1. Installazione

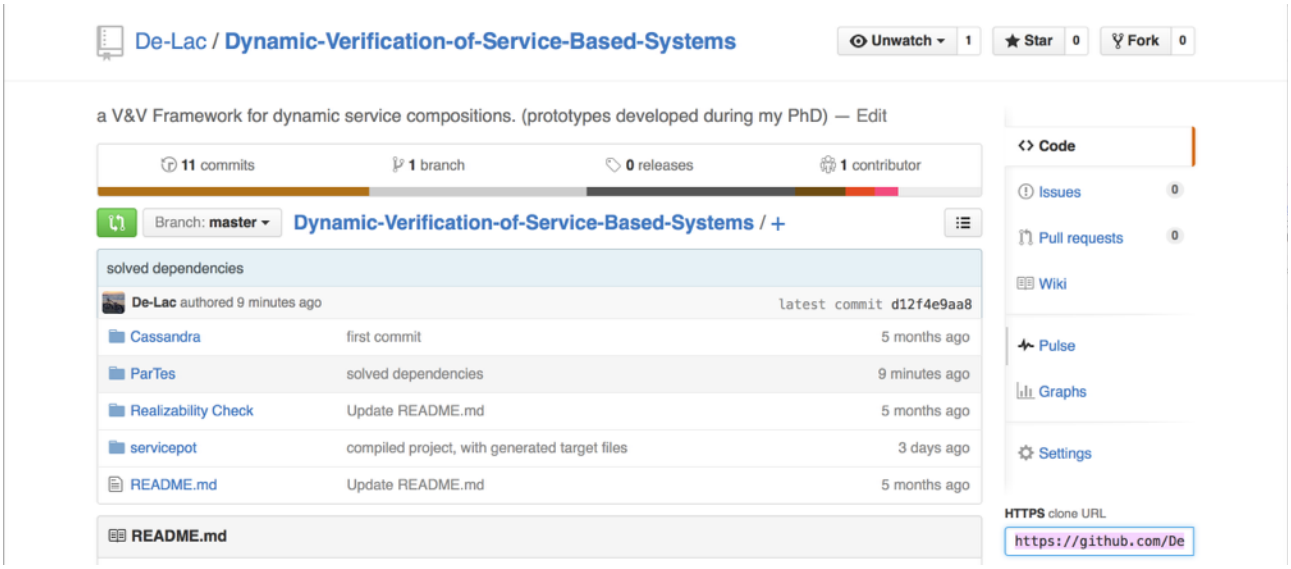
1.1. Requisiti

Questo è il software che ho attualmente in uso:
-) Eclipse IDE for Java Developers, Mars Release (4.5.0), Build id: 20150621-1200
-) plug-in Git (già incluso)
-) plug-in Maven2Eclipse (già incluso) m2e connector for jaxws-maven-plugin 0.0.1.2015051000425
-) BPMN2 Modeler (da installare, sotto SOA development) v.1.2.1.201507081507

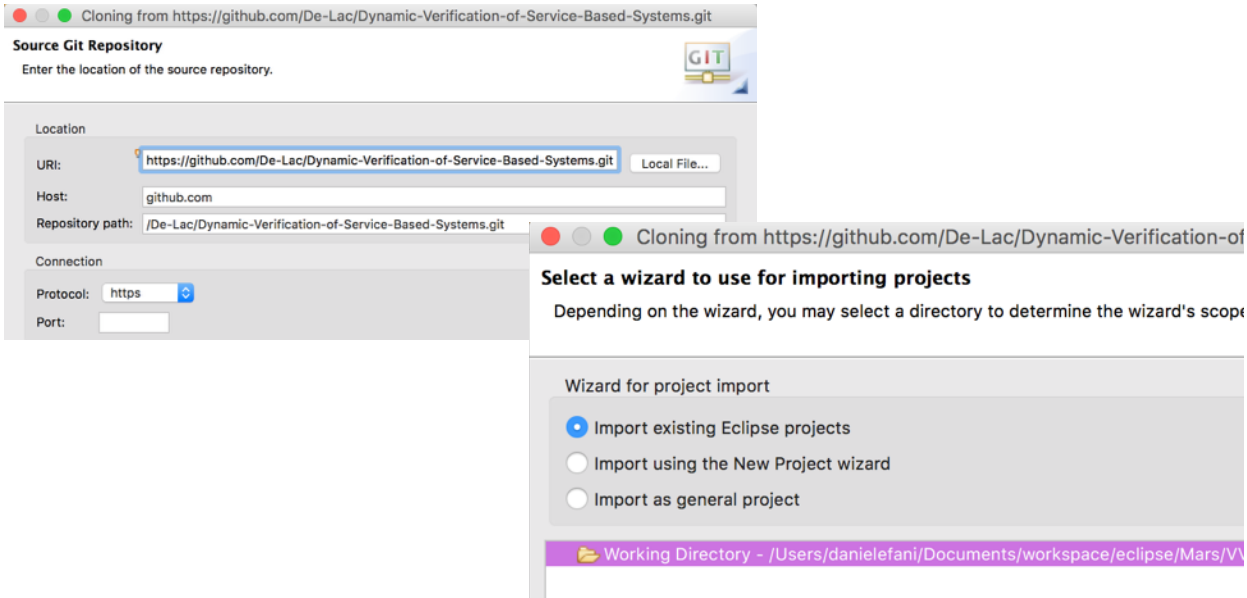
1.2. Procedura

1.2.1 Repository Git

Pagina web : https://github.com/De-Lac/Dynamic-Verification-of-Service-Based-Systems
Repository git: https://github.com/De-Lac/Dynamic-Verification-of-Service-Based-Systems.git



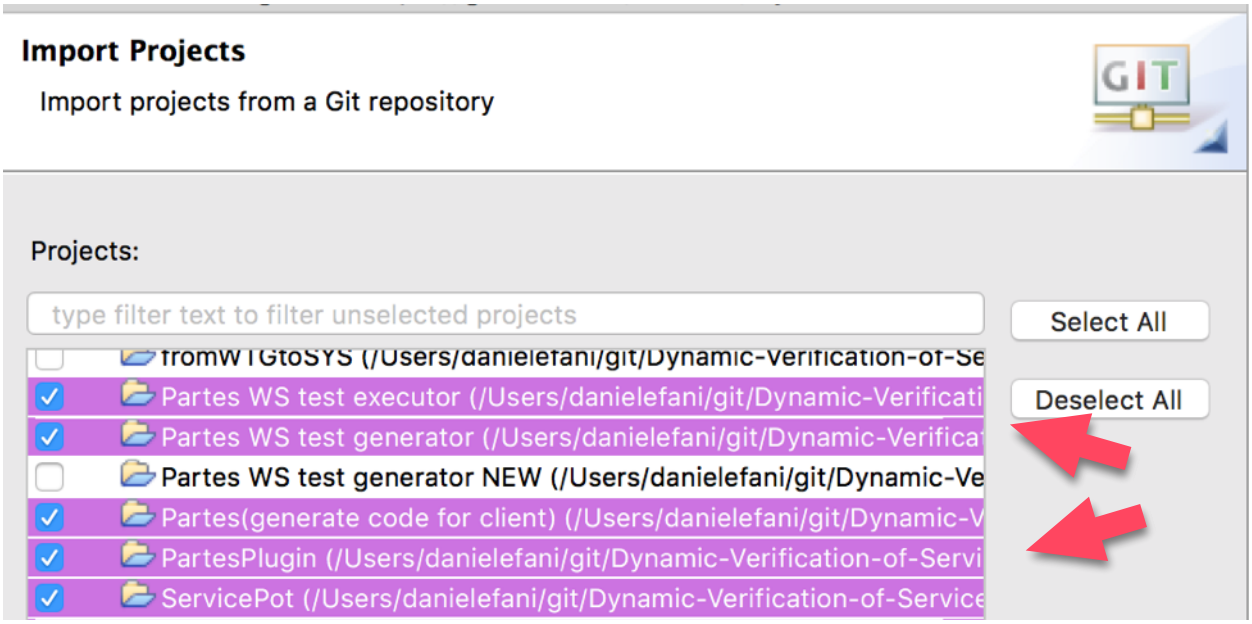
1.2.2 Clona repository su Eclipse



1.2.3 Seleziona progetti da clonare

Selezionare:
-) Partes WS test executor
-) PartesWS test generator
-) Partes (generate code for client)
-) PartesPlugin
-) ServicePot

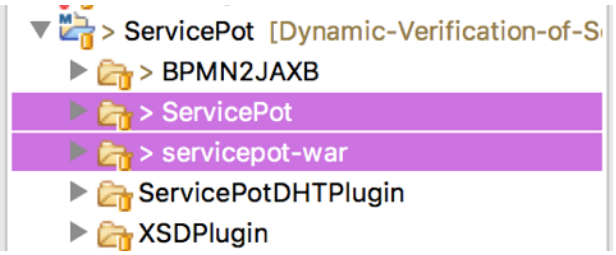
Dopo l'importazione è bene aggiornare i progetti (tasto destro su un progetto —> Maven —> Update Project)



2. Struttura

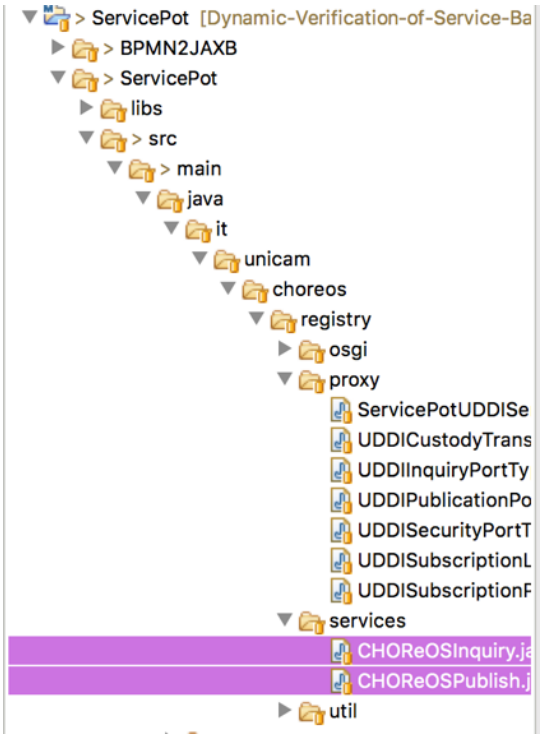
2.1. ServicePot

I moduli del progetto ServicePot da analizzare sono:
-) **ServicePot**, il vero è proprio registry, contenente l'implementazione delle interfacce del registry UDDI;
-) **servicepot-war**, il modulo che genera il .war e la pagina web che fa da GUI per l'utente.



BPMN2JAXB utilizza la libreria JAXB per trasformare oggetti XML in oggetti Java. Nello specifico, questo modulo trasforma tutti gli elementi XML del metamodello di una coreografia BPMN in classi Java.

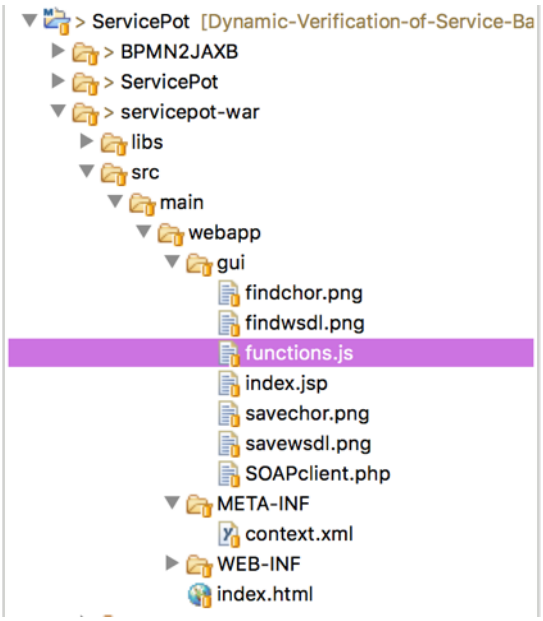
2.1.1. ServicePot OSGI



Il modulo ServicePot implementa sia le funzionalità base di un registry UDDI v.3, sia le funzionalità extra dichiarate dalle interfacce Inquiry e Publish.
L'implementazione di queste interfacce si trovano sotto `it.unicam.choreos.registry.services`.
L'attivazione automatica dei plugin di ServicePot, come ad esempio ParTes, può essere collegata ai metodi implementati proprio dalle classi CHOReOSInquiry e CHOReOSPublish. Nello specifico, per ogni metodo "xyz", l'attivazione può essere collegata all'evento "before xyz" o "after xyz".
La classe `it.unicam.choreos.util.ChoreographyRegistryWorker` implementa il metodo `getUDDIChoreographyFromBPMN()` responsabile nell'estrarre i ruoli (e i relativi WSDL collegati se presenti) direttamente dalla coreografia BPMN.

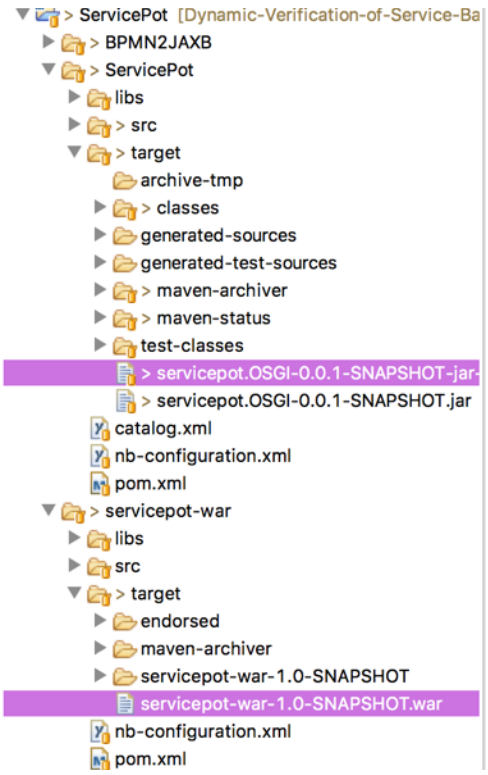
2.1.2. ServicePot.war

Il modulo ServicePot-war genera il .war di ServicePot, infatti include la libreria `org.ow2.choreos.servicepot.OSGI-0.0.1-SNAPSHOT-jar-with-dependencies` generata dal modulo precedente. Inoltre, questo modulo implementa una semplice pagina che permette di invocare alcune delle funzionalità di ServicePot come "cerca coreografia" e "salva coreografia". I file sorgenti della pagina web sono nella cartella webapp/gui.



2.1.3. Target

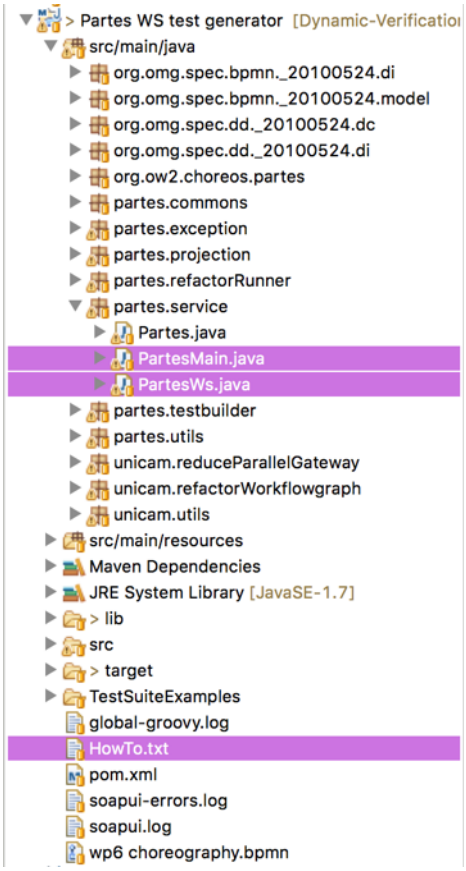
Eseguendo il progetto ServicePot (RunAs—>Maven install), per ogni modulo sarà generata una cartella "target" contenente le librerie generate.
Il modulo ServicePot genererà la libreria `servicepot.OSGI-0.0.1-SNAPSHOT.jar` in due versioni, una che comprende tutte le dipendenze, e una senza.



Il modulo servicepot-war genererà il file .war `servicepot-war-1.0-SNAPSHOT.war` da utilizzare con un contenitore di servlet java (Es. Jetty o Tomcat).

2.2 ParTes WS Test Generator

ParTes Test Generator implementa l'algoritmo per la generazione dei casi di test per SoapUI. Il progetto può essere eseguito tramite il main contenuto nella classe `PartesMain`, oppure come Web Service riferendosi alla classe `PartesWs`. Tuttavia, in entrambi i casi, La generazione vera e propria viene eseguita dal metodo `generateTestSuite()` della classe `partes.service.Partes`.
Il file di testo `HowTo.txt` contiene delle informazioni su come eseguire ParTes, e sui requisiti della coreografia da dare come input.
Eseguendo direttamente ParTes senza passare per ServicePot, richiede di passare come input, oltre alla coreografia, un file .yml con il mapping tra ruolo e WSDL relativo. Questo perché la procedura di estrazione dei WSDL dalla coreografia BPMN è implementata da ServicePot, ed eseguita durante la registrazione della coreografia nel registry. Le istruzioni di come preparare il file .yml sono contenute nel file HowTo.



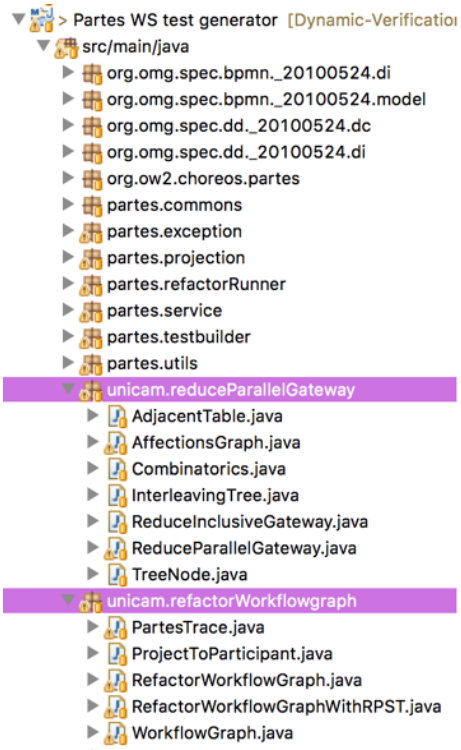
2.2.1. Input

Nel main `partes.service.PartesMain#main()`, se nessun parametro viene passato, vengono scelti dei parametri di default. Si consiglia di cambiare i parametri di default inserendo il percorso dei file di input del proprio pc.

```
…
if (args.length==0)
    { // if no argument is passed, default is taken (for testing purpose)
      formatter.printHelp("PartesMain", options, true);
      args = new String[2];  // or [3] if we want also debug = true
      args[0]="-p /…/…/coreografia.bpmn";
      args[1]="-y /…/…/participants.yml";
      // args[2]="-d";   // if we want debug = true
    }
…
```

In realtà, ParTes accetta anche il contenuto del file .yml piuttosto che il path del file. Questo viene utilizzato quando ServicePot estrae ruoli e WSDL, crea il contenuto del file .yml e lo manda a ParTes (vedi `PartesMain#run_withYmlContent()` e `PartesMain#run_withYmlPath()`).

2.2.2. Componenti



L'algoritmo utilizzato da ParTes per la creazione del WorkflowGraph, e la sua esplorazione comprendente la trasformazione in Grafo e la sua riduzione, è implementato nei package unicam (reduceParallelGateway e refactorWorkflowgraph).
In sostanza, la coreografia viene trasformata in un Workflowgraph. Poi si parte dal frammento SESE (single entry single exit) più interno: se è un parallelo si riduce, trasformandolo in un ExclusiveChoice con solo le tracce considerate rilevanti da testare; altrimenti si passa ad un altro frammento e si cerca di combinarli seguendo determinate regole di Workflow refactoring.
Al termine dell'algoritmo, si avrà un solo frammento EclusiveChoice, contenente tutte le tracce da testare. Le tracce sono poi raggruppate in base alle condizioni che soddisfano (le condizioni di tutti i gateway ExclusiveChoice incontrati durante la traccia).

Se si volesse cambiare/rifare la generazione di TestCase utilizzando ParTes per la generazione delle tracce, sarebbero da considerare questi due package. L'esecuzione dell'algoritmo inizia invocando prima il metodo `from_BPMN_to_WorkflowGraph()` e poi `refactorWorflowGraph()`, che a sua volta richiama tutti gli altri metodi per il refactoring e la riduzione.

2.3. ParTes (generate code for client)

E' utilizzato per la generazione automatica di codice sorgente per implementare un client per il WS di ParTes. Fare riferimento alla guida http://geertschuring.wordpress.com/2009/06/26/how-to-create-a-webservice-client-with-maven-and-jax-ws/.
Prima di eseguirlo, cambiare nel pom la URL del WSDL relativo al WS per il quale si vuole generare il client (in questo caso ParTes Test Generator). Il WS deve essere già attivo. La URL sarà del tipo:
`http://indirizzo_server/nome_war/services/PartesWsTestGenerator?wsdl`

2.3. ParTes Plugin

Questo progetto genera il plugin OSGI di ParTes per ServicePot. Dopo aver copiato i file sorgenti generati da `ParTes (generate code for client)`, includendo anche il Manifest, il .jar generato esportando il progetto come package, sarà compatibile con ServicePot. Il plugin non farà altro che inviare messaggi al WS ParTes Test Generator. Il binding tra "evento ServicePot" e "esecuzione automatica plugin" è possibile grazie alle interfacce che mette a disposizione ServicePot. Infatti la classe `PartesPlugin` implementa l'interfaccia `CHOReOSPublishMonitor`. Tutti i metodi di questa interfaccia saranno eseguiti automaticamente da ServicePot. Quindi ad esempio, implementando il metodo `afterSaveChoreography()` nel plugin di ParTes, la sua esecuzione sarà attivata automaticamente dopo l'evento `SaveChoreography` di ServicePot.

3 Esecuzione

Ancora da scrivere.. comunque nel progetto ParTes test generator, sotto la cartella TestSuiteExample, c'è un esempio di coreografia, wsdl e relativa TestSuite generata.
1) la cartella WSDL con i .wsdl astratti dei ruoli.;
2) la coreografia bpmn non integra al momento i riferimenti ai .wsdl, il binding è fatto tramite il file role_matching.yml
3) la coreografia può essere visualizzata su Eclipse utilizzando l'editor BPMN2;
4) La coreografia può essere utilizzata con ParTes e ServicePot.