Streamlining Self-Service BI with Data Virtualization and a Business Directory

A Technical Whitepaper

Rick F. van der Lans
Independent Business Intelligence Analyst
R20/Consultancy

March 2015

Sponsored by

Table of Contents

1 Introduction

Self-service business intelligence tools (SSBI) are a valuable enrichment of the palette of BI tools already available. So may users already benefit from this intuitive and easy-to-use technology. These tools allow users to develop and maintain their own reports through a *do-it-yourself* approach. Several reasons have made self-service BI so popular; including improved time-to-market, improved flexibility, instant dealing with emergencies, and bypassing the business intelligence backlog.

However, practice shows that SSBI projects start hopeful, but eventually become bogged down. A study has shown that 64% of the organizations struggle with self-service BI. The problem is related to the technical complexity of *data access* and to the *metadata specifications* that have to be entered to make the reports work.

Problems with self-service BI are related to complexity of data access and metadata specifications.

With respect to data access, data is available in abundance in today's organizations. It's everywhere. When users have access to a few production systems, a large data warehouse, several data marts, some external data sources, and quite a number of private spreadsheets, the number of tables and columns they have access to can be staggering and overwhelming. There can be hundreds of them. It's not unimaginable that users can't see the forest for the trees anymore. The effect is that users lose quite some time to find the right data, or they can't find the right data at all, they use the wrong data, they don't understand the data, and sometimes they don't even know that certain types of data are available.

To develop their own reports, users have to enter metadata specifications. These describe what the data means, how data can be accessed, what the security aspects are, how data from different data sources can be combined, how data in particular columns must be standardized, and so on. Metadata specifications come in all forms and shapes, and are indispensable for data access. They are the key to unlock the door to the right data.

The problems related to the metadata specifications can be classified in four groups: proliferation of specifications, inconsistent report results, specifications that are too technical, and costly maintenance of specifications.

This whitepaper describes how self-service BI can be promoted to *managed self-service BI* by combining it with the C*isco Information Server* (CIS) and the *Cisco Business Directory* (CBD). With managed SSBI, most of the common problems can be avoided, without users losing flexibility, high productivity, self-serviceness, and independency. The whitepaper describes how to streamline a self-service BI environment in which metadata specifications are implemented in CIS and descriptive information is maintained in CBD to improve search capabilities and to explain what data means.

Promoting self-service BI to managed *self-service BI.*

2 Self-Service Business Intelligence

Self-Service BI Promotes a Do-It-Yourself Approach – With the arrival of tools such as Qlikview, Spotfire, and Tableau, *self-service business intelligence* (SSBI) has become an indispensable form of reporting and analytics in the wide range of tools already available. TechTarget[1] defines SSBI as follows: "SSBI is an approach to data analytics that enables business users to access and work with corporate information without the IT department's involvement …" SSBI tools allow users to develop and maintain their own reports through a *do-it-yourself* approach. Because of their graphical, intuitive, and easy-to-use interface, there is no need for business users to involve IT specialists.

The Popularity of Self-Service BI – Several reasons have made self-service BI so popular:

- **Improved time-to-solution:** By allowing users to create their own reports, these reports may be available in minutes or hours. With IT involvement, it may take weeks. Reasons why it could take so long are, first, the possible unavailability of IT staff resources, and second, the time it takes for IT specialists to familiarize themselves with the information needs, which isn't an issue for the users. In other words, users can develop the reports more quickly.

- **Improved flexibility:** With SSBI users are not only able to create their own reports, they can also change them easily. This higher level of flexibility allows users to adapt their reports on the spot when information needs change due to business events.

- **Instantly dealing with urgencies:** On every level of management, from strategic to operational, urgent matters may suddenly arise. For example, on the strategic level it can be a hostile takeover by a competing organization, and on the operational level it can be a breakdown of a critical factory machine. Such matters must be solved instantly and requires access to all kinds of data as quickly as possible. There is no time to involve IT. With SSBI urgencies can be handled instantly.

- **Bypassing the BI backlog:** IT departments allocate their staff resources based on availability, priorities, and budget. Therefore, when a report has to be developed for a user, it may end up at the bottom of the *business intelligence backlog*. By allowing users to develop their own reports and to do their own analysis, they bypass the BI backlog. SSBI is like having an express pass at Universal Studios that allows you to bypass the lines.

Practical Challenges of Self-Service BI Tools – SSBI tools are popular and feature-rich. The value of self-service BI to business users has been heralded over and over again. But not everything is entirely rosy. Initially, all these tools look easy to use, but a study by Wayne Eckerson[2] shows some drawbacks. To quote the report: 64% of the organizations struggle with self-service BI, giving their self-service BI initiatives a grade of "average" or lower, with 29% rating self-service BI "fair" or "poor." Clearly, deploying self-service BI tools is not without problems.

64% of organizations struggle with self-service BI.

[1] M. Rouse, *Self-Service Business Intelligence (BI)*; see http://searchbusinessanalytics.techtarget.com/definition/self-service-business-intelligence-BI

[2] W. Eckerson, *The Promise of Self-Service BI*, April 2013, see http://insideanalysis.com/2013/04/the-promise-of-self-service-bi

3 Self-Service BI Needs Metadata Specifications

Data is available in abundance.

Self-Service BI Needs Data – Self-Service BI tools need data. Without data, an SSBI tool is like a chef without ingredients. In today's organizations, data is available in abundance. It's everywhere. Data is stored in old and new production systems, data warehouses, data marts, spreadsheets, piles of digital documents, data is available on social media networks and on open data sources, and the list goes on. It's great that so much data is available and that users with an SSBI tool can access all that data for reporting and analysis. More data enriches the potential analytical and reporting capabilities.

Self-Service Data Access – With so much data available, the question is where do users start? If they have a certain business problem to solve, which data sources contain the answer? Which ones should be analyzed? And when they know which data source to access, what exactly do all these tables mean? If a table is called X1XTAD, what type of data does it store? If a spreadsheet contains a column with production numbers, what exactly does that mean? It's like the London or Paris underground. Plenty of trains going into every possible direction, lots of street names, but which one should you pick? You need a map that explains it all.

It's the users' responsibility to bring together their SSBI tool and the data. They have to organize *data access* themselves.

Self-service BI implies self-service data access.

In a traditional environment, in which reports and dashboards are developed by IT, IT is responsible for handling data access and tackling all these questions. They have to study what users want to analyze, which data elements are required to support that analysis, which data sources and tables contain that data, what all the data elements really mean, and only then can they deliver the data in a form that makes sense to the users. In fact, this activity is relatively easy for IT as they developed most of these data sources themselves, so they already know the answers to most of these questions. With SSBI, the user is on his own. He has to understand all aspects of data access; self-service BI implies self-service data access.

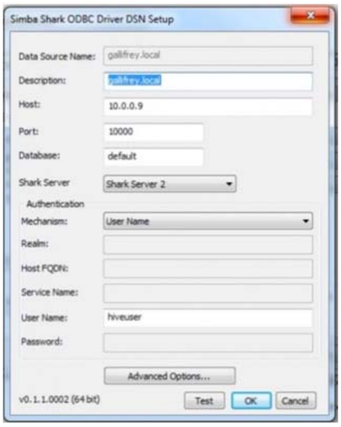
Metadata specifications are the key to unlock the door to the right data.

Self-Service BI Needs Metadata Specifications – To develop their own reports on the right data, users have to enter *metadata specifications*. Metadata specifications describe what the data means, how data can be accessed, what the security aspects are, how data from different data sources can be combined, how data in particular columns must be standardized, and so on. Metadata specifications come in all forms and shapes, and indispensable for data access. They are the key to unlock the door to the right data.

As indicated, when using SSBI tools, business users have to enter most of these metadata specifications themselves. For example, to access a data source, they have to enter the correct connection specifications, such as users-ids and password, transformation specifications, integration specifications, and so on. Finally, they have to enter visualization specifications to indicate how the results must be presented.

Some of these specifications are entered by filling in fields; see Figure 1. Here technical data has to be entered to make it possible to connect the user to a specific ODBC driver to extract data from an Apache

Hive server. It includes network specifications, such as host and ports, and security aspects, such as user name and password. Especially the former group can be difficult for users.



Simba Shark ODBC Driver DSN Setup

Data Source Name: gallifrey.local
Description: gallifrey.local
Host: 10.0.0.9
Port: 10000
Database: default
Shark Server: Shark Server 2
Authentication
Mechanism: User Name
Realm:
Host FQDN:
Service Name:
User Name: hiveuser
Password:

Figure 1 *Connection specifications are usually entered by filling in the fields.*

Some of the specifications require coding using some *graphical flow language*; see Figure 2. In this diagram, data from two data sources, called Customer Data and Transactions, is merged. Each icon requires detailed specifications. For example, the purple icon indicates the join of the two data sources. It's the user who develops this graphical program and who defines how these two data sources have to be integrated.

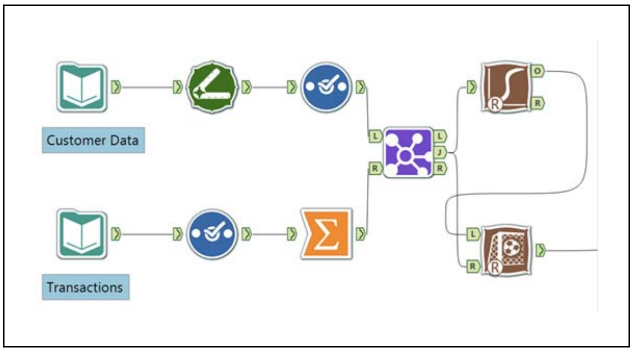
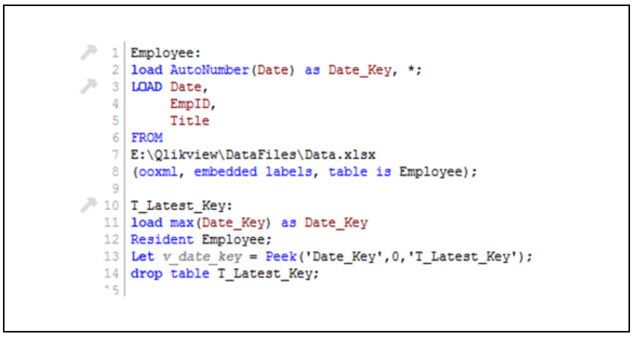


Figure 2 *An example of a graphical flow language to enter specifications to integrate data sources.*

Occasionally, specifications are in the form of *scripts*. As an example, Figure 3 contains an example of a short piece of scripting code. Here the user has to understand formal programming languages. For several SSBI tools, if the specifications become complex, neither graphical nor fill-in-the-field interface is sufficient anymore, and the users have to fall back to script programming.



```
Employee:
load AutoNumber(Date) as Date_Key, *;
LOAD Date,
     EmpID,
     Title
FROM
E:\Qlikview\DataFiles\Data.xlsx
(ooxml, embedded labels, table is Employee);

T_Latest_Key:
load max(Date_Key) as Date_Key
Resident Employee;
Let v_date_key = Peek('Date_Key',0,'T_Latest_Key');
drop table T_Latest_Key;
```

Figure 3 *Particular specifications must be entered by programming a scripting code.*

Types of Metadata Specifications – Regardless of the SSBI tool, when users develop their own reports, they must enter the following types of metadata specifications to get their report going:

- **Connection specifications:** These specifications deal with how to connect to data sources. Most of these specifications can be very technical and quite complex. They commonly include network-related and security-related details.

- **Transformation specifications:** Data transformation specifications transform data structures and data values. For example, the data structure of a data source may be in the form of a normalized data structure and it must be transformed to a star schema to simplify reporting and analysis. Data values may have to be transformed to make them conform to the company standards or to adapt them to the users' local situation. For example, euro amounts have to be transformed to US Dollars.

- **Integration specifications:** In many reports, data from multiple data sources must be integrated. Integration specifications indicate how the tables and files are combined. The complexity of integration can range from a straightforward join specified in the SQL language up to a complex solution that involves a *master data management* system (MDM). The latter is needed when key values in different systems have not be designed in cooperation, because, for example, two customer tables from two different companies are joined. Discovering the same customer in the two files requires MDM functionality.

- **Cleansing specifications:** Cleansing specifications indicate how incorrect values must be transformed to correct ones. They differ from transformation specifications because they transform correct values to other correct values. An example of a typical cleansing operation involves changing an incorrect and non-existing address to the correct one. Not all cleansing specifications are deterministic, while transformation specifications always are.

- **Descriptive specifications:** The table and column names of data sources don't always have the most intuitive names. For example, in SAP customer data can be found in a table called KNA1, and material plant data can be found in the tables MARC, MAPR, MBEW, MDKP, and STXH. Based on their names, it's not obvious to users what type of data these tables contain. And even if the names of tables and columns are plain English words, such as price and address, it's still not 100% clear what they mean exactly. Is the price including or excluding VAT? Does address mean delivery address or home address? Definitions, descriptions, synonyms, and personal notes are required to describe what a table or columns means. Only then can users analyze the data correctly and trust the results.

- **Visualization specifications:** These specifications deal with how data is presented on the screen. These visualization specifications deal with color usage, graph types, dashboard icons, page layouts, and so on. These specifications also indicate how certain results should be presented on different devices, such as desktop screens, tablets, and smartphones.

- **Collaboration specifications:** Many reports and forms of analytics are not developed in isolation by one user. In many situations, users work together and therefore need collaboration features in the tools. So, collaboration specifications must be defined as well.

- **Security specifications:** Security specifications deal with authentication and authorization aspects. They include login data, data security aspects, user-ids, and passwords.

4 Challenges of Metadata Specifications

Most of the problems that organizations have with SSBI tools are related to the metadata specifications. This section discusses four of the more dominant problems.

With SSBI there is no incentive to develop reusable specifications.

Proliferation of Specifications – Most of the specifications entered by users in their SSBI tool are private. They are stored by the tool for that particular user; see Figure 4. The consequence is that each user has to invent its own solutions and define its own specifications. For example, if two users, independently of each other, integrate the same two data sources, both have to implement a solution by entering a set of integration specifications. If the data needs to be cleansed, they both have to enter identical transformation specifications. In other words, the wheel is reinvented over and over again. Note that it's not that these specifications can't be made public with the SSBI tools, because with some it can, it's just that it's not in the user's best interest to share specifications. If users have to consider how colleagues can reuse their specifications, it will slow them down. So, there is no incentive to develop reusable specifications. The consequence is a *proliferation* of specifications.

SSBI leads to a proliferation of specifications.

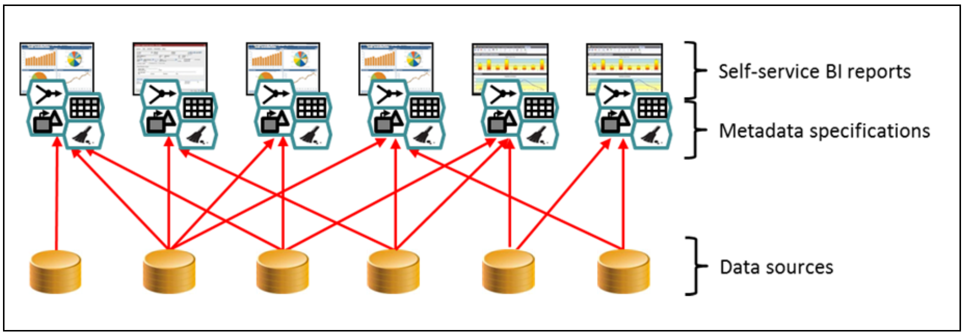


Figure 4 *Self-service BI tools can lead to proliferation of metadata specifications.*

Proliferation of specifications also arises when users use different SSBI tools. Because different tools don't share metadata specifications, similar specifications must be entered in all the tools. And in each tool a different language to define the specifications is used.

Inconsistent Report Results – If two users are developing the same solution (with or without the same tool), there is no guarantee that these solutions always return the same results. If every user enters his own specifications, how can we guarantee that the results are consistent? In an environment in which users don't share specifications, reports can return inconsistent results.

Not sharing specific-ations can return inconsistent results.

Specifications Too Technical – In an ideal situation, users focus on analysis, thus they should spend most of their time on visualization specifications, not on the more technical specifications, such as connection, integration, transformation, and security specifications. Else, they play the role of professional IT developer. The more time business users having to spend on issues not directly advancing analysis, the less likely they find the answer they're looking for.

Costly Maintenance of Specifications – If the data sources being accessed change, IT must inform all the users that they probably have to change some of their report definitions. But how does the IT department know which reports are impacted? And if IT is able to reach all the users, then all the users have to change their own metadata specifications accordingly. Especially in a situation where each user develops his own metadata specifications, maintenance is costly and complex to organize. Guaranteeing that all the proper changes are made throughout all the reports is impossible.

Maintenance is costly and complex to organize.

5 How to Streamline Self-Service BI?

The problems described in the previous sections can hamper and obstruct a successful acceptance of SSBI in an organization, they can make SSBI much more expensive than expected, they can reduce the return-of-investment of SSBI, and they can eventually lead to a complete failure of the SSBI project. The following concepts can help streamline SSBI and minimize the problems described in the previous sections:

- **Business glossary:** A *business glossary* allows users to describe and define *business objects* in a non-technical fashion. Definitions, descriptions, list of synonyms, classifications, and categorizations explain in different ways what business objects mean. In addition, it allows users to define relationships between all the business objects and to classify and categorize them. A business glossary can link the business objects to implementation concepts, such as data sources, tables, and columns.

A business glossary helps to streamline self-service BI.

- **Comprehensive search features:** The business glossary is needed for users to easily find the right data sources, the right tables, and the right columns by navigating and searching all the business objects. For example, if a user wants to study recent factory spills, the search capability must lead him to all the business objects related to factory spill. Next, it should show which data sources, tables, and columns contain factory spill numbers.

- **Reuse and sharing of metadata specifications:** It must be easy to reuse and share all types of metadata specifications, such as the connection, integration, transformation, and descriptive specifications. This must be possible in a way that doesn't slow down the user's development speed and that encourages them to share specifications. This improves reporting consistencies and minimizes specification proliferation.

- **Simplified maintenance:** It must be easier to implement global changes. For example, if the structure of a data source changes, it must be easy to implement that change in all the reports in such a way that all the specifications related to that data source are updated correctly.

- **Centralized authentication and authorization:** Data sources and SSBI tools have their own mechanisms for data security. A more centralized solution simplifies the setup and management of all authentication and data authorization specifications.

Some of these concepts are very technical by nature and some are more business-oriented. The next section describes how some of the more technical issues can be solved using data virtualization, and Section 8 describes how a *business directory* can help with the more business-oriented issues.

6 Streamlining Self-Service BI with Data Virtualization

Direct Data Access – In many organizations, self-service BI tools extract data directly from the data sources. A data source can be a data mart, a data warehouse, an internal production system, or some external data source, such as a social media network or a public database; see Figure 5. In between the SSBI tool and the data sources, there is nothing except for some driver technology.

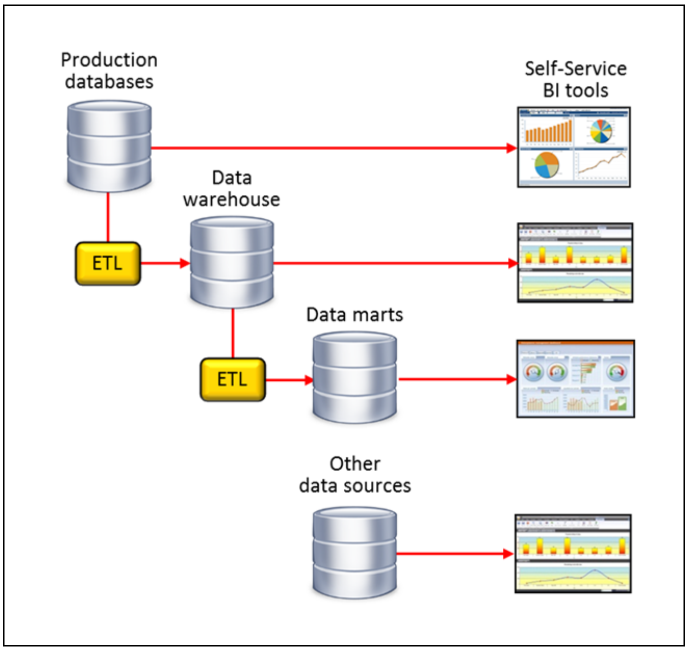


Figure 5 *Self-service BI tools can access directly data sources, such as a production system, a data warehouse, or some other source.*

This implies that users must deal with all the idiosyncrasies of all these data sources. They must deal with standards and solutions used in these systems, the naming conventions, and the coding conventions. And if the data source uses a non-SQL interface, somehow the data must be transformed to a SQL interface. For example, if users want to extract data from Salesforce.com, they must understand its proprietary interface that is not SQL-based, or they have to switch to Salesforce's own analytical tool called *Analytics Cloud*. But this involves getting acquainted with another BI tool, and this tool doesn't allow data stored in Salesforce to be integrated with other data sources. Another example of a data source that can be complex to access is an old mainframe-based production database in which all the data is structured hierarchically. Such older data sources can also contain repeating groups in the table structures.

Business users can lose valuable time analyzing how to access non-SQL data sources.

In such situations, quite some in-depth technical knowledge is required by the users to map these data structures to straightforward and flat SQL data structures. Business users can lose valuable time analyzing how to access some of these non-SQL data sources. Because this technical problem has nothing to do with the business problem he is trying to solve, in a way, it's a waste of time.

Another issue is whether users understand the data structures and all the data values? A mistake is easily made. Or, when two tables from different data sources must be integrated, but the columns on which to integrate don't contain standardized codes, will the business users know how to integrate them? Maybe access to an MDM system is required to implement a meaningful integration solution. Business users can easily drown in the technology.

When users implement incorrect metadata specifications, the consequence is incorrect reporting results, which leads to *incorrect business decisions*.

Implementing incorrect metadata specifications leads to incorrect business decisions.

Data Access Via Data Virtualization – Direct access of the data sources is the reason for several of the more technical issues of SSBI. Many of these issues can be solved or reduced by installing *Cisco Information Server* (CIS) to handle all the technical issues related to data access; see Figure 6.

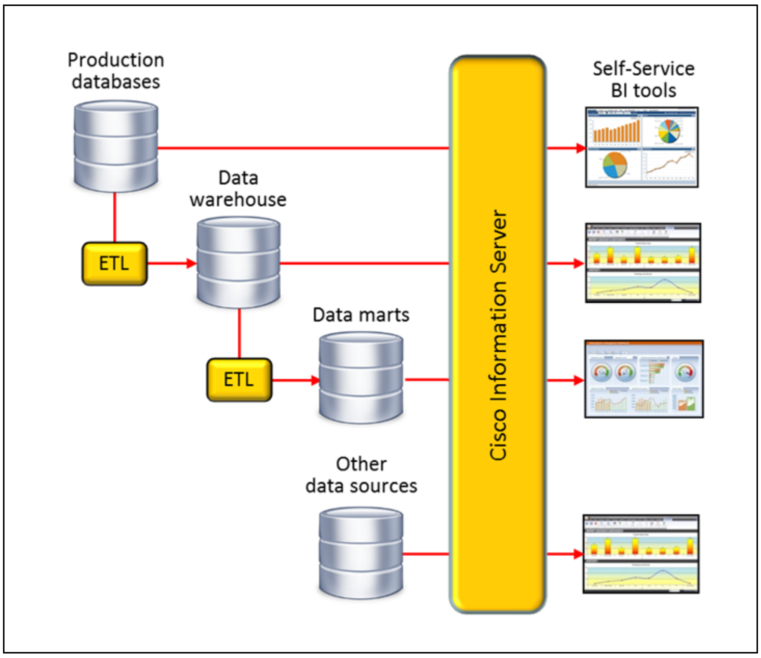


Figure 6 *Self-service BI tools access all data sources via the CIS data virtualization server.*

But it's not just a matter of placing CIS "in between" the SSBI tools and the data sources. To really get rid of all the issues, existing metadata specifications must be migrated from the SSBI tools and to CIS, and new ones must be defined in CIS right away. Especially the connection, transformation, integration, cleansing, descriptive, and security specifications must be extracted from the reports, and re-implemented in CIS. Only then can specifications be reused and shared, even across different SSBI tools. The visualization and collaboration specifications remain the domain of the SSBI tools.

Important to understand is that entering metadata specifications is as easy in CIS as it is in the SSBI tools. For example, defining connection and transformation specifications can be done with a similar effort. From the business user perspective there is no productivity loss.

Give SSBI users their own self-service data virtualization environment.

Migrating existing specifications to and implementing new ones in CIS implies that users must have the privilege to define these specifications in

CIS. It's recommended to give SSBI users their own *self-service data virtualization environment* within CIS.

Managed Self-Service BI – By using data virtualization, a self-service BI environment turns into a *managed self-service BI* environment, without the users losing flexibility, high productivity, self-serviceness, and independency. In fact, they benefit from it: their productivity improves, correctness of reports results improves, query performance improves, report maintenance simplifies, and so on.

When CIS is used, metadata specifications entered by users can be managed, maintained, and optimized by IT specialists and/or the users themselves. It's like working on the same set of specifications from two angles; see Figure 7. This does not influence reporting results, nor does it delay development of new reports. Management, maintenance, and optimization can all be done by IT transparently to the users.

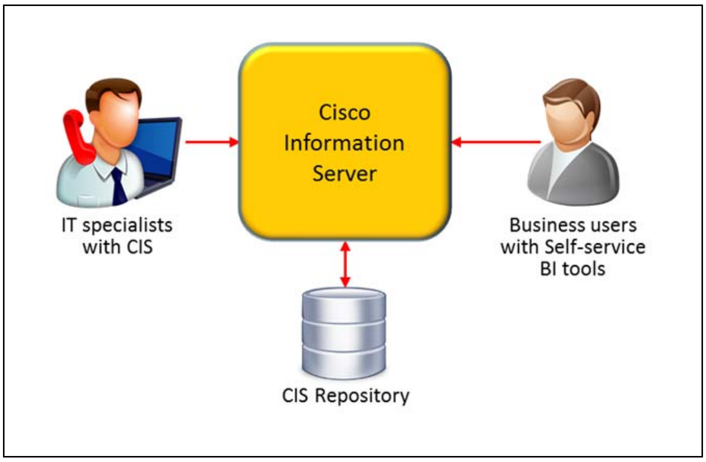


Figure 7 *With managed self-service BI, business users and IT specialists work in collaboration.*

In this architecture, IT specialists monitor different aspects. For example, IT monitors the performance of access to views and study whether the user's implementation can be optimized with more efficient code. In addition, IT can monitor the user's specifications. They can check whether users have implemented similar but not identical specifications. If so, the IT specialist can verify whether the implementations return consistent results. They can also check the validity of code. Do users implement the correct transformation logic? Do they forget specific cleansing rules? Do they use the right columns to join the tables? IT specialists can also help with the more technical aspects, such as predefining the connection specifications to ease the work of the business users.

To summarize, by combining SSBI with data virtualization, a close collaborative form of self-service development commences where IT specialists and users work together: *managed self-service BI*.

Combining SSBI with data virtualization results in managed self-service BI.

Benefits of Data Access via Data Virtualization – Managed SSBI through data virtualization has the following benefits:

- **Easy centralization of specifications:** The CIS data virtualization server acts as a centralized component in the architecture. All the specifications entered in CIS are stored centrally and can be reused by others. If a user implements a solution for integrating two data sources in CIS, it can be reused by all other users.

- **Improved user productivity:** The more users define their specifications in CIS, the more they can be reused. This improves user productivity, because the need to reinvent the wheel diminishes over time.

- **Hidden technical aspects:** As indicated in Section 3, extracting data from certain data sources can be quite a technological challenge. If the IT specialists implement all the connection specifications and transform the non-SQL data structures, users are able to use that data more easily.

- **Easy maintenance:** The more specifications implemented in CIS, the less specifications are proliferated across many tools, simplifying maintenance considerably. A global change is easier to implement, because most of the specifications are defined within CIS, and are not distributed over all the tools and reports.

- **Consistent report results:** The more specifications are shared, the more consistent the reporting results are. This applies for reports developed by different users with the same tool as well as for reports developed with BI tools from different vendors.

- **Higher quality report results:** By having IT specialists collaborate with the business, they can prevent the implementation of incorrect specifications. But this requires that IT and the business users work together and that they have short lines of communication and direct contact.

7 Implementing Managed Self-Service BI with CIS

This section describes how to organize the views in CIS to support and streamline managed self-service BI.

Four Layers of Views – Figure 8 presents the overall architecture of all the views in CIS. It consists of four layers of views. All the metadata specifications, except for the descriptive, visualization, and collaboration specifications, are implemented in one of the view layers. The SSBI tools access the top layer of views. The business users can define and change views defined in the top two layers. IT is responsible for the bottom two view layers. Note that while at first this might seem an unnecessary complication, the approach actually simplifies development and maintenance.

In this architecture, data access by the SSBI tools is combined with data access by the more traditional BI tools to increase the reusability level of metadata specifications and the consistency of reports (self-service or not). Note that this is not a requirement. Alternatively, two separate CIS environments can be set up, one for the SSBI users and one for the traditional users. This is a joined decision by IT and the business users where they have to carefully balance the advantages and disadvantages of both. One aspect to be discussed thoroughly is that a particular user can be an SSBI user and a user of traditional reports or dashboards. If these two tools don’t share specifications, results can be inconsistent and thus limiting the user’s trust in both systems; which one presents the correct result?

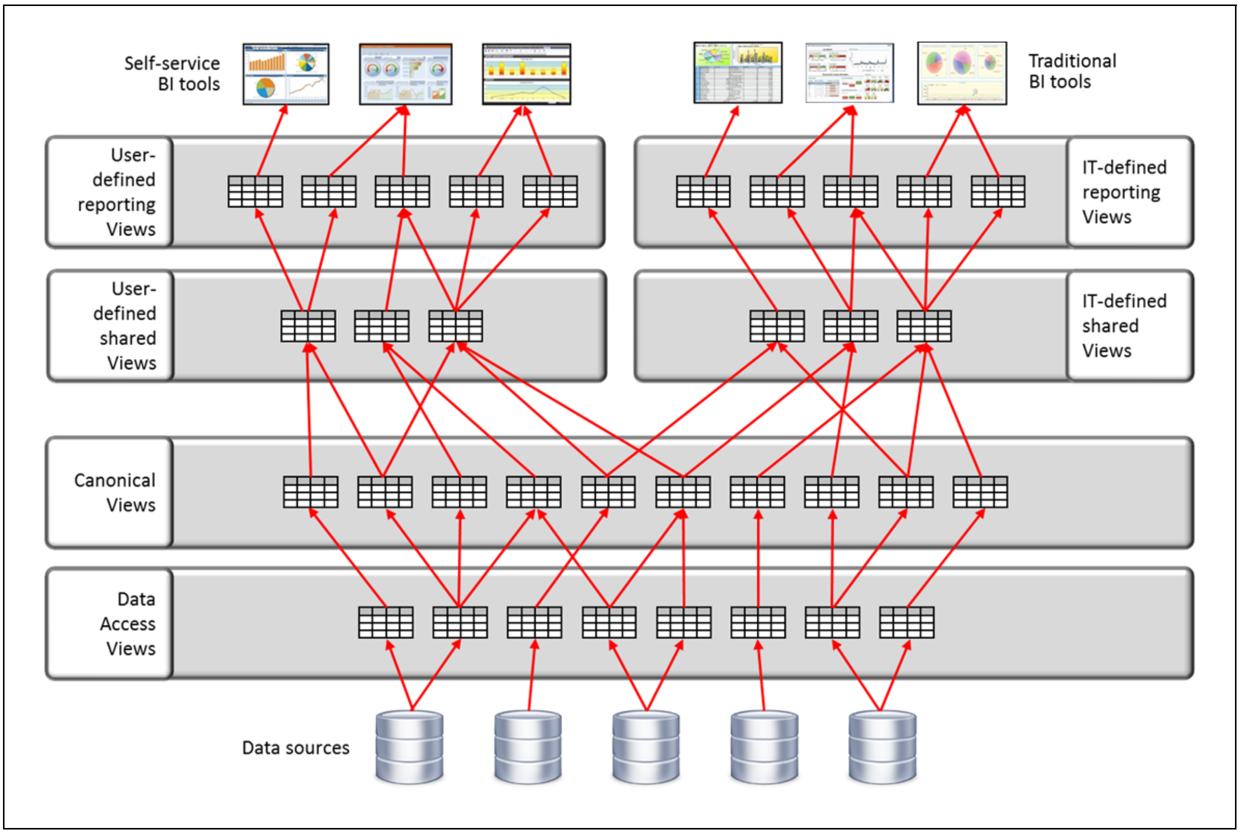


Figure 8 *Four layers of views; from data access to user-defined views.*

Data access views contain connection and transformation specifications.

Layer 1: Data Access Views – At the bottom of the diagram resides a set of views that offers access to all the data sources used by the traditional and the SSBI users. This layer of views is defined and maintained by the IT department, because it requires in-depth understanding of the data source technology and its data structures. Data access views typically contain all the connection specifications and the transformation specifications to turn non-typical data structures to straightforward relational data structures.

Canonical views have normalized data structures.

Layer 2: Canonical Views – On top of the data access layer, views are defined that show the data with a neutral data structure. Such a set of views is sometimes called a *canonical data model*. The data structures are normalized or highly normalized. The view definitions include specifications to integrate, cleanse, and transform data. The virtual contents of these views are easy to use and easy to integrate. Canonical views are defined by IT specialists who understand the data.

Data access rules are defined on these canonical views: which user is allowed to access which canonical view. If the query performance on particular views is poor, the IT specialists determine whether caches must be defined.

Layer 3: User-Defined Shared Views – Two separate stacks of views are defined on top of the canonical views: one stack of views for SSBI (on the left) and one for the traditional reports (on the right). The *user-defined* *shared views* (on the top left-hand side) contain specifications common to the reporting views. They are owned and defined by SSBI users and they can be shared amongst users. In addition, when IT specialists think they can simplify the maintenance on views by combining specifications from multiple views into one, they can define new views and make changes to views here as well. In fact, users and IT specialists collaborate closely on this view layer.

User-defined shared views contain specifications common to reporting views.

If specifications in the user-defined shared views are so generic and apply to every form of data access, it's recommended to push the specifications down in the canonical views. For example, if a user has discovered that a certain code in a column has been misspelled categorically, a cleansing specification can be added to the user-defined shared view to transform the incorrect code to the correct one. A better solution would be to implement that transformation in the canonical views, so that the IT-defined views benefit as well. Note however that users must be informed about this, because it may change their report results.

Layer 3: IT-Defined Shared Views – The views on the top right-hand side in Figure 8 are defined by IT. They are used by the views on the top level that are defined for the more traditional reports and dashboards. These views contain common specifications[3]. The IT-defined shared views exist to centralize metadata specifications that are not defined in the canonical model. These specifications are only relevant for the reporting views. IT-defined shared views are defined and maintained by the IT department. Aspects such as data governance and auditing play an important role here.

What applies to specifications in the user-defined shared views, applies to the IT-defined ones as well. When view specifications are so generic and apply to every form of data access, it's recommended to push them down to the canonical views

Layer 4: User-Defined Reporting Views – These are the views accessed by the SSBI tools and are defined by business users themselves.

IT specialists must monitor user-defined reporting views to identify common specifications. If common specifications are found, they must recommend and help users to relocate them to the shared views. Also, if common specifications are discovered that have been implemented inconsistently, IT must report this, so that the users can determine how to solve this inconsistency problem.

User-defined reporting views are defined for the self-service BI tools.

IT specialists can be involved in optimizing the efficiency of user-defined reporting views. Monitoring the query performance of these views gives IT insight in their efficiency. They may recommend to reformulate the query. Or, IT may propose users to cache their views if, for example, they're accessed frequently. If users have cached some views, IT may recommend to cache the underlying view on the shared view layer instead. In this case, other users may benefit from the cache as well. This saves disk space and shortens the time to refresh caches (one instead of multiple).

[3] These IT-defined shared views are comparable to the views with common definitions as shown in Figure 5 of the whitepaper *Migrating to Virtual Data Marts using Data Virtualization*; see http://www.cisco.com/web/services/enterprise-it-services/data-virtualization/documents/Whitepaper_Cisco_VirtualDM.pdf

Layer 4: IT-Defined Reporting Views – These are the views accessed by the traditional BI tools. These views are defined by IT specialists. In many cases these views have a structure that fits the requirements of the reports and the tools. So, quite often these views will have a star schema pattern. IT-defined reporting views must only contain specifications specific to a very small set of reports.

Views Versus Metadata Specifications – To summarize, Table 1 shows in which view layer it's best to implement specific metadata specifications. Because there can be all kinds of practical reasons to deviate from this table, it must be seen as a general recommendation.

	Connection specifications	Integration specifications	Transformation specifications	Cleansing specifications	Descriptive specifications	Visualization specifications	Collaboration specifications	Security specifications
BI Tool					✓	✓	✓	✓
Reporting views			✓		✓			
Shared user views		✓	✓	✓	✓			✓
Canonical views		✓	✓	✓	✓			✓
Data access views	✓	✓			✓			✓

Table 1 *A high-level overview of where to implement metadata specifications.*

Pushing Down Specifications – From the perspective of reuse, sharing, maintenance, productivity, and report result consistency, it's recommended to push down metadata specifications to lower-layered views. But it makes the architecture less flexible. Changing lower-level specifications has an impact on a larger audience and may have to be approved by a larger group of users. Implementing them on the highest level makes them more private and agile. IT specialists and business users have to balance these aspects to find the optimal solution.

8 Streamlining Self-Service BI with a Business Directory

The Data Avalanche – When users have access to a few production systems, a large data warehouse, several data marts, some external data sources, and quite a number of private spreadsheets, the number of tables and columns they have access to can be staggering and overwhelming. There can be hundreds of them. For example, a full SAP implementation alone can already contain thousands of tables. It's not unimaginable that users can't see the forest for the trees anymore. To adapt John Naisbitt's[4] famous quote: "Users are drowning in data sources, but they're starving for information."

Business users are confronted with a data avalanche.

[4] John Naisbitt, *Megatrends; Ten New Directions Transforming Our Lives*, Warner Books, first edition, 1982.

This *data avalanche* can lead to the following practical problems:

- **Users don't know data is available:** Because there is so much data, users may not be aware that certain types of data are available. And if they are not aware that the data exists, they won't use it for analysis, and this leads to missed and unexploited business opportunities. Users do not and cannot analyze data they have no knowledge of.

- **Users need time to find the right data:** If data is "hidden" in a large set of tables, it may take users a long time to locate the right tables for their queries, and a long time to convince themselves that they are really using the right tables. For example, sales data can be available in several tables distributed over several data sources. What is the right table for the user's question? This search for the right table decreases the user's productivity.

- **Users can't find the right data:** The data structures may be so complex, that, although the users know the data exists, they can't locate it. Especially if they start to access production databases directly, the table names can be quite cryptic. Even the data values themselves may be cryptically coded. As long as the right data hasn't been found, analysis can't start. It's like being data-rich and information-poor.

- **Users use the wrong data:** Users sometimes select the wrong tables to run their reports. For example, instead of using all the sales data, which they assume is updated yesterday evening, they may have picked a table that only contains European sales data and that is only up to date to last week. The table they picked may be a derivative of the table they should have picked. Their report results can still look good, especially if they look at average values and aggregates over long periods of time, such as months or weeks. They won't immediately see that the result is a few days short and that the average value hasn't been calculated based on all the stores, but only on those located in Europe.

- **Users don't understand the data:** To users it may not be clear what particular data values represent. For example, the value Santa Clara by itself means nothing. Does it refer to the city of Santa Clara located in California, or to the one in Utah, or is it the name of a city based in the south of Argentina? In fact, is it really a city name, because it could also refer to the mission of Santa Clara, or the Santa Clara County in California. Data is not always understood by users.

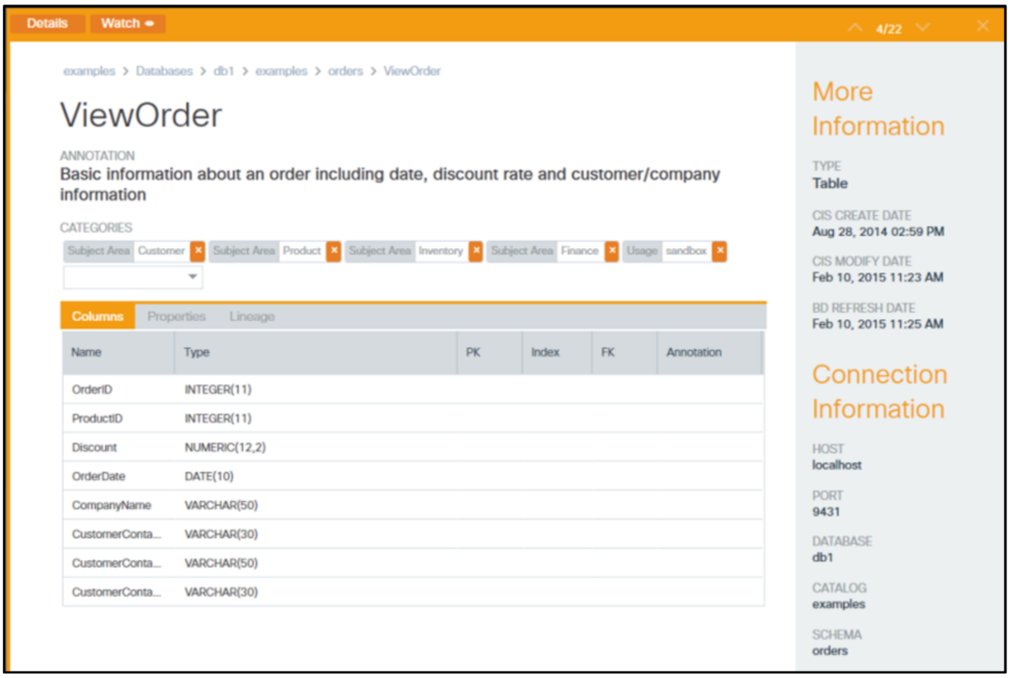
The Need for a Business Directory – Data virtualization does solve many of the problems of self-service BI as indicated in the previous sections, but not all of them. What's missing is a comprehensive and business-oriented search facility. CIS allows users to search for table and column names that contain certain words, but there is no guarantee that the right tables are found. For example, if a table in a data warehouse is called DW_DOM1_SAL34, that table is not found easily when a user searches for the word SALES.

Users must be able to retrieve full 360 degrees views of business objects.

Cisco Business Directory creates a business context for data.

What is needed is the ability for a more intelligent search feature. When users search for sales data, all the tables that contain sales data, plus their definitions, descriptions, and links to sales-related data must be shown. Users must be able to retrieve a full 360 degrees business view of sales data. The *Cisco Business Directory* (CBD) allows for this *business lens*. It allows users to introduce business terms, to describe them in non-technical terms, to organize and categorize the terms, and to search data. All this business information allows users to find the right data they need for their reporting and analytics. It creates a *business context* for all the available data.

Describing Business Objects – In general, names can be misleading. For example, in commerce an order is a confirmed request by one party to another to buy, sell, deliver, or receive goods or service, whereas in a military context it's a directive. Or, if an airline indicates that a "flight" takes six hours, what exactly does that mean? Does a flight include or exclude transfer time? CBD allows users to define and describe *business objects*, such as customer, order, SKU, flight, and product category. As an example, Figure 9 shows a CBD screenshot in which the business object ViewOrder is defined. It includes a definition and description.



examples > Databases > db1 > examples > orders > ViewOrder

ViewOrder

ANNOTATION
Basic information about an order including date, discount rate and customer/company information

CATEGORIES
Subject Area Customer | Subject Area Product | Subject Area Inventory | Subject Area Finance | Usage sandbox

Name	Type	PK	Index	FK	Annotation
OrderID	INTEGER(11)				
ProductID	INTEGER(11)				
Discount	NUMERIC(12,2)				
OrderDate	DATE(10)				
CompanyName	VARCHAR(50)				
CustomerConta...	VARCHAR(30)				
CustomerConta...	VARCHAR(50)				
CustomerConta...	VARCHAR(30)				

More Information

TYPE
Table

CIS CREATE DATE
Aug 28, 2014 02:59 PM

CIS MODIFY DATE
Feb 10, 2015 11:23 AM

BD REFRESH DATE
Feb 10, 2015 11:25 AM

Connection Information

HOST
localhost

PORT
9431

DATABASE
db1

CATALOG
examples

SCHEMA
orders

Figure 9 *With CBD new business objects can be defined and described.*

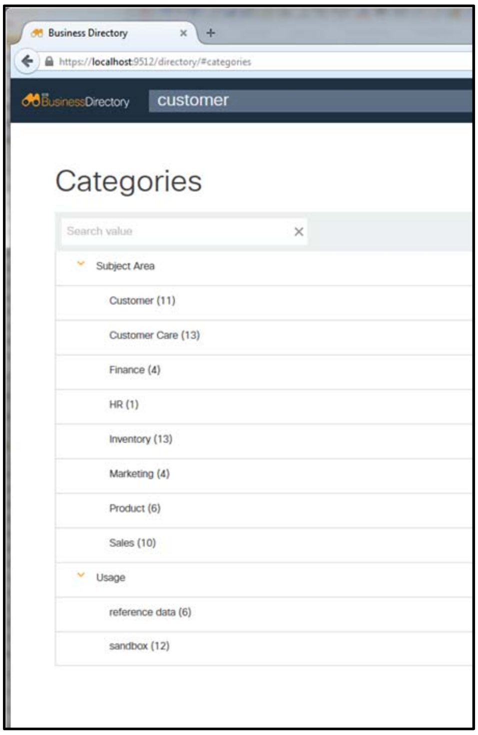
For each business object a list of *synonyms* can be defined as well. A synonym can be an acronym of a word or a shorter version of a word. Synonyms can also be handy if user groups use different names to refer to the same business object. Users and IT specialists can also add private and public *comments* in the business directory. CBD remembers who entered what. When business objects are defined, they can be linked to the views in CIS that contain the right data.

Note that defining business objects in CBD is done independently of CIS. So, when a business object is defined there is no need to link it straightaway to some technical construct, such as a data source, a view, or column. This can be done later.

Descriptive information drives the user's search capabilities.

Besides having a definition, a description, and a list of synonyms, the value of all this descriptive information is that it drives the searching capabilities. The more descriptive information is available, the easier the right business objects are found.

Categorizing Business Objects – Users can categorize all the business objects they define. Categorization is done through hierarchical grouping. For example, costs can be divided in labor costs, capital costs, and costs for fixed assets. Or, costs can be divided in fixed and variable costs. As an example, Figure 10 shows a list of categories, including Customer, Customer Care, and Finance.



Categories

Search value

Subject Area
- Customer (11)
- Customer Care (13)
- Finance (4)
- HR (1)
- Inventory (13)
- Marketing (4)
- Product (6)
- Sales (10)

Usage
- reference data (6)
- sandbox (12)

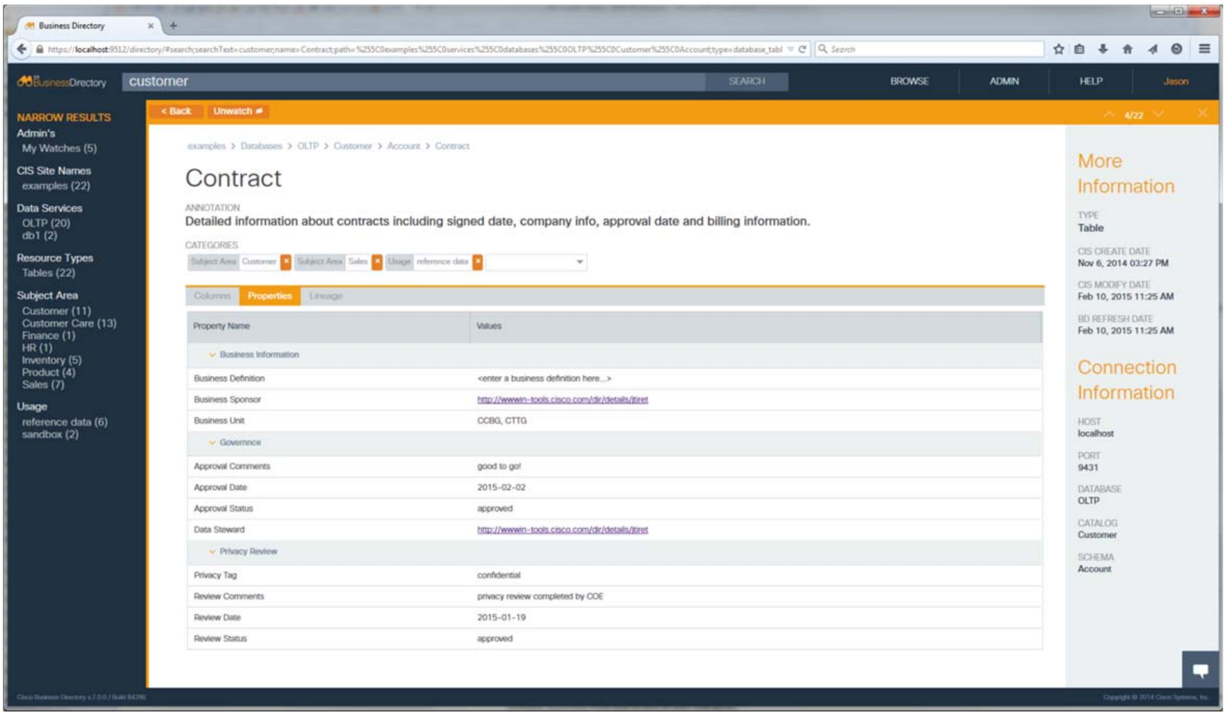
Figure 10 *With CBD business objects can be classified in categories.*

One business object can be assigned to multiple categories. For example, Figure 11 shows that the business object called Contract belongs to the categories Customer and Sales.

Categorizing is a very valuable instrument for users to search the right business object, but also to understand the data they are looking at. For example, it's important for a user to understand that the table he is analyzing contains the variable costs, but not the fixed costs. This may not be clear by just looking at the data in the table. The more descriptive business data is available to describe data, the more valuable the data is.

Searching for Business Objects – When users want to find the right data, they search for the business object they're interested in. For example, Figure 12 shows a user searching for customer data. The result consists of the business objects AccountShare, Contract, Account, Account History, and so on. It's clear that CBD allows users to locate data through a business context lens.

Business Specifications – All the definitions, descriptions, categorizations of business objects form metdata specifications as well. These are referred to as *business specifications*. Table 2 is an extension of Table 1 and includes these business specifications and CBD.


Business Directory
https://localhost:9512/directory/#search;searchText=customer;name=Contract;path=%255C0examples%255C0services%255C0databases%255C0OLTP%255C0Customer%255C0Account;type=database_tabl
Search
BusinessDirectory
customer
SEARCH
BROWSE
ADMIN
HELP
Jason
< Back
Unwatch
4/22
Columns
Properties
Lineage

NARROW RESULTS
Admin's
My Watches (5)
CIS Site Names
examples (22)
Data Services
OLTP (20)
db1 (2)
Resource Types
Tables (22)
Subject Area
Customer (11)
Customer Care (13)
Finance (1)
HR (1)
Inventory (5)
Product (4)
Sales (7)
Usage
reference data (6)
sandbox (2)

examples > Databases > OLTP > Customer > Account > Contract

Contract

ANNOTATION
Detailed information about contracts including signed date, company info, approval date and billing information.

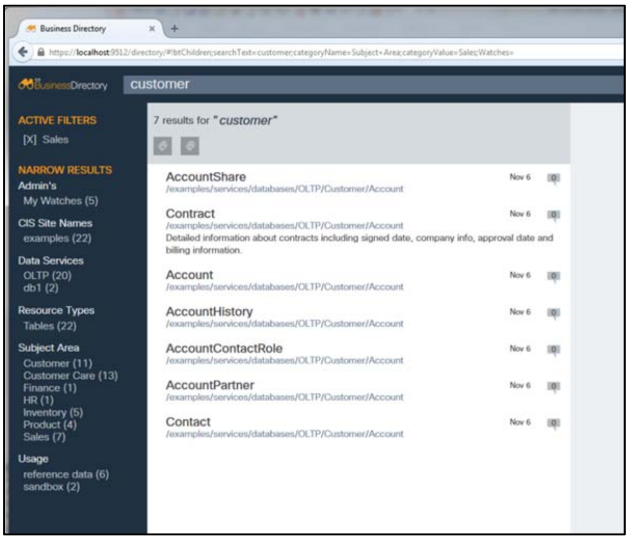
CATEGORIES
Subject Area Customer ☒ Subject Area Sales ☒ Usage reference data ☒

Property Name	Values
Business Information	
Business Definition	<enter a business definition here...>
Business Sponsor	http://wwwin-tools.cisco.com/dir/details/jtiret
Business Unit	CCBG, CTTG
Governnce	
Approval Comments	good to go!
Approval Date	2015-02-02
Approval Status	approved
Data Steward	http://wwwin-tools.cisco.com/dir/details/jtiret
Privacy Review	
Privacy Tag	confidential
Review Comments	privacy review completed by COE
Review Date	2015-01-19
Review Status	approved

More Information
TYPE
Table
CIS CREATE DATE
Nov 6, 2014 03:27 PM
CIS MODIFY DATE
Feb 10, 2015 11:25 AM
BD REFRESH DATE
Feb 10, 2015 11:25 AM

Connection Information
HOST
localhost
PORT
9431
DATABASE
OLTP
CATALOG
Customer
SCHEMA
Account

Figure 11 *With CBD business objects can be assigned to multiple categories.*


Business Directory
https://localhost:9512/directory/#!btChildren;searchText=customer;categoryName=Subject+Area;categoryValue=Sales;Watches=
BusinessDirectory
customer

ACTIVE FILTERS
[X] Sales

NARROW RESULTS
Admin's
My Watches (5)
CIS Site Names
examples (22)
Data Services
OLTP (20)
db1 (2)
Resource Types
Tables (22)
Subject Area
Customer (11)
Customer Care (13)
Finance (1)
HR (1)
Inventory (5)
Product (4)
Sales (7)
Usage
reference data (6)
sandbox (2)

7 results for *"customer"*

AccountShare Nov 6 0
/examples/services/databases/OLTP/Customer/Account

Contract Nov 6 0
/examples/services/databases/OLTP/Customer/Account
Detailed information about contracts including signed date, company info, approval date and billing information.

Account Nov 6 0
/examples/services/databases/OLTP/Customer/Account

AccountHistory Nov 6 0
/examples/services/databases/OLTP/Customer/Account

AccountContactRole Nov 6 0
/examples/services/databases/OLTP/Customer/Account

AccountPartner Nov 6 0
/examples/services/databases/OLTP/Customer/Account

Contact Nov 6 0
/examples/services/databases/OLTP/Customer/Account

Figure 12 *Searching for business objects with CBD.*

	Business specifications	Connection specifications	Integration specifications	Transformation specifications	Cleansing specifications	Descriptive specifications	Visualization specifications	Collaboration specifications	Security specifications
BI Tool						✓	✓	✓	✓
Cisco Business Directory	✓								
Reporting views				✓		✓			
Shared user views			✓	✓	✓	✓			✓
Canonical views			✓	✓	✓	✓			✓
Data access views		✓	✓			✓			✓

Table 2 *A high-level overview of all the metadata specifications including the business specifications.*

Summary – Problems with analyzing data values without having the right metadata, can lead to strange situations. This was illustrated by a brief and well-known discussion on Twitter. A woman tweeted the message "I hate Dell." Dell, the computer manufacturer, responded by tweeting that they were sorry to hear that, and that they were happy to fix the problem. Her response was "Hahaha, my boyfriend's name is Dell, not you guys!" Without descriptive specifications, data values can be meaningless. With Cisco Business Directory data can be described in detail allowing business users to find the right data in the data avalanche. Working with data without a business directory is like reading a book written in words from an unknown foreign language for which you do not have the dictionary.

But working with a business directory such as CBD requires discipline on the side of IT and on the side of the business users. All the data sources that are made available through CIS, and all the views developed by IT on the bottom two layers, must be defined and described in CBD by IT. Without this descriptive information, the views may be worthless to the users, because they won't know exactly what the values mean.

Working with a business directory requires discipline.

9 Getting Started

This section describes how to streamline a self-service BI environment and to promote it to a managed self-service environment in which metadata specifications are implemented in CIS and descriptive information is maintained in CBD to improve search capabilities and to explain what data means.

What Not To Do! – The general recommendation to streamline self-service BI is *not* to let IT opt for the enterprise-wide approach where IT first defines, documents, and categorizes all the business objects with CBD, identifies all the data sources the SSBI users need, develops all the integration and transformation specifications, links the business objects to views defined in CIS, makes all the views available, and so on. Such an exercise can take many months. Undoubtedly, this would meet a lot of resistance from the business users, because it clashes with the experimental and somewhat opportunistic approach used by SSBI users. It would turn self-service BI in a more classical form of BI where users have to wait for IT staff to define and develop reports. Gradually, the popular do-it-yourself approach will dissolve, and all the problems described in Section 2 will return.

A Reactive Development Approach – By deploying CBD and CIS in the right way, users can still enjoy the do-it-yourself approach and benefit from a more centralized approach of metadata specifications and descriptions. For this, IT must adopt a *reactive development approach* extended with monitoring and recommendations. They must operate like a garage does: They react only when a customer needs his car serviced.

Managed self-service BI demands a reactive development approach.

The Starting Point – It all starts with a user who needs access to a new data source. Without CBD and CIS, when users need access to a new data source, they must ask the IT department to help out. Organizing the connection specifications can be quite technical. In addition, there may be issues with respect to data authorization. Moreover, the source may not be accessed directly, because that can lead to interference and performance degradation. So, whether or not CBD and CIS are involved, the first step involves almost always the help of IT staff.

When users need access to a new data source, make it available through CIS. In other words, be sure that users don’t access data sources directly, but always through CIS. This simplifies access to data for the users, because to them it feels as if all the data is available through one source. It also hides the fact that some sources are not SQL-based. The IT specialists use CIS to transform the non-SQL data sources to SQL, thus simplifying access for the business users. When it’s not allowed to access the data source directly, IT can develop caches for the tables in these data sources.

To speed up the process to make data available, IT must develop the required data access views. On top, they develop a set of canonical views to make access easy. There is no need yet to include all the imaginable transformation and integration specification. These will be developed gradually. Right now, it’s all about making the views and data available as quickly as possible. After making these canonical views available, users can develop their own user-defined shared and reporting views.

Monitor and React – After making the views available to the users, the IT department waits, monitors, and reacts. The following aspects must be monitored and can lead to action:

- **Performance:** Is access to certain views by definition slow? If so, the IT specialists can propose a more efficient solution. If they can guarantee that the virtual contents of the views stay identical, users will never object. Optimizing views only makes sense if a view is being used regularly.

- **Deduplication of code:** If views defined by different users are identical or close to identical, IT may propose to merge the views. This leads to a higher level of centralization of specifications. Another situation can be if two views contain similar specifications. In such a case it may be smart to define a shared view that contains the similar specifications and then remove these same specifications from the original views.

- **Documentation:** Keep reminding users to document their data sources and views in CBD.

- **Clean up:** If access to views has been reduced to zero for some time, IT can propose to drop views in order to clean up the environment. This makes finding the right data easier. Too many “dead” views clutters the picture.

A Balanced Distribution of Work – This highly reactive, agile, and collaborative approach tries to combine the strengths of SSBI with those of data virtualization and a business directory. In fact, it helps to avoid the pitfalls of SSBI as described in Section 6. It helps to make every SSBI project a long-term success.

Combining the strengths of SSBI with those of data virtualization and a business directory.

In this approach work is distributed more evenly over business users and IT specialists. Figure 13 illustrates the distribution of development work in the days before SSBI. Almost all the work was done by IT specialists. They were responsible for coming up with and implementing all the metadata specifications. Then, SSBI came along, and the balance was reversed: almost all the work was done by the users, and IT was kept in the dark. Both situations are far from ideal, as explained in this whitepaper.

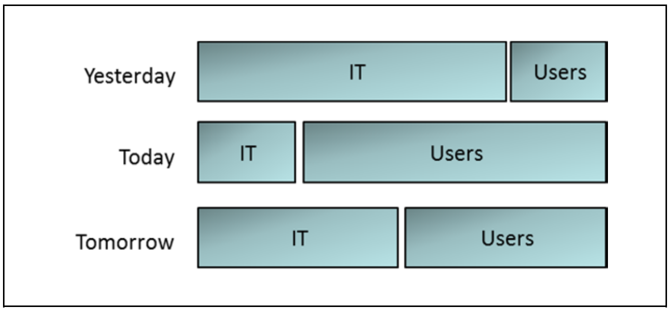


Figure 13 *The shift of responsibilities and work between business users and IT.*

By extending SSBI with CIS and CBD, a better balance can be reached. The work is more evenly balanced across the users and IT specialists. CIS and CBD allow organizations to find that ideal balance.

About the Author Rick F. van der Lans

Rick F. van der Lans is an independent analyst, consultant, author, and lecturer specializing in data warehousing, business intelligence, database technology, and data virtualization. He works for R20/Consultancy (www.r20.nl), a consultancy company he founded in 1987.

Rick is chairman of the annual European Enterprise Data and Business Intelligence Conference (organized annually in London). He writes for Techtarget.com[5], B-eye-Network.com[6] and other websites. He introduced the business intelligence architecture called the *Data Delivery Platform* in 2009 in a number of articles[7] all published at BeyeNetwork.com. The Data Delivery Platform is an architecture based on data virtualization.

He has written several books on SQL. Published in 1987, his popular *Introduction to SQL*[8] was the first English book on the market devoted entirely to SQL. After more than twenty five years, this book is still being sold, and has been translated in several languages, including Chinese, German, and Italian. His latest book[9] *Data Virtualization for Business Intelligence Systems* was published in 2012.

For more information please visit www.r20.nl, or email to rick@r20.nl. You can also get in touch with him via LinkedIn and via Twitter @Rick_vanderlans.

About Cisco Systems, Inc.

Cisco (NASDAQ: CSCO) is the worldwide leader in IT that helps companies seize the opportunities of tomorrow by proving that amazing things can happen when you connect the previously unconnected. Cisco Information Server is agile data virtualization software that makes it easy for companies to access business data across the network as if it were in a single place.

For more information, please visit www.cisco.com/go/datavirtualization.

[5] See http://www.techtarget.com/contributor/Rick-Van-Der-Lans
[6] See http://www.b-eye-network.com/channels/5087/articles/
[7] See http://www.b-eye-network.com/channels/5087/view/12495
[8] R.F. van der Lans, *Introduction to SQL; Mastering the Relational Database Language*, fourth edition, Addison-Wesley, 2007.
[9] R.F. van der Lans, *Data Virtualization for Business Intelligence Systems*, Morgan Kaufmann Publishers, 2012.