Il Registro dei Servizi di OpenSPCoop

Indice

Elenco delle figure

1 Introduzione

Attualmente OpenSPCoop mette a disposizione quattro versioni del registro dei servizi:

- *Registro XML*, il registro è realizzato tramite un singolo file xml. Questa versione è perfettamente funzionale, e più semplice da gestire, in scenari non molto dinamici e con un basso numero di soggetti e servizi in gioco. In ogni caso, è il modo migliore di usare OpenSPCoop in ambiente di testing e di sviluppo. Al contrario degli altri, questo registro può essere gestito solo manualmente tramite l'editing del file xml.
- *Registro su Database*, il registro è realizzato come un database relazionale; questo registro può essere gestito tramite l'interfaccia web di gestione del Registro.
- *Registro Web*, il registro è realizzato come un sito web che mantiene un insieme di oggetti descritti in xml e accessibili via http; questo registro può essere gestito tramite l'interfaccia web di gestione del Registro.
- *Registro UDDI*, il registro è realizzato tramite un registro in tecnologia UDDI che indicizza un insieme di oggetti descritti in xml, mantenuti in un repository accessibile via http; questo registro può essere gestito tramite l'interfaccia web di gestione del Registro.

Il Registro XML è ampiamente documentato nel Manuale di Gestione XML della Porta di Dominio OpenSPCoop

Per quanto riguarda il Registro su Database, la gestione avviene tramite le interfacce grafiche messe a disposizione da OpenSPCoop, nello specifico la *pddConsole* e la *regserv*. La documentazione d'uso per la configurazione del registro su database tramite le interfacce grafiche è disponibile nella Guida Utente.

Nelle prossime due sezioni documenteremo prima in dettaglio il formato xml di configurazione del Registro dei Servizi, quindi l'uso dell'Interfaccia Web di Gestione, tramite la quale avviene la configurazione degli altri tipi di Registro.

2 Visualizzazione del registro dei servizi HTTP

Il registro dei servizi HTTP è consultabile tramite un qualsiasi browser WEB. La struttura dei dati memorizzati in un repository HTTP , a partire dalla URL di accesso al repository (es. http://server:8080/repository/), è la seguente:

- accordi_cooperazione
  - AccordoCooperazione1.xml
  - allegatiAccordoCooperazione1
    - allegato1.xml
    - allegato2.doc
    - . . .
    - allegatoN.txt
  - specificheSemiformaliAccordoCooperazione1
    - specificaSemiformale1.xml
    - specificaSemiformale2.uml
    - . . .
    - specificaSemiformaleN.doc
  - ....
  - AccordoCooperazioneN.xml
  - allegatiAccordoCooperazioneN
    - . . .
  - specificheSemiformaliAccordoCooperazioneN
    - . . .

- accordi
  - AccordoServizio1.xml
  - wsdlAccordoServizio1
    - definitorio.xsd
    - concettuale.wsdl
    - logicoErogatore.wsdl
    - logicoFruitore.wsdl
  - wsblAccordoServizio1
    - concettuale.wsbl
    - logicoErogatore.wsbl
    - logicoFruitore.wsbl
  - allegatiAccordoServizio1
    - allegato1.xml
    - allegato2.doc
    - . . .
    - allegatoN.txt
  - specificheSemiformaliAccordoServizio1
    - specificaSemiformale1.xml
    - specificaSemiformale2.uml
    - . . .
    - specificaSemiformaleN.doc
  - ....
  - AccordoServizioN.xml
  - wsdlAccordoServizioN
    - . . .
  - wsblAccordoServizioN
    - . . .
  - allegatiAccordoServizioN
    - . . .
  - specificheSemiformaliAccordoServizioN
    - . . .
- porte_di_dominio
  - PortaDiDominio1.xml
  - . . .
  - PortaDiDominioN.xml
- TipoNomeSoggetto1
  - TipoNomeSoggetto1.xml
  - servizi
    - TipoNomeServizio1.xml
    - wsdlTipoNomeServizio1
      - implementativoErogatore.wsdl
      - implementativoFruitore.wsdl
      - TipoNomeSoggettoFruitore1 [Error: itemizedlist too deeply nested]
      - ....

      - TipoNomeSoggettoFruitoreN [Error: itemizedlist too deeply nested]
    - allegatiTipoNomeServizio1
      - allegato1.xml
      - allegato2.doc
      - . . .
      - allegatoN.txt
    - specificheSemiformaliTipoNomeServizio1
      - specificaSemiformale1.xml
      - specificaSemiformale2.uml
      - . . .
      - specificaSemiformaleN.doc
    - specificheLivelliServizioTipoNomeServizio1
      - specificaLivelliServizi1.wsla
      - . . .
      - specificaLivelliServiziN.wsla
    - specificheSicurezzaTipoNomeServizio1
      - specificaSicurezza1.xml
      - . . .
      - specificaSicurezzaN.doc
    - ....
    - TipoNomeServizioN.xml
    - wsdlTipoNomeServizioN
      - . . .
    - allegatiTipoNomeServizioN
      - . . .
    - specificheSemiformaliTipoNomeServizioN
      - . . .
    - specificheLivelliServizioTipoNomeServizioN
      - . . .
    - specificheSicurezzaTipoNomeServizioN
      - . . .
- ...
- TipoNomeSoggettoN
  - . . .

La figura seguente mostra un esempio di navigazione del repository http, attraverso un browser web:

Index of /repository

Name	Last modified	Size	Description
Parent Directory		-	
SPCMinistero1/	08-Aug-2007 10:02	-	
SPCMinistero2	09-Aug-2007 14:33	-	
accordi/	31-Aug-2007 11:21	-	

Index of /repository

Name	Last modified	Size	Description
Parent Directory		-	
Sincrono.xml	08-Aug-2007 10:02	435	
wsdlSincrono/	08-Aug-2007 10:02	-	

Index of /repository

Name	Last modified	Size	Description
Parent Directory		-	

```
<!-- Messaggi utilizzati nel WSDL -->
<wsdl:message name="movOneWayRequest">
   <wsdl:part element="types:movOneWayRequest" name="parameters"/>
</wsdl:message>

<wsdl:message name="movOneWayResponse">
   <wsdl:part element="types:movOneWayResponse" name="parameters"/>
</wsdl:message>

<wsdl:message name="movSincronoRequest">
   <wsdl:part element="types:movSincrono" name="parameters"/>
</wsdl:message>

<wsdl:message name="movSincronoResponse">
   <wsdl:part element="types:movSincronoResponse" name="parameters"/>
</wsdl:message>
```

Figura 1: Repository HTTP

3 Visualizzazione del registro dei servizi UDDI

Il registro dei servizi UDDI è consultabile tramite un qualsiasi browser UDDI. Le entità UDDI utilizzate per la rappresentazione delle varie entità sono descritte di seguito:

- *Registrazione di un accordo di Servizio*, viene creata una TModel specifica per l'accordo di Servizio, nella quale, nel campo OverviewURL, è presente il riferimento al manifest xml che descrive la parte comunde dell'accordo di servizio.
- *Registrazione di un accordo di Cooperazione*, viene creata una TModel specifica per l'accordo di Cooperazione, nella quale, nel campo OverviewURL, è presente il riferimento al manifest xml che descrive la parte comunde dell'accordo di servizio. Il nome della TModel contiene come prefisso *AccordoCooperazione@*.
- *Registrazione di una Porta di Dominio*, viene creata una TModel specifica per la porta di dominio, nella quale, nel campo OverviewURL, è presente il riferimento al manifest xml che descrive la parte comunde dell'accordo di servizio. Il nome della TModel contiene come prefisso *PdD@*.
- *Registrazione Soggetti SPCoop*, per la registrazione di un soggetto nel registro UDDI viene utilizzata una chiave di tassonomia appositamente definita. La registrazione del soggetto comporta la creazione di una BusinessEntity che possiede:
  - Un Identifier Bag che memorizza una KeyedReference a cui è associato l'identificativo del dominio nel campo KeyName, e l'identificatore del soggetto (inteso come coppia tipo/nome, ad es. SPC/MinisteroInterno) nel campo KeyValue.
  - Una Category Bag che memorizza una KeyedReference a cui è associato nel campo KeyValue la url che indicizza il file xml che descrive il soggetto.
- *Registrazione Servizi SPCoop*, la registrazione di un servizio richiede la precedente registrazione del soggetto che lo eroga e la registrazione dell'Accordo di Servizio che dovrà implementare. Viene creato un Business Service con il campo nome contenente l'identificativo del servizio (inteso come coppia tipo/nome, ad es. SPC/ComunicazioneVariazione). Associato al Business Service viene creato un Binding Template contenente nel campo Access Point il riferimento al manifest XML che descrive la parte specifica dell'accordo. Inoltre, associata al Binding Template viene collegata la TModel dell'accordo di servizio implementato.

La figura seguente mostra la struttura di un registro dei servizi UDDI di OpenSPCoop, evidenziando come il registro UDDI sia utilizzato per indicizzare le definizioni XML memorizzate in un repository HTTP, uguale a quello del registro dei servizi WEB.

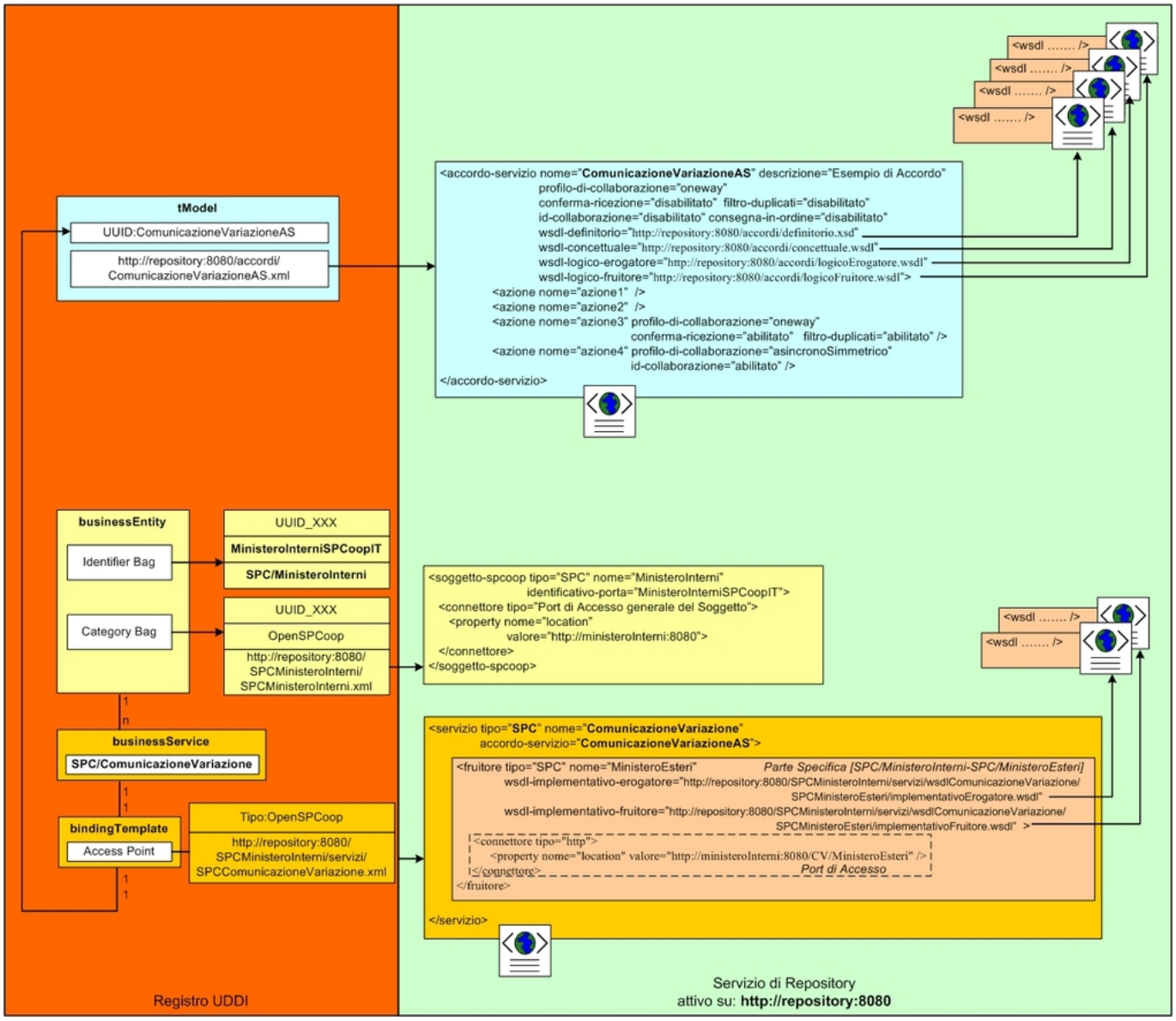

Figura 2: Repository UDDI

Le figure seguenti mostrano un esempio di navigazione di un registro UDDI, attraverso il software UDDIBrowser (http://uddibrowser.org/

Nella prima figura viene effettuata una ricerca di un accordo di servizio che possiede il nome *ComunicazioneVariazioneAS*: Indirizzando la OverviewDoc URL (http://repository:8080/accordi/ASComunicazioneVariazione,xml), individuata nella ricerca UDDI, attraverso un qualunque browser http, è poi possibile visualizzare la definizione dell'accordo di servizio. Sempre tramite un browser http, usando le chiavi di accesso definite nell'xml per i vari wsdl, è possibile scaricare il wsdl definitorio, quello concettuale e quello logico-erogatore/fruitore.

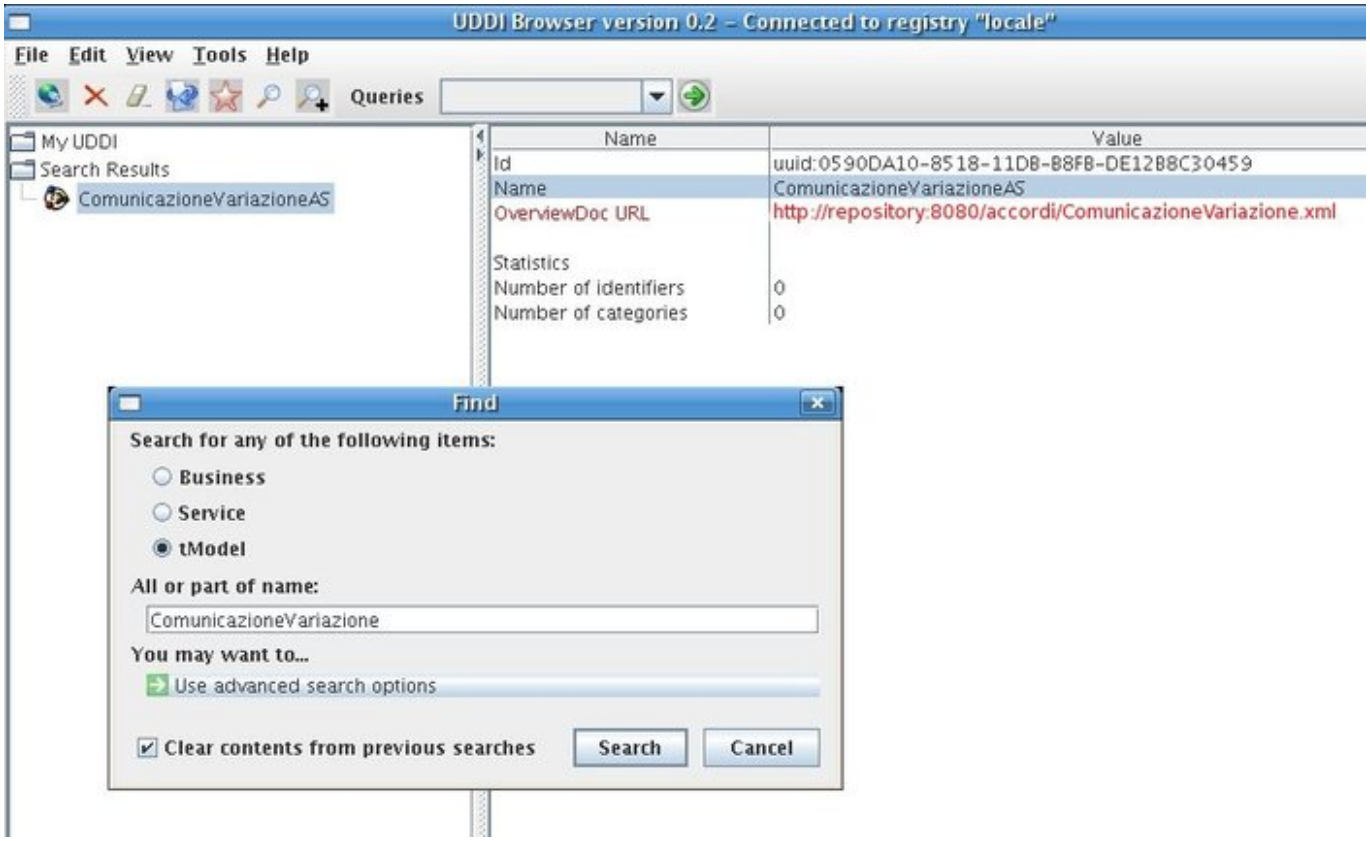

Figura 3: UDDI Browser, accordo di servizio

La seguente figura illustra un'ulteriore ricerca tramite UDDIBrowser di tutti i soggetti che forniscono un servizio che implementa l'accordo di servizio 'ComunicazioneVariazioneAS'. Per far questa ricerca si deve utilizzare l'UUID associato alla TModel dell'accordo di servizio precedentemente individuato. A questo punto sarà possibile prelevare dalla BusinessServices (Servizio SPC/ComunicazioneVariazione) nel soggetto erogatore SPCMinisteroInterni la chiave di accesso alla definizione xml del servizio. La url potrà poi essere utilizzata con un browser http per ottenere la definizione xml ed i wsdl implementativi.

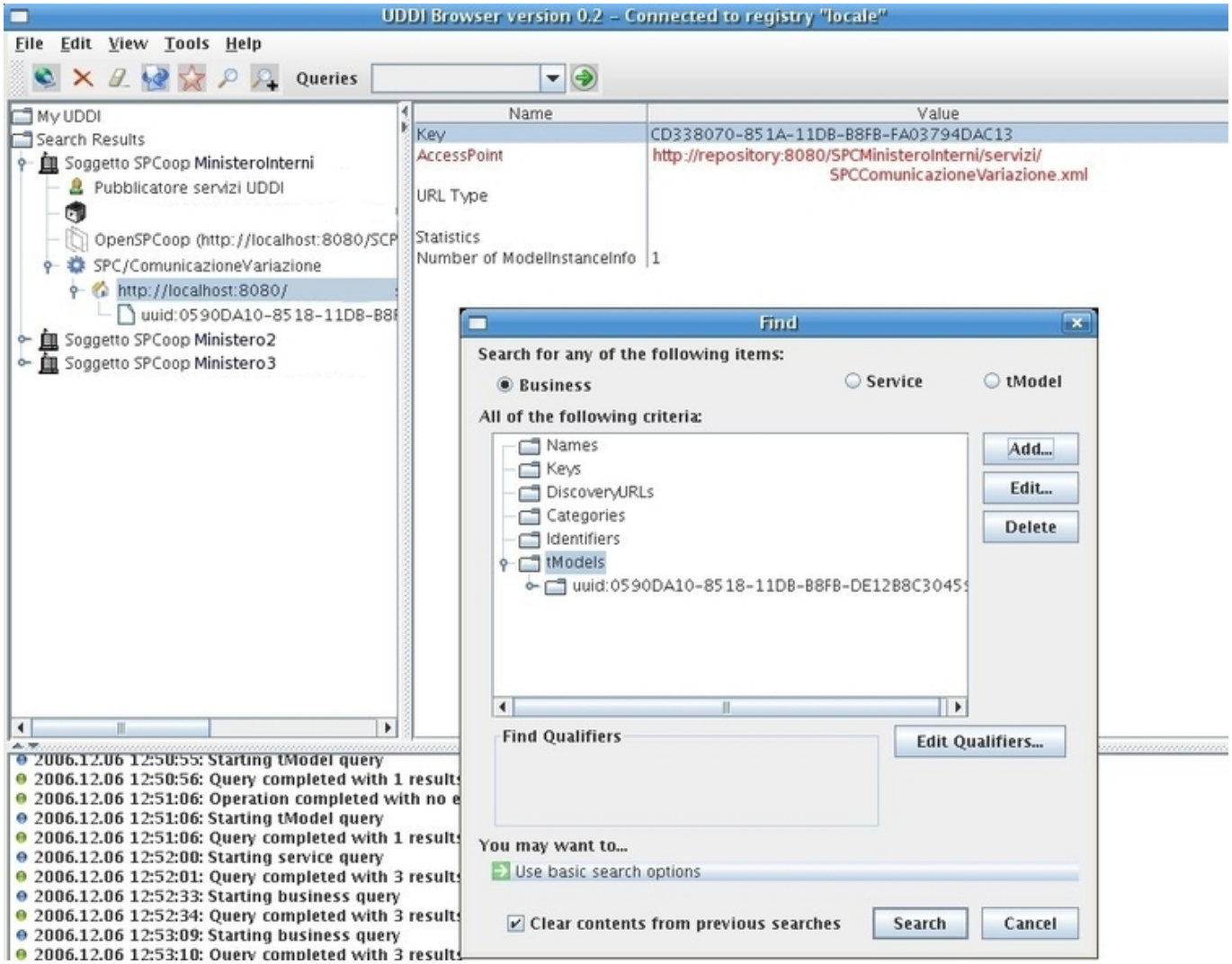

Figura 4: UDDI Browser, definizione xml di un servizio

4 L'Interfaccia WebService del Registro

I registri di OpenSPCoop possono essere interrogati attraverso l'interfaccia WebService WSRegistrySearchService appositamente predisposta. Il servizio WSRegistrySearchService può gestire uno qualsiasi dei tipi di registri descritti nei precedenti paragrafi.

Diamo adesso una descrizione delle funzionalità del WebService di interrogazione, utilizzando per ragioni di comodo il formalismo Java.

```
AccordoServizio getAccordoServizio(IDAccordo idA)

/* Ricerca Accordo di Servizio per ID.
si occupa di restituire l'oggetto AccordoServizio corrispondente ai criteri ↩
     passati tramite il parametro
di input. IDAccordo è l'oggetto che incapsula i criteri per l' ↩
    identificazione di un Accordo di Servizio
L'oggetto di tipo AccordoServizio è il bean contenente i valori di  ↩
    dettaglio dell'accordo. */

IDAccordo[] getAllIdAccordiServizio(FiltroAccordiSPCoop filtroRicerca)
```

```
/* Ricerca gli Accordi di Servizio che soddisfano determinati criteri di filtro.
La ricerca tramite i criteri impostati produce come risultato un insieme di ID di Accordi di Servizio
che soddisfano tali criteri. */

PortaDominio getPortaDominio(String nomePdD)

/* Ricerca Porta di Dominio per nome. Viene restituita la porta di dominio avente come nome la stringa passata in input.
L'oggetto di tipo PortaDominio restituito contiene tutti i dettagli della porta di dominio recuperata. */

String[] getAllIdPorteDominio(FiltroSPCoop filtroRicerca)

/* Ricerca le Porte di Dominio che corrispondono a determinati criteri di filtro.
La ricerca tramite i criteri impostati produce come risultato un vettore di stringhe che rappresentano i nomi delle
Porte di Dominio che soddisfano tali criteri. */

SoggettoSpcoop getSoggettoSPCoop(IDSoggetto idS)

/* Ricerca Soggetto per ID.
Si occupa di restituire il soggetto avente per id il parametro in input.
L'oggetto di tipo IDSoggetto incapsula i criteri per l'identificazione di un Soggetto SPCoop
L'oggetto di tipo SoggettoSpcoop è il bean con i valori di dettaglio del soggetto. */

IDSoggetto[] getAllIdSoggettiRegistro(FiltroSPCoop filtroRicerca)

/* Ricerca i Soggetti che soddisfano determinati criteri di filtro.
La ricerca tramite i criteri impostati produce come risultato un insieme di Soggetti che corrispondono a tali criteri.
Tale insieme viene restituito come vettore di oggetti di tipo IDSoggetto.*/

ServizioSpcoop getServizioSPCoop(IDServizio idS)

/* Ricerca di un Servizio per ID.
L'oggetto di tipo ServizioSpcoop restituito è quello identificato tramite il parametro in input.
L'oggetto di tipo IDServizio incapsula i criteri per l'identificazione di un Servizio SPCoop.
L'oggetto di tipo ServizioSpcoop è il bean con i valori di dettaglio del servizio. */

ServizioSpcoop getServizioSPCoopCorrelato(IDServizio idS)

/* Ricerca del Servizio Correlato ad un Servizio con dato ID.
Viene restituito il servizio correlato al servizio identificato dal parametro in input. */

ServizioSpcoop getServizioSPCoopCorrelatoByAccordo(IDSoggetto idS, IDAccordo idA)

/* Ricerca del Servizio Correlato ad un Servizio corrispondente al soggetto e all'accordo specificati in input.
Il servizio restituito è il correlato al servizio che implementa l'accordo idA ed è erogato dal soggetto idS. */
```

```
IDServizio[] getAllIdServiziSPCoop(FiltroServiziSPCoop filtroRicerca)

/* Ricerca i Servizi che corrispondono a determinati criteri di filtro.
L'oggetto di tipo FiltroServiziSPCoop incapsula i criteri di ricerca ↩
    impostabili per filtrare i servizi.
Il metodo restituisce un vettore di oggetti di tipo IDServizio che ↩
    identificano i servizi che corrispondono
ai criteri impostati. */

IDServizio[] getAllIdServiziSPCoopCorrelati(FiltroServiziSPCoop ↩
    filtroRicerca)

/* Ricerca i Servizi correlati ai servizi che corrispondono a determinati ↩
    criteri di filtro.*/

AccordoCooperazione getAccordoCooperazione(IDAccordoCooperazione idA)

/* Ricerca Accordo di Cooperazione tramite ID.
Si occupa di restituire l'Accordo di Cooperazione identificato tramite ↩
    IDAccordoCooperazione.
L'oggetto di tipo IDAccordoCooperazione incapsula i criteri per l' ↩
    identificazione di un Accordo di Cooperazione.
L'oggetto di tipo AccordoCooperazione contiene il dettaglio dell'accordo di ↩
     cooperazione recuperato. */

IDAccordoCooperazione[] getAllIdAccordiCooperazione(FiltroAccordiSPCoop ↩
    filtroRicerca)

/* Ricerca gli Accordi di Cooperazione che corrispondono a determinati ↩
    criteri di filtro.
L'oggetto di tipo FiltroAccordiSPCoop incapsula i criteri per filtrare gli ↩
    Accordi di Cooperazione.
Il metodo restituisce un vettore di oggetti di tipo IDAccordoCooperazione ↩
    che identificano gli Accordi di Cooperazione
che corrispondono ai criteri di ricerca passati in input. */

void isAlive()

/* Test della Connessione al Registro
Metodo che verifica la connessione al registro. Se la connessione non è ↩
    presente viene sollevata un'eccezione che
contiene nella sua descrizione il motivo della mancata connessione. */
```

Nota
Per informazioni sull'installazione dell'interfaccia web service del Registro, si rimanda alla Guida di Installazione.