SP – Architettura di dettaglio

Autore: Simone Ballarin

Data: 25/06/18

Destinatari: Athesys

# Diario delle modifiche

|  |  |  |
| --- | --- | --- |
| Data | Descrizione | Autore |
| 27/06/18 | Creazione documento. Inserimento Istruzioni generali per usare il progetto e architettura di dettaglio Starter. | Simone Ballarin |
|  |  | Simone Ballarin |
|  |  | Simone Ballarin |
|  |  | Simone Ballarin |

# Scopo del documento

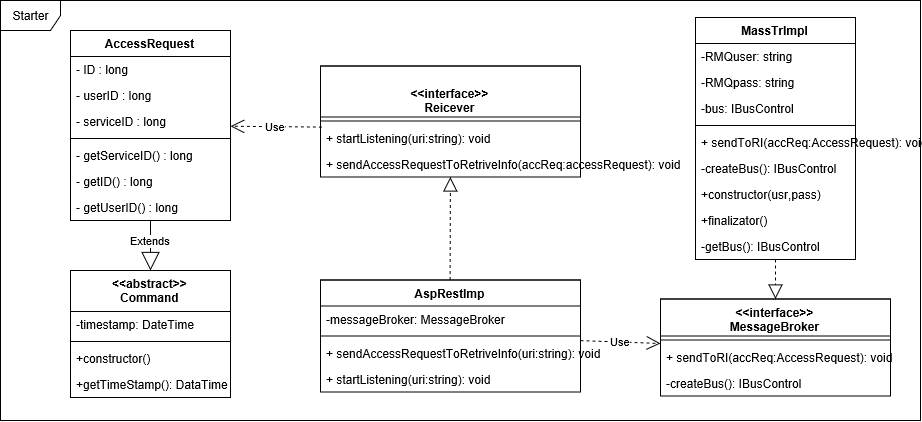
Il seguente documento *SP – Architettura di dettaglio* ha lo scopo di presentare in maniera dettagliato ogni classe presentata in *SP – Architettura di massima*. Il documento deve fornire una descrizione dei metodi e campi dati presenti.

# Breve nota sull’ambiente

* Installare erlang (21.0 non funziona con Windows)
* Installare RabbitMQ
* Abilitare il plugin RabbitMQAdministrator
* https://stackoverflow.com/questions/3850019/running-two-projects-at-once-in-visual-studio
* <http://localhost:15672/#/connections>
* http://www.html.it/pag/19603/implementare-un-web-service-restful/

# Architettura di dettaglio

## Dettaglio Starter



### Reicever (interfaccia)

Questa classe ha lo scopo di ricevere e inoltrare i messaggi alla coda successiva (RetrieveInfo). Rappresenta il driver del componente Starter.

Metodi:

##### Public void startListening(uri: string)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rimanere in ascolto di eventuali comunicazioni da parte dei RSP |
| **Parametri** | * uriPrefixes []: string |
| **Pseudo codice** | Non presente |
| **Note** |  |

##### Private void sendAccessRequestToRetriveInfo (accReq: AccessRequest)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rimanere inviare le richieste di accesso incapsulate negli oggetti AccessRequest alla coda del RetrieveInfo |
| **Parametri** | * accReq: AccessRequest |
| **Pseudo codice** | Non presente |
| **Note** |  |

### AspRestImp (implementa Reicever)

#### Campi dati:

**messageBroker**: è un riferimento all’oggetto che implementa il messageBroker, tramite questo oggetto vengono inoltrate le richieste alla coda del RetrieveInfo.

Metodi:

##### void startListening(uri: string)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rimanere in ascolto di eventuali comunicazioni http |
| **Parametri** | * uriPrefixes[]: string |
| **Pseudo codice** | check (uriPrefixes not null);  check(uriPrefixes not empty)  listener = HttpListener;  foreach (uri in uriPrefixes) {  listener.add(uri);  }  listener.start();  while true {  context = listener.getContext(); //bloccante  request = context.request;  response = context.responce;  accReq = new AccessRequest(getData(responce));  sendAccessRequestToRetriveInfo(accReq);  } |
| **Note** | https://docs.microsoft.com/it-it/dotnet/api/system.net.httplistener?view=netframework-4.7.1#definition |

##### Private void sendAccessRequestToRetriveInfo (accReq: AccessRequest)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rimanere inviare le richieste di accesso incapsulate negli oggetti AccessRequest alla coda del RetrieveInfo |
| **Parametri** | * accReq: AccessRequest |
| **Pseudo codice** | Non presente |
| **Note** |  |

### MessageBroker (interfaccia)

Metodi:

##### Public void sendToRI (accReq: AccessRequest)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rimanere inviare le richieste di accesso incapsulate negli oggetti AccessRequest alla coda del RetrieveInfo |
| **Parametri** | * accReq: AccessRequest |
| **Pseudo codice** | Non presente |
| **Note** |  |

##### Private IBusControl createBus()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito creare il canale di comunicazione verso la coda del componente RetrieveInfo |
| **Parametri** | * user: string * pass: string |
| **Pseudo codice** | Non presente |

### MassTrImpl (implementa MessageBroker)

Questa classe implementa il MessageBroker utilizzano la libreria MassTransit che a sua volta usa RabbitMQ. Per queste ragioni è necessario avere installato RabbitMQ e preparato gli utenti.

#### Campi dati:

**RMQuser:** stringa che contiene il nome dell’account RabbitMQ da usare per inviare i messaggi.

**RMQpass:** stringa che contiene la password dell’account RabbitMQ da usare per inviare i messaggi.

**bus:** riferimento all’oggetto IBusControl cha rappresenta il canale di comunicazione, se questo non è stato creato, o la sua creazione ha fallito presenta un valore a null.

Metodi:

##### Public constructor (user:string, pass: string)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito costruire l’oggetto MassTrImpl e di stabilire la connessione |
| **Parametri** | * user: string * pass: string |
| **Pseudo codice** | This.bus = createBus();  this.user = user;  this.pass = pass;  bus.start()  return this.bus; |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

##### Public finalizator (user:string, pass: string)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rilasciare tutte le risorse di sistema ottenuto durante la vita dell’oggetto. |
| **Parametri** |  |
| **Pseudo codice** | Bus.stop;  rilascio altre risorse; |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

##### Public IBusControl getBus()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito ritornare l’oggetto IBusControl creato dal costruttore |
| **Parametri** |  |
| **Pseudo codice** | Check (bus not null);  return this.bus; |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

##### Public void sendToRI (accReq: AccessRequest)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di rimanere inviare le richieste di accesso incapsulate negli oggetti AccessRequest alla coda del RetrieveInfo |
| **Parametri** | * accReq: AccessRequest |
| **Pseudo codice** | getBus();  client.createRequesClient();  client.request(accReq) |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

##### Private IBusControl createBus(user:string, pass:string)

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito creare il canale di comunicazione verso la coda del componente RetrieveInfo |
| **Parametri** | * user: string * pass: string |
| **Pseudo codice** | create b = new Bus.Factory.CreateUsingRabbitMQ;  setUsername(user);  setPassword(pass);  setDatiServerRabbitMQ();  return b; |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

### Command (abstract)

Questa classa astratta rappresenta un generico messaggio che può essere trasportato dal MessageBroker.

#### Campi dati:

**timestamp**: è un riferimento all’oggetto DateTime che contiene il timestamp della creazione del Command.

Metodi:

##### Public construct ()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito costruire l’oggetto Command. |
| **Parametri** |  |
| **Pseudo codice** | This.timeStamp = DataTime.UtcNow |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

##### Public DateTIme getTimeStamp()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito di restituire il timestamp che rappresenta il momento di creazione dell’oggetto Command |
| **Parametri** |  |
| **Pseudo codice** | Return this.timestamp; |
| **Note** | masstransit-project.com/MassTransit/learn/samples/request-response.html |

### AccessRequest (estende Command)

Rappresenta una richiesta di accesso, nella visione Event Driven rappresenta un evento iniziale.

#### Campi dati:

**ID**: è un long che rappresenta una chiave della richiesta, questo codice deve dovrebbe condiviso anche dall’ITF, Monokee e RSP.

**userID:** è un long che rappresenta una chiave dell’utente che ha richiesto l’accesso, questo codice dovrebbe essere condiviso anche dall’ITF, Monokee e RSP.

**serviceID:** è un long che rappresenta una chiave del servizio a cui l’utente a richiesto l’accesso, questo codice dovrebbe essere condiviso anche dall’ITF, Monokee e RSP.

#### Metodi:

##### Public long getID()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito restituire l’ID della richiesta d’accesso al servizio |
| **Parametri** |  |
| **Pseudo codice** | Return this.ID; |
| **Note** |  |

##### Public long getUserID()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito restituire l’ID dell’utente che ha effettuato la richiesta di accesso. |
| **Parametri** |  |
| **Pseudo codice** | Return this.userID; |
| **Note** |  |

##### Public long getServiceID()

|  |  |
| --- | --- |
| **Descrizione** | Il metodo ha il compito restituire l’ID del servizio a cui la richiesta di accesso si riferisce |
| **Parametri** |  |
| **Pseudo codice** | Return this.serviceID; |
| **Note** |  |