Result and evaluation

A. Evaluation of our approach

We need to validate the results of the two cases described above. The cases we use come from real practice, and we can take the actual module we divide as consideration. To make it more reasonable and formal, we create a checklist as follows:

1. Functionality: Does the decomposed services in the result can fulfill the functional requirements of the system?

2. Cohesiveness: Do the data in one service are cohesive to each other?

3. Compatibility: Do the data not in one service have lose couples?

4. Consistency: Do the operation’s input data all join this operation?

5. Consistency: Are the operation’s output data all generated by this operation?

6. Expect Result: Does the result similar to the system which has been realized currently?

In order to give a general evaluation to the results, we define 3 categories according to the checklist:

1. excellent (true or part of true to all questions in the list)

2. reasonable (4 or more are true or part of true)

3. unreasonable (<4 are true or part of true)

From discussion above, we can see that the result of the first case is excellent while the second case is reasonable but a little different from the best result we expect.

B. Comparison to Service-Cutter

We also compare our method with use the method in Service-Cutter[]. Since our methods are based on different models, we should build another model based on Entity Relation Model and other user representations according to their criteria. So, we convert our two cases to Service-Cutter’s input format and get the decomposition results as follows:

//2个手绘图

Compared the procedure and result:

1. Procedure to build the model: There are some differences when using these two methods based on one common business logic. Our method based on the Data Flow Diagram and it well described the workflow of the real system so that it’s easy to understand. However, the method used by Service Cutter is based on Entity Relation Model and many other complex criteria which make the modeling process more complex and not intuitionistic.

2. The decomposition results: We can easily see our methods’ advantage over theirs in these two cases.

Case 1: The first case using their method divide all entities to just one service which obviously is unreasonable.

Case 2: They break the system into 3 services, which seems reasonable. However, one obvious problem is that entity DB\_helper’s two nanoentities which should be in one service are divided into two different services. Also, from our experience and Entity Relation Model, entity Time should be together with entity Play which are actually not in their result. So, its result has some problems.