Università degli Studi di Napoli Federico II

Corso di Laurea in Informatica

VOGCLUSTERS: Una Web Application per il trattamento e l'analisi di ammassi globulari

Tutor accademico:
Dott.ssa Anna Corazza

Tutor aziendale:
Dott. Massimo Brescia

Candidato:
Sabrina Checola
matr: 566/1577

Anno Accademico 2008/2009

Cosa sono gli ammassi globulari?

Globular cluster "47 Tuc" (NGC 104)

- Un ***ammasso globulare*** è un insieme di stelle che orbita come un satellite attorno al centro di una galassia;
- Ogni ammasso globulare è caratterizzato da una serie di parametri. (es. Ascensione retta, Declinazione etc.);
- Il valore di ogni parametro è destinato a variare nel tempo a causa dell'evoluzione dello studio di tali oggetti;

National Virtual Observatory

Query Results: VII/195

NVO Home | Modify Query | New Query | Scripting | Help

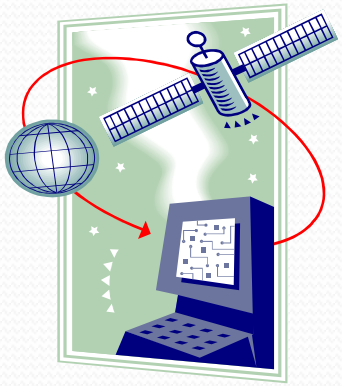
Graph | Export to ...

Results **1-20** of **146** Show 20 results per page

Text boxes under columns select matching rows Apply Filter Clear Filter

ID	Name	RA2000	DE2000	GLON	GLAT
NGC 104	47 Tuc	00 24 05.2	-72 04 51	305.9	-44.9
NGC 288		00 52 47.5	-26 35 24	152.3	-89.4
NGC 362		01 03 14.3	-70 50 54	301.5	-46.2
NGC 1261		03 12 15.3	-55 13 01	270.5	-52.1
Pal 1		03 33 23.0	+79 34 50	130.1	19.0
AM 1	E 1	03 55 02.7	-49 36 52	258.4	-48.5
Eridanus		04 24 44.5	-21 11 13	218.1	-41.3
Pal 2		04 46 05.9	+31 22 51	170.5	-9.1
NGC 1851		05 14 06.3	-40 02 50	244.5	-35.0
NGC 1904	M 79	05 24 10.6	-24 31 27	227.2	-29.4
NGC 2298		06 48 59.2	-36 00 19	245.6	-16.0
NGC 2419		07 38 08.5	+38 52 55	180.4	25.2
Pyxis		09 07 57.8	-37 13 17	261.3	7.0
NGC 2808		09 12 02.6	-64 51 47	282.2	-11.3
E 3		09 20 59.3	-77 16 57	292.3	-19.0
Pal 3		10 05 31.4	+00 04 17	240.1	41.9
NGC 3201		10 17 36.8	-46 24 40	277.2	8.6
Pal 4		11 29 16.8	+28 58 25	202.3	71.8
NGC 4147		12 10 06.2	+18 32 31	252.9	77.2
NGC 4372		12 25 45.4	-72 39 33	301.0	-9.9

Il Problema



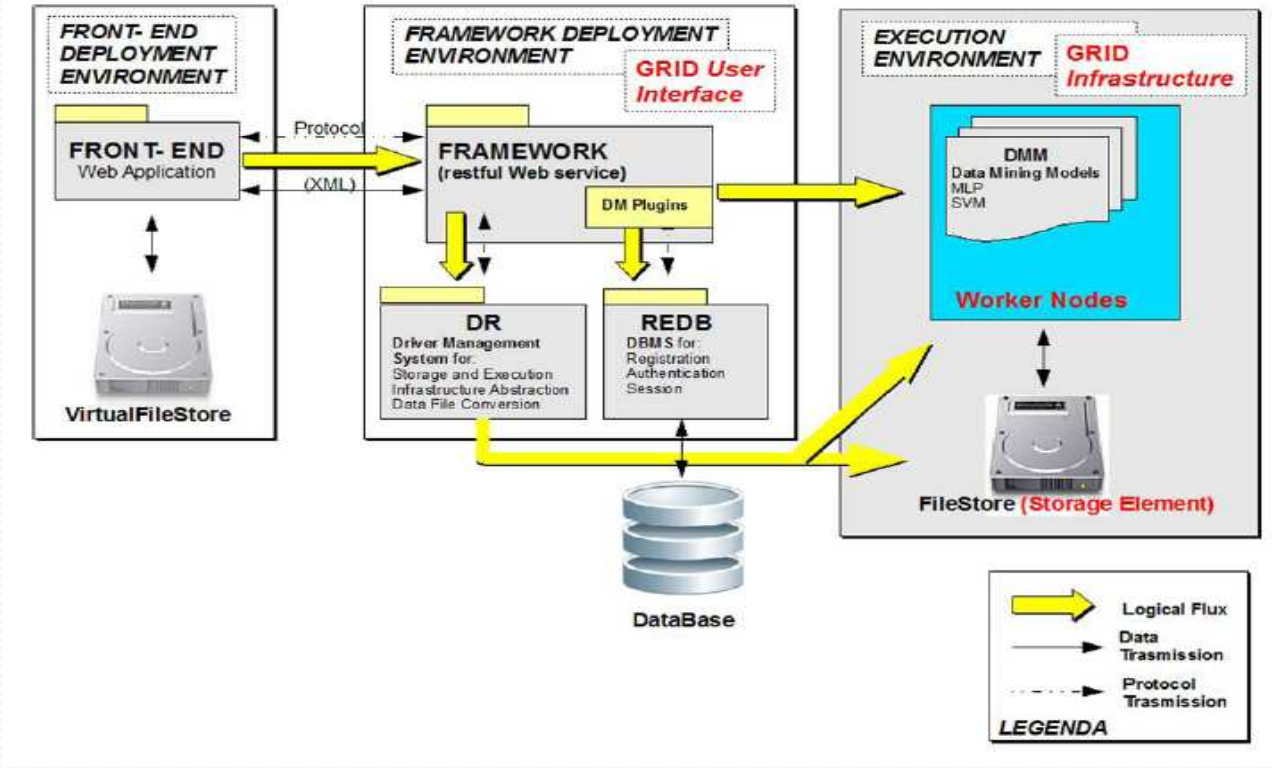
1. Presenza di numerosi archivi delocalizzati contenenti informazioni relative ad ammassi globulari;
2. Gran parte degli archivi non presentano uniformità e omogeneità nella rappresentazione dei dati;
3. I dati necessitano di un costante aggiornamento;

NECESSITA' DI STANDARDIZZAZIONE

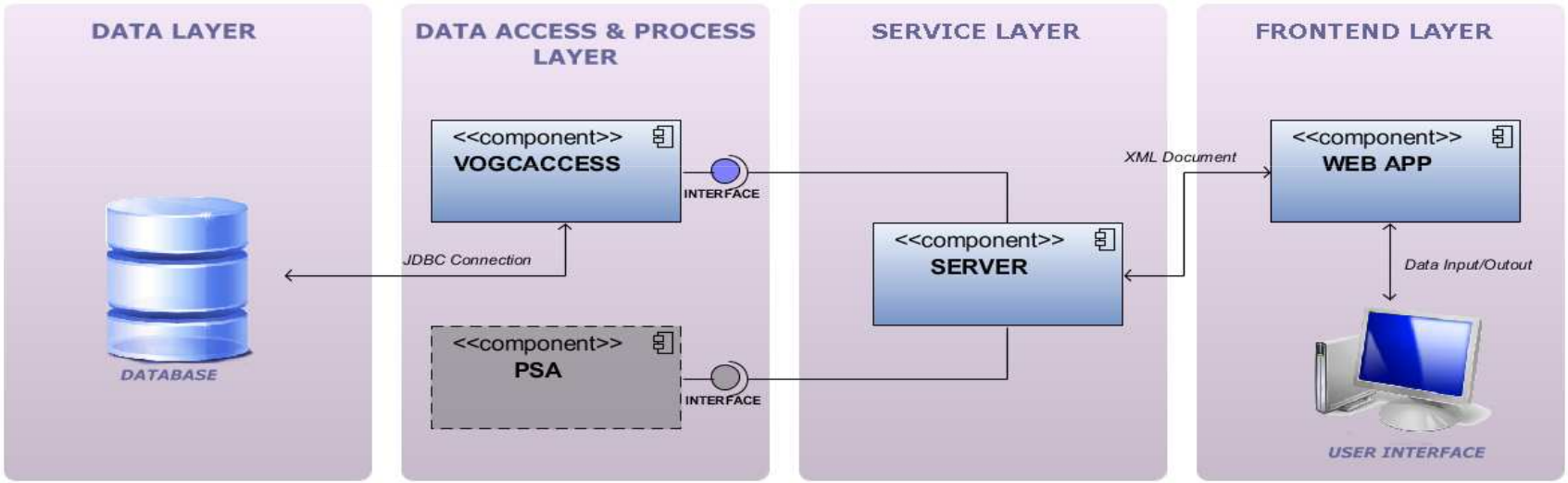
(IVOA International Virtual Observatory Alliance)

Dove si colloca *VOGCLUSTERS*?

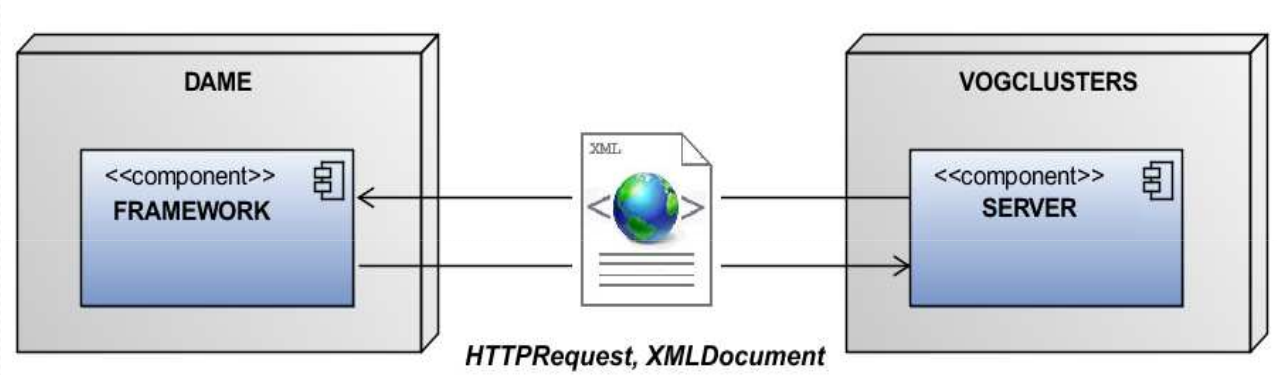
Il progetto DAME consiste nello sviluppo di una "*Data Mining Suite*" che fornirà alle comunità astronomiche una web application specializzata per enormi *dataset* in un ambiente di calcolo distribuito rispettando gli standard internazionali e i requisiti IVOA.



Architettura *VOGCLUSTERS*

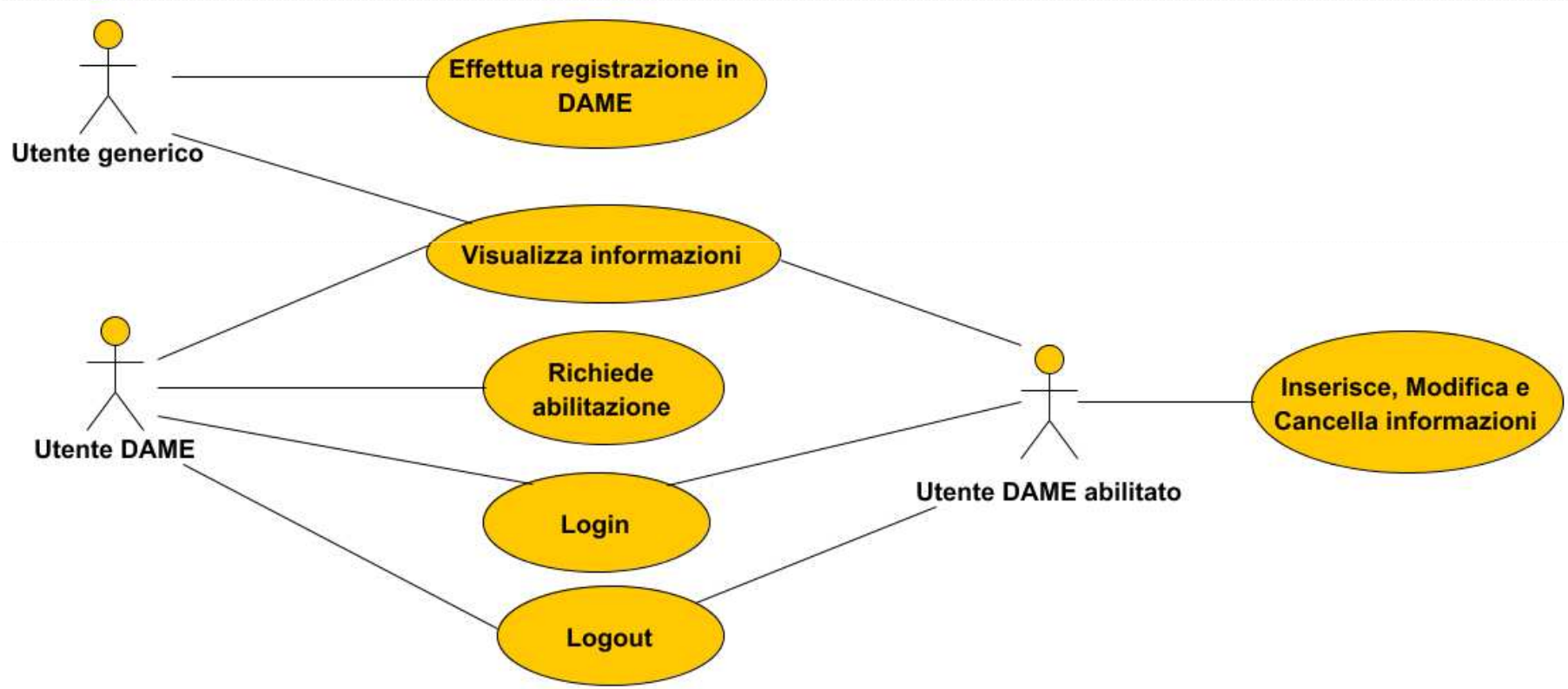


Comunicazione DAME - VOGCLUSTERS

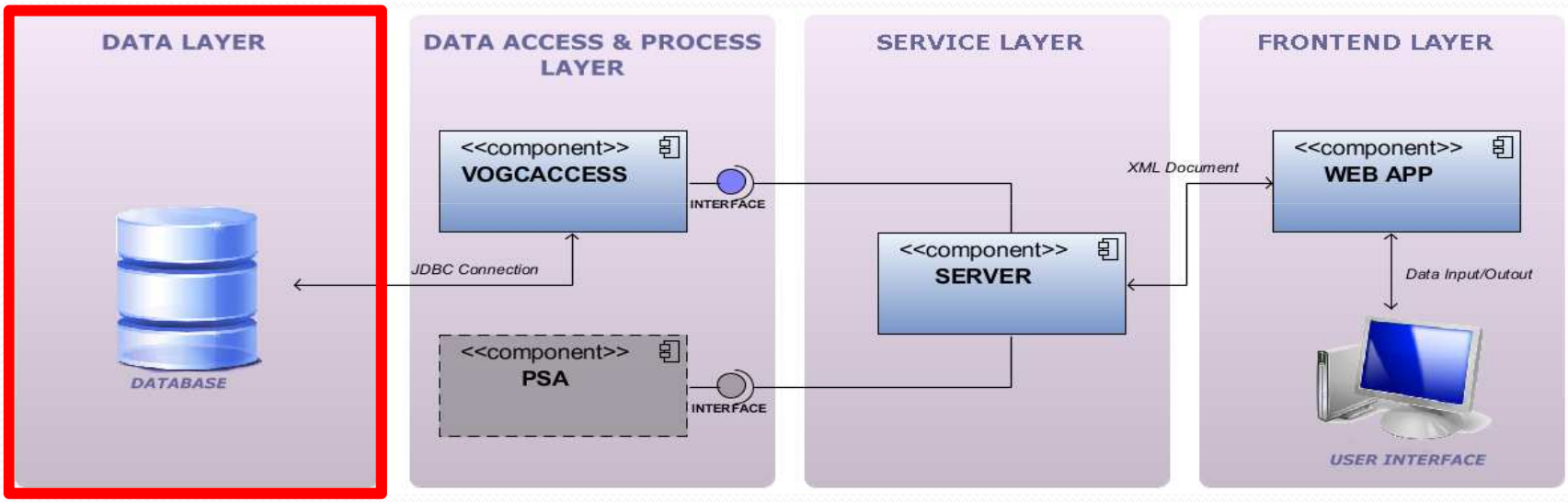


- Comunicazione attraverso documenti XML (VOTable)
- Condivisione delle informazioni degli utenti registrati: due database differenti, ma con una tabella condivisa

Casi d'uso


Utente generico
Effettua registrazione in DAME
Visualizza informazioni
Utente DAME
Richiede abilitazione
Login
Logout
Utente DAME abilitato
Inserisce, Modifica e Cancella informazioni

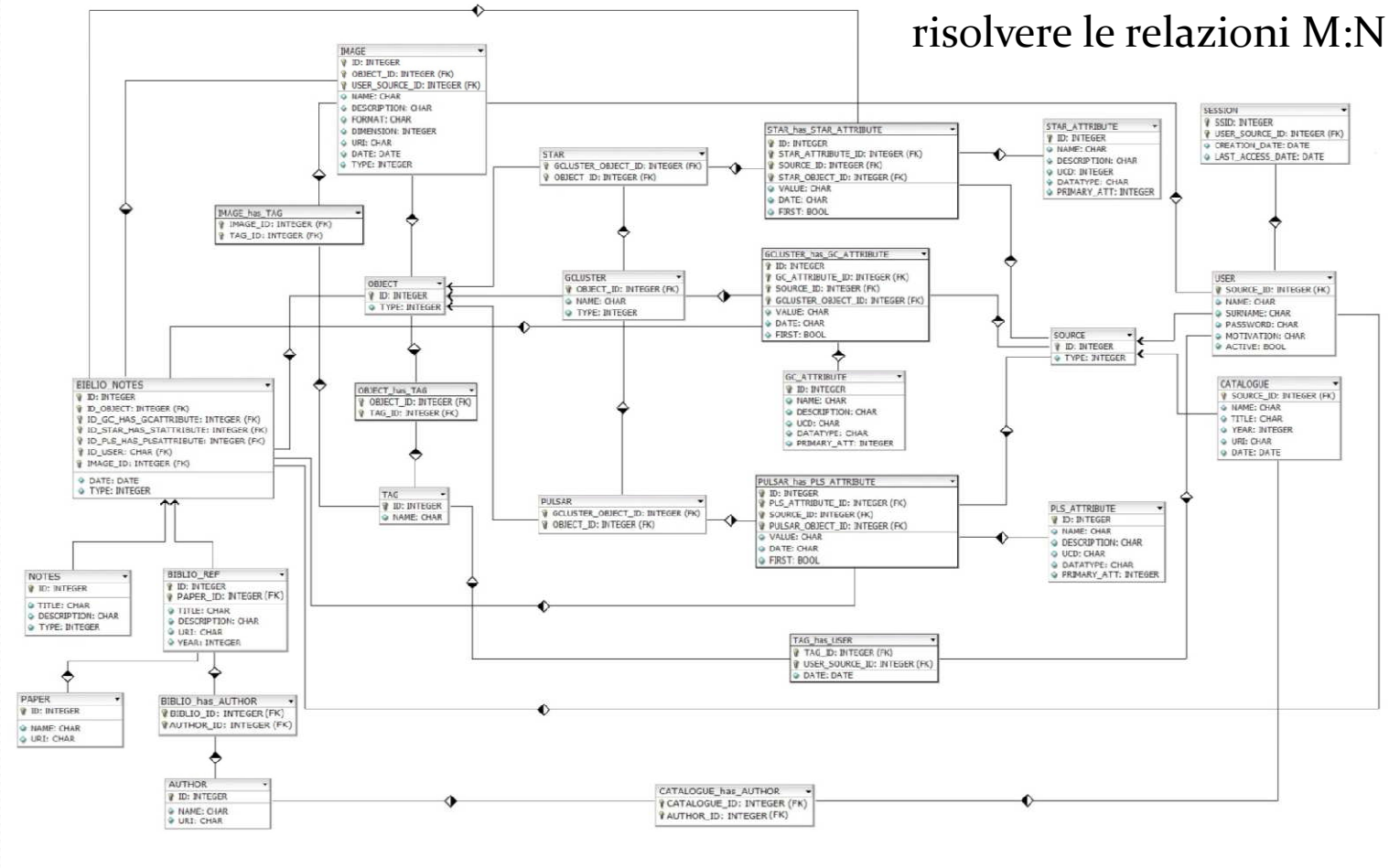
Architettura di *VOGCLUSTERS*



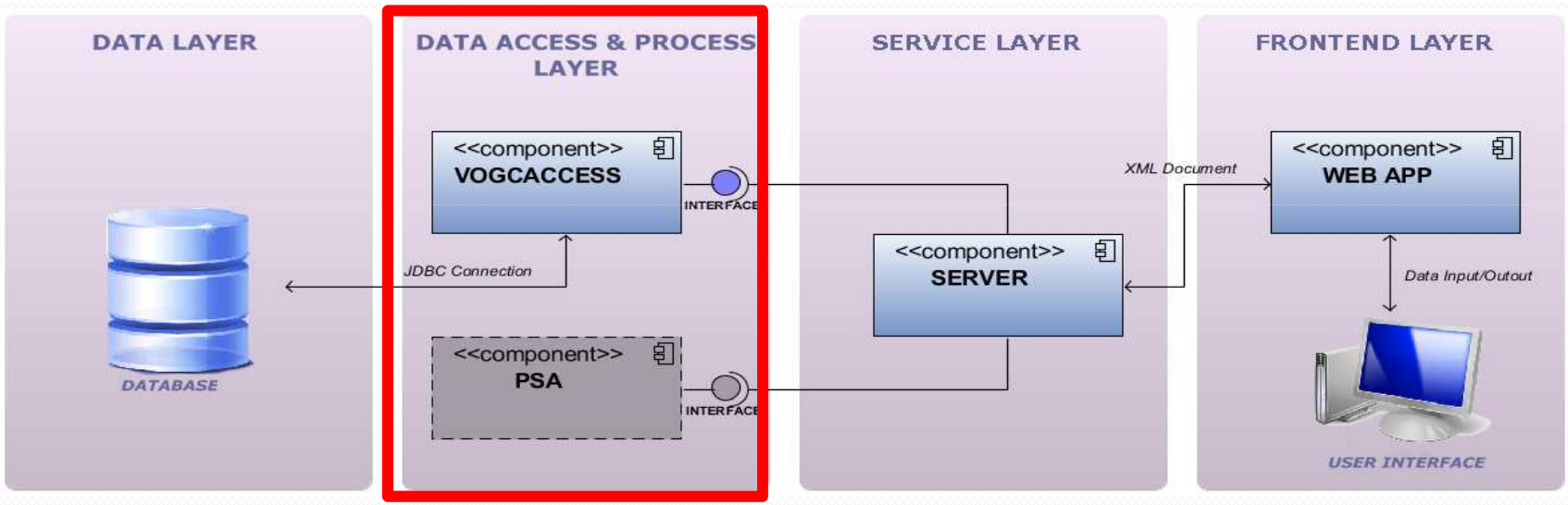
DATA LAYER: persistenza dei dati

Il database

Il database si struttura in 18 entità e 25 relazioni di cui 8 vengono convertite in altrettante tabelle per risolvere le relazioni M:N

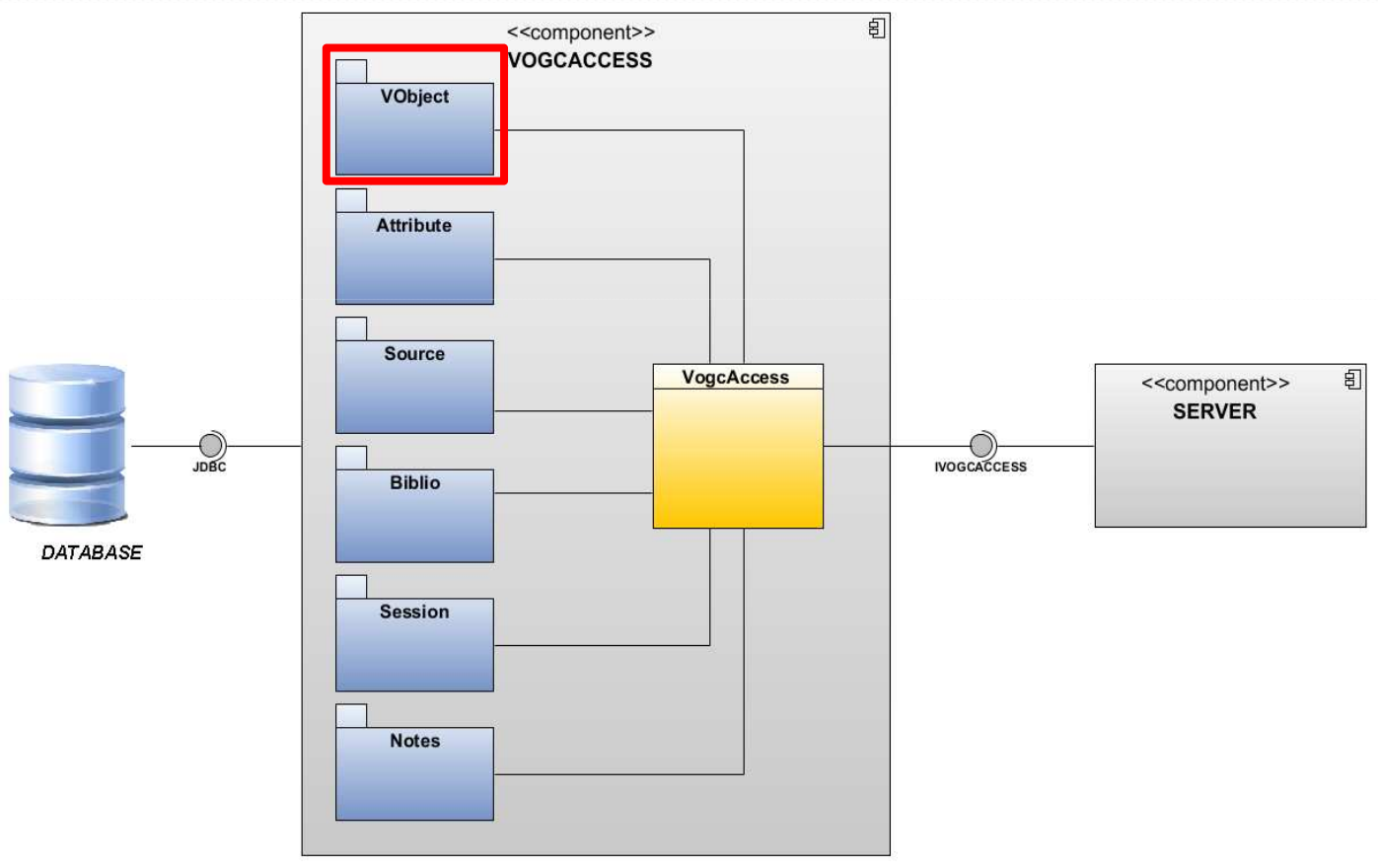


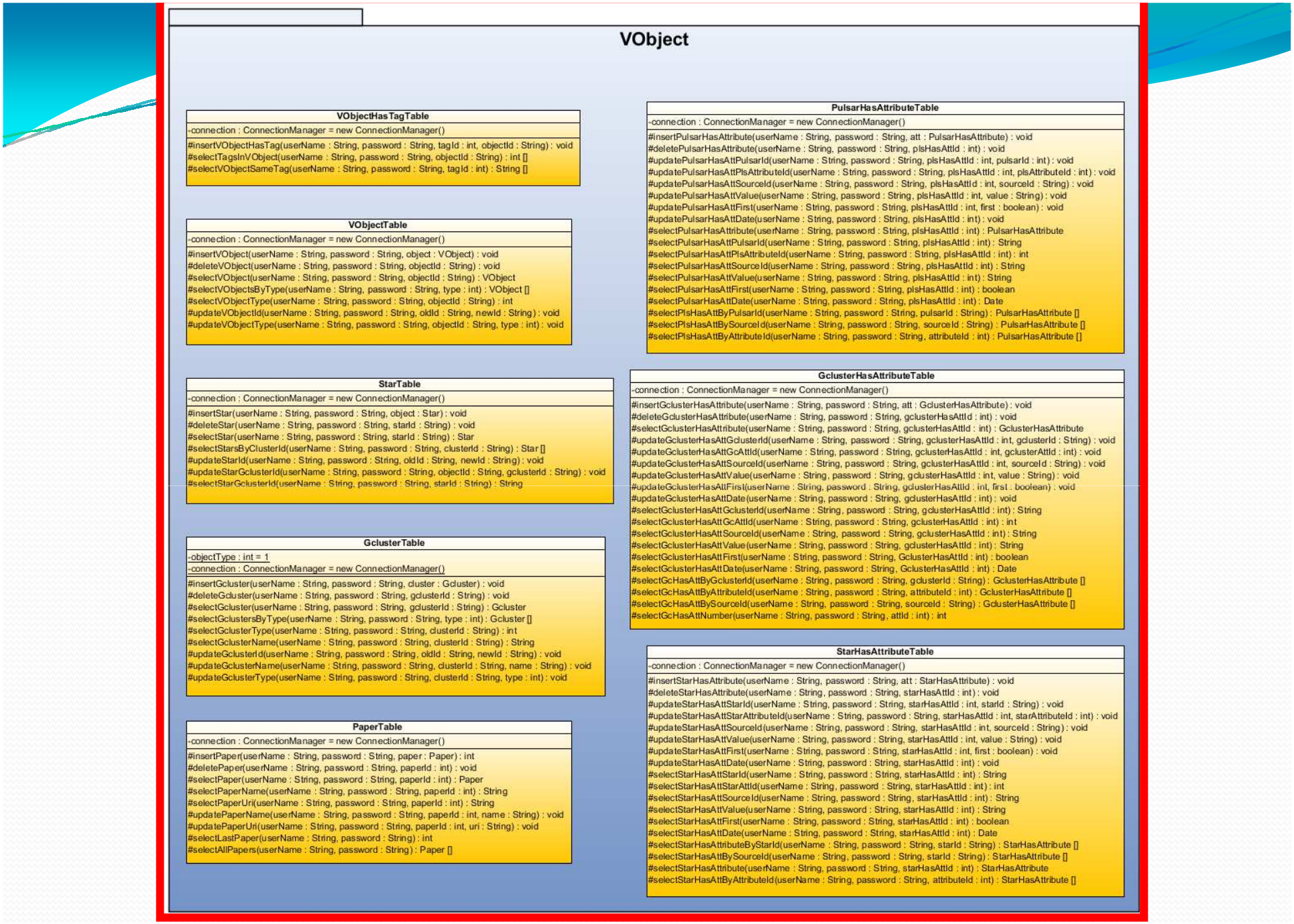
Architettura di VOGCLUSTERS



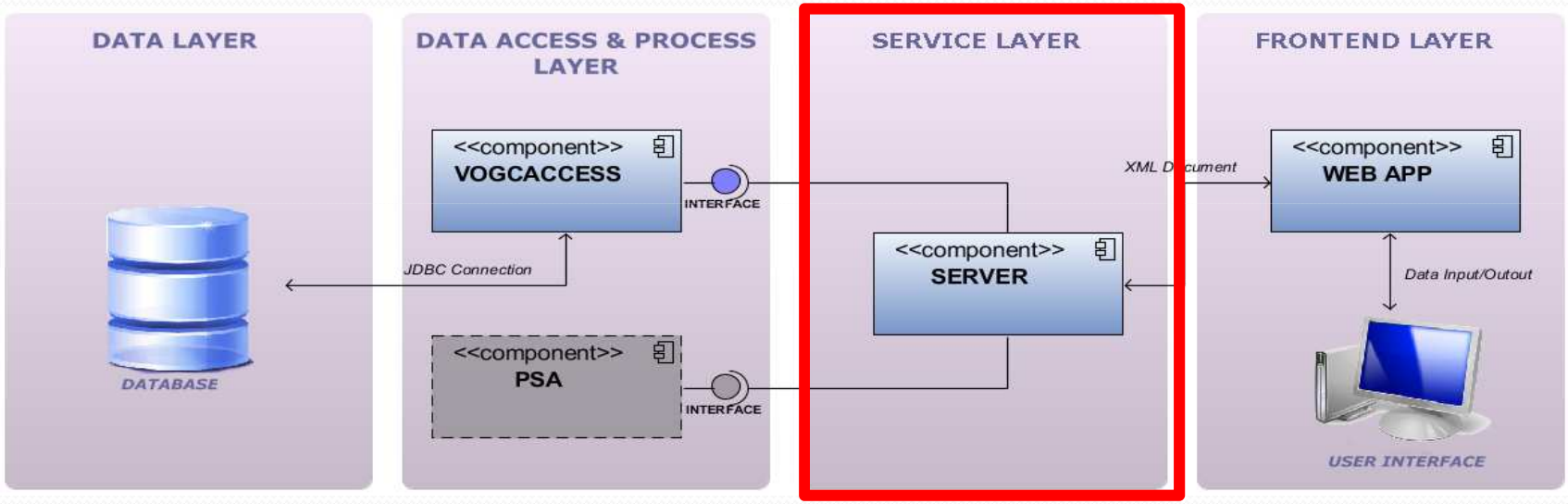
DATA ACCESS & PROCESS LAYER: accesso e trattamento statistico e analitico dei dati permanenti.

Accesso al database: VOGCACCESS



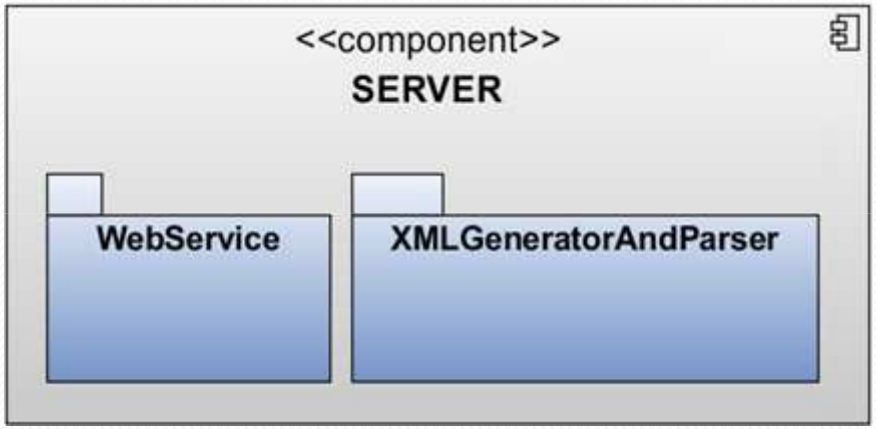

VObject
VObjectHasTagTable
-connection : ConnectionManager = new ConnectionManager()
#insertVObjectHasTag(userName : String, password : String, tagId : int, objectId : String) : void
#selectTagsInVObject(userName : String, password : String, objectId : String) : int []
#selectVObjectSameTag(userName : String, password : String, tagId : int) : String []
VObjectTable
-connection : ConnectionManager = new ConnectionManager()
#insertVObject(userName : String, password : String, object : VObject) : void
#deleteVObject(userName : String, password : String, objectId : String) : void
#selectVObject(userName : String, password : String, objectId : String) : VObject
#selectVObjectsByType(userName : String, password : String, type : int) : VObject []
#selectVObjectType(userName : String, password : String, objectId : String) : int
#updateVObjectId(userName : String, password : String, oldId : String, newId : String) : void
#updateVObjectType(userName : String, password : String, objectId : String, type : int) : void
StarTable
-connection : ConnectionManager = new ConnectionManager()
#insertStar(userName : String, password : String, object : Star) : void
#deleteStar(userName : String, password : String, starId : String) : void
#selectStar(userName : String, password : String, starId : String) : Star
#selectStarsByClusterId(userName : String, password : String, clusterId : String) : Star []
#updateStarId(userName : String, password : String, oldId : String, newId : String) : void
#updateStarGclusterId(userName : String, password : String, objectId : String, gclusterId : String) : void
#selectStarGclusterId(userName : String, password : String, starId : String) : String
GclusterTable
-objectType : int = 1
-connection : ConnectionManager = new ConnectionManager()
#insertGcluster(userName : String, password : String, cluster : Gcluster) : void
#deleteGcluster(userName : String, password : String, gclusterId : String) : void
#selectGcluster(userName : String, password : String, gclusterId : String) : Gcluster
#selectGclustersByType(userName : String, password : String, type : int) : Gcluster []
#selectGclusterType(userName : String, password : String, clusterId : String) : int
#selectGclusterName(userName : String, password : String, clusterId : String) : String
#updateGclusterId(userName : String, password : String, oldId : String, newId : String) : void
#updateGclusterName(userName : String, password : String, clusterId : String, name : String) : void
#updateGclusterType(userName : String, password : String, clusterId : String, type : int) : void
PaperTable
-connection : ConnectionManager = new ConnectionManager()
#insertPaper(userName : String, password : String, paper : Paper) : int
#deletePaper(userName : String, password : String, paperId : int) : void
#selectPaper(userName : String, password : String, paperId : int) : Paper
#selectPaperName(userName : String, password : String, paperId : int) : String
#selectPaperUri(userName : String, password : String, paperId : int) : String
#updatePaperName(userName : String, password : String, paperId : int, name : String) : void
#updatePaperUri(userName : String, password : String, paperId : int, uri : String) : void
#selectLastPaper(userName : String, password : String) : int
#selectAllPapers(userName : String, password : String) : Paper []
PulsarHasAttributeTable
-connection : ConnectionManager = new ConnectionManager()
#insertPulsarHasAttribute(userName : String, password : String, att : PulsarHasAttribute) : void
#deletePulsarHasAttribute(userName : String, password : String, plsHasAttId : int) : void
#updatePulsarHasAttPulsarId(userName : String, password : String, plsHasAttId : int, pulsarId : int) : void
#updatePulsarHasAttPlsAttributeId(userName : String, password : String, plsHasAttId : int, plsAttributeId : int) : void
#updatePulsarHasAttSourceId(userName : String, password : String, plsHasAttId : int, sourceId : String) : void
#updatePulsarHasAttValue(userName : String, password : String, plsHasAttId : int, value : String) : void
#updatePulsarHasAttFirst(userName : String, password : String, plsHasAttId : int, first : boolean) : void
#updatePulsarHasAttDate(userName : String, password : String, plsHasAttId : int) : void
#selectPulsarHasAttribute(userName : String, password : String, plsHasAttId : int) : PulsarHasAttribute
#selectPulsarHasAttPulsarId(userName : String, password : String, plsHasAttId : int) : String
#selectPulsarHasAttPlsAttributeId(userName : String, password : String, plsHasAttId : int) : int
#selectPulsarHasAttSourceId(userName : String, password : String, plsHasAttId : int) : String
#selectPulsarHasAttValue(userName : String, password : String, plsHasAttId : int) : String
#selectPulsarHasAttFirst(userName : String, password : String, plsHasAttId : int) : boolean
#selectPulsarHasAttDate(userName : String, password : String, plsHasAttId : int) : Date
#selectPlsHasAttByPulsarId(userName : String, password : String, pulsarId : String) : PulsarHasAttribute []
#selectPlsHasAttBySourceId(userName : String, password : String, sourceId : String) : PulsarHasAttribute []
#selectPlsHasAttByAttributeId(userName : String, password : String, attributeId : int) : PulsarHasAttribute []
GclusterHasAttributeTable
-connection : ConnectionManager = new ConnectionManager()
#insertGclusterHasAttribute(userName : String, password : String, att : GclusterHasAttribute) : void
#deleteGclusterHasAttribute(userName : String, password : String, gclusterHasAttId : int) : void
#selectGclusterHasAttribute(userName : String, password : String, gclusterHasAttId : int) : GclusterHasAttribute
#updateGclusterHasAttGclusterId(userName : String, password : String, gclusterHasAttId : int, gclusterId : String) : void
#updateGclusterHasAttGcAttId(userName : String, password : String, gclusterHasAttId : int, gclusterAttId : int) : void
#updateGclusterHasAttSourceId(userName : String, password : String, gclusterHasAttId : int, sourceId : String) : void
#updateGclusterHasAttValue(userName : String, password : String, gclusterHasAttId : int, value : String) : void
#updateGclusterHasAttFirst(userName : String, password : String, gclusterHasAttId : int, first : boolean) : void
#updateGclusterHasAttDate(userName : String, password : String, gclusterHasAttId : int) : void
#selectGclusterHasAttGclusterId(userName : String, password : String, gclusterHasAttId : int) : String
#selectGclusterHasAttGcAttId(userName : String, password : String, gclusterHasAttId : int) : int
#selectGclusterHasAttSourceId(userName : String, password : String, gclusterHasAttId : int) : String
#selectGclusterHasAttValue(userName : String, password : String, gclusterHasAttId : int) : String
#selectGclusterHasAttFirst(userName : String, password : String, GclusterHasAttId : int) : boolean
#selectGclusterHasAttDate(userName : String, password : String, GclusterHasAttId : int) : Date
#selectGcHasAttByGclusterId(userName : String, password : String, gclusterId : String) : GclusterHasAttribute []
#selectGcHasAttByAttributeId(userName : String, password : String, attributeId : int) : GclusterHasAttribute []
#selectGcHasAttBySourceId(userName : String, password : String, sourceId : String) : GclusterHasAttribute []
#selectGcHasAttNumber(userName : String, password : String, attId : int) : int
StarHasAttributeTable
-connection : ConnectionManager = new ConnectionManager()
#insertStarHasAttribute(userName : String, password : String, att : StarHasAttribute) : void
#deleteStarHasAttribute(userName : String, password : String, starHasAttId : int) : void
#updateStarHasAttStarId(userName : String, password : String, starHasAttId : int, starId : String) : void
#updateStarHasAttStarAttributeId(userName : String, password : String, starHasAttId : int, starAttributeId : int) : void
#updateStarHasAttSourceId(userName : String, password : String, starHasAttId : int, sourceId : String) : void
#updateStarHasAttValue(userName : String, password : String, starHasAttId : int, value : String) : void
#updateStarHasAttFirst(userName : String, password : String, starHasAttId : int, first : boolean) : void
#updateStarHasAttDate(userName : String, password : String, starHasAttId : int) : void
#selectStarHasAttStarId(userName : String, password : String, starHasAttId : int) : String
#selectStarHasAttStarAttId(userName : String, password : String, starHasAttId : int) : int
#selectStarHasAttSourceId(userName : String, password : String, starHasAttId : int) : String
#selectStarHasAttValue(userName : String, password : String, starHasAttId : int) : String
#selectStarHasAttFirst(userName : String, password : String, starHasAttId : int) : boolean
#selectStarHasAttDate(userName : String, password : String, starHasAttId : int) : Date
#selectStarHasAttributeByStarId(userName : String, password : String, starId : String) : StarHasAttribute []
#selectStarHasAttBySourceId(userName : String, password : String, starId : String) : StarHasAttribute []
#selectStarHasAttribute(userName : String, password : String, starHasAttId : int) : StarHasAttribute
#selectStarHasAttByAttributeId(userName : String, password : String, attributeId : int) : StarHasAttribute []

Architettura di VOGCLUSTERS



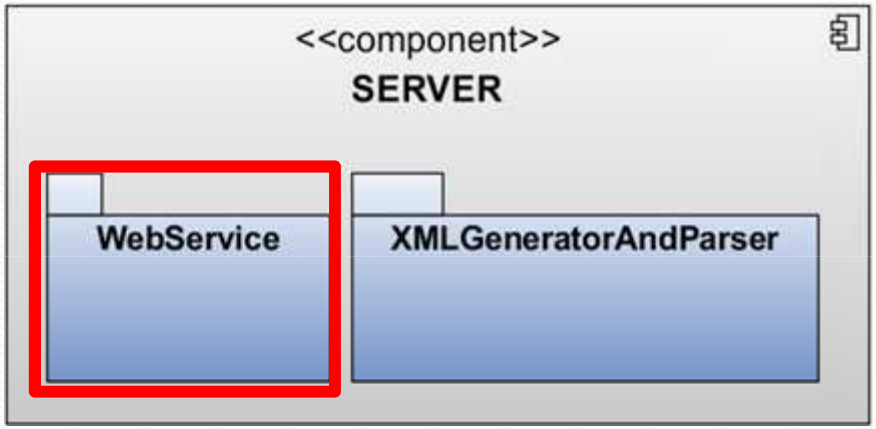
SERVICE LAYER: raccoglie i servizi che il sistema mette a disposizione del FE Layer fornendo a questo l'accesso al DB locale e al DB di DAME.

La componente SERVER



Il server si occupa di ricevere le richieste provenienti dal *FrontEnd Layer*, di elaborarle e di comunicare con il *Data Access and Process Layer* per salvare o recuperare i dati presenti nel database.

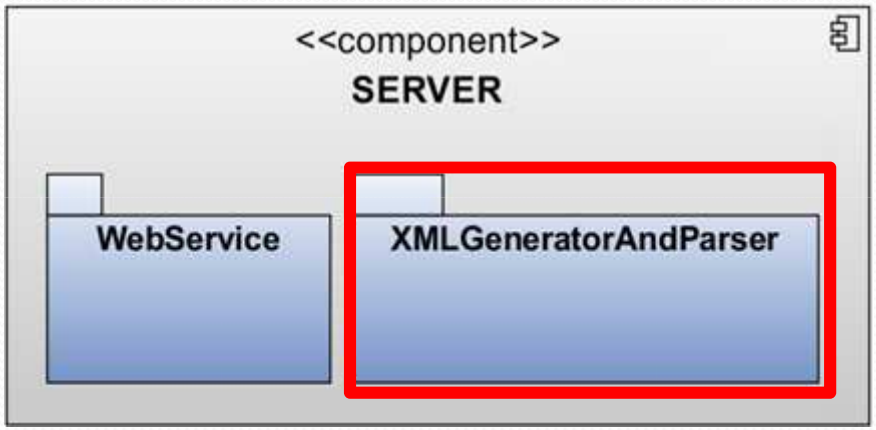
La componente SERVER



Il server si occupa di ricevere le richieste provenienti dal *FrontEnd Layer*, di elaborarle e di comunicare con il *Data Access and Process Layer* per salvare o recuperare i dati presenti nel database.

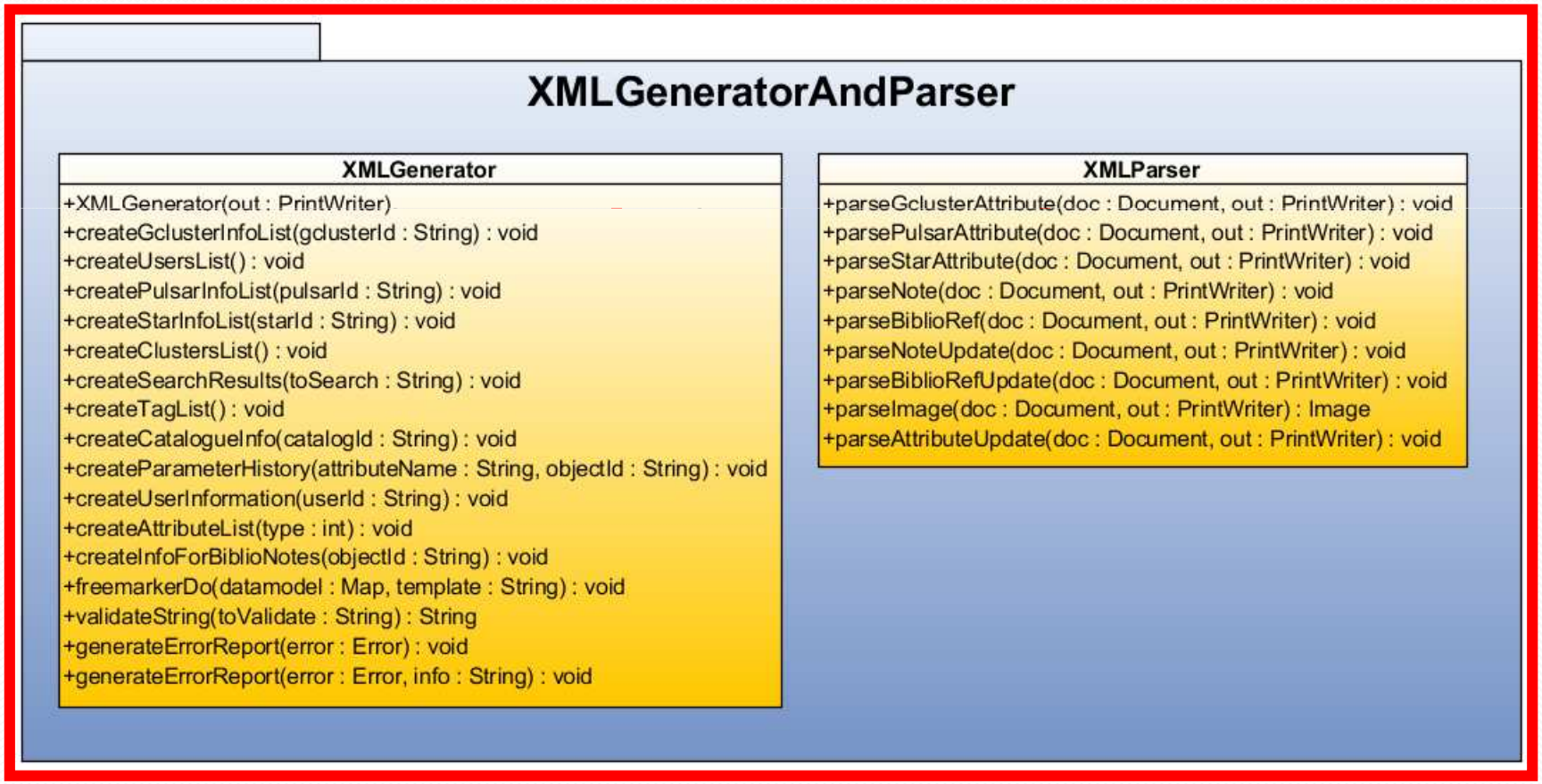
USE CASE	SERVER URI
Autenticazione utente	GET/autenticate?userId=x
Abilitazione utente	GET/enable?userId=x
Visualizza ammasso	GET/vobject?objectId=x
Visualizza catalogo	GET/catalogue?catId=x
Salva nota bibliografica	POST/saveNote?type=x
Salva oggetto	POST/saveVobject?type=x
Ricerca oggetto	GET/search?string=x
Inserimento nuovo attributo	GET/newAttribute?userId=x&objectId=y&attName=z&attDesc=t&attUcd=q&attDatatype=r&attValue=s&type=k
Inserimento nuovo valore attributo	GET/newValue?userId=x&objectId=y&attId=z&attValue=t
Aggiorna attributo	POST/updateAtt
Aggiorna nota bibliografica	POST/updateNote?type=x
Aggiorna valore attributo	GET/updateValue?userId=x&objectType=y&objectHasAttId=t&value=z
Richiedi lista attributi	GET/attList?type=x
Visualizza lista ammassi	GET/clusters
Visualizza lista utenti	GET/users
Cancella attributo	GET/deleteAttribute?userId=x&objectId=y&attName=z
Cancella nota bibliografica	GET/ deleteBiblioNote?userId=x&noteId=y;
Cancella valore attributo	GET/deleteValue?userId=x&objectId=y&vobjHasAttId=z;
Cancella ammasso	GET/deleteVObject?objectId=x
Cancella immagine	DELETE/deleteImage?imageId=x
Recupera informazioni per note bibliografiche	GET/ newNoteRequest?objectId=x

La componente SERVER

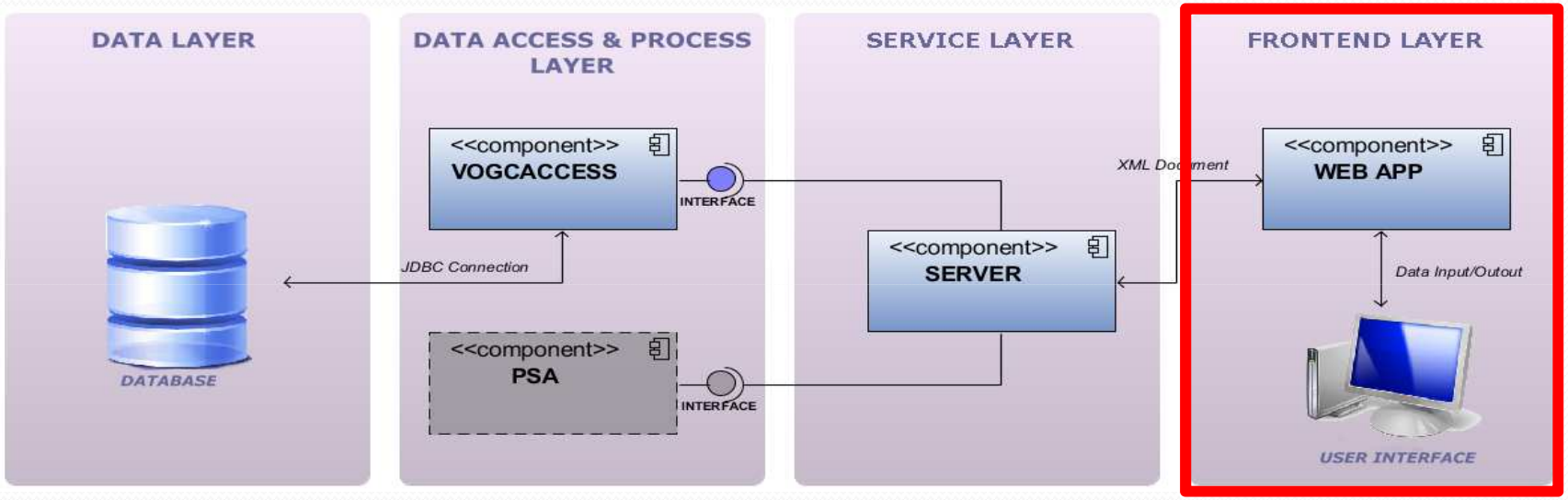


Il server si occupa di ricevere le richieste provenienti dal *FrontEnd Layer*, di elaborarle e di comunicare con il *Data Access and Process Layer* per salvare o recuperare i dati presenti nel database.

La componente SERVER



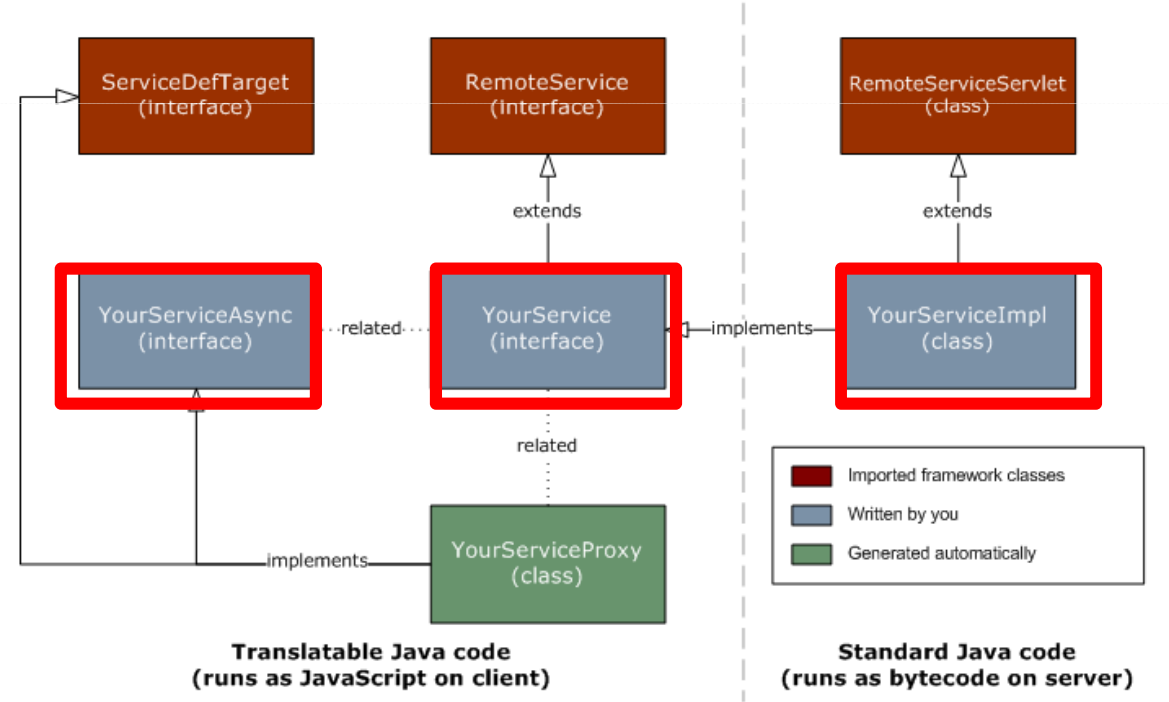
Architettura di *VOGCLUSTERS*



Questa componente, sviluppata con **GWT**, ha il compito di interagire sia con l'utente finale che con le componenti del *Service Layer*, in particolare con le servlet presenti nel package *Web Service*.

GWT (Google Web Toolkit)

GWT è un toolkit open source, sviluppato da Google, che permette di creare applicazioni Web con AJAX utilizzando il linguaggio Java.



La comunicazione client-server avviene attraverso l'invocazione di RPC (*Remote Procedure Call*) attraverso lo scambio di oggetti Java serializzati.

Il compilatore GWT traduce tutto il codice Java in codice per il browser (HTML, CSS e JavaScript).

GWT: l'architettura

GWT Hosted Web Browser: permette di eseguire le applicazioni in *hosted mode*. Il codice Java è eseguito nella JVM senza convertirlo in JavaScript. È possibile utilizzare tutti gli strumenti per il debug disponibili nell'ambiente di sviluppo.

GWT Java-to-JavaScript Compiler: traduce tutto il codice Java in Javascript.

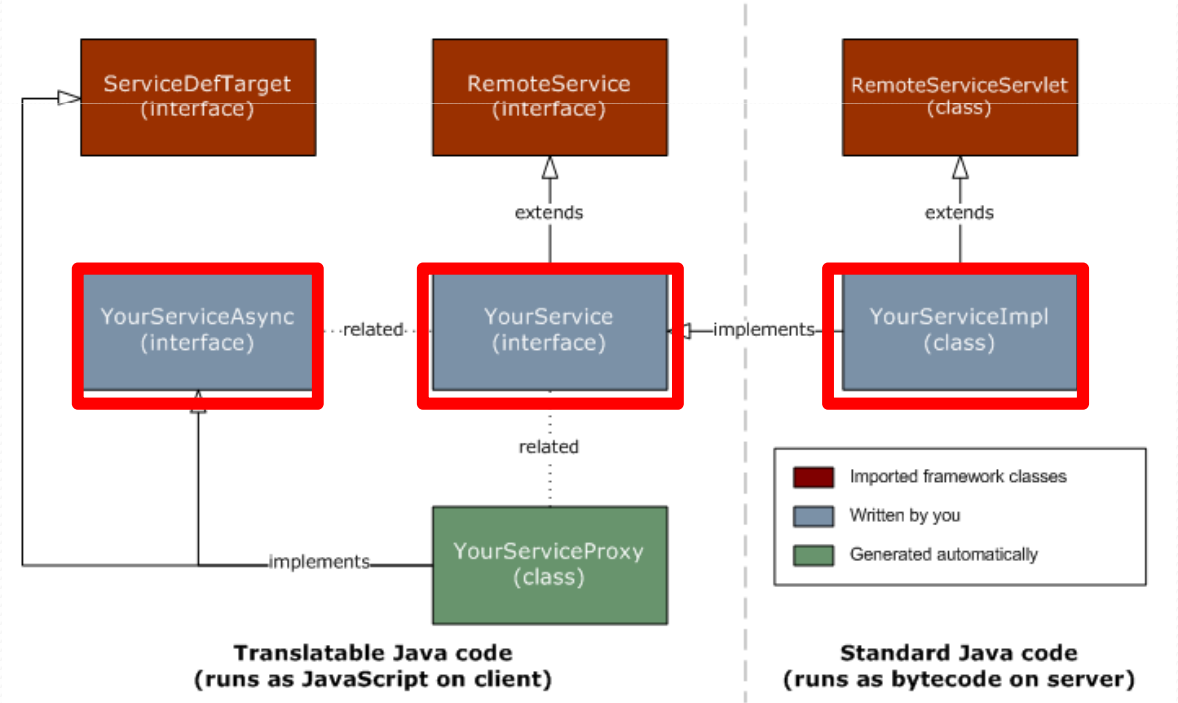
GWT: l'architettura

UI Building Library: fornisce un insieme di classi che consentono di creare widget vari, come pulsanti, caselle di testo, immagini e testo.

JRE Emulation Library: contiene le implementazioni in linguaggio Javascript delle librerie Java standard maggiormente utilizzate (package java.lang.*, java.util.*).

GWT: la comunicazione

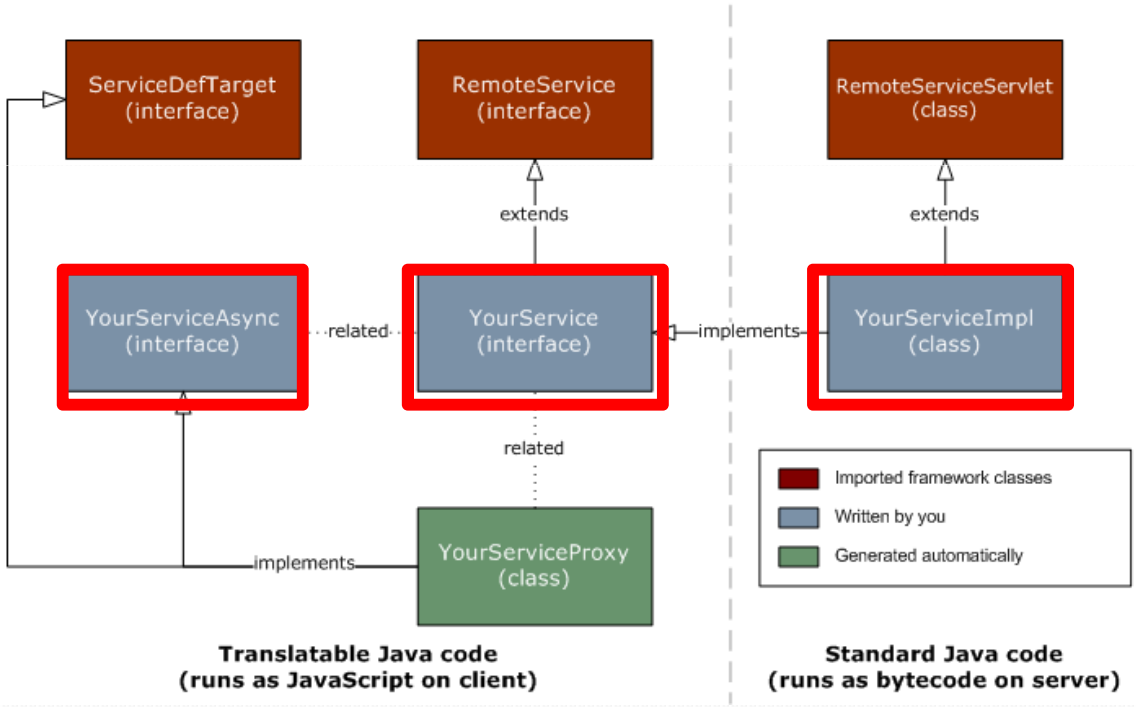
La comunicazione client-server in GWT avviene attraverso l'invocazione di RPC (Remote Procedure Call) attraverso lo scambio di oggetti Java serializzati.



Un'interfaccia che estende RemoteService, all'interno della quale ci sarà la dichiarazione del servizio offerto dal server (la dichiarazione del metodo RPC);

GWT: la comunicazione

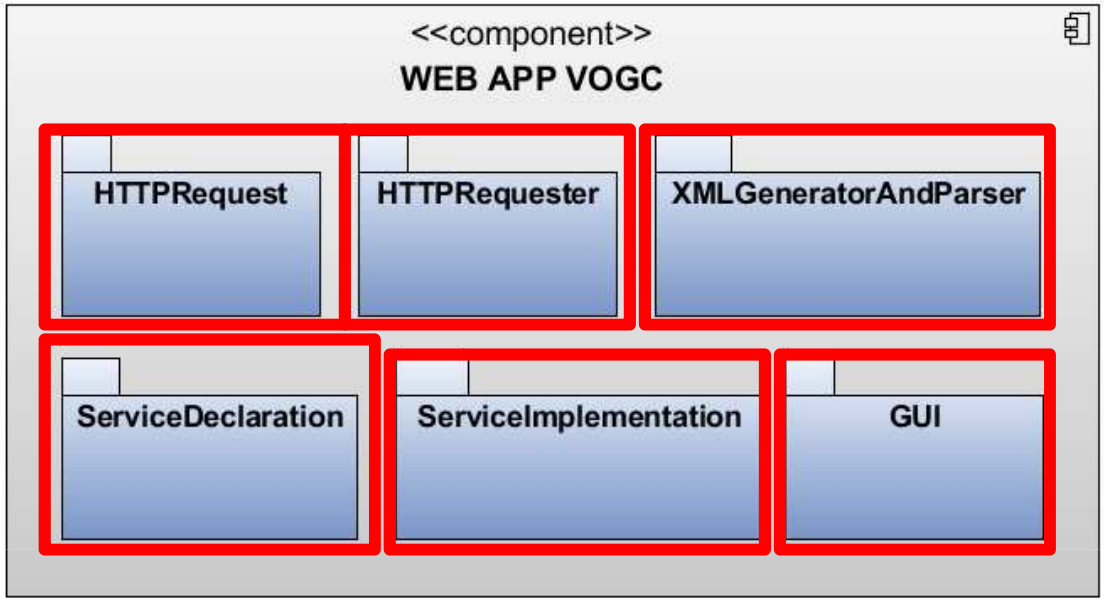
La comunicazione client-server in GWT avviene attraverso l'invocazione di RPC (Remote Procedure Call) attraverso lo scambio di oggetti Java serializzati.



Una classe per l'implementazione del codice lato server che estende RemoteServiceServlet e implementa l'interfaccia creata in precedenza.

Un'interfaccia asincrona, basata sulla definizione dell'interfaccia sincrona corrispondente, utilizzata per richiamare il servizio lato client.

La componente WEB APP VOGC

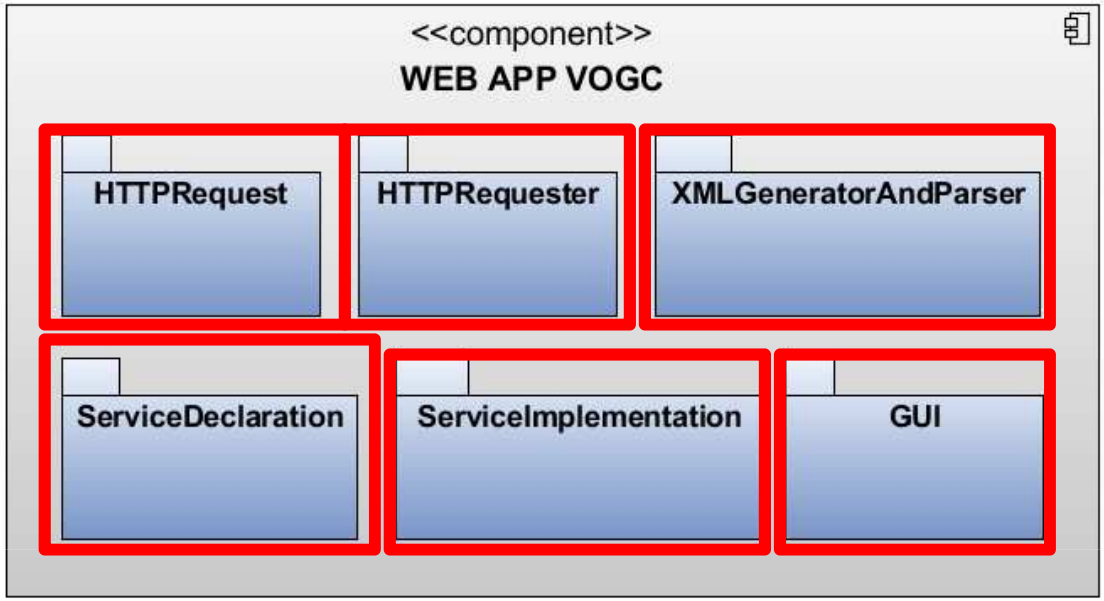


XMLGeneratorAndParser contiene classi che effettuano *parsing* dei documenti XML ricevuti dal Server e la generazione dei documenti XML da inviare al Server.

In **ServiceImplementation** sono presenti le classi che estendono *RemoteServiceServlet* e che danno corpo ai metodi RPC dichiarati nelle interfacce sincrone descritte in precedenza

In **ServiceDeclaration** sono presenti le interfacce sincrone con le dichiarazioni dei metodi RPC. Ogni interfaccia sincrona, implementa *RemoteService* e descrive un particolare "servizio". Per ognuna di essa è presente la corrispondente interfaccia asincrona.

La componente WEB APP VOGC



In **HTTPRequester** è presente la classe *WebAppUseCase*. Questa viene utilizzata per effettuare le richieste HTTP verso il *Service Layer*. Usufruisce dei metodi di *HTTPRequest* ed effettua le varie richieste GET, POST, PUT e DELETE a seconda del caso

In **HTTPRequest** è presente la classe *VogcServerUseCase* che contiene tanti metodi, quanti sono i servizi offerti dal *Service Layer*. Ognuno dei metodi presenti ritorna l'URL del servizio richiesto

In **GUI** sono presenti le classi che si occupano della visualizzazione delle informazioni

Sviluppi futuri

- Realizzare l'interfaccia grafica con SmartGWT.
  1. SmartGWT offre maggiore interattività rispetto agli elementi grafici di base di GWT.
  2. Uniformità con la nuova release della suite DAME.

- Includere la componente PSA per il plotting di grafici statistici e analitici

Conclusioni

VOGCLUSTERS:

- è un progetto integrato nel Programma DAME, condividendone molte peculiarità strutturali e standard progettuali;
- è un framework per l'esplorazione e mining di archivi di dati relativi ad ammassi globulari, oggetti astronomici, che richiedono strumenti di cross-correlation anche basati su ricerche complesse di tipo bibliografico e inferenziale;
- garantisce l'interoperabilità tra archivi delocalizzati di dati astronomici e archivi bibliografici e ipertestuali;
- costituisce uno strumento a disposizione della comunità astrofisica, potenzialmente in grado di migliorare tempi e risultati della speculazione scientifica sugli ammassi globulari.