Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Validazione & Testing del Middleware per e-business Grid

Sabrina Celia, Danilo Cistaro,
Giuseppe Papuzzo

RT-ICAR-CS-11-07 **Ottobre 2011**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*

Progetto SFIDA-PMI

Soluzioni informatiche per Filiere, Distretti ed Associazioni di PMI

Prot. MIUR n.446/ICT - Tema 1 "SFIDA-PMI"

R2.2.4
Validazione & Testing del Middleware per e-business Grid

Deliverable numero :	R2.2.4
Titolo:	Validazione & Testing del Middleware per e-business Grid
OR / Task :	OR2 / T2.2.1, T2.2.2, T2.2.3
Data consegna :	31/12/2008
Versione documento:	1.0
Autore(i):	- Dipartimento di informatica università di Pisa (UNIPI) - ICAR-CNR Responsabile: Ing. Giandomenico Spezzano

Revisioni

Rev.	***Data***	***Autori***	***Variazioni***

Indice

Indice delle Figure

No table of figures entries found.

Indice delle Tabelle

1 Executive Summary

Il presente documento illustra i risultati delle attività di validazione e testing del middleware per e-business Grid. Le attività si collocano nell'ambito dell'Obiettivo Realizzativo OR2 ("Piattaforma abilitante ICT di Grid Computing orientata ai processi") che si propone di fornire una piattaforma abilitante di e-business Grid che consenta di sviluppare software complesso per supportare gli strumenti e ambienti di gestione dei processi di e-business collaborativo. In questo contesto, è stato sviluppato il framework Sunflower per la gestione di workflow autonomici che è basato sull'adozione di una piattaforma multi-agente P2P che abilita il sistema ad auto-proteggersi, ricoverare da guasti, auto-riconfigurarsi in base alle modifiche dell'ambiente, e mantenere il suo funzionamento il più possibile vicino alla performance ottimale.

La fase di testing presentata di seguito nel documento è stato condotta testando le principali funzionalità del dimostratore realizzato. Le funzionalità analizzate riguardano:

- l'utilizzo di strumenti disponibili commercialmente per la creazione di nuovi workflow e la verifica che il codice XML generato sia compatibile con il framework Sunflower;
- la gestione dinamica della piattaforma Sunflower attraverso l'aggiunta di nuovi nodi;
- la registrazione di Web/Grid service appartenenti ad una semplice ontologia di dominio;
- l'uso di un'interfaccia grafica per la gestione dell'ontologia di dominio;
- la gestione dell'upload dei workflow;
- il deployment dei workflow;
- il monitoraggio dell'esecuzione di workflow.

L'attività di validazione è stata invece condotta per misurare le performance di alcuni aspetti caratterizzanti il sistema Sunflower. In particolare, sono stati misurati i tempi necessari per:

- la decomposizione del workflow in frammenti in modo da poterne valutare la complessità;
- l'esecuzione di workflow attraverso un motore tradizionale (Oracle BPEL) e il framework Sunflower in modo da poterli confrontare;
- l'esecuzione di un workflow con recovery multipli in caso di guasti;
- la reattività del framework in presenza di variazione dei parametri di QoS.

Il documento riporta i risultati delle prove effettuate e commenta le misure di performance.

2 Ambiente di Validazione e Testing

L'ambiente utilizzato per il testing e la validazione del framework Sunflower è costituito da una rete di Pc ripartiti in due classi distinte in modo da poter analizzare il funzionamento nel caso più generale in cui sono presenti sia nodi di Griglia e sia nodi Internet.

2.1 Riassunto Funzionalità

Sunflower è un framework per la gestione dinamica di workflow basato sull'adozione di una piattaforma multi-agente e di meccanismi autonomici che abilitano il sistema ad auto-proteggersi, ricoverare da guasti, auto-riconfigurarsi in base alle modifiche dell'ambiente, e mantenere il suo funzionamento il più possibile vicino alla performance ottimale.

Sunflower è caratterizzato da:

- una piattaforma P2P con meccanismi bio-ispirati per la gestione del sistema informativo di Griglia in grado di scoprire, selezionare e allocare i servizi necessari per l'esecuzione di workflow;
- l'adozione di meccanismi self-organizing per il coordinamento decentralizzato di agenti che implementano in maniera distribuita l'enactement del workflow.
- l'utilizzo di tecniche di migrazione degli agenti per il deployment dinamico del workflow tenendo conto dei parametri di QoS specificati dagli utenti;
- l'integrazione nel WMS di meccanismi autonomici per garantire le proprietà di self-healing e self-tuning, self-configuration durante l'esecuzione di un workflow.
-

Di seguito sono riportate le principali funzionalità testate e validate:

ID	Nome Funzionalità	Descrizione
f.3.1.1	Possibilità di creare un nuovo workflow e di modellare i componenti base del linguaggio BPEL	L'utente deve essere in grado di creare un workflow BPEL e deve avere a disposizione un'interfaccia grafica che gli permetta di modellare tutti gli elementi costituenti del linguaggio BPEL: attività strutturate e primitive
f.3.1.2	Possibilità di aggiungere una macchina host alla piattaforma Sunflower, attraverso l'installazione del Sunflower Distribuite Engine	L'utente deve avere a disposizione un'interfaccia grafica che gli permetta di aggiungere al framework Sunflower nuove macchine host sulle quali verranno eseguiti i frammenti di workflow

f.3.1.3	Possibilità di registrare un Grid/Web Service, legandolo a un Peer JXTA e a un'ontologia di dominio	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di registrare i Grid/Web service che ha creato, inoltre deve essere in grado di legarlo al relativo servizio astratto dell'ontologia di dominio e al Peer JXTA che risiede sulla stessa macchina host su cui si trova il servizio
f.3.1.4	Possibilità di creare, modellare e gestire una semplice ontologia di dominio relativa a Grid/Web Services	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di poter creare modella e gestire un'ontologia di dominio
f.3.1.5	Possibilità di upload dei workflow creati	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di caricare nella Sunflower Console i workflow creati
f.3.1.6	Possibilità di selezione, inizializzazione ed esecuzione di un workflow	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di selezionare un workflow e successivamente di inizializzare i parametri richiesti per l'esecuzione in coreografia
f.3.1.7	Possibilità di monitoraggio dell'esecuzione decentralizzata e visualizzazione del risultato	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di monitorare l'esecuzione dell'istanza di workflow e successivamente di poter visualizzare il risultato dell'elaborazione
f.3.1.8	Valutazione dei tempi di scomposizione di un workflow BPEL in frammenti	Misurare i tempi impiegati dall'algoritmo di scomposizione del workflow BPEL per valutarne in modo empirico l'ordine di complessità
f.3.1.9	Confronto dei tempi di esecuzione tra Oracle BPEL (orchestrazione) e Sunflower (coreografia)	Confrontare, a parità di condizioni, il tempo impiegato dai due motori di workflow ad eseguire lo stesso codice BPEL
f.3.1.10	Misura del tempo di esecuzione di un workflow con Recovery multiplo	Valutare il tempo supplementare necessario all'esecuzione di un workflow in caso di Recovey
f.3.1.11	Verifica dell'adattività di SunFLOWer alla variazione della QoS	Valutare il meccanismo di Routing per la sostituzione soft di un servizio con un equivalente durante l'esecuzione di un workflow, il tutto guidato da strategie basate sulla QoS

2.2 Lo scenario di Validazione

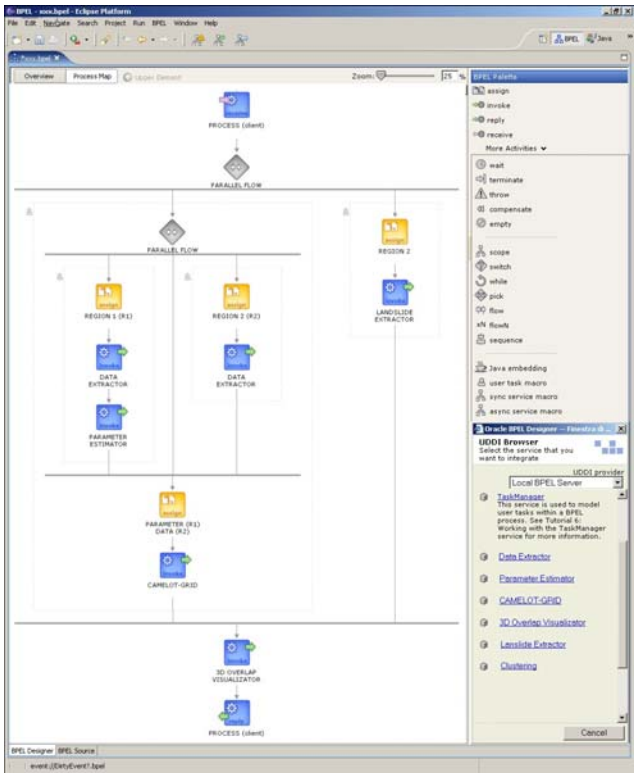
Per la validazione si è utilizzata una rete di PC su cui è stata installata la piattaforma Sunflower necessaria per l'esecuzione di workflow autonomici. Su ognuno dei nodi sono stati predisposti i Web e i Grid service necessari per lo scenario di validazione. Alcuni dei servizi sono stati replicati e sono stati simulati alcuni guasti per testare la capacità di auto-riconfigurazione di Sunflower.

Su uno dei nodi della rete è stata installata la Sunflower Console necessaria per l'avvio dell'esecuzione dei workflow e per la raccolta dei risultati. La fase di testing è stata condotta da un utente attraverso l'utilizzo della Web GUI messa a disposizione dalla Sunflower Console.

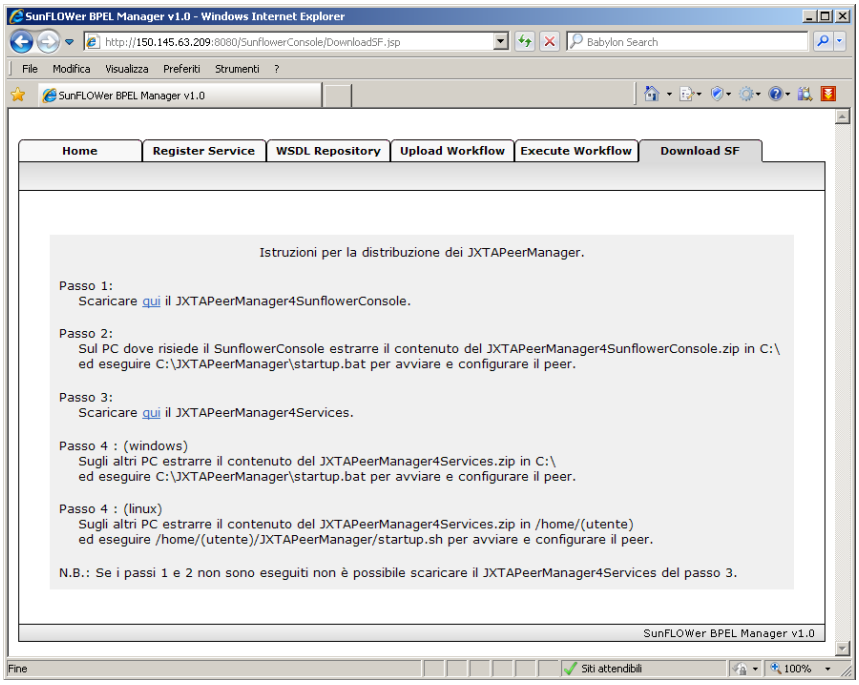
3 Validazione delle Funzionalità

Nel seguito per ognuna delle funzionalità di Sunflower definita nel paragrafo 2.1 è presentata una scheda con l'indicazione degli obiettivi da raggiungere, il set-up necessario per avviare il testing delle funzionalità e i risultati finali rappresentati attraverso screenshot grafici che mostrano i risultati del testing e la corretta esecuzione.

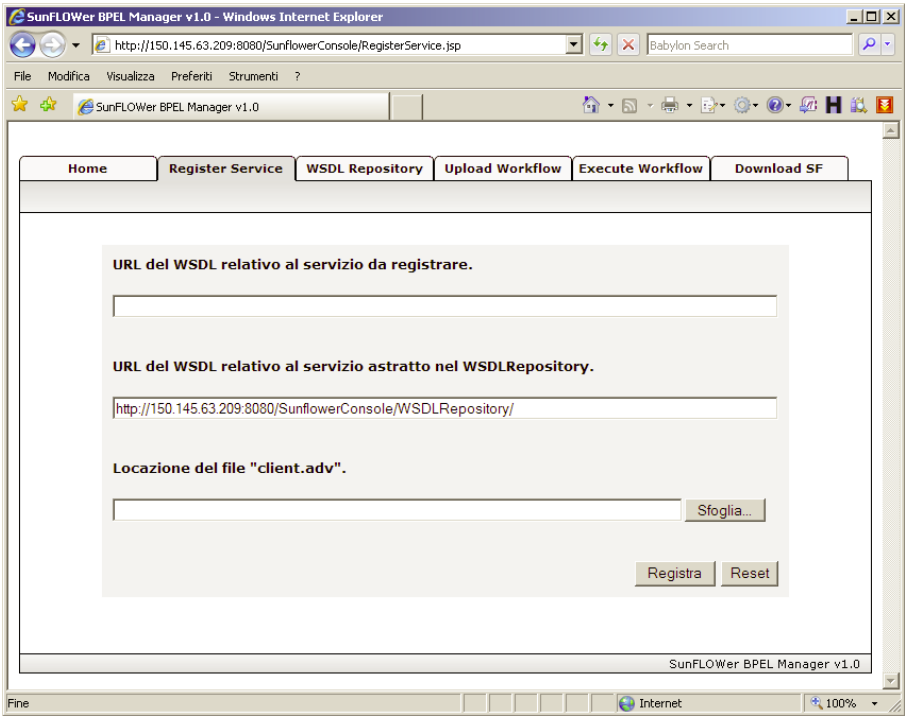
3.1.1 Test Funzionalità f3.1.1

TITOLO	Possibilità di creare un nuovo workflow e di modellare i componenti base del linguaggio BPEL
OBIETTIVO	L'utente deve essere in grado di creare un workflow BPEL e deve avere a disposizione un'interfaccia grafica che gli permetta di modellare tutti gli elementi costituenti del linguaggio BPEL: attività strutturate e primitive
SET-UP (input)	Avere a disposizione: Eclipse e Oracle BPEL Designer
STEP ESEGUITI	Apro Oracle BPEL Designer Creo o carico un worflow Modello le attività strutturate e primitive del linguaggio BPEL 
RISULTATI (output)	Il workflow che ha superato la fase di validazione è pronto per l'upload verso la Sunflower Console

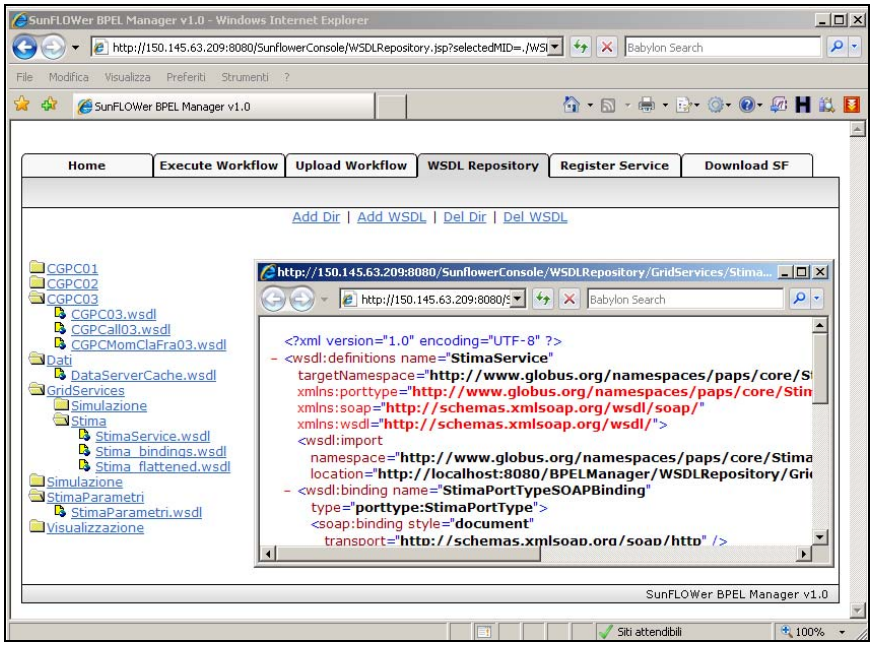
3.1.2 Test Funzionalità f3.1.2

TITOLO	Possibilità di aggiungere una macchina host alla piattaforma Sunflower, attraverso l'installazione del Sunflower Distribuite Engine
OBIETTIVO	L'utente deve avere a disposizione un'interfaccia grafica che gli permetta di aggiungere al framework Sunflower nuove macchine host sulle quali verranno eseguiti i frammenti di workflow
SET-UP (input)	Aver creato almeno un workflow (vedi test precedente), aver installato la Sunflower Console (vedi manuale d'uso) e avere a disposizione un browser web
STEP ESEGUITI	Apro nel browser la pagina predisposta al download della piattaforma per l'esecuzione in coreografia del wokflow ed eseguo le istruzioni in essa contenute 
RISULTATI (output)	Un'altra macchina host è stata aggiunta alla piattaforma Sunflower per l'esecuzione decentralizzata di workflow BPEL

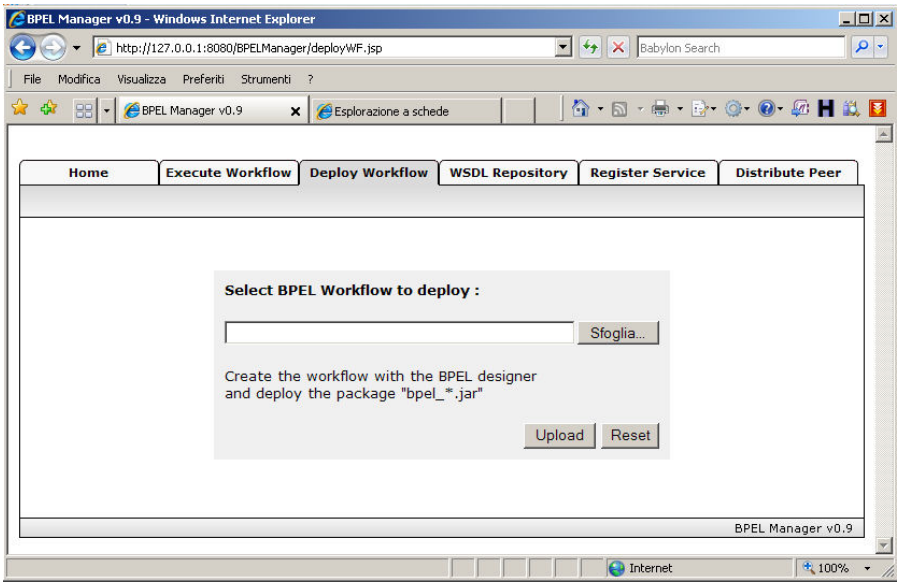
3.1.3 Test Funzionalità f3.1.3

TITOLO	Possibilità di registrare un Grid/Web Service, legandolo a un Peer JXTA e a un'ontologia di dominio
OBIETTIVO	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di registrare i Grid/Web service che ha creato, inoltre deve essere in grado di legarlo al relativo servizio astratto dell'ontologia di dominio e al Peer JXTA che risiede sulla stessa macchina host su cui si trova il servizio
SET-UP (input)	Aver creato almeno un Grid/Web service, aver selezionato un servizio astratto nell'ontologia di domino (vedi test successivo) e aver installato il Peer JXTA sulla macchina su cui risiede il servizio (vedi test precedente)
STEP ESEGUITI	Apro nel browser la pagina predisposta alla registrazione dei servizi e riempio i campi con i dati richiesti 
RISULTATI (output)	Un'altro servizio è legato alla piattaforma Sunflower e la sua QoS può essere monitorata costantemente attraverso le funzioni fornite dal PeerJXTA

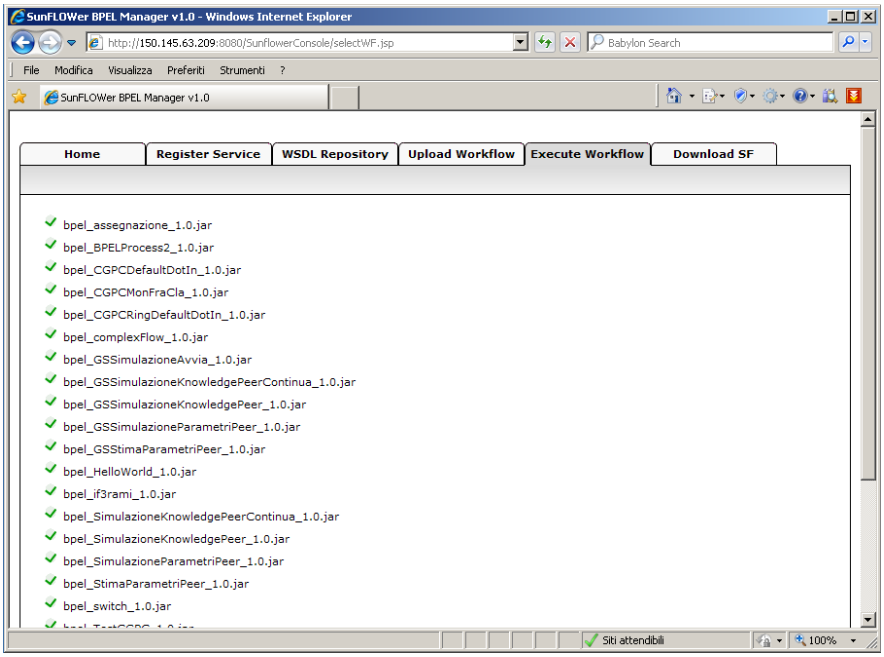
3.1.4 Test Funzionalità f.3.1.4

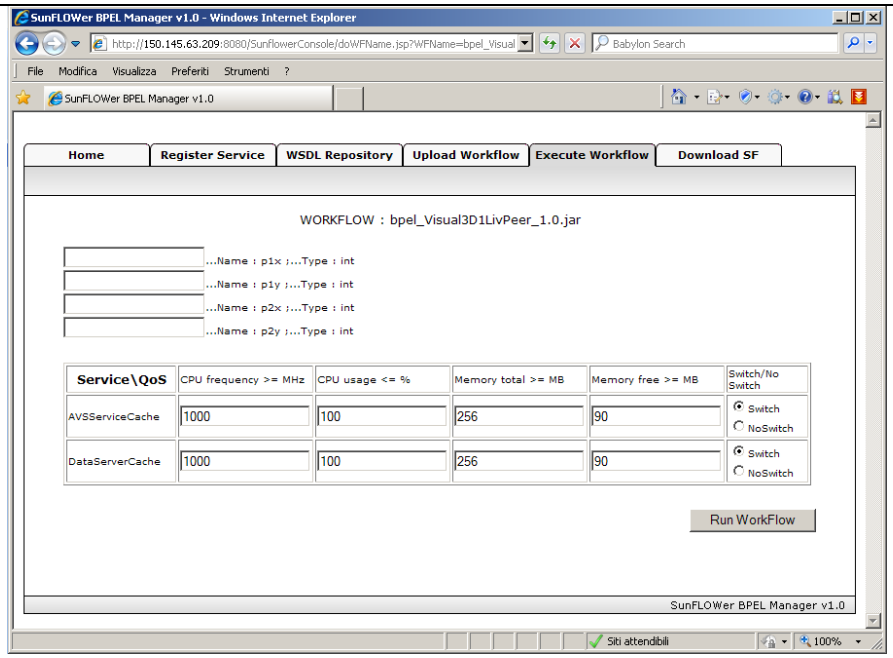
TITOLO	Possibilità di creare, modellare e gestire una semplice ontologia di dominio relativa a Grid/Web Services
OBIETTIVO	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di poter creare modella e gestire un'ontologia di dominio
SET-UP (input)	Aver creato il WSDL relativo a un servizio che si vuole implementare o avere a disposizione il WSDL di servizio già creato
STEP ESEGUITI	Apro nel browser la pagina predisposta alla gestione dell'ontologia di dominio e uso i seguenti metodi: • "Add Dir", aggiunge un ramo di ontologia nella posizione corrente. • "Add WSDL", aggiunge una foglia (servizio astratto o reale) nella posizione corrente • "Del Dir", cancella il ramo di ontologia corrente. • "Del WSDL", cancella la foglia (servizio astratto) corrente. 
RISULTATI (output)	Un'ontologia di dominio che organizza WSDL astratti relativi a Grid/Web Services

3.1.5 Test Funzionalità f3.1.5

TITOLO	Possibilità di upload dei workflow creati
OBIETTIVO	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di caricare nella Sunflower Console i workflow creati
SET-UP (input)	Aver creato almeno un workflow BPEL basato sui a Grid/Web Services astratti nell'ontologia di dominio
STEP ESEGUITI	Apro nel browser la pagina predisposta al deploy del workflow e carico il file contenete Il workflow BPEL 
RISULTATI (output)	Un'altro worflow BPEL è pronto per l'esecuzione nella scheda "Execute Workflow " della Sunflower Console (vedi test successivo)

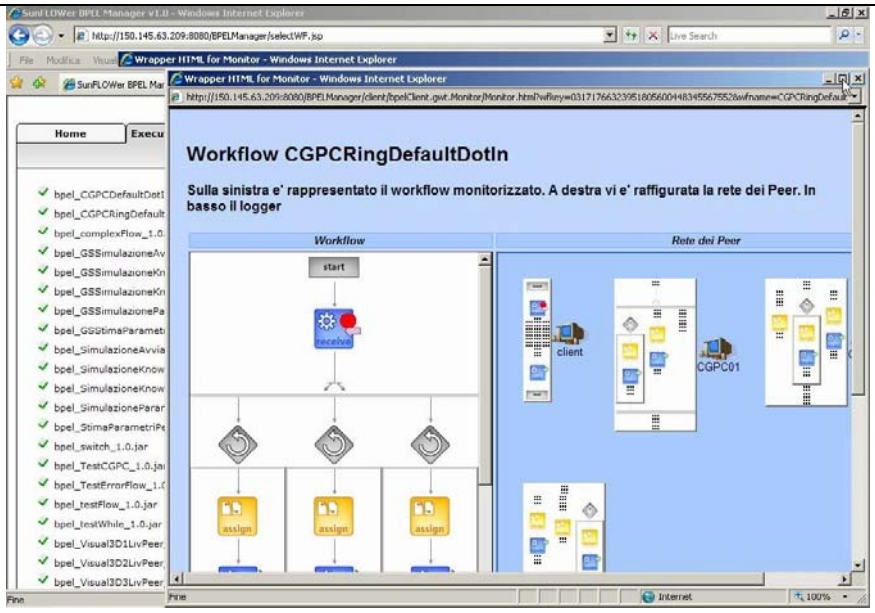
3.1.6 Test Funzionalità f3.1.6

TITOLO	Possibilità di selezione, inizializzazione ed esecuzione di un workflow
OBIETTIVO	L'utente deve avere a disposizione un'interfaccia grafica che gli consenta di selezionare un workflow e successivamente di inizializzare i parametri richiesti per l'esecuzione in coreografia
SET-UP (input)	Nessuno
STEP ESEGUITI	Apro nel browser la pagina "Execute Workflow" e seleziono un workflow BPEL tra quelli disponibili  Inizializzo attraverso la successiva form i parametri di ingresso del workflow e impongo la QoS per singolo servizio

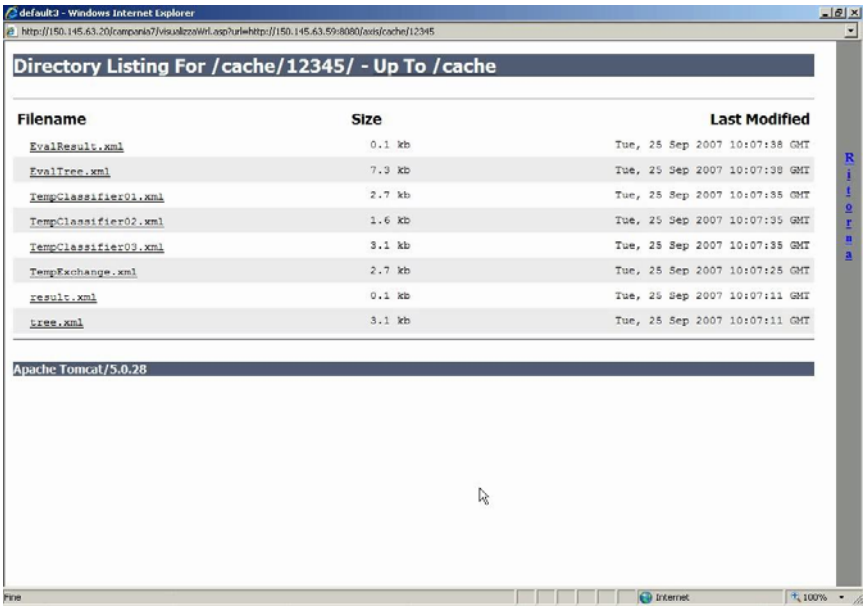
	 Avvio il workflow cliccando su “Run Workflow”
RISULTATI (output)	L’istanza del worflow prescelto è stata inizializzata e mandata in esecuzione. L’esecuzione può essere seguita attraverso la finestra di monitoring a pop-up che appare (vedi test successivo)

3.1.7 Test Funzionalità f3.1.7

TITOLO	Possibilità di monitoraggio dell’esecuzione decentralizzata e visualizzazione del risultato
OBIETTIVO	L’utente deve avere a disposizione un’interfaccia grafica che gli consenta di monitorare l’esecuzione dell’istanza di workflow e successivamente di poter visualizzare il risultato dell’elaborazione
SET-UP (input)	Nessuno
STEP ESEGUITI	Nella finestra di pop-up, che appare all’avvio dell’esecuzione di un workflow, si può seguire l’esecuzione sui vari BPEL Engine dei frammenti di workflow

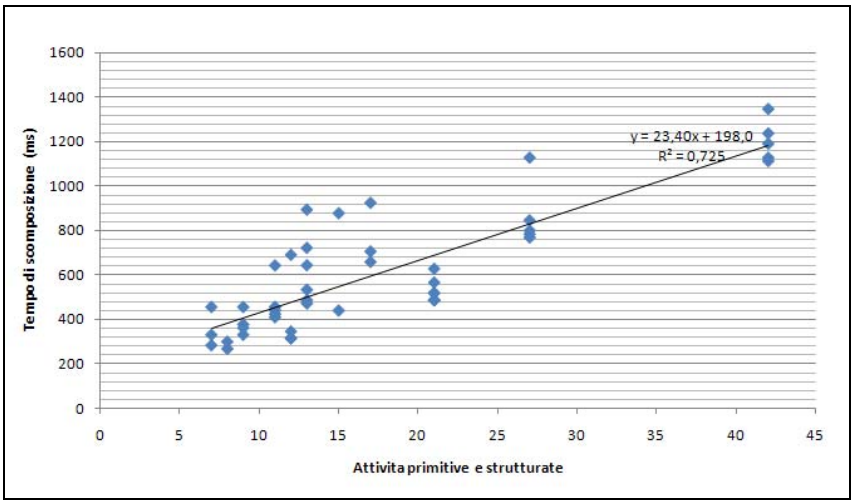


Al termine dell'elaborazione, in base al tipo restituito, il risultato è automaticamente visualizzato nella finestra di pop-up oppure è eseguito un redirect verso la posizione che contiene i dati

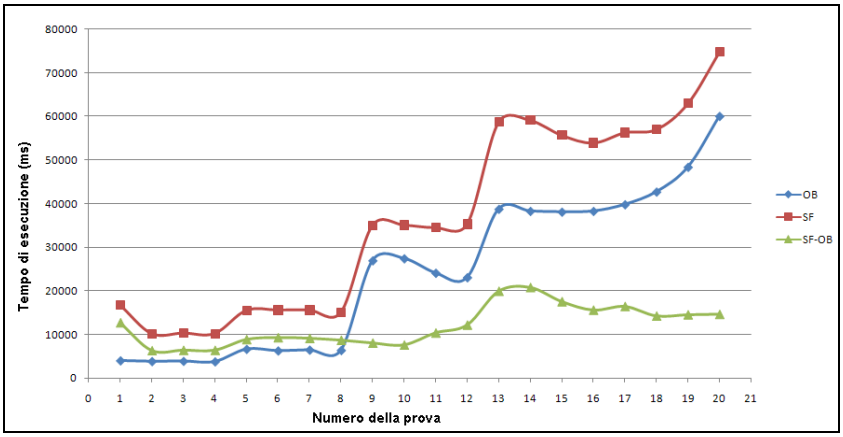


RISULTATI (output)	L'istanza del worflow in esecuzione può essere seguita attraverso la finestra di monitoring e al termine dell'elaborazione il risultato è automaticamente restituito all'utente

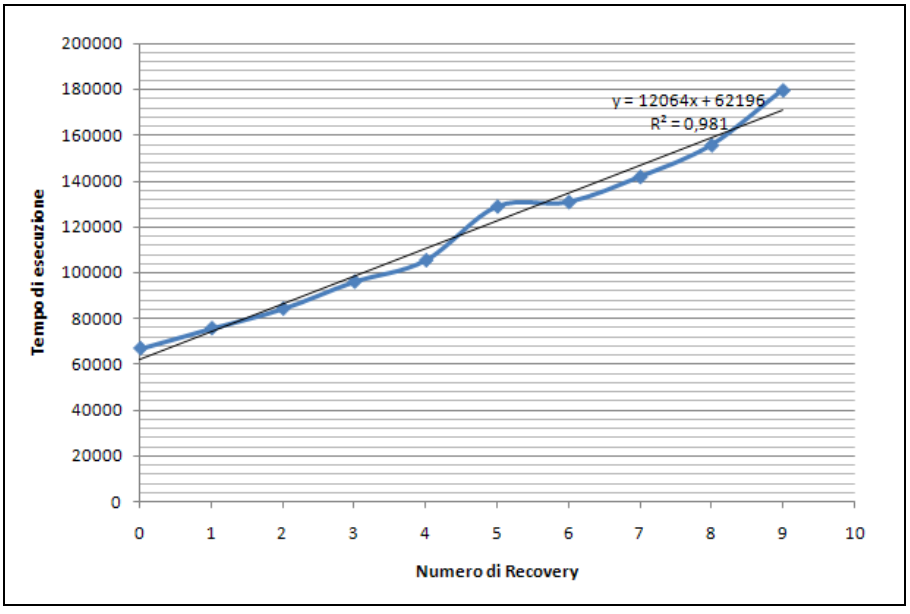
3.1.8 Test Funzionalità f3.1.8

TITOLO	Valutazione dei tempi di scomposizione di un workflow BPEL in frammenti
OBIETTIVO	Misurare i tempi impiegati dall'algoritmo di scomposizione del workflow BPEL per valutarne in modo empirico l'ordine di complessità
SET-UP (input)	A tal fine è stato predisposto un esperimento che prevede: • Undici workflow BPEL con crescente numero di attività primitive e strutturate. • Ogni workflow BPEL è eseguito più volte, per ridurre l'effetto stocastico generato dai tempi latenza dei dischi rigidi e del multithreading del processore.
STEP ESEGUITI	I dati ricavati dalle varie esecuzioni dell'algoritmo di scomposizione sono stati riportati nel grafico seguente, dove: in ascissa troviamo il numero di attività di cui è composto il workflow e in ordinata il tempo di scomposizione in milli-secondi 
RISULTATI (output)	Attraverso analisi e interpolazione lineare dei punti si è verificato che i tempi di scomposizione dipendono sostanzialmente in modo lineare dal numero di attività primitive e strutturate. Questo ci conferma sperimentalmente che l'algoritmo di scomposizione è dell'ordine di complessità di O(n), infatti l'algoritmo lavora analizzando ricorsivamente tutti i tag xml dell'albero contenuti nel workflow BPEL, e come è ben noto in letteratura la visita di tutti i nodi di un albero ha complessità O(n).

3.1.9 Test Funzionalità f3.1.9

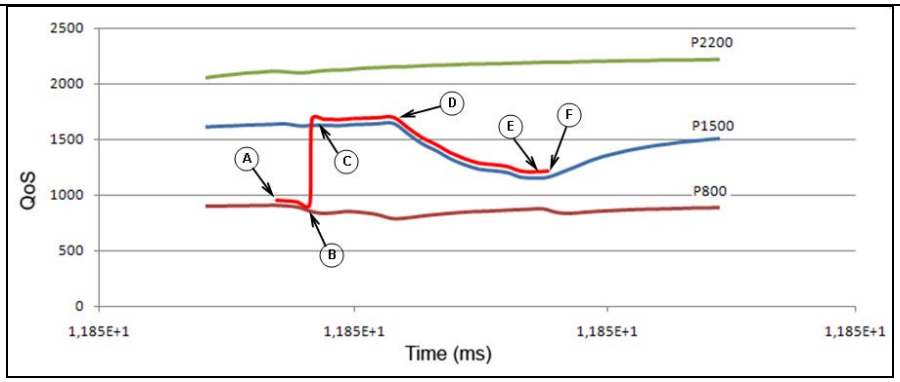
TITOLO	Confronto dei tempi di esecuzione tra Oracle BPEL (orchestrazione) e Sunflower (coreografia)
OBIETTIVO	Confrontare, a parità di condizioni, il tempo impiegato dai due motori di workflow ad eseguire lo stesso codice BPEL
SET-UP (input)	A tal fine è stato predisposto un esperimento che prevede: • Quattro workflow BPEL, con crescente numero di attività primitive e strutturate. • Esecuzione del workflow BPEL su Oracle BPEL e su SunFLOWer, utilizzando gli stessi servizi e variali. • Ogni workflow BPEL è eseguito più volte, per ridurre l'effetto stocastico generato dai tempi latenza dei dischi rigidi, del multithreading del processore e dei ritardi di comunicazione sulla rete.
STEP ESEGUITI	I dati ricavati dalle varie esecuzioni dei workflow sono stati riportati nel grafico seguente, dove: in ascissa troviamo il numero della prova eseguita e in ordinata il tempo di esecuzione in milli-secondi ottenuto da Sunflower e Oracle BPEL 
RISULTATI (output)	Come si può notare dal grafico SunFLOWer (SF) è più lento dell'Oracle BPEL (OB), infatti, la differenza tra SF e OB (SF-OB) è sempre positiva, ciò è dovuto ai tempi di comunicazione tra i Peer JXTA che eseguono i vari frammenti del workflow. La curva SF-OB è crescente al crescere del numero della prova, questo è dovuto al fatto che le prove: da 1 a 4 sono state effettuate con un workflow con 9 attività, da 5 a 8 con 11 attività, da 9 a 12 con 13 attività e da 13 a 20 con 17 attività. Di conseguenza al crescere del numero di attività primitive e strutturate aumentano il numero di comunicazioni tra i Peer JXTA e a sua volta aumenta il tempo di esecuzione del workflow.

3.1.10 Test Funzionalità f3.1.10

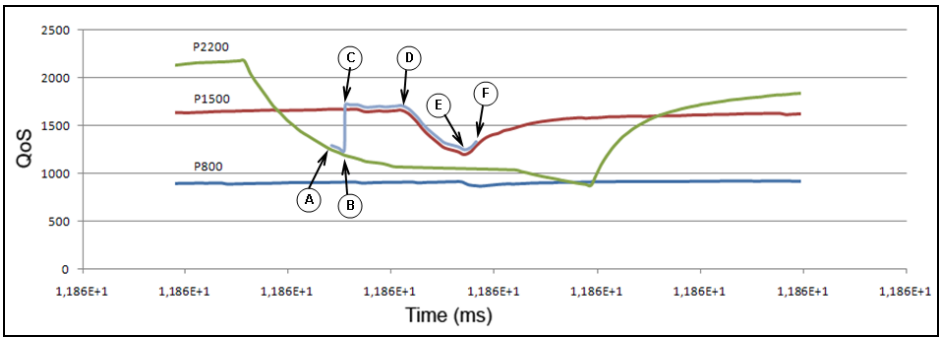
TITOLO	Misura del tempo di esecuzione di un workflow con Recovery multiplo
OBIETTIVO	Valutare il tempo supplementare necessario all'esecuzione di un workflow in caso di Recovey
SET-UP (input)	A tal fine è stato predisposto un esperimento che prevede: • L'esecuzione per dieci volte dello stesso workflow • Ogni prova si è forzato nel workflow un numero crescente di Recovery
STEP ESEGUITI	I dati ricavati dalle varie esecuzioni del workflow sono stati riportati nel grafico seguente, dove: in ascissa troviamo il numero di recovery forzati nella singola esecuzione e in ordinata il tempo di esecuzione in milli-secondi ottenuto da Sunflower 
RISULTATI (output)	Come si può notare dal grafico ad ogni Recovery aggiuntivo il tempo di esecuzione aumenta solo di una piccola quantità costante di tempo dovuta alle operazioni di Recovery. È evidente che i tempi di esecuzione in caso di errore, tra le operazioni eseguite manualmente dall'utente e il Recovery di SunFLOWer, risultano differenti di qualche ordine di grandezza e quindi assolutamente non paragonabili.

3.1.11 Test Funzionalità f3.1.11

TITOLO	Verifica dell'adattività di SunFLOWer alla variazione della QoS
OBIETTIVO	Valutare il meccanismo di Routing per la sostituzione soft di un servizio con un equivalente durante l'esecuzione di un workflow, il tutto guidato da strategie basate sulla QoS
SET-UP (input)	A tal fine è stato predisposto un esperimento che prevede: • Tre computer, sui quali è stato installato lo stesso servizio. • I tre computer differiscono principalmente per la velocità del processore, che ha un peso importante nel calcolo della QoS. • Il workflow, usato per le prove, utilizza il servizio di una delle tre macchine a condizione che la sua QoS sia almeno di 1500.
STEP ESEGUITI	La prima figura riporta le QoS misurate durante l'esecuzione del workflow. Le curve indicate dall'etichetta P800, P1500 e P2200 indicano rispettivamente le QoS misurate su un Pentium 3 a 800 Mhz, un Pentium 4 a 1500 Mhz e un Pentium 4 a 2200 Mhz. Mentre la curva indicata dalle frecce con etichette A...F individua il servizio prescelto per essere utilizzato nel workflow. Le frecce invece indicano le varie fasi di elaborazione del workflow, più precisamente: A. Inizio dell'esecuzione del Workflow. B. Creazione dell'attività di Routing. C. Creazione della nuova attività di Invoke. D. Inizio dell'esecuzione dell'attività di Invoke. E. Fine dell'esecuzione dell'attività di Invoke. F. Fine dell'esecuzione del Workflow. Come si può notare, inizialmente, era stato prescelto un servizio con QoS inferiore alla soglia stabilita (P800), quindi sul Peer si è innescato il meccanismo di Routing che ha sostituito il servizio corrente con uno più performate (P1500), riuscendo cosi a rispettare i vincoli sulle QoS.



La seconda figura evidenzia il comportamento del sistema nel caso in cui processi diversi utilizzino la stessa macchina. Sulla macchina P2200 prima di avviare il workflow è stato avviato un processo per generare un sovraccarico di lavoro, questo processo serve a simulare un eventuale utente che utilizza la macchina o un altro workflow che sta usando i Servizi. Il servizio scelto inizialmente P2200, che in teoria è il più performante, all'avvio del workflow si trova a non avere più la QoS necessaria a soddisfare le richieste, quindi scatta il meccanismo di Routing, che riporta il sistema in condizioni ottimali.



RISULTATI (output)	Com'è facilmente intuibile, il meccanismo di Routing lavorando localmente per migliore le prestazioni dei singoli workflow, ma fa emergere a livello globale un bilanciamento del carico, facendo migrare il lavoro da eseguire sulle macchine più performanti e meno utilizzate