

## RESEARCH OUTPUTS / RÉSULTATS DE RECHERCHE

### Business-Driven Data Recommender System

Pinon, Sarah; Burnay, Corentin; Linden, Isabelle

*Published in:*

Journal of Computer Information Systems

*DOI:*

[10.1080/08874417.2023.2237450](https://doi.org/10.1080/08874417.2023.2237450)

*Publication date:*

2023

*Document Version*

Peer reviewed version

[Link to publication](#)

*Citation for published version (HARVARD):*

Pinon, S, Burnay, C & Linden, I 2023, 'Business-Driven Data Recommender System: Design and Implementation', *Journal of Computer Information Systems*. <https://doi.org/10.1080/08874417.2023.2237450>

#### General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.




- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

#### Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.




## Title page

### **Business-driven Data Recommender System: Design and implementation**

Sarah Pinon<sup>a</sup> , Corentin Burnay<sup>a</sup>  and Isabelle Linden<sup>a, b</sup> 

<sup>a</sup> Namur Digital Institute, MINDIT Research Center, University of Namur, Namur, Belgium; <sup>b</sup> Namur Digital Institute, FOCUS Research Center, University of Namur, Namur, Belgium

# Business-driven Data Recommender System: Design and implementation

Sarah Pinon<sup>a</sup> , Corentin Burnay<sup>a</sup>  and Isabelle Linden<sup>a, b</sup> 

<sup>a</sup> Namur Digital Institute, MINDIT Research Center, University of Namur, Namur, Belgium; <sup>b</sup> Namur Digital Institute, FOCUS Research Center, University of Namur, Namur, Belgium

## ARTICLE HISTORY

Compiled July 5, 2023

## ABSTRACT

Self-Service Business Intelligence (SSBI) increases decision-making reactivity of companies by facilitating the data use by non-IT experts. An important SSBI dimension is data querying where businesspeople create their own queries by reducing the technical complexity of formal languages like SQL. However, existing solutions ignore two other key challenges of data querying identified in the literature: the databases technical jargon and the data overload. In this paper, we propose, following the Design Science Research methodology, a framework (i.e. DatAssistant) to complement existing querying solutions with two new theoretical artifacts. The first bridges the semantic gap between technical databases and businesspeople via a business-aware ontology of the Data Warehouse mapped to the business Data Catalog. The second artifact filters data overload by mobilizing a hybrid recommender engine combining semantic systems and business rules. This paper then demonstrates the validity and applicability of the framework through its technical implementation in a real-world environment.

## KEYWORDS

Self-Service Business Intelligence; Data Query; Recommender System; Semantic Systems; Data Catalog;

## Introduction

Self-Service Business Intelligence (SSBI) is used in companies to produce timely, factual and contextualised information for consumption by decision makers, thereby reducing uncertainty and enhancing decision outcome quality.<sup>1,2,3</sup> In a typical BI approach, IT Experts transform raw data originating from various business data sources (ERP, CRM, etc.) into information that can be consumed by business people under the form of reports or interactive dashboards.<sup>4</sup> The logic behind SSBI is to remove the cumbersome process of collecting and satisfying the fast changing requirements of business people and to let themselves produce the analytics they want.<sup>5,6</sup> Making BI actually self-service is hard and requires various mechanisms to ease the interaction between people without in-depth IT skills and the BI system. This challenge can be treated on different dimensions and we focus in this paper on the data dimension, where support is provided to business users to extract from databases the data they seek. This is achieved with the help of a Data Query Support Solution (DQSS). Data querying is, at the moment, mostly performed via formal and technical languages that are not familiar to business people, such as SQL.<sup>7,8</sup> The interest of democratizing DQSS has been widely demonstrated in the literature through the development of Visual Query Languages or Natural Language Interfaces for databases (NLIDBs).<sup>7,8,9</sup>

Recent studies on the challenges faced by non-IT experts when using data<sup>10,11,12</sup> however emphasize significant improvements are still possible to make querying more accessible. Existing solutions focus on the technical complexity of data query languages – the syntax – but tend to overlook the meaning and content behind the queries, – the semantic. Recent studies on the challenges of non-IT experts’ when using data identify indeed two other major issues which are currently not treated by existing DQSS. The first challenge corresponds to the technical jargon used in database management. Technical nomenclature, decided by IT-experts during database design stages, is not systematically aligned with the business jargon, leading to possible semantic confusion for business users.<sup>10,13</sup> The second challenge is the data fields overload of current databases. With the increasing volume of data,<sup>14</sup> it is becoming more and more complex to identify the relevant information, and this task requires more and more technical skills which are not readily available for business people.<sup>11,3</sup> We found no DQSS explicitly dealing with these two last, yet critical, challenges. To fill this literature’s gap, our paper contributes to information systems theory using the Design Science Research (DSR) methodology of Peffers and Tuunanen.<sup>15</sup> We propose two types of contributions articulated according to prescriptions of Gregor and Hevner:<sup>16</sup>

- **Design artifacts:** We have developed the DatAssistant framework, a DQSS framework for non-IT experts that builds and extends existing DQSSs, and more specifically existing NLIDBs, composed of two new theoretical artifacts:
  - **Artifact 1 - High-Level Business DWH Ontology:** Semantic bridge between the technical terminology of databases and the business jargon of non-IT experts built using Ontological Semantic Systems<sup>17</sup> and the Data Catalog concept;<sup>18</sup>
  - **Artifact 2 - Hybrid Data Recommender Engine Architecture:** Complete architecture of a Recommender Engine<sup>19</sup> that semantically and technically filters data fields, using a set of Business Rules<sup>20</sup> and Semantic Systems,<sup>21</sup> to satisfy the user’s business data query.
- **Implement artifacts:** We have technically implemented the DatAssistant framework and applied it to a real case to demonstrate the feasibility of the new artifacts.

In the rest of this paper, the ”Methodology” section describes the methodology followed to develop our framework. The ”Related Works” section presents the added value of our proposition compared to existing solutions. The ”Design artifacts” section explains the two artifacts designed for the DatAssistant framework. The ”Implement DatAssistant framework in a real environment” section develops the implementation of the framework on a real case. The ”Discussion” section highlights the importance of our scientific contributions and states our proposition’s limitations and future works. The ”conclusion” section ends our paper.

## Methodology

To develop our framework, we applied the Design Science Research (DSR) methodology. This methodology has been defined to create artifacts, in the field of Information Systems, that generate new knowledge for the scientific community but that are also usable for practitioners.<sup>22</sup> Peffers and Tuunanen<sup>15</sup> lists the main steps necessary to construct such an artifact that we followed. The process is graphically illustrated on Figure 1 and described below:

- (1) **Problem identification and motivation:** The importance of the research problem must be clearly established in order to justify the solution’s value.<sup>15</sup> The problem of interest in this paper is fully developed in ”Related Works” section;
- (2) **Definition of the solution’s objectives:** Objectives of the solution must be inferred from the identified problem.<sup>15</sup> The objectives of the proposed framework, and in particular its two new artifacts, are stated as research questions in the ”Related Works” section following the definition of the interest problem;
- (3) **Design and development:** The functionality and architecture of the artifact must be determined, and then the artifact created.<sup>15</sup> The ”Design artifacts” section presents the

architecture and functioning of the two artifacts of our framework and how they answer our research questions.

- (4) **Demonstration and evaluation:** The feasibility of the artifact proposed must be demonstrated.<sup>15</sup> Our framework is implemented in the "Implement DatAssistant framework in a real environment" section in order to demonstrate the feasibility of our new artifacts.

## Related Works

Several solutions have been developed to democratize data querying for non-IT experts. There are two categories of solutions having this objective.

The first category of papers treating this objective are called Visual Query Languages (VQLs). These languages are the precursors of the data access democratization. VQLs are languages that enable users to construct data queries using visual representations rather than text like in SQL.<sup>23</sup> These languages employ various visual representations such as tables, diagrams, or icons to represent concepts and relationships.<sup>24</sup> By leveraging visual representations, users can focus more on the meaning of their queries and less on the syntactical aspects of constructing a query, which can lead to enhanced reasoning and problem-solving capabilities.<sup>25,26</sup> However, this approach to query modeling may not be suitable for all query types, especially complex queries, as reported in previous studies.<sup>26</sup> Moreover, despite the existence of numerous VQLs in the literature, very few have been successfully implemented in real-world environments. And some works conclude that these systems are not solutions for non-IT experts due to their lack of practical implications.<sup>27,25</sup> The second category of papers identified in the literature corresponds to Natural Languages Interfaces for databases (NLIDBs). These NLIDBs were born in the 70's after the development of VQLs. The recent frameworks are increasingly robust and commercially available.<sup>28,29</sup> These interfaces represent a more advanced version of VQLs, since they allow people to write their data queries in natural language rather than using a formal query language (e.g. DAX, SQL, SPARQL) or using visual representations (i.e. VQLs).<sup>30,31</sup> Several methods are used in these systems such as machine learning or syntactic analysis. Regardless of the method used, the main challenges for these tools are the ambiguity of natural language and the increasing complexity of data queries.<sup>32</sup> The DQSSs described above focus mainly on the complexity that technical data query languages represent for non-IT experts. However, the literature on SSBI, and more specifically on the challenges faced by business people to use the self-service solutions, identifies other technical barriers complicating the interaction of these people with data. Lennerholt and Van Laere<sup>10,11</sup> identify technical data query languages as a primary challenge for non-IT users, but they also identify two major features of current databases that are complex for these people to understand:

- **The technical jargon:** Because they are designed and implemented by IT experts, databases often utilize highly technical jargon and terminology that may not be aligned with the vocabulary used on a day-to-day basis by business people. For instance, data labels can correspond to code names or have multiple definitions depending on the type of business person;<sup>11,13</sup>
- **The data fields overload:** In the current Big Data era, the data volume is constantly increasing.<sup>14</sup> This growth is driven by advancements in domains such as digital sensors, communication or storage.<sup>33</sup> While collecting data has become easier, identifying relevant information from that data has become increasingly complex. Some data fields may not provide any valuable insights, or may not be relevant to all people within a company.<sup>11</sup> Identifying relevant data requires specialized skills and technologically advanced tools, which come as barriers for business users.<sup>3,14</sup>

In this paper, we propose a framework to complement existing DQSSs described above, and in particular NLIDBs, as illustrated in Figure 2 and thereby treat the following two research questions:

- **RQ1:** How to bridge the semantic gap between database jargon and business jargon within NLIDB ?

- **RQ2:** How to enrich NLIDB so as to cope with the growing volume of data in databases ?

## Design artifacts

The two new artifacts proposed in our DatAssistant framework aim to answer the two research questions stated above, respectively. This relationship between artifacts and research questions is illustrated in Figure 2. These artifacts are theoretically developed in the subsections below.

- (1) **High-Level Business DWH Ontology:** Construction of an Ontology bridging the semantic gap between technical databases and business people (see "Artifact 1 - High-Level Business DWH Ontology" Section);
- (2) **Hybrid Data Recommender Engine Architecture:** Complete architecture of a Hybrid Data Recommender Engine, semantically and technically filtering data volume (see "Artifact 2 - Hybrid Data Recommender Engine" Section).

### *Artifact 1 - High-Level Business DWH Ontology*

Our RQ1 focuses on the integration of business jargon in DQSSs, and especially NLIDBs, which are currently unaware of the vocabulary used by business people. Stated differently, existing NLIDBs can interpret and translate sentences into technical queries only if people name explicitly each table and column they use. As discussed in our introduction, this working assumption is hardly applicable in real business environment, where databases have technical labeling meaningless to non-IT experts. One way to bridge this gap is to design a mapping between the business semantics of non-IT people and the databases' technical semantics. To do this, we propose a high-level abstraction of the mapping that can be done between Data Warehouses (DWHs) (the "technical" side of the SSBI system) and their Data Catalogs, i.e. business metadata (the "business" side of the SSBI system) by mobilizing ontological semantic systems.

Ontologies are formal and explicit specifications of conceptualizations.<sup>17</sup> These semantic systems represent concepts, relations, instances, and axioms in a way that is machine and human-readable.<sup>34</sup> The relevance of using these particular semantic systems in our framework is twofold. First, ontologies, via their formal and structured representations, are suitable to visualize the different data fields types and their relations. Secondly, the exploitation of ontologies within recommender systems solves several barriers encountered by traditional recommender systems,<sup>35</sup> as further developed in the "Artifact 2 - Hybrid Data Recommender Engine Architecture" Section.

To identify the different concepts needed in the Business DWH Ontology, we have created a High-Level Ontology following prescriptions from the Methontology methodology.<sup>36</sup> Figure 3 illustrates the different steps we went through:

- (1) **Specification:** The objective and scope of the ontology must be clearly defined.<sup>36</sup> In our framework, the objective of the Business DWH Ontology is, as stated before, to represent in a structured way the metadata of a DWH required for a data query with their associated business semantics. To refine this general objective, we have defined a series of Competency Questions. These questions are, in fact, used to extract the main concepts, properties, relations and axioms of the ontology.<sup>36</sup> The list of Competency Questions that we have defined in consensus with three BI experts can be found in Table 1;
- (2) **Knowledge Acquisition:** To identify the different concepts of the ontology answering the Competency Questions, we must collect a set of knowledge from different sources.<sup>36</sup> In our case, we mainly exploited two scientific references, namely Vaisman and Zimanyi<sup>37</sup> and Ehrlinger and Schrott.<sup>18</sup> The first reference<sup>37</sup> contains a set of knowledge the decomposition of a data query in terms of DWH's components. The second reference<sup>18</sup> is a systematic literature review on the new trend in companies, namely the Data Catalog. This catalog consists in centralizing a series of metadata from the company's database. Among these metadata, we are interested in the business metadata.<sup>18</sup> This information is essential for our

- system since they represent the business semantic layer that bridges the gap between the familiar language of the business user and the DWH's technical language;
- (3) **Conceptualization:** the set of knowledge collected in the second step is conceptually modeled ;<sup>36</sup>
  - (4) **Integration:** An analysis of existing and reusable ontologies having a similar goal must be done ;<sup>36</sup> To build our ontology, we reused various components of the RDF Data Cube Vocabulary Ontology defined by Cyganiak and Reynolds<sup>38</sup> and the Data Catalog Vocabulary Ontology built by Albertoni and Browning.<sup>39</sup> The RDF Data Cube Vocabulary Ontology defines the metadata of multidimensional data sets, partly identical to the components of a DWH required for a data query. The Data Catalog Vocabulary Ontology aims to facilitate interoperability between data catalogs by representing a range of information about these catalogs. We selected from this Ontology some classes and properties allowing us to answer our Competency Questions about the business semantics of data and so, complete our Ontology (e.g. keyword, description). In Table 1, the classes and relations of our Ontology coming from the two ontologies mobilized are explicitly identified by their prefix "qb" for the RDF Cube Vocabulary Ontology and the prefix "dcat" for the Data Catalog Vocabulary Ontology. The "dwh" prefixes correspond to the components we have added to satisfy the purpose of our ontology;
  - (5) **Implementation:** the fifth step is the technical implementation of the ontology. In our case, we have implemented our High-Level Ontology in the well-known tool Protégé as illustrated in the Figure 4.<sup>40</sup> In this illustration, we have clearly identified the parts of our ontology that come from the two mobilized ontologies. The components we have added to answer all our Competency Questions are shown in light blue (e.g. "Fact" Class, "hasAttribute" ObjectProperty).
  - (6) **Evaluation:** the last step in the construction of an ontology is to evaluate its correctness.<sup>36</sup> For the validation of our High-Level Ontology, we have checked if all our Competency Questions have been answered by the different components of our ontology. As we can see in the Table 1, each Competency Question is associated with one or more components of the ontology and all the Ontology's components are associated to a Competency Question. This validates the correctness and completeness of our ontology.

## ***Artifact 2 - Hybrid Data Recommender Engine Architecture***

Our RQ2 aims to adapt DQSSs to the data overload of current databases. Regardless of the wording used during the querying (we treat this with artifact 1), DQSSs expect business users to have a precise idea of the data they actually are searching for and where it is located. Because the number of tables and columns available in today's analytical databases can be overwhelming, this occurrence is less likely to happen. In practice, it is far more common to have business users with a good idea of what they search, but no idea of where to find it. This adds on top of the challenge for business users to deal with DWH's concepts and principles (e.g. fact table, dimension table, primary key). One way to cope with this NLDB's challenge is to allow the user express his/her data needs naturally without any knowledge of the underlying DWH structure and composition and then, automatically identify among the mass of data the fields best suited for such request. To achieve this, we propose a Hybrid Recommendation Engine Architecture that semantically and technically filters data fields to identify relevant ones.

Recommender System (RS) aims to identify among a mass of items, those which are the most likely to interest the user.<sup>19</sup> There are three main types of RS, namely: Collaborative Filtering RS, Content-Based RS and Knowledge-based RS. The first two types of RSs are the most traditional. They mainly use the behavior of other users (i.e. their past ratings, their similar profile to the user) to define their recommendations.<sup>41</sup> These systems are widely used but also criticized for the limitations they face such as the lack of behavioural history for a new user (i.e. the cold start problem) or the overspecialization.<sup>42</sup> Conversely, Knowledge-Based RS (KBRS) uses a knowledge base gathering a lot of information about the user and the items to identify the items that best match the users'



requirements. In this way, KBRS is known to offer more relevant recommendations.<sup>20, 43, 44</sup> To design our framework, we decided to mobilize KBRS and more particularly KBRSs based on Ontology. The Ontology-based RS implies that the knowledge base of the system is represented in ontological format, which echoes the development of our high-level ontology in previous section.<sup>45, 42</sup> Our recommendation engine uses two inputs:

- (1) **The instantiated High-Level Business DWH Ontology:** The instantiation of our High-Level Ontology serves as a knowledge base with which the recommendation engine interacts to filter data fields.
- (2) **The user's business data query processed by a DQSS:** RS requires data about the user, which in our case corresponds to his/her data query expressed in natural language. In order to decompose the natural language data query to identify the different elements of a formal query (e.g. aggregation function, measure, dimension), we propose to exploit existing DQSSs. These systems are, as explained in the "Related Works" section, well known for being widely investigated in natural language-formal language translation of data queries.

These two inputs are used by the recommendation engine, which is itself made up of two components, hence the "hybrid" name. Our engine mobilizes, in fact, the two types of recommendation engines well known in KBRS, namely: the Rules-based Recommendation Engine and the Case-based Recommendation Engine. These are described in more detail in the subsections below.

### *Rules-based Recommendation Engine*

The Rules-based Recommendation Engine is the first filter in our recommendation engine architecture. This type of recommendation engine, also called Constraints-based Recommendation Engine, uses explicit rules or constraints to generate recommendations for the user. These rules determine the mapping between the item (e.g. products, services, data) and the user. This engine is often used in situations where the user does not have enough knowledge on the items to select one.<sup>20</sup> In our framework, this approach is adapted because business people do not have sufficient IT background to understand data fields and understand the specific features of a DWH which allow, for example, joins between tables or not.<sup>12</sup> Our Rules-based Recommendation Engine is therefore composed of a set of Business Rules corresponding to the basic principles of DWH data manipulation. Thanks to these rules, an initial "technical" filter is performed on all available data fields.

Concretely, these rules depend on the type of DWH. Indeed, DWH with a Star Schema type logical schema implies a less constraining exploration of the data than a DWH with a Snowflake type logical schema that have more types of relationships between data fields.<sup>37</sup> For this reason, we do not explicitly define a set of Business Rules for our recommendation engine. These will have to be defined between the business people and the IT experts of the company. Nevertheless, for the expression of these rules, the logical format is commonly adopted, i.e. IF(antecedent) THEN(consequent).<sup>46</sup> In this type of rules, if the antecedent is validated then the consequent is executed. The translation of a DWH's principle is: IF the data need is a measure THEN considers all data fields belonging to fact tables. Furthermore, the execution of these rules within the recommendations engine involves a series of interactions with the Business DWH Ontology to obtain the available data fields. These interactions are performed in the form of SPARQL queries. SPARQL is the well-known language used to interact with ontologies encoded in the standard OWL or rdf format.<sup>34, 47</sup> As an example, the Business Rule on the user's need for measure is translated into SPARQL language as follows:

---

```

SELECT ?table ?column ?title ?description
WHERE-
  ?table rdf:type dwh:Fact.
  ?table dwh:hasMeasure ?column.
  ?column dwh:IsA ?resource.
  ?resource dcat:identifier ?identifier.

```



```
?column dcat:title ?title.  
?column dcat:description ?description.
```

---

### *Case-based Recommendation Engine*

The Case-based Recommendation Engine represents the second data filter of our recommender engine architecture. In theory, it consists in finding items in the knowledge base that are similar to those described by the user's requirements.<sup>20</sup> This engine exploits different types of clustering techniques (e.g. KNN algorithm) or similarity measures.<sup>48</sup> This RS type is used when items are well described in terms of features to compute similarity between them.<sup>21</sup>

In our framework, this approach is also suitable because we aim to identify the data field similar to the user's data need expressed in his/her business data query. This approach is complementary to the rules-based approach, since it will identify the data field in the subset identified by the rules-based engine that is semantically similar to the user's need.

Concretely, the semantic analysis performed by this second recommendation engine aims to compute a semantic distance between the data need identified by the DQSS in the data query expressed by the user and the semantic properties of the data fields present in the filtered data subset. These properties are represented in the Business DWH Ontology using the components of the Data Catalog Ontology, as illustrated in Figure 4. There are different techniques to measure semantic similarity such as structured-based measures which focus on the position of the terms, or feature-based measures which study the description given to each term.<sup>49</sup> This subject has been widely studied in the literature,<sup>50,51</sup> so we leave the choice of algorithm to the framework implementation.

The resulting complete architecture of the recommendation engine developed in our framework is shown in Figure 5.

## **Implement DatAssistant framework in a real environment**

The objective of this section is to demonstrate the feasibility of our DatAssistant framework, with a particular focus on its two new artifacts. This section serves as the fourth phase of our methodology, wherein we demonstrate the applicability and evaluate the efficacy of our artifacts. Concretely, we collaborated with the BI department of the University of Namur (UNamur) to get a real use case. Through this collaboration, we obtained access to the metadata of their DWH and their Data Catalog. The data from this use case is highly relevant to our framework, as it perfectly illustrates the problem our solution is designed to solve. In fact, as we consulted the data and discussed with the BI experts in our use case, we found that their DWH was made up of a large number of codes meaningless for business people (the university community, professors, assistants, secretaries, etc.). Many codes are also present several times in different data tables, but with distinct business meanings. This use case is a good illustration of the challenges addressed by our solution, namely the semantic gap between databases and business people, and the overloading of data fields. To implement our framework on our use case data, we followed a process illustrated in Figure 6. This process is made up of three main stages carried out in parallel, representing the implementation of the three framework's modules, as illustrated in Figure 2 in the "Related Works" section:

- (1) **Module 1 - Existing DQSS:** The implementation of the first module consists in the identification and selection of an existing DQSS, and more specifically an existing NLIDB algorithm, and its adaptation to the objective of our module, i.e. the natural language processing of the user's business query.
- (2) **Module 2 - High-Level Business DWH Ontology:** The implementation of the second module of the framework represents the instantiation of our High-Level Business DWH Ontology with the data from our use case.
- (3) **Module 3 - Hybrid Data Recommender Engine:** Implementing the recommendation

engine involves defining Business Rules adapted to the DWH schema of the use case, and selecting and applying a semantic similarity measure algorithm.

The three subsections below describe in more detail the implementation of each module of our DatAssistant framework on our real use case.

### ***Module 1 - ChatGPT as DQSS for our framework***

The first module's goal is to process the natural form of the user's business data query to identify the different sets of words representing the formal query's components, namely: aggregation function, measure, dimension(s). To achieve this, we exploit the well-known GPT3 algorithm.<sup>52</sup> This algorithm, and its exploitation in the ChatGPT tool, represents a real evolutionary opportunity for NLIDBs. This Large Language Model exploits artificial intelligence techniques (i.e. neural networks, transformers, etc.) to generate coherent natural language responses based on a given input. This model has the particularity of training on an huge amount of text data, enabling it to understand the language nuances and generate contextually appropriate responses.<sup>53</sup> In the context of NLIDB, this model makes it possible to overcome many of the limitations of existing NLIDBs in terms of natural language complexity (e.g. ambiguity, synonyms, typographical errors).<sup>54</sup> GPT3 has already been implemented within a NLIDB. This implementation has demonstrated the superiority of the algorithm for natural language understanding even when natural language expressions are underspecified or misspecified, reducing development costs.<sup>55</sup> Despite its obvious superiority, this new NLIDB does not consider the possible semantic gap between the database jargon introduced in the algorithm's prompt and the business user's jargon, as well as the large volume of data. In the implementation of our framework, we reuse GPT3 algorithm for its strengths and so, to parse the user's query and identify the key components of a formal data query. Concretely, we implemented this first module utilizing the widely recognized Python language<sup>56</sup> and mobilized the "text-davinci-003" model from OpenAI.<sup>52</sup> To align the algorithm with our framework's requirements, we configured the prompt by explicitly defining its objective and specifying the desired output format.

To illustrate the output of our module, we asked UNamur's business people to formulate three distinct business data queries with their own business words. We introduced these queries in our module. The outputs obtained are shown in Table 2.

### ***Module 2 - Instantiation of the High-Level Business DWH Ontology***

The implementation of the second module aims to map the technical jargon of the use's case database with its business semantic. To achieve it, we instantiated the High-Level Business DWH Ontology with two input's:

- **The DWH Data Definition Language file:** the DDL file of a DWH contains the DWH schema. It includes information on metadata of the DWH such as tables and columns names, primary and foreign keys and the tables' relationships.<sup>37</sup> For our study, we had access to the DDL file of the UNamur DWH;
- **The DWH Data Catalog:** this catalog contains several additional information to the DDL file, as explained before, such as some data column values but especially business terms associated to the different data and a business description.<sup>18</sup> The Data Catalog of UNamur DWH is partially represented in Table 3 (i.e. a part of the rows representing column values' business metadata).

Technically, the instantiation was done in OWL2 in the form of RDF schema.<sup>57</sup> OWL and RDF schema are standards in the Semantic Web.<sup>34</sup> We indeed exploit the Python module OWLReady2<sup>47</sup> and the Python package rdflib<sup>58</sup> which are two well-established python libraries. A subpart of the instantiated Ontology is illustrated in Figure 7.

### ***Module 3 - Implementation of the Hybrid Data Recommender Engine***

The third module's implementation aims to personalize the rules-based recommender engine to the DWH's type by defining Business Rules and implement the case-based recommender engine by selecting a Semantic Similarity Measure algorithm.

Regarding the Business Rules, we defined, in collaboration with business people and UNamur Data Scientists, a whole set of rules relevant to the DWH type of UNamur. Once defined, these rules were encoded in logical format as explained in the "Rules-based Recommendation Engine" Section. They are separated into two categories: Measure Business Rules and Dimension Business Rules. The different rules defined are presented in Table 4.

Concerning the semantic algorithm, we decided to exploit the Bert algorithm to compute the semantic similarity score.<sup>59</sup> We have chosen this well-established NLP model for its conceptual simplicity and empirical power.<sup>59</sup> No evaluations have been done to identify if this algorithm was the most appropriate, the goal of the artifact implementation is to prove its feasibility not to provide a technically commercializable implementation.

Based on these technical implementation choices, our recommendation engine processed the different data requirements (i.e. measure and dimension(s)) of the three test queries. In concrete terms, the engine first analyzed the measure requirement of each query by applying the appropriate Business Rule(s). For each rule, if the antecedent was met, the engine applied the consequent. The application of these rules required various semantic similarity calculations. Table 5 details, for each query measure, the rule applied, the results obtained by the query for the antecedent of the rule and the recommendation provided. Then, the recommendation engine processed the query dimension(s). The engine computes a semantic similarity score between the dimension and the column values available in the ontology as well as a score between the need and the columns of the dimension tables. These different scores were then compared and the data associated with the highest score was recommended (i.e. Dimension Business Rules presented in Table 4). About the outputs, Table 6 details the 10 highest scores obtained for the values and columns compared to the different needs. The final output provided by the recommender engine for each query, by applying the set of rules, is shown in the Table 7. The recommendations provided by our implemented framework have been validated by UNamur's Data Scientists. They confirmed that the data fields identified by our tool matched perfectly to the data needs expressed by the users in their respective queries. Regarding the business users, they stated that the data recommendations made would help them in the operation of their current SSBI tool.

### **Discussion**

In this paper, we propose the DatAssistant framework that complements existing DQSSs, and more specifically NLIDBs. DatAssistant further simplifies data access for non-IT experts by addressing two key challenges identified by the BI literature but ignored by the IT literature.

The theoretical contribution made by this paper is a major one for the information systems community, since it provides an architecture and a set of operational principles that will enable anyone to implement our proposal in other contexts.<sup>60</sup> Gregor and Hevner<sup>16</sup> classify the scientific contributions made by the Design Science Research methodology on three levels. They recognize frameworks, the architecture of a system as a level 2 contribution, called "nascent design theory".<sup>16</sup> In addition, thanks to the technical development of our framework on a real use case, our paper makes an equally important knowledge contribution to the scientific community. Gregor and Hevner<sup>16</sup> qualify the implementation of a framework as a level 1 contribution, called "situated implementation of artifact". The instantiation of the framework shows the framework works in a real situation and so, provides tangible evidence of the framework's validity and applicability.<sup>61</sup> Our paper has certain limitations. The first limitation of our paper is the lack of empirical validation of our framework through direct engagement with business people. Indeed, we did not collect any requirements for our framework from non-IT experts, basing ourselves on the literature and existing studies carried out with these people on broader topics. In addition, the technical

development of our framework and its outputs have been validated by various BI experts in our use case. Nevertheless, we are aware of the importance of rigorously validating our framework with several companies and users. We aim to achieve this in our next works.

The second limitation of our paper lies in the output returned by our recommendations engine. Indeed, our framework sends back to the user the technical data fields to be used for his/her query. However, the next step for this person is to write the data query and execute it in the BI tool to obtain the analyzed data. Our framework is limited to data field recommendations, as all our added value lies in the DWH business ontology and in the recommendations engine. Translation between formal SQL-like queries is a task widely addressed in the literature. However, we aim to integrate this functionality into the next version of our tool to propose a complete solution.

The third limitation of our paper relates to the natural language algorithms used in our tool. We have not, in fact, selected the models to be used, leaving the choice to the implementer of our framework. Nevertheless, it would be interesting to scan the literature to identify the most suitable algorithms for the various tasks involved.

## Conclusions

In conclusion, this paper applied the Design Science Research methodology to address the challenges of data overload and databases' technical jargon faced by business people in the existing DQSSs of the literature. Our research objective was to propose a new framework mobilizing the existing DQSS complemented by two new artifacts addressing these two challenges.

Concretely, our framework responds to the challenge of the technical jargon of the databases by exploiting the Data Catalog of the Data Warehouse within the Business Ontology that it builds. To answer the challenge of data field overload, our framework proposes the architecture of a Hybrid Data Recommender Engine mobilizing Business Rules and Semantic Systems.

Our framework has been developed following the well-known Design Science Research methodology and its feasibility has been demonstrated through its implementation in a real environment.

## Acknowledgements

I would like to thank the Walloon Region to finance my research in the scope of the ARIAC project.

## Declaration of interest statement

The authors report there are no competing interests to declare.

## References

1. Tavera Romero CA, Ortiz JH, Khalaf OI, Ríos Prado A. Business intelligence: business evolution after industry 4.0. *Sustainability*. 2021;13(18):10026. doi: 10.3390/su131810026.
2. Power DJ. *Decision support systems: concepts and resources for managers*. Westport (CT): Greenwood Publishing Group; 2002.
3. Provost F, Fawcett T. Data science and its relationship to big data and data-driven decision making. *Big data*. 2013;1(1):51–59. doi: 10.1089/big.2013.1508.
4. Negash S, Gray P. Business intelligence. in *Handbook on decision support systems* 2:175–193. Switzerland: Springer Nature 2008. doi: 10.17705/1CAIS.01315.
5. Schlesinger PA, Rahman N. Self-service business intelligence resulting in disruptive technology. *J. of Comput. Inf. Sys.*. 2016;56(1):11–21.

6. Imhoff C, White C. Self-service business intelligence: Empow. Users to Gener. Insights. 2011. TDWI Best practices report: TWDI; 2011.
7. Khurana D, Koli A, Khatter K, Singh S. Natural language processing: State of the art, current trends and challenges. *Multimed. tools and appl.*. 2023;82(3):3713–3744. doi: 10.1007/s11042-022-13428-4.
8. Bhowmick SS, Choi B. Data-driven Visual Query Interfaces for Graphs: Past, Present, and (Near) Future. Paper presented at the 2022 International Conference on Management of Data; 2022 12-17 June; Philadelphia, PA, USA.
9. Li H, Chan CY, Maier D. Query from examples: An iterative, data-driven approach to query construction. *The VLDB J.*. 2015;8(13):2158–2169. doi: 10.14778/2831360.2831369.
10. Lennerholt C, Van Laere J, Söderström E. Implementation challenges of self service business intelligence: a literature review. Paper presented at: the 51st Hawaii International Conference on System Sciences; 2018 3-6 January; Hawaii, USA.
11. Lennerholt C, Van Laere J, Söderström E. User-related challenges of self-service business intelligence. *Inf. Syst. Manag.*. 2020:1–15. doi: 10.1080/10580530.2020.1814458.
12. Alpar P, Schulz M. Self-service business intelligence. *Bus & Inf. Syst. Eng.*. 2016;58(2):151–155. doi: 10.1007/s12599-016-0424-6.
13. Smuts M, Scholtz B, Calitz A. Design guidelines for business intelligence tools for novice users. Paper presented at: the 2015 annual research conference on south african institute of computer scientists and information technologists; 2015 28-30 September; Stellenbosch, Africa.
14. Tsai CW, Lai CF, Chao HC, Vasilakos AV. Big data analytics: a survey. *J. of Big data.* 2015;2(1):1–32. doi: 10.1186/s40537-015-0030-3.
15. Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S. A design science research methodology for information systems research. *J. of manag. inf. syst.*. 2007;24(3):45–77. doi: 10.2753/MIS0742-1222240302.
16. Gregor S, Hevner AR. Positioning and presenting design science research for maximum impact. *Manag. Inf. Syst. q.*. 2013:337–355. doi: 10.25300/MISQ/2013/37.2.01.
17. Gruber TR. A translation approach to portable ontology specifications. *Knowl. Acquis.*. 1993;5(2):199–220. doi: 10.1006/knac.1993.1008.
18. Ehrlinger L, Schrott J, Melichar M, Kirchmayr N, Wöß W. Data catalogs: a systematic literature review and guidelines to implementation. Paper presented at the Database and Expert Systems Applications-DEXA 2021 Workshops: BIODDD, IWCFCS, MLKgraphs, AI-CARES, ProTime, AISys 2021; 2021 27-30 September; Virtual Event.
19. Ibrahim ME, Yang Y, Ndzi DL, Yang G, Al-Maliki M. Ontology-based personalized course recommendation framework. *IEEE Access.* 2018;7:5180–5199. doi: 10.1109/ACCESS.2018.2889635.
20. Ameen A. Knowledge based recommendation system in semantic web-a survey. *Int. J. of Comput. Appl.*. 2019;182(43):20–25.
21. Smyth B. Case-based recommendation. in *The adaptive web*:342–376Springer 2007. doi: 10.1007/978-3-540-72079-9\_11.
22. Hevner AR, March ST, Park J, Ram S. Design science in information systems research. *Manag. Inf. Syst. q.*. 2008;28(1):6. doi: 10.2307/25148625.
23. Bhowmick SS, Choi B, Dyreson C. Data-driven visual graph query interface construction and maintenance: challenges and opportunities. *The VLDB J.*. 2016;9(12):984–992. doi: 10.14778/2994509.2994517.
24. Catarci T, Mecella M, Kimani S, Santucci G. Visual Query Interfaces. *The Wiley Handb. of Hum. Comput. Interact.*. 2018;2:561–577. doi: 10.4018/978-1-60566-314-2.ch002.
25. Silva E, Fidalgo R, Ferro M, Franco N. Visual query languages to design complex queries: a systematic literature review. *Softw. and Syst. Modeling.* 2022:1–33. doi: 10.1007/s10270-022-01071-4.
26. Silva E, Franco N, Ferro M, Fidalgo R. Mental workload impact of a visual language on understanding sql queries. Paper presented at the Anais do XXX Brazilian Symposium on Informatics in Education 2019; 2019 November 11-14; Brasilia.
27. Lloret-Gazo J. A survey on visual query systems in the web era. Paper presented at the 27th

- International Conference on Database and Expert Systems Applications; 2016 September 5-8; Porto, Portugal.
28. Bukhari SAC, Dar HS, Lali MI, Keshtkar F, Malik KM, Kadry S. Frameworks for querying databases using natural language: A literature review–NLP-to-DB querying frameworks. *Int. J. of Data Wareh. and Mining (IJDWM)*. 2021;17(2):21–38. doi: 10.4018/IJDWM.2021040102.
  29. Pereira A, Almeida JR, Lopes RP, Oliveira JL. Querying semantic catalogues of biomedical databases. *J. of Biomed. Inf.*. 2023;137:104272. doi: 10.1016/j.jbi.2022.104272.
  30. Francia M, Gallinucci E, Golfarelli M. COOL: A framework for conversational OLAP. *Inf. Sys.*. 2022;104:101752. doi: 10.1016/j.is.2021.101752.
  31. Androutsopoulos I, Ritchie GD, Thanisch P. Natural language interfaces to databases—an introduction. *Nat. Lang. Eng.*. 1995;1(1):29–81. doi: 10.1017/S135132490000005X.
  32. Özcan F, Quamar A, Sen J, Lei C, Efthymiou V. State of the art and open challenges in natural language interfaces to data. in *Paper presented at: the 2020 ACM SIGMOD international conference on management of data*.
  33. Emani CK, Cullot N, Nicolle C. Understandable big data: a survey. *Comput. sci. rev.*. 2015;17:70–81. doi: 10.1016/j.cosrev.2015.05.002.
  34. Antunes AL, Cardoso E, Barateiro J. Incorporation of ontologies in data warehouse/business intelligence systems-a systematic literature review. *Int. J. of Inf. Manag. Data Insights*. 2022;2(2):100131.
  35. Burke R. Knowledge-based recommender systems. *Encycl. of libr. and Inf. Syst.*. 2000;69(Supplement 32):175–186.
  36. Fernández-López M, Gómez-Pérez A, Juristo N. Methontology: from ontological art towards ontological engineering. Paper presented at the 1997 AAAI Spring Symposium; 1997 March 24-25; Palo Alto, California.
  37. Vaisman A, Zimányi E. Data warehouse systems. *Data-Centric Syst. and Applications*. 2014.
  38. Cyganiak R. The RDF Data Cube Vocabulary - W3C Recommendation 16 January 2014. <https://www.w3.org/TR/vocab-data-cube/>. Accessed 2023 May 31.
  39. Albertoni R, Browning D, Cox S, Beltran AG, Perego A, Winstanley P. Data Catalog Vocabulary (DCAT) – W3C Working Draft 07 March 2023. <https://www.w3.org/TR/vocab-dcat-3/>. Accessed 2023 May 31.
  40. Noy NF, Crubézy M, Fergerson RW, et al. Protégé-2000: an open-source ontology-development and knowledge-acquisition environment.. Paper presented at the 2003 AMIA annual symposium; 2003 November 12; Washington DC; USA.
  41. Melville P, Sindhvani V. Recommender systems.. *Encyclopedia of mach. learn.*. 2010;1:829–838. doi: 10.1007/978-1-4899-7687-1\_964.
  42. Tarus JK, Niu Z, Mustafa G. Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artif. Int. Rev.*. 2018;50(1):21–48. doi: 10.1007/s10462-017-9539-5.
  43. Yang S, Cai X. Bilateral knowledge graph enhanced online course recommendation. *Inf. Sys.*. 2022;107:102000.
  44. Pinon S, Burnay C, Linden I. Opportunities of semantic recommendation systems for self-service business intelligence. Paper presented at the EWG-DSS 2022 international conference on decision support system technology; 2022 23-25 May; Thessaloniki, Greece.
  45. Peis E, Castillo JM, Delgado-López JA. Semantic recommender systems. analysis of the state of the topic. *Hipertext. net*. 2008;6(2008):1–5.
  46. Chen RC, Huang YH, Bau CT, Chen SM. A recommendation system based on domain ontology and SWRL for anti-diabetic drugs selection. *Expert Syst. with Appl.*. 2012;39(4):3995–4006. doi: 10.1016/j.eswa.2011.09.061.
  47. Lamy JB. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. *Artif. Intell. in Med.*. 2017;80:11–28. doi: 10.1016/j.artmed.2017.07.002.
  48. Bouraga S, Jureta I, Faulkner S, Herzsens C. Knowledge-based recommendation systems: a survey. *Int. J. of Int. Inf. Technol.*. 2014;10(2):1–19. doi: 10.4018/ijit.2014040101.
  49. Slimani T. Description and evaluation of semantic similarity measures approaches. *Int. J. of*

- Comput. Appl.*. 2013;80(10):25–33. doi: 10.5120/13897-1851.
50. Elavarasi SA, Akilandeswari J, Menaga K. A survey on semantic similarity measure. *Int. J. of Res. in Advent Technol.*. 2014;2(3):389–398. doi: 10.1109/ICAC347590.2019.9036843.
  51. Gomaa WH, Fahmy AA, others . A survey of text similarity approaches. *Int. j. of Comput. Appl.*. 2013;68(13):13–18. doi: 10.5120/11638-7118.
  52. OpenAI, GPT3 algorithm. <https://platform.openai.com/docs/models/gpt-3-5>. Accessed 2023 June 14.
  53. Kalla D, Smith N. Study and Analysis of Chat GPT and its Impact on Different Fields of Study. *Int. J. of Innovative Sci. and Res. Technol.*. 2023;8(3).
  54. Katsogiannis-Meimarakis G, Tsapelas C, Koutrika G. Natural Language Data Interfaces: From Keyword Search to ChatGPT, are we there yet?. Paper presented at the 6th International Workshop on Big Data Visual Exploration and Analytics co-located with EDBT/ICDT 2023 Joint Conference; 2023 March 38-31; Ioannina, Greece.
  55. Maddigan P, Susnjak T. Chat2vis: Generating data visualisations via natural language using chatgpt, codex and gpt-3 large language models. *IEEE Access*. 2023. doi: 10.48550/arXiv.2302.02094.
  56. Python language. <https://www.python.org/>. Accessed 2023 June 14.
  57. Carroll J, Herman I, Patel-Schneider PF. OWL 2 web ontology language RDF-based semantics. *W3C Recomm. (October 27, 2009)*. 2015.
  58. rdflib documentation. <https://rdflib.readthedocs.io/en/stable/>. Accessed 2022 Nov 10.
  59. Devlin J, Chang MW, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 2018. doi: 10.48550/arXiv.1810.04805.
  60. Purao S. Design research in the technology of information systems: Truth or dare. *GSU Department of CIS Working Paper*. 2002;34:45–77.
  61. Venable J, Pries-Heje J, Baskerville R. FEDS: a framework for evaluation in design science research. *Eur. j. of inf. syst.*. 2016;25:77–89. doi: 10.1057/ejis.2014.36.



## Tables

Table 1.: Competency Questions of the High-Level Business DWH Ontology.

Number	Competency Question	Ontology components
CQ1	What are the different facts available ?	"qb:Dataset" Class, "qb:component" ObjectProperty, "qb:Component" Class, "dwh:Fact" Class
CQ2	What are the different measures of each fact ?	"qb:Measure" Class, "qb:hasMeasure" ObjectProperty
CQ3	Under which dimensions can be viewed facts ?	"dwh:Fact" Class, "dwh:isAssociatedToDimension" ObjectProperty, "qb:Dimension" Class, "qb:SliceKey" Class, "qb:sliceKey" ObjectProperty
CQ4	What are the different attributes of each dimension ?	"qb:Dimension" Class, "dwh:hasAttribute" ObjectProperty, "qb:Attribute" Class
CQ5	What are the different values of any dimension's attributes ?	"qb:Attribute" Class, "dwh:hasAttributeValue" ObjectProperty, "dwh:AttributeValue" Class
CQ6	What are the primary keys of fact and dimension tables ?	"qb:SliceKey" Class, "qb:sliceKey" ObjectProperty
CQ7	What's the business semantic associated to data fields ?	"dcat:Catalog" Class, "dcat:resource" ObjectProperty, "dcat:Resource" Class, "dcat:language" DataProperty, "dcat:identifier" DataProperty, "dcat:description" DataProperty, "dcat:keyword" DataProperty, "dcat:title" DataProperty

Table 2.: GPT3 as business data query processor - Outputs

User's business data query	Aggregation function	Measure	Dimension
Give me the average score obtained from the diplomas granted for men and women	average	score obtained from the diplomas granted	for men and women
Count the number of diplomas by region of residence	count	the number of diplomas	region of residence
What's the registrations number in bachelor for students with specific needs	count	the registrations number	bachelor, students with specific needs

Table 3.: Part of the Use Case Data Catalog.

DWH's column name	Column's values	Value's Business Name	Value's description
Distinction_Distinction	S	Satisfaction	The student has passed with an average score between 10 and 13 on 20
Distinction_Distinction	D	Distinction	The student has passed with an average score between 14 and 15 on 20
Distinction_Distinction	GD	Great Distinction	The student has passed with an average score between 16 and 17 on 20
...	...	...	...
RegistrationType_Code	P	Main regular registration	This is the "normal" situation for a student enrolled at University
RegistrationType_Code	S	Secondary registration	When a student is registered at University in more than one program, registrations in addition to his main registration are considered secondary
...	...	...	...

Table 4.: Recommendation Business Rules.

Rules	Antecedent	Consequent
<b>Measure Business Rules</b>		
MR1	aggregation function != count	recommend the DataWithHighestScore(measure, columns of fact tables)
MR2	aggregation function == count AND DataWithHighestScore(measure, tables) == table of fact type	recommend the Primary Key of the table with the highest score
MR3	aggregationfunction == count AND DataWithHighestScore(measure, tables) != table of fact type AND Primary Key of the table with the highest score IN a single fact table	recommend the Foreign Key representing the Primary Key of the table with the highest score present in the fact table
MR4	aggregation function == count AND DataWithHighestScore(measure, tables) != table of fact type AND Primary Key of the table with the highest score IN a several fact table	recommend the Foreign Key representing the Primary Key of the table with the highest score present in the DataWithHighestScore(measure, fact tables)
<b>Dimension Business Rules</b>		
DR1	HighestScore(dimension, columns' values of dimension tables) > HighestScore(dimension, columns of dimension tables)	recommend DataWithHighestScore(dimension, columns' values of dimension tables)
DR2	HighestScore(dimension, columns' values of dimension tables) < HighestScore(dimension, columns of dimension tables)	recommend DataWithHighestScore(dimension, columns of dimension tables)

Table 5.: Rules applied on the measure of queries.

Query	Measure	Rule executed	Antecedent results	Recommended data field
1	score obtained from the diplomas granted	MR1	aggregation function != "count" (i.e. average)	FactDiploma_AverageScore
2	the number of diplomas	MR2	aggregation function == "count" AND the table with the Highest Similarity Score is a fact table (i.e. FactDiploma)	FactDiploma_ID (Primary Key of the FactDiploma table)
3	the registrations number	MR3	aggregation function == "count" AND the table with the Highest Similarity Score is a fact table (i.e. FactRegistration)	FactRegistration_ID (Primary Key of the FactRegistration table)

Table 6.: Scores of 10 best similarity matches (values & columns) with dimension(s) of queries.

Query	Dimension	Value	Value 's score	Column	Column's score
1	for men and women	Female	0.506236	Gender	0.514335
		Cohabitant	0.497335	Student_SHORT_DomicileRegion	0.45925
		Married	0.493692	Study_CycleCategory	0.44193
		Other	0.467627	Student_CivilStatus	0.436535
		Male	0.457291	Distinction_Distinction	0.422892
2	region of residence	abroad	0.721447	Student_SHORT_DomicileRegion	0.7645506
		aggregat.	0.718099	Study_CycleCategory	0.719492
		yes	0.694434	Study_Faculty	0.697273
		certif.	0.677611	Student_LastName	0.657695
		with	0.652237	Study_Certificate	0.638495
3	bachelor	BAC	0.896624	Student_CivilStatus	0.540314
		single	0.715391	Student_LastName	0.507161
		1e	0.714997	Student_FirstName	0.49759
		widower	0.632437	Study_Certificate	0.459763
		male	0.553546	Student_SHORT_DomicileRegion	0.437172
	students with specific needs	with	0.896445	Student_SpecificNeeds	0.873501
		other	0.650312	Student_Gender	0.629521
		education and technology	0.64995	Study_Faculty	0.62181
		technical transition	0.621376	Study_CycleCategory	0.620102
		cohabitant	0.611897	Student_SHORT_DomicileRegion	0.603606

Table 7.: Recommender Engine Outputs

Query	Fact fields	Dimension fields
Give me the average score obtained from the diplomas granted for men and women	The query requires the following column: FactDiploma_AverageScore from the table: FactDiploma	The query includes a drill-down with the column: Student_Gender of the table: DimStudent
Count the number of diplomas by region of residence	The query requires the following column: FactDiploma_ID from the table: FactDiploma	The query includes a drill-down with the column: Student_SHORT_DomicileRegion of the table: DimStudent
What's the registrations number in bachelor for students with specific needs	The query requires the following column: FactRegistration_ID from the table: FactRegistration	<p>The query includes a slice by the value: BAC present in the column: Study_CycleCategory of the table: DimStudy</p> <p>The query includes a slice by the value: With present in the column: Student_SpecificNeeds of the table: DimStudent</p>



## Figures

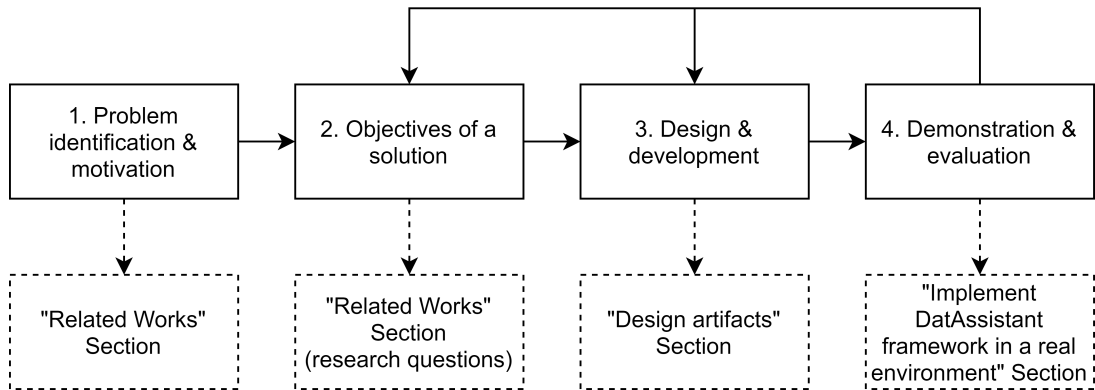


Figure 1.

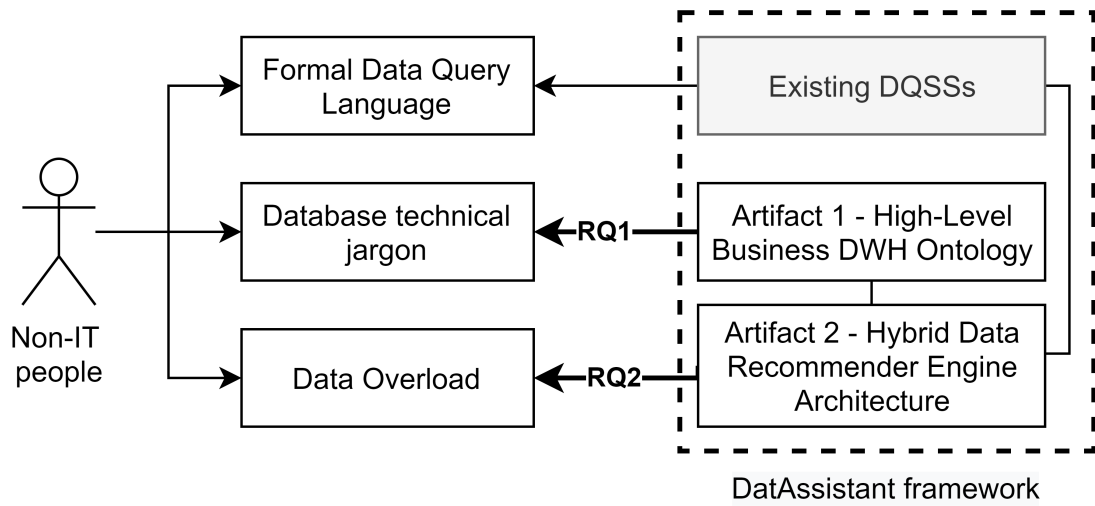


Figure 2.

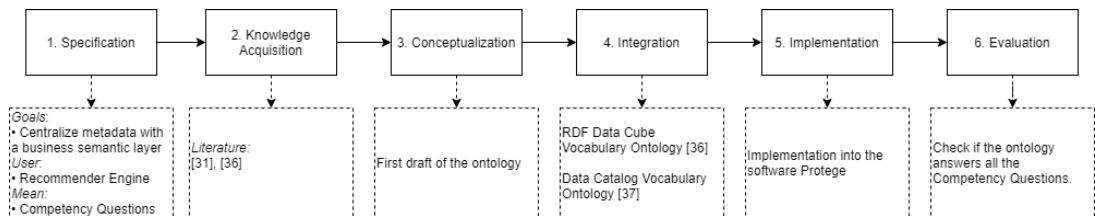
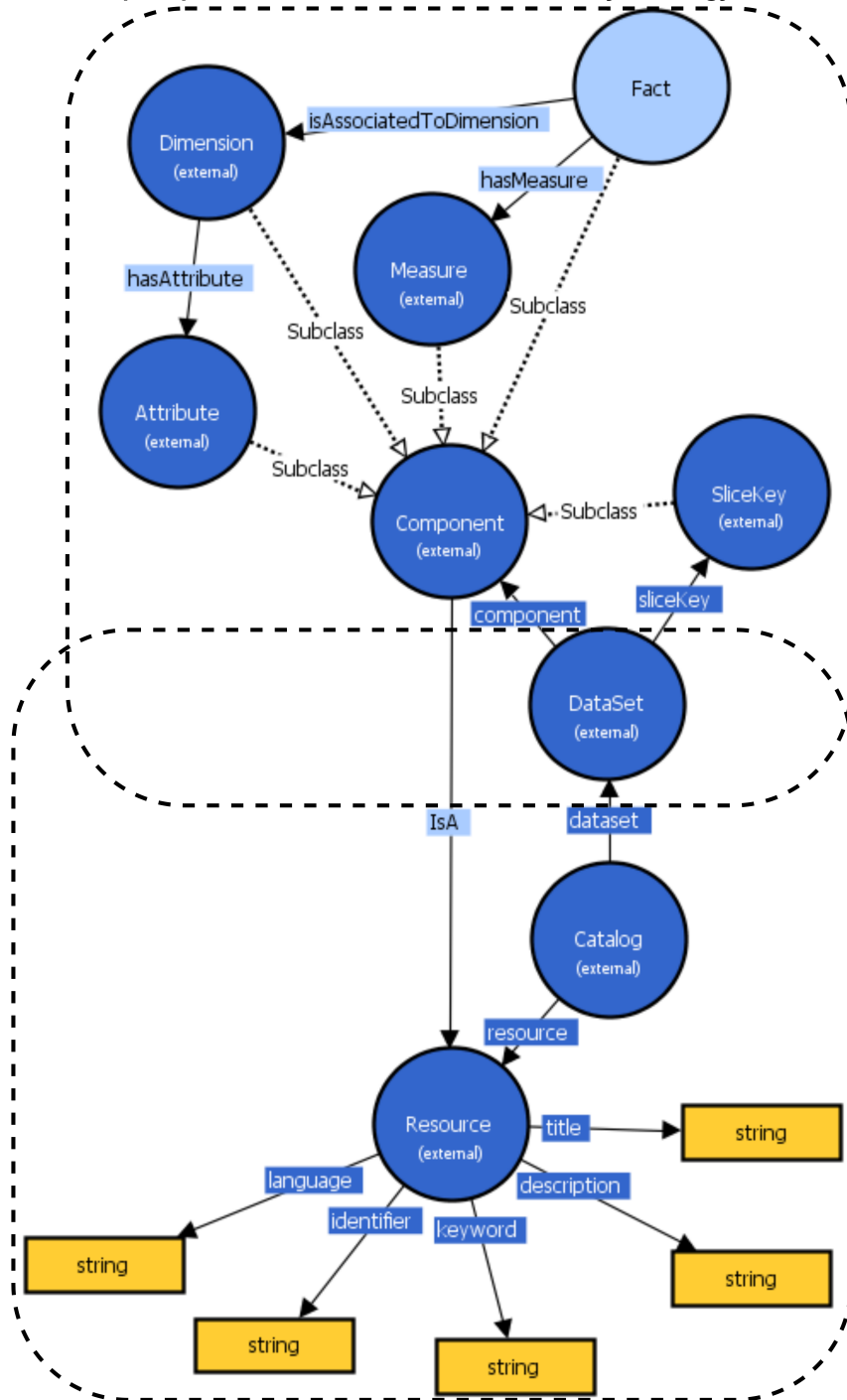


Figure 3.

Adapted part of the RDF Data Cube Vocabulary Ontology



Part of the Data Catalog Vocabulary Ontology

Figure 4.

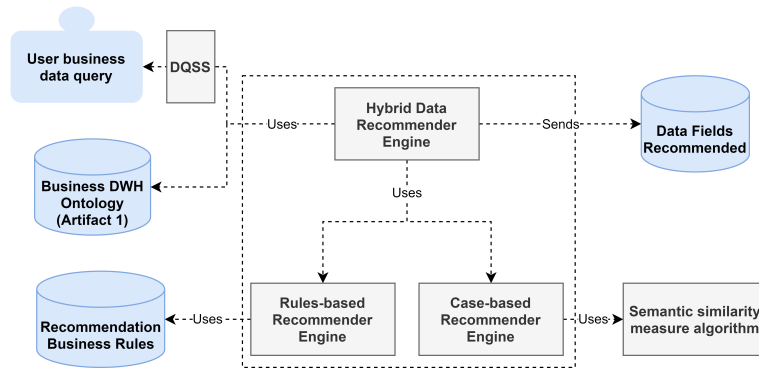


Figure 5.

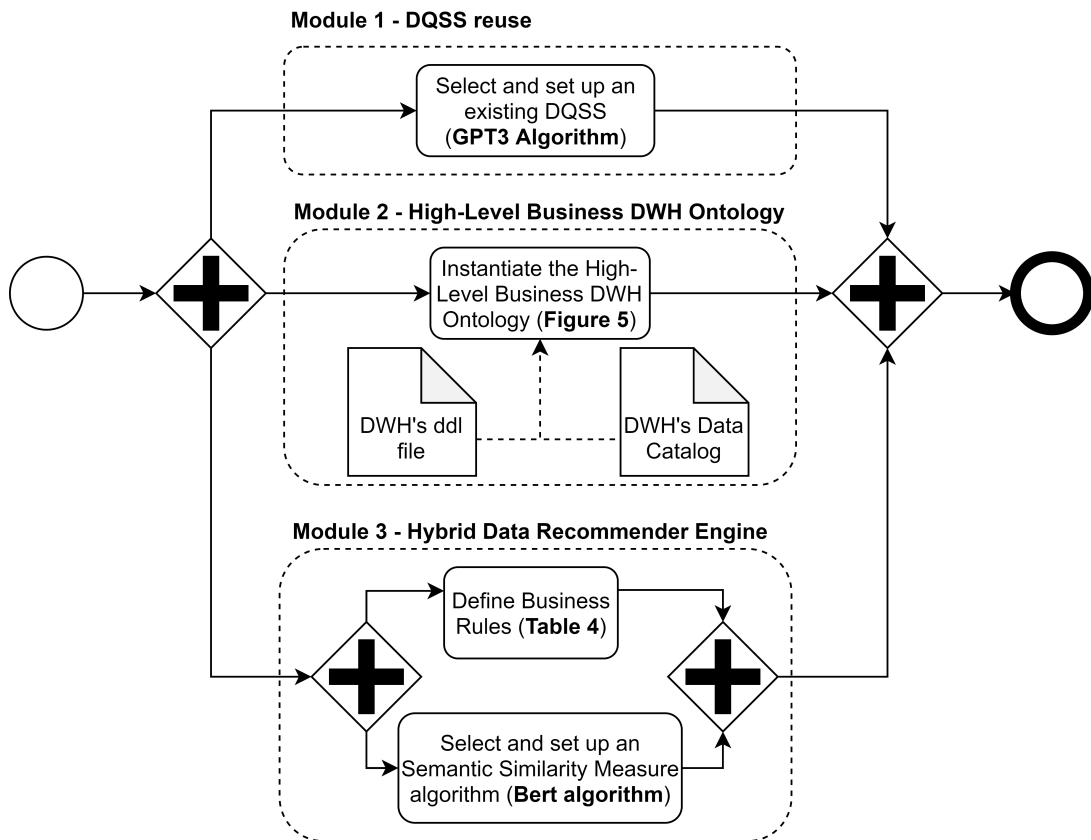


Figure 6.

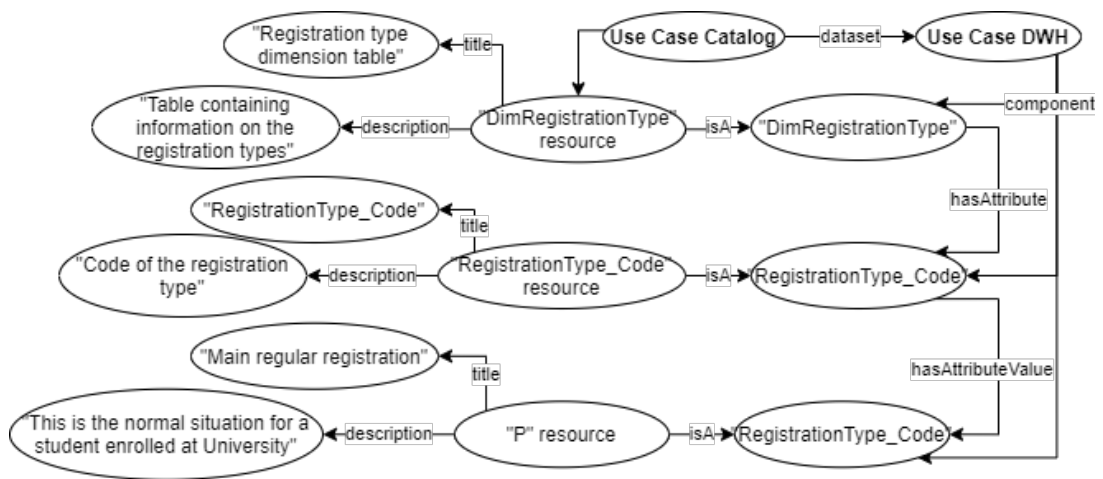


Figure 7.

## List of Figure Captions

**Figure 1:** DSR methodology applied.

**Figure 2:** Framework's architecture and literature positioning.

**Figure 3:** High-Level Business DWH Ontology: Methontology

**Figure 4:** High-level Business DWH Ontology.

**Figure 5:** Hybrid Data Recommender Engine Architecture.

**Figure 6:** Framework's implementation: Process.

**Figure 7:** Part of the instantiated High-Level Business DWH Ontology.