Porta di Dominio

Architettura del Software

Versione 1.0 del 25/08/2009

Sommario

1 Introduzione

Questo documento è stato realizzato in collaborazione con Vitrociset S.p.a. in quanto ditta aggiudicataria dell'appalto concorso per la realizzazione del "Sistema Regionale per la Cooperazione Applicativa in Sicurezza" della Regione Campania.

Il disciplinare tecnico [BG/DT] relativo al "Sistema Regionale per la cooperazione applicativa in Sicurezza" definisce l'infrastruttura di riferimento per ottenere funzionalità di cooperazione tra servizi offerti da diversi enti della Regione Campania, preservando l'autonomia e la peculiarità dei sistemi interconnessi. Oggetto del presente documento è l'individuazione dell'architettura del solo componente Porta di Dominio, mentre si rimanda al documento [SPICCA_06_Registro_SPICCA_Architettura del Software_v1] per l'architettura del Registro SPICCA.

1.1 Scopo

Il presente documento si inserisce nell'ambito della documentazione relativa al progetto SPICCA e costituisce la specifica degli elementi strutturali costituenti la Porta di Dominio.

1.2 Obiettivo

Obiettivo del presente documento è quello di individuare gli elementi strutturali della Porta di Dominio, specificandone le interfacce e le modalità di collaborazione, in conformità alle specifiche tecniche emesse dal CNIPA relativamente al Sistema Pubblico di Connettività e Cooperazione (SPCOOP).

1.3 Definizioni, Acronimi, e Abbreviazioni

Per le definizioni si faccia riferimento al documento "Glossario dei termini tecnici". Invece, di seguito è riportato l'elenco degli acronimi utilizzati:

- **CNIPA:** Centro Nazionale per l'Informatica nella Pubblica Amministrazione;
- **SPCOOP:** Sistema Pubblico di Connettività e Cooperazione;
- **PDD:** Porta di Dominio;
- **PA:** Pubblica Amministrazione;
- **SA:** Servizio Applicativo;
- **SIL:** Sistema Informativo Locale.

1.4 Riferimenti

1. [BG] Bando di gara appalto-concorso per l'acquisizione del "Sistema Regionale per la Cooperazione Applicativa in Sicurezza" della Regione Campania – BURC n.41 del 30/08/2004 .
2. [BG/DT] Bando di gara: appalto concorso per l'acquisizione del "Sistema Regionale per la Cooperazione Applicativa in Sicurezza" della Regione Campania – BURC n.41 del 30/08/2004 – Disciplinare Tecnico.

3. [SPCoop_01] Sistema Pubblico di Cooperazione: QUADRO TECNICO D'INSIEME. Versione 1.0, 14/10/2005.
4. [SPCoop_04] Sistema Pubblico di Cooperazione: PORTA DI DOMINIO. Versione 1.0, 14/10/2005.
5. [SPCoop_06] Sistema Pubblico di Cooperazione: SERVIZI DI REGISTRO. Versione 1.0, 14/10/2005.

2 Rappresentazione dell'Architettura

Per la rappresentazione dell'architettura si seguiranno le raccomandazioni del Rational Unified Process (RUP) come dettato dalle direttive aziendali relative al ciclo di vita del software.

Il processo RUP suggerisce la rappresentazione di un'architettura software tramite diverse ***viste architetturali***, ognuna delle quali pone enfasi su di un particolare aspetto dell'architettura considerata. In particolare, il predetto processo prevede le viste architetturali riportate in Figura 1, di seguito brevemente descritte:

- ***logical view*** (vista logica): rappresenta l'organizzazione concettuale del sistema in termini di *layer*, *componenti*, *subsystem* e *package*, ed evidenzia le funzionalità dei principali componenti e subsystem.
- ***implementation view*** (vista implementativa): esprime il cosiddetto "Modello implementativo" del RUP, ovvero rappresenta l'organizzazione del sistema in componenti "fisici", come, a titolo esemplificativo, l'organizzazione di classi in package Java, l'organizzazione di bytecode Java in file ".jar".
- ***use-case view*** (vista dei casi d'uso): rappresenta i casi d'uso più importanti dal punto di vista architetturale, ovvero i casi d'uso la cui realizzazione coinvolge una parte significativa dei componenti architettuarali individuati;
- ***process view*** (vista dei processi): evidenzia i processi ed i thread costituenti il sistema software, riportando le loro responsabilità, le modalità di collaborazione;
- ***deployment view*** (vista di dispiegamento): rappresenta il deployment dei processi e componenti costituenti il sistema in considerazione su nodi fisici d'elaborazione.

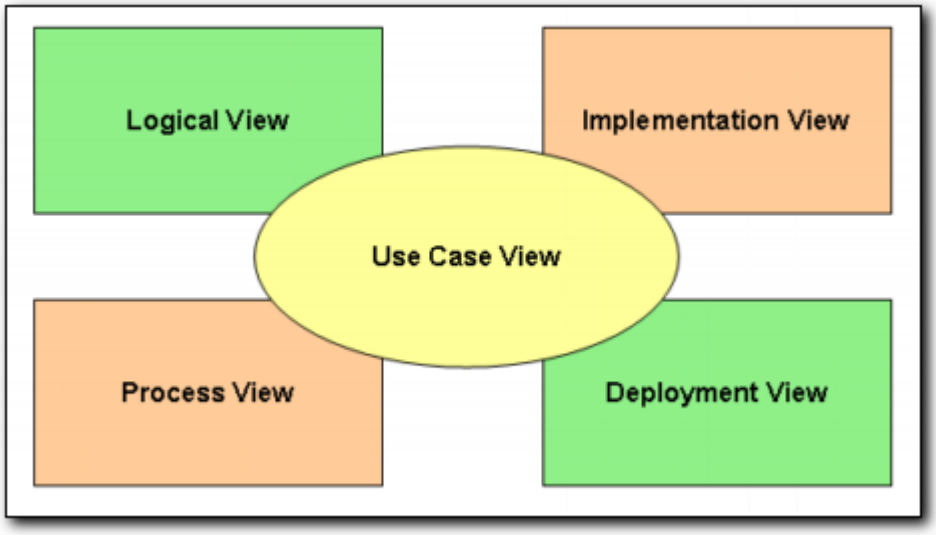


Figura 1 - le 4+1 viste architetturali (P. Kruchten 1995)

Relativamente all'architettura del software del componente Porta di Dominio, nel prosieguo del documento si riporterà le viste logica, implementativa e dei casi d'uso.

3 Descrizione dell'Architettura

3.1 Obiettivi

Come descritto nel documento [BG/DT], "*per l'implementazione delle componenti che permettono la semplificazione dei problemi di cooperazione*", tra cui la Porta di dominio, "*è richiesto l'utilizzo di tecnologie che facciano riferimento a standard aperti quali, a titolo esemplificativo, SOAP, UDDI, WS-Security, etc…*".

Ci si pone pertanto l'obiettivo di definire un'architettura del componente "Porta di Dominio" che soddisfi il predetto requisito e, nel contempo, requisiti non funzionali di affidabilità e prestazioni.

3.2 Schema a livelli

Il requisito relativo al supporto dell'interfaccia OpenPDD determina una suddivisione dell'architettura in diversi livelli funzionali. Ognuno di questi livelli sarà implementato, come meglio descritto nelle apposite sezioni del documento, da determinati componenti costituenti la Porta di Dominio. Prima di entrare nel merito di tali componenti, si preferisce fornire una descrizione generale dell'interfaccia OpenPDD, al fine di individuare i livelli software costituenti l'architettura della Porta di Dominio.

3.2.1 Architettura logica

L'utilizzo della PDD-SPICCA è totalmente incentrato sulla necessità di permettere la cooperazione applicativa tra i ventisei NDOM del CUP. Gli enti in questione (ASL NA/1, ASL AV1, ASL CE/1,...) espongono, attraverso altrettante PDD, i servizi:

1. Consunltazione Anagrafica Assistiti
2. Consultazione Anagrafica Medici di Base
3. Prenotazione Prestazione
4. Cancellazione Prestazione
5. Visualizzazione Prestazione

Questi servizi vengono gestiti centralmente passando sempre attraverso la PDD-SPICCA e utilizzando il sistema di autenticazione federata implementato nella soluzione. La stessa PDD-SPICCA è totalmente integrata con le problematiche di autenticazione federata descritte dallo standard WS-SAML.

L'architettura di riferimento è definita nella seguente figura:

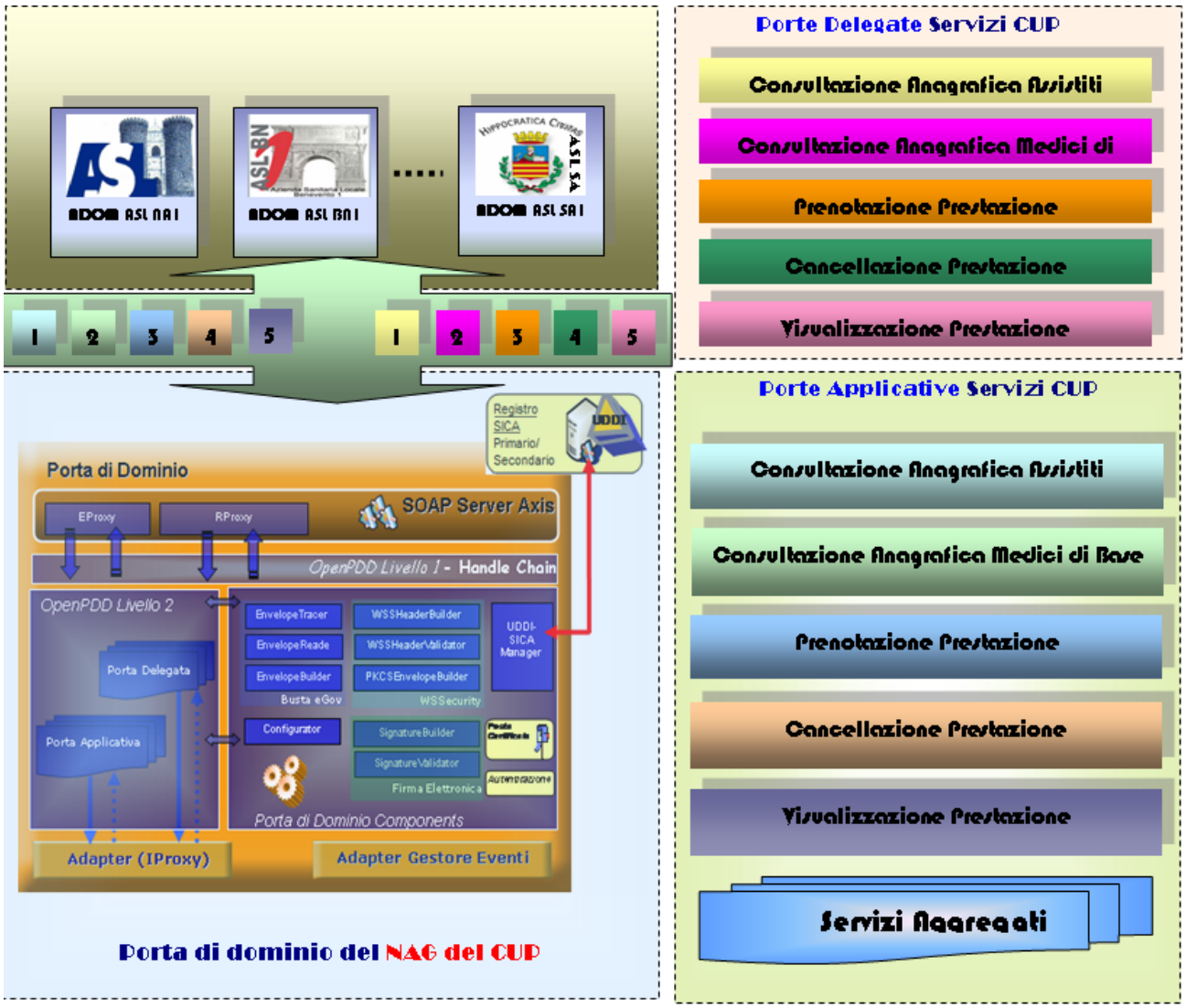


Figura 2 - Architettura logica

Lo stack tecnologico è invece descritto nella seguente immagine:

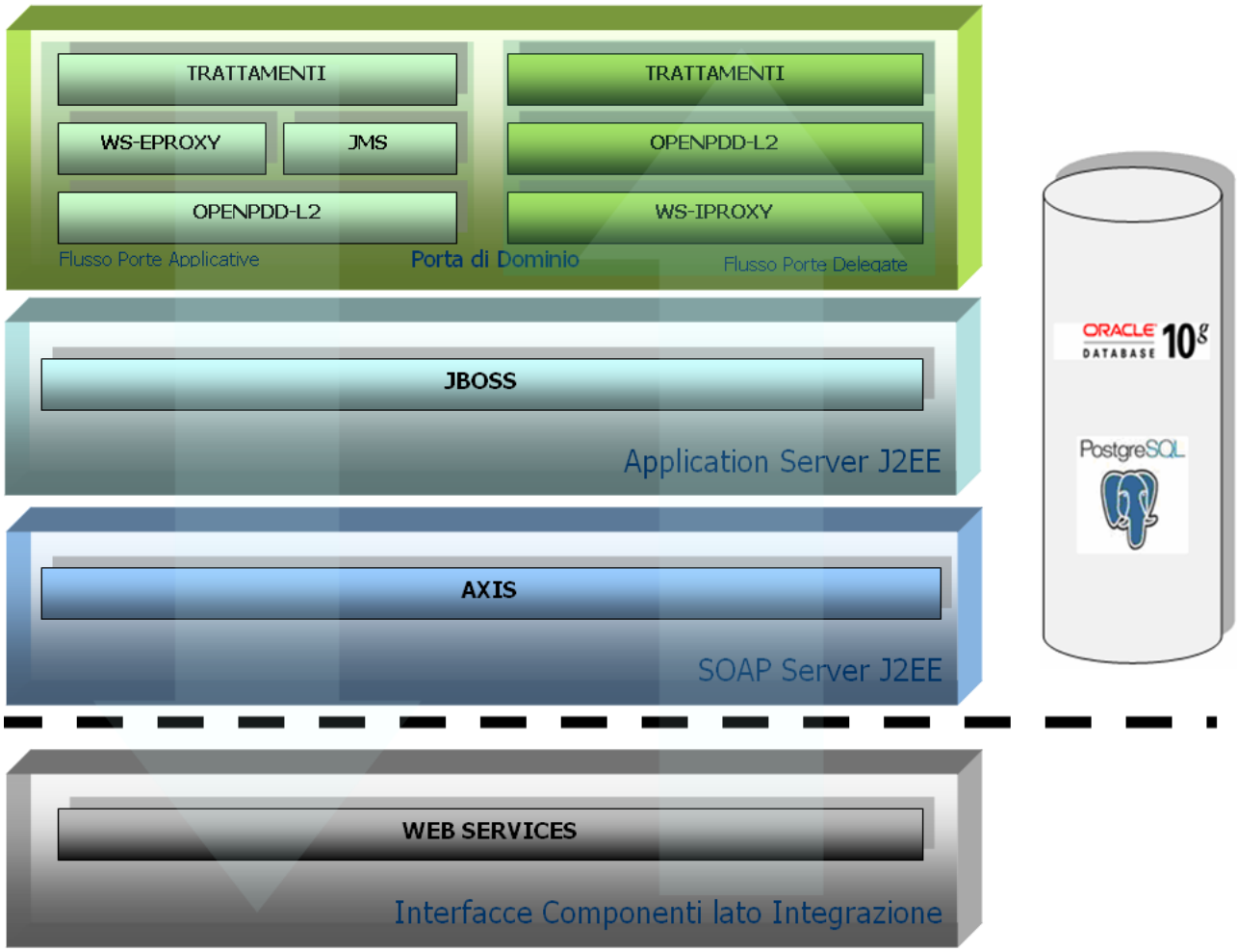


Figura 3 - Layout tecnologico

3.2.2 OpenPDD

OpenPDD è un set di API, interfacce e classi astratte che normalizza e standardizza l'accesso ai servizi delle Porte Di Dominio. Per mezzo delle API di OpenPDD le applicazioni ed i servizi dei progetti di e-Gov accedono alle PDD in maniera snella e semplificata potendo concentrarsi sui contenuti applicativi e potendo ignorare la complessità derivante dalla Porta Di Dominio e dalla Busta di e-Gov. In un'ottica semplificata OpenPDD definisce sia *l'interfaccia tra le applicazioni che usufruiscono della Porta di Dominio e la Porta di Dominio stessa*, consentendo di svincolare le prime dalla seconda, sia *l'interfaccia posta tra la Porta di Dominio ed il sistema di trasporto dei messaggi XML utilizzato dalla Porta di Dominio*.

Ad un alto livello di astrazione, da un generico Sistema Informativo Locale (SIL) che interagisce con il SPCOOP per mezzo di una Porta di Dominio, al SPCOOP stesso, possono essere individuati tre livelli di software, come si evince dalla figura 1: il *livello C* rappresenta le applicazioni del SIL, il *livello B* rappresenta l'implementazione delle funzionalità relative alla Porta di Dominio mentre il *livello A* costituisce l'implementazione del Gateway utilizzato per il trasporto dei messaggi (SOAP). Inoltre, le specifiche

SPCOOP prevedono che ogni Porta Di Dominio sia suddivisa logicamente in una componente di "Cooperazione" ed una di "Integrazione": la prima, rivolta verso l'esterno della PDD, è incaricata delle comunicazioni fra Porte Di Dominio, mentre la seconda interagisce con il Sistema Informativo Locale interno al Dominio. Quando due Porte di Dominio interagiscono per mezzo della componente di Cooperazione, la Porta di Dominio che svolge il ruolo di fornitore viene chiamata "Porta Applicativa" (server), mentre quella che usufruisce di un servizio viene definita "Porta Delegata" (client).

In questo contesto, l'obiettivo che si pone OpenPDD è la definizione di interfacce tra i vari livelli definiti in precedenza. In particolare nella comunicazione fra i livelli B e C si frappone l' interfaccia OpenPDD Livello 2 mentre la comunicazione fra i livelli A e B è modellata dall' interfaccia OpenPDD Livello 1.

Pertanto, OpenPDD Livello 2 rappresenta un'interfaccia tra i Sistemi Informativi Locali ed i servizi di "Porta Delegata" e "Porta Applicativa" forniti dalla Porta di Dominio, mentre OpenPDD Livello 1 è un' interfaccia che si pone tra i servizi di "Porta Delegata" e "Porta Applicativa" ed il meccanismo utilizzato per il trasporto XML, che garantisce la scalabilità della soluzione implementata.

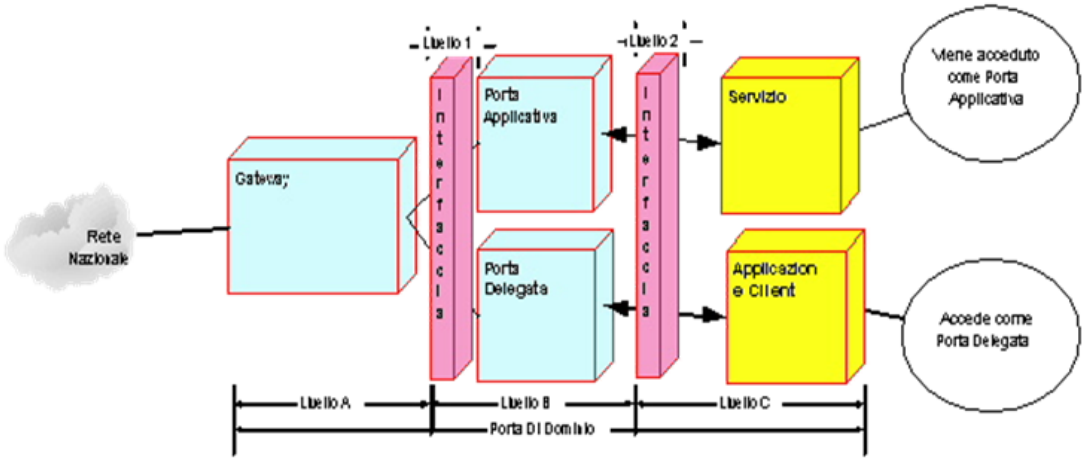


Figura 4 - Architettura OpenPDD

Nell'ambito della Porta Di Dominio SPICCA, è possibile individuare due livelli di software: il primo costituito dai componenti che implementano l'interfaccia OpenPDD livello 1 ed il secondo costituito dai componenti che implementano l'interfaccia OpenPDD livello 2.

3.2.2.1 OpenPDD l.2 – Fetures Implementate e principi di utilizzo

In questa sezione si descriveranno le specifiche OpenPdd l.2 implementate dalla Porta di Dominio SPICCA e si forniranno alcuni esempi di utilizzo della Porta di Dominio attraverso le interfacce OpenPdd level2.

La PDD Spicca implementa tutte le features obbligatorie definite da OpenPdd level2, ovvero quelle features che consentono la realizzazione dei servizi minimi e indispensabili di una Porta Di Dominio. In particolare, le features implementate sono:

1. supporto per le Porte Delegate;
2. supporto per le Porte Applicative;
3. supporto dei profili di comunicazione OneWay, Sincrono, Asincrono Asimmetrico, Asincrono Simmetrico.

Nel paragrafo n.6 - Implementation View - , è riportato il diagramma delle classi della Porta Di Dominio SPICCA che mette in evidenza le relazioni esistenti con le interfacce definite da OpenPDD level 2.

Esempi di utilizzo

Esempio di utilizzo di Porta delegata con profilo sincrono simmetrico

```
PDDPortaDelegataBuilder pdb = PDDPortaDelegataBuilderFactory.newInstance().
        newPortaDelegataBuilder();

PortaDelegata portaDelegata = pdb.newPortaDelegata(id_porta, (Object)"test"
);

Richiesta richiesta = portaDelegata.newRichiesta();

String contenutoApplicativo = …

InputStream contenutoApplicativo = new StringBufferInputStream(
        contenuto_applicativo);

Contenuto body = richiesta.addContenuto("body", "application/xml",
                                        contenutoApplicativo);
Document d = null;

Risposta r = null;

r = portaDelegata.send(richiesta);

Contenuto[] contenuti = r.getContenuti();

for (int i = 0; i < contenuti.length; i++)
  {
     InputStream is     = contenuti[i].getInputStream();
     byte[]      inbuff = new byte[is.available()];
     is.read(inbuff);
       //interpretazione del contenuto applicativo
       ...
 }
```

Esempio di utilizzo di Porta delegata con profilo one way

```
PDDPortaDelegataBuilder pdb = PDDPortaDelegataBuilderFactory.newInstance().
        newPortaDelegataBuilder();

PortaDelegata portaDelegata = pdb.newPortaDelegata(id_porta,
        (Object) "test");

Richiesta richiesta = portaDelegata.newRichiesta();

String contenutoApplicativo = …

InputStream contenutoApplicativo = new StringBufferInputStream(
        contenuto_applicativo);

Contenuto body = richiesta.addContenuto("body", "application/xml",
                                        contenutoApplicativo);
Risposta r = null;
portaDelegata.send(richiesta);
```

Esempio di utilizzo di Porta delegata con profile asincrono asimmetrico

```
PortaDelegataBuilder pdb = PDDPortaDelegataBuilderFactory.newInstance().
          newPortaDelegataBuilder();

PortaDelegata portaDelegata = pdb.newPortaDelegata(id_porta);

Richiesta richiesta = portaDelegata.newRichiesta();

String contenuto_applicativo = …;

InputStream contenutoApplicativo =
      new StringBufferInputStream( contenuto_applicativo);

Contenuto body =
      richiesta.addContenuto("body", "application/xml",
contenutoApplicativo);

Risposta risposta = portaDelegata.send(richiesta);

boolean isCompleted = false;
while (!isCompleted)
{

  Thread.sleep(5000);
  richiesta = portaDelegata.newRichiesta(risposta);
```

```
     String xmlBodyRequest =
"<p891:richiesta_RichiestaRispostaAsincrona_checkTestAsincronoAsimmetrico
xmlns:p891=\"http://sica.spcoop.it/servizi/QualificazionePDDWS/types\"><p891:Tok
enSessione>"+sessioneEgov+"</p891:TokenSessione></p891:richiesta_RichiestaRispos
taAsincrona_checkTestAsincronoAsimmetrico>";

     contenutoApplicativo = new StringBufferInputStream(xmlBodyRequest);

     // Aggiungo il contenuto applicativo alla richiesta
     body=           richiesta.addContenuto("body",           "application/xml",
contenutoApplicativo);


     System.out.println( "Richiesta Stato in corso..." );
     risposta = portaDelegata.send(richiesta);

     // valuta il body applicativo per effettuare ulteriori richieste
     Contenuto[] contenuti = risposta.getContenuti();

     String xmlBodyResponse =
org.apache.commons.io.IOUtils.toString(contenuti[0].getInputStream());
     XPath xpath = XPathFactory.newInstance().newXPath();
     XPathExpression xpathExpr = xpath.compile("//Esito/text()");
     Document doc =
DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(new
InputSource(new StringReader(xmlBodyResponse)));
     String result = (String)xpathExpr.evaluate(doc, XPathConstants.STRING);

     if (!result.equalsIgnoreCase("RISPOSTA_NON_PRONTA")) {
       isCompleted = true;
     }
   }
```

Esempio di utilizzo di Porta delegata con profilo asincrono simmetrico

```
   PortaDelegataBuilder pdb = PDDPortaDelegataBuilderFactory.newInstance().
            newPortaDelegataBuilder();
   PortaDelegata portaDelegata = pdb.newPortaDelegata(id_porta);

   Richiesta richiesta = portaDelegata.newRichiesta();
   String contenuto_applicativo = …
         InputStream   contenutoApplicativo   =   new   StringBufferInputStream(
contenuto_applicativo);

   Contenuto body =
           richiesta.addContenuto("body", "application/xml",
contenutoApplicativo);

   MyPortaApplicativa myPortaCallback =
```

```
                              new MyPortaApplicativa("MyPortaApplicativa");

   Risposta risposta = portaDelegata.send(richiesta, myPortaCallback);

   //la risposta ottenuta e' una ricevuta. La risposta applicativa arrivera' su
//myPortaCallback
```

4 Logical View

Come si evince dal *component diagram* riportato in figura 2, la Porta di Dominio è organizzata nei seguenti sottosistemi, ognuno dei quali è costituito da diversi componenti:

- **e-Gov Message Manager**
  - Envelope Builder
  - Envelope Validator
- **IPROXY**
  - SOAP Message Receiver
  - Indetificatore Porta Delegata
- **EPROXY**
  - e-Gov Message Receiver
  - Identificatore Porta Applicativa
- **Invoker**
  - Cooperation
  - Integration
- **Porta Delegata**
- **Porta Applicativa**
- **Security Manager**
  - WSS Manager
  - PKCS Manager
- **SICARegistryFacade**
- **Configuratore di Piattaforma**
- **Persistence**
- **Logger**

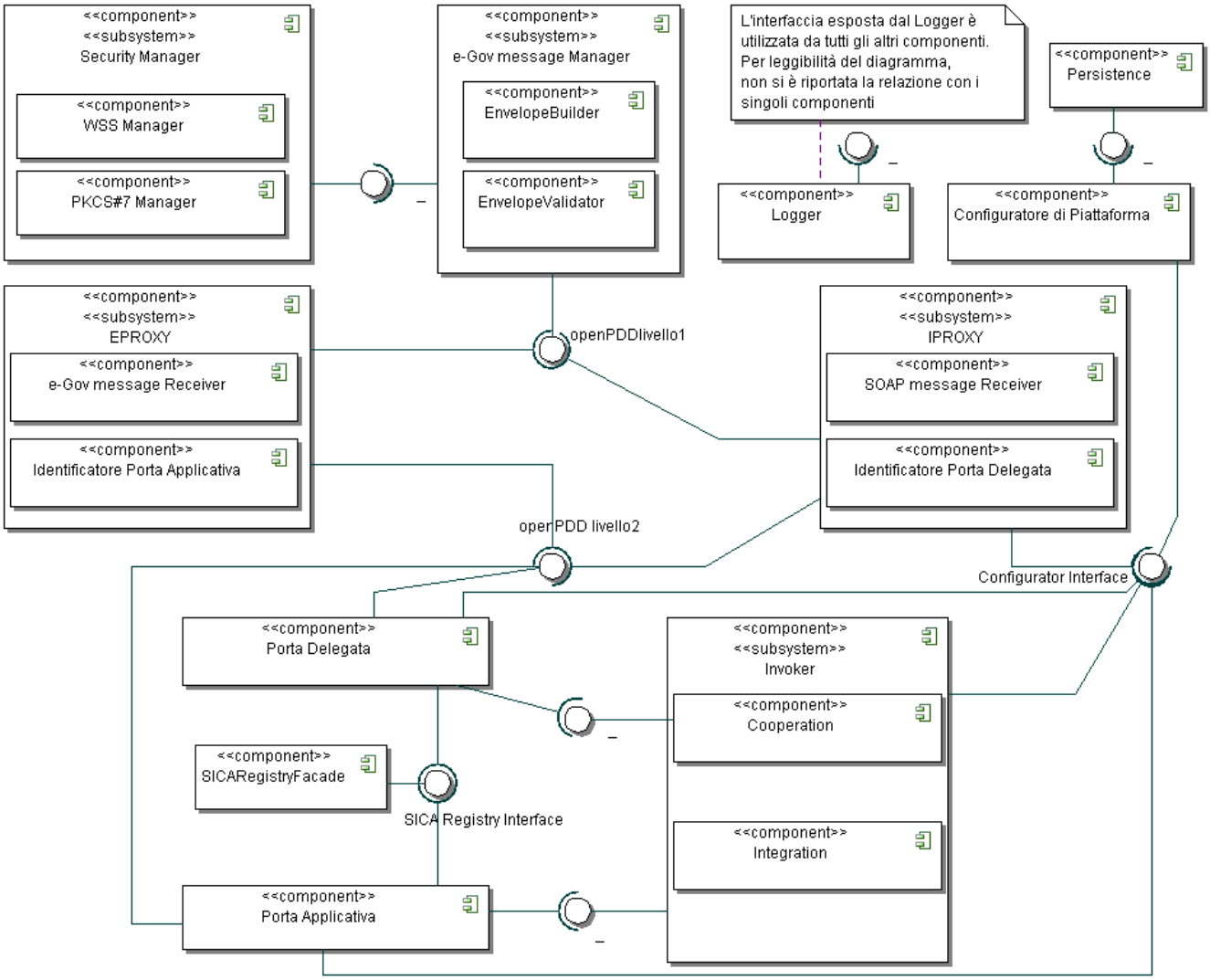


Figura 5 - Component Diagram della Porta di Dominio

4.1 e-Gov Message Manager

Questo componente implementa l'interfaccia OpenPDD livello 1, verso il sistema di trasporto dei messaggi SOAP, secondo lo schema a livelli precedentemente descritto. Esso consente una modellazione in termini di oggetti Java del documento XML costituente la busta e-Gov, permettendo la manipolazione del documento XML sottostante evitando l'accesso diretto al documento stesso.

Il componente e-Gov Message Manager agisce come "pre-processor" e "post-processor", rispettivamente, di messaggi e-Gov entranti ed uscenti dalla Porta di Dominio. Esso, interponendosi tra

Porte di Dominio esterne alla PDD SPICCA, ed il componente EPROXY della PDD SPICCA, nel caso di *ricezione* di un messaggio e-Gov, effettua una *serie* di "manipolazioni" del messaggio. Ogni diversa manipolazione è effettuata da un componente definito Handler. La *serie* di Handler invocata dall' e-Gov Message Manager è definita dal pattern Handler Chain. Viceversa, all'*invio* di un messaggio verso Porte di Dominio esterne alla PDD SPICCA, l'e-Gov message Manager effettuerà le stesse manipolazioni realizzate nel caso della ricezione, ma in ordine inverso. Ogni Handler sarà invocato dall'e-Gov Message Manager per realizzare operazioni sul messaggio e-Gov in ingresso quali, a titolo esemplificativo, il log del messaggio stesso, la verifica dell'autenticità del messaggio, etc.

In particolare, i componenti Envelope Builder ed Envelope Validator saranno implementati come degli Handler dell' e-Gov Message Manager.

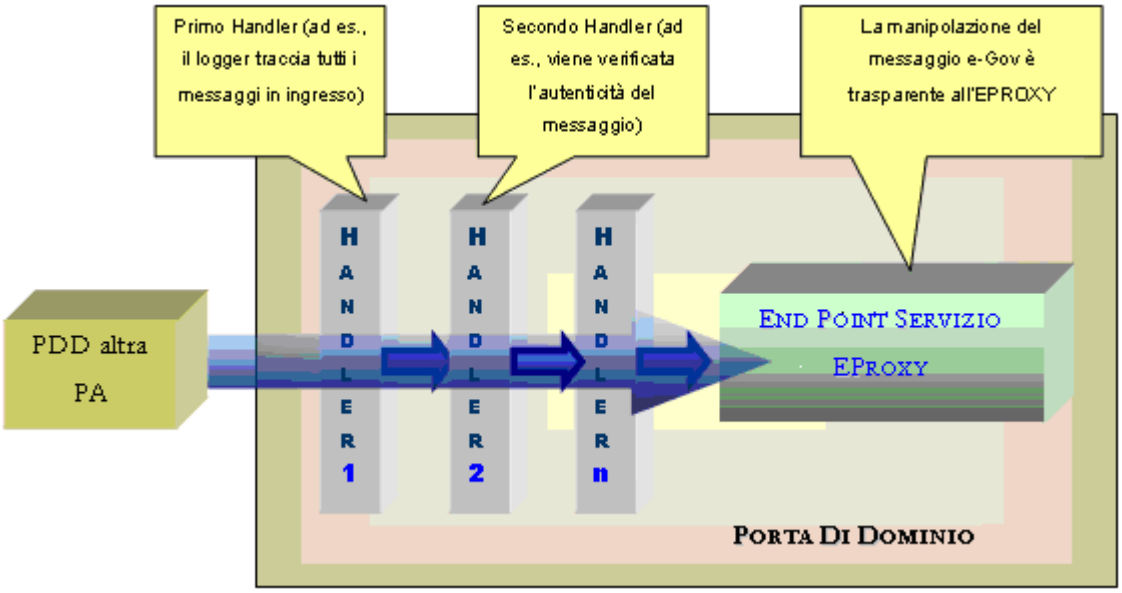


Figura 6 – Esempio di Handler Chain (Gestore dei trattamenti)

4.2 IPROXY

L'idea di base è quella di fornire la possibilità di invocare porte applicative remote utilizzando i servizi della PDD SPICCA *senza* possedere alcuna infrastruttura di PDD. In altre parole, si vuole dare la possibilità ad enti appartenenti alla PA su cui è installata la PDD SPICCA, di sfruttarne il servizio senza *sapere* di stare invocando un servizio di Porta Delegata:

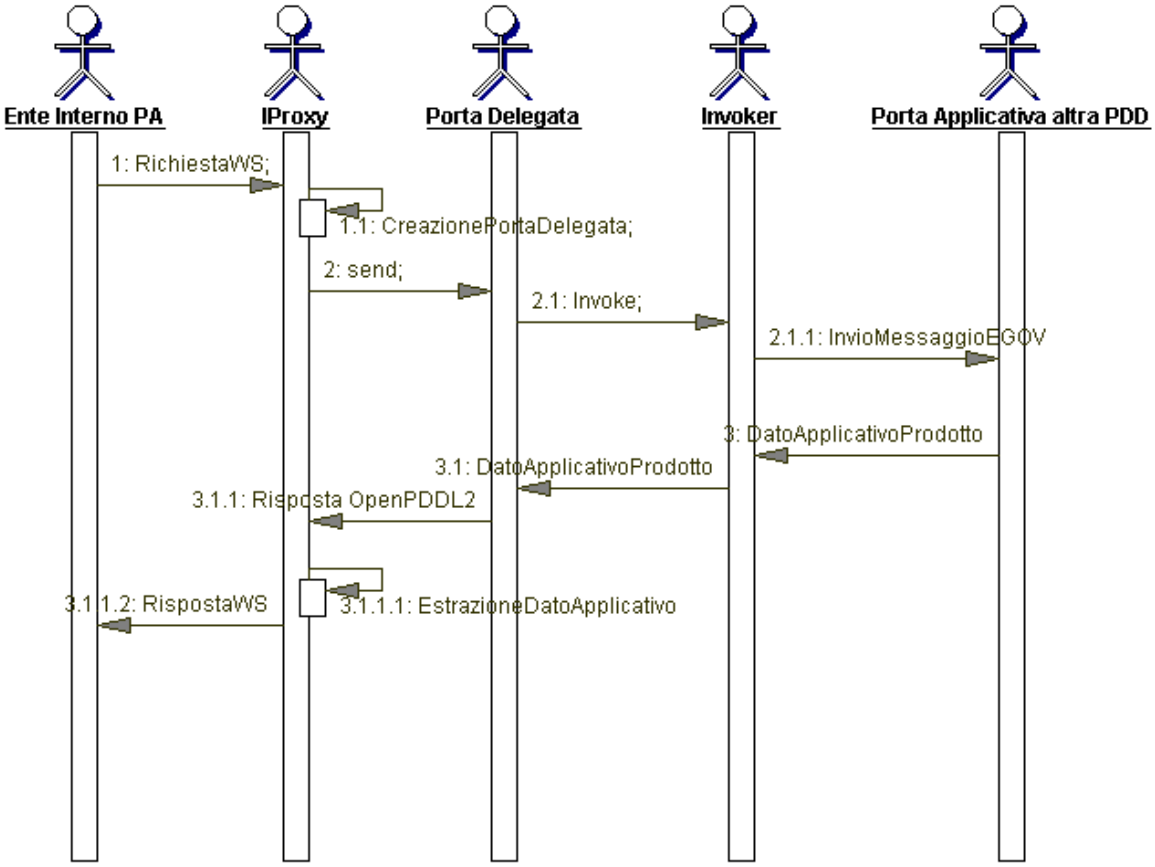


Figura 7- Flusso client da attore interno al dominio della PA

É importante notare come questo componente identifichi una funzionalità opzionale. Esso è da considerarsi solo uno strumento che potrebbe permettere ad attori interni alla PA di effettuare cooperazione applicativa sfruttando la PDD SPICCA installata presso la PA. Essendo l'IProxy stesso un client OpenPDDL2, non è in alcun modo legato alla capacità della PDD SPICCA di poter emettere e/o ricevere messaggi e-Gov.

Un'altra modalità di utilizzo dell'Iproxy potrebbe essere quello di utilizzarlo per le orchestrazioni che richiedono dati applicativi prodotti da porte applicative remote con profilo di comunicazione Sincrono. Ovviamente questa cosa la si può fare anche senza l'Iproxy; con l'Iproxy risulterebbe più semplice e meno costoso in termini di tempi di esecuzione.

Il componente IPROXY, implementato come Web Service, funge da proxy dinamico, ovvero identifica dinamicamente il Servizio Destinatario richiesto da un Sistema Informativo Locale avvalendosi di informazioni di supporto (ottenibili attraverso il componente "Configuratore di Piattaforma"). Prima di specificare le informazioni di supporto necessarie all'IPROXY ed esplicitarne il funzionamento, riprendiamo il concetto di Porta Delegata, brevemente accennato nella sezione relativa ad OpenPDD.

Una Porta Delegata permette ad un SIL (Sistema Informativo Locale) interno ad un dominio di usufruire di un servizio erogato da un SIL risiedente in un altro dominio. Essa, fondamentalmente, contiene le informazioni di trasporto necessarie alla Porta di Dominio per spedire una busta e-Gov, che conterrà i dati pervenuti al momento dell'invocazione della Porta Delegata stessa da parte di un SIL. Le informazioni di trasporto saranno utilizzate, oltre che per la costruzione della busta e-Gov, anche per ottenere l'indirizzo della porta di dominio destinataria a cui spedire la busta, attraverso una interrogazione del registro dei servizi. **Le informazioni di supporto necessarie all'IPROXY sono costituite dall'insieme delle Porte Delegate registrate sulla Porta di Dominio**.

L'IPROXY, ha il compito di ricevere i messaggi SOAP provenienti dai SIL interni al dominio, individuare il servizio desiderato dal SIL e demandare l' erogazione del servizio desiderato all'apposita Porta Delegata.

L'**utilizzo dell'IPROXY** da parte di un SIL interno al dominio è pressoché analoga a quella dell'utilizzo di un qualsiasi altro Servizio WEB, con una piccola eccezione: l'endpoint di destinazione. Il SIL interno costruisce uno stub di Web Services sulla base dell'Accordo di Servizio tra SIL interno e SIL risiedente nel dominio esterno, sulla base quindi del WSDL relativo al SIL risiedente nel dominio esterno. A questo punto, per usufruire dell'IPROXY per l'invocazione del SIL esterno, il SIL interno uitilizzerà come come endpoint di destinazione quello dell'IPROXY, anziché quello del SIL esterno al dominio.

Le predette funzionalità sono implementate dai componenti "SOAP Message Receiver" e "Identificatore Porta Delegata", descritti nel seguito.

4.3 SOAP Message Receiver

Questo componente si occupa di prendere in carico le richieste dei SIL per le Porte Delegate registrate sulla porta di dominio. Per ogni richiesta ricevuta, viene effettuata l'autenticazione del mittente a livello trasporto, tramite il certificato X.509 usato per la connessione SSL. Tale componente è in grado di gestire richieste tipo SOAP ed è dunque compatibile con client di tipo Web Services. Si osservi che tale

componente consente di evitare che i SIL debbano essere modificati per dialogare con la Porta di Dominio: lo faranno semplicemente invocando un Web Services, ed, inoltre, non hanno visibilità dell'infrastruttura di trasporto.

4.4 Identificatore Porta Delegata

La principale funzione di questo componente è quella di identificare a quale Porta Delegata sia relativa la richiesta ricevuta dal componente SOAP message receiver. A tal fine tale componente interagisce con il componente "Configuratore di Piattaforma", da cui evince le informazioni relative alle Porte Delegate registrate. L'indicizzazione delle Porte Delegate è effettuata considerando la parte finale dell'URI di accesso alla Porta di Dominio, utilizzata dal SIL che richiede un servizio. Ad esempio, supponendo che l'indirizzo di accesso alla Porta di Dominio sia configurato come "http://www.<spicca>.it/pdd", il SIL per invocare una porta delegata registrata con l'URI "ServizioTest" dovrà utilizzare l'indirizzo http://www.<spicca>.it/pdd/IPROXY?id_servizio =ServizioTest

4.5 EPROXY

Il componente EPROXY, anch'esso implementato come un Web Services, rappresenta il punto d'ingresso offerto dalla Porta di Dominio ad altre Porte di Dominio, ovvero le altre PDD che vogliono usufruire di servizi esposti dalla PDD SPICCA dovranno inviare le loro richieste a tale componente. In modo analogo all'IPROXY, l'obiettivo dell'EPROXY è identificare dinamicamente il Sistema Informativo Locale erogatore di un servizio richiesto da un'altra PDD, appoggiandosi anch'esso ad informazioni di supporto (ottenibili attraverso il componente Configuratore di Piattaforma). Prima di specificare le informazioni di supporto necessarie all'EPROXY ed esplicitarne il funzionamento, riprendiamo il concetto di Porta Applicativa.

Una Porta Applicativa consente ad un sistema informativo risiedente in un altro dominio di usufruire di un servizio erogato da un SIL (Sistema Informativo Locale) interno ad un dominio. Essa, fondamentalmente, contiene le informazioni necessarie alla Porta di Dominio per individuare il SIL interno al dominio cui inoltrare una busta e-Gov ricevuta. **Le informazioni di supporto necessarie all'EPROXY sono costituite dall'insieme delle Porte Applicative registrate sulla Porta di Dominio.**

L'EPROXY, in modo speculare all'IPROXY, ha il compito di ricevere i messaggi e-Gov provenienti da altre PDD, individuare il SIL interno al dominio erogatore del servizio richiesto e demandare l'erogazione del servizio desiderato all'apposita Porta Applicativa.

Le predette funzionalità sono implementate dai componenti "e-Gov Message Receiver" e "Identificatore Porta Applicativa", descritti nel seguito.

4.6 e-Gov Message Receiver

Questo componente si occupa di prendere in carico le buste e-Gov in arrivo da altre Porte di Dominio. Per ogni richiesta ricevuta viene effettuata l'autenticazione del mittente, con le modalità specificate nell'Accordo di Servizio.

4.7 Identificatore Porta Applicativa

La principale funzione di questo componente è quella dell'identificazione della Porta Applicativa a cui è destinata la busta e-Gov pervenuta. Per identificare il servizio applicativo da invocare, il componente utilizza le apposite informazioni presenti nella busta e-Gov pervenuta ed interagisce con il modulo "Configuratore di Piattaforma", da cui evince le informazioni relative alle Porte Applicative

4.8 Invoker

A tale sottosistema è delegato l'invio di messaggi da parte delle Porte Delegate e Applicative. Più precisamente, una Porta Delegata demanda a tale componente l'invio di un messaggio e-Gov destinato ad un'altra Porta Applicativa, mentre una Porta Applicativa delega a tale componente l'invio di un messaggio, SOAP ad esempio, ad un SIL interno al Dominio.

Le due predette funzionalità sono realizzate da due componenti costituenti l'Invoker, Integration Invoker e Cooperation Invoker, di seguito descritti.

4.9 IntegrationInvoker

Questo componente ha il compito di consegnare il contenuto applicativo delle buste e-Gov pervenute alla PDD, ormai già sbustate dal componente "e-Gov Message Manager", ai SIL destinatari interni al Dominio di Cooperazione. L'IntegrationInvoker viene utilizzato *solo* dalla PortaApplicativa che ha ricevuto il messaggio e-Gov. Quest'azione potrebbe essere fatta già direttamente dalla Porta Applicativa stessa. Si è deciso di inserire questo ulteriore strato al fine di rendere completamente "pluggubile" l'integrazione con i sistemi informativi interni alla PA. Qualora il dato applicativo del SIL venga prodotto da un WebService interno, la Porta Applicativa verrà configurata per utilizzare un Invoker per Web Service; per un dato applicativo prodotto da un EJB, la Porta Applicativa verrà configurata per utilizzare un Invoker in grado di invocare una RemoteInterface, ecc...

La PDD SPICCA implementa in maniera nativa invoker per i componenti di tipo WEBSERVICE. Qualora dovesse nascere l'esigenza di integrarsi con servizi che implementano protocolli proprietari, basterà implementare l'interfaccia Invoker, registrarlo sulla base dati (via configuratore di piattaforma) e associarlo alla relativa porta applicativa; il tutto senza modificare la logica della porta applicativa stessa.

4.10 CooperationInvoker

Questo componente ha il compito di inviare le buste e-Gov, già costruite per mezzo del componente "e-Gov Message Manager", alle Porte di Dominio competenti. Così come detto per l'IntegrationInvoker, la Porta Delegata potrebbe effettuare direttamente l'invocazione della porta applicativa remota. Le specifiche del CNIPA definiscono il trasporto SOAP over HTTP/S come quello di riferimento. Ciò non toglie che nel futuro possano svilupparsi servizi basati su SOAP ma via FTP o SMTP. Lo scopo del disaccoppiare la logica della Porta Delegata dal trasporto sottostante è proprio quello di permettere un facile adeguamento a possibili nuove esigenze.

Qualora si voglia sviluppare una porta delegata per un servizio remoto via SOAP over FTP, ad esempio, basterà implementare l'interfaccia Invoker, registrare il nuovo componente nel db (via configuratore di piattaforma) ed associarlo alla porta delegata stessa.

4.11 Porta Applicativa

Questo componente implementa le interfacce definite dal livello 2 di OpenPDD relative all'**erogazione dei servizi esposti dalla Porta Di Dominio**. Come precedentemente accennato, la Porta Applicativa è la "componente server" di una PDD ed è interrogata dalle componenti Porta Delegata di altri domini.

Ogni Porta Applicativa è associata ad una specifica Porta Delegata, che realizza un particolare profilo di collaborazione. In pratica come meglio illustrato nella sezione relativa alla vista dei casi d'uso, la creazione di una Porta Applicativa è sempre conseguenza di una invocazione fatta da una Porta Delegata di un altro dominio, relativa ad uno specifico profilo di collaborazione. La Porta Applicativa ha il compito di soddisfare la particolare richiesta di servizio e restituire la risposta applicativa alla Porta Delegata chiamante. E' in quest'ottica che la Porta Applicativa realizza le funzionalità di componente server di una PDD.

Relativamente ad OpenPDD, le interfacce implementate dal componente Porta Applicativa sono le seguenti:

- org.openpdd.level2.PortaApplicativaBinderFactory
- org.openpdd.level2.PortaApplicativaBinder
- org.openpdd.level2.PortaApplicativa

In definitiva, attraverso il componente "Porta Applicativa" sarà possibile:

- ricevere messaggi da Porte Delegate;
- inviare messaggi a Porte Delegate;
- la realizzazione dei diversi profili di comunicazione.

E' da notare come la PDD SPICCA abbia delle implementazioni di riferimento per porte applicative in grado di cooperare secondo tutti i profili di comunicazione definiti dalle specifiche della busta e-Gov: Profilo OneWay, Profilo Sincrono, Profilo Asincrono Asimmetrico, Profilo Asincrono Simmetrico.

4.12 Porta Delegata

Questo componente implementa le interfacce definite dal livello 2 di OpenPDD relative alla **fruizione di servizi remoti esposti da una Porta Applicativa**.

La Porta Delegata rappresenta la “componente client” di una Porta di Dominio, e si occupa di soddisfare le richieste provenienti da Sistemi Informativi Locali interni al dominio: essa ha il compito di inoltrare la richiesta di servizio pervenuta alla Porta di Dominio dal SIL, alla Porta Applicativa ad essa associata. Sarà poi quest’ultima ad effettuare le operazioni necessarie a soddisfare la richiesta ed a restituire la risposta alla Porta Delegata, che la consegnerà, attraverso i componenti del implementanti il livello 1 di OpenPDD, al SIL che aveva richiesto il servizio.

All’interno della Porta di Dominio dovranno essere presenti diverse implementazioni di Porta Delegata, ognuna dedita alla realizzazione di uno specifico profilo di collaborazione (OneWay, Sincrono, ...).

Relativamente ad OpenPDD, le interfacce implementate dal componente Porta Delegata sono le seguenti:

- org.openpdd.level2.PortaDelegataBuilderFactory
- org.openpdd.level2.PortaDelegataBuilder
- org.openpdd.level2.PortaDelegata

In definitiva, attraverso il componente “Porta Delegata” sarà possibile:

- inviare richieste a Porte Applicative;
- ottenere risposte da Porte Applicative;
- la realizzazione dei diversi profili di comunicazione.

E’ da notare come la PDD SPICCA abbia delle implementazioni di riferimento per porte delegate in grado di invocare porte applicative remote secondo tutti i profili di comunicazione definiti dalle specifiche della busta e-Gov: Profilo OneWay, Profilo Sincrono, Profilo Asincrono Asimmetrico, Profilo Asincrono Simmetrico.

4.13 SecurityManager

Tale sottosistema si occupa della gestione della Sicurezza al livello messaggio e-Gov. Esso è costituito dai due componenti descritti di seguito.

4.13.1 WSS Manager

Questo componente ha il compito di creare e validare l'header WS-Security, in conformità alle specifiche definite nello standard OASIS Web Services Security che definisce un meccanismo per l'estensione di messaggi SOAP finalizzato ad assicurare l'integrità, la riservatezza e l'autenticazione del singolo messaggio.

Il componente è in grado di gestire i seguenti token di autenticazione/attributo:

- Token X.509
- Token SAML
- Token Username/Password

Questo componente viene utilizzato dagli handler per la gestione del ws-security qualora le relative porte applicative e/o delegate fossero configurate per dover trattare le buste in ingresso/uscita attraverso azioni di applicazione/verifica di firma digitale; creazione del token SAML; estrazione degli attributi dal token di attributo SAML stesso, ecc.

Tutta la gestione dei certificati e dei keystore è completamente demandata all'infrastruttura di sicurezza con la quale il componente è completamente integrato.

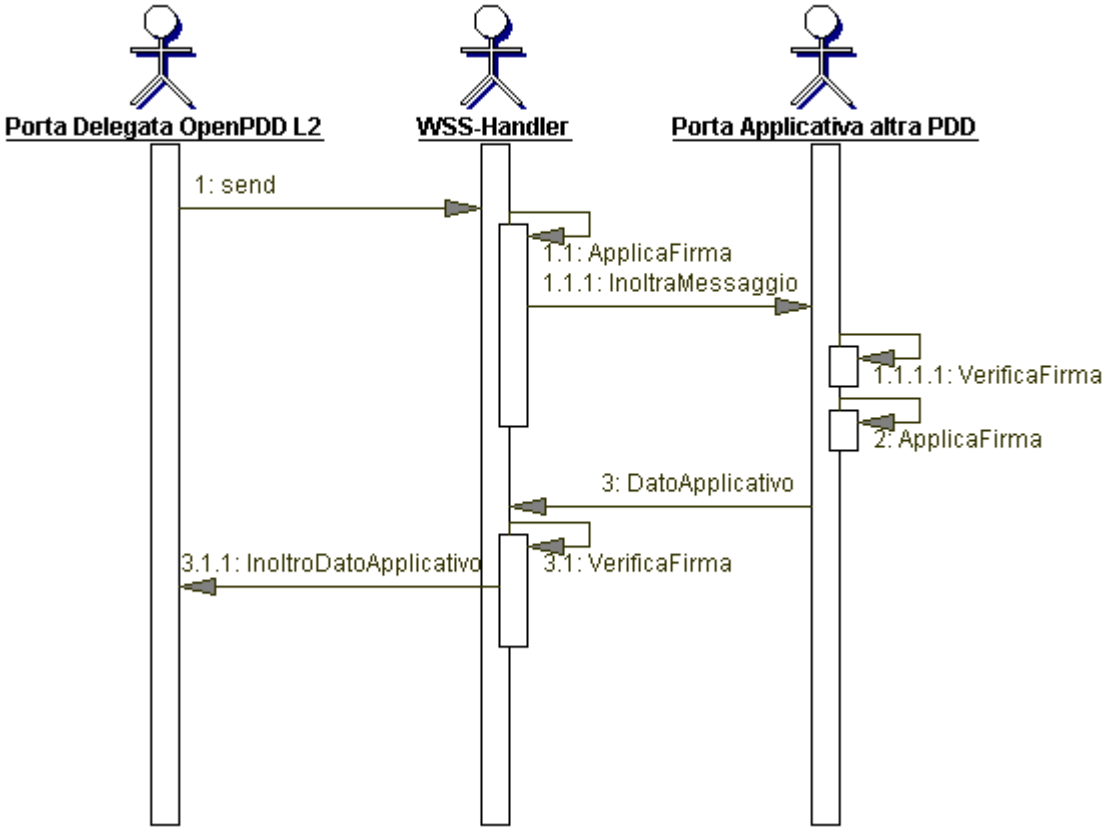


Figura 8- Flusso Sincrono con Applicazione/Verifica Firma Digitale

Dal diagramma precedente, è importante sottolineare come la porta delegata da cui parte la richiesta, non si occupa di problematiche legate alla firma elettronica. La Porta Delegata invia il messaggio direttamente all'endpoint della porta applicativa. L'intervento dell'wss-handler è totalmente trasparente al livello applicativo di openpddl2.

4.13.2 PKCS#7 Manager

Le specifiche SPCOOP prevedono che per la verifica dell'autenticità e della provenienza degli allegati ai messaggi e-Gov, venga utilizzato il formato standard per la sintassi dei messaggi cifrati denominato PKCS#7. Il compito del componente PKCS#7 manager è quello di assicurare la riservatezza e l'integrità degli allegati utilizzando lo standard PKCS#7 per le operazioni di firma, verifica ed eventualmente cifratura (e decifratura) degli allegati scambiati dalle PDD.

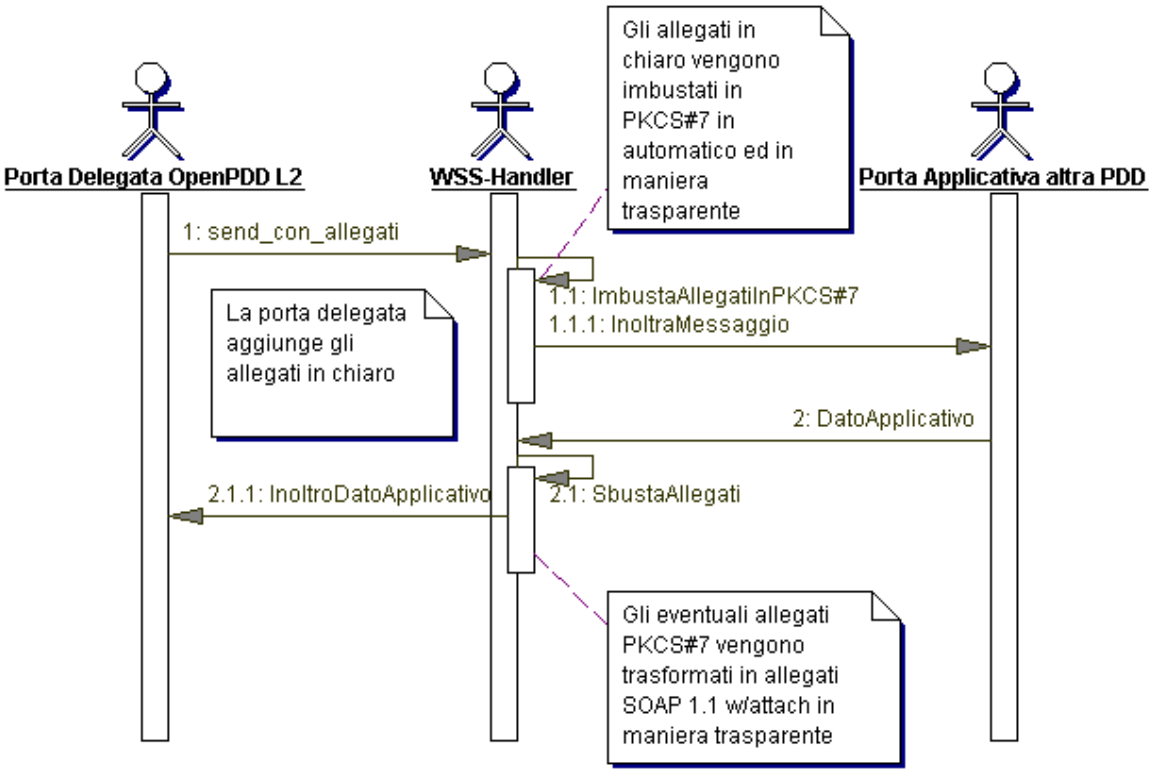


Figura 9- Esempio di invio/ricezione di allegati PKCS#7

4.14 SICARegistryFacade

Per poter gestire l'accordo di servizio in tutte le sue parti, è necessario un componente di piattaforma che permetta all'applicazione di gestione dell'accordo di servizio stesso, di poter interfacciarsi in maniera semplice e veloce con il registro SICA Secondario SPICCA per la pubblicazione, modifica, dismissione etc., di un servizio applicativo. Questo componente ha il compito di semplificare l'interfacciamento al Registro SICA esponendo un facade per l'integrazione con il registro SICA stesso.

In base a quanto definito dalle specifiche CNIPA riguardo il Registro dei servizi SICA, tutte le funzionalità del registro devono essere utilizzabili anche attraverso interfacce web-service. Dunque questo componente sarà fondamentalmente un helper sugli stub applicativi dei web service del registro SICA su cui la PDD SPICCA andrà a pubblicare e/o sottoscrivere qualche servizio.

4.15 Configuratore di piattaforma

Questo componente si occupa della gestione di tutti i parametri di configurazione di piattaforma della Porta Di Dominio. Il Configuratore di piattaforma legge tutti i parametri di configurazione presenti nella base dati e li carica in memoria allo startup dell'application server. Essendo implementato secondo il pattern Singleton, sarà presente, per ogni istanza di Virtual Machine Java, un'unica istanza di configuratore a cui tutti gli altri componenti di sistema potranno attingere.

4.16 Persistence

Questo componente si occupa della gestione dei dati persistenti mascherando ai componenti utilizzatori il meccanismo di fatto utilizzato per la gestione dei dati persistenti.

4.17 Logger

Questo componente espone delle API per tutte le operazioni di logging e tracing. Il Logger astrae il logger Log4J offrendo una serie di helper per un log facilitato a tutti i componenti di piattaforma. Attraverso il configuratore di piattaforma, sarà possibile utilizzare un helper LoggerUtil che fornirà un riferimento al logger desiderato. Tutti i logger saranno configurati in base dati e caricati in memoria dal configuratore di sistema.

5 Use-Case View

In questa sezione vengono rappresentati i casi d'uso più importanti dal punto di vista architetturale, ovvero i casi d'uso la cui realizzazione coinvolge una parte significativa dei componenti architetturali individuati.

5.1 Realizzazione dei Casi d'Uso

I casi d'uso riportati, sono quelli relativi ai diversi profili di collaborazione che dovranno essere supportati dalla Porta di Dominio. Al fine di presentare le modalità di collaborazione dei diversi componenti architetturali per la realizzazione dei casi d'uso, si utilizzeranno i Sequenze Diagram UML che evidenzieranno, tra l'altro, l'ordine temporale delle diverse azioni eseguite dai componenti stessi.

5.1.1 Profilo di collaborazione One-Way – ruolo Server

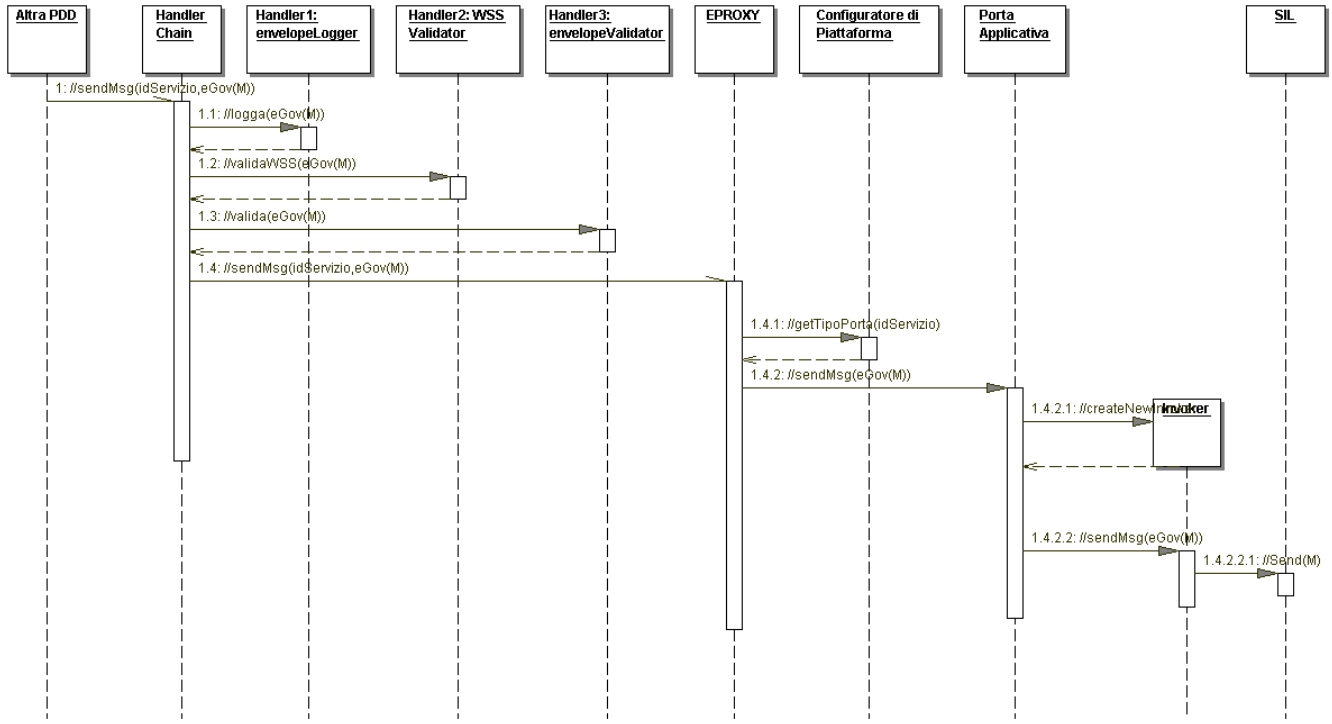


Figura 10 – Profilo di collaborazione One-Way ruolo Server: sequence diagram

1) Una Porta di Dominio esterna al Dominio SPICCA stabilisce una connessione con la Porta Di Dominio SPICCA ed effettua una richiesta per un determinato servizio, opportunamente identificato, interno al dominio SPICCA

2) Il messaggio e-Gov viene inviato all'Handler Chain che ha il compito di richiamare diversi Handler al fine di:
    a. loggare il messaggio e-Gov ricevuto
    b. validare l'Header WSS del messaggio e-Gov ricevuto
    c. validare il messaggio e-Gov ricevuto
3) La richiesta viene quindi ricevuta dall'EProxy che si occupa di recuperare dal "Configuratore di Piattaforma" le informazioni relative al tipo di Porta Applicativa da istanziare in base al servizio richiesto (idServizio)
4) L'EProxy invia dunque il messaggio e-Gov alla Porta Applicativa che a sua volta si preoccupa di istanziare l'Invoker (futuro responsabile della chiamata verso il SIL)
5) L'Invoker si fa carico di prelevare il contenuto applicativo dal messaggio e-Gov ed effettuare la chiamata verso il SIL.

5.1.2 Profilo di collaborazione One-Way – ruolo Client

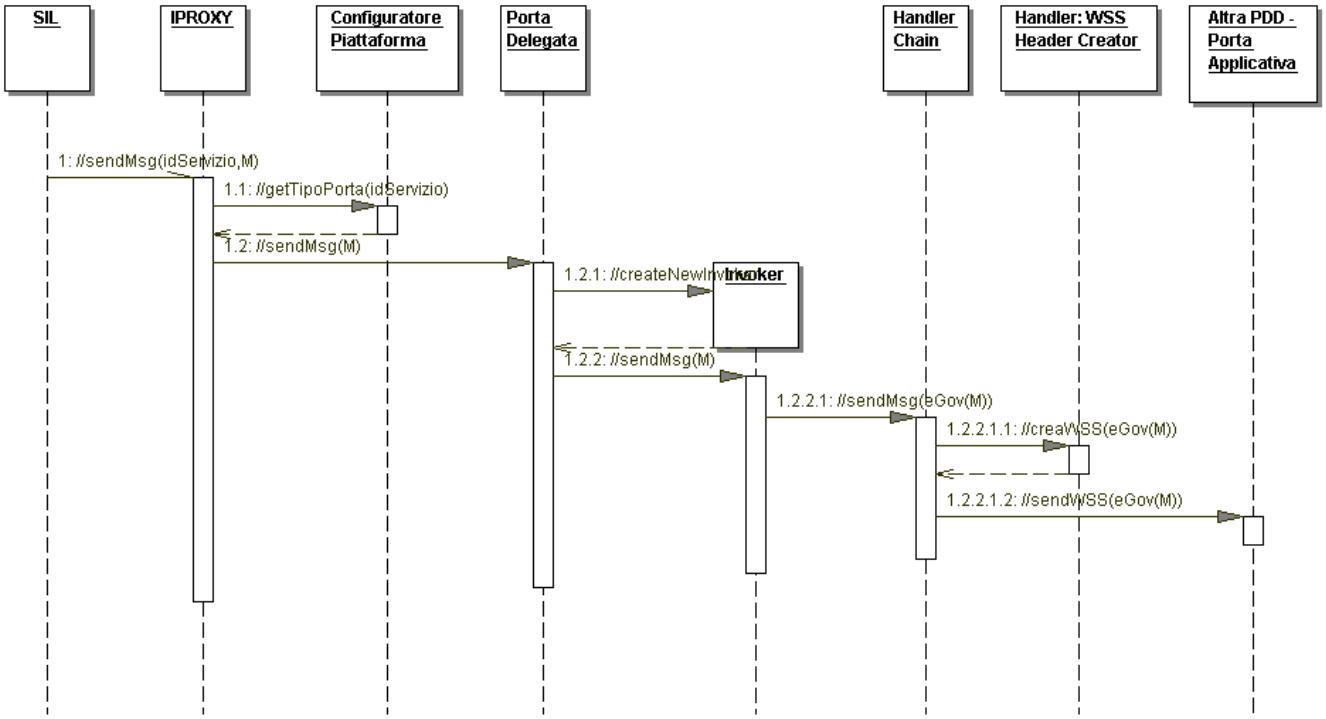


Figura 11 – Profilo di collaborazione One-Way ruolo Client: sequence diagram

1) Un SIL effettua una richiesta per un determinato servizio (idServizio) all'IProxy per una Porta di Dominio esterna al dominio SPICCA
2) L'IProxy si occupa di recuperare dal "Configuratore di Piattaforma" le informazioni relative alla Porta Delegata da istanziare in base al servizio richiesto (idServizio)
3) L'IProxy invia il messaggio M alla Porta Delegata che, sapendo quale Porta Applicativa dovrà contattare, ha l'onere di istanziare l'Invoker (responsabile dell'effettiva chiamata)
4) L'Invoker si fa carico di imbustare il messaggio M in un messaggio e-Gov ed effettua la chiamata verso la Porta di Dominio esterna al dominio SPICCA cui invia il messaggio e-Gov costruito
5) Il messaggio e-Gov passa viene inviato all'Handler Chain che ha il compito di costruire l'Header WSS del messaggio stesso
6) Il messaggio e-Gov viene quindi inviato alla Porta di Dominio esterna al Dominio SPICCA

5.1.3 Gestione SOAP with Attachments

La porta di dominio permette l'invio e la ricezione di allegati ai messaggi e-Gov scambiati durante una cooperazione applicativa. Gli allegati devono essere inviati secondo quanto definito dallo standard SOAP with Attachments. Viene supportato il WS-I Attach profile 1.0.

Come descritto nelle specifiche CNIPA della Busta e-Gov, la presenza di attach implica la presenza del XML element Descrizione all'interno del body applicativo del messaggio e-Gov. Di seguito viene riportato un sequence diagram di una tipico scambio di messaggi e-Gov con attach. Nel seguente diagramma sono omessi i flussi per la gestione degli errori e si assume che vi siano attach in ingresso e in uscita.

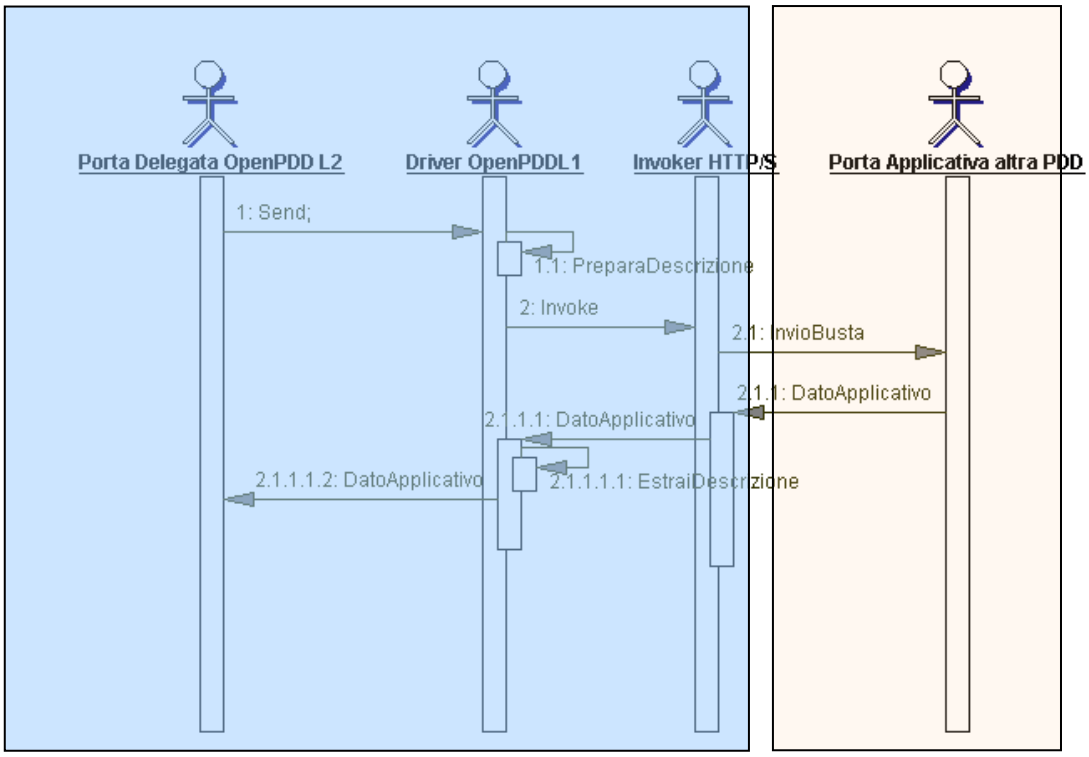


Figura 12- Scenario di invocazione sincrona con attach in ingresso e in uscita

1. Il programmatore di Livello 2 istanzia la relativa porta delegata (precedentemente configurata nel DB), allega uno o più allegati (come semplici stream binari). Di seguito è mostrato un esempio di aggiunta di un allegato alla richiesta della Porta delegata:

```
...
File f = new File("qualche_file.txt");
FileInputStream s = new FileInputStream(f);

Contenuto c = richiesta.addContenuto("allegato","text/plain", s);

c.setPosizioneSchema("cid:"+f.getName());
c.setLingua("Italiano");
c.setRuolo("Richiesta");
c.setTitolo("Allegato immagine");
c.setVersione("1.0");

Risposta r = portaDelegata.send(richiesta);
...
```

1.1. Il driver della PDD SPICCA controlla la presenza di attach. Quindi crea una struttura di "Descrizione" e-Gov con i dati presenti nella richiesta e la assegna a quello che sarà il Body applicativo. Quindi aggiunge tutti i Contenuti OpenPDDL2 come attachment part SOAP.

2. Il driver della PDD SPICCA istanzia il relativo Invoker associato alla Porta Delegata ed invoca il metodo Invoke() dell'invoker stesso.

2.1. L'invoker crea una chiamata SOAP per l'endpoint della porta applicativa e trasmette il messaggio SOAP completo di attach SOAP w/attachment utilizzando le API del SOAP server configurato per la PDD SPICCA; si creano gli Attachment Part costruendoli in base agli inputstream ottenuti dallo strato OpenPDDL2. E' da notare come l'header e-Gov non venga inserito in questa fase ma da un trattamento agganciato al flusso Client-Request (ovvero in uscita verso una porta applicativa).

2.1.1. La Porta applicativa remota ritorna (assumendo un esito positivo) il dato applicativo prodotto con, si assume, un certo numero di attach.

2.1.1.1. L'invoker ritorna la risposta SOAP al driver di Livello 1.

2.1.1.1.1. Il driver di livello 1, controlla la presenza di attach in risposta, estrae le Descrizioni e le singole attachment part e costruisce un oggetto Risposta (specifiche OpenPDDL2) dove vengono mappati tutti gli attach in termini di stream con associate le singole descrizioni.

2.1.1.1.2. La porta delegata consuma il dato applicativo prodotto e i vari attach leggendoli come semplici stream di dati.

6 Implementation view

In questo paragrafo viene riportata la struttura complessiva del modello di implementazione, la decomposizione del software in sottosistemi con particolare riferimento alle relazioni tra i componenti della Porta Di Dominio SPICCA e le interfacce OpenPDD.

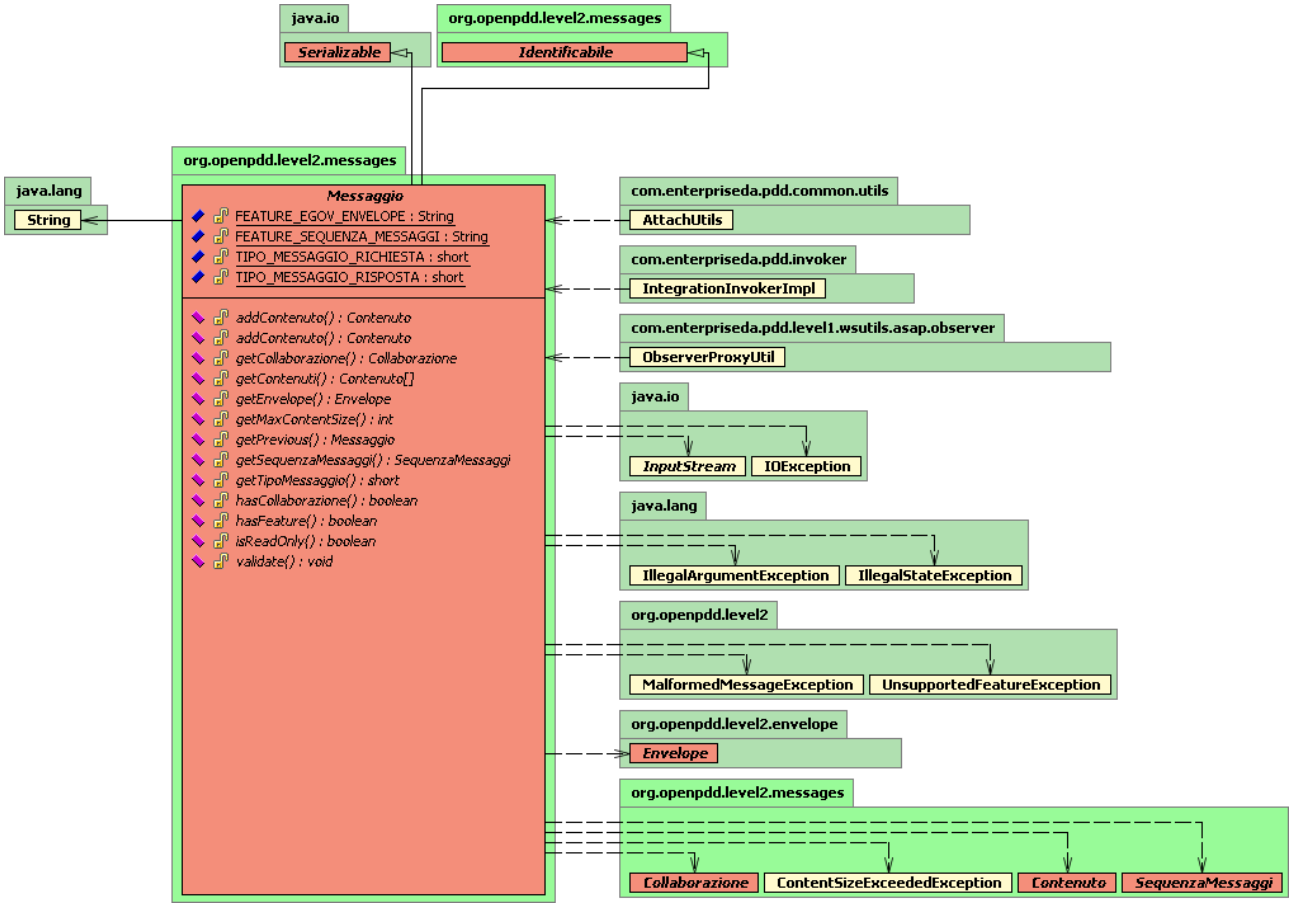


Figura 13 - Relazioni tra l'interfaccia org.openpdd.level2.Messaggio e i componenti della Porta Di Dominio SPICCA

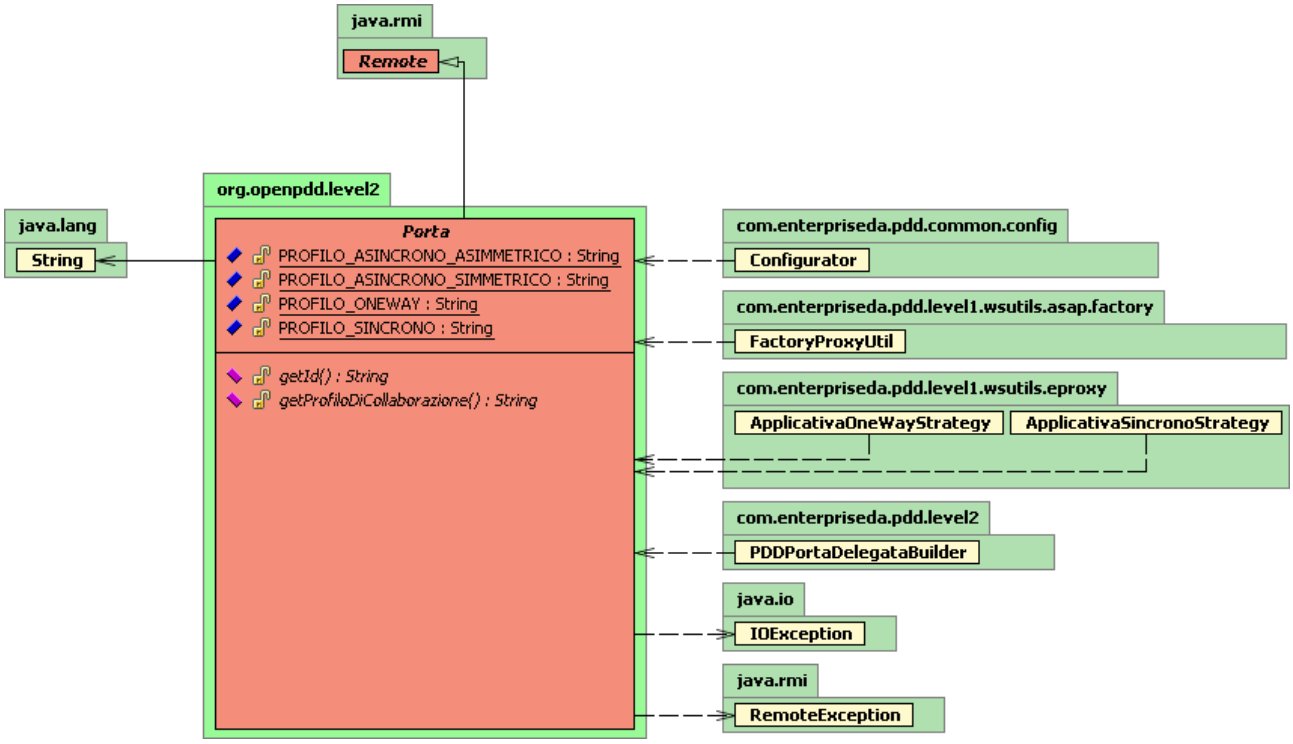


Figura 14 – Relazioni tra l'interfaccia org.openpdd.level2.Porta e i componenti della Porta Di Dominio SPICCA

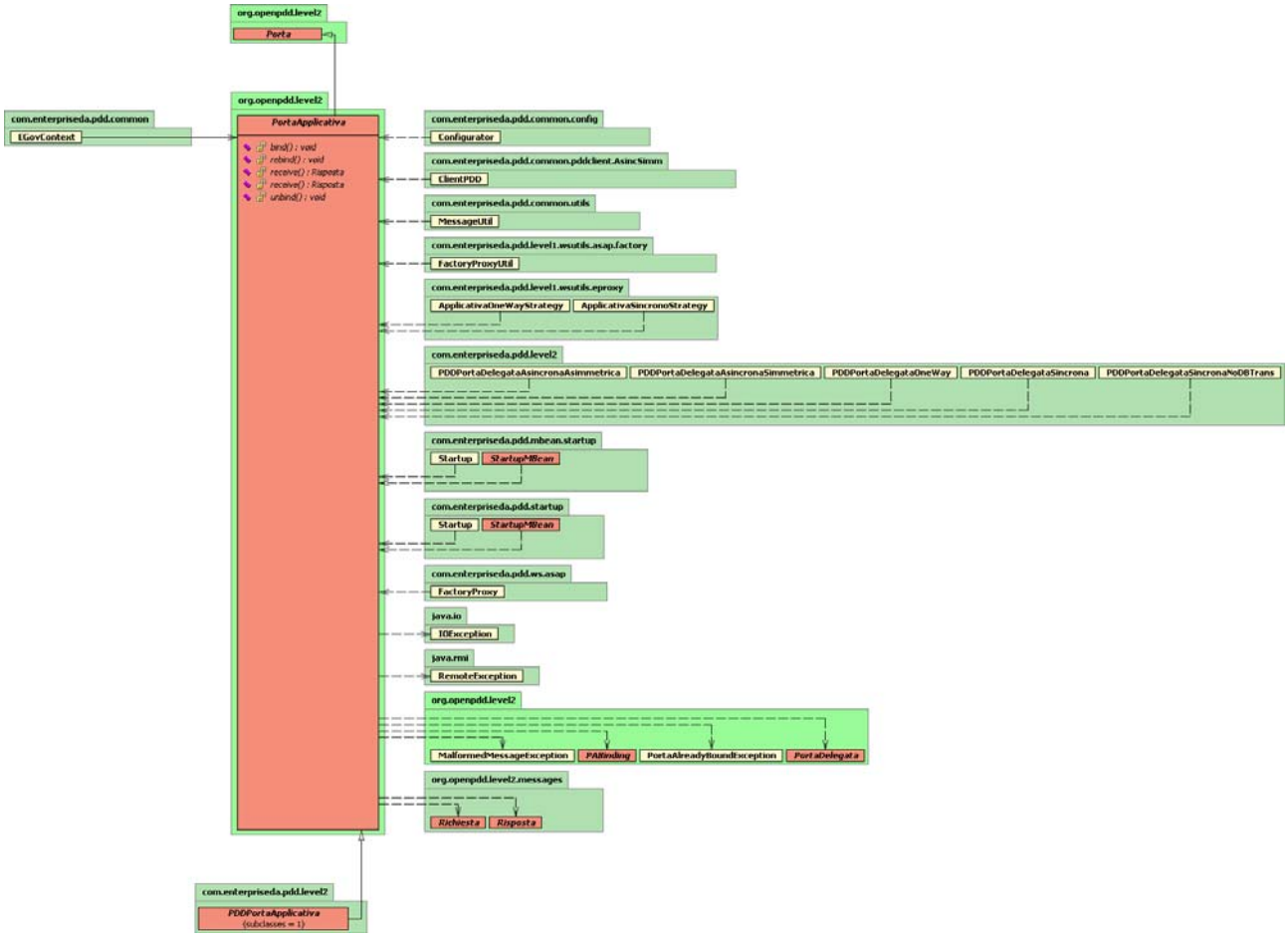


Figura 15 - Relazioni tra l'interfaccia org.openpdd.level2.PortaApplicativa e i componenti della Porta Di Dominio SPICCA

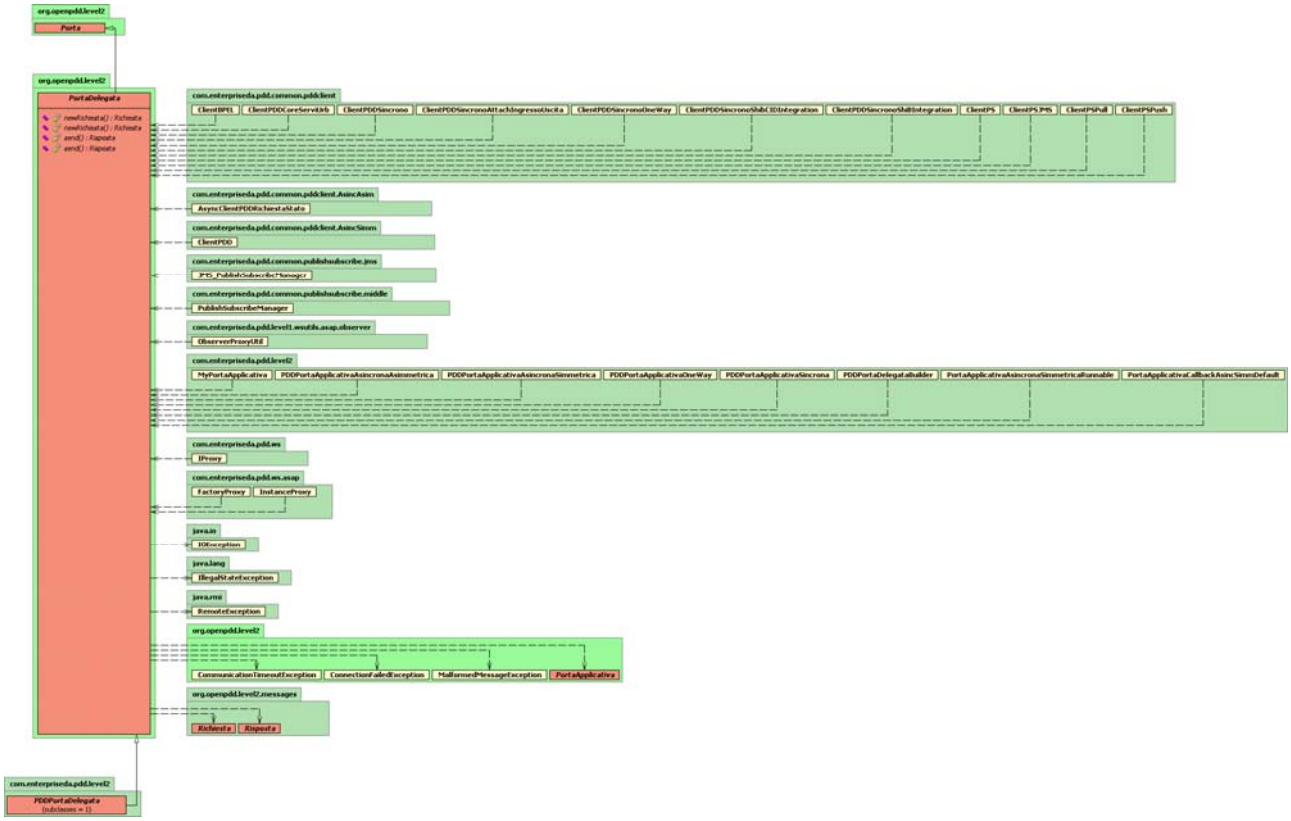


Figura 16 - Relazioni tra l'interfaccia org.openpdd.level2.PortaDelegata e i componenti della Porta Di Dominio SPICCA

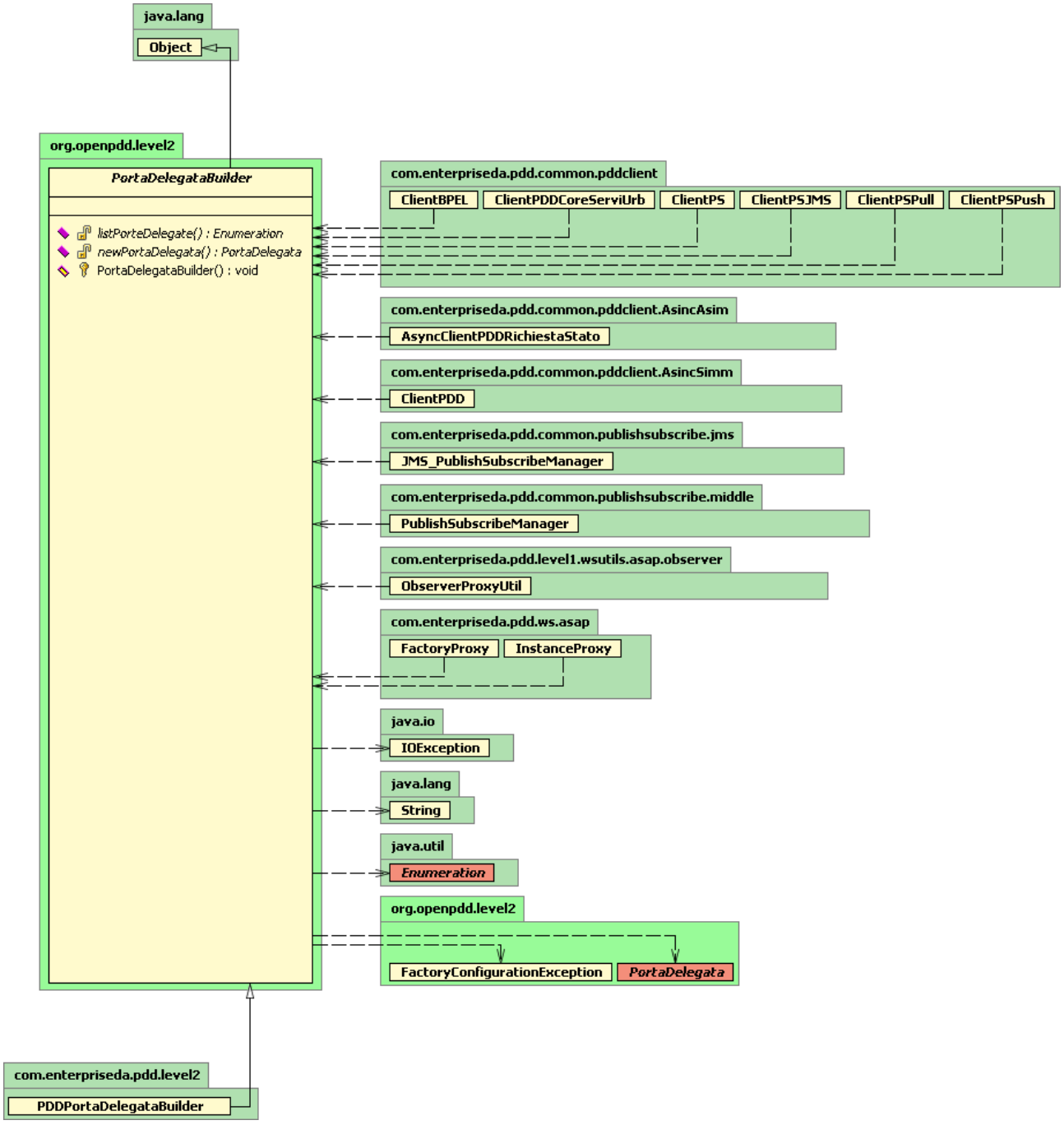
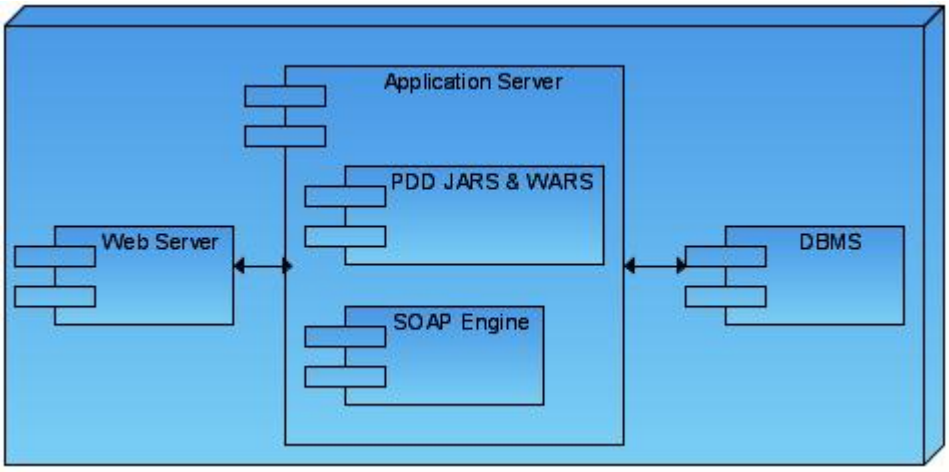


Figura 17 - Relazioni tra l'interfaccia org.openpdd.level2.PortaDelegataBuilder e tra i componenti della PDD-SPICCA

7 Deployment View

In questo paragrafo viene riportata la configurazione in cui la Porta di Dominio sarà rilasciata, mediante il Deployment Diagram in figura. Com'è possibile notare, a livello di deployment, la PDD è costituita da un insieme archivi jar e war, rispettivamente librerie JAVA ed applicazioni WEB in teconologia Java, che saranno deployate in un Application Server J2EE. Tali librerie e applicazioni necessitano di un SOAP Engine, AXIS per l'esattezza, anch'esso deployato nello stesso Application Server. Inoltre, l'Application Server mette a disposizione della PDD il canale di comunicazione con il DBMS, attraverso i Data Source J2EE, creando dei pool dinamici di connessioni. Un ulteriore elemento della PDD è costituito dal WebServer le cui funzionalità sono realizzate dall'Application Server stesso o, in alternativa, è possibile utilizzare un Web Server esterno, ad esempio Apache httpd, che colloquia con l'Application Server J2EE attraverso il protocollo AJP.



8 Data View

Nella PDD-SPICCA, la base dati ricopre un ruolo fondamentale. In base dati sono inseriti tutti i parametri di configurazione della PDD-SPICCA, tutte le porte applicative/delegate, tutti i trattamenti afferenti le singole porte, tutti i componenti software utilizzati, tutte le transazioni in ingresso e in uscita, ecc...; nel prosieguo del paragrafo verranno riportate le principali entità con la relativa descrizione dell'utilizzo che ne fa la PDD-SPICCA.

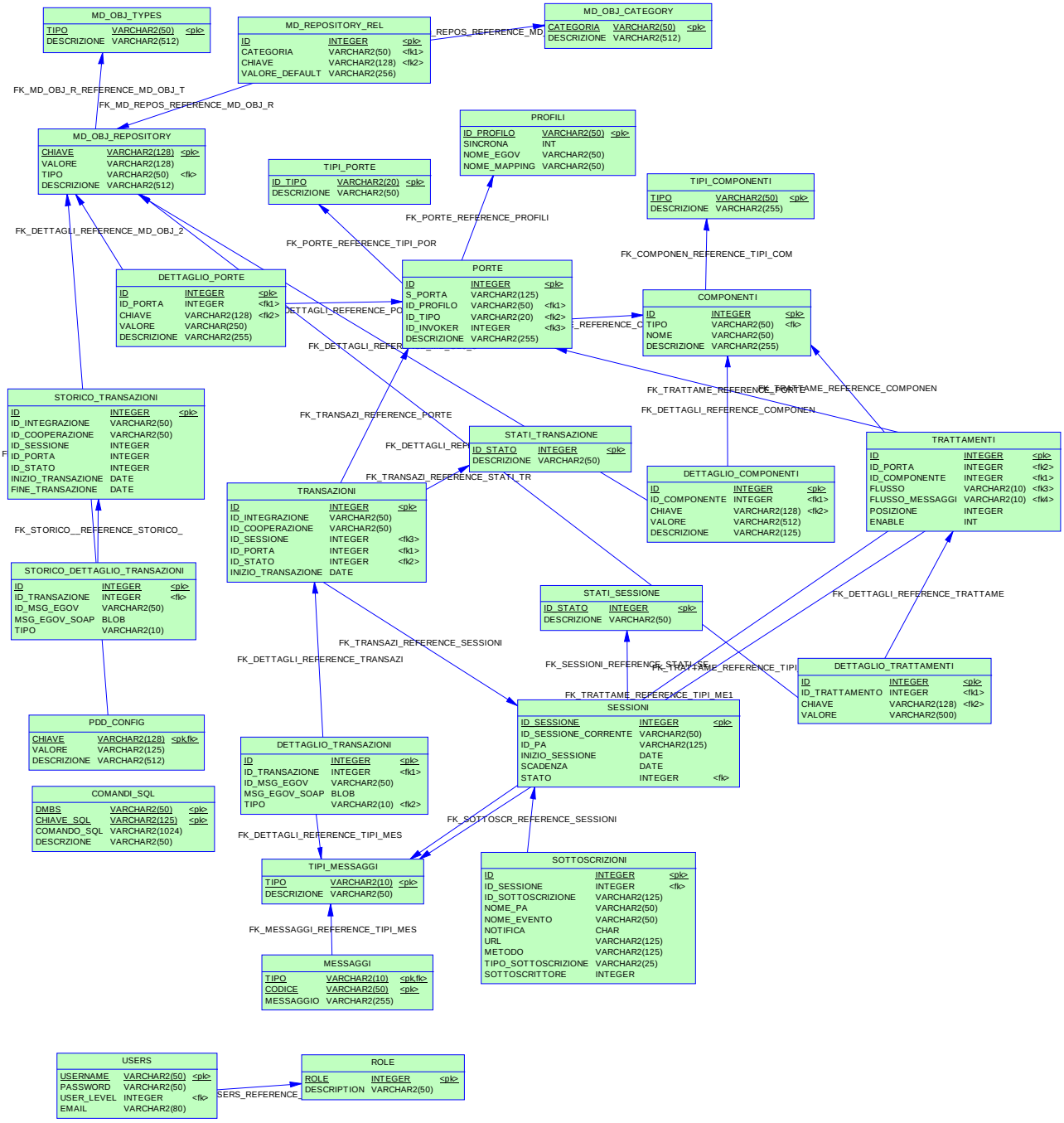


Figura 18- Modello Entiry-Relationship

Metadata Repository:

Queste tabelle hanno il compito di "ingabbiare" tutte le variabili di configurazione della base dati. La PDD-SPICCA sceglie di sposare il modello di configurazione del tipo CHIAVE=VALORE abbinato a strutture MASTER/DETAIL. Questo approccio offre da un lato una grande elasticità: è sempre possibile aggiungere nuovi parametri di configurazione senza modificare la struttura del DB; dall'altro però permette un non

tipizzazione del dato di configurazione. Queste tabelle hanno appunto il compito di Tipizzare fortemente tutti gli elementi presenti nel DB al fine di non incappare in tipici errori applicativi derivanti dall'utilizzo di oggetti non tipizzati.

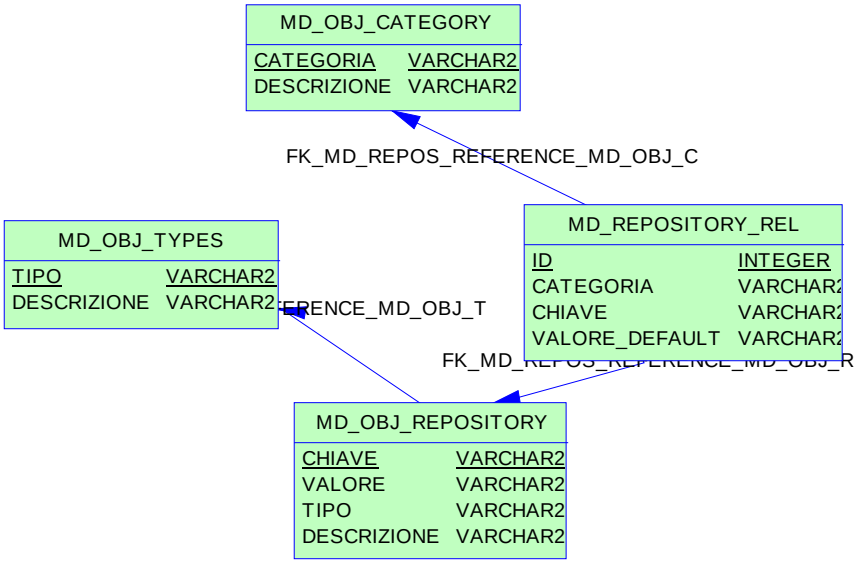


Figura 19- Schema E-R Metadata Repository

Porte Applicative/Delegate:

L'elemento principale della PDD-SPICCA è rappresentato dalla possibilità di aggiungere nuove porte applicative/delegate attraverso una semplice operazione di configurazione. Data la natura di puro proxy applicativo della PDD, e dato l'utilizzo di OpenPDDL2, è possibile esporre servizi interni attraverso una semplice operazione di configurazione sfruttando le implementazioni di riferimento delle porte applicative/delegate della PDD, senza dover scrivere ulteriori righe di codice.

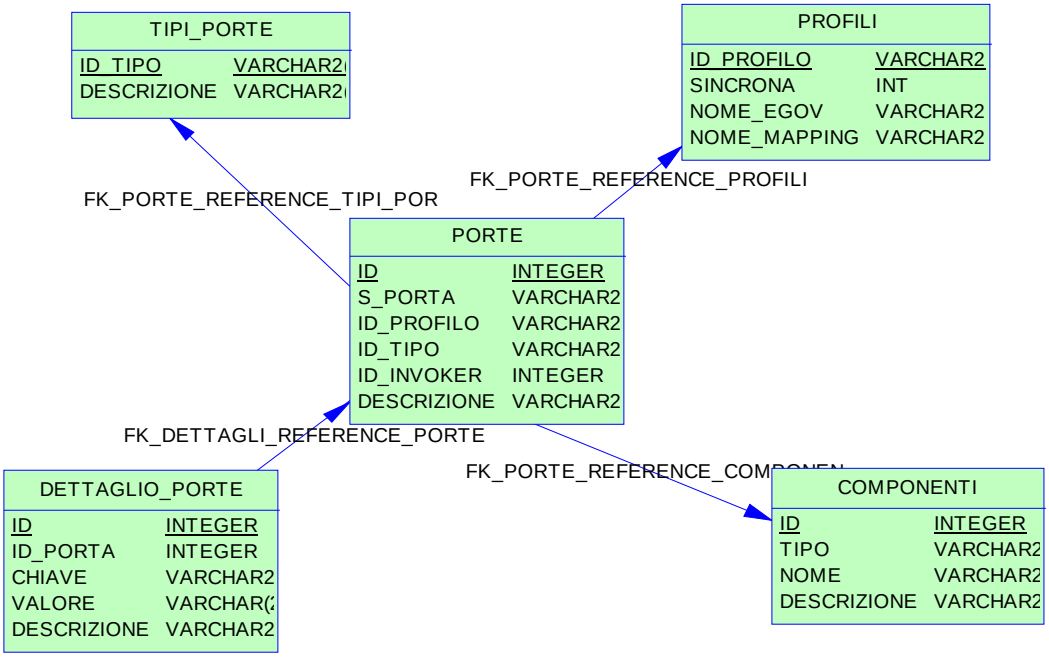


Figura 20- Schema E-R Porte

Componenti di piattaforma:

La loro definizione è contenuta nelle entità: COMPONENTI e DETTAGLIO_COMPONENTI.

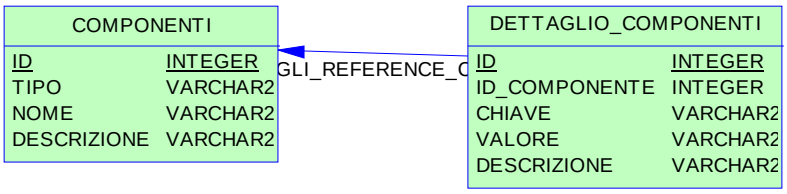


Figura 21- Schema E-R Componenti di piattaforma

Tra i componenti ci sono il logger, gli invoker, gli handler (i trattamenti) ecc..

Ogni componente è una singola unità software utilizzabile via API dalla piattaforma/programmatore. Ogni componente, richiede la configurazione di una Factory definita nel DETTAGLIO_COMPONENTI attraverso il meccanismo CHIAVE-VALORE.

Trattamenti:

Uno dei principali elementi caratterizzanti la PDD-SPICCA è la possibilità di associare a tutti i possibili flussi applicativi (Client Request, Client Response, Server Request e Server Response) un numeno non limitato superiormente, di handler per la gestione dei Trattamenti della busta in ingresso e in uscita. Ogni trattamento è una specializzazione di un Componente HANDLER (tabella componenti) che, descritto da una specifica Factory (configurata in DETTAGLIO_COMPONENTI sotto forma di CHIAVE-VALORE), permetti di "Pluggure" nella piattaforma un particolare trattamento a "Caldo" nel sistema. Ogni Trattamento può essere Abilitato e Disabilitato in tempo reale senza dover far ripartire l'Application Server Host.

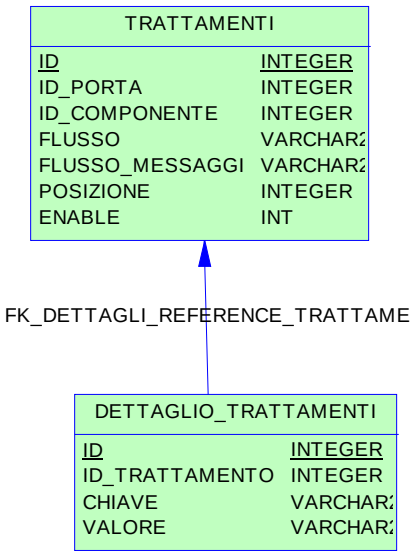


Figura 22- Schema E-R Trattamenti

Transazioni:

Tutte le transazioni attive (in esecuzione in un dato istante)m, vengono memorizzate sul DB per essere poi spostate nello storico a completamento della transazione stessa. Per ogni transazione viene memorizzato (con relativi messaggi di Richiesta e di risposta) l'intera busta e-Gov.

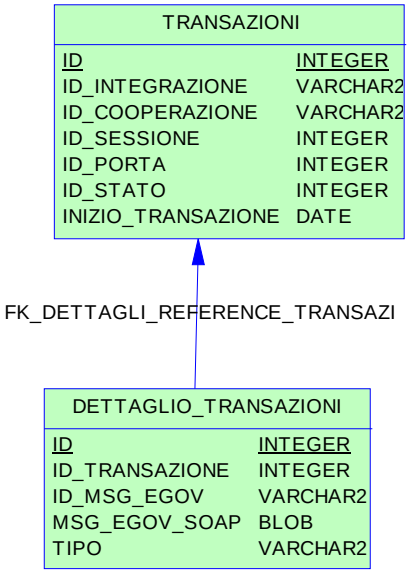


Figura 23- Schema E-R Transazioni

Storico Transazioni:

In queste tabelle viene mantenuto lo storico delle transazioni. Ad ogni transazione è sempre associato uno stato che ne caratterizza l'esito.

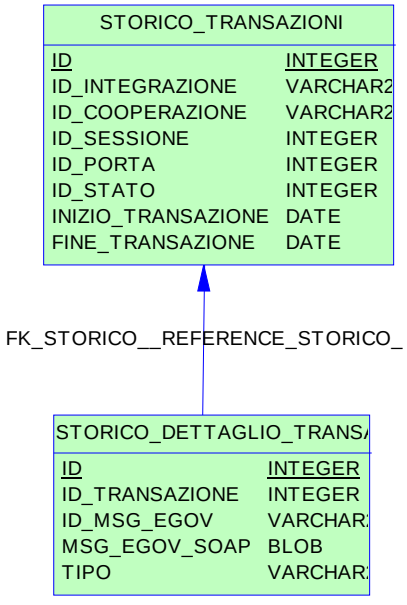


Figura 24- Schema E-R Storico Transazioni

NOME	DEFINIZIONE
PORTE	Repository di tutte le porte APPLICATIVE/DELEGATE di piattaforma.
PROFILI	Questa tabella mantiene una anagrafica di tutti i profili supportati.
DETTAGLIO_PORTE	Tabella Detail delle Porte Applicative/Delegate configurate nel sistema
TIPI_PORTE	Anagrafica delle tipologie di porte. Possono essere APPLICATIVA o DELEGATA.
COMPONENTI	Tabella Master dei componenti di piattaforma. Questi componenti sono utilizzati come API dalla Porta Di Dominio. Ogni componente ha un ID stringa associato e una classe di implementazione
DETTAGLIO_COMPONENTI	Questa tabella contiene tutte le properties di configurazione dei componenti utilizzati dalla PDD SPICCA. La tabella è un Detail del Master *Componenti*. Tutte le properties sono espresse nella forma *Chiave-Valore*
TIPI_COMPONENTI	Contiene tutte le tipologie di componenti supportate dalla piattaforma.

Tabella 1 - Lista Tabelle DB PDD SPICCA

8.1 Tabella COMPONENTI

Nome	Commento	Tipo di Dato	Lunghezza
ID	Identifica l'ID di riga	NUMBER	11
TIPO	Identifica il tipo di componente.	VARCHAR2	50
DESCRIZIONE	Descrizione del Componente	VARCHAR2	255
NOME	Identifica il nome del Componente	VARCHAR2	50

8.2 Tabella DETTAGLIO_COMPONENTI

Nome	Commento	Tipo di Dato	Lunghezza
ID	ID Univoco del componente. Viene implementato come sequence e incrementato da un trigger per la gestione dei progressivi	NUMBER	11
ID_COMPONENTE	ID del componente master a cui appartengono le properties di	NUMBER	11

Nome	Commento	Tipo di Dato	Lunghezza
	dettaglio.		
CHIAVE	Chiave della properties avente VALORE come contenuto	VARCHAR2	255
VALORE	Valore associato alla chiave CHIAVE	VARCHAR2	512
DESCRIZIONE	Descrizione della properties identificata dal campo CHIAVE	VARCHAR2	125

8.3 Tabella DETTAGLIO_PORTE

Nome	Commento	Tipo di Dato	Lunghezza
ID	Identificativo univoco del dettaglio porta.	NUMBER	11
ID_PORTA	Identificativo della Porta Master a cui le properties di configurazione fanno riferimento	NUMBER	11
CHIAVE	Ogni porta ha delle properties di configurazione associate nella forma CHIAVE,VALORE	VARCHAR2	50
VALORE	Valore della properties di configurazione identificata da CHIAVE	VARCHAR2	250
DESCRIZIONE	Descrizione della properties di configurazione	VARCHAR2	255

8.4 Tabella PORTE

Nome	Commento	Tipo di Dato	Lunghezza
ID	Identificativo Porta. Questo campo contiene il campo AZIONE della Busta e-Gov. Questo è un campo discriminante che identifica il servizio che verrà implementato dalla Porta	NUMBER	11
ID_PROFILO	Tipo di profilo supportato dalla porta: OneWay, Sincrono,	VARCHAR2	50

Nome	Commento	Tipo di Dato	Lunghezza
	Asincrono Simmetrico, Asincrono Asimmetrico		
ID_TIPO	Tipo della Porta: APPLICATIVA/DELEGATA	VARCHAR2	20
ID_INVOKER	ID del componente per invocare questa porta	NUMBER	11
DESCRIZIONE	Descrizione della Porta	VARCHAR2	255

8.5 Tabella PROFILI

Nome	Commento	Tipo di Dato	Lunghezza
ID_PROFILO	Contiene il profilo implementato dalla porta.	VARCHAR2	50
SINCRONA	Rappresenta il tipo di profilo. 1 se il profilo è sincrono. 0 se il profilo è asincrono	INT	

8.6 Tabella TIPI_COMPONENTI

Nome	Commento	Tipo di Dato	Lunghezza
TIPO	Tipo di componente: WEBSERVICE oppure API	VARCHAR2	50
DESCRIZIONE	Descrizione della tipologia	VARCHAR2	255

8.7 Tabella TIPI_PORTE

Nome	Commento	Tipo di Dato	Lunghezza
ID_TIPO	Tipo di porta: APPLICATIVA oppure DELEGATA	VARCHAR2	20
DESCRIZIONE	Descrizione del servizio implementato dalla Porta	VARCHAR2	50