DDD: one-to-many relationship between user aggregate root and almost all entities in other aggregates

Asked 6 years, 10 months ago Active 2 years, 4 months ago Viewed 3k times

2

4

I have the following DDD scenario, grouped into the following aggregates:

User,
Friends (User Associations),
File (for user uploading),
Galleries (grouping of files),
Messages (user communication),
Groups (users can create and other members can join),
GroupMessages (messages sent to all members of a group),
GroupForums (group members can discuss various topics)

This is where it gets confusing. A user is associated with everything down to GroupForums. It seems illogical to have to go through the User repository to access the other aggregates although, from a cascading standpoint, if I removed the user, technically, the records associated with the user should go away as well.

It seems as if I should not add all of the one-to-many associations that exist here to the user entity either, as hydrating from the database seems to be ridiculous, especially if I try pulling every record associated with the user. What is the recommended strategy for organizing your aggregates, and repositories as well as proper way of dealing with a lot of one-to-many relationships for a given entity?

c# domain-driven-design ddd-repositories

edited Nov 9 '17 at 20:36
halfer
17.6k ● 8 ● 69 ● 140

asked May 4 '13 at 20:31
user1790300
2,221 ● 4 ● 26 ● 73

7

fine for aggregate roots to be associated or even for one to 'belong' to another. However, you need to look at whether an entity can exist without the AR. If it can it probably has its own life-cycle and should be an AR. If it can not it is part of the aggregate. This can be tricky to distill.

You need to have a very clear boundary around your ARs. For example, even though a Forum may require a User to create it this does not mean that the Forum needs to (or even can) be deleted when the user is deleted. So the User in the Forum may become, say, the `ForumCreator` (a value object) that contains the user name and id only. When the User is deleted then the forum can continue its existence.

In the Order/OrderLine/Product scenario it would not make much sense to delete all order lines that contain a specific product if you choose to delete it. I know that a product probably should never be deleted but we'll use it as an example. You would simply have the relevant product data 'denormalized' into the order line, e.g.: product id, product name. So even if the product name happens to change it does not mean that all order lines need updating, or even should be updated. In fact, the order line represents a point in time and the 'original' product name should be retained. The purchaser may have ordered 'Some lirril product' and then the name changed to 'Little product'. Not the same thing although it is the exact same product. The purchaser only remembers the original.

I hope that makes sense and helps in some way. You definitely need to find those hard edges to your object graph to get to the real aggregates.

answered May 5 '13 at 16:53
Eben Roux
11.2k ● 2 ● 19 ● 41

For the user aggregate example, I used the Entity Framework POCO entity generation utility and it created the User object and added IList<> properties for all of the entities it has one-to-many relationships with. By the way, I am using dapper as a micro orm. Should I remove the IList<> property entries since I would rather access them from their respective repositories? In this scenario, should I remove the IList(one-to-many) association from the User entity and have access to Groups go through a GroupRepository instead of a User repository? – user1790300 May 5 '13 at 20:17

I really do not know EF so I wouldn't know how to manipulate the files correctly. For that matter, I do not know any ORM :) --- w.r.t. the User->Groups it is going to depend on your domain but I'm guessing that user to group is many to many so I would remove that list from the user. Maybe have either a list of group ids the user belongs to or some Value Object (UserGroup) that contains the id and the group name. – Eben Roux May 6 '13 at 4:47

Let's look at it from a pure entity and aggregate perspective, what is the recommended approach for organizing the relationships between the user and the Friends, File, Galleries, Messages, Groups, GroupMessages, GroupForums entities? To me, it seems like there should be no IList entries in the user entity for each of these and instead each of the other entities should contain a property to the User entity. – user1790300 May 6 '13 at