

---

## ONTOLOGY AND DATA WAREHOUSING

### HOW DATA WAREHOUSING INVESTMENTS CAN DELIVER CONTINUOUS ROI BY AUGMENTING THEM WITH ONTOLOGIES AND INFERENCE CAPABILITIES

#### Executive Summary

Data warehouses are designed to aggregate data and allow decision makers to obtain accurate, complete and up to date information. A data warehouse stands or falls on the quality of the data it contains. Most existing warehouses contain a subset of the available data and allow a limited range of queries on that data, because it is impossible to justify the cost of development of a more complete system. As a result, the quality and value of information suffers while managers are missing key facts required to make informed decisions.

This white paper shows how Network Inference's Cerebra Suite of products can help to make a data warehouse more effective for less developer effort by simplifying ETL efforts, exploiting BI tools more effectively, and leveraging important metadata for more agile IT infrastructure. Describing a data warehouse with ontologies can lead to better information flow within an organization and to more effective decision-making.

#### Introduction

Despite the best efforts of the enterprise systems vendors, few organizations store all their data in a single database. Mergers and acquisitions, upgrades, legacy systems that are essential and cannot be phased out, internal politics and sheer common sense ensure that multiple databases will continue to exist for the foreseeable future. Much of the useful information in many organizations is contained in the spreadsheets and single-user databases on users' desktops, and this is also unlikely to change.

Organizations recognize that the quality of their information is a key competitive factor. Streamlined internal information flows and high-quality reporting are considered essential to a modern business — but the required information is fragmented, held in several on-line transaction processing (OLTP) databases and dozens or hundreds of small, hand-crafted reporting systems, all of which have different definitions of terms, different scopes, different user interfaces and different goals.

A data warehouse aims to crystallize all of this different information into a single, central system, with real-time querying of data properties based on frequently updated operational data. These on-line analytical processing (OLAP) systems may store terabytes of data and support queries from thousands of users. A data mart is a smaller version of a warehouse, with its structure optimized for a particular department or business function; these may still run to tens or hundreds of gigabytes. A typical architecture is shown in Figure 1.

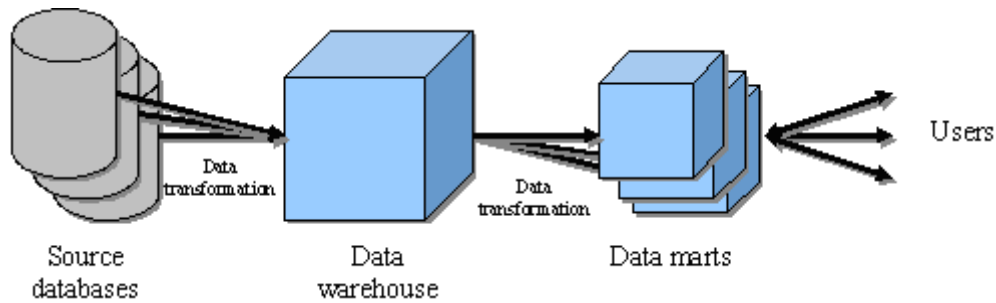


Figure 1: Architecture of an OLAP system

### Data Warehouse Components

A typical data warehouse or data mart contains three components:

- A relational database optimized for queries;
- One or more multi-dimensional aggregations stored in some custom data structure, typically a hypercube (see later);
- A way of transforming data from multiple OLTP schemata into a single schema for the warehouse.

The remainder of this section looks at these three components in more detail.

### Relational Database Star Schema

The optimized relational database typically uses a star or snowflake schema. A star contains a single, large table of facts, and several dimension tables that map identifiers to values. There may be several separate stars in one warehouse. A four-dimension example for a retailer is shown in Figure 2:

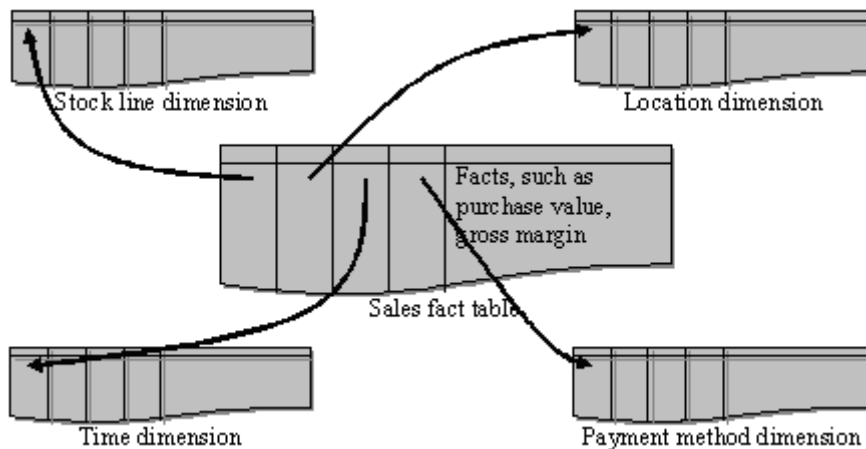


Figure 2: A star schema for a retailer's data warehouse

The fact table holds identifiers for the various dimensions, and numeric values. Rows contain the finest level of detail available through the warehouse.

Each dimension has one associated dimension table that holds all its data. In this example there are four dimensions: Stock line, Location, Time and Payment method.

Each dimension table holds several levels of hierarchical information, stored in multiple columns. For example, the Location dimension might hold, for each location identifier:

- Store name
- District name
- City name
- County name
- State name
- Country name

This approach allows a user to move between levels quickly, and to ‘drill down’ to more detailed information.

### Relational Database Snowflake Schema

Sometimes a dimension table is complex enough to be split into its own star: a snowflake schema:

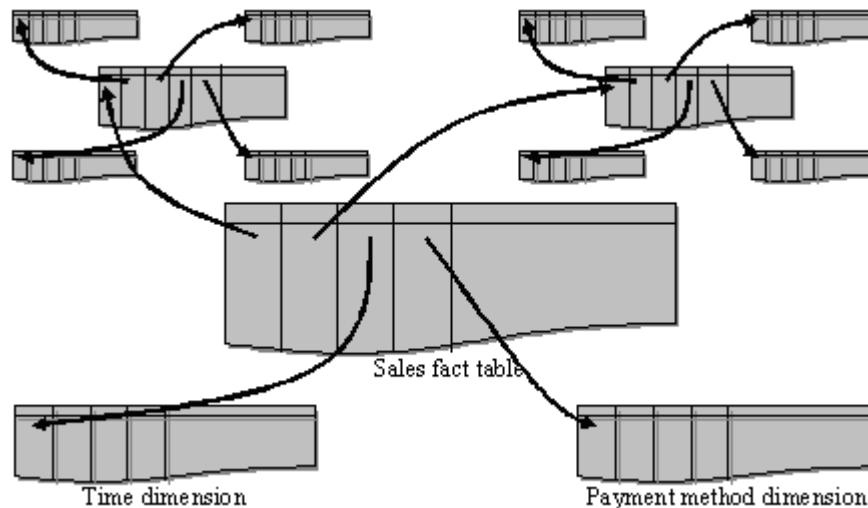


Figure 3: A snowflake schema for a retailer's data warehouse

### Relational Database Hypercubes

Relational databases are good at storing large volumes of similar data and retrieving small parts of that data. They are less successful at calculating summaries, such as totals, over large parts of that data. As a result, other techniques have been developed for storing those summaries which allow them to be queried quickly and efficiently. These are usually based around hypercubes.

A cube has three dimensions — for example time, location, and payment method. If it is divided up on all three dimensions into many tiny cells, it can store combinations of three OLAP dimensions in those cells. Each cell corresponds to one possible combination of values.

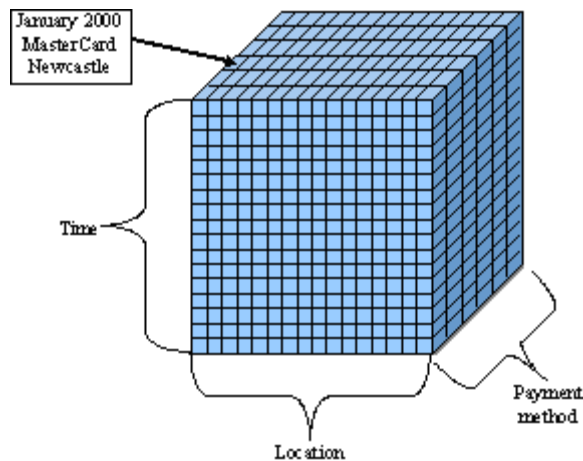


Figure 4: A cube of cells

However, many warehouses contain fact tables with more than three dimensions. Our simple retail warehouse contains four: stock line, time, location and payment method. A hypercube is simply a cube with more than three dimensions. It's more difficult to visualize, but simply allows the same cell construction with more complex fact tables.

Reporting tools will treat the hypercube and the underlying OLAP database as alternative ways of retrieving information. Top-level summary information will usually be obtained from the hypercube; as a user drills down to smaller amounts of data, queries will be run against the underlying relational database.

### Relational Database Transformation

Stars, snowflakes and cubes are good for reporting, but poor for transaction processing as each approach requires its own schema. In addition, a data warehouse typically obtains data from many different data sources. This requires complex transformations to obtain the source data, scrub it to remove or repair missing and inconsistent values, and to add it to the warehouse with attribution so that inaccuracies can be traced. The larger the system becomes, the more complex the transformations.

### Limitations of Existing Data Warehouse Technologies

There are four problem areas in existing data warehouses that the Cerebra Suite can provide value by introducing an ontology-based layer into the warehouse infrastructure:

- Uneconomical Methods for Describing Complex Data
- Impossible to Perform Concept-based Queries
- Limited Transformation Support for Federated Data
- Minimal Reuse of Important Business Metadata

### Describing Complex Data is Uneconomical

It's easy to design a data warehouse capable of storing large amounts of similar data. It's difficult to design one that can handle complex data and retain details from each source. While it is indeed true that third normal form systems can represent nearly any kind of data, the greater the complexity of the data relationships, the more

tables, keys, and eventually joins on queries will need to be put in place to negotiate the tabular and two-dimensional physical table space. To combat this, many data warehouses have a mix of normalized and dimensional data representations to give them performance in querying summaries as well as flexibility to do ad hoc queries. Nonetheless, inserting data from external systems into warehouses is often like fitting a square peg into a round hole.

- Hypercubes are only just starting to be able to aggregate hierarchical data. Microsoft SQL Server 2000 has support for ragged and unbalanced hierarchies, but not for more complex structures such as multiple-inheritance hierarchies.
- Star and snowflake schemata are inflexible when you need to represent complex relationships within a dimension. The data in a warehouse often ends up as the lowest common denominator of the data from its sources, with the complex information removed in the interests of uniformity.
- A minor addition to the warehouse schema is often as difficult to implement as a major revision. This discourages change and makes it more likely that a new source will be represented poorly – by simply adding new columns, tables or flags rather than accurately changing the data model.

### **Performing Concept-Based Queries is Impossible**

In relational and cube warehouses, with or without dimensional information, ad hoc queries as well as stored procedures are issued to the physical table space – making direct use of the table and column information that describes the physical persistence structure. Usually this physical table space has little or nothing to do with the actual business concepts of concern to the data consumer. Users are left to refer to data dictionaries – often out of date – to discover what certain columns are supposed to contain. This makes using and querying a warehouse problematic for all but a few highly-trained database experts. Overall, querying is problematic, particularly in the support given to users of the warehouse:

- Frequently, few or no constraints are placed on the data in the warehouse, on the assumption that all combinations of data might be generated by a source. A user is therefore presented with a multidimensional querying interface with little or no guidance as to what combinations are possible. For example, payment with a UK-only debit card would not be possible in a US store, but it is often possible to drill down to that cell with no indication that the combination is invalid.
- The dimensions are simple hierarchies. A user can drill down an existing hierarchy, but typically cannot search for values in that hierarchy matching particular criteria because the criteria themselves are not in the warehouse. For example, it might be difficult to use the retailing warehouse to identify trends in large stores, because there is no dimension for store size and the store size is not retained in the location dimension table. Although new information can be appended to existing models, most non-trivial additions would require a dump and rebuild of the tablespace, which is oftentimes prohibitive.

- If information is not retained, it cannot be displayed. Data warehouses are typically poor at retaining the complex descriptions surrounding terms in the dimension tables; instead, they tend simply to retain names. This makes it impossible for a user to find additional information about a term without going to the data source.

### Limited Transformation Support for Federated Data

Support for the warehouse transformation process is limited. It is worth noting that despite the wide array of available tools to assist with transformations – ETL, EII, EAI – there are still major difficulties with unifying disparate silos of data from federated sources. For example, ETL still relies on the use of point-to-point transformation code. Although more sophisticated systems will assist code generation with metadata-driven dictionaries – the metadata itself is highly proprietary and a wasted effort unless an organization is willing to standardize enterprise-wide on a given vendor. Likewise, EII tools rely on complicated federated queries to join data during transforms and leverage proprietary metadata repositories for partial automation of the process. Additional problems include:

- There is rarely any declarative support for combining and summarizing source fields as the data is transformed. Complex transformations can only be expressed as code. Changing and testing that code is difficult and time-consuming.
- Adding a new data source to the warehouse is a laborious task, as each source requires its own set of transformations and checks. It is difficult to re-use logic from existing transformations as it is buried in source-specific code. Conversely, we frequently see the same source feeding multiple data marts, each with a separately coded extract and transform, leading to redundancy and poor productivity as well as inconsistency and error.
- There is little support for populating and updating the dimension tables that contain the de-normalized terms on which the user searches. Hierarchies are often difficult to represent even when they can be gleaned from the source data.

### Minimal Reuse of Important Business Data

Finally, there is the problem that coping with the previously mentioned challenges and staying current with integrations of all relevant silos of data, will always lead to an environment of different data representations of your warehouses and different representations from your data sources. The manual and transformation code intensive methods for staying current with these challenges necessitates a dispersal of important business rules, logic, domain knowledge and other transformation metadata in various incompatible, hard-wired, formats. This leads to:

- Rediscovery of the same inconsistencies and incompatibilities, but little that can be done to assuage the need to re-write transformation code, joins, queries, and procedures over time again and again to support the business as new marts come online or new warehouses service organizations as they evolve.

- Re-implementation of the same fixes in code on a repetitive basis. As changes to an overall environment need to roll out to different warehousing centers, the same kinds of metadata is required again and again to assist with code generation and rationalization of data semantics.
- Procedural code going out of date, being difficult to change and almost impossible to test thoroughly. As schemata changes inevitably occur it is nearly impossible to trace back to the code residing throughout an enterprise – where its touch points and dependencies are – without a central façade to broker input and output to the warehouse environment. In many cases, maintenance problems may quickly overwhelm an organization with many warehouses.

### The Net Result

An attempt to solve these problems with conventional methods would tie up resources for years and be prohibitively expensive. As a result, your warehouse is less effective than it could be: valuable information is missing or difficult to find. Users cannot find relevant information and business decisions are either slower to make or are made without all the facts, leading to less effective decision-making. Furthermore, there is very little ability to adapt to business change.

Data warehouses have proven their worth: first, by providing consistent, comprehensive and accurate reporting (answering “what happened?”), second, as a basis for analytic detection (answering “why did it happen?”), third, by using data mining to do predictive analysis for things like forecasting (answering “what’s going to happen?”). Data warehouses are large investments which need to be further leveraged to provide continuous ROI (“making the right things happen”).

Well-implemented data warehouses can be further extended to play a more active role in a company’s decision-making, not just at the strategic level but at the tactical level as well. Many technologies are now in place to support that effort, including SOA, Web services, XML standard. It’s reasonable to expect data warehouses will continue to evolve. By augmenting the physical data schemas in a warehouse with more flexible, expressive, and adaptive ontologies, data warehouses can be used generate even greater returns on their initial investment. To accomplish this continuous ROI with existing warehouses, business should turn to the emerging Semantic Web data specifications as a lever to further optimize historic investments in warehousing technology.

Network Inference Cerebra provides a simple on-ramp to emerging data standards that can dramatically improve the effectiveness of a warehouse ecosystem.

### How Cerebra Can Help

- Ontologies and inference address many of warehousing’s current limitations. A flexible, extensible data representation system that allows you to store, change and extend properties of terms in the warehouse so that the warehouse itself becomes more change resilient

- A fine-grained data representation system so that you can make more complex searches and be given more complete information about terms in the warehouse
- A system that can represent terms and complex, fine-grained constraints between terms and can act as your basis for guided exploration and querying, eliminate obviously inconsistent queries
- A system that can do all of the above to act as a central resource to aid transformation from many data sources, retain the fine detail in each and allow re-use of your work in creating and maintaining the representation.

The Cerebra product range provides such a solution. Cerebra allows an organization to construct, maintain and use ontologies. A relational database is a structured, formal representation of a set of data; in the same way, an ontology is a structured, formal representation of an area of knowledge. It defines and restricts what can be said about that area of knowledge.

Ontologies are flexible and extensible. An ontology is often changed as an organization gains insight into its data and its knowledge, and it is often extended to cover new areas of knowledge as new systems are implemented.

The Cerebra product range contains several products that can help with data warehousing:

- **Cerebra Server** is the core inference engine that stores an ontology, checks it for consistency as it is changed and extended, and accepts queries about terms in the ontology such as ‘What kinds of payment methods exist?’ or ‘Is “handset” an equivalent term for “mobile phone”?’
- **Cerebra Construct** provides an ontology construction and editing environment, enabling an organization to extend and alter an existing ontology.
- **Cerebra VRMS** allows non-engineering knowledge contributors to create vocabularies, rules, logic and data constraints that will be reflected in the ontology – without the end-user knowing that they are constructing a formal knowledge representation

### Representing Complex Terms

Ontologies represent concepts, relationships between concepts and combinations of concepts (terms). Part of an ontology surrounding one mobile phone (the Motorola L7089) is shown in Figure 5:

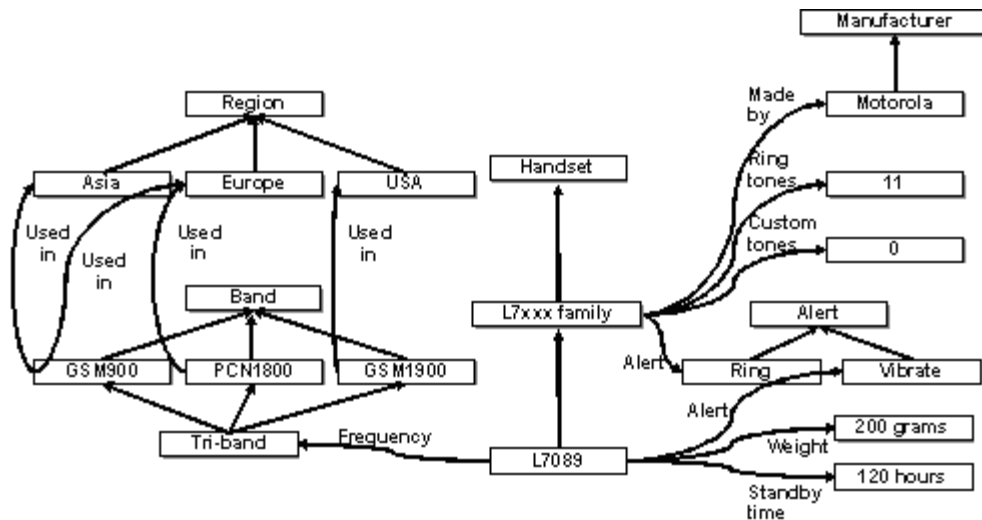


Figure 5: An ontology of mobile phones, showing one phone

This representation is very fine-grained: each concept can be associated with a different set of properties. By contrast, a relational database associates a fixed set of properties with each concept, and stores the result in a table. The ontology also allows simple and complex hierarchies: Tri-band is defined as a combination of the GSM900, PCN1800 and GSM1900 bands, for example. Properties can be inherited: the L7089 is a member of the L7xxx family of phones, so is made by Motorola.

Identifiers to describe terms from the ontology can be stored in fact tables and in hypercubes. The ontology stores the structures that used to be stored in dimension tables. This allows much more flexible representation of terms, and more of the source's detail can be retained in the warehouse.

As an organization's rationalization of its data increases, the ontology will change and will be extended. Identifiers are retained between versions of the ontology. This allows previous meanings to be kept. This can be done within a relational database, but involves the use of snapshots and links. Using ontologies is a much more straightforward and flexible way to attack the problem. The fine structure of an ontology also encourages developers to retain fine distinctions, so it's more likely that you will know what has changed between the old and new definitions and be able to correlate new and old data. This gives your organization the flexibility to extend and change its definitions without changing the meaning of old data.

### Assisting the User

Your data warehouse only becomes valuable when your users can search it to gain timely, accurate answers to their questions. Most of these questions are ad-hoc queries. Time may be wasted trying to find out whether the warehouse contains the data to answer the query, and how to phrase the query. A user needs assistance in navigating the warehouse and in phrasing the query.

The hierarchical structures of the concepts allow Cerebra to assist the user in navigating around the warehouse to find areas of interest, and to drill down the hierarchies. Cerebra also allows concepts to be defined as equivalents, so that 'handset' and 'mobile phone' can be defined as synonyms and a user can see familiar terms.

An ontology represents not only hierarchies of concepts, but also constraints on those concepts. Cerebra uses these constraints and also stores further information about what it is reasonable to say. Combined, these give enough information to create forms and rank results automatically. Unlike other approaches, Cerebra can also show which combinations of concepts are invalid. This gives extra guidance to your users, and can reduce the load on the warehouse as users can answer some queries just using this information.

### **Aiding the Transformation Process**

Relational databases frequently contain concepts at different levels of detail. For example, your helpdesk call-logging system might contain a detailed set of terms for the type of call and another set for the resolution; you might want to combine these two fields in a different way in your warehouse.

Typically, existing systems manage this conversion either through program code or through a mapping table. Neither approach is ideal:

- Program code is difficult to verify and frequently difficult to modify. Once the conversion program is in service, it is often treated as a black box.
- Mapping tables cannot easily express more-specific and more-general cases: for example, that a reported hardware fault fixed over the phone should be reported as category UE, but all other reported hardware faults should be reported as category HF. This kind of hierarchical structure is often implemented using an extra column for relative importance; updates to the mapping table often ignore this column, with unpredictable results.

By contrast, you can represent exactly the desired structure in an ontology and Cerebra can retrieve codes from that ontology. The representation is also something that is easy for you to view, understand, and amend. The ontology is naturally able to represent complex combinations of conditions at various levels of generality, and therefore the resulting mapping is easier to create and easier to maintain as your conversion rules change.

Before data is transferred from a source to a warehouse, it is typically scrubbed to detect incomplete and inconsistent data. Scrubbing processes may be very simple or very sophisticated, but they are typically developed in isolation for each source. It is very difficult to re-use scrubbing code developed for one database in another database so, although it is possible to develop sophisticated approaches, few developers bother. This leads to simple scrubbing that aims at syntactic validation.

In and of themselves, ontologies and inference engines are not suitable data scrubbing tools. In fact, an inference engine is usually a garbage-in and garbage-out type environment. However, the expressiveness of ontologies allows data modelers to specify very detailed constraints with regard to what data is valid in certain models. In effect, it is like applying business rules directly inside the data model rather than using a third-party process.

Ontologies provide formal, sound, declarative statements of what is allowed, and what is not allowed, within the ontology. These constraints limit the terms that may

be expressed in the ontology; any term that violates one or more of these constraints is marked as inconsistent. You can use these constraints to record legal and illegal terms and combinations of terms in databases.

Once the constraints are recorded in a declarative way, Cerebra can make inferences about them. If you are dealing with multiple source databases, and record the constraints of each, Cerebra can find combinations of constraints that are inconsistent and highlight problem areas in the warehouse before they become critical.

Constraints also allow Cerebra to check your data for inconsistency, simply and automatically.

The terms and constraints can be re-used for other purposes: merging with other databases, data modeling, and for assisted search and querying interfaces using Cerebra Forms™. A one-off job provides continuing benefits. Thus, while an inference engine and ontology are not replacements for data quality/scrubbing tools, they can provide a degree of consistency checking during the transformation process that is both simpler and more powerful than other transformation techniques.

### **Different World Views**

Applications tend to use different terms for the same concept (synonyms), and the same term for different concepts (homonyms). More subtly, apparently identical terms might have slight differences in meaning. All of these problems can lead to inaccurate data in a warehouse.

One ontology may be used to encode and translate between terms used in many different relational databases. The transformation for each database uses its own terms in the ontology, and the ontology mediates between the subtle differences in the definitions of those terms. This allows a single, central representation of the precise meaning of terms in all the databases that share the ontology. This representation has several benefits:

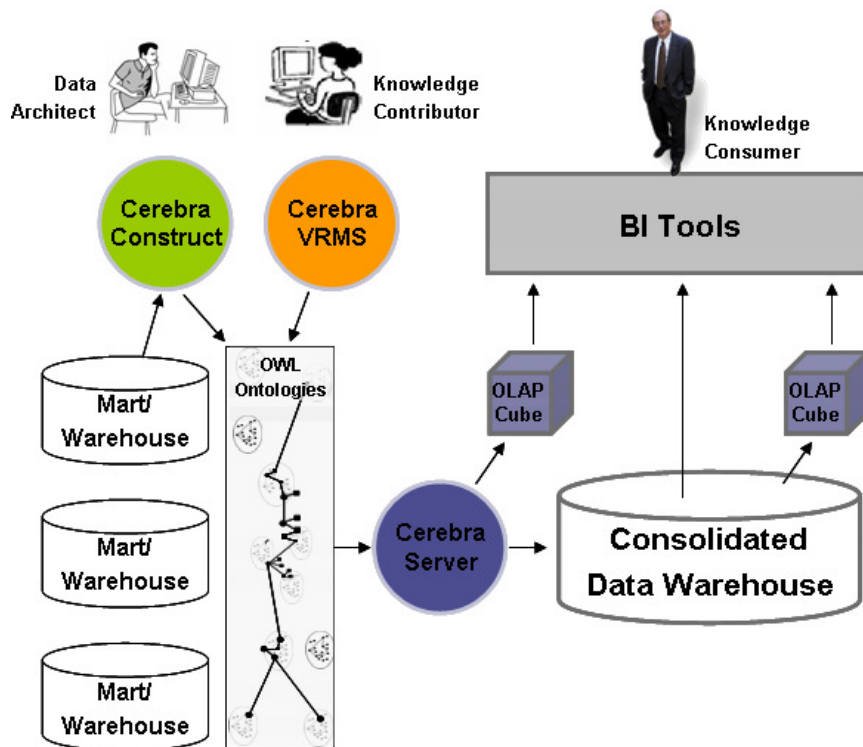
- Once defined, Cerebra can translate between terms in any of your databases that have been mapped to the ontology;
- The fine distinctions between terms within each database can be retained in the ontology, allowing for precise mapping rather than a lowest-common-denominator approach and hence improving the quality of the information in your warehouse;
- The ontology can be extended without affecting existing mappings, allowing you to add further databases to your warehouse without blurring the meaning of terms;
- During the mapping process, the precise meaning of the terms is stated in a formal, declarative manner. This is invaluable documentation for your organization. It often finds and resolves uncertainty in the meaning of the terms that would not have been found using other methods.

### **Deploying the Solution**

The Network Inference Cerebra Suite deploys like many other enterprise software packages. Operating systems such as Windows NT, 2000, and XP are supported in

in addition to many versions of Linux, including Red Hat. Cerebra Server can directly connect to existing data warehouses via ODBC and JDBC connection protocols and issue queries directly against those systems, or alternatively, via a third-party interface such as an adaptor or middleware application that exposes those connection protocols. Cerebra Server also imports ontologies from the file system, a metadata repository, or any Web location via a URI. Stored queries inside the Cerebra Server can be triggered via a Web service or directly over an RMI channel to facilitate warehouse consolidation scenarios.

Ontologies may be designed by expert data architects and mapped directly to the underlying marts/warehouses for consolidation. Alternatively, knowledge contributors (analysts) can also create ontologies via an intuitive web-based interface.



### Summary and Conclusion

Data warehouses are designed to aggregate data and allow decision makers to obtain accurate, complete and up to date information, but a data warehouse is only as good as the data it contains. Most existing warehouses contain a subset of the available data and allow a narrow range of queries on that data, simply because the extra benefit is not considered sufficient to justify the cost of development of a more complete system.

The Cerebra product suite represents complex data, provides flexible and easy-to-use search facilities, and aids the process of loading data into the warehouse. Cerebra can help to reduce the cost and increase the range of information stored in your data warehouse, giving your analysts and decision makers more useful and flexible tools to do their jobs.

# Network Inference at a Glance

## Facts:

- Based in San Diego, CA
- Private, Venture-Backed
- Founded in 2000
- Top-Tier Management

## Customers:

- Fortune 500 Install Base
- Cross-Industry Qualifications
- Federal/Military Deployments
- Achieve significant ROI

## Leadership:

- W3C – Semantic Web
- OASIS - ebXML
- IEEE – Web Services
- DAML – Rule and Service

## The Changing World of Enterprise Data

We are at the cusp of a new IT revolution. Tomorrow's solutions will be adaptive, dynamic, and self-configuring. They will leave behind the legacy of brittle, custom-coded and hard-to maintain systems and interfaces.

The crucial pain point in today's IT departments is the amount of time and resources required to monitor and modify system business rules, integration points, warehouse federation, and data vocabularies in order to keep up with rapid and constant business changes. The closer a business gets to true real-time operations, the more formidable this "speed of change barrier" becomes. Smarter systems are required to sense and respond to events within the enterprise – thereby improving capabilities while simultaneously driving down costs.

Network Inference changes the old rules of IT. Unlike traditional approaches, our technology combines commercially-developed frameworks, such as web services, with academically-developed, semantic technologies grounded in sound science.

## Business Value Proposition

Network Inference helps you build a smarter software infrastructure. This advanced infrastructure software leverages a highly-scalable inference engine alongside a robust query mediation engine – to enable the enterprise infrastructure to draw inferences at runtime. The key advantage of this approach is that enterprise business rules and data vocabularies no longer need to be explicitly defined in brittle code. Now, with Network Inference technology, the implicit enterprise semantics can be queried exactly as the explicit data and rules would be. The bottom line is: smarter infrastructure is good for business.

### Increased Profits

- ▶ Faster response to market demands
- ▶ Better decision making
- ▶ Cheaper adoption of new strategies

### New Capabilities

- ▶ Dynamic information repurposing and reconfiguration
- ▶ Adaptive service-oriented networks
- ▶ Loosely-coupled application connections and information

### Reduced Costs

- ▶ Faster data analysis and dissemination
- ▶ Better quality of service in partner exchanges
- ▶ Cheaper maintenance costs for IT systems integration

## Headquarters

5963 La Place Court  
Suite 300  
Carlsbad, CA 92008  
USA

Tel: 760-476-0650

Fax: 760-476-0648

## Silicon Valley

1600 Adams Drive  
Suite 115  
Menlo Park, CA 94025  
USA

Tel: 650-688-5788

Fax: 650-688-5783

## Toll Free

1-888-NOW-4OWL  
(888-669-4695)

