciclos de pruebas de rendimiento donde:

CC: Conexiones Concurrentes.
TMP: Duración de la prueba en segundos.
TMR: Tiempo Medio de Respuesta en milisegundos.
TPE: Total Peticiones WS.
PER: Nº Peticiones erróneas.
RAT: Ratio error peticiones WS[1]
CPU: Comportamiento CPU[2]
ICPU: Nº Conexiones concurrentes a partir de las cuales la CPU se vuelve inestable[3]
RAM: Comportamiento RAM[2]
IRAM: Nº Conexiones concurrentes a partir de las cuales la RAM se vuelve inestable[3]

Detalle Ciclo 1

		Tipos de prueba			
Operación	**Metrica**	*Carga*	*Estrés*	*Estabilidad*	*Escalabilidad*
ConsultarParametros Produ	*CC*	10,00	12,00	0,00	0,00
	TMP	300,00	300,00	0,00	0,00
	TMR	40,75	41,80	0,00	0,00
	TPE	8.624,00	51.162,00	0,00	0,00
	PER	0,00	0,00	0,00	0,00
	RAT	0,00	0,00	0,00	0,00
	CPU				
	ICPU				
	RAM				
	IRAM				

Una vegada informat el INP hi ha que arxivar tots els fitxers corresponents (INP, *logs*, projecte de *SoapUI*...) en el repositori de SVN.

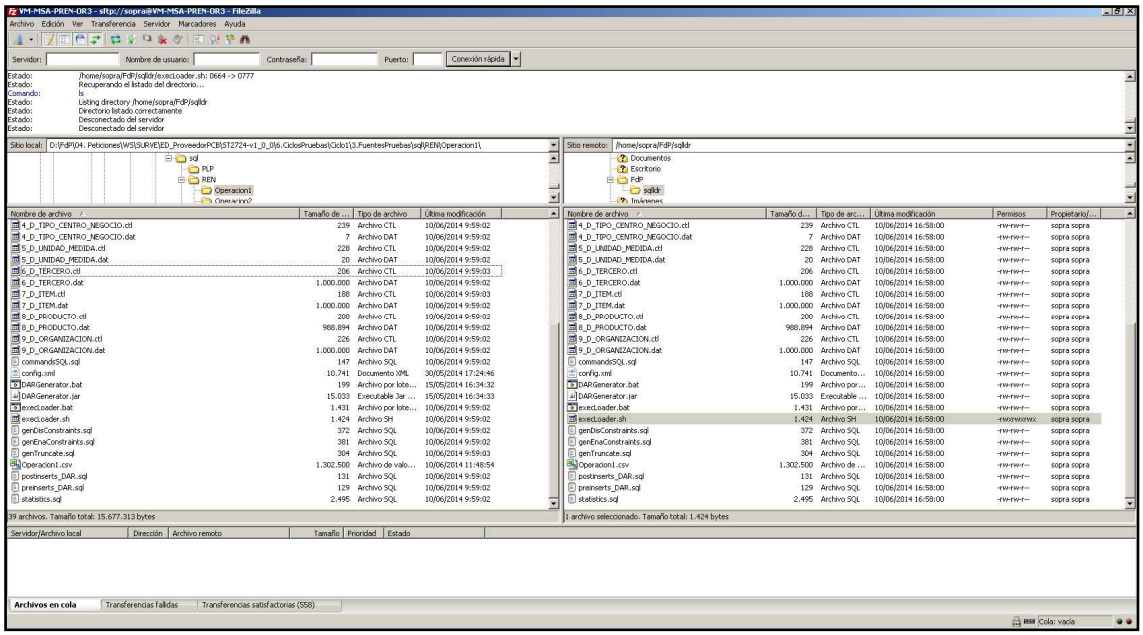
4.6.2.3.Yourkit Java Profile

Quan el resultat de les proves no funcionals de rendiment és un defecte el departament de sistemes realitza un anàlisi per veure quines són les possibles causes d'aquest defecte. Però quan el resultat és un defecte i l'informe des de sistemes està tot correcte s'ha de dur a terme una prova addicional per determinar quina és la causa d'aquest KO.

Aquesta prova addicional és la realització d'una execució per operació amb defecte amb la ferramenta Yourkit. Aquesta ferramenta captura a un nivell de traça molt detallat el funcionament del WebService. D'aquesta manera es pot veure al mínim detall quins recursos i en quins punts l'operació utilitza més recursos del sistema, és a dir, on hi ha un coll de botella.

Per realitzar aquesta prova es deu enviar un correu electrònic al departament de sistemes per a que despleguen l'aplicació en un entorn en mode Yourkit. Una vegada els de sistemes contesten al e-mail vol dir que el servidor amb Yourkit ja està en funcionament

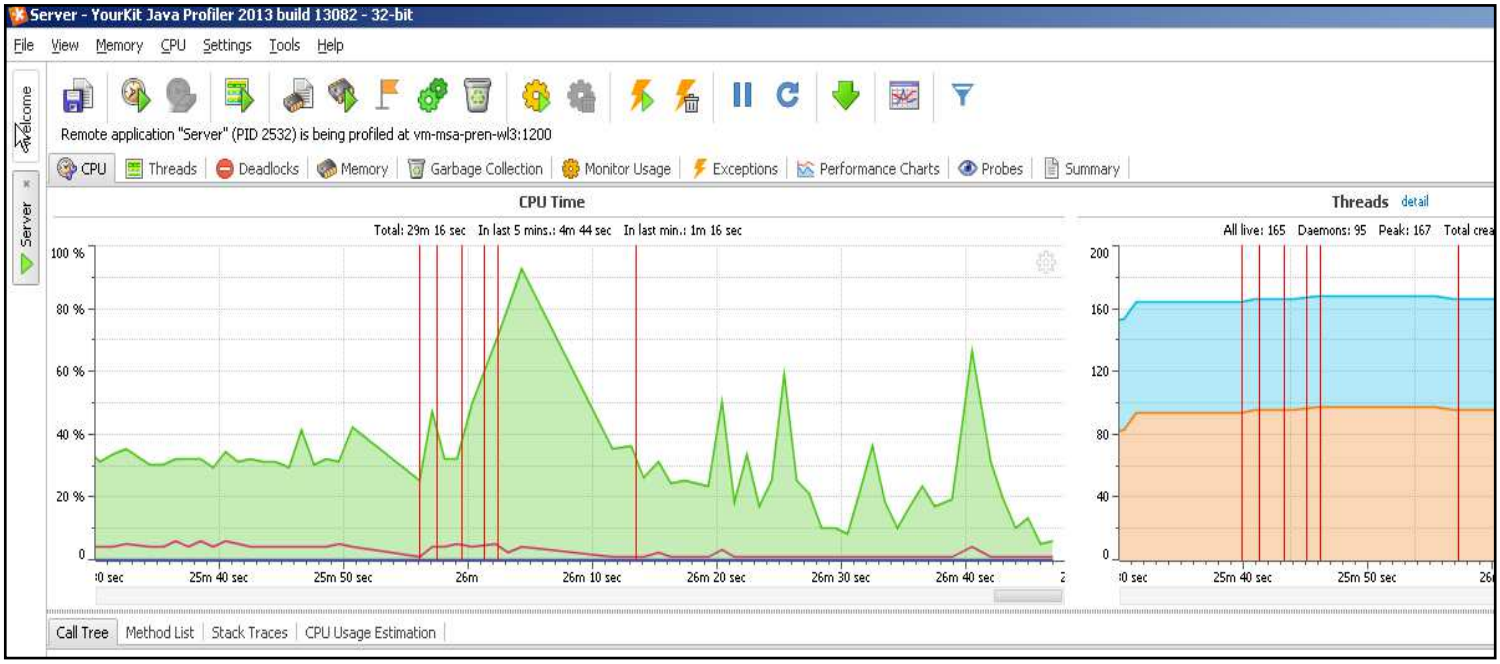
Per a la prova amb Yourkit no es deu utilitzar el mateix projecte de SoapUI SCP sinó que es deu crear un projecte específic, anomenat SYK i la ubicació per guardar-lo és la mateixa que el SCP. Per dur a terme la prova hi ha que carregar les dades en l'entorn de proves mitjançant el Filezilla com si es tractarà d'una prova de rendiment normal. La següent imatge mostra un exemple de com carregar les dades:



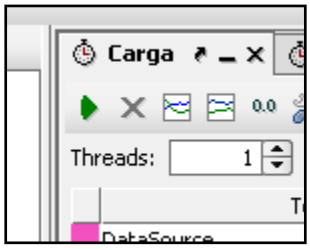
Simplement es tracta de traslladar les dades preparades amb anterioritat a l'entorn de proves. Una vegada copiat en el nou directori hi ha que executar el arxiu execLoader.sh mitjançant el Putty de la següent manera:

```
@VM-PREN-OR3 sqlldr]$ ./execLoader.sh
```

Aquest fitxer és l'encarregat de compilar tots els arxius que em copiat i crear l'estructura de dades corresponent per a dur a terme la prova. Una vegada carregades les dades des de la ferramenta Yourkit hi ha que encendre tots els nivells a analitzar, és a dir, encendre tots els filtres els quals es vol capturar el tràfic:



Ara la ferramenta ja està capturant l'operació a executar, ja s'està en disposició per a llançar una prova de carrega de la operació amb defecte. Tan sols hi ha que prestar atenció al nombre de fils, que deu ser 1:



Una vegada l'operació ha finalitzat s'ha de guardar la traça que ha generat el Yourkit. La traça es guarda amb el nom de l'operació i extensió .snapshot. Aquesta traça s'envia a FdD qui és l'encarregada d'analitzar-la i extreure les conclusions pertinents. També es guarda en el repositori de SVN dins del apartat de rendiment de la petició.

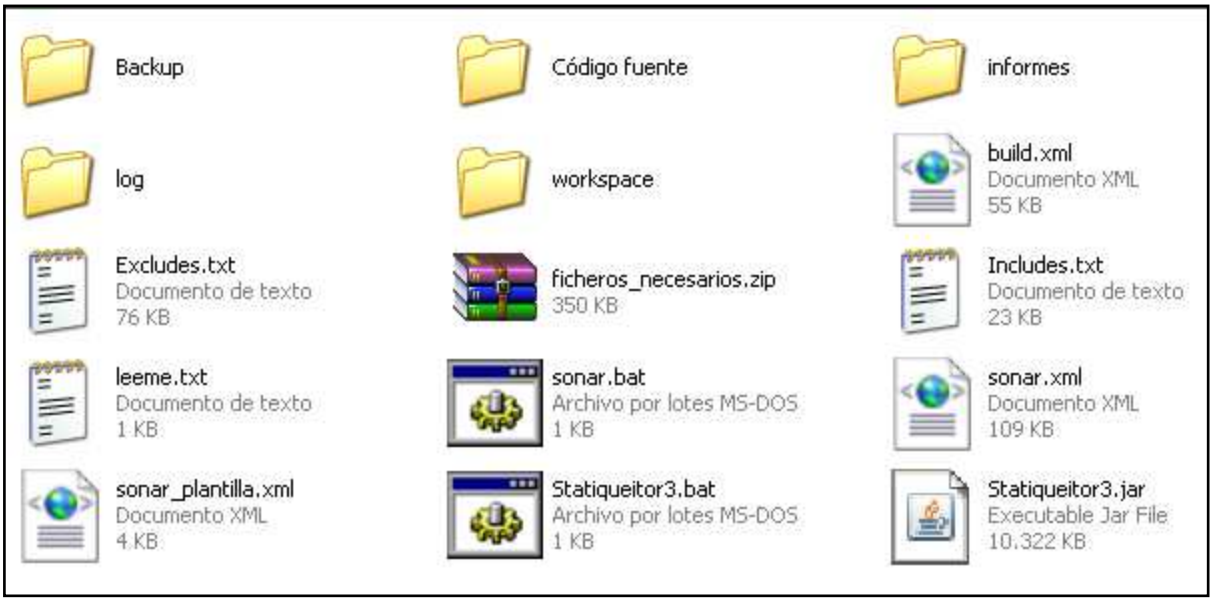
4.6.3.Proves estàtiques

Les proves estàtiques, com indica el seu nom, són proves que no necessiten de l'execució de codi per a dur a terme l'anàlisi d'aquest. Aquestes proves, a diferència de les altres, no es poden tindre preparades per a quan entreguen des de FdD.

La ferramenta que s'utilitza en FdP per a dur a terme les proves estàtiques de codi és Sonar. Aquesta ferramenta presenta una interfície gràfica una vegada s'han finalitzat les proves per a veure els resultats.

Els documents que fan falta per a realitzar les proves són: el codi entregat per FdD i un fitxer anomenat IND que conté els directoris dels *JUnit* que ha d'executar *Sonar*. El IND és un document on llista el arxius entregats per FdD.

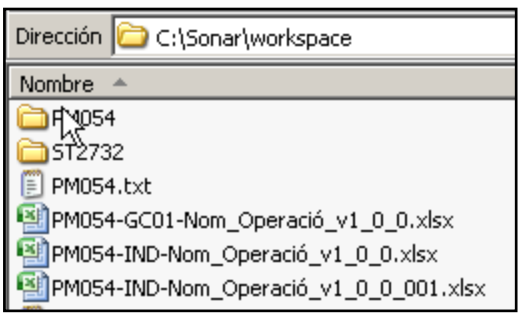
Sonar està estructurada de la següent manera:



No és tracta d'una ferramenta gràfica per poder executar, funciona mitjançant ordres de comandament. Els directoris interessants per a dur a terme les proves són el "*log*" on, com indica el nom, es guarden els logs de les execucions i el "*workspace*" on s'ha de preparar la petició per executar-la.

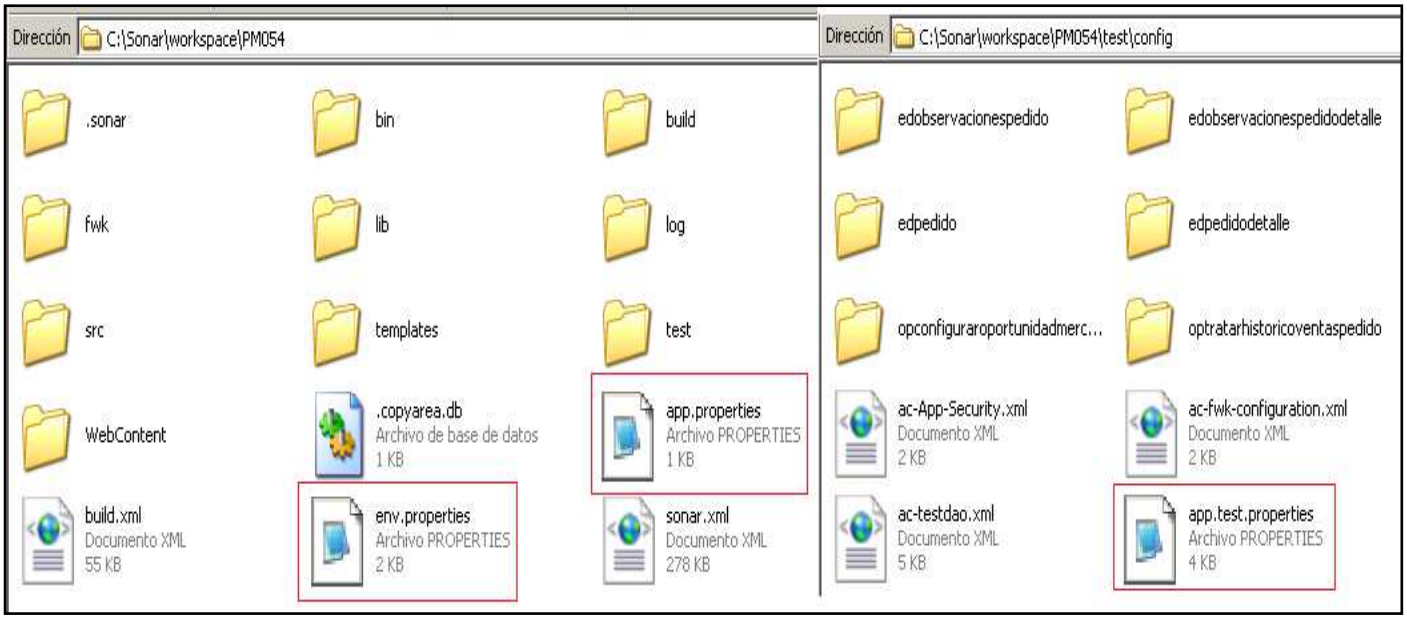
Per automatitzar les proves a nivell intern de FdP s'ha desenvolupat una aplicació per automatitzar aquestes proves i no haver d'utilitzar les ordres de comandament. La ferramenta és Statiqueitor3.

Per a preparar les proves dins del directori "*workspace*" s'ha de crear un directori amb el nom de la petició, un fitxer de text amb el mateix nom que el directori anterior i copiar els arxius IND propis de la petició. Un exemple:



Per a preparar aquesta estructura cal dur a terme els següents passos:

1. Crear el directori amb el mateix identificador que la petició i crear el fitxer de text.
2. Descarregar de CC els següents arxius: els IND que contenen les rutes dels objectes de prova, els fitxers d'entrega de la petició, que és el codi, i els fitxers base, que conté el *framework* i les llibreries per a executar el codi de la petició.
3. Incloure dins del fitxer de text les rutes dels IND de la prova.
4. Dins del directori de la petició guardar els fitxers de la base i els fitxers de l'entrega junts.
5. Configurar els fitxers que es troben dins del directori de la petició i que acabem de copiar. Els fitxers a configurar són els mostrats en la següent imatge:



Dins d'aquests fitxers s'han de realitzar els següents canvis:

En el fixter "*app.properties*" s'ha de canviar el nom del projecte, la versió i l'any que correspon al nom de la petició, a la versió del DTAN i l'any actual. La següent imatge mostra un exemple de la nomenclatura a seguir:

```
# Nombre de la aplicacion
app.name=PM054_ED_Pedido_Detalle_v1_0_0_c1
# Version de la aplicacion
app.version=3.2.0
# Año copyright de la aplicacion
app.copyright.year=2014
```

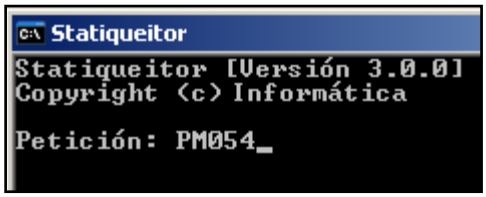
En el fitxer "*env.properties*" s'ha d'afegir la llibreria de *Jacoco* per a poder executar les proves. Es copiaran al final del fitxer les següents línies:

```
# Path de JaCoCo (cobertura)
env.jacoco.path=C:/jacoco/lib/jacocoant.jar
# Path de WebLogic
env.wl.path=C:/oracle/Middleware/wlserver_10.3/server/lib
```

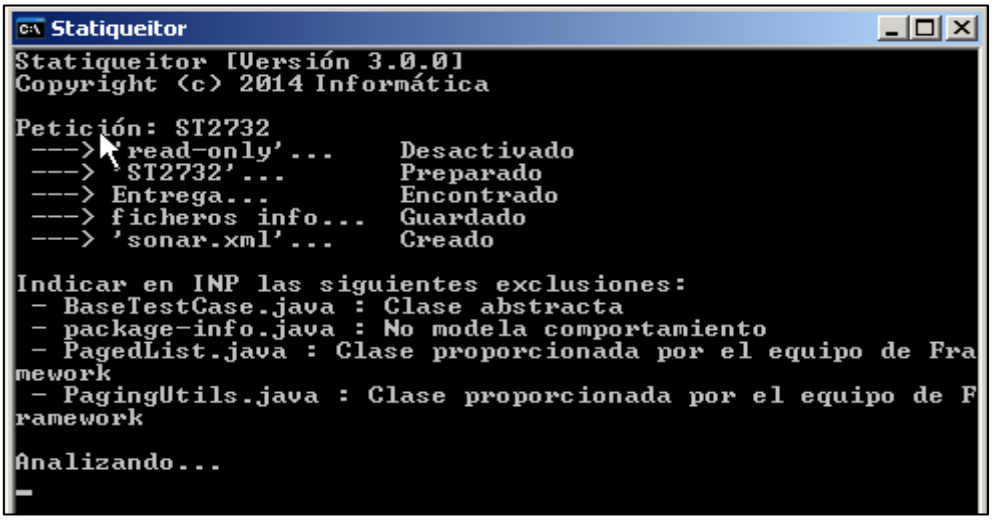
Per últim, del fitxer "*app.test.properties*" s'ha de configurar la connexió de BBDD. Els camps de la direcció, l'usuari i la contrasenya:

```
#Cadena de conexión a la base de datos para tests
test.cadenaConexion=jdbc:oracle:thin:@172.31.106.183:1521
#Usuario de la base de datos de tests
test.usuario=ADM_MAESTRAS_ESTATICAS
#Password del usuario de la base de datos de tests
test.password= *******
```

Una vegada tots el fitxers copiats i configurats en el directori "*workspace*" ja es pot dur a terme l'execució. Per executar obrirem l'arxiu "Statiqueitor3.bat" on només es demana la identificació de la petició. Presenta la següent interfície:



Una vegada posat el nom amb la tecla "*Intro*" llancem l'aplicació. Aquesta ferramenta ens ajuda creant una serie de fitxers necessaris per a l'execució de Sonar. Els arxius de "*includes.txt*", "*excludes.txt*", que es poden veure en la primera imatge d'aquest punt i són arxius que genera Statiqueitor ,contenen els arxius a incloure i excloure de l'entrega. Altres fitxers que genera són el "build.xml" i "sonar.xml". Aquests contenen la configuració de l'execució, com per exemple el *path* dels directoris a analitzar. La següent imatge mostra quines són les ordres automàtiques que llança aquesta aplicació de FdP:



L'anàlisi dura una mitjana de 30 minuts. Durant aquest temps *Sonar* està analitzant el codi. Podem veure com es va desenvolupant en el log de l'execució que es troba en el directori de logs anomenat anteriorment.

```
friendyError:
     [echo] Copying FwkFriendyErrorMessages.properties to build folder
     [exec]         1 archivos copiados.

run-test-sopra:
     [echo] Running tests from C:/Sonar/workspace/ST2732/build/tests to C:/Sonar/workspace/ST2732/build/doc/reports
    [mkdir] Created dir: C:\Sonar\workspace\ST2732\build\doc\reports
[jacoco:coverage] Enhancing junit with coverage
    [junit] Running surve.business.v100.consultarpctcentroporproducto.ConsultarPCTCentroPorProductoManagerconsultarPCTCentroPorProducto10TestCase
    [junit] Tests run: 1, Failures: 0, Errors: 0, Time elapsed: 42,795 sec
    [junit] Running surve.business.v100.consultarpctcentroporproducto.ConsultarPCTCentroPorProductoManagerconsultarPCTCentroPorProducto11TestCase
    [junit] Tests run: 1, Failures: 0, Errors: 0, Time elapsed: 14,953 sec
    [junit] Running surve.business.v100.consultarpctcentroporproducto.ConsultarPCTCentroPorProductoManagerconsultarPCTCentroPorProducto12TestCase
    [junit] Tests run: 1, Failures: 0, Errors: 0, Time elapsed: 16,39 sec
    [junit] Running surve.business.v100.consultarpctcentroporproducto.ConsultarPCTCentroPorProductoManagerconsultarPCTCentroPorProducto13TestCase
    [junit] Tests run: 1, Failures: 0, Errors: 0, Time elapsed: 16,046 sec
```

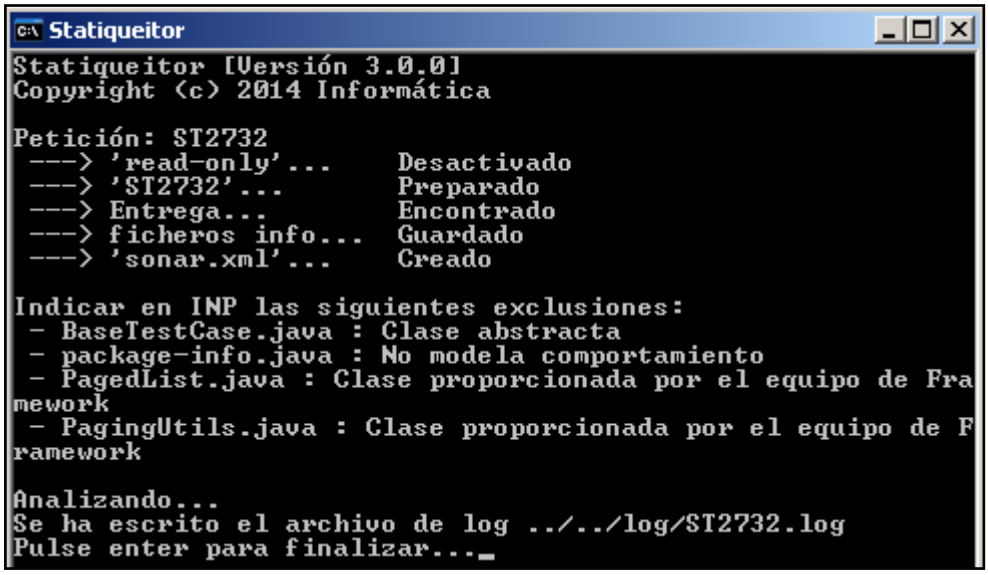
En la imatge d'exemple tot està executant-se correctament. Si hi haguera algun defecte es mostraria el text "*FAILED*" o "*ERROR*" al costat de cada *JUnit*. Podríem veure el defecte entrant al directori corresponent on dóna l'error i veure una xicoteta descripció del mateix. Alguns exemples de defectes trobats durant l'execució són:

```
Testcase: test( surve.business.v100.consultarpctcentroporproducto.ConsultarPCTCentroPorProducto
ORA-00942: la tabla o vista no existeu
java.sql.SQLException: ORA-00942: la tabla o vista no existe

        at oracle.jdbc.dbaccess.DBError.throwSqlException(DBError.java:134)
        at oracle.jdbc.ttc7.TTIoer.processError(TTIoer.java:289)
        at oracle.jdbc.ttc7.Oall7.receive(Oall7.java:582)
        at oracle.jdbc.ttc7.TTC7Protocol.doOall7(TTC7Protocol.java:1986)
        at oracle.jdbc.ttc7.TTC7Protocol.parseExecuteFetch(TTC7Protocol.java:1144)
        at oracle.jdbc.driver.OracleStatement.executeNonQuery(OracleStatement.java:2152)
        at oracle.jdbc.driver.OracleStatement.doExecuteOther(OracleStatement.java:2035)
        at oracle.jdbc.driver.OracleStatement.doExecuteWithTimeout(OracleStatement.java:2876)
        at oracle.jdbc.driver.OracleStatement.execute(OracleStatement.java:945)
        at org.h2.tools.RunScript.execute(RunScript.java:167)
        at test.common.TestManager.ejecutarScript(TestManager.java:31)
        at test.common.TestManager.recargarEscenario(TestManager.java:39)
        at surve.business.edpcttiendadetallev100.v100.consultarpctcentroporproducto.ConsultarP
```

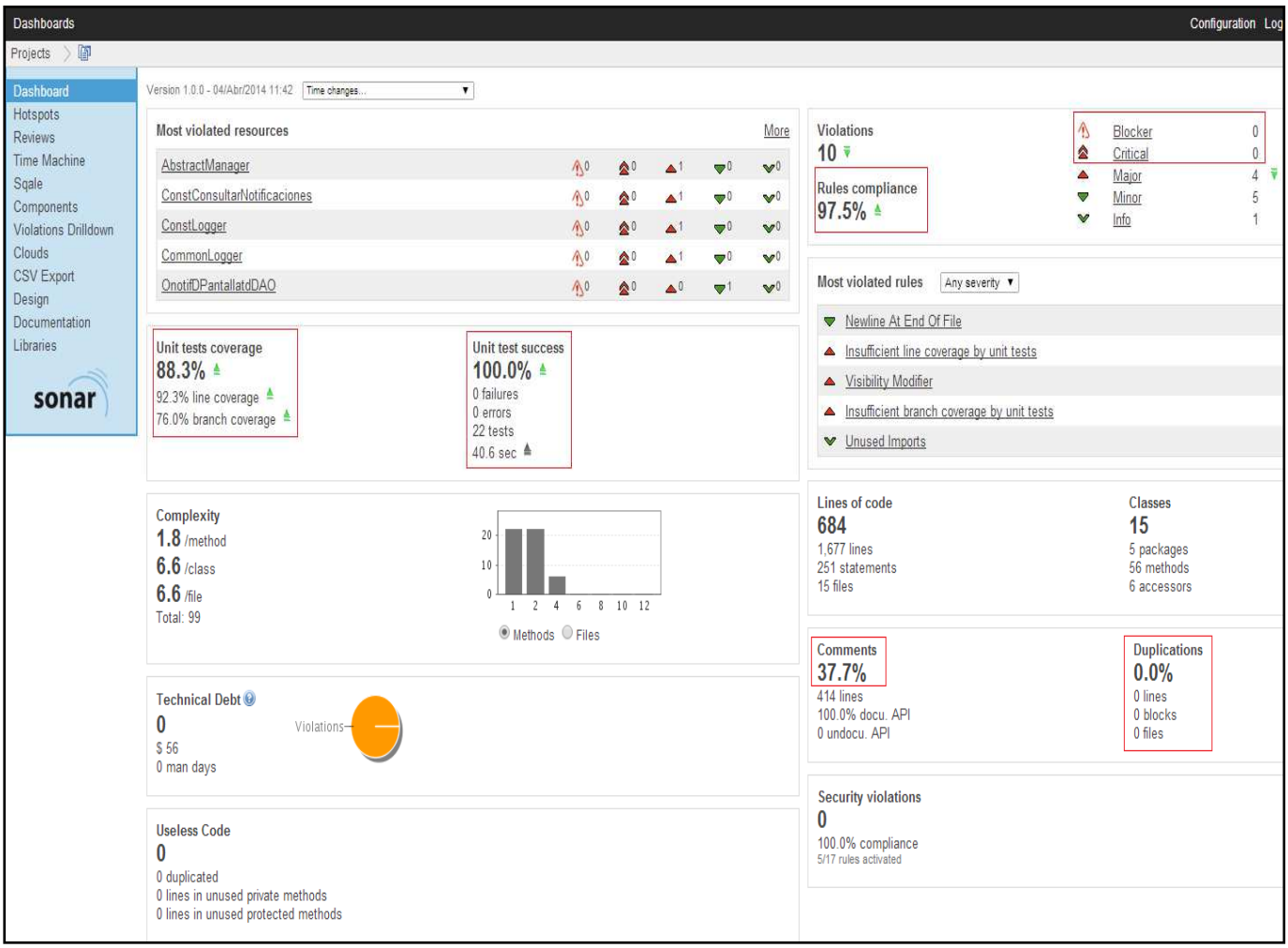
Com podem veure en les imatges anteriors s'ha produït una fallada en un *JUnit*. Un test falla perquè deuria tornar com a resultat un valor i ha sigut un altre. Es poden veure els defectes al buscar aquest *JUnit* en el directori "*\workspace\<dirPetició>\build\tests*" en un fitxer de text on està la descripció. En

aquest cas la fallada ha estat provocada perquè una de les taules que opera la petició no es troba en les DDL que s'han carregat per a dur a terme les proves.

La descripció que mostra el *Statiqueitor* al finalitzar les proves si tot ha anat correctament és la següent:



Ara cal entrar en la direcció del servidor de Sonar i buscar les proves que acabem llançar. Al obrir les proves apareix una finestra on es mostren els resultats de les proves estàtiques:



Aquests resultats que presenta el Sonar s'han d'informar en el INP en l'apartat corresponent a aquestes proves, pestanya "Pruebas estáticas". En aquests apartats cal emplenar els punts de proves de qualitat de codi i de proves de caixa blanca:

3.2. Auditoría de cumplimiento de estándares y pruebas de calidad de código

3.2.1. Resultados de las pruebas

Tipología de No-conformidad	Umbral de aceptación	Ciclo 1	Ciclo 2	Ciclo 3
Violaciones Bloqueantes	0	0	0	0
Violaciones Críticas	0	0	0	0
% Reglas Sonar cumplidas	90%	96%	96%	0%
% Mínimo de Comentarios	30%	36%	36%	0%
% Código Duplicado	5%	0%	0%	0%
Número de métodos de Complejidad Ciclomática > 10	0	0	0	0
% Código Muerto	0%	0%	0%	0%
Severidad del defecto reportado		No genera defecto	No genera defecto	

Tabla 3.7 Resultados Auditoría de estándares y pruebas de calidad de código

Elementos excluidos del análisis:

Elemento	Motivo
BaseTestCase.java	Clase abstracta.
package-info.java	No modela comportamiento.
PagedList.java	Clases proporcionadas por el equipo de Framework.
PagingUtils.java	

Tabla 3.8 Detalle elementos excluidos

Violaciones detectadas:

Violaciones	Criticidad	Descripción

Tabla 3.9 Detalle violaciones detectadas

3.2.2. Comentarios adicionales

[Si hay comentarios adicionales a realizar, incluir el siguiente texto 'Sin comentarios adicionales.']

3.3. Verificación de pruebas de caja blanca

3.3.1. Resultados de las pruebas

Tipología de No-conformidad	Umbral de aceptación	Ciclo 1	Ciclo 2	Ciclo 3
% Cobertura Sentencias	85%	88%	88%	0%
% Éxito Pruebas Unitarias	100%	100%	100%	0%
Severidad del defecto reportado		No genera defecto	No genera defecto	

Tabla 3.10 Detalle violaciones detectadas

3.2.2. Comentarios adicionales

2. Resultados de las pruebas | **3. Pruebas estáticas** | 4. Funcionales | 5. No funcionales | Ane

Després d'emplenar l'informe amb els resultats deu pujar-se al repositori SVN.

4.7.Resolució de cicle 1

Passats els tres dies de prova, quan totes les proves estiguen executades, suposant que hagen aparegut defectes en alguna de les proves el següent pas correspon a la resolució per part de FdD dels defectes reportats.

En aquest moment FdD disposa de dos dies per a resoldre els defectes trobats en el cicle 1 per part de factoria de proves i entregar-los a la mateixa per a començar el cicle 2 de les proves. A partir d'aquest cicle tant FdD com FdP disposen d'un dia per a la resolució de defectes i realització de proves de regressió respectivament.

4.8.Entrega al client

Una vegada la petició està lliure de defectes, o el client vol dur-la endavant a pesar els defectes, es procedeix a l'entrega de la petició al client. En aquest punt es realitza la revisió de l'entrega, el QA Proves. Es disposa d'un document anomenat "Índex d'entrega QA Proves" que es la guia per assegurar que es disposa de tots els documents i es segueix una estructura determinada a l'hora d'entregar la documentació.

Per dur a terme l'entrega simplement hi ha que seguir aquest guió i recollir els documents corresponents per emmagatzemar-los en un paquet. Una vegada tots els documents estiguen en el seu lloc el cap d'equip és el responsable de pujar aquest conjunt de documents a CC.

5. Ferramentes

5.1.*Jira & ClearQuest*

5.1.1.Jira

JIRA és una aplicació multiplataforma que funciona via web per al seguiment d'errors, d'incidents i la gestió operativa de projectes. Es tracta d'un producte comercial, i per tant adquirir una llicència per utilitzar-lo té una despesa anual.

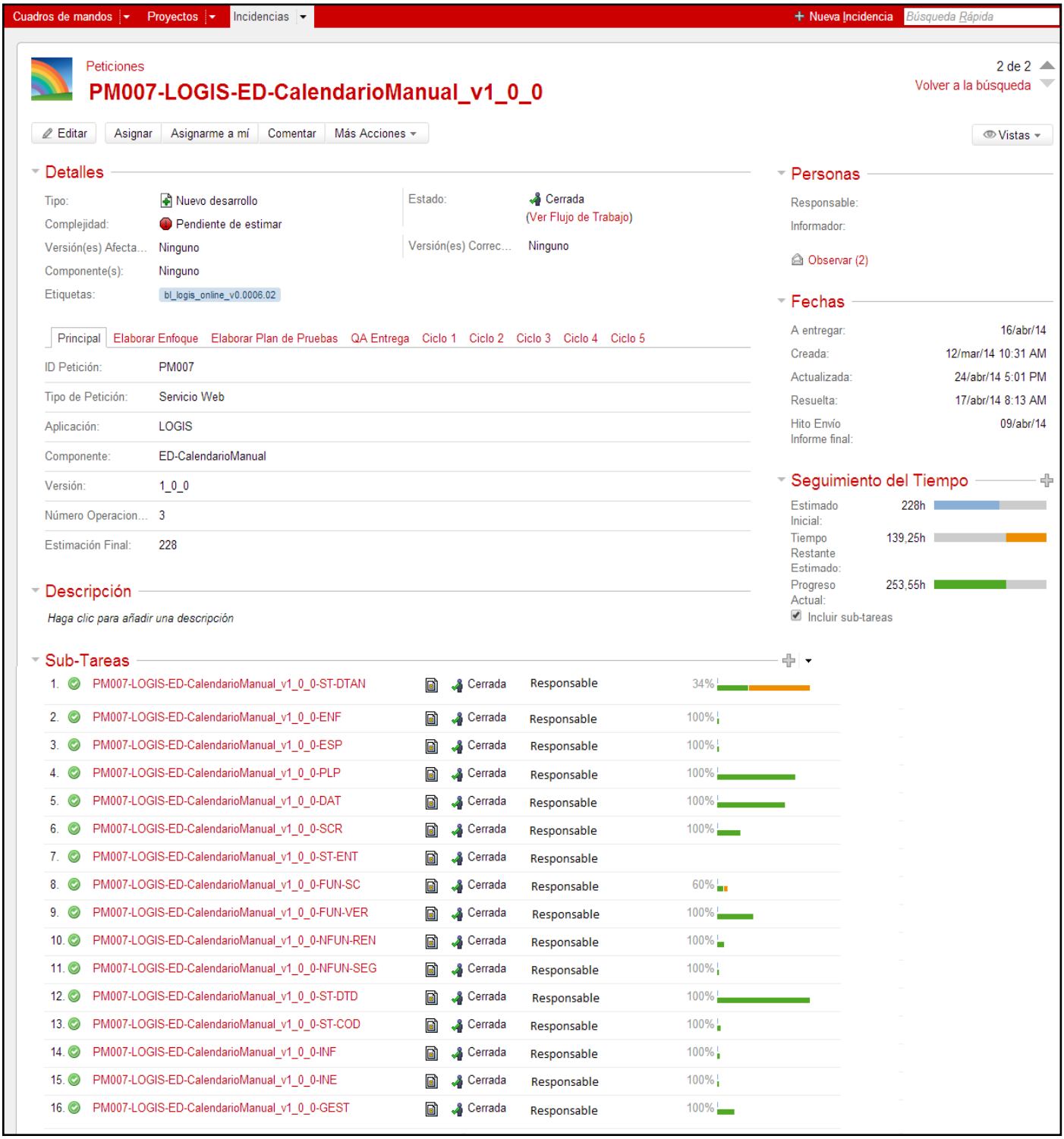
Per tal d'organitzar el projecte, prioritzar i actuar sobre allò més important i estar al dia amb el que està succeint, la direcció de l'empresa decideix utilitzar *JIRA*, ja que és una ferramenta que gràcies a la seua interfície simple i intuïtiva permet compartir la informació més fàcilment i de forma més eficient que altres ferramentes organitzatives. A banda, s'ha decidit utilitzar *JIRA* perquè permet una connexió amb el repositori utilitzat per a la compartició de fitxers (*SubVersioN*).

S'utilitza a nivell intern per FdP com a ferramenta de gestió de peticions. JIRA proporciona una gestió de les peticions del servei amb diferents nivells d'abstracció, seguiment i anàlisi de traçabilitat. A més, permet modelar l'execució d'una activitat mitjançant flux de treball personalitzats que determinen l'evolució de les peticions.

Les funcionalitats que s'utilitzen per a organitzar el projecte són:

- **Panels de control**: per tal de saber en quin estat es troba la petició, obtindré estadístiques, etc.
- **Filtres de recerca**: buscador ràpid de peticions amb possibilitat d'aplicar filtres.
- **Flux de treball**: informa en quin estat es troba la petició i per quins estats ha passat.
- **Imputar hores**: per a saber quin serà el total d'hores dedicades a cada tasca.

Quan arriba una nova petició a FdP es crea l'estructura per a la seua gestió. La petició tindrà un identificador, el nom del *WebService* i la versió. La següent imatge mostra l'estructura triada com a determinada per a cada petició:



Com es pot apreciar en la imatge, donada una petició es divideix en moltes subtasques. D'aquesta manera es porta un control més al detall del responsable de cada part de la petició i al mateix moment, l'estat en que es troba.

Per tal que els equips siguen més eficaços i eficients a l'hora de capturar, assignar i prioritzar el treball s'assignen les tasques a determinades persones de l'equip, així tot l'equip sap exactament que cal fer i qui ha realitzat una determinada tasca per si

li sorgeixen dubtes respecte a una tasca realitzada prèviament per un company. Les tasques s'assignen a les persones mitjançant la interfície del *JIRA*. La persona assignada és responsable del flux de treball de la mateixa, ha de posar-la "en progrés" i una vegada ha acabat de realitzar la faena que li corresponia passar-la a "finalitzat". També es deu imputar les hores dedicades a realitzar les proves, de manera que els organitzadors puguen determinar el volum d'hores que ha invertit l'empresa en la petició. Quan s'informen les hores es deu acompanyar d'un comentari informant si el resultat de les proves ha sigut correcte o no.

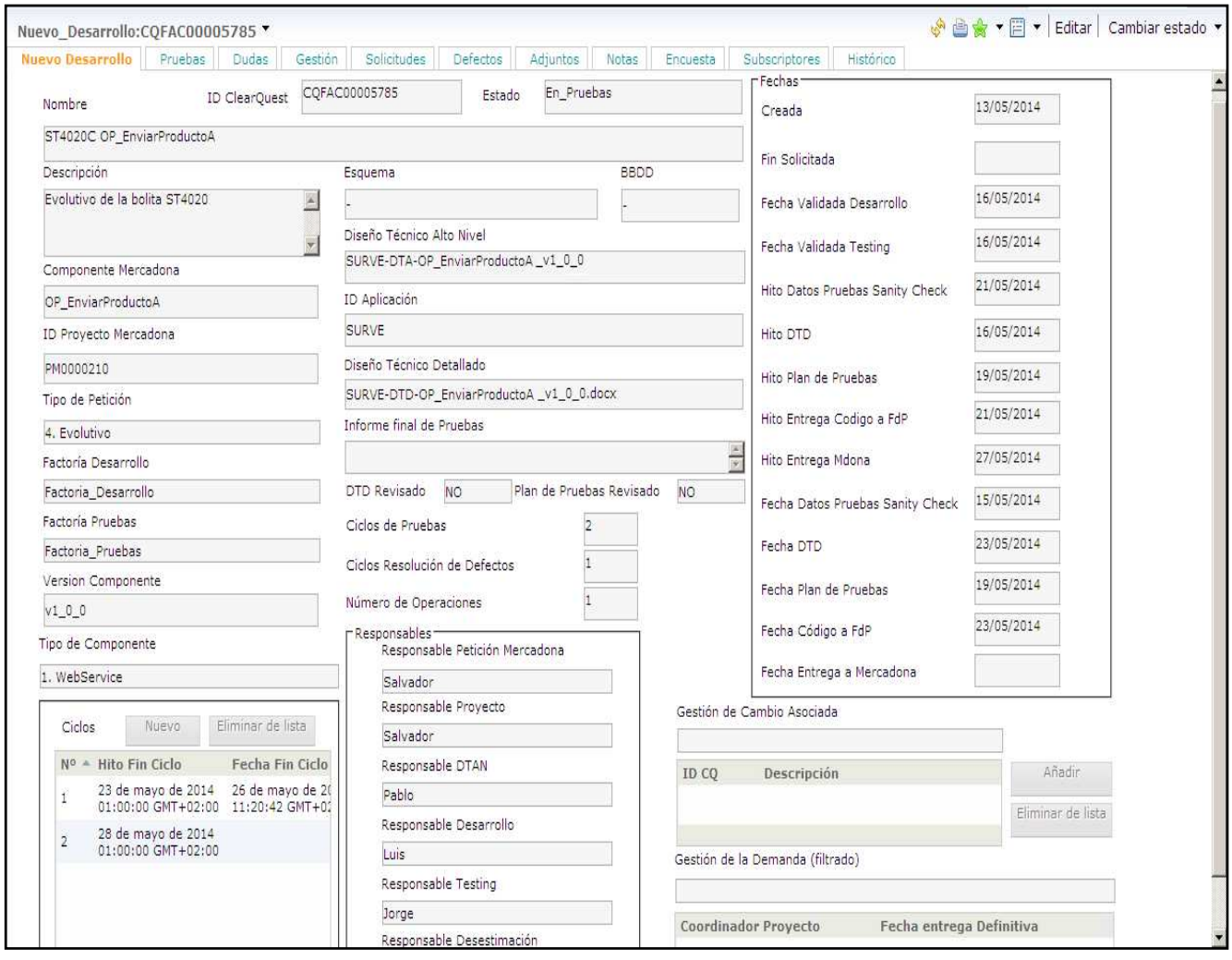
Per poder finalitzar una petició totes les tasques que la conformen deuen estar finalitzades. Una vegada s'ha acabat el cicle de vida d'una petició el responsable accedirà al *JIRA* per agafar la informació pertinent i dur-la al respectiu informe final.

5.1.2. Rational ClearQuest

Es tracta d'un software per a gestionar el cicle de vida de les peticions, proporciona un seguiment flexible dels canvis i defectes en temps reals per a cadascuna.

És un programari multiplataforma, escalable a organitzacions de qualsevol tamany.

Cadascuna de les peticions obertes pel Client en CQFAC té el següent aspecte:



Com es veu en la imatge una petició ve acompanyada d'una sèrie de camps per gestionar-la. Al igual que en *Jira*, en aquesta ferramenta la petició també canvia d'estat depenent en quin moment del cicle de vida es troba. De les pestanyes que es veuen en el menú superior les més importants per a provar la petició són "Dudas" i "Defectos". En la primera apareixen els dubtes oberts durant la validació del DTAN i en quin estat es troben. La segon pestanya fa referència als defectes oberts durant les proves.

La primera part mostra un exemple de dubtes mentre que la segon de defectes:

Per política d'empresa els testers no tenen accés a aquesta ferramenta i són els analistes o caps d'equip els encarregats de dur a terme totes les gestions en CQFAC.

5.2.*Subversion & ClearCase*

5.2.1.Subversion

SubVersioN (SVN) és una ferramenta de control de versions basada en un repositori per a la compartició de documents. SVN s'utilitza com a repositori al que s'accedeix a través de la xarxa, permetent així utilitzar-se per a la compartició col·laborativa.

SVN permet als usuaris crear, copiar i eliminar carpetes i documents amb la mateixa flexibilitat amb la que ho faries de forma local. La possibilitat de que diverses persones puguen modificar i administrar el mateix document s'ha de posar cura a l'hora de fer modificacions. Permet utilitzar l'opció de bloquejar un determinat document o directori de manera que ningú puga modificar-lo mentre treballes.

Al treballar sobre versions en un determinat moment es pot desfer un canvi.

Abans de fer qualsevol modificació els usuaris deuen assegurar-se que estan treballant sobre l'última versió del repositori.

S'utilitza de forma col·laborativa perquè presenta molts avantatges:

- Historial de les accions realitzades sobre el repositori
- Permet bloquejar arxius per a evitar conflictes
- Ideal per al treball en equip sobre el mateix repositori
- Permet la modificació en paral·lel de documents del repositori, és un avantatge sobre altres ferramentes de control de versions que obliguen a bloquejar algunes zones del repositori.

Desavantatges:

- El re-nomenament no està optimitzat, es realitza mitjançant una còpia i una eliminació.
- Fàcil arribar a un conflicte, és a dir, que dos usuaris modifiquen un arxiu al mateix temps, provocant així un conflicte entre fitxers. Per a evitar aquest estat s'ha de fer *update* abans de treballar amb uns determinats documents i *commit* per a actualitzar els canvis en el repositori. En cas que els usuaris modifiquen el mateix element a la vegada, la ferramenta integrarà els canvis de forma automàtica, obligant a l'usuari a corregir-ho de forma manual per a assegurar la correcta integració.

La mecànica de treball és la següent:

Abans de començar a treballar amb una tasca, es deu assegurar la sincronització amb el repositori o bé mitjançant l'opció de *checkout* o amb un *update*. Finalment es deu fer un *commit* per a fer públic a la resta de l'equip els canvis realitzats, l'abast *checkout* es descarrega a l'entorn local una còpia del codi del repositori. Amb l'opció *update* es descarreguen a l'entorn local únicament les modificacions que hagen tingut lloc des de l'última sincronització, aquesta opció només està disponible si es disposa d'una versió local. Amb l'opció de *commit* s'actualitzarà el contingut del repositori amb els canvis de l'entorn local. SVN tan sols es durà a terme si no existeixen conflictes amb el repositori.

5.2.2. Rational ClearCase

Rational ClearCase és el sistema de control de versions d'IBM. Aquesta ferramenta proporciona control de versions, gestió d'espais de treball i suport al desenvolupament en paral·lel. S'utilitza per a gestionar arxius en tot el cicle de vida de les peticions. A banda, s'utilitza per a tota la factoria per gestionar i solucionar problemes.

El repositori on s'emmagatzemen els elements s'anomena base d'objectes versionats (VOB). Les modificacions que es duen a terme durant les tasques es

registren en activitats, que consisteixen en diverses versions de diferents elements d'un component.

Una de les principals característiques de CC són les vistes. Una vista és una llista selectiva dels elements que es mostren, així no fa falta fer una descàrrega de tot el directori. Es poden seleccionar els arxius necessaris i treballar sobre ells, de manera que quan es faça un *update*, tan sols s'actualitzen els arxius configurats en la vista.

L'arbre de versions és una altra característica dels sistemes de control de versions, açò permet recuperar arxius anteriors en cas d'error.

Aquesta característica ajuda a treballar de forma paral·lela sobre el mateix arxiu sense que hi hagen col·lisions a l'hora de modificar els mateixos elements. Aquests seran fusionats posteriorment en les operacions *deliver* i *rebase*. Sobre una vista també es fan operacions d'actualització per a que els canvis apareguen sobre ella.

Està imposat en Factoria que tots els proveïdors deuen allotjar tots els arxius en *ClearCase*. D'aquesta manera en FdP el cap d'equip és el responsable de descarregar tots els arxius referents a la petició en la qual està treballant-se i allotjar-los en SVN, què és la ferramenta a nivell intern. De la mateixa manera, quan les proves acaben, tots els documents deuen allotjar-se en *ClearCase* per tal que estiguen disponibles per a tota la Factoria.

A l'estar imposada pel client no s'han tingut en compte alternatives a aquest gestor de versions. Encara que a nivell intern s'utilitza SVN per la seua senzillesa i també perquè les llicències *d'IBM Rational ClearCase* tenen un cost molt elevat.

5.3.*Microsoft Office*

Del paquet d'ofimàtica de l'empresa *Microsoft* s'utilitza el *Microsoft Word* per al processament de text de tots els documents de la factoria. Tots els documents (DTAN, DTD, etc) de text que es tracten al llarg del cicle de vida tenen l'extensió de *Microsoft Word*, ja que el client ha decidit adquirir llicències per a treballar amb aquesta ferramenta.

Del paquet d'ofimàtica el programa més utilitzat és l'*Excel*. Aquesta ferramenta s'utilitza per a la representació de totes les dades de la petició i en alguns casos també s'utilitza com a ferramenta per a passar algunes proves. Per exemple, a l'hora de passar les proves estàtiques s'utilitza una plantilla de *checklist* desenvolupada en graelles en *l'Excel*. Per altra banda, també s'utilitza la ferramenta per a la representació de les dades del pla de proves.

L'Excel es tracta d'una ferramenta que, gràcies a la seua gran funcionalitat, proporciona una gran ajuda a l'hora de representar la informació.

5.4.Sql Developer & Mysql

5.4.1.Mysql

MySQL és un sistema d'administració de bases de dades (*Relational Database Management System*) per a bases de dades relacionals, multifil i multiusuari que utilitza el llenguatge SQL. Així doncs, *MySQL* és una aplicació que permet gestionar arxius de bases de dades.

MySQL, com a bases de dades relacional, utilitza múltiples taules per a emmagatzemar i organitzar la informació.

Es tracta d'una ferramenta d'utilització gratuïta d'*open source* que funciona molt bé amb aplicacions web, sobretot amb PHP. Aquestes raons han afavorit positivament en fer que *MySQL* siga una de les ferramentes més utilitzades per a bases de dades en aplicacions web.

5.4.2.Sql Developer

Oracle SQL Develper és una ferramenta gràfica gratuïta que proporciona Oracle per a que no siga necessari utilitzar ferramentes de tercers (*TOAD. PL/SQL Developer*, etc) per a desenvolupar, o simplement executar consultes o *scripts* SQL sobre bases de dades Oracle.

En les últimes versions s'han incorporat millores per a permetre connectar-se a BBDD no Oracle (*SQLServer, MySQL o Access*). La connexió amb *MySQL* es realitza a través de JDBC.

JDBC és una API de Java que s'utilitza per a executar sentències SQL (com a curiositat JDBC es un nom d'una marca registrada i no un acrònim, encara que de vegades s'interpreta com *Java DataBase Connectivity*). Consta d'un conjunt de classes i interfícies en el llenguatge de programació Java.

Una vegada establerta la connexió s'utilitza amb total funcionalitat per a executar sobre la BBDD sentències SQL. Aquesta ferramenta millora la productivitat i simplifica les tasques de desenvolupament per a les bases de dades.

En FdP s'utilitza *SQL Developer* per a carregar els jocs de dades de les proves en BBDD i realitzar sentències sobre BBDD.

En algunes peticions s'utilitza *SQL Developer* per a guardar una mena de logs. Si la petició a provar té operacions de Crear/Eliminar/Modificar es realitzarà prèviament a llançar l'operació des del *SoapUI* unes consultes sobre les taules implicades. Després d'executar el cas de prova es tornarà a realitzar les consultes pertinents a les taules implicades en l'operació per comprovar que efectivament s'ha creat/eliminat/modificat els registres pertinents.

5.5.Pròpies

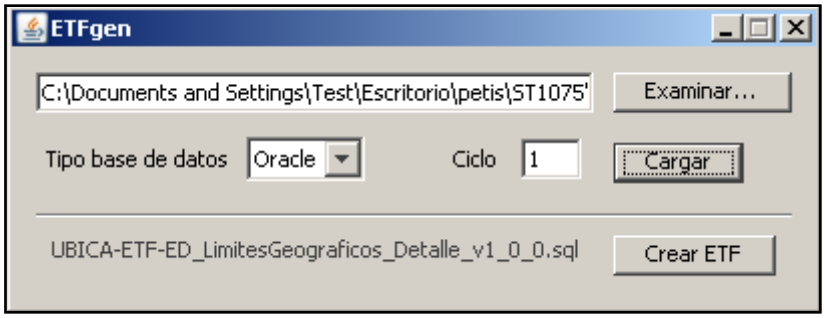
Per tal d'estalviar en temps i automatitzar les proves, a nivell intern FdP ha desenvolupat una sèrie de programes. Entre ells destaquen: *Statiqueitor* v3, *ETFGen* i *DARGenerator*. Aquestes ferramentes estan desenvolupades en el llenguatge de programació Java.

5.5.1.Statiqueitor v3

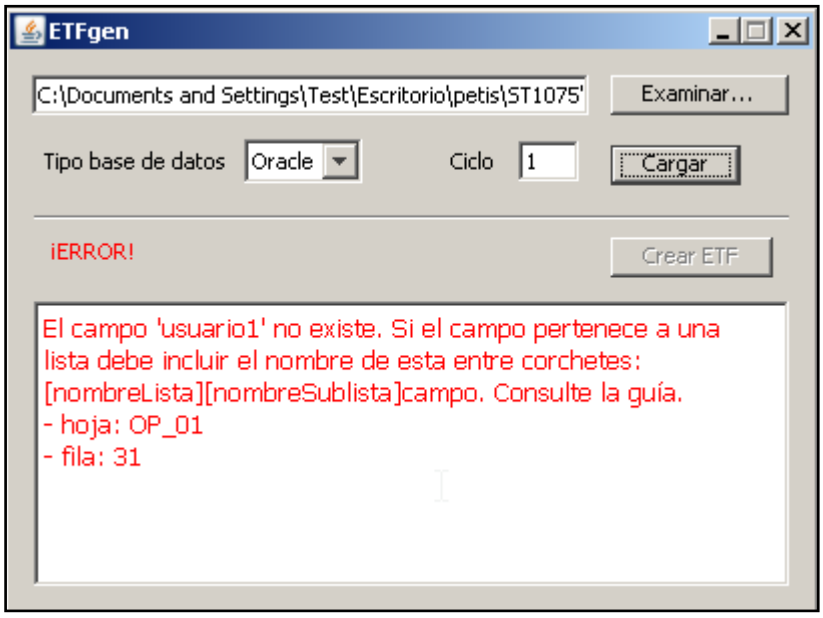
Una vegada preparat el directori de Sonar es llança aquesta ferramenta que permet automatitzar les proves estàtiques de codi. Està explicada en l'apartat d'execució de proves estàtiques.

5.5.2.ETFGen

Es tracta d'una ferramenta per a generar de forma automàtica els jocs de dades ETF a partir del PLP. La ferramenta presenta la següent interfície gràfica:



El botó "Examinar..." serveix per buscar la ruta del PLP. El camp "Ciclo" s'utilitza per a indicar en quin cicle de les proves ens troben. Una vegada aquest dos informats hi ha que premer el botó de "Cargar". Si no hi ha errors en la carrega del ETF el botó de "Crear ETF" estarà disponible com en la imatge anterior. En canvi, si hi han errors es mostraran davall i fins que no estiguin solucionats no es podrà generar el ETF.



5.5.3.DARGenerator

Es tracta d'una ferramenta per a generar automàticament els joc de dades per a les proves de rendiment. A través d'un executable ".bat" s'executa el programari de *java*. Per a executar el .bat és necessari haver emplenat un fitxer de configuració *.xml* en el mateix directori. El fitxer de documentació conté el següent:

```
<schema>

    <name>adm_maestras</name>


    <table>
        <name>D_TIPO_LOCALIZACION</name>
        <columns>
            <column>
                <name>COD_N_TIPO_LOCALIZACION</name>
                <type>Number</type>
                <init>35732</init>          → Valor inicial
                <incremental>true</incremental>
                <frequency>1</frequency>    → la freqüència en que canvia
                <minValue>1</minValue>
                <maxValue>99999</maxValue>  → Valor mínim i màxim
            </column>
            <column>
                <name>TXT_NOMBRE</name>
                <type>varchar2</type>
                <init>ST3570 TIPO_LOCALIZACION OP_03</init>
                <incremental>false</incremental>
                <frequency>1</frequency>
                <minValue>1</minValue>
                <maxValue>1</maxValue>
            </column>
        </columns>
        <nrows>500</nrows>                  → nombre de valors a insertar
        <commitFrequency></commitFrequency>
    </table>
```

Per a cada taula del xml el *DARGenerator* genera dos fitxers: "taula.ctl" i "taula.dat". El primer és l'encarregat de complementar al segon. Es tracta d'un fitxer de configuració de l'altre on s'informen entre altres els camps, la separació entre valors... El fitxer "taula.dat" conté els inserts de la taula on estan els valors de les columnes separats en ';' per defecte i les files per salts de línia.

```
# fichero .ctl
load data
infile '5_d_tipo_tercero.dat'
badfile '5_d_tipo_tercero.bad'
discardfile '5_d_tipo_tercero.discard'
APPEND
into table d_tipo_tercero
FIELDS TERMINATED BY ';'

trailing nullcols
(cod_n_tipo_tercero,
txt_nombre)
```

```
# fichero .dat
27251;tipoTercero27251;
27252;tipoTercero27251;
27253;tipoTercero27251;
27254;tipoTercero27251;
27255;tipoTercero27251;
27256;tipoTercero27251;
27257;tipoTercero27251;
27258;tipoTercero27251;
27259;tipoTercero27251;
27260;tipoTercero27251;
27261;tipoTercero27251;
```

Un altre fitxer important autogenerat és el *statistics* que conté les estadístiques (Preguntar a María) per a que la base de dades siga més eficient a l'hora de fer les proves de rendiment.

Per últim, es genera un fitxer anomenat "*execLoader.sh*" que és l'encarregat de llançar els ".ctl" a l'hora de realitzar les proves de rendiment.

```
# dargenerator version 3.0.08 Linux
rm *.log
rm *.bad
rm *.discard
exit | sqlplus -L -S adm_maestras/psw @commandsSQL.sql
sqlldr adm_maestras/psw control=0_d_idioma.ctl errors=1000 log=0_d_idioma.log
sqlldr adm_maestras/psw control=1_d_unidad_medida.ctl errors=1000 log=1_d_unidad_medida.log
sqlldr adm_maestras/psw control=2_d_unidad_medida_i18n.ctl errors=1000 log=2_d_unidad_medida_i18n.log
sqlldr adm_maestras/psw control=3_d_impuesto.ctl errors=1000 log=3_d_impuesto.log
sqlldr adm_maestras/psw control=4_d_tipo_item.ctl errors=1000 log=4_d_tipo_item.log
sqlldr adm_maestras/psw control=5_d_tipo_tercero.ctl errors=1000 log=5_d_tipo_tercero.log
sqlldr adm_maestras/psw control=6_d_tipo_organizacion.ctl errors=1000 log=6_d_tipo_organizacion.log
sqlldr adm_maestras/psw control=7_d_tipo_centro_negocio.ctl errors=1000 log=7_d_tipo_centro_negocio.log
sqlldr adm_maestras/psw control=8_d_tipo_empresa.ctl errors=1000 log=8_d_tipo_empresa.log
sqlldr adm_maestras/psw control=9_d_tercero.ctl errors=1000 log=9_d_tercero.log
sqlldr adm_maestras/psw control=10_d_organizacion.ctl errors=1000 log=10_d_organizacion.log
sqlldr adm_maestras/psw control=11_d_empresa.ctl errors=1000 log=11_d_empresa.log
sqlldr adm_maestras/psw control=12_d_tipo_impuesto.ctl errors=1000 log=12_d_tipo_impuesto.log
sqlldr adm_maestras/psw control=13_d_item.ctl errors=1000 log=13_d_item.log
sqlldr adm_maestras/psw control=14_d_producto.ctl errors=1000 log=14_d_producto.log
sqlldr adm_maestras/psw control=15_d_tipo_proveedor.ctl errors=1000 log=15_d_tipo_proveedor.log
sqlldr adm_maestras/psw control=16_d_proveedor.ctl errors=1000 log=16_d_proveedor.log
sqlldr adm_maestras/psw control=17_d_proveedor_pcb.ctl errors=1000 log=17_d_proveedor_pcb.log
exit | sqlplus -L -S adm_maestras/psw @postinserts_DAR.sql
exit | sqlplus -L -S adm_maestras/psw @statistics.sql
```

5.5.4.Buscador de peticions

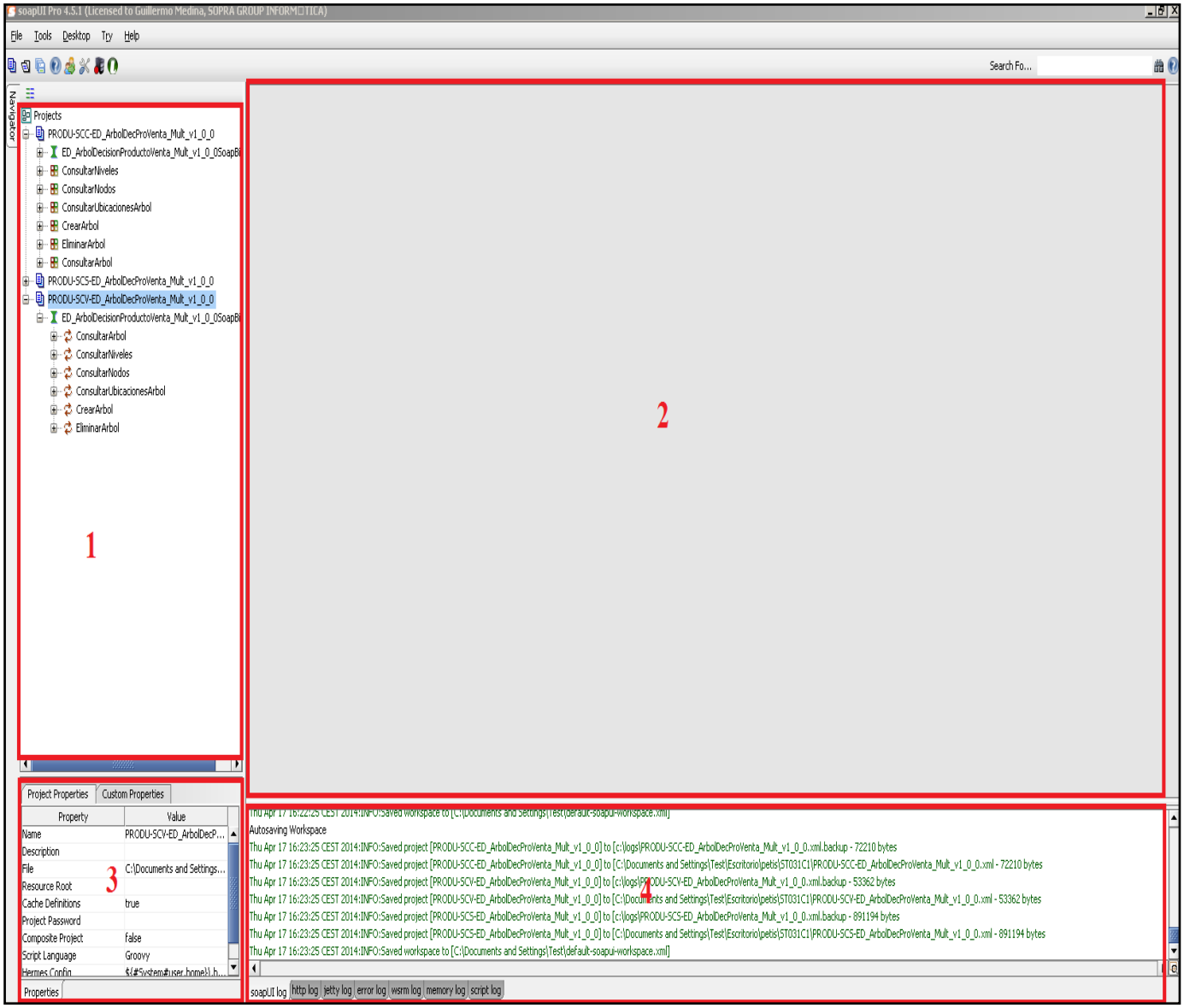
Aquesta ferramenta té una extensió ".bat" ja que està desenvolupada mitjançant una sèrie de comandaments DOS i te una extensió .baten el llenguatge .bat. El buscador de peticions s'utilitza per a realitzar recerques de les peticions mitjançant el seu codi. Aquesta ferramenta busca la petició que se li passa per paràmetre en el repositori de SVN i el descarrega en el lloc pertinent. Resulta de molta ajuda ja que s'automatitza, ja

no només la recerca, sinó que també la descàrrega en el lloc que li correspon en l'entorn local.

5.6.*SoapUI*

SoapUI Pro és una ferramenta de software lliure amb una interfície gràfica. Està basada en Java i serveix principalment per a provar *WebService*. La finestra de SoapUI esta dividida en les següents vistes:

1. Part esquerra: navegador de projectes.
2. Part dreta: regió de treball on s'obrin les visualitzacions seleccionades en el navegador.
3. Part inferior esquerra: el panell de propietats mostra la informació de l'objecte seleccionat en el navegador.
4. Part inferior dreta: mostra els diferents missatges del log del SoapUI.



Actualment l'automatització de les proves es realitza mitjançant aquesta ferramenta. La versió PRO permet, entre altres coses, l'execució de consultes sobre BBDD per automatitzar l'entrada o l'eixida de les nostres proves.

Els projectes de *SoapUI* són *scripts* en XML. Aquests scripts es guarden en el repositori de SVN per a que tot l'equip el tinga a la seua disposició.

Es disposa de tres nomenclatures distintes per als projectes de proves segons FdP:
- SCC: projecte *SoapUI* per a provar els casos de *Sanity check*.
- SCV: projecte que conté tots els altres casos de proves funcionals.
- SCS: s'utilitza per a realitzar les proves de seguretat. Es caracteritzen per contrindre "*Security Tests*".
- SCP: *script* amb les proves no funcionals de rendiment. Aquestes proves contenen "*Load tests*", que són *Steps* per executar proves de rendiment.

Aquesta ferramenta necessita dels conceptes de:
- *TestSuit*: normalment es crea un *TestSuit* per operació de la petició i se li anomena igual que la operació a provar. Serveix per contrindre els casos de prova (*TestCase*) que poden executar-se seqüencialment o en paral·lel.
- *TestCase*: es tracta del cas de prova a provar. Es crea un *TestCase* per cas de prova del PLP i se li anomena seguint la numeració del PLP. Serveix per a provar un cas específic de l'operació. Està compost per dos *TestStep* (*JDBC Request* i *Test Request*).
- *TestStep*: és tracta del pas a seguir dins d'un *TestCase*. En tractem dos en aquest cas: *JDBC Request*, es tracta de la connexió amb BBDD per tal d'obtindre les dades d'entrada del *WebService*, i *Test Request*, que conté la configuració dels paràmetres d'entrada obtinguts pel JDBC del *WebService*.

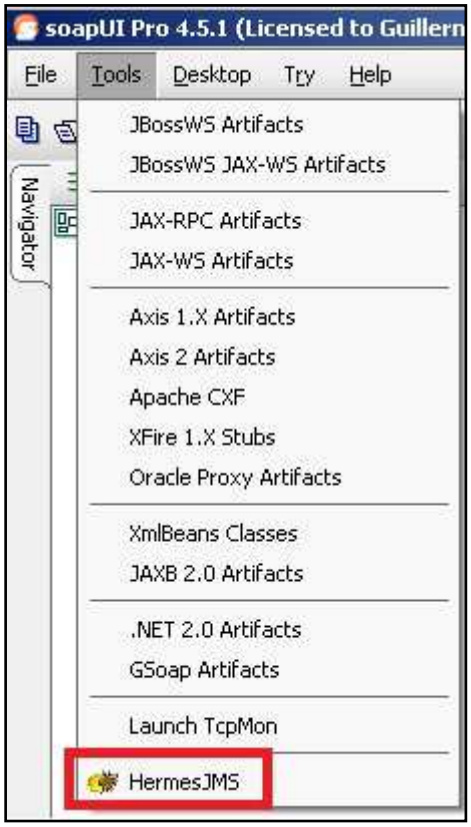
SoapUI resulta molt útil per a les proves sobre *WebService* perquè permet processar la informació obtinguda de cadascun dels *TestCase* per poder determinar el resultat de la prova.

Un projecte es compon de les següents parts:

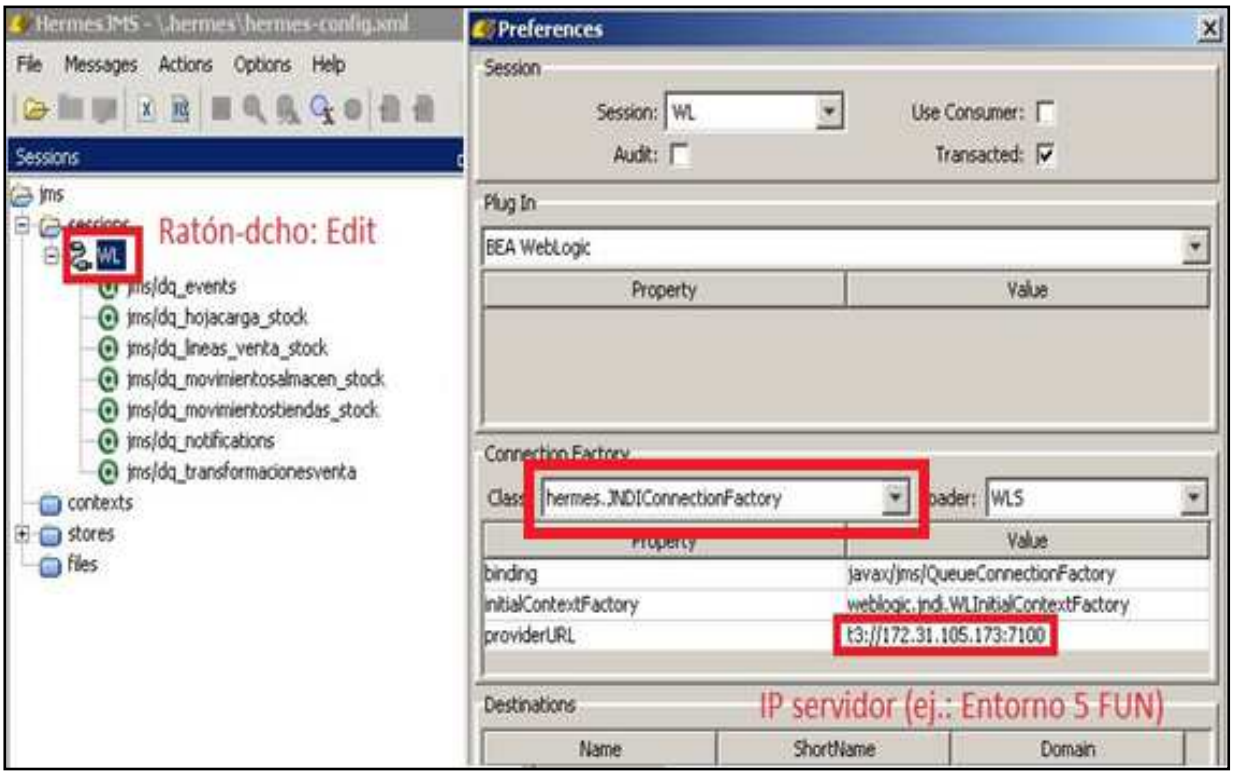
- *SoapBinding*. Conté el WSDL i els mètodes associats
- Test Suite 1
    - Test case 1
        - Step 1
        - Step 2
        - Step N
    - Test case 2
    - Test case N
- Test Suite 2
- Test Suite N

Com s'observa, poden existir tants *Steps* com siguen necessaris i cada *TestSuite* pot contindre tants *TestCases* com es considere apropiat. Principalment es van a provar WS, per tant, hi ha que saber els mètodes que té i quins són els paràmetres d'entrada de cadascun d'ells, aquesta informació ve donada pel WSDL que es converteix en el punt inicial del projecte.

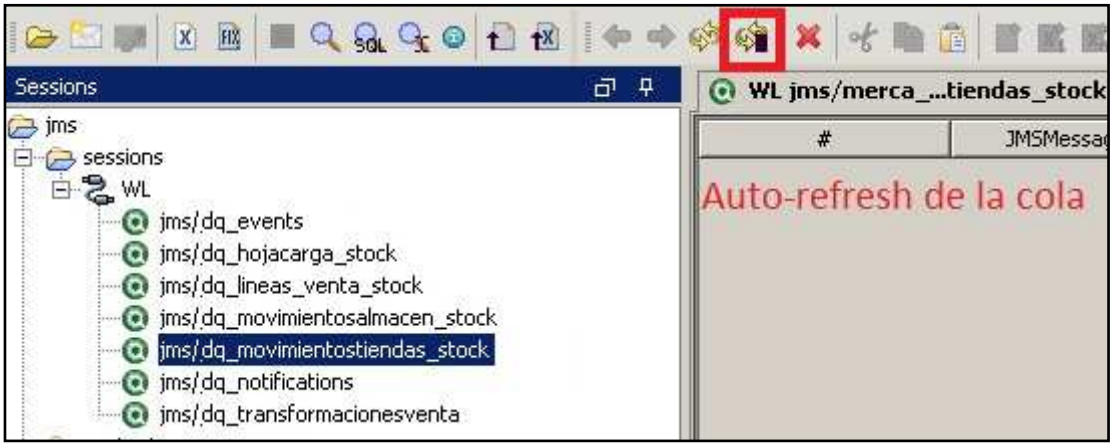
Una altra funcionalitat de *SoapUI* es la possibilitat de visualitzar variables de tipus cua o tòpic. El visor de cues s'anomena Hermes JMS i es pot obrir des del menú superior en la pestanya "Tools" i seleccionar l'última opció "HermesJMS":



Aquesta funcionalitat es utilitzada per veure els missatges enviats i rebuts mitjançant JMS (Java Message Service). Es disposa d'un fitxer xml que conté la configuració per utilitzar aquesta ferramenta amb els entorns locals de FdP. Aquesta configuració crea una connexió on tan sols s'ha de configurar el tipus de missatge JMS (cua o tòpic) i l'entorn de proves que es vol consultar per veure les cues. Per canviar aquests paràmetres ens situem en la connexió "WL" i premem el botó dret per obrir les opcions de preferències. Segons el tipus de missatge es configura de la següent manera:



Una vegada configurat es pot activar la recarrega automàtica de la cua per tal de quan es llance un TestCase automàticament la cua el mostra en la interfície del HermesJMS.S'activa des del menú d'opcions de la part superior, amb la cua seleccionada a autocarregar es prem:



Quan s'execute el TC de la petició en la part dreta de la ferramenta apareixen les dades que s'escriuen en la cua o el tòpic per poder determinar si el resultat és l'esperat.

5.7.*Filezilla*

Filezilla és un client FTP de codi obert baix els termes de la llicència pública general de GNU. Inicialment fou dissenyat per a treballar sobre *Windows*, però actualment es tracta d'una ferramenta multiplataforma. El codi font de la ferramenta es pot descarregar de la pàgina oficial.

La finestra principal es divideix en tres parts diferenciades: la consola, el directori local i el directori remot (falta remarcar les parts) .

La part de dalt de la finestra correspon a la consola on es mostren les ordres enviades al servidor remot i les respostes d'aquest. Per connectar-nos a una màquina remota utilitzem el botó remarcat en roig (falta remarcar el botó) on accedim al llistat de connexions ja configurats prèviament mitjançant un fitxer ".xml" i seleccionem la màquina a la qual ens volem connectar.

El directori local és la part esquerra de la finestra per davall de la consola. Proporciona una interfície gràfica que permet navegar per les carpetes i veure i alterar els continguts. La part dreta per davall de la consola correspon al directori remot que té la mateixa funcionalitat que el local.

Algunes alternatives a *Filezilla* són:

WinSCP és una aplicació de Software lliure. Es tracta d'una ferramenta gràfica d'un client SFTP per a Windows. La major part de la documentació està en anglès, però existeix una versió en castellà, es pot obtindre en la pàgina oficial i descarregar-se l'arxiu d'instal·lació de multillenguatge.

La interfície gràfica de la ferramenta es pot representar amb dos tipus, amb moltes configuracions personalitzades que permet un major nivell de funcionalitat o una més senzilla definida per defecte que és pareguda al Explorador de Windows.

Captain-ftp és un client FTP per a MAC OS que destaca per la gran varietat d'opcions disponibles, a més, permet tindre diverses tasques actives al mateix temps. Compta amb una interfície gràfica que facilita les tasques de l'usuari ja que és molt intuïtiva.

FdP decideix utilitzar *Filezilla* per homogeneïtzació amb el client.

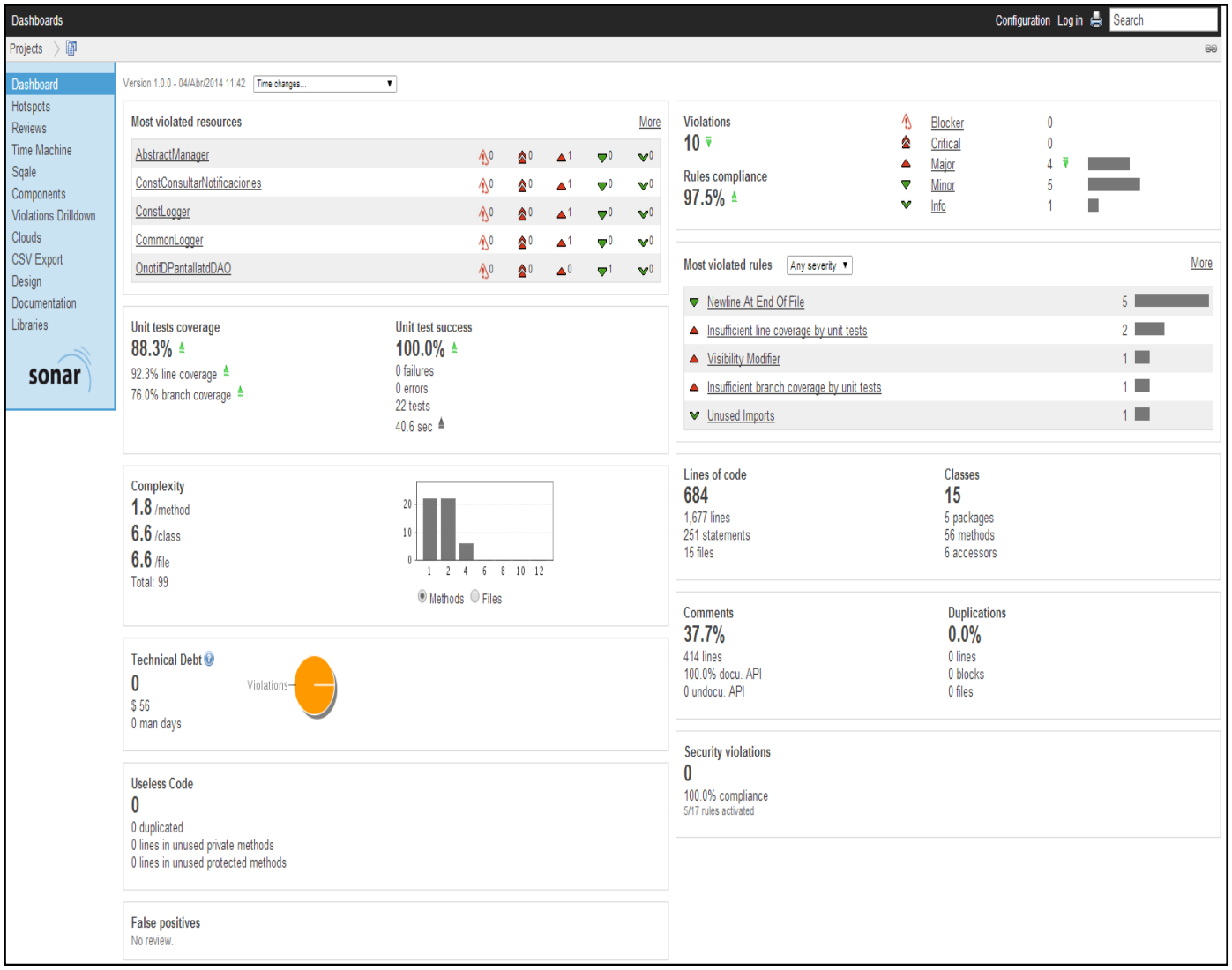
Aquesta ferramenta s'utilitza per a la visualització dels logs dels *WebService*. Sobretot, per a veure si quan es provoquen els errors segueixen la traça esperada o a l'hora d'utilitzar *mockups* veure que s'estan cridant de la forma correcta.

5.8.*SonarQube*

SonarQube, també anomenat Sonar, és una ferramenta que s'utilitza per realitzar anàlisi estàtic del codi font. A l'aplicar Sonar s'obtenen mètriques, que ajuden a millorar la qualitat del codi programat. Les funcionalitats utilitzades per extreure les mètriques en FdP per a les proves estàtiques de codi són: codi duplicat, estàndards de

codificació, proves unitàries, cobertura de codi, complexitat ciclomàtica, possibles errades, comentaris de codi i les regles pròpies.

Aquests resultats es poden veure en el servidor de Sonar una vegada s'ha executat la ferramenta pròpia *Statiqueitor*. Aquesta ferramenta és l'encarregada de compilar tots els fitxers corresponents a la petició sobre la qual està passant-se la prova i envia els resultats al servidor.



5.9.Yourkit Java Profiler

Yourkit és una ferramenta d'anàlisi exhaustiu de les aplicacions en memòria i en temps de CPU. Permet detectar en quins punts de l'aplicació s'utilitzen més recursos i d'aquesta manera poder dur a terme una millor estructuració del codi.

L'únic inconvenient d'aquesta ferramenta és que es de pagament, encara que existeixen llicències per a usos acadèmics.

6. Documents

6.1. Document tècnic d'alt nivell (DTAN)

El document més important de tota la factoria és el DTAN. En aquest document es defineixen les especificacions que deu acomplir el *WebService* a desenvolupar.

Aquest document és generat de forma externa a la factoria. És una entrada del procés global de factoria, és a dir, el client crea un nou DTAN, que al seu mateix temps genera una nova petició.

Es tracta d'un document que de vegades és molt extens (100 pàgines aproximadament), per això a mode d'exemple s'inclou només l'índex d'un DTAN:

ÍNDICE DE CONTENIDOS

Hi ha un apartat abans de la introducció que no apareix en l'índex, es tracta d'una taula on apareix una descripció dels canvis fets, les dates i el responsable en les diverses versions del DTAN.

La part de la introducció ajuda a la lectura del document amb un xicotet resum de l'objectiu i una taula de definicions i acrònims.

Seguidament es detallen les especificacions generals del *WebService*, com el tipus d'accés, la criticitat... En l'abast funcional del model de dades físic s'esmenten les taules del model de dades en les que treballa aquesta petició i en quin ordre es deuen carregar. El següent punt inclou les definicions del *WebService*, en aquest apartat s'inclou el domini d'informació o aplicació, perfil de seguretat, tipus de transport...

El punt "2.3.1", com diu el seu nom, són els requeriments no funcionals que es deuen assegurar quan es passen les proves no funcionals de rendiment.

El següent apartat és una especificació més detallada de cada operació de la petició, en aquest exemple només en té una. S'expliquen els requisits funcionals (nom, descripció, transaccionalitat, paginació, ordenació...), els paràmetres d'entrada, els d'eixida, el diagrama d'activitat, els comportaments principals i els alternatius. Un xicotet exemple seria:

REQ-FUN-0001 - Comportamiento principal	
Descripción	El WS realizará una consulta sobre la tabla D_PARAMETROS_PRODU y esta devolverá el valor del parámetro requerido.
Parámetros de entrada	Lista de campos que se ha especificado en los parámetros de entrada.
Parámetros de salida	txtValor, codError, desError.
Paso 1	Validar parámetros multilenguaje.
Paso 2	Validar el formato de datos de los parámetros de entrada
Paso 3	Se lanzara la consulta SELECT contra la tabla D_PARAMETROS_PRODU devolviendo el campo TXT_VALOR. En la clausula WHERE se buscará por el campo COD_V_PARAMETRO y con un = sobre el valor dado en **codParametro**. .
Paso 4	Finalización correcta de la consulta, concluirá en una devolución del parámetro de salida con la siguiente composición: TABLA: D_PARAMETROS_PRODU | ATRIBUTO TABLA: TXT_VALOR | PARÁMETRO SALIDA: txtValor A parte de txtValor, se devolverá NULL en los campos codError y desError en caso de terminar correctamente.

REQ-FUN-E001 - Comportamiento alternativo	
Descripción	Se produce un error parámetros multilenguaje, cuando alguno de los parámetros multilenguaje no cumple el rango de datos establecido.
Parámetros de entrada	• localeLanguage • localeCountry
Parámetros de salida	Lista de campos que se ha especificado en los parámetros de salida.
Error de salida	ErrorParametrosMultilenguaje
Paso 1.1	Si al validar el rango de datos de los parámetros multilenguaje es KO.
Paso 1.2	Escribir en traza el mensaje trazado descrito en el error.
Paso 1.3	Establecer codError y desError, con el código de error y mensaje amigable descrito en el error. El resto de parámetros de salida se establece a *null*.

REQ-FUN-E002 - Comportamiento alternativo	
Descripción	Se produce un error en la validación de los parámetros de entrada, cuando alguno de los parámetros de entrada no cumple su formato.
Parámetros de entrada	Los parámetros de entrada del WBS
Parámetros de salida	Lista de campos que se ha especificado en los parámetros de salida.
Error de salida	ErrorFormatoDatos

Els errors gestionats és el punt on es detallen amb més profunditat els errors que es defineixen en aquesta petició. De cadascun dels errors es detalla el nom, la descripció, el codi d'error, el missatge amigable (missatge visible que torna el *WebService* a l'usuari quan es dóna l'error) i el missatge traçat (el missatge que apareix en els logs de l'aplicació).

L'últim punt correspon a l'especificació, si hi ha, interacció entre més d'un servei.

6.2.Checklist (CH)

Les *check list* són uns documents *d'Excel* que conformen algunes proves estàtiques de codi. La seua funcionalitat és la de formar una plantilla per a utilitzar-la en les proves de revisió de documentació i d'entrega. Existeix un document d'aquest per cada versió de DTAN i per cada entrega de codi (modificació del WSDL o del DTD). Hi ha dos tipus de documents de CH, la revisió del DTAN i la revisió de l'entrega, però els dos segueixen la mateixa estructura: diversos punts que s'han de verificar del document.

La primera vegada que es passa aquesta prova en cadascun dels documents es deuen passar tots els punts de la *check list*. En cicles posteriors només es verifiquen els punts marcats com a "DOC-OBLIG" o els que en cicles anteriors no hagen acomplit les especificacions. Hi ha tres tipus de resultats possibles per a cada punt: si (S), no (N) i no aplica (X). S'ha implementat una plantilla de manera que compte automàticament els resultats obtinguts al llarg del document.

La següent imatge mostra una xicoteta ullada d'aquest tipus de document:

TOTALES NO CONFORMIDADES	
Bloqueante	0
Alta	1
Baja	0
	1

Numero	Tipo	Responsable verificación	Severidad	Area	CheckPoint	Pasa Test
CHP-TC004	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Todas las revisiones registradas tienen un identificador de versión asociado.	S
CHP-TC005	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Todas las revisiones registradas tienen un ID de petición asociado.	S
CHP-TC006	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Todas las revisiones registradas tienen una fecha asociada.	S
CHP-TC007	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Todas las revisiones registradas tienen una descripción asociada.	S
CHP-TC008	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Todas las revisiones registradas tienen un responsable de la modificación asociado.	S
CHP-TC009	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Hay secuencialidad entre los identificadores de las revisiones registradas.	S
CHP-TC010	DOC-OBLIG	Testeador	Alta	Historial de Cambios	Para cada una de las revisiones, excepto en la inicial, se especifican las modificaciones realizadas.	N
CHP-TC011	DOC-OBLIG	Testeador	Baja	Pie de página	La paginación debe estar actualizada.	S
CHP-TC012	DOC-OBLIG	Testeador	Alta	Referencias	Todas las referencias indicadas tienen un título asociado.	S
CHP-TC013	DOC-OBLIG	Testeador	Alta	Referencias	Todas las referencias indicadas tienen una versión asociada.	S
CHP-TC014	DOC-OBLIG	Testeador	Alta	Referencias	Todas las referencias indicadas tienen una ubicación asociada.	S
CHP-TC015	DOC-OBLIG	Testeador	Alta	Referencias	Existe la referencia a la versión de la plantilla de DTAN utilizado.	S
CHP-TC016	DOC	Testeador	Alta	Referencias	La versión de la plantilla de DTAN utilizada debe ser la última versión aprobada por Factoría: 2.6.2.	S
CHP-TC017	DOC-OBLIG	Testeador	Alta	Referencias	Existe la referencia al/los documento/s de Modelo de datos físico por dominio.	S
CHP-TC018	DOC	Testeador	Baja	Referencias	Si el servicio web llama a otro servicio web que tiene DTAN asociado, se incluye la referencia a dicho documento.	S
CHP-TC019	DOC-OBLIG	Testeador	Baja	Índice de Contenidos	El índice está actualizado.	S
CHP-TC020	DOC	Testeador	Baja	Introducción - Objetivo	El objetivo es un breve resumen del objetivo que persigue el WebService	S
CHP-TC021	DOC	Testeador	Baja	Introducción - Definiciones y acrónimos	Se incluyen todos los términos a los que se hace referencia a lo largo del documento.	S
CHP-TC022	DOC	Testeador	Baja	Introducción - Definiciones y acrónimos	No se incluyen términos técnicos ni propios del framework.	S
CHP-TC023	DOC	Testeador	Alta	Especificaciones del Servicio Web - Características generales	Se incluyen los campos: Aplicación asociada, Descripción, Tipo de acceso, Ámbito de uso, Patrón de utilización, Usuarios nominales, Grado de evolución y Criticidad.	S
CHP-TC024	DOC	Testeador	Baja	Especificaciones del Servicio Web - Características generales	Tipo de acceso es uno o varios de los siguientes: intranet, extranet, internet	S
CHP-TC025	DOC	Testeador	Baja	Especificaciones del Servicio Web - Características generales	El ámbito de uso es uno o varios de los siguientes: tiendas , almacenes, oficinas.	X
CHP-TC026	DOC	Testeador	Baja	Especificaciones del Servicio Web	Patrón de actualización puede ser: horario, diario, semanal o mensual.	S

6.3.Enfocament (ENF)

En aquest document es pretén plasmar la particularització de l'Estratègia de proves definit en FdP per al desenvolupament de la petició a provar. A continuació es presenta l'índex del document:

ÍNDICE DE CONTENIDOS

En cadascun d'aquests punts s'especifiquen les particularitats respecte a l'estratègia definida on, en cas contrari, les definicions per defecte per a les proves. La major part de les vegades aquests documents són iguals uns respecte als altres perquè es realitzen les mateixes proves en moltes peticions. Però de vegades hi ha alguna petició que necessita d'un enfocament diferent. El punt on poden haver canvis més significatius són: l'abast de les proves i els entorns. Hi ha peticions que no necessiten passar totes les proves o de vegades els entorns de prova poden ser diferents als usuals, aquests canvis solen ser per política del client.

Les següents imatges mostren un exemple d'aquest dos punts:

Tipo	Nombre	Se ejecuta
Estáticas	Revisión de la Entrega	✓
	Revisión del Cumplimiento de Estándares y de la Calidad del Código	✓
	Verificación de pruebas de caja blanca	✓
	Revisión de Documentación	✓
Funcionales	Pruebas de Versión	✓
	Pruebas de Regresión	✓
No funcionales	Pruebas de Seguridad	✓
	Pruebas de Rendimiento	✓
	Pruebas de Compatibilidad de servidores	✓
	Pruebas de Compatibilidad de navegadores	N/A
	Pruebas de Compatibilidad de dispositivos	N/A
	Pruebas de Localización	✗
	Pruebas de Usabilidad	N/A
	Pruebas de Accesibilidad	N/A

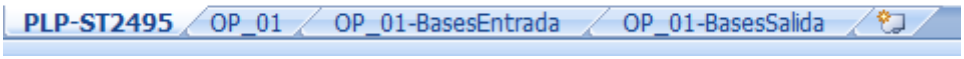
✓: Indica que la prueba es aplicable al tipo de componente y se va a ejecutar.
✗: Indica que la prueba es aplicable al tipo de componente pero no se va a ejecutar.

En aquesta es presenten les proves a executar per a una petició donada i en la següent on s'han d'executar aquestes proves.

Tipo	Nombre	Entorno
Estáticas	Revisión de la Entrega	Local FdP
	Revisión del Cumplimiento de Estándares y de la Calidad del Código	Local FdP
	Verificación de pruebas de caja blanca	Local FdP
	Revisión de Documentación	Local FdP
Funcionales	Pruebas de Versión	Integración FdP
	Pruebas de Regresión	Integración FdP
No funcionales	Pruebas de Seguridad	Integración FdP
	Pruebas de Rendimiento	Preproducción FdP

6.4.Pla de proves (PLP)

Aquest document està escrit en Excel. S'utilitza a nivell de factoria de proves per plasmar el pla de proves corresponent a cada petició. El PLP està format per diferents pàgines o pestanyes, tal com es mostra a continuació:



La primera correspon a la descripció de la petició, s'enumeren les operacions i les mètriques. Com a ajuda també s'inclouen les definicions dels camps de les pestanyes posteriors.

A continuació es defineixen tres pestanyes per operació:

- OP_01: on estan definits els cassos de prova corresponents al document PLP.
- OP_01-BasesEntrada: en aquesta pàgina es defineixen els paràmetres d'entrada de cada cas. Apareix tal que així:

Operación ConsultarParametrosPro

	localeLanguage	localeCountry	localeVariant	txtDescripcion
OP_03-BaseEntrada-01	es	ES	(null)	ST2495_Descripcion1001
OP_03-BaseEntrada-02	es	ES	(null)	ST2495_Descripcion1002
OP_03-BaseEntrada-03	es	ES	FdP	ST2495_Descripcion1003
OP_03-BaseEntrada-04	es	ES	(null)	cion140
OP_03-BaseEntrada-05	es	ES	(null)	(cadena vacia)
OP_03-BaseEntrada-06	es	ES	(null)	(3 espacios en blanco)
OP_03-BaseEntrada-07	es	ES	(null)	a
OP_03-BaseEntrada-08	es	ES	(null)	aaaaaaaaaaaaaaaaaaaaa
OP_03-BaseEntrada-ER	es	ES	(null)	ST2495_Descripcion1077

- OP_01-BasesSalida: i per últim la definició de les eixides esperades per a cadascun dels cassos de prova. La següent imatge mostra un exemple d'un full de les eixides esperades d'una operació:

Operación ConsultarParametrosProdu

	txtValor	codError	desError
OP_03-BaseSalida-01	ST2495_Valor1001	(null)	(null)
OP_03-BaseSalida-02	ST2495_Valor1102 ST2495_Valor1202	(null) (null)	(null) (null)
OP_03-BaseSalida-03	ST2495_Valor1003	(null)	(null)
OP_03-BaseSalida-04	ST2495_Valor1004	(null)	(null)
OP_03-BaseSalida-05	ST2495_Valor1005	(null)	(null)
OP_03-BaseSalida-06	ST2495_Valor1006	(null)	(null)
OP_03-BaseSalida-07	ST2495_Valor1007	(null)	(null)
OP_03-BaseSalida-08	ST2495_Valor1008	(null)	(null)
OP_03-BaseSalida-09	(null)	produ.common.error.parametrosMultilenguaje	Parámetros multilenguaje con rango incorrecto. [Parámetros: localeLanguage]
OP_03-BaseSalida-10	(null)	produ.common.error.parametrosMultilenguaje	Parámetros multilenguaje con rango incorrecto. [Parámetros: localeCountry]
OP_03-BaseSalida-11	(null)	produ.common.error.formatoDatos	Parámetros de entrada con formato incorrecto. [Parámetro: txtdescripcion]
OP_03-BaseSalida-12	(null)	produ.common.error.noRegistro	No existe registro en [Literales txtdescripcion] con el valor establecido [txtdescripcion: st2495_dESCRIPCION1077]
OP_03-BaseSalida-13	(null)	produ.common.error.noRegistro	No existe registro en [Literales txtdescripcion] con el valor establecido [txtdescripcion: ST2495_Descripcion1088]
OP_03-BaseSalida-14	(null)	produ.common.error.bbdd	Error ejecutando operación sobre Base de Datos.
OP_03-BaseSalida-15	(null)	produ.common.error.noControlado	Error no controlado.

Existeixen tres tipus de PLP segons el punt en el cicle de vida d'una petició en que ens trobem: *Sanity check*, el genèric i el complet. Cadascun d'ells es deu entregar en una fita determinada del cicle. Aquests tres tipus presenten unes certes diferències:

- (SC) NomAplicació-PLP-NomPetició: en aquest tipus només es defineixen els cassos de prova corresponents a la funcionalitat mínima que deu acomplir, els cassos de *sanity check*. Es defineixen les entrades i les eixides corresponents.

- (Generico) NomAplicació-PLP-NomPetició: document PLP que conté tots els cassos de prova que es desitgen executar. Però sense incloure els valors per a cada cas, ni les entrades i eixides, aquestes queden com en el PLP de *sanity check*.
- (Completo) NomAplicació-PLP-NomPetició: es tracta del PLP complet de la petició. S'inclouen tots els valors, les entrades i les eixides per a tots els cassos especificats.

A continuació s'inclouen les imatges de cadascun dels tipus de PLP per a que es puga apreciar l'evolució d'aquest al llarg del cicle de vida de la petició. La primera imatge mostra un PLP de *sanity check*, la següent un genèric i per acabar una imatge d'un PLP complet:

Operación ConsultarParamet

Leyenda Tipo de prueba

	Funcionales
	No Func. - Seguridad
	No Func. - Rendimiento
	Estáticas

Leyenda Valor

[] Dato pendiente de informar
() Información para el testeador
XXXX Valor definitivo

							ENTRADA			SALIDA			EJECUCIÓN	
							Base	Parámetro Funcional Interesante		Resultado esperado	Base de salida	Comprobaciones adicionales	Resultado obtenido	Comentarios
REQUERIMIENT	TC ID	Objetivo		Descripción	Precondiciones	Postcondiciones		Parámetro	Condició / Valor					
REQ-FUN-0001	OP_01-F-001	SC	Funcional	Se consulta un único registro.	Cargar juego de datos mediante el fichero (SC) PRODU-DSC-ED_Parametros_Produ_v1_0 0.sql	Eliminar juego de datos mediante el fichero (SC) PRODU-DMA-.sql	OP_01-BaseEntrada 01	(todos los parámetros son de interés)		OK	OP_01-BaseSalida 01		---	

							ENTRADA				SALIDA			EJECUCIÓN	
							Base	Parámetro Funcional Interesante			Resultado esperado	Base de salida	Comprobaciones adicionales	Resultado obtenido	Comentarios
REQUERIMIENTO	TC ID	Objeti		Descripción	Precondiciones	Postcondiciones		Parámetro	Condición	Valor					
REQ-FUN-0001	OP_01-F-001	SC	Funcional	Se consulta un único registro.	Cargar juego de datos mediante el fichero (Generico) PRODU-DSC-ED_Parametros_Produ_v1	Eliminar juego de datos mediante el fichero (Generico) PRODU-DMA-ED_Parametros_Produ_v	OP_01-BaseEntrada-01	(todos los parámetros son de interés)			OK	OP_01-BaseSalida-01		----	
REQ-FUN-0001	OP_01-F-002	VER	Funcional	Se consulta un único registro usando todos los parámetros.	Cargar juego de datos mediante el fichero (Generico) PRODU-DAT-ED_Parametros_Produ_v1	Eliminar juego de datos mediante el fichero (Generico) PRODU-DMA-ED_Parametros_Produ_v	OP_01-BaseEntrada-02	(todos los parámetros son de interés)			OK	OP_01-BaseSalida-02		----	
REQ-FUN-0001	OP_01-F-003	VER	Validación	Se informa con la longitud mínima los parámetros de tipo string.	Cargar juego de datos mediante el fichero (Generico) PRODU-DAT-ED_Parametros_Produ_v1	Eliminar juego de datos mediante el fichero (Generico) PRODU-DMA-ED_Parametros_Produ_v	OP_01-BaseEntrada-03	Parámetros String	(longitud mínima)	[Informado]	OK	OP_01-BaseSalida-03		----	
REQ-FUN-0001	OP_01-F-004	VER	Validación	Se informa con la longitud máxima los parámetros de tipo string.	Cargar juego de datos mediante el fichero (Generico) PRODU-DAT-ED_Parametros_Produ_v1	Eliminar juego de datos mediante el fichero (Generico) PRODU-DMA-ED_Parametros_Produ_v	OP_01-BaseEntrada-04	Parámetros String	(longitud máxima)	[Informado]	OK	OP_01-BaseSalida-04		----	
REQ-FUN-E001	OP_01-F-005	VER	Validación - Erroneo	Se informa con formato incorrecto el parámetro 'localeLanguage'.	Cargar juego de datos mediante el fichero (Generico) PRODU-DAT-ED_Parametros_Produ_v1	Eliminar juego de datos mediante el fichero (Generico) PRODU-DMA-ED_Parametros_Produ_v	OP_01-BaseEntrada-EE	localeLanguage	(formato incorrecto)	[Informado]	ErrorParametrosMultilenguaje	OP_01-BaseSalida-05		----	

							ENTRADA				SALIDA			EJECUCIÓN	
							Base	Parámetro Funcional Interesante			Resultado esperado	Base de salida	Comprobaciones adicionales	Resultado obtenido	Comentarios
REQUERIMIENTO	TC ID	Objeti		Descripción	Precondiciones	Postcondiciones		Parámetro	Condición	Valor					
REQ-FUN-0001	OP_01-F-001	SC	Funcional	Se consulta un único registro.	Cargar juego de datos mediante el fichero PRODU-DSC-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-01	(todos los parámetros son de interés)			OK	OP_01-BaseSalida-01		----	
REQ-FUN-0001	OP_01-F-002	VER	Funcional	Se consultan dos o más registros usando un único elemento en las listas de entrada.	Cargar juego de datos mediante el fichero PRODU-DAT-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-02	(todos los parámetros son de interés)			OK	OP_01-BaseSalida-02		----	
REQ-FUN-0001	OP_01-F-003	VER	Funcional	Se consulta un único registro usando todos los parámetros.	Cargar juego de datos mediante el fichero PRODU-DAT-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-03	(todos los parámetros son de interés)			OK	OP_01-BaseSalida-03		----	
REQ-FUN-0001	OP_01-F-004	VER	Funcional	Se informa con subcadenas de los valores en BBDD los parámetros de tipo string (LIKE en el WHERE)	Cargar juego de datos mediante el fichero PRODU-DAT-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-04	Parámetros String	(informado con subcadena de BBDD 'ST2495_Descripcion1404')	cion140	OK	OP_01-BaseSalida-04		----	
REQ-FUN-0001	OP_01-F-005	VER	Validación	Se informa con cadena vacía los parámetros de tipo string/char.	Cargar juego de datos mediante el fichero PRODU-DAT-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-05	Parámetros String/Char	(cadena vacía)	(cadena vacía)	OK	OP_01-BaseSalida-05		----	
REQ-FUN-0001	OP_01-F-006	VER	Validación	Se informa con espacios en blanco los parámetros de tipo string/char.	Cargar juego de datos mediante el fichero PRODU-DAT-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-06	Parámetros String/Char	(espacios en blanco)	(3 espacios en blanco)	OK	OP_01-BaseSalida-06		----	
REQ-FUN-0001	OP_01-F-007	VER	Validación	Se informa con la longitud mínima los parámetros de tipo string.	Cargar juego de datos mediante el fichero PRODU-DAT-ED_Parametros_Produ_v1_	Eliminar juego de datos mediante el fichero PRODU-DMA-ED_Parametros_Produ_v1_	OP_01-BaseEntrada-07	Parámetros String	(longitud mínima)	a	OK	OP_01-BaseSalida-07		----	

Com es pot veure en les últimes imatges l'única diferència entre un genèric i d'un complet és que en el primer els valors de les proves no estan informats.

6.5.Document tècnic de Disseny (DTD)

El document tècnic de desenvolupament inclou la descripció per part de la factoria de desenvolupament del codi a implementar per a una petició donada. Es tracta d'un document paregut al DTAN però més detallat quan al codi i la funcionalitat. El WSDL forma part de l'entrega amb conjunt amb el DTD.

L'índex del DTD és el següent:

2 **INDICE**

Com es pot veure segueix la mateixa estructura que el DTAN, amb una introducció a l'inici, un desenvolupament més detallat de les operacions i per acabar uns annexos corresponents al tractament d'errors de la petició.

El WSDL és un document amb llenguatge XML que s'utilitza per a descriure el *WebService*. En aquest document es descriu la interfície pública dels serveis, la comunicació, llista amb els paràmetres i els formats de la petició. La següent imatge mostra un exemple:

```
<xsd:complexType name="Output_ConsultarParametrosProdu">
<xsd:sequence>
 <xsd:element minOccurs="0" name="codError" nillable="true" type="xsd:string" />
 <xsd:element minOccurs="0" name="desError" nillable="true" type="xsd:string" />
 <xsd:element minOccurs="0" name="errorList" nillable="true" type="ns0:ErrorList" />
 <xsd:element minOccurs="0" name="serviceContext" nillable="true" type="ns0:ServiceContext" />
 <xsd:element minOccurs="0" name="txtValor" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
 <xsd:element name="ConsultarParametrosProdu" type="tns:ConsultarParametrosProdu" />
<xsd:complexType name="ConsultarParametrosProdu">
<xsd:sequence>
 <xsd:element minOccurs="0" name="input" type="tns:Input_ConsultarParametrosProdu" />
</xsd:sequence>
</xsd:complexType>
 <xsd:element name="ConsultarParametrosProduResponse" type="tns:ConsultarParametrosProduResponse" />
<xsd:complexType name="ConsultarParametrosProduResponse">
<xsd:sequence>
 <xsd:element minOccurs="0" name="out" type="tns:Output_ConsultarParametrosProdu" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

6.6.Índex d'entrega QA Proves (INE)

El document INE és una guia que serveix per a revisar que tota la documentació a entregar al client està en el directori de SVN, dit d'una altra manera, que totes les proves han sigut dutes a terme. S'utilitza Excel per a representar, mitjançant una taula, l'índex d'entrega. La primera columna correspon al nom del document, la segona a la ubicació i la tercera una xicoteta descripció, tal com mostra la imatge:

Indice Entrega QA Pruebas

Plantilla de ejecución: Plantilla - Indice Entrega 0_3

[$ruta Entrega QA Pruebas] = c_PRODU_doc/1-Definir/3-Disenar/3.04-ModeloDeImplementacion/ED_Parametros_Produ/v1_0_0

[$ruta Entrega Software] = c_PRODU_online_src/

FICHERO	UBICACIÓN ENTREGA CLEARCASE	DESCRIPCIÓN
PRODU-INE-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/	Indice entrega
PRODU-DTA-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/DOC/	Diseño Técnico Alto Nivel
PRODU-ENF-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/DOC/	Enfoque de pruebas
PRODU-PLP-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/DOC/	Plan de pruebas
PRODU-DTD-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/DOC/	Diseño Técnico Detallado
ED_Parametros_Produ_v1_0_0.xml	[$ruta Entrega QA Pruebas]/DOC/	Contrato WSDL
Código Fuente WS	[$ruta Entrega Software]	Elementos de Desarrollo (Junit, Código Fuente, Librerías etc)
PRODU-INP-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/DOC/	Informe Final
Anexos: Sistemas y BDD	[$ruta Entrega QA Pruebas]/DOC/01.Anexos	Anexos de BDD, Sistemas etc,.
Anexos: SoapUI - PRODU-ICA-ED_Parametros_Produ_v1_0_0 - PRODU-IST-ED_Parametros_Produ_v1_0_0 - PRODU-IEC-ED_Parametros_Produ_v1_0_0 - PRODU-IET-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/DOC/01.Anexos	Anexos de SoapUI
PRODU-GER-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/	Guía de Tester para reproducir ejecución de pruebas Rendimiento
PRODU-GEP-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/	Guía de Tester para reproducir ejecución de pruebas.
PRODU-DAT-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/01.Juego de Datos	Juego de Datos de pruebas
PRODU-DSC-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/01.Juego de Datos	Juego de Datos de pruebas de sanity-check
PRODU-DMA-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/01.Juego de Datos	Marcha atrás del Juego de Datos de pruebas
PRODU-ETF-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/01.Juego de Datos	Datos entrada SoapUI para pruebas funcionales
PRODU-DAR-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/01.Juego de Datos	Juego de Datos de pruebas para rendimiento
PRODU-DMR-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/01.Juego de Datos	Marcha atrás del Juego de Datos de pruebas rendimiento. (OBSOLETO)
PRODU-SCC-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/02.Scripts de Pruebas	Proyecto SoapUI Sanity Check
PRODU-SCV-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/02.Scripts de Pruebas	Proyecto SoapUI Version (solo existente en el Ciclo 1)
PRODU-SCR-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/02.Scripts de Pruebas	Proyecto SoapUI Regresión (existente a partir del Ciclo 2)
PRODU-SCS-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/02.Scripts de Pruebas	Proyecto SoapUI Seguridad
PRODU-SCP-ED_Parametros_Produ_v1_0_0	[$ruta Entrega QA Pruebas]/TEST/02.Scripts de Pruebas	Proyecto SoapUI Rendimiento

6.7.Índex d'enviament de petició (IND)

En aquest document es presenten els *JUnit* proporcionats per la factoria de desenvolupament i la seua ubicació.

S'utilitza per a realitzar les proves estàtiques de codi, en concret l'utilitza el *Statiqueitor*. Aquest document és utilitzat per *Statiqueitor* quan es duu a terme la compilació. Inclou tots el *JUnit* a executar en l'apartat d'inclusions del document sonar.xml i les exclusions corresponents.

A continuació es presenta un exemple del contingut del document d'índex de control d'entrega:

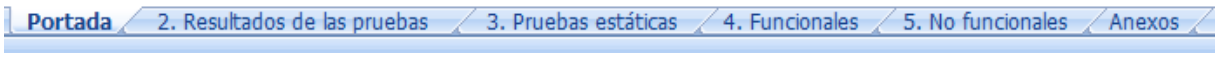
Control Entrega

COMPONENTES ENTREGADOS	FECHA ENTREGA	NUEVO / MODIFICAD	DESCRIPCIÓN	DEPENDENCIAS	MODIFICACIÓN REALIZADA	PETICIÓN	INCIDENCIA	OBSERVACIONES
lib/app/ear/ws-produ-1.4.0.jar	07/05/2014	NUEVO	Librería	(ninguna)		ST2495		
lib/app/test/h2-1.3.169/h2-1.3.169.jar	07/05/2014	NUEVO	Librería	(ninguna)		ST2495		
src/business/es/mercadona/common/business/AbstractErrorMsgValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/AbstractManager.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/common/business/DateValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/DoubleValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/ExtendedError.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/common/business/InjectionSqlValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/InjectionXssValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/RangeNumberValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/RangeValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/RequiredValidator.java	07/05/2014	NUEVO	Validador	(ninguna)		ST2495		
src/business/es/mercadona/common/business/ValidatorDelegate.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/common/business/package-info.java	07/05/2014	NUEVO	Fichero de documentación	(ninguna)		ST2495		
src/business/es/mercadona/common/log/CommonLogger.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/common/log/ConstLogger.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/common/log/package-info.java	07/05/2014	NUEVO	Fichero de documentación	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/exception/ErrorFormatoDatosException.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/exception/ErrorNoRegistrosException.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/exception/ErrorParametrosMultilenguajeException.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/exception/package-info.java	07/05/2014	NUEVO	Fichero de documentación	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/ConstConsultarParametrosProdu.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/ConsultarParametrosProduManager.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		
src/business/es/mercadona/produ/edparametrosprodu/business/package-info.java	07/05/2014	NUEVO	Fichero de documentación	(ninguna)		ST2495		
src/dao/es/mercadona/produ/edparametrosprodu/dao/impl/DParaproduConsuDAOImpl.java	07/05/2014	NUEVO	Fichero de código fuente	(ninguna)		ST2495		

6.8.Informe final de proves (INP)

L'informe final de proves és un document on es recullen tots els resultats del conjunt de proves executat sobre una petició concreta. Per representar-lo s'utilitza la ferramenta *d'Excel*, ja que incorpora més facilitats per a treballar amb taules.

Aquest document es reparteix en diverses pestanyes. La primera correspon a una introducció a la petició i una taula amb conceptes que ajuden a la lectura, l'última a annexos que es necessiten que acompanyen a la petició i les altres corresponen a fulls per als resultats de cadascuna de les proves:



El full dos inclou un resum de tots els resultats de les proves, així com propietats associades a elles (dates, *baseline*, entorns...). La següent imatge mostra una xicoteta ullada d'aquesta pàgina:

Resultado final de las pruebas:

FUN			NFUN				ST	
SC	VER	REG	REN	SEG	LOC	COM	COD	DOC
✓	✓	N/A	✋	✓	N/A	✓	✓	✓

Tabla 2.1 Resultado final de las pruebas

✓: Indica que se han resuelto todos los defectos detectados.
!: Indica que hay defectos de severidad media o baja pendientes de resolver.
x: Indica hay defectos de severidad alta o bloqueante pendientes de resolver.
N/A: Indica que ese tipo de pruebas no se ejecutan.

Resultados por ciclos de ejecución de pruebas:

			Ciclo 1	Ciclo 2	Ciclo 3
Etiquetado (BaseLine)			bl_produ_online_v0.10 05.62		
Fecha Inicio			09/05/2014		
Fecha Fin			09/05/2014		
Tipología pruebas ejecutadas	ST	ENT	✓		
		COD	✓		
		VCB	✓		
	FUN	SC	✓		
		VER	✓		
		REG	N/A		
	NFUN	REN	No ejecutado		
		SEG	✓		
		LOC	N/A		
		COM	✓		

Tabla 2.2 Resultados por ciclos de ejecución de pruebas

Per exemple, mostra les proves executades i les que no apliquen a aquesta petició. A continuació es presenta una taula on s'especifiquen els resultats (si són correctes o no) de les proves executades. També s'acompanyen d'una *baseline* associada al conjunt de proves i les dades d'inici i fi de les proves.

La següent pestanya correspon al resultat de les proves estàtiques. En aquest full es deuen informar els resultats de les proves de revisió de documentació, les de l'entrega les del compliment d'estàndard i les proves de qualitat de codi i les de verificació de caixa blanca. La següent imatge mostra un exemple de les taules a informar en aquestes proves:

3.1. Revisión de documentación

3.1.1. Resultados de la revisión del DTAN

[Cuando el número de revisiones es menor o igual a 3, incluir la siguiente tabla]

		Revisión 1	Revisión 2	
Versión del DTAN revisada		1.4	x.y	
Checkpoints Evaluados		0	0	
Dudas Detectadas	*Bloqueante*	0	0	
	Alta	0	0	
	Media			
	Baja	0	0	
Dudas Detectadas en el ciclo anterior que quedan Pendientes de resolución.	*Bloqueante*		0	
	Alta		0	
	Media			
	Baja		0	

Tabla 3.1 Resultados DTAN

[Cuando el número de revisiones es mayor a 3]

		Resumen
Total de Versiones de DTAN Revisadas		0
Total de Checkpoints Evaluados		146
Total de Dudas Detectadas	*Bloqueante*	0
	Alta	0
	Media	
	Baja	0
Dudas Pendientes de Resolución tras el Último Ciclo	*Bloqueante*	0
	Alta	0
	Media	
	Baja	0

Tabla 3.1 Resultados DTAN

Dudas pendientes de resolver:

No existen dudas pendientes de resolución.

Com es pot veure s'inclou la versió del document a revisar, el nombre de validacions realitzades i ,si hi ha, el nombre de defectes trobats i el seu nivell grau. Aquest exemple mostra en concret els resultats d'una prova de revisió de documentació sobre el document del DTAN. La taula de revisió del WSDL i de l'entrega són molt similars a aquesta. En la taula corresponent a les proves executades mitjançant la ferramenta Sonar s'inclouen també els llindars que es deuen acomplir. S'inclouen per a facilitar l' interpretació dels resultats de forma més ràpida. A continuació un exemple:

3.2. Auditoría de cumplimiento de estándares y pruebas de calidad de código

3.2.1. Resultados de las pruebas

Tipología de No-conformidad	Umbral de aceptación	Ciclo 1	Ciclo 2	Ciclo 3
Violaciones Bloqueantes	0	0	0	0
Violaciones Críticas	0	0	0	0
% Reglas Sonar cumplidas	90%	98%	0%	0%
% Mínimo de Comentarios	30%	37%	0%	0%
% Código Duplicado	5%	0%	0%	0%
Número de métodos de Complejidad Ciclomática > 10	0	0	0	0
% Código Muerto	0%	0%	0%	0%
Severidad del defecto reportado		No genera defecto		

Tabla 3.7 Resultados Auditoría de estándares y pruebas de calidad de código

Elementos excluidos del análisis:

Elemento	Motivo
BaseTestCase.java	Clase abstracta
package-info.java	No modela comportamiento

Tabla 3.8 Detalle elementos excluidos

3.2.2. Comentarios adicionales

Sin comentarios adicionales.

3.3. Verificación de pruebas de caja blanca

3.3.1. Resultados de las pruebas

Tipología de No-conformidad	Umbral de aceptación	Ciclo 1	Ciclo 2	Ciclo 3
% Cobertura Sentencias	85%	88%	0%	0%
% Éxito Pruebas Unitarias	100%	100%	0%	0%
Severidad del defecto reportado		No genera defecto		

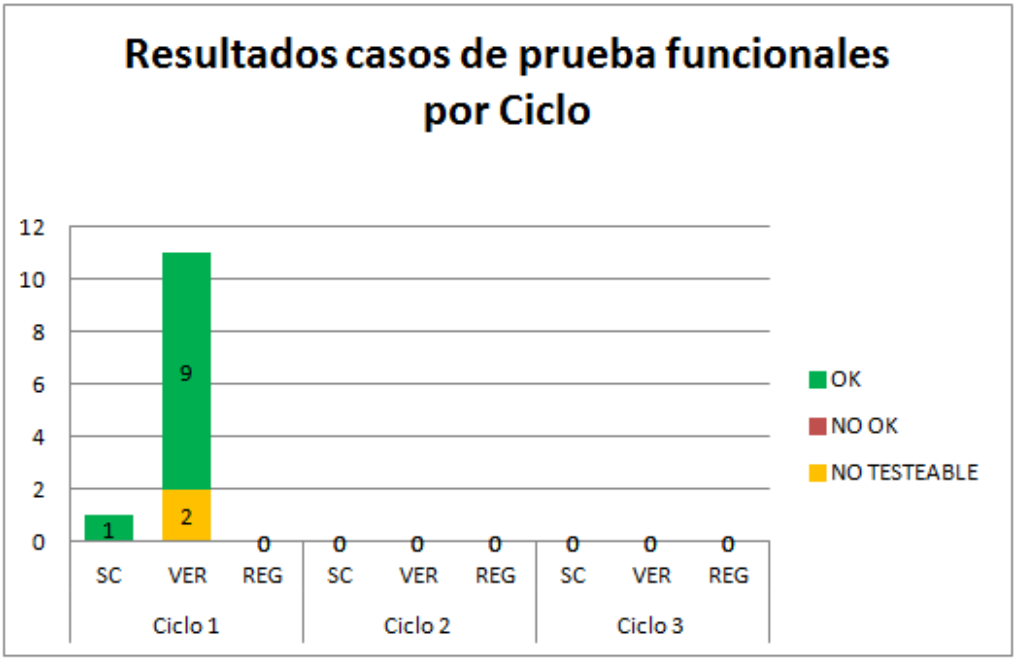
Tabla 3.10 Detalle violaciones detectadas

El full número quatre correspon als resultats de les proves funcionals. Les taules segueixen l'estructura mostrada en les proves estàtiques. Però, a banda, s'inclouen gràfiques per veure de forma més ràpida els resultats:

		Ciclo 1			Ciclo 2			Ciclo 3		
		SC	*VER*	*REG*	*SC*	*VER*	*REG*	*SC*	*VER*	*REG*
Casos de prueba	*OK*	1	9	0						
	NO OK	0	0	0						
	NO TESTEABLE	0	2	0						
Defectos reportados	*Bloqueante*	0	0	0						
	Alta	0	0	0						
	Media	0	0	0						
	Baja	0	0	0						

Tabla 4.1 Resultados de las pruebas funcionales

En la gráfica siguiente se indican los resultados de los casos de prueba ejecutados en cada ciclo:



Gráfica 4.2 Resultados por ciclo

L'última pestanya correspon als resultats de les proves no funcionals. En aquest full s'inclouen primer els resultats de les proves no funcionals de seguretat i a continuació els de rendiment. La següent imatge mostra un exemple de com s'informen les proves de rendiment:

A continuación se plasma el detalle de cada uno de los ciclos de pruebas de rendimiento donde:

CC: Conexiones Concurrentes.
TMP: Duración de la prueba en segundos.
TMR: Tiempo Medio de Respuesta en milisegundos.
TPE: Total Peticiones WS.
PER: Nº Peticiones erróneas.
RAT: Ratio error peticiones WS[1]
CPU: Comportamiento CPU[2]

ICPU: Nº Conexiones concurrentes a partir de las cuales la CPU se vuelve inestable[3]
RAM: Comportamiento RAM[2]
IRAM: Nº Conexiones concurrentes a partir de las cuales la RAM se vuelve inestable[3]

[1] Se indica el dato en Rojo en caso de prueba no superada.
[2] Se indican los símbolos ✗ o ✓ dependiendo del resultado obtenido.
[3] Si aplica, se indica el valor. Si no, N/A.

Detalle Ciclo 1

		Tipos de prueba			
Operación	**Metrica**	*Carga*	*Estrés*	*Estabilidad*	*Escalabilidad*
ConsultarParametros Produ	*CC*	10,00	12,00	0,00	0,00
	TMP	300,00	300,00	0,00	0,00
	TMR	40,75	41,80	0,00	0,00
	TPE	8.624,00	51.162,00	0,00	0,00
	PER	0,00	0,00	0,00	0,00
	RAT	0,00	0,00	0,00	0,00
	CPU	✓	✓	✓	✓
	ICPU	✓	✓	✓	✓
	RAM	✓	✓	✓	✓
	IRAM	✓	✓	✓	✓

Com es pot veure en les anteriors imatges tots els documents de factoria estan en forma de plantilla per estalviar temps i per ajudar a fer més homogènia tota la documentació.

7. Conclusió

Les conclusions que resulten d'aquest treball sobre una factoria de *software* són molt positives si s'analitzen des del punt de vista d'una empresa. Afegir una factoria de proves suposa una millora de la qualitat del codi optés. A banda de la visió del cicle de desenvolupament se n'ha de tenir una altra més crítica. El procés d'avaluació d'un *WebService* es deu fer des d'un punt de vista crític. Les proves han de seguir un enfocament basat en el risc per equilibrar l'esforç de les proves (intentar abastir el màxim comportament possible) amb les conseqüències d'una fallada de codi. S'ha de tenir en compte que mai es deu provar el *software* en un entorn de producció. És necessari realitzar les proves en un entorn físicament separat. Existeixen ferramentes per emular els entorns de producció. El conjunt de proves que s'han definit en FdP permet descobrir errors en totes les fases del cicle de desenvolupament.

En el model de negoci que representa una factoria s'han de tenir cura a l'hora de gestionar correctament tot el procés. Si es centra la gestió en els terminis i el lliurament de fites el desig d'arribar a conclusions ràpidament pot embrutir les proves, ja que la pressió se sent més a donar respostes en lloc de fer preguntes. També s'ha d'evitar la temptació de conduir les proves amb l'objectiu d'oferir "la resposta simple" que les parts interessades esperen, és a dir, eludir les conclusions ràpides.

Els beneficis que podem extreure d'aquest treball són molts. El primer és l'estandarització de tota la documentació relacionada amb la factoria, aquest factor també influeix amb el manteniment del codi. Un altre avantatge d'estandarditzar la documentació és que resulta més fàcil l'aprenentatge dels nous empleats que es puguen afegir a la factoria.

Les proves realitzades en factoria també ajuden a la detecció primerenca d'errors del codi. A banda d'estalviar econòmicament, ja que una correcció d'un error primerenc resulta menys car que d'un en fases més tardanes. També ajuda a millorar la funcionalitat del *software* perquè es poden corregir els punts dèbils.

Un altre benefici del *testing* es la reducció del cost de manteniment del codi gràcies a que aquest es regeix per uns estàndards. Les prestacions de les aplicacions també es coneixen, ja que en la factoria de proves s'han dut a terme proves de rendiment posant al límit les aplicacions. D'aquesta manera es pot prevenir quan el sistema informàtic es quedarà curt quant a recursos. No obstant, el més significatiu és la rapidesa i agilitat en que es realitzen canvis sobre un codi que ja ha passat per la factoria de proves.

Per últim, facilita la integració dels components que s'han provat unitàriament en la factoria de proves, ja que s'assegura que abans d'arribar a aquest punt, els *WebService* acomplien la funcionalitat desitjada.

Degut a tots aquests beneficis el resultat del codi que passa per la factoria de proves és un codi amb major qualitat, ja no només quant a la funcionalitat dels components sinó que també en les línies del codi.

A banda dels beneficis referits al *software*, aquesta tècnica també estalvia econòmicament a llarg termini degut a què redueix el cost del manteniment.

La visió del *testing* no està molt estesa. Però degut a la informatització massiva en molts tipus d'indústria no serà d'estranyar que cada vegada s'utilitze més la tècnica del *testing* per millorar els productes *software* des de les fases més primerenques del cicle de desenvolupament de software fins la fase de producció.

8. Acrònims

Concepte	Definició/descripció
Factoria de Software	Entitat dedicada al desenvolupament de codi
Factoria de proves	Composada pel proveïdor encarregat de certificar la qualitat dels *WebService* dins de factoria.
Factoria de desenvolupament	Proveïdor encarregat de desenvolupar els *WebService*.
Cas de prova	Especifica una forma de provar un component, incloent l'entrada de la prova, el resultat esperat i les condicions en que ha de provar-se.
Pla de proves	Conjunt de casos de prova. Existeixen tres tipus.
Pla de proves Sanity Check	Especifica el mínim nombre de TC per a comprovar la funcionalitat de l'operació. Si aquest cas falla serà bloquejant i fins que no es solucione no es podrà dur a terme res més. S'utilitza per a que FdD puga "assegurar" que la mínima funcionalitat va bé.
Pla de proves Genèric	Aquest pla de proves ja és més complet, conté tot el conjunt de cassos de prova que seran duts a terme per FdP. Els valors de les dades que s'utilitzaran en les proves no s'informa ([informado] en el PLP).
Pla de proves Complet	Es tracta del pla de proves al complet. S'entregarà al client al finalitzar el cicle de la petició. En aquest PLP si que estan totes les dades informades.
Error	Acció realitzada per un ser humà que produeix un resultat incorrecte.
Defecte	El resultat d'un error en el codi o en un document que pot provocar una desviació respecte al funcionament esperat.
Fallada	Desviació respecte al funcionament esperat.
Dubte	No conformitat o estat d'incertesa respecte a un funcionament específic.
Baseline	Etiqueta que se li dóna a un codi entregat per FdD.
Falsos positius	La detecció d'un defecte quan no hi ha, provocat per un error humà o un falsejament de les proves.

Terme	Definició/descripció
FUN	proves FUNcionals
NFUN	proves No FUNcionals
TC	Test Case
REN	proves no funcionals de RENdiment
SC	Sanity Check
PLP	PLa de Proves
FdP	Factoria de Proves
FdD	Factoria de Desenvolupament
DTAN	Document Tècnic d'Alt Nivell
DAT	Joc de dades de prova
DSC	Joc de dades de proves de sanity-check
DMA	Marxa enrere de joc de dades de proves
ETF	Dades d'entrada SoapUI per a proves funcionals
ETR	dades d'entrada SoapUI per a proves de rendiment
DTD	Document Tècnic de Desenvolupament
SCC	projecte SoapUI Sanity-check
SCV	projecte SoapUI Versió
SCS	Projecte SoapUI Seguretat
SCP	projecte SoapUI rendiment
ICA	generat per SoapUI en proves de càrrega
IST	informe generat per SoapUI d'estrés
INP	Informe final de proves
QA	proves índex de peticions i ubicació en l'estructura
WS	WebService
BBDD	Base de dades

IND	Índex de control d'entrega
WSDL	Web Services Description Language
INE	Índex d'entrega QA Proves

9. Bibliografia

Atlassian. *Issue & Project Tracking Software | Atlassian* . <https://www.atlassian.com/software/jira>[Consulta: 26 de juny de 2014]

IBM Software. *IBM-Rational Quest*. <http://www-03.ibm.com/software/products/es/clearquest> [Consulta: 26 de juny de 2014]

IEEE Advancing Technology for Humanity. *IEEE - The world's largest professional association for the advancement of technology*. <http://www.ieee.org/> [Consulta: 26 de juny de 2014]

ISTQB Incorporated society. *Certifying Software Testers Worldwide - ISTQB® International Software Testing Qualifications Board*. <http://istqb.org/> [Consulta: 26 de juny de 2014]

Oracle Corporation. *MySQL :: The world's most popular open source database*. <http://www.mysql.com/>[Consulta: 26 de juny de 2014]

SonarSource. *SonarQube*. <http://www.sonarqube.org/> [Consulta: 26 de juny de 2014]

The Apache Software Foundation. *Apache Subversion*. <http://subversion.apache.org/>[Consulta: 26 de juny de 2014]

Yourkit. *Java Profiler - .NET Profiler - The profilers for Java and .NET professionals*. <http://www.yourkit.com/> [Consulta: 26 de juny de 2014]