Week 4 Notes

Building the Data Warehouse: the Kimball method

Kimball proposes a traditional information-system life cycle approach that is driven by business requirements and partitions the life of the data warehouse into several stages. The stages are modeled as discrete components that work together by passing data from one to another according to well-defined paths. Kimball's DW/BI life cycle is illustrated in Figure 1.

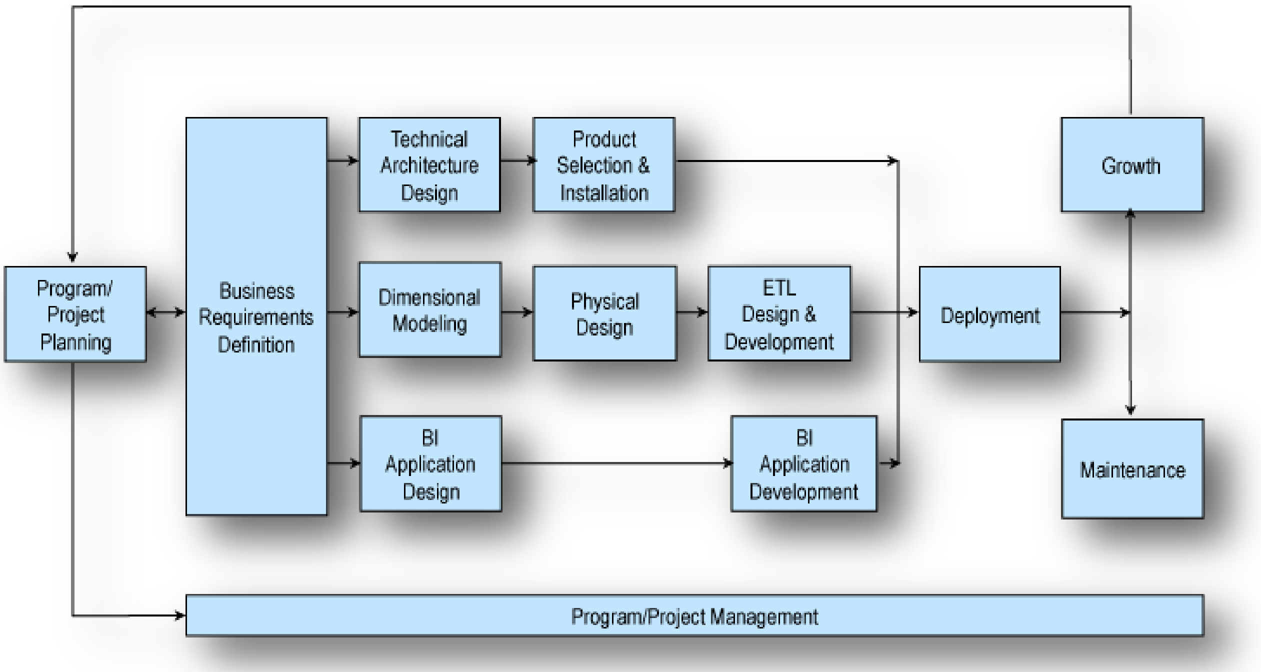
Program/ Project Planning
Business Requirements Definition
Technical Architecture Design
Product Selection & Installation
Dimensional Modeling
Physical Design
ETL Design & Development
BI Application Design
BI Application Development
Deployment
Growth
Maintenance
Program/Project Management

Figure 1 Kimball's data warehouse lifecycle

The 13 blocks in Figure 1 can be grouped into the four life stages of an information system: **initiation**, **development**, **implementation**, and **operation and maintenance**. The Program/Project planning, Program/Project Management, and the Business Requirements Definition blocks make up the initiation stage. The Deployment block represents the implementation stage. The Growth and Maintenance blocks make up the operation and maintenance stage. The remaining 7 blocks that appear at the center of Figure 1 make up the development stage.

The bidirectional arrow between Program/Project planning and Business Requirements Definition shows that these two components exchange data with each other. Program/project planning starts with the gathering of business requirements, thus some may consider placing the latter block inside the former. However, placing the business requirements in a separate block emphasizes that (1) The business requirement definition keeps evolving over time (which drives the evolution of the data warehouse itself, which is a key insight derived from experiences of many DW developers); and (2) Business requirements have a direct impact on the three parallel tracks (**technology**, **data**, and **BI apps**) that are fed by this block. These three tracks should be developed in parallel and asynchronously, though their vertical alignment in the diagram shows implied dependencies between them.

Business Requirement Definition (Chapter 3) is the very first step in Kimball's DW/BI life cycle. The analysts must understand and translate the key business driving factors into design specifications.

The Technology Track (Chapters 4 and 5) starts with **Technical Architecture Design**, driven by both business requirements and technical environment, and continues with **Product Selection** and **Installation**. In this latter step we evaluate and select hardware platforms, DBMS, ETL tools, querying and reporting tools.

The Data Track starts with **Dimensional Modeling** (Chapters 6 and 7). When the DW/BI team gathers the business requirements, they create a preliminary **enterprise data warehouse bus matrix**, which contains the key business processes and their dimensionality. This matrix is an architectural blueprint to ensure that DW/BI data can be integrated and expanded over time. Following detailed row-by-row analysis of the matrix, data modelers identify the numeric facts, the dimensions, and their respective granularities. This is followed by the **Physical Design** (Chapter 8), where the team sets up the DB and access policies. **ETL Design** and **Development** (Chapters 9 and 10) remains one of the biggest challenges in DW/BI development. It is estimated that 70% of the risk and effort come from this step.

The BI Application Track (Chapters 11 and 12) starts with the design phase, which occurs as soon as the business requirements have been identified, and continues with the development phase. **Navigation interfaces** and the **navigation portal** are important outcomes of this track.

These three tracks converge in the **Deployment** stage (Chapter 13). The team must make sure that they fit together and also synchronize them with the **training** and **support** infrastructure. In the **Maintenance** phase (Chapter 13), the team monitors the DW/BI usage, tunes the performance, maintains the indexes, and performs backups. Finally, the **Growth** phase (Chapter 14) reminds us that change is a sign of success, not failure, in a DW/BI system. The life cycle keeps looping during the existence of the DW/BI, as indicated by the arrow pointing back from Growth to Program/Project planning.

Program and Project Planning and Management

The blocks in the life cycle diagram cannot be developed in total separation. The **program manager**, like an orchestra conductor, must make sure that each instrument plays the right chords at the right times. This section and this week's required reading provide a blueprint for his task.

In a Kimball life cycle, a **project** is a single iteration of the entire cycle, from the beginning to the end. A project starts and ends at specific moments in time. A **program** is a collection of multiple projects that are coordinated together in terms of resources, infrastructures, timelines and communications, to obtain a whole that is greater than its parts. A program rarely ends at a specific moment. Some organizations start with a project and later discover the need for a program; others have a program planned from the beginning, and then partition it into individual projects.

A project is usually broken down into smaller pieces called **subprojects**, which can be defined according to criteria such as project's phases, shared methods, or technologies. For example, the project that builds the initial DW that integrates data from 10 different transactional systems and supports 5 different business units can be broken down into sub-projects in a few different ways:

- 5 projects, each building the reporting capabilities of a business unit;
- a sub-project for each component: infrastructure, DB, ETL, reporting;
- 10 sub-projects, each responsible with the ETL from one data source.

The first major stage in **Program/Project Planning** block is **Project Definition**. This starts with the assessment of the organization readiness for a DW/BI, in which finding **strong business sponsors** is paramount. The best way of finding a sponsor is to perform a high-level **business requirement analysis**. Proof-of-concept implementations should be used with caution to demonstrate the potential business impact to potential business sponsors, as they may create false expectations regarding the duration of the actual project. Dashboards should also be avoided at the beginning of the project; it is better to deliver detailed data for one business process at a time.

The next planning step is the production of a document called **Project Charter**, or Scope Charter (Figure 2-1, pp 26). **Financial investments** and **costs** are also estimated during project definition. Rapid application development (**agile**) methods are good for BI applications. They are not as good for the ETL because of the lengthier and more complex nature of the task.

Project Definition is followed by Project Planning, which consists of four major tasks: **naming** the project, **staffing** it, producing a document called **Project Plan** (Figure 2-2, pp 42), and a document called **Communication Plan** (Figure 2-3, pp 44).

The Project Management block starts with a **Kickoff Meeting** (sample agenda in Figure 2-4, pp 48) and continues with regular **Project Status Meetings** (sample agenda in Figure 2-5, pp 49). Additional important project management steps: **weekly update** and **circulate** the **Project Plan**, manage the scope and the expectations by **implementing** the **Communication Plan** (the project manager may need to remind business people that every new major data source adds 6 months to the development time), **track** the **issues** and **changes** (i.e., larger issues), and recognize and deal with **trouble signs**.

The BLUEPRINT FOR ACTION box (pp 58-60) summarizes some key points of project planning and management. Kimball's life cycle is a DW-adaptation of general project management practices. Both Inmon and Kimball's approaches are flexible and can be adapted to your DW development needs.

Collecting the Business Requirements

Introduction

A requirement-driven **System Development Life Cycle** (**SDLC**) starts by collecting the business requirements. The Kimball DW/BI life cycle is no exception: this is the first step in the Program/Project Planning block. We also mentioned that the requirements keep evolving in time, and this **evolution** drives a similar evolution in the DW information system. This lecture focuses on the process of collecting the business requirements for a DW/BI.

In a nutshell, this process consists of designated members of the DW/BI development team talking to the DW business users to understand their decision-making needs, and then translating these needs into design specifications for the DW/BI system. The team members also talk to IT personnel. Together, they perform a data audit that will help them understand the actual data availability and quality with respect to the business requirements that are being gathered.

As a side note, one key difference between Kimball and Inmon's approaches to building a DW is mostly a matter of focus, or priority: Inmon places the data availability and quality in the driving seat, and the business requirements in the passenger seat; Kimball switches seats.

In the following sections, we will introduce the members of the team that collects the requirements for a data warehouse, then we will list the typical requirements that are collected, and then we will show how the team collects these requirements.

The Team

The team that collects business requirements is led by a **Business Analyst**. He is assisted by the **Program/project Manager** and the **Business Project Lead**. For requirement analysis, they are assisted by a **data modeler,** and a **BI App Architect** or developer. The team may also include a Storage Area Network (SAN) Architect, a Database Administrator (DBA), and an ETL Architect. Individual interview teams consist of a **lead interviewer**, a **scribe**, and an **observer**.

The Requirements

The team collects a variety of requirements. For example, the program/project manager asks about key performance indicators (KPIs), response-time requirements, storage requirements, software requirements, support requirements such as Service Level Agreements (SLAs), contents, format, and frequency of status reports, format of the interviews, internal and external resources and their availability, project milestones, deliverables, and deadlines, project dependencies, change management and release management process, testing methodologies, issue and risk escalation methodologies.

The data modeler needs to know reporting requirements, query patterns, and the typical ways in which users "slice and dice" the information (select rows and columns). He groups this information into business entities and attributes, relationships between them (including hierarchical ones), business measures used for analysis, the facts required to compute these measures, calculations required to create new facts, and if and how facts are aggregated. The data modeler will use this information to create the necessary indexes, constraints, relationships and database objects. For example, if users want reports at the store level, it would be inefficient to aggregate the sales at this level every time a report is generated. It's better to pre-aggregate (summarize) and store the data at store level for quick response times. The ability of a data warehouse to store summary data comes in handy here.

The SAN Architect must assess the size of the DW and its various components: schemas, instances, environments (development, test, staging, production). The DBA gathers requirements on DB size, type of DB constructs (ODS, DW, DM), backup and recovery requirements, query performance and storage needs, DB roles and privileges, users, data access methods. The ETL Architect needs to learn about the type of data sources (relational DBs, flat files, legacy systems), the best time to extract and load information from these sources to the respective target areas, the type of extraction (data push or pull), type of loading mechanism (load, truncate, insert/update). Check out the Task List on pp 107 for a summary of team roles and tasks.

The Process of Requirement Gathering

The process of collecting business requirements has three major phases: a **preparation** phase, an interactive **face-to-face** phase, and a final **documentation-writing** phase. The BLUEPRINT FOR ACTION box (pp 104-106) summarizes some key elements of the business requirement collection process.

Interview Preparation

The preparation phase consists of the following steps:

- build the interview team;
- research the organization: read the annual report; minutes of the latest strategy meeting; company website; competition websites; existing information systems;
- select the business interviewees: use the organization chart to find leaders; build the enterprise data warehouse bus architecture matrix for the big picture; talk to middle management for details; send a pre-interview briefing letter first;
- select the IT interviewees: IT people who know the data; IT management for a vision;
- prepare the interview questionnaires: see Figure 3-2 (pp 74) for a sample;
- schedule the interviews: start at the middle of the hierarchy; go to their offices; private interviews for executives, peer group interviews for others (don't mix them).

The Interview

The interactive phase (generally called "the interview") consists of either face-to-face **interviews**, **facilitated sessions**, **immersing** a business analyst into the business department, or a combination thereof. Kimball advocates the **interview-first** approach: start with interviews to gather the gory details, and then switch to facilitated sessions to set priorities and make a plan.

The interview starts with an introduction in which the interviewee is reminded the goal of the interview and the overall requirements process. A round of easy questions follows the introduction, asking the interviewees to describe their job responsibilities. After the easy questions, the interviewer uncovers the details along separate threads with each interviewee. Each interview thread starts with high-level questions (see the sample on pp 82), and then follow-up questions. Be conversational rather than rigid. When wrapping up the interview, ask about **success criteria**. Cluster the collected business requirements around **core business processes**. They are the fundamental building blocks of the DW/BI because they represent a manageable chunk of data and analysis.

An interview should not exceed 90 minutes, should group up to four interviewees, and there should be at most three interviews a day. Recording devices may be used during interviews, although they can make some people less comfortable. **Surveys** and **questionnaires** are not good replacements for the interview. Neither is the practice of **analyzing existing business reports**. However, both can complement the interactive phase.

Documentation Writing

Each interview should be quickly followed by a **write-up** based on a template such as the one on pp 88. At the completion of the interviews, all write-ups should be consolidated into an **overall requirements findings** document, organized as shown on pp 88-89. This document establishes the relevance and credibility of the DW/BI program. As part of the DW/BI life cycle, this document marks the first time when the business requirements are tied to the available data.

The requirement findings document also contains a first version of **enterprise data warehouse bus matrix** (Fig 3-5, pp 90). The rows of this matrix correspond to business processes, while the columns correspond to **conformed dimensions** – the "by" words heard during interviews, such as "we need to look at underwriting transactions by policy holder". An **opportunity matrix** (Fig 3-6, pp 90) is similar, but its columns correspond to departments/groups rather than conformed dimensions.

Once we have all business process requirements collected in the bus matrix, we set up a meeting to prioritize them. To this end, we use a **two-by-two prioritization grid** (Figure 3-7, pp 92). The closest the BP to the top-right corner of the grid, the higher priority it should receive. Once we establish the priorities and agree on a roadmap, we distribute it and then we go back to the first box (Project Planning) in the life cycle. We are now ready to start collecting requirements for the designated initial project in the program.