GAMMASYSTEMS

White Paper:

Data Warehouse Implementation

Streamlined Implementation of Data Warehouses

- Eliminate redundant and repetitive activities
- Automate time-consuming manual efforts
- Bridge project participants and technologies

Table of Contents

Introduction

With the evolution of the data warehousing industry, the concept of data warehousing has gained significant recognition from businesses. In today's business environment, data warehouses are becoming increasingly essential for enterprises, and the use of data warehouses has expanded tremendously to support day-to-day operations, reporting, analytics, and decision making across multiple lines of business.

Software and hardware vendors, recognizing the trends in the market, are coming up with a multitude of products that help businesses to design and implement data warehouses. Product families such as Business Intelligence, Extract, Transform, and Load (ETL), and data warehouse appliances have become a standard part of data warehouse implementations.

While the latest technological innovations indeed help companies to reduce costs of data warehouse projects, many challenges and inefficiencies still exist within data warehouse projects, which the currently available technologies and methodologies do not solve effectively. Despite a high price tag for software and hardware products involved in the implementation, the largest part of data warehouse costs is still attributed to staffing -- an indication of a significant amount of manual work absorbed by data warehouse projects and inability of the current technologies to considerably reduce the extent of manual efforts. High costs and long time-to-market not only impose considerable constraints on the implementation phase of a data warehouse project, but make it hard to justify the project overall.

Considering these observations, a question arises on how implementation of data warehouses can be done in a more efficient manner, resulting in reduced costs, shorter timelines, and lower risks.

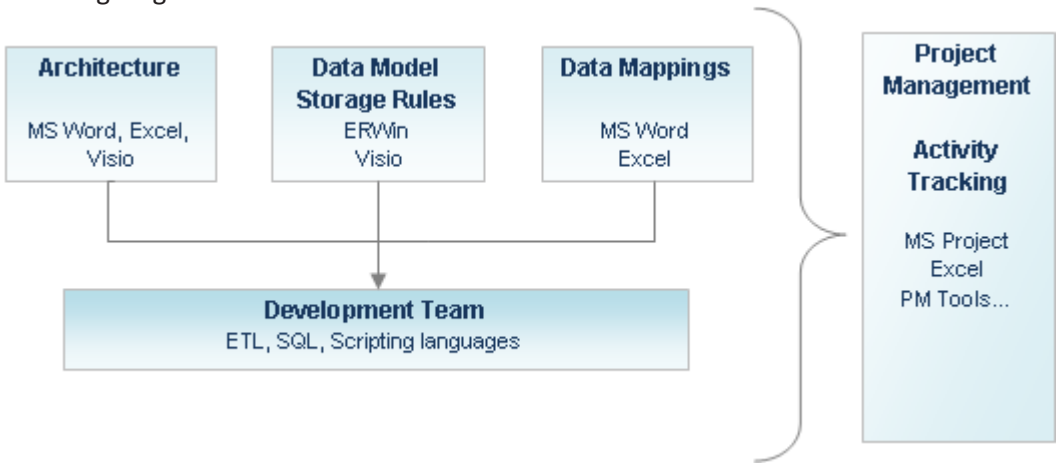
Data Warehouse Studio™, a software package designed specifically to facilitate implementation of data warehouses, allows enterprises to effectively address challenges of data warehouse projects. By providing a single integrated platform to support core data warehouse activities, Data Warehouse Studio™ helps companies to streamline implementation of data warehouses, automate time-consuming manual efforts, and eliminate redundant activities on a project.

The following sections detail challenges presented by data warehouse projects, analyze the nature of the issues common to data warehouses, and explain how Data Warehouse Studio™ facilitates the implementation of data warehouses.

Conventional Data Warehouse Implementation Methodology

Activity Flow

Data warehouse implementation methodology traditionally follows the flow depicted in the following diagram:



Teams of participants, using disparate tools available to them, document their specifications. The specifications then flow to the development team, who needs to absorb these specifications coming from multiple subject domains and produce code that adheres to all inputs. More specifically, the input to the developers includes:

- Architectural specifications – process, data, and system architecture, staging requirements, naming conventions, ETL repository structure, security and encryption requirements.
- Data model and data storage rules – generic RI validation specifications, history management, surrogate key generation methods, audit or housekeeping column behavior, data purging.
- Feed-specific data maps – the business logic depicting transformations to take place during individual feed processing.

Deficiencies of Conventional Approach

The conventional approach depicted above has several costly deficiencies:

- The input specifications to developers are not actionable, meaning that the development team has to develop the code from the scratch, not leveraging the bulk of work contributed during prior phases of the work stream. Because of that, the development effort may take up to 50% of all costs associated with the overall data warehouse implementation.
- Every change in any of the input domains results in iterative recoding and retesting, adding to the costs and timelines of the project. For example, a change in individual feed specification may lead to changing design and code for that feed. A change in

architectural guidelines may affect the entire data warehouse system and lead to redesign of the majority of incoming feeds.

- Errors made during the data analysis phase are discovered only during the coding or testing phases, which results in iterative development cycles.
- The input specifications may be ambiguous on occasion and can be misinterpreted by developers. The errors made by developers are discovered during the testing phase, again resulting in development cycle iterations.
- Coding consistency is hard to achieve, since each developer may have his/her own style of coding and tends to introduce proprietary techniques, possibly deviating from the architectural specifications. Inconsistencies add to the future support costs of the system, as developers inheriting the implementation will need to spend time studying individual processes.
- The traditional approach may impair the evolution of a data warehouse system. As new technologies and features are introduced by relevant vendors, conventional data warehouse implementations are not able to effectively benefit from them, since that may require a re-engineering effort across an entire data warehouse.

Analysis of Conventional Implementation Methodology

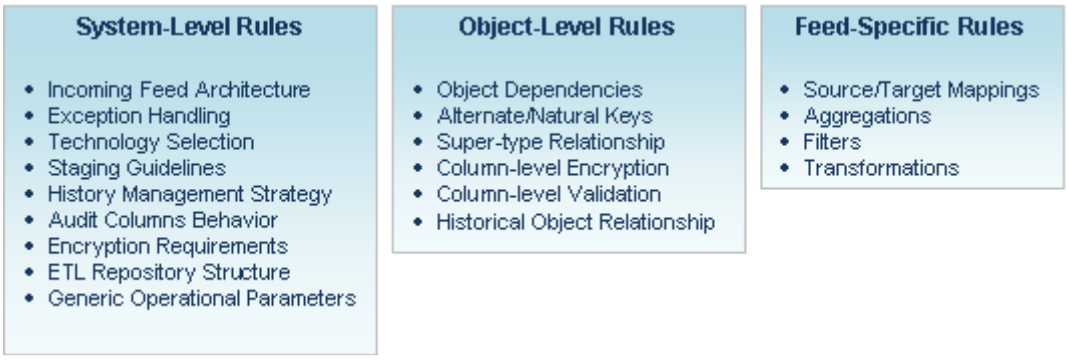
If we look at a data warehouse project not as a set of isolated and incremental efforts, but in its entirety, we start recognizing recurring patterns that can help us better understand how to streamline activities of the project and eliminate redundant phases.

If we look at feed-specific mappings, we find that feeds can share certain architectural details. For example, staging area guidelines, referential integrity (RI) validation strategy, and population of audit columns can be expressed at the warehouse level generically. Naming conventions can share the same patterns. ETL repository can be structured in a uniform manner, following certain generic rules. Sensitive fields can be encrypted using the same encryption algorithm. Same target warehouse tables, populated by different feeds, at a certain point of processing start sharing the same data storage rules, regardless of feed specifics.

Considering these observations, most specifications existing within a data warehouse project can be broken out into three major categories:

- System-level rules
- Object-level rules
- Feed-specific rules

The following chart represents the breakdown of the rule types:



The issue with the traditional implementation methodology is that these rules are often not separated, but instead are repeated in multiple specifications and re-implemented in different modules, leaving room for redundancies and inconsistencies.
Feed-specific rules usually share common patterns. For example, as each data stream reaches the core warehouse tables, it has to go through the same set of steps for each target table:

- Delta Capture
- RI Validation
- Conversion of business keys to internal (surrogate) keys
- Surrogate key generation for new records
- Record versioning

Since in a data warehouse environment multiple feeds often target the same core tables, the steps above are exactly the same, regardless of the feed. For feeds that do not share the same target tables, the processing pattern still persists, which means that the logic for executing these steps can be expressed in generic terms.

Reflecting on these observations, it is apparent that a data warehouse project has a large potential in regards to leveraging concepts, efforts, and code across an entire project, ultimately leading to significantly shorter timelines and lower costs and risks.

Conclusion

Understanding the nature of issues and challenges in data warehouse projects and activity trends, it is clear that a more methodical and systematic approach to building data warehouses is required. However, just having an alternative methodology alone may not be sufficient to reduce costs and timelines and mitigate risks associated with data warehouse projects. A systematic approach would also require an upfront investment into a project, and in turn will push initial downstream deliverables further out.

However, a new technology is available now that effectively solves the challenges posed by data warehouse projects.

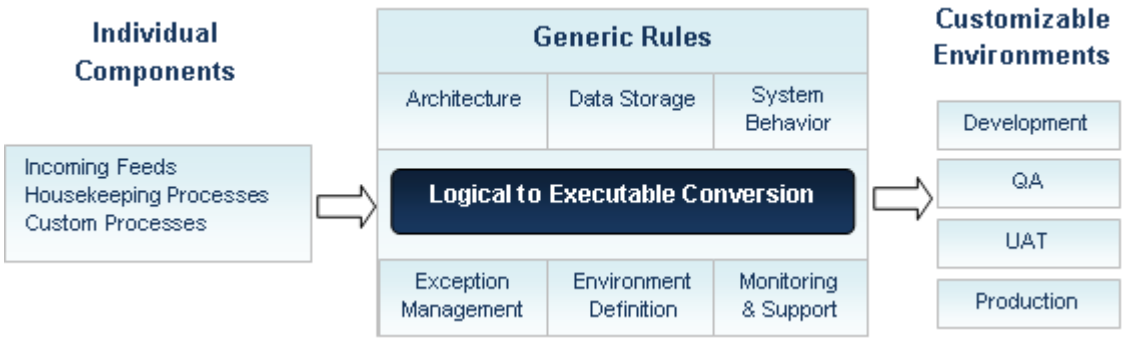
Data Warehouse Studio

Data Warehouse Studio™ is an integrated platform that facilitates implementation and management of data warehouses. Data Warehouse Studio™ focuses specifically on activities that comprise a data warehouse project and provides a level of customization that allows teams to achieve their goals efficiently and quickly. In contrast to other technologies available on the market that solve a specific aspect of a data warehouse implementation, Data Warehouse Studio™ encompasses the major areas of the data warehouse lifecycle and bridges all participating teams and technologies of choice. Data Warehouse Studio™ enables enterprises to streamline data warehouse implementation and keep costs low without compromising on technical and functional aspects of the system.

Methodology

One of the key measures of a successful system implementation is having a well thought-through methodology. In the real world, often the case is that teams do not have an opportunity to make an upfront investment into working out the methodology of building a data warehouse. The result can be equivalent to constructing a building without laying a foundation. With data warehouses, the initial deliverables may be quick, but with time the results of scarce planning start to surface in different forms: re-coding iterations, system instability, degraded performance, inability to expand, limited flexibility, high maintenance costs. Ultimately, the data warehouse would have to be decommissioned or re-engineered.

Data Warehouse Studio™ takes into account the importance of having a methodology and enforces a system-level definition phase before users can proceed to definitions of individual components. In essence, Data Warehouse Studio™ requires that its users cover all the necessary aspects of the implementation to create an "assembly line", through which all the subsequent specifications transition from a logical representation into executable. By enforcing this approach, Data Warehouse Studio™ helps companies accomplish three important objectives: **a)** Establish a methodology that will be followed throughout a lifetime of the warehouse; **b)** Keep the costs and timelines of the methodology definition phase low by providing predefined customizable solutions; **c)** Eliminate the need to re-specify a large amount of rules, applicable on the system level, further down in the project, thus ensuring a high level of consistency across all data warehouse components.
The diagram below depicts the workflow provided by Data Warehouse Studio™.



The methodology offered by Data Warehouse Studio™ allows enterprises to completely eliminate manual coding efforts. Since all the relevant information is collected during the architecture, data modeling, and data mapping phases, there is no need to hand-code the incoming feed processes with Data Warehouse Studio™.

Feature Highlights

Along with offering a methodological foundation, Data Warehouse Studio™ presents a set of features customized for implementing data warehouses:

- Metadata Repository: Stores information on all relevant project components and interfacing technical assets of an enterprise
- Project Specifications UI: Allows users to specify a wide range of system-level settings regarding architecture, administration, exception handling, process behavior, and physical environments
- Data Mapping UI: Allows users to express feed-specific business rules
- Code Generation Module: Based on specifications, the module generates ETL, SQL, and scripting code
- Code Promotion Module: Promotes code between environments and keeps track of deployed components
- Process Management Framework: Customizable component responsible for execution of all processes within a data warehouse.
- Specifications version management: Allows to version data mapping specifications, freeze specifications, and restore previous versions.
- Document Generation: Generates system-level and feed-specific specifications documents in Microsoft Word and HTML.

Technologies Supported by Data Warehouse Studio™

Data Warehouse Studio's flexible architecture allows it to integrate with software packages widely available on the market. Data Warehouse Studio™ integrates with ETL tools such as Informatica's PowerCenter, Talend Open Studio, SnapLogic, and widely used relational database systems such as Oracle, Teradata, Sybase, MS-SQL Server, and MySQL, and can easily be extended to integrate with data warehouse appliance technologies such as Netezza and GreenPlum. Depending on the ETL and RDBMS technologies that the data warehouse project utilizes, Data Warehouse Studio™ helps to take advantage of features specific to selected technologies, thus enabling companies to fully capitalize on their investments into these technologies.