Talus

A Practical Guide to

Building Data Marts

About Talus

Talus is a knowledge company dedicated to excellence in data warehousing. Talus provides training, design, and deployment services for organizations that demand results from their data warehouse. Talus' data warehouse specialists implement extensible data warehouse solutions with PrimeMover, the industry's only scalable data mart methodology.

Contents

Part One: Introduction

Practical Advice

In 1904 there were no practical guides describing how to build and fly an airplane. Aviation had been born in 1903 when the world learned from the Wright brothers that powered flight was indeed possible. Only a year later, a few, intrepid aviators were flying, but practical knowledge of aviation remained a rare commodity. The information available at that time was almost entirely theoretical, little of it tested, almost none of it to be trusted.

Today, data marts are in a situation not unlike the airplane in 1904 — they are all the rage, but the discussion concerning them is largely theoretical. And much of the theory is suspect!

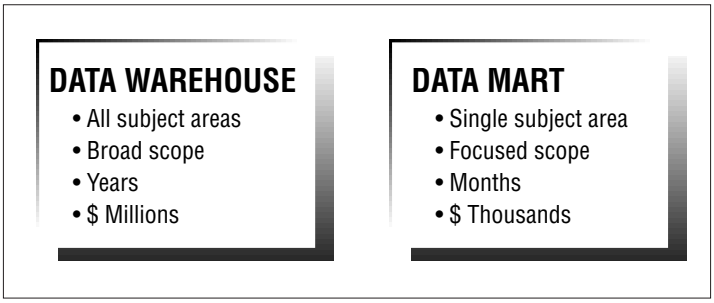
This abundance of data mart theory isn't really a problem unless your boss expects you to implement a data mart in the very near future. If you're under that kind of pressure to deliver a data mart, you're going to need information you can trust to answer your questions. Where is the best place to start? What are the most important design issues? What are the things to avoid? How can you tell if the advice you're getting is based on actual experience rather than untested theory?

This paper can help. It presents PrimeMover — a concise, step-by-step approach to building data marts based on real-world project experience. It is intended as a practical (not theoretical) guide for IT managers planning to build that first data mart.

Data Marts: Decision Support that Works

The data mart is an idea whose time has come. Its focus and simplicity promise the rapid deployment of decision support capability with the quick return on investment demanded by the pace of modern business.

The data mart has evolved from the data warehouse concept. The scope has become highly focused — concentrating on a single subject area rather than the entire enterprise. This is important. In fact, it changes everything. By controlling the scope in this way, the total investment in time and money is reduced dramatically. Risk is mitigated correspondingly as the data mart delivers earned value in a fraction of the time required by the enterprise-scale data warehouse. The following table summarizes the key differences between data warehouses and data marts.



Clearly, a successful data mart strategy can mitigate the risk, limit the expense, and reduce the time required to deliver data warehouse functionality. Because it's scalable (when correctly implemented by the experienced practitioner), the data mart can work well for organizations of any size and

Introduction

level of complexity. It isn't hard to understand why the data mart is now viewed as the most effective mechanism for delivering fast, reliable decision support capability.

While it is true that the data mart greatly reduces the risk associated with building a decision support system, it nevertheless requires considerable skill and knowledge to properly implement one. The craft of building the data mart is still young and experienced practitioners remain few in number. The fastest way for an aspiring data mart builder to acquire the needed skills is to study the lessons learned by experienced professionals.

Using this Guide

The ideas expressed in this paper were developed from lessons learned over many years of building decision support systems in a wide variety of settings. It is designed to serve as a high-level guide for a project manager planning his first data mart. In order to clearly illustrate those factors that every project manager should account for, this paper describes the laws that affect all data mart projects.

The reader will also find a detailed description of PrimeMover. PrimeMover is a data mart development methodology designed specifically to address these laws. By using PrimeMover as a starting point for your plan, you can substantially reduce the risk of project failure.

PrimeMover should be viewed as starting point — not a prescription that guarantees success or demands unthinking conformance. In all likelihood, the individual needs of your organization will dictate how some of the project steps are organized.

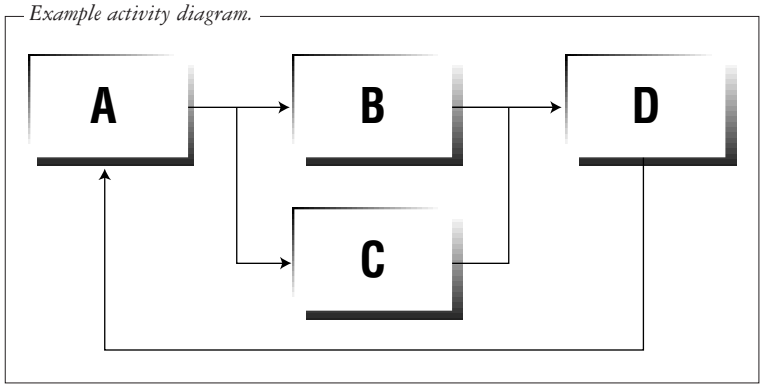
PrimeMover is presented graphically using diagrams that show precedence and dependencies among activities. For example, Figure 1 specifies that:

1. Activities B and C cannot start until Activity A has been completed.
2. Activities B and C can execute simultaneously.
3. Activity D cannot start until both Activities B and C have been completed.
4. After Activity D has been completed, Activity A may be executed again.

The specific tasks that make up each activity are described in detail under the appropriate section heading.

Figure 1

Example activity diagram.



The Immutable Laws of Data Mart Development

It is absolutely vital for the first-time project manager to understand the challenges his project will have to overcome in order to achieve success. To clearly illustrate these challenges, two "immutable laws" of data mart development are described here. Both laws are illustrated using case studies.

These laws have been identified and refined over a period of several years and are a direct result of building many decision support systems. They represent the kind of insight and understanding that only comes with experience.

The effects of these laws are felt on all data mart projects, regardless of scope, complexity, or cost. The key to success is taking these effects into account when planning and executing the project.

The First Law: The Law of Loading Complexity

The most complex part of building a data mart is loading the data.

Loading data is an extraordinarily complex undertaking. The process itself is made up of the following three steps:

1. Extraction from operational (legacy) systems
2. Transformation and aggregation
3. Database loading

The factors that contribute to complexity of data loading are:

- Poor source data quality
- Missing source data
- Lack of source system documentation
- Loading from multiple source systems
- Redundant source data

When one or more of these factors come into play the complexity of data loading is greatly compounded.

The problems associated with data loading are usually many times worse because their complexity is hidden. Only after the process is underway does the magnitude of the undertaking reveal itself.

Given all this complexity, it's not hard to see why the loading process is, by far, the most difficult, time consuming, and riskiest part of building data marts. It is a common cause of project failure and very often, overlooked until it's too late. To illustrate this point, let's look at a case study that shows what happens when you ignore the law of loading complexity.

Case Study: Ignoring the Law of Loading Complexity

Deregulation has come to the telecommunications industry and companies are scrambling to merge, reposition, and move into new markets. Recently, a start-up telecommunications firm merged with one of the larger companies in the industry. This new conglomerate had an urgent need to quickly leverage the customer information from both companies in order to cross-sell services.

The CIO recommended building a data mart because it would allow rapid delivery of the customer information his company needed. The data mart idea was approved, a project team was assembled, and a plan was drawn up. The project was organized in a "waterfall" life cycle consisting of four stages: Analysis, Design, Build, and Deploy. The schedule called for the project to be completed in seven months.

The project manager was unaware of the law of loading complexity. As a result, his plan called for the initial loading of legacy data during the build stage. The build stage was scheduled to occur during the last two months of the project.

The analysis and design stages went according to plan. Business requirements were gathered from users and a star schema was designed that reflected these requirements. With the help of production DBAs, legacy data was mapped to the design. Both the users and the IT organization were encouraged by the rate of progress. Then, in the build stage (four months into the project), the law of loading complexity exerted its influence.

There were multiple mainframe sources of data for the data mart. As a result, the project team had to write COBOL programs for data extraction, C programs for data transformation, and SQL scripts for aggregation. All this custom software development took much longer than planned and the project fell behind schedule.

The first time the data was loaded, it was discovered that the source systems contained large volumes of redundant and incorrect data. More C programs were written to cleanse the data before loading. These changes did not alleviate all the problems. Consequently, the project team was forced to modify and test their software and database design in an iterative fashion.

With each cycle of code, re-design, and reload, the team got closer to solving the load problems. However, their progress was not rapid enough and time eventually ran out. Nine months into the project (which was planned to last seven months) company management decided it could wait no longer for customer information and turned to a third-party telemarketing firm for customer data. The data mart project was canceled and the project team was reassigned.

The problems in this example arose because the law of loading complexity was ignored. Let's look at how the project should have been planned and executed.

Advice for the Telecommunications Company: Load Early

The law of loading complexity affects all data mart projects. Rather than ignoring it, you should plan your project with the firm conviction that moving data from legacy systems to the data mart will be difficult and time-consuming.

Organize the project so that loading starts as early in the life cycle as possible. This means moving the loading tasks from later stages of the project to near the beginning. This is illustrated in Figure 2. By starting the data loading process early in the project, risk identification will occur early and the project will have more time to deal with the problems that arise.

The Second Law: The Law of Design Uncertainty

It is impossible to evaluate any data mart design until legacy data is loaded and shown to users.

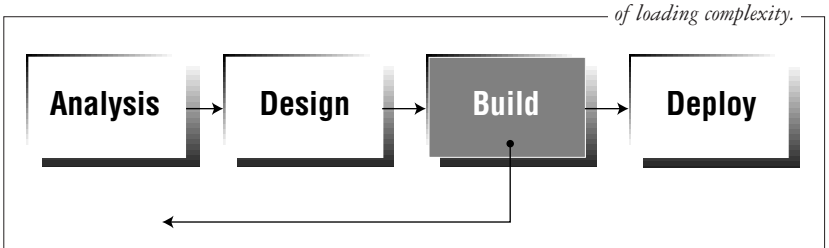
Designing databases for data marts is fundamentally different than designing them for operational systems. Operational database design is concerned principally with capturing business requirements. The design process is considered a success if the design accurately represents these requirements.

For data marts, the design process can only be considered successful if the database design accurately reflects user requirements and can accommodate data from one or more legacy systems. The only way to be sure the design meets both criteria is to load actual source data, and have end users evaluate queries and reports based on that data.

The law of design uncertainty is hard to accept for many data architects familiar with entity — relationship modeling. They have come to believe that models alone could ensure an appropriate database design. For operational databases this assumption is often valid. However, database designs for decision support systems like data marts must strike a balance between user requirements and legacy data. To better illustrate this point, let's examine a case study that shows the consequences of ignoring the law of design uncertainty.

Figure 2

Planning for the law of loading complexity.



Most data mart projects don't start loading data until late in the life cycle. By this point in the project, it is too late to recover from the problems that inevitably arise. The key to minimizing the effects of loading complexity is to start loading as early as possible.

Case Study: Ignoring the Law of Design Uncertainty

The banking industry has undergone changes as fundamental as those sweeping through the telecommunications industry. Competition is the order of the day and banks are aggressively moving into new lines of business. These changes were the catalysts behind a new marketing initiative at a mid-sized regional bank. The

bank initiated a campaign designed to market new services to their existing customers. To execute the campaign, the bank needed a system that would help them analyze the banking habits of these customers.

The IT organization convinced the bank's management that the solution to their needs was a data mart. Like our telecommunications case study, the project was organized in a "waterfall" life cycle consisting of four stages: Analysis, Design, Build, and Deploy. The schedule called for the project to be completed in six months.

The project manager was aware of the law of loading complexity but did not know about the law of design uncertainty. As a result, he planned to load legacy data early in the project. However, the plan did not call for gathering user feedback on the populated design. This error in planning proved to be disastrous.

The analysis and design stages went according to plan. Business requirements were gathered from users. A star schema was designed and loaded with actual legacy data. Although loading problems did occur, they occurred early enough in the life cycle that the team had enough time to react and build successful extraction and load programs. The project team's morale was very high. They delivered the data mart in less than the planned six months. Then, during one of the first end-user training sessions after deployment, the law of design uncertainty showed its effects.

The users had not seen anyone on the data mart project team since team members interviewed them early in the project. During the training class problems became apparent. The users were not seeing the information they needed! In some cases the information was there but is was presented via cryptic codes and abbreviations — not plain English as they had requested. In other cases, the data that users said they wanted was not in the legacy systems at all. There were even several instances of users changing their minds about what data they needed after seeing actual data from the legacy systems.

A meeting between the data mart project manager and user representatives was called immediately after training concluded. The project manager explained that he understood what information the users wanted, but that in many cases the information did not exist in the legacy systems. If the data did exist, very often it was not in the format the users said they wanted. The users were not happy and wanted to know why they were not told about this situation earlier. The project manager tried to explain that his team was too busy overcoming the problems of data loading to inform the users.

Eventually, the users decided that the data mart did not meet their needs and reverted to their old ways of analyzing customer information using PC databases and spreadsheets. The data mart was still available, but it was rarely used.

The problems in this example were caused by ignoring the law of design uncertainty. Although the law of loading complexity was handled well, a successful data mart project must account for both laws. Let's look at how the project should have dealt with the law of design uncertainty.

Advice for the Bank: Obtain User Feedback

The law of design uncertainty affects all projects. Under no circumstances should you ignore it and hope its effects don't show up in your project — because they will.

The only way to be confident of your data mart design is to create a "feedback cycle" early in the project. This feedback cycle should involve end-users evaluating prototypes built with live data in an iterative process that continues through much of the life cycle.

This feedback mechanism should be created by including loading and evaluation steps early in the project as shown in Figure 3. In this way, the project team can monitor user acceptance and discover design flaws early in the project.

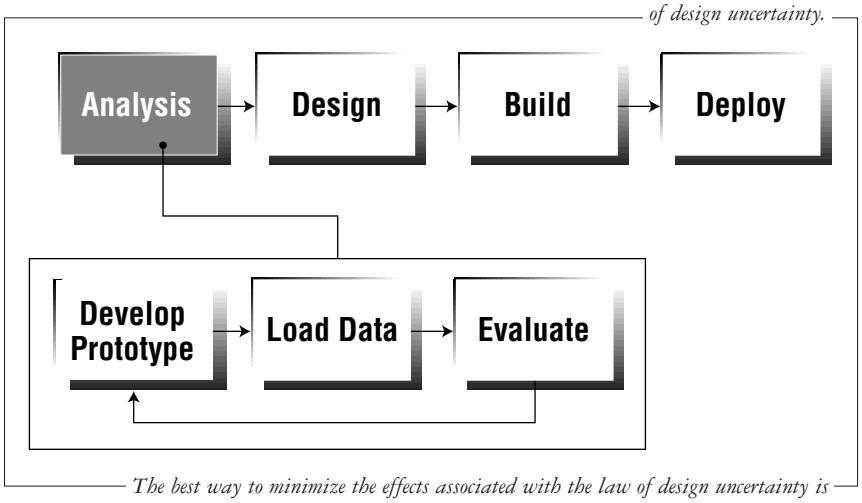
Applying this Advice in Your Project

At this point, hopefully you're convinced that the key to successfully building data marts lies in understanding and planning for the immutable laws of data mart development.

Don't stop reading here! The following section describes a specific approach for addressing these laws. It's called PrimeMover and it's based on real-world experience and first-hand knowledge of loading complexity and data mart design. Unlike many data mart approaches that gloss over the hard parts, PrimeMover was developed specifically to address head-on those issues that cause most data mart projects to fail.

Figure 3

Planning for the law of design uncertainty.



The best way to minimize the effects associated with the law of design uncertainty is to create a user feedback cycle early in the project.

Part Three: Application

PrimeMover Overview

Single Subject Area

PrimeMover is a scalable data mart development methodology that reduces the risks, cost, and time associated with building decision support systems. By focusing on a single subject area, PrimeMover delivers a data mart that answers critical business questions in just five to seven months. PrimeMover is designed for implementation by a highly-skilled team of 4-8 people in three stages — Planning, Development, and Deployment. Figure 4 shows the three stages that make up the PrimeMover life cycle.

Early Data Load and Discovery

PrimeMover is unique in its approach because it confronts the hardest part of building a data mart — loading operational data — early in the project. Developers create a first-cut star schema and immediately put this design to the test by loading operational data. Business users, with the aid of ad hoc query tools, assist the project team in reviewing the data mart design and quality of the source data. The design and load scripts are adjusted iteratively until requirements are met and the project team can confidently deliver a production data mart.

Scalable Design

PrimeMover can be used to build a single data mart, a comprehensive enterprise data warehouse, or something in between. In the case of the enterprise option, PrimeMover builds incrementally, subject-area by subject-area, rather than attempting to build the entire data warehouse in one large undertaking. Each functional data mart builds upon its successors, logically integrating by conforming dimensions, until the enterprise's overall decision-support needs are met.

Enabling Technology

PrimeMover fully exploits the latest generation of development and analysis technology. Automated extraction/transformation/loading tools enable early data discovery, while query/reporting tools optimally deliver the benefits of OLAP to decision makers.

PrimeMover Requisites

PrimeMover's success lies not only in its incremental approach and willingness to tackle the most difficult aspects of building data marts, but also in its sound project management structure. PrimeMover's Planning Stage includes techniques for negotiating data mart features and project parameters. Scope is clearly defined to avert the problems associated with overly broad data warehouse projects. To ensure that developers and users meet the demanding schedule and stay within budget, disciplined change management procedures are instituted. Initial success is best achieved by starting with the subject area that has the most

Figure 4

The PrimeMover life cycle

accurate data source. This is recommended because a project becomes jeopardized immediately if the source data are not of sufficient quality. Due to its demanding schedule, once a PrimeMover project is underway, resources cannot be diverted to the time-consuming process of data cleansing. External to the project, other resources can be applied to data cleansing in preparation for the next data mart.

Advantages of Using PrimeMover

Although PrimeMover requires disciplined project management, the payoffs are high. PrimeMover's technically sound approach to extraction, transformation and loading eliminates one of the most frequent causes of project failure — loading complexity. PrimeMover's iterative design approach with live data verification ensures that the final product meets end users' expectations. Providing decision-makers with obvious signs of progress early in the project establishes team credibility and provides an immediate record of success. Rather than waiting years for business questions to be answered, analysts and decision makers can have their analytic requirements met right away. This record of success garners support for follow-on data marts, as business people begin to view the data mart as a practical, cost-effective decision-support solution.

PrimeMover Details

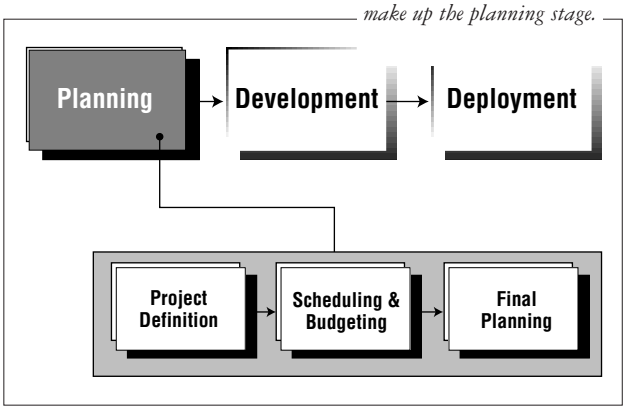
The Planning Stage

Description

The planning stage establishes the scope and business objectives for the data mart. Agreement is reached on the data mart subject area and extent of the star schema. The plan clearly establishes a realistic project schedule, budget, staffing resources, assumptions, and constraints. A risk mitigation plan anticipates potential risks and outlines specifies measures to ensure project success. Figure 5 shows the steps that occur during the planning stage of the PrimeMover life cycle.

Figure 5

The detailed steps that make up the planning stage.



Deliverables

There are two major deliverables created in the planning stage: the project overview document and the project plan. Each deliverable is outlined below:

Project Overview Document

- Clearly defined scope
- Overall schedule of activities for the project
- Project organization
- Clearly defined project control procedures

Project Plan

- Detailed schedule of activities
- Assignment of activities
- Budget

Planning Steps

Project Definition

Determine the scope of the data mart, including the business subject area, number of star schemas, and key business objectives. Identify staff requirements, project assumptions, and time and resource constraints. Identify risks and corresponding mitigation measures.

Scheduling and Budgeting

Identify the level of effort necessary for the data mart, and develop a budget accordingly. Develop a detailed schedule of activities for each development step. Document all schedule and budget assumptions.

Final Planning

Reach an agreement on the project scope, schedule, and budget. Assign staff resources by name and responsibilities. Establish lines of communication, develop procedures for project changes, and deliver the project plan.

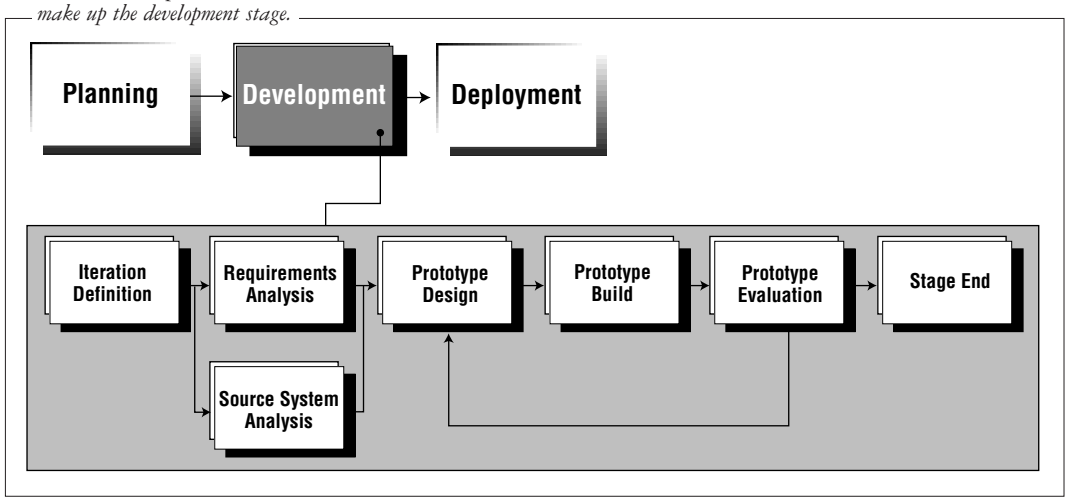
The Development Stage

Description

Using an iterative design and build approach, the development stage produces a working prototype loaded with actual source data. Key business objectives are defined by interviewing managers, analysts, and DBAs for requirements. A prototype is built and

Figure 6

The detailed steps that make up the development stage.



loaded with legacy data. Users query the system to verify that the data mart meets their requirements. Iteration of the design/build processes ensures that problems associated with loading complexity are discovered early and the design is improved to balance requirements with source data availability and quality. The prototype is developed to a level sufficient for users and developers to agree on the final data mart. The development stage is 12 to 16 weeks in duration. Figure 6 shows the steps that occur during the development stage of the PrimeMover life cycle.

Deliverables

The development stage creates two principal deliverables: the data mart prototype and a revised plan for executing the deployment stage. Each deliverable is outlined below:

Data Mart Prototype

- Database loaded with live operational data
- Extract and load scripts/sessions
- Installation and configuration of the query environment
- Design documents including star schema and metadata

Revised Project Plan for Deployment Stage

- Detailed schedule of activities
- Assignment of activities
- Budget
- Agreed-upon final scope

Development Steps

Iteration Definition

Determine the number of data mart iterations and the dependencies between iterations. Define prototype exit criteria to avoid runaway prototyping. Assign objectives, standards, and milestones for each iteration. Determine what data will be analyzed and loaded for each iteration and how the results will be presented to users.

Requirements Analysis

Interview managers and business analysts in the assigned subject area to determine their requirements for decision support. Record the business questions that the data mart must answer. Define the overall technical architecture and performance requirements.

Source System Analysis

Interview database administrators and other individuals with knowledge of source system data and structures. Analyze the data to determine quality problems and identify areas of loading complexity. Identify areas where data are missing to determine if the source data can support the business requirements. Record detailed metadata about the sources, quality of data, and business rules.

Prototype Design

From the understanding of the business requirements and based upon source data availability, design a star schema for the data mart subject area, including any agreed-upon aggregates. Develop transformation logic for mapping source data into the target data mart schema. Design the end-user query environment including any required canned reports. Verify the design iteratively with managers, analysts, and DBAs.

Prototype Build

Extract data from the source system. Load actual production data into the target tables to determine its conformance with the design and requirements. Uncover any additional source data deficiencies by iteratively loading data and ensuring the correct results are achieved. Develop the end-user query environment and reports. Record detailed metadata about the source and target data elements, source to target table mappings, data cleansing steps applied to each field, etc.

Prototype Evaluation

Present the prototype to business analysts, managers, and DBAs in interactive sessions using the query tool to present the actual loaded data. Validate that the data mapping and the load is correct by ensuring that the data loaded meet the business objectives and are correct. Determine how well the loaded data supports the data mart design. Identify areas where the design does not match user requirements and expectations (within subject area scope) and determine how to change the design for the next prototype iteration. Review the exit criteria for the iteration and confirm the need to conduct another iteration. Reconfirm scope to ensure the project is on track. Assess the status of identified risks or new risks and evaluate risk mitigation measures.

Stage End

Assess whether the development stage has met the project objectives and determine next stage activities. Eval-uate the level of effort necessary for finalizing the prototype into an operational data mart. Identify any corrective measures that need to be taken and changes to the project plan that may be necessary.

Deployment Stage

Description

During the deployment stage, one engineers and deploys the data mart by building on the prototype. Data loading, change data capture, slowly changing dimensions, and quality assurance procedures are implemented. Users are trained to successfully navigate the data mart using the query tool. By the time the data mart is moved into the production environment and becomes the responsibility of the production support staff, it has been tuned for optimal performance and thoroughly tested. Procedures for monitoring usage and collecting query profile data are established. Figure 7 shows the steps that occur during the deployment stage of the PrimeMover life cycle.

Deliverables

The deployment stage creates two principal deliverables: the data mart prototype and a revised plan for executing the deployment stage. Each deliverable is outlined below:

Production Data Mart

- Fully populated database optimized for performance
- Production extract and load scripts/sessions
- Production end user query environment
- Production reports

Documentation

- System documentation
- User documentation
- Final design documents including star schema and metadata

Initial training

- Users
- Support staff

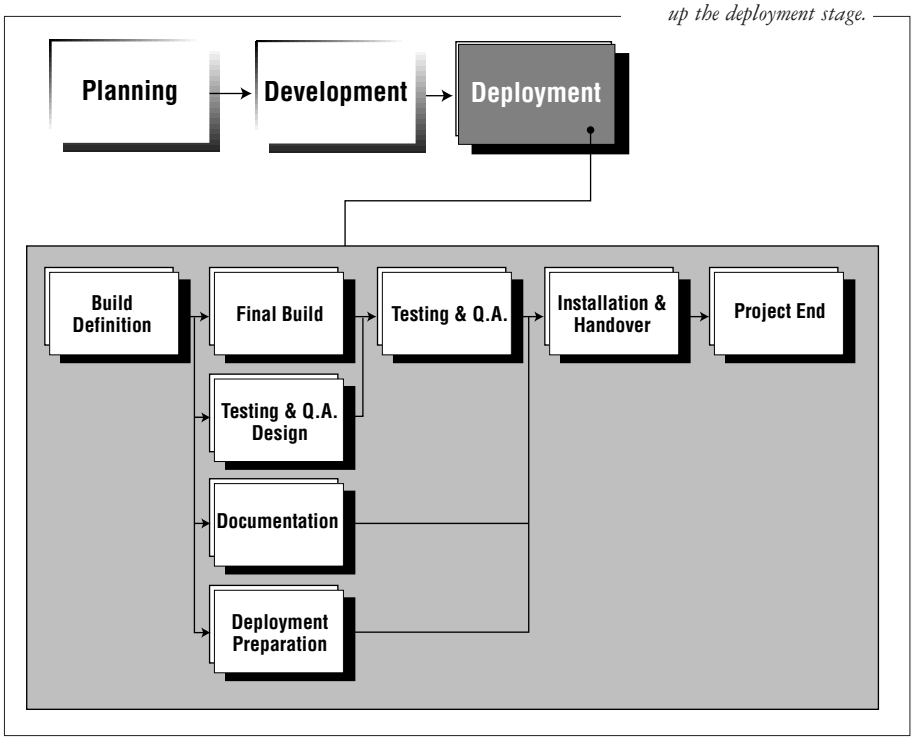
Deployment Steps

Build Definition

Building on the prototype, determine the activities necessary for finalizing the prototype into an operational data mart. Develop a strategy for identifying and handling slowly changing dimensions. Specify detailed steps and timelines. Assign resources and staff roles and responsibilities.

Figure 7

The detailed steps that make up the deployment stage.



Final Build

Finalize the prototype design and the extract, transformation and load scripts. Implement the strategy for identifying and handling slowly changing dimensions. Establish change capture procedures and schedule load sessions. Finalize the query tool environment and reports. Conduct performance optimization and tuning of the data mart components. Build indices

to improve query performance.

Testing and Quality Assurance Preparation
Using the business questions identified during requirements analysis, develop and run test cases against the loaded data to ensure that the data mart answers business questions correctly. Ensure the data mart is aggregating data correctly. Establish procedures for quality assuring each data mart load.

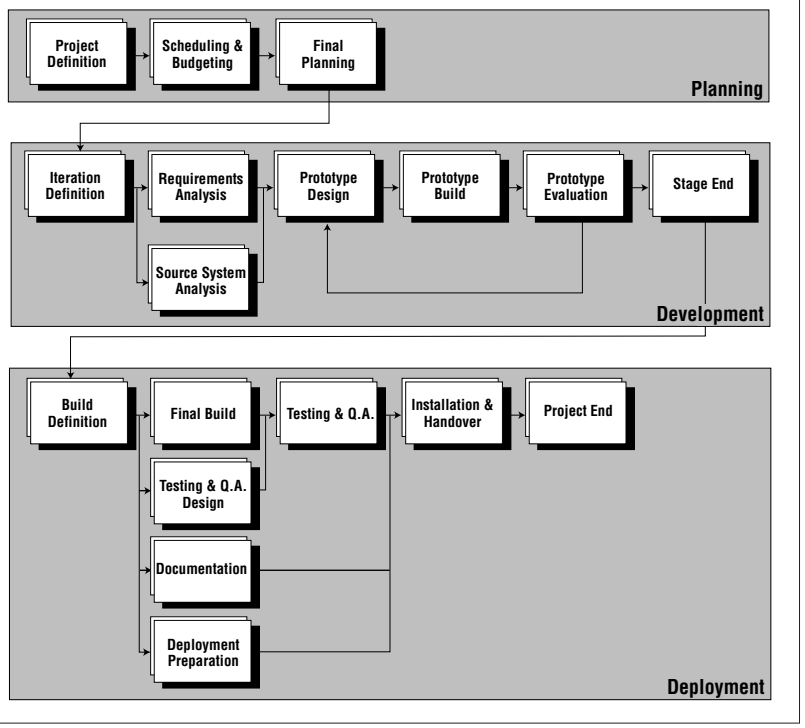
Documentation
Using the development metadata repository, prepare system documentation. Write user documentation including information about the data mart subject area and objects available for query. Record end-user metadata in the data mart repository.

Deployment Preparation
Determine procedures for system administration and management. Establish a data mart help-desk and end-user support procedures. Create backup and restoration plans. Develop a training plan and conduct training.

Testing and QA
Perform acceptance testing, evaluate the results, and take corrective actions. Perform quality assurance "spot checks" on the data loaded into the data mart and the extract and load programs/scripts. Formally test all pre-defined reports. Perform load testing on the database.

Figure 8

All the detailed steps that make up the entire PrimeMover life cycle.



Installation and Hand Over
Load the data mart and implement the schedule for regular loads. Install the query tool server, repository and all clients. Conduct a final check to ensure all hardware and software are correctly installed and that the system is fully functional. Hand over system responsibilities from the development team to the data mart support staff. Initiate live operations of the data mart.

Project End
Evaluate the development process and identify process improvements based on lessons learned. Choose candidates for the next data mart subject area and ascertain issues associated with getting new data marts to conform with existing data marts. Estimate the time frame and high-level goals for initiating the next data mart development cycle. Figure 8 provides a detailed overview of the entire PrimeMover process.

Conclusion

This guide has been mostly about technology. While technology is important, experience teaches us that the human factor is equally critical. Unfortunately, it is not possible to reduce the human factor to words and pictures. What is possible, and what this guide has attempted to provide, is some assistance with the technical aspects of building data marts. Hopefully, we have also sparked some enthusiasm for the process. After all, it is enthusiasm, hard work, and perseverance that led the Wright brothers to overcome all technical obstacles and fly.

Talus

3601 Eisenhower Avenue
Suite 130
Alexandria, Virginia 22304
Phone: 1.800.214.5641
Fax: 703.713.8961