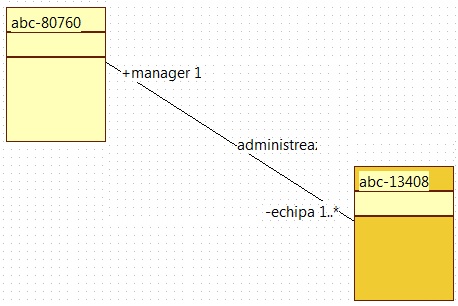
**Elementele grafice**

*Tipuri*  - clasa, interfata, asociere

Intr-o sesiune fiecare element grafic poate fi identificat prin id-ul sau.

*Datele asociate*  - o asociere pe langa reprezentarea grafica contine urmatoarele informatii:



* Numele asocierii
* Numele, vizibilitatea si multiplicitatea entitatilor conectate de asociere

Datele sunt stocate in format XML

<association>

<element\_id>3</element\_id>

<start\_element\_id>1</start\_element\_id>

<end\_element\_id>2</end\_element\_id>

<element\_title>administreaza</element\_title>

<start\_name>manager</start\_name>

<start\_visibility>+</start\_visibility>

<start\_is\_navigable>1</start\_is\_navigable>

<start\_aggregation>NONE</start\_aggregation>

<start\_multiplicity>1</start\_multiplicity>

<start\_connected\_element>abc342258536</start\_connected\_element>

<end\_name>echipa</end\_name>

<end\_visibility>-</end\_visibility>

<end\_is\_navigable>0</end\_is\_navigable>

<end\_aggregation>NONE</end\_aggregation>

<end\_multiplicity>1..\*</end\_multiplicity>

<end\_connected\_element>abc-1560389792</end\_connected\_element>

</association>

Clasa AssociationElementConvertor si ClassElementConvertor se ocupa de serializarea si deserializarea datelor.

**Conceptul de sesiune**

Intreaga logica asociata conceptului de sesiune este organizata in clasa **Session** din pachetul **org.umldiagram.** Interactiunea cu baza de date este intermediate de clasa SessionHelper.

Rolul sesiunii este de a asigura cadrul desenarii unei diagrame UML. Sesiunea este implementata folosind o camera de chat (MultiUserChat) si parametrii care ii descriu starea. Pentru a functiona corect sesiunea trebuie sa indeplineasca urmatoarele cerinte:

1. Sa genereze fiecarui element grafic un id unic prin care sa poata fi identificat de catre orice client.

**Implementare:** La pornirea unei sesiuni existente este incarcat din baza de date in atributul elementId, id-ul ultimul element grafic creat.

Metoda: joinsSession

elementId = SessionHelper.*getLastElementIdOfSession*(sessionId);

La primirea unei cereri de creare a unui element

Metoda: processPacket

**if** (lReceivedMessage.getSubject().equals("createElementRequest")) {

………

++elementId;

}

1. Sa asigure persistenta spatiului de lucru.

Implementare: In momentul in care toti utilizatorii au parasit sesiunea, reprezentarea XML a spatiului de lucru este salvata intr-un fisier cu numele sessionId.txt .

Metoda: userLeftTheSession

**if** (noOfUsers == 0) {

updateSessionContentFile();

}

1. Initializarea camerei de discutii (room chat) in interiorul careia se vor difuza instructiunile de desenare.

**if** (muc == **null**) {

muc = **new** MultiUserChat(connection.getXMPPConnection(), "session" + sessionId + "@conference.smsfeedback.com");

muc.create("UmlDesigner-Bot");

muc.sendConfigurationForm(**new** Form(Form.*TYPE\_SUBMIT*));

}

In cazul in care camera nu exista, este creata si este introdus un bot care va asculta discutia din camera.

Prin intermediul botului aplicatia server v-a transmite instructiuni catre ceilalti participanti.

1. Sa tina evidenta elementelor selectate.

Maparea utilizator - element selectat se face printr-un table de hash, in cod ii corespunde obiectul de tipul HashMap numit selectedElements. Cheia este reprezentata de numele utilizatorului, iar valoarea de id-ul elementului.

1. Sa mapeze adresele reale de adresele asignate de chat.
2. Sa deblocheze elementul selectat de utilizator in momentul in care acesta paraseste camera.
3. Sa furnizeze in format XML ultima versiune a spatiului de lucru.

Datele despre o sesiune sunt pastrate in baza de date in tabelul sessions. Tabelul contine id-ul sesiunii si ultimul id generat pentru elementele grafice.

La pornirea unei sesiuni este preluat din baza de date ultimul id

Logica sesiu

Un client poate crea o noua sesiune sau poate participa intr-o sesiune care exista. Mai jos sunt prezentate cele doua procese.

**Crearea unei sesiuni**

Ce am scris aseara ….. + ca organizare

* Comportament asteptat
* Flowchart
* Bucati relevante de cod

**Participarea intr-o sesiune**

**…………**

**Blocare element**

Pentru a opera modificari asupra unui element, mutare, schimbare parametrii, clientul trebuie sa obtina controlul asupra lui. Prin selectia elementului clientul isi exprima aceasta intentie. Daca elementul nu este selectat de nici un alt participant, botul (mediatorul) ii va da un raspuns afirmativ la cerere si marcheaza elementul ca selectat. Ceilalti participanti vor marca elementul ca selectat si ii vor actualiza reprezentarea grafica. Pentru un element blocat nu se poate face cerere pentru a obtine controlul asupra lui.

**Deblocare element**

Clientul care are controlul asupra unui element anunta toti participantii din sesiune ca elementul este deblocat. Botul (mediatorul) intercepteaza acest mesaj si actualizeaza **“starea de selectie”** a respectivului element.

**Explicare mecanism de deblocare element**

**Motivatie:** un utilizator poate parasii o sesiune de lucru, iar elementul pe care il are selectat va ramane blocat, fara ca alti utilizatori sa il poata debloca si sa poata lucra pe el.

Pentru a detecta inchiderea conexiunii unui client si deblocarea ultimei resurse pe care acesta a folosit-o se fac urmatorii pasi:

* Clientul trimite o prezenta directa catre aplicatia server, pentru ca aplicatia server sa fie notificata atunci cand clientul inchide conexiunea.
* La participarea intr-o camera de chat clientul trimite un mesaj cu subiectul myIdentiy avand in body id-ul real. (clasa Session -> metoda processPacket). Se adauga in obiectul de tip HashMap addressMap intrarea id real – id chat.
* La primirea unei cereri de selectie din partea unui user, in cazul in care selectia este posibila in tabelul selected