Critical Success Factors for Engineering Data Lakes

This paper describes critical success factors for developing, implementing, and managing data lakes for strategic information management.

Table of Contents

Introduction 1
Critical Success Factors (CSF) 1
CSF: Sponsorship and Involvement 1
CSF: Business Requirements 2
CSF: Enterprise Architecture 3
CSF: Data Lake Architecture and Design 4
CSF: Data Lake Technology 9
CSF: Information Quality 12
CSF: Development Environment 13
Summary 15

Introduction

A data lake is more than an archive for corporate data and more than a new way of accessing corporate information. A data lake is a subject-oriented repository designed for enterprise-wide information access. It provides tools to satisfy the information needs of enterprise managers at all organizational levels — not just for complex data queries, but as a general facility for getting quick, accurate, and often insightful information. A data lake is designed so that its users can recognize the information they want and access that information using simple tools.

One of the principal reasons for developing a data lake is to integrate operational data from various sources into a single and consistent structure that supports analysis and decision-making within the enterprise. Operational (legacy) systems create, update and delete production data that "feed" the data lake.

A data lake is analogous to a physical lake. Operational systems create data “parts” that are loaded into the lake. Some of those parts are summarized into information “components” that are stored in the lake. Data lake users make requests and are delivered information “products” that are created from the stored components and parts.

A data lake is typically a blending of technologies, including relational and multidimensional databases, client/server architecture, extraction/transformation programs, graphical user interfaces, and more.

Data warehousing is one of the hottest industry trends — for good reason. A well-defined and properly implemented data lake can be a valuable competitive tool.

A data lake has its own unique peculiarities and characteristics that make developing a data lake unlike developing just another application. Not every enterprise is able to successfully develop an effective data lake -- in fact there are many more failures than successes.

Critical Success Factors (CSF)

The critical success factors for data lake engineering are:

- Sponsorship and Involvement
- Business Requirements
- Enterprise Architecture
- Data Lake Architecture and Design
- Data Lake Technology
- Information Quality
- Development Environment

CSF: Sponsorship and Involvement

Enterprise executives and managers must sponsor data lake development. Equally important, all potential users must be involved in data lake engineering. Without both management sponsorship and near universal involvement, enterprise-wide data lake projects usually fail.

Management

Enterprise **management** must fully sponsor data lake development and usage. Sponsorship includes ensuring sufficient resources are available. Sponsorship also

means consistent commitment to implementing a data lake that is the single source for corporate measurement and decision support data.

Data lake development and usage often requires significant culture change. This cannot happen without management commitment. Managing internal change, particularly culture change, requires three things: management commitment, universal approval, and appropriate measures and rewards.

Management Commitment: In order for anything to happen in an enterprise, including change, executives and managers must be *consistently* committed to making it happen. Only enterprise leaders can ensure that resources necessary to effect the change are available. Consistent commitment means that the change becomes both an enterprise strategy and an enterprise goal that leaders continuously and obviously support. The visibility of leadership support is a primary factor in achieving universal approval for change.

Universal Approval: Change is successful only when the people involved approve of the change. They understand the need for the change. They believe the change is good for the enterprise and good for them. They agree that the change being undertaken is the right change. Peter Senge, in his book *The Fifth Discipline,* describes the need for universal approval in order to implement systemic change: *"People want change, they don't want to be changed."*

Measures and Rewards: Getting everyone to want change is difficult. It requires a level and degree of communication and cooperation not found in most enterprises. Maintaining universal approval is even more difficult. The best way to achieve and maintain universal approval is to ensure that the process and results of change are measured appropriately and accurately and communicated enterprise-wide. Good results and changed behavior must be rewarded. At the same time, unchanged behavior and poor results should not be rewarded. Employees will not work toward change if they continue to be rewarded for old practices.

Potential Users

All **potential users** of the data lake, even executives, from every organizational unit and level, must be actively involved in data lake design, development, and management. Data lake users will have the most influence on acceptance of the lake, so it is imperative that their needs are addressed. They are also the "owners" and "stewards" of operational data and thus are the best source for subject matter expertise.

CSF: Business Requirements

Developing a data lake without first determining strategic business requirements is a sure recipe for failure. The best source for these requirements is the enterprise **strategic plan** and the **performance measures** identified in the plan. These become the basis for the enterprise architecture as well as the data lake architecture and design (other critical success factors described later in this paper). An enterprise should never undertake system development efforts, particularly engineering a data lake, without first determining its strategic business and information requirements.

Strategic Plan

A **strategic plan** outlines an enterprise's mission and purpose, goals, strategies and performance measures (business requirements). Properly used, a strategic plan is the tool with which effective managers guide their organizations and ensure corporate success.

An enterprise's strategic plan not only provides a guide for effective management; it also provides the guiding force for internal change and the guidelines for responding to external change. Through the strategic planning process, the enterprise defines and documents its purpose, goals, and objectives, along with strategies for achieving them. Included in the process is an assessment of external opportunities and threats as well as an assessment of internal strengths and weaknesses.

The most useful strategic plans are multi-dimensional, incorporating the enterprise's overall plan with the subordinate plans of every enterprise element, and including performance measures for every critical outcome.

Performance Measures

Establishing the right performance measures is the key to successful enterprise management. An enterprise must be able to tell whether progress is being made on its critical goals and whether stakeholder expectations are being met.

The most effective and useful performance measures are cross-functional and are linked to the appropriate strategies, objectives, and performance criteria. Management's targets and thresholds for the measures, often based upon external benchmarks, form the structure for an enterprise performance measurement system.

Performance measurement documentation should include not only the content of reports and queries, but also document the path of the data from source to ultimate information recipient. The combination of all the reports of all the performance measures becomes the basis for a data lake and a Strategic Information System that is truly tailored to the enterprise's requirements.

Executives and managers use the information produced from the data lake to reinforce initiatives, reward behavior and change strategies. Employees use it to adjust operations and respond to strategic needs. Linking timely accurate measures to specific goals and objectives begins to make enterprise management more of a science and less of an art.

CSF: Enterprise Architecture

Linking the **enterprise business architecture (EBA)** (strategic plans, goals, objectives, measures) with its **enterprise information architecture (EIA), enterprise service component architecture (ESA)** and **enterprise technical architecture (ETA)** results in enterprise architecture. This architecture is a logical organization of corporate information requirements, descriptions of application systems that support the enterprise's strategic requirements. It includes the relationships between application systems via shared software components and shared data elements. The enterprise information architecture also establishes guidelines, standards, and operational services that define the enterprise's computing technology environment.

Before an enterprise can define, design, and implement the architecture for its strategic information management systems, including data lake, data mart, decision support, and executive information systems, it must first document the environment in which these systems will be implemented.

Enterprise Information Architecture

The EIA is a fully normalized data model that describes all the data and information necessary to the enterprise. It includes relationships between "business data objects," business rules concerning usage of the data elements, and identification of the "owner" of the data. In addition, it is important for the model to indicate the circumstances (who, when, where, how) for creating, updating, using, and deleting enterprise data. For ease

of use, subsets of the enterprise data architecture model should be established. These subsets, or views, can represent functions, organizations, regions, systems, and any other significant grouping of information.

Enterprise Service Component Architecture

The ESA documents all the information systems in use by the enterprise to create, read, update, and delete enterprise data. In order to be useful, the information systems should be linked to appropriate data elements in the Enterprise Information Architecture. Every system should also be linked to appropriate elements of the enterprise technology architecture.

Enterprise Technical Architecture

This third segment of the Enterprise Architecture documents the enterprise's hardware platforms, operating systems, and telecommunications infrastructure. The ETA is also where guidelines, standards, and operational services that define the enterprise's systems development environment are documented.

More detail concerning "*Enterprise Architecture Engineering*" can be found in the authors' White Paper with that title.

CSF: Data Lake Architecture and Design

The key to success in scaleable data lake development and the single factor that contributes most to data warehousing success is a data lake architecture.

The architecture and design of an enterprise's data lake should reflect the performance measurement and business requirements of the enterprise. Its **data model, structure, components,** and **metadata** should all be based upon internal information requirements -- not specific technologies.

BLUEPRINT FOR A DATA LAKE

"Engineering" a data lake is like engineering a physical lake. Both involve a rigorous development cycle and require the right tools.

A building is constructed using architectural diagrams (blueprints) that clearly depict the building's infrastructure (structural elements, walls, electrical wiring, plumbing, etc.). The best data warehouses are built from architectural models of enterprise infrastructure (policies, goals, measures, critical success factors, etc.).

Blueprints are also used to enlarge a building or make any significant modifications. Without a diagram of the infrastructure, such changes are quite difficult and very costly and can even be dangerous. It is the same with data warehouses. First update the enterprise's data lake architecture model so that it reflects changes (e.g., new performance measures, product lines, or services) and then modify the data lake to support the changed enterprise.

Data lake engineering is easier and less costly when based upon an accurate architectural model of the enterprise. Further, a data lake is easier to use and consistently produces desired outcomes when decision-makers have access to an enterprise architecture (metadata) that accurately reflects enterprise infrastructure.

Data Lake Data Model

A **data model** documents the data elements whose values at any point in time are necessary to tell data lake users how well their enterprise is performing. The data lake model provides a clear and unambiguous definition of every key data entity, describing the way each is used, as well as defining derivation formulas, aggregation categories, and refreshment time periods. The data lake model, linked with the enterprise information architecture, becomes both requirement documentation and a source for communicating the contents of the data lake to its users and developers. Issues that must be addressed in the data model include what legacy data will be used to populate the data lake, how data will be moved from legacy environments to the data lake, and how the legacy data will be integrated or transformed to ensure data quality and integrity in the data lake. *The two most important issues for any data lake are data quality and data access***.**

Data Lake Metadata

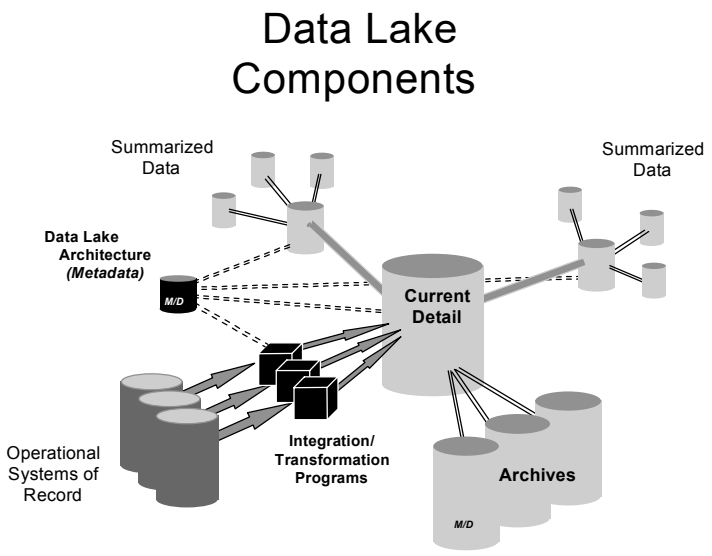
Metadata, or data about data, is the nerve center of a data lake. Metadata is essential to all levels of the data lake, but exists and functions in a different dimension from other lake data. Metadata used to manage and control data lake creation and maintenance resides outside the data lake, often in a digital repository. Metadata for data lake users is part of the data lake itself and is available to control access and analysis of the data lake. To a data lake user, metadata is like a "card catalog" to the subjects contained in the data lake. The two types of data lake metadata are called structural and access.

Structural metadata is used for creation and maintenance of the data lake. It fully describes data lake structure and content. The basic building block of structural metadata is the data lake model that describes its data entities, their characteristics, and how they are related to one another. The way potential data lake users currently use, or intend to use, enterprise measures provides insight into how to best serve them from the data lake; i.e., what data entities to include and how to aggregate detailed data entities. The data lake model provides a means of documenting and identifying structural metadata. This includes both strategic and operational uses of enterprise measures, as well as multi-dimensional summarization. Structural metadata also includes performance metrics for programs and queries so that users and developers know how long programs and queries should run. Data lake performance tuning also uses these metrics.

Access metadata is the dynamic link between the data lake and end-user applications. It generally contains the enterprise measures supported by the data lake and a dictionary of standard terms including user-defined custom names and aliases. Access metadata also includes the location and description of data lake servers, databases, tables, detailed data, and summaries along with descriptions of original data sources and transformations. Access metadata provides rules for drill up, drill down and views across enterprise dimensions and subject hierarchies like products, markets, and customers. Access metadata also allows rules for user-defined custom calculations and queries to be included. In addition, access metadata contains individual, work group, and enterprise security for viewing, changing, and distributing custom calculations, summaries, or other analyses.

Data Lake Components

The data lake architecture also contains descriptions data lake components**: current detail, summarized data,** and **archives** as well as **systems of record** and **integration/transformation programs**.



The heart of a data lake is its **current detail**. It is the place where the bulk of data resides. Current detail comes directly from operational systems and may be stored as raw data or as an aggregation of raw data. Current detail, organized by subject area, represents the entire enterprise, rather than a given application. Current detail is the lowest level of data granularity in the data lake. Every data entity in current detail is a snapshot, at a moment in time, representing the instance when the data are accurate. Current detail is typically maintained for two to five years, but some enterprises may require detail data for significantly longer periods. When initially implemented, a data lake may include current detail more than two years old, but the often questionable quality of older data must be considered and measures taken to ensure its validity. Current detail refreshment occurs as frequently as necessary to support enterprise requirements.

Lightly summarized data are the hallmark of a data lake. All enterprise elements (department, region, function, etc.) do not have the same information requirements, so effective data lake design provides for customized, lightly summarized data for every enterprise element (see Data Mart, below). An enterprise element may have access to both detailed and summarized data, but typically much less than the total stored in current detail.

Highly summarized data are primarily for enterprise executives. Highly summarized data can come from either the lightly summarized data used by enterprise elements or from current detail. Data volume at this level is much less than other levels and represents an eclectic collection supporting a wide variety of needs and interests. In addition to access to highly summarized data, executives also should have the capability of accessing increasing levels of detail through a "drill down" process.

Data lake **archives** contain old data (normally over two years old) of significant, continuing interest and value to the enterprise. There is usually a massive amount of data stored in the data lake archives that has a low incidence of access. Archive data are most often used for forecasting and trend analysis. Although archive data may be stored with the same level of granularity as current detail, it is more likely that archive data are aggregated as they are archived. Archives include not only old data (in raw or summarized form); they also include the *metadata* that describes the old data's characteristics.

A **system of record** is the source of the best or "rightest" data that feed the data lake. The "rightest" data are those which are most timely, complete, accurate, and have the best structural conformance to the data lake. Often the "rightest" data are closest to the source of entry into the production environment. In other cases, a system of record may be one containing already summarized data. Often, “rightest” data is created from diverse sources through a reconciliation process.

Data Lake Structure

A data lake may have any of several **structures**. The structure that best meets the data lake needs of an enterprise is fully dependent upon the enterprise business, data, and access requirements. The basic data lake structures are:

Physical Data Lake - physical database in which all the data for the data lake are stored, along with metadata and processing logic for scrubbing, organizing, packaging and processing the detail data.

Logical Data Lake - also contains metadata including enterprise rules and processing logic for scrubbing, organizing, packaging and processing the data, but does not contain actual data. Instead it contains the information necessary to access the data wherever they reside. This structure is possible *only* when operational systems exactly reflect the enterprise data architecture and system capacities can support both operational and management functions.

Data Mart - subset of an enterprise-wide data lake. Typically it supports an enterprise element (department, region, function, etc.). The organization of data in a data mart reflects the needs of the enterprise element it supports, and may be different from the organization of the enterprise data lake. Specific data elements may be stored redundantly in both the data mart and the data lake. As part of an iterative data lake development process, an enterprise builds a series of physical data marts over time and links them via an enterprise-wide logical data lake or feeds them from a single physical lake.

Both within the Data Lake as a whole and within the individual Data Marts, different groups of users have needs for differing slices of data. For example, users at a branch generally need the “horizontal slice” of data that pertains to their branch (i.e. they need all the data elements - tables and columns - but only the rows pertaining to their branch). Other users need “vertical slices” or a combination of horizontal and vertical slices.

The general approach is to try to make data that are needed by a user group available on a machine that is as close to the users as is feasible -- a Data Mart server. Only that slice of data that is regularly used by the user group should be on their Data Mart server. All other data accessible to the user group should be available on other machines in the network when needed.

The major issues to be addressed in implementing a particular data lake structure involve data distribution and data replication. How much data? How often? Detail or

summarized? Uni-directional or bi-directional update? Data distribution and replication decisions will also have application implications.

Distribution refers to the parceling out of segments of the data to distinct multiple independent computers (or clusters). Replication refers to the copying of portions of the data in one (or more) databases to a second database (often on a different computer) and guaranteeing that whenever the data is changed that all the replicas implement the change in order to stay in synchronization.

The challenges of distribution and replication are not only exist in developing the initial design, but also the ongoing management and maintenance of the overall system. Data Lake architecture design must take into account the following:

Replication: An effective design must consider

- From what database and to what servers is the data moving? Is the data moved one-way or bi-directionally?
- How many replicas will be needed and are they all identical?
- By operational application, how much data - whole databases or selected sets of fields - will be moved throughout the network?
- By application, how time sensitive is the data?
- How is data delivery guaranteed? Who is responsible for the guarantee?
- What should be done if the data to be replicated cannot be delivered due to temporary problems?
- How is data replication tracked?

Network: The network issues that revolve around distribution and replication include

- What are the physical characteristics of the network architecture that connects the data sources to the replicas? Can its bandwidth handle the amount to data to be transferred?
- What is the overall processing speed of each component of the network? How fast can data effectively move between each node?
- What type of replication process will be implemented (synchronous or asynchronous)? Will it require 100% availability of the network? How is the replication process affected if the network is temporarily down? How will failed replication attempts be managed?

Data Lake Application(s): Applications must be designed to be aware of the replicas available.

- Each application at each site must know where to find the data (preferred site)
- If the data is not available at the preferred site, how does the application detect the problem?
- Should the application have the ability to switch to an alternate site from which to retrieve the data?

Scheduling: Efficient scheduling of replication must consider

- What events or conditions will trigger a dynamic data transfer? How much data is transferred during a triggered data move?
- How long will it take to perform the data transfer under various conditions? Can the required time be minimized through better scheduling

- What time zones play an important consideration when moving data?

Change Management: Replication is not a static concept. Ongoing changes must be anticipated in the overall replication strategy.

- What technological changes could impact the operation or performance of the current replication strategy? What investments will improve the strategic business impact of the data?
- What organizational, product, service, regulatory, market, competitive, or other environmental changes could impact the operation or performance of the present replication strategy?
- Does the replication strategy fit today's users' needs?

Distribution and replication of data lake data is primarily a physical architecture design and implementation issue.

CSF: Data Lake Technology

Only after the data lake architecture has been defined should an enterprise begin selecting and implementing its data lake technology. Otherwise, there is a high probability that the technology will not support enterprise requirements. Further, if the enterprise data lake solution is designed for a specific technology it will be difficult, if not impossible, to change technologies as requirements change and as technologies improve and mature. There are more and more technologies available to support enterprise data warehousing. They can be conveniently grouped into **user interfaces, lake engines, hardware platforms, system software, and security.**

User Interface(s)

Data lake users get useful information from the data lake and data marts through **user interfaces**. It is these user interfaces that have the most impact on how effective and useful the data lake will be perceived. Therefore, users must be actively involved in selecting their own interface to the data lake. Two primary criteria for selecting an effective user interface are ease of use and performance. For ease of use, most enterprises turn to graphical user interfaces. For performance, developers must ensure that the hardware/software platform fully supports and is optimized for every chosen user interface. The most important selection criteria for user interfaces are the information needs and the level of computer literacy of potential users who will retrieve the information they need from the data lake. The following data lake user categories are based on levels of literacy and information needs:

Information Systems Challenged - - data lake users who are totally uninvolved with information systems. In management roles they rely on their secretaries or assistants to retrieve information for them. These users need an extremely easy to use and highly graphical interface or standard queries and reports with a limited number of parameters.

Variance Oriented - users who are focused on the variances in numbers over time. These users mainly want a set of standard reports that they can generate or receive periodically so that they can perform their analyses.

Number Crunchers - users who are spreadsheet aficionados. They will take whatever data are available and refine it, re-categorize it and derive their own numbers for analyzing and managing the enterprise. Their needs can best be met by providing a spreadsheet extract output format for any reports or ad hoc queries provided.

Technically Oriented - users who are either already familiar with computers or have sufficient motivation to learn and use everything they can get their hands on. These people want to have complete control over the way they retrieve and format information. They are often business or systems analysts who have moved into an enterprise function. They want to have all of the tools the data lake development staff uses.

Most enterprises have all of these categories of individuals. This makes it advisable to provide each type of data lake user interface.

The final user interface criterion is that it supports the access metadata designed for the data lake. If a user interface is easy to use, allows all potential users to get the information they need in the format they need, and does it in an acceptable amount of time, it is the right interface.

Data Lake Engine(s)

Once the information requirements and metadata for a data lake have been identified and documented, user interfaces have been designed, and the data lake structure has been selected, a **data lake engine** that will support the data lake and all access approaches should be selected. Key issues include capability for loading information into the data lake, implementing access control (security) mechanisms and support for one or more user interface tool sets. The architecture, performance requirements, and overall size of the data lake will determine software requirements. For example, a data lake that includes data marts will require not only relational technology, but also multidimensional access and a client/ server architecture. See table 1.

Feature/Function	Relational	Super-Relational	Multi-Dimension (logical)	Multi-Dimension (physical)	Object-Relational
Normalized structures	✓	✓			✓
Abstract data types					✓
Parallelism	✓				
Multi-dimensional structures		✓	✓	✓	
Drill-down			✓	✓	✓
Rotation			✓	✓	✓
Data-dependent operations					✓

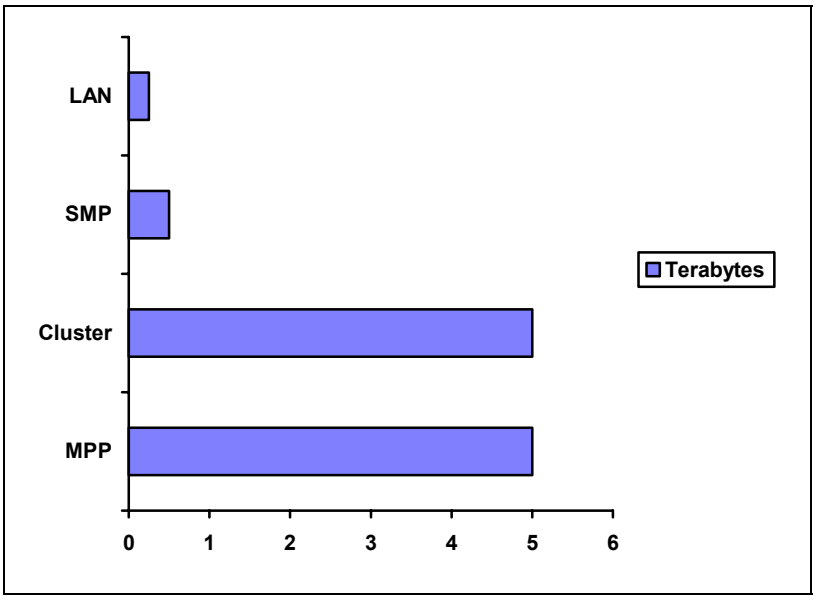
Table 1 - Database Management System Criteria

Hardware Platform(s)

The selection of one or more **hardware platforms** involves answering the following questions: How much data will be in the data lake and how much can the platform economically accommodate? How scaleable is the platform? Is it optimized for data

lake performance? Will the platform support the software selected for the data lake? How many users will simultaneously access the data lake? Will their queries be simple or complex? These are the most important criteria for selecting hardware to support a data lake. In answering these questions it is important to consider all hardware platform characteristics; not just CPU speed and disk capacity, but memory capacity and the input/output system capabilities as well. I/O capacity is often the most critical to overall data lake performance. While increasing the number of servers can usually increase memory and CPU capacity, increasing I/O capacity is not as simple. Nevertheless, it is vital that the hardware platform(s) supporting a data lake have sufficient capacity. This often requires multiple, independent I/O channels or busses.

Data lake capacity planning is not an exact science. Underestimating is the rule rather than the exception. Some experts advise doubling initial estimates of hardware requirements because data lake users and query complexity increases exponentially over the first few months after initial data lake implementation. Even with sufficient initial capacity, it is critical to choose scaleable systems to support inevitable but hard-to-quantify future growth.



Despite the many vagaries of data warehousing and the relative youth of the field, early adopters and vendors agree on a few general rules when estimating server capacity.

Small databases, simple queries -- LAN (local area network) servers, with a single I/O bus are appropriate for data marts where the database is under 5GB.

Medium to large databases, more complex queries -- Response time is faster on SMP (symmetric multiprocessing) systems than it is on uniprocessors, and they tend to be more cost effective the systems where nothing is shared. Large amounts of memory reduce outside seek time during queries, speeding performance when querying large databases. Conventional wisdom suggests that SMP machines begin to exceed capacity between 500GB and the low terabyte range. Good performance for a medium-sized database also requires at least two I/O channels. As the size of the database and complexity of queries grows, more I/O channels are needed to maintain performance.

Very large databases, very complex queries -- very large data warehouses (up to 5 terabytes) require clusters of SMP servers or MPP (massively parallel processor) servers. The platforms with the best performance for very large

databases, excluding MPP's tend to be the ones with a large number of I/O channels.

Huge databases, extremely complex queries -- data warehouses that exceed 10 terabytes may need the processing power and I/O channels provided by mainframe systems.

For these environments…			Choose…		
Requirements	**Users**	**Support**	**Architecture**	**Server**	**DBMS**
Departmental; data analysis	Small; single location	Minimal local; average central	Consolidated; turnkey package	Single processor or SMP	MDDB
Departmental; analysis plus informational	Large; analysts at single location; dispersed informational users	Minimal local; average central	Tiered; detail central; summary local	Cluster SMP central; SP or SMP local	RDBMS central; MDDB local
Enterprise; analysis plus informational	Large; geographically dispersed	Strong central	Centralized	Clustered SMP	Object-relational ; Web support
Departmental; exploratory	Small; few sites	Strong central	Centralized	MPP	RDBMS with parallel support

Table 2 - System Selection Criteria

System Software

Concurrent with hardware selection is the selection of **system software** to support the data lake. The operating systems must support the selected user interfaces, data lake structure and lake engine.

Security

Data lake **security** includes both user access security and physical data security. A data lake is a read-only source of enterprise information; therefore developers need not be concerned with controlling create, update and delete capabilities through *access security*. But, developers do need to address the trade off between protecting a valuable corporate asset against unauthorized access and making the data accessible to anyone within the enterprise who can put it to good use. The best solution is to allow everyone in the enterprise to have access to the enterprise measure definitions and derivations, but only allow access to the underlying detailed data on an approved, need-to-know basis. Developers also need to provide sufficient *data security*, through backup, off-site storage, replication, fault-tolerant and/or redundant hardware, etc., to protect the data from loss due to power failures, equipment malfunction, sabotage, and so on.

CSF: Information Quality

The single most important success factor for data warehousing is the quality of information provided to data lake users. Data in the data lake must be of the highest possible

quality. It must be accurate, relevant, complete, and concise. It must be timely and current. It must be presented in a way that is clear and understandable. A data lake that contains trusted, strategic information, becomes a valuable enterprise resource for decision makers at all organizational levels. If it's users discover that it contains bad data, the data lake will be ignored and will fail. Worse, if it contains bad data, but its users never find out and make decisions based upon the data, it is possible that the enterprise will fail.

Operational Data Quality

The source data that feeds the data lake should be of equal quality. This can be accomplished through practices such as rigorous edits and enforcing a single point of entry for any data element.

Extract, Transform & Load

When source data is questionable or is in many disparate sources, the ETL process may have to be designed to ensure data lake information quality. The components that link operational systems with the data lake are the **integration/transformation programs**. Even the "rightest" operational data cannot usually be copied, as is, into a data lake. Raw operational data are virtually unintelligible to most end users. Additionally, operational data seldom conform to the logical, subject-oriented structure of a data lake. Further, different operational systems represent data differently, use different codes for the same thing, squeeze multiple pieces of information into one field, and more. Operational data can also come from many different physical sources: old mainframe files, non-relational databases, indexed flat files, even proprietary tape and card-based systems. Thus operational data must be cleaned up, edited, and reformatted before being loaded into a data lake.

As operational data items pass from their systems of record to a data lake, integration and transformation programs convert them from application-specific data into enterprise data.

- Reformatting, recalculating, or modifying key structures and other data elements.
- Adding time elements
- Identifying default values
- Supplying logic to choose between multiple data sources
- Summarizing, tallying, and merging data from multiple sources
- Reconciling data from multiple sources

When either operational or data lake environments change, integration and transformation programs must be modified to reflect that change.

CSF: Development Environment

The most ignored critical success factor is the one that can have the greatest impact. In order to consistently design, develop, and implement a data lake, an enterprise must have a development environment that uses best practices and techniques. The elements of this environment include **project teams, methodology,** and **tools.**

Data Lake Project Team(s)

In addition to consistent management commitment and sponsorship and data lake user involvement, there is another critical enterprise culture element. The **teams** that will be actually designing, developing, implementing, and managing the enterprise data lake must have certain characteristics. They must understand the importance of strategic information. They must be able to analyze and document business requirements in business language. They must be dedicated to the data warehousing project. They must have sufficient resources. They must practice effective project management. Every team member must have appropriate skills, knowledge and experience (see below), be sufficiently familiar with the enterprise development methodology, and be able to effectively use the enterprise data lake engineering tool set.

Development Methodology

Enterprises that consistently produce quality information systems rigorously use a full life cycle development **methodology**. Such a methodology is characterized by a sequence of interrelated steps beginning with determining business requirements and resulting in system design, development, and implementation. The Software Engineering Institute, which established the industry-standard, software development capability maturity model (CMM), declares that a methodology is an absolute necessity in order to be an effective software developer.

Having a strategically-driven, customer-focused, information-centric, model-based, disciplined, rigorous, and repeatable methodology is absolutely essential for successful data lake engineering.

Development Tools

A data lake is too complex and too massive to be developed using manual methods. **Development tools** such as modeling tools, repositories and fourth/ fifth generation programming languages are useful for data lake engineering. In addition, there are several Executive Information System (EIS) and Decision Support System (DSS) tools that can help with data lake access. There are also many special purpose data lake tools including middleware and data integration/transformation tools.

Some combination of these tools is necessary to quickly and effectively develop and maintain a data lake. The specific tool set will an enterprise uses will depend upon its data warehousing needs. No matter what tools are used, it is important that the tools work together and that they can be used within the enterprise's chosen technology environment.

Skills and Knowledge

A specialized set of **skills and knowledge** is required to efficiently develop a data lake. They include experience with online analytical processing (OLAP) tools and systems integration; strong technical background with emphasis on operating systems, data bases, decision support tools, user interfaces and client-server; high conceptual level of relational theory; strong communication (speaking and writing) skills; and the ability to interact with everyone in an organization from office workers to the CEO. The necessary skills and knowledge may be acquired by hiring experienced consultants, or by training internal staff. The most effective approach is for consultants to begin development while helping internal staff become skilled so that the enterprise eventually becomes self-sufficient.

Summary

Data lake engineering is not like normal application development. Its scope is broader, its visibility is greater, its user community is larger, and it is more prone to failure.

Before beginning a data lake project, an enterprise should evaluate whether it has adequately addressed the critical success factors for data lake engineering.

Sponsorship & Involvement

Management
Potential Users

Business Requirements

Strategic Plan
Performance Measures

Enterprise Architecture

Enterprise Information
Information Systems
Enterprise Technology

Data Lake Architecture and Design

Data Model
Metadata
Components
Structure

Data Lake Technology

User Interface(s) Lake
Engine(s) Hardware
Platform(s)s System
Software Security

Information Quality

Operational Data Quality
Extract, Transform & Load

Development Environment

Project Teams
Methodology
Development Tools
Skills & Knowledge

Addressing the Critical Success Factors for data lake engineering will help you deliver effective strategic information that exactly meets the needs of your enterprise -- public or private, large or small -- to the right people, in the right place, at the right time, in the right format.

For more information please contact:

Visible Systems Corporation
711 Atlantic Avenue
Boston, MA
contact@visiblesystemscorp.com

Alan Perkins has been a Systems Analyst on the White House staff, Director of the US Army Data Processing School in Germany, Vice President of R&D for a virtual corporation, Vice President of Consulting for Visible Systems Corporation and General Manager of a high-tech consulting firm. He has provided information and enterprise management consulting to numerous companies, associations and government agencies.

Mr. Perkins specializes in Enterprise Architecture Engineering. He helps clients quickly engineer enterprise architectures that are actionable and adaptable. His approach results in architectures that enable and facilitate enterprise initiatives such as Corporate Portals, Enterprise Data Warehouses, Enterprise Application Integration, Software Component Engineering, etc.

The following are papers are available at www.visiblesystemscorp.com:

"Enterprise Architecture Engineering"

"Enterprise Architecture Engineering Critical Success Factors"

"*High-Performance* Enterprise Architecture Engineering – Implementing the Zachman Framework for Enterprise Architecture"

"Enterprise Change Management – An Architected Approach"

"Getting Your Acts Together – An Architected Solution for Government Transformation" "A Strategic Approach to Data lake Engineering"

"Data Lake Architecture – A Blueprint For Success"

"Critical Success Factors for Data Lake Engineering"

"How to Succeed in the 21st Century – Critical Information Management Success Factors"

"XML Metadata Management – Controlling XML Chaos"

"Business Rules Are Meta-Data"

"Enterprise System Modernization – Solving IT's Biggest Problem"

"Strategic Enterprise Application Integration"

"e-Engineering – A Unified Method"

"Enterprise Portal Engineering"

"Quality Software [Component] Engineering"

"Software Engineering Process Improvement"