any). In order to do that, the Explanation Engine implements the black-box method presented in [10].

The feedback is translated back to the original SQL representation by the Test Controller and the Mapping and Mapped Schemas Extractor, and shown to the user through the GUI. If the CQC_E Method Engine provides a database instance, and since this instance corresponds to the integrated schema that resulted from the problem reformulation, it has to be translated in terms of the original mapped schemas. Similarly, if the feedback is an explanation (a set of constraints) and since these constraints belong to the integrated schema, they have to be translated in terms of the original mapped schema constraints and mapping assertions.

The whole MVT tool has been implemented in the C# language using Microsoft Visual Studio as a development tool. Our implementation can be executed in any system that features the .NET 2.0 framework. Some screenshots are shown in Figure 2.

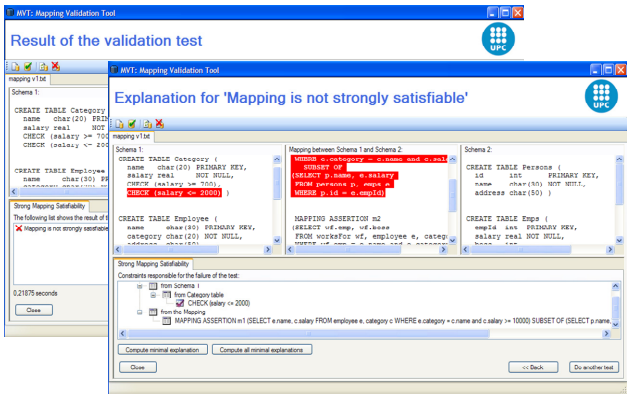

Figure 2: Screenshots of MVT.

4. RELATED WORK

Recently, other tools related with the mapping validation problem have been presented [2, 4]. The main difference of our tool with respect to them is that they need schema instances in order to perform the validation. Our tool only requires the mapping and mapped schemas definitions to be provided, and it is therefore able to reason over the mapping itself rather than relying on specific instances that may not reveal all the potential pitfalls.

The SPIDER tool demonstrated in [2] is a mapping debugger for source-to-target tuple-generating dependencies mappings, based on the computation of *routes* [5]. Basically, the user can select a set of target tuples, and see how this tuples were obtained from the source instance through the mapping. Since routes are intended to allow the user to explore and understand a given schema mapping, this work can be seen as complementary to ours. As we demonstrate here, our tool sometimes provides schema instances as feedback for a certain validation test. Therefore, routes could be used to help the designer to understand this feedback, and to make easier the detection and fixing of the problems.

The Spicy system [4] is aimed at helping the designer to choose among the different candidate mappings (tuple-generating dependencies) the ones that represent better transformations of the source into the target. Source and target schema instances are required. Each candidate mapping is executed over the source instance in such a way that a new instance for the target schema is obtained, and this new instance is compared with the available target instance. At the end, the user gets a ranked list of mappings, suggesting which ones are believed to better reproduce the target. At this point, the validation information provided by our tool, combined with the similarity measure attached by Spicy, might help the designer to choose and refine the final mapping.

Another tool that has appeared recently is Muse [1], a mapping design wizard that assists designers in understanding and refining schema mappings. In particular, it guides the designer on the choice among alternative mapping definitions by constructing synthetic examples that illustrate the differences among them. However, the construction of such examples is not aimed at revealing potential mapping-definition flaws such as redundancies and information loses. In this sense, our tool complements Muse by enabling the designer to validate and refine the chosen mapping definitions. Although our current version of the tool does not deal with nested mappings and nested relational schemas as Muse does, we are already working to include such features in a next release. Instead, our tool does handle mapping definitions featuring negations and order comparisons, which Muse does not consider.

5. REFERENCES

[1] B. Alexe, L. Chiticariu, R. J. Miller, D. Pepper, W.-C. Tan: Muse: a system for understanding and designing mappings. In SIGMOD Conference, pp. 1281-1284, 2008.

[2] B. Alexe, L. Chiticariu, W.-C. Tan: SPIDER: a Schema mapPIng DEbuggeR. In VLDB, pp. 1179-1182, 2006.

[3] P. A. Bernstein, L. Haas: Information integration in the enterprise. Commun. ACM 51(9), pp. 72-79, 2008.

[4] A. Bonifati, G. Mecca, A. Pappalardo, S. Raunich, G. Summa: The Spicy system: towards a notion of mapping quality. In SIGMOD Conference, pp. 1289-1294, 2008.

[5] L. Chiticariu, W.-C. Tan: Debugging Schema Mappings with Routes. In VLDB, pp. 79-90, 2006.

[6] C. Farré, E. Teniente, T. Urpí: Checking query containment with the CQC method. Data Knowl. Eng. 53(2), pp. 163-223, 2005.

[7] J. Madhavan, P. A. Bernstein, P. Domingos, A. Y. Halevy: Representing and Reasoning about Mappings between Domain Models. In AAAI/IAAI, pp. 80-86, 2002.

[8] L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernández, R. Fagin: Translating Web Data. In VLDB, pp. 598-609, 2002.

[9] E. Rahm, P. A. Bernstein: A survey of approaches to automatic schema matching. VLDB J. 10(4), pp. 334-350, 2001.

[10] G. Rull, C. Farré, E. Teniente, T. Urpí: Computing explanations for unlively queries in databases. In CIKM, pp. 955-958, 2007.

[11] G. Rull, C. Farré, E. Teniente, T. Urpí: Validation of mappings between schemas. Data Knowl. Eng. 66(3), pp. 414-437, 2008.

[12] G. Rull, C. Farré, E. Teniente, T. Urpí: Providing Explanations for Database Schema Validation. In DEXA, pp. 660-667, 2008.

[13] E. Teniente, C. Farré, T. Urpí, C. Beltrán, D. Gañán: SVT: Schema Validation Tool for Microsoft SQL-Server. In VLDB, pp. 1349-1352, 2004.