- MDRR: A custom-made MongoDB-based repository.
- AT: AssessDocking, a custom-made tool that filters' "good" docking results based on a threshold.
- ADS: PubChem, the existing external database of ligand properties.
- DM: a custom-made DM.

Because this diagram was derived from the basic diagram of the framework, it is reasonable to assume that Scenario 2 fits the framework. To provide a more precise analysis, the detailed diagram for each element and its interfaces will be described in this section. A list of the interfaces, and a formal description will be used to confirm that the proposed elements can be used as element types prior to starting the coding step. This section will focus on segments that are different from the description of Scenario 1, and comment on the added value of implementing Scenario 2 using the methodology.

8.3.1 Abstract descriptions of Scenario 2

MDE: The extended version of Raccoon2 The MDE in Scenario 2 can be the same as the one in Scenario 1 (Section 8.2.1). Practically the entire element can be reused with the difference that instead of a DeepAlign threshold as in Scenario 1, now the user inputs a PubChem property name and threshold. Figure 8.14 shows that a detailed diagram of the Raccoon2 extension for Scenario 2 can be derived from the generic diagram of an MDE. More importantly, it shows how the methodology allows the reuse of previously implemented elements. When drawing the detailed diagram, it becomes evident that most of the interfaces are exactly the same, and the core computation (the docking) is the same as in the detailed diagram of Raccoon2 used for Scenario 1. The conclusion is that the abstract description of Raccoon2 is similar enough for it to be used as an MDE in Scenario 2 as well. The fact that the element can be reused is determined now, before any coding begins. It is determined by searching a library of abstract descriptions of already implemented elements (Technique 2 in Section 7.3). At the moment, this library contains only one MDE, but the same method of drawing the detailed diagram and comparing the interfaces and the core computation, can be used to search a large library.

Raccoon2 interfaces Since the Raccoon2 element can be reused, the list of interfaces is nearly the same. The only difference being the need for a PubChem threshold instead of a DeepScore threshold.

Formal description of Raccoon2 The formal description of the Raccoon2 element of Scenario 2 (Appendix D page 179) can be derived in the usual way from the formal description of the element type MDE. The result is the same formal description as in Scenario 1 (Figure 8.3).

MDRR: a custom-made MongoDB repository Similarly, Scenario 2 can reuse the same MDRR as in Scenario 1 (Section 8.2.1). The same technique for searching a library of already implemented elements can be used to determine whether an element can be reused. The detailed diagram of this MDRR (which can be seen in Figure 8.14), is nearly identical to the detailed diagram of the MDRR in Scenario 1 which is the only MDRR present in the library of implemented elements.

Interfaces of the custom-made MongoDB-based MDRR The interfaces of this MDRR are nearly the same as the MDRR in Scenario 1. The only difference is in interfaces 5a-d which are providing the AutoDock Vina affinity and the PubChem property threshold, the list of current docking results, and a list of ligands of "good" docking results.

Formal description of the custom-made MongoDB-based MDRR When deriving the formal description of this MDRR (Appendix D, page 179), the focus is on the interfaces for inserting and selecting docking results. This results in the same formal description as the one for the MDRR of Scenario 1.

AT1: Assess docking results The same technique for searching a library of already implemented elements can be used when implementing this AT (Technique 2 in Section 7.3). When drawing the detailed diagram for this AT, it can be compared to a library of three already implemented elements (as implemented in Scenario 1).

When compared to the AT:DeepAlign, there is a clear difference in the interfaces. AT:DeepAlign needs to send the results and a user-provided threshold to another AT for assessment. Whereas, the needed AT assesses docking results and sends them to an MDRR for storage, and a DM for summarising. When compared to the AT:AssessDeepAlign, there are more similarities in the interfaces. The difference is that AT:AssessDeepAlign requires input from another AT, whereas the needed AT requires input from an MDRR. There is a big difference in the core computation, the AT:AssessDeepAlign filters structural alignment results, and the needed AT should filter docking results. Finally, when comparing the needed AT with the AT:AssessDocking, it is clear that the interfaces are the same (require input from MDRR, provide results to MDRR and DM), and the core computation is the same (assess docking results).

Therefore, it can be concluded that the AT:AssessDocking can be reused. The same name for the AT can be used and the detailed diagram is nearly the same. The only difference is in the naming of the interfaces 5a and 5c.

This comparison was done manually, examining and comparing the abstract descriptions of the required element to the already implemented elements by hand. Ideally, this comparison would be automated and it would use a database of already implemented abstract descriptions (Section 10.2).

Formal description of the custom-made threshold-based docking result assessment AT The formal description of this AT (Appendix D, page 182) has the same method *goodDocking* and Z schema *AssessPreviousDocking* as in Scenario 1.

ADS: PubChem Implementing Scenario 2 will show how a new element type can be used, since in Scenario 1, there was no ADS (as previously stated in Section 8.1). Since there is no library of previously implemented elements of the type ADS, this scenario can use the third technique of the methodology (Section 7.3) and create an abstract description of an existing tool, then compare it to the generic abstract description of the element type to determine whether it can be used.

PubChem is a repository that contains data about chemical substances. It is split into three databases: Substance, Compound, and BioAssay [17, 240]. As of March 2018, the Compounds database contains more than 94 million items. PubChem can be accessed programmatically through the Power User Gateway (PUG) interface [241]. In Scenario 2, PubChem is proposed as the core computation component of an ADS. The detailed diagram of an ADS of the framework shows that an ADS needs to provide the data through an interface for an MDRR and a DM. When creating the detailed diagram of PubChem, the existence of such an interface was checked. Indeed, PubChem's PUG-REST API is an interface provided by PubChem which can be used to read the data it stores. If the PUG-REST API can be used to obtain the value of a ligand property for a list of ligands, as required by the interfaces 5b and 5d, then PubChem can be used as an ADS. If in the coding of Scenario 2, an HTTP request is sent to the PUG-REST API for each ligand in the list, this will be possible.

Therefore, since the PubChem element, as drawn in the detailed diagram (Figure 8.12), can provide and require the needed interfaces as an ADS, and the core computation segment (the Compounds database) contains data about ligand properties, the PubChem element can be the ADS for Scenario 2. However, the value returned by PubChem would need to be additionally compared with the threshold.

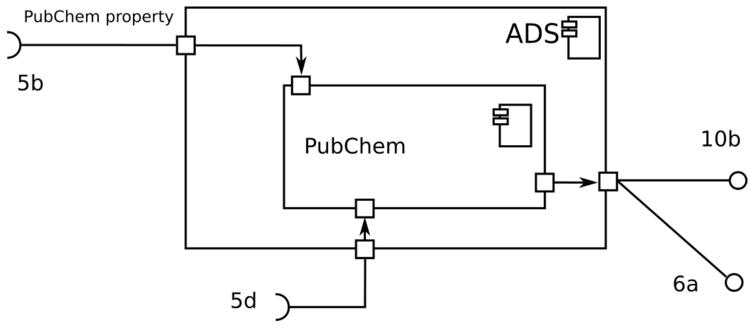

Figure 8.12: The diagram of the ADS PubChem.

PubChem Interfaces

5b, 5d. PubChem requires the user provided PubChem property, and a list of ligands.

6a. PubChem provides the ligand property value, to the MDRR to keep track.

10b. PubChem provides the ligand property value, to the DM for summarising.

Formal description of PubChem This scenario is the first example of a formal description of an ADS element. One of the aims of implementing of Scenario 2 is to show how a new element type, which hasn't been used before, can be introduced (as mentioned in Section 8.1). The abstract description of an ADS in the framework is generic and does not provide details about the type of data a particular ADS element would store. The specific formal description of PubChem (Appendix D, page 182) includes the *checkPubChem* method. This method specifies that it requires a ligand and a property as input. After checking the data stored in PubChem, it provides a positive or negative response based on whether the property of the ligand is within a given threshold. The schema *PubChem*, which is derived from the generic schema *SelectAdditionalDataInfo*, shows how the filtered results based on the ligand property can be obtained.

DM: custom-made element The DM combines the result of AT1 (if the docking is good) and the ADS (if the ligand property is within the threshold). The DM is an element which is specific to each scenario. This can be seen after comparing the detailed diagram of this DM to the DM from Scenario 1. The interfaces seem equivalent, however the main difference is the core computation. Because each scenario would provide a different decision as a final result, the core computation of each DM would be different. In Scenario 2, the DM provides a list of ligands filtered based on a property from an external additional data source. Whereas, in Scenario 1, the DM suggested a ligand for the next docking. Therefore, the DM from Scenario 1 cannot be reused.

Interfaces of the custom-made DM

10a-b. The DM should require the results from the AT1 and the ADS.

13 The decision, in this case the list of filtered ligands according to the specified property, should be provided to the MDRR.

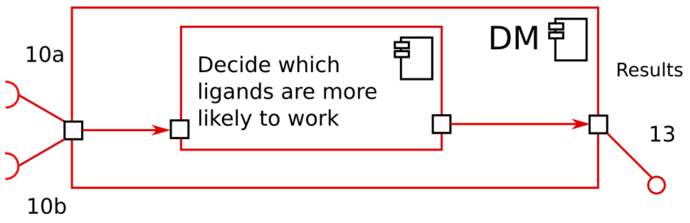

Figure 8.13: The diagram of the DM.

Formal description of the custom-made DM The formal description of this custom-made DM is derived from the element type DM of the framework. The Z schema *Decision-Maker_Custom* (Appendix D, page 183), which is derived from the schema *DecisionMaker* of the framework, specifically uses *makeADecisionPreviousResults*. It requires two previous results as input. In this case the first result is a list of assessed previous docking results, and the second is a list of filtered results based on ligand properties.

Detailed diagram of entire Scenario 2 Similarly to Scenario 1, a complete detailed diagram of Scenario 2 can be created (Figure 8.14). It confirms that Scenario 2 fits the framework because it is equivalent to the detailed diagram of the entire framework (Figure 6.1). There are a total of 19 interfaces between the elements (numbered according to the numbering-scheme of the framework).

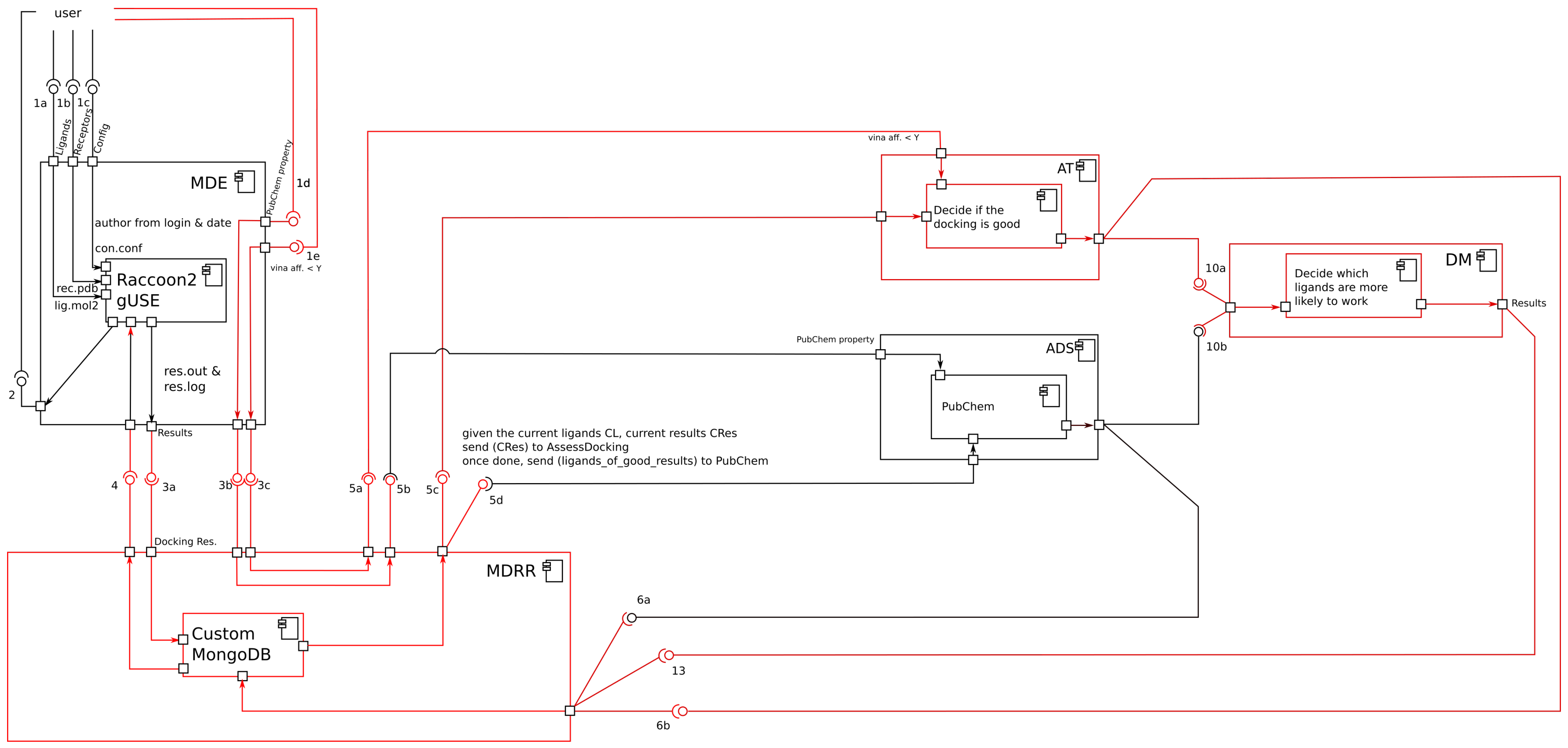

Figure 8.14: The detailed diagram of Scenario 2.