



"turning data into dollars"

Tom's Ten Data Tips – August 2009

Data Warehouse Development

Software projects usually begin by gathering requirements and then building what is needed. Data warehouse (DWH) projects on the other hand typically begin by building what is needed, and only *then* do you wind up with requirements. This calls for a radically different approach to development and project planning.

Inmon et al (2008): "... the requirements for a data warehouse are often not known when it is first built." Instead, confusion and ambiguity reign. Gathering DWH requirements involves negotiating among stakeholders which functionality should be delivered first, and ensuring robust alignment with corporate strategy. Given ubiquitous lack of strategic clarity, this is no easy task.

1. Be Ready For The Information Democracy

When organizations embark on their first DWH project, they often strive to establish "one version of the truth." 'Spreadmarts' (data silos maintained by individuals in spreadsheet programs) across the organization need to be surpassed by a central place to turn to for the definitive truth. And indeed a DWH can be very effective for that.

What is much less obvious, and often not anticipated, is that a dramatic change in (informal) power structure will take place. Poignant facts about business success (and lack thereof...) will emerge, waiting for anyone to be discovered in the DWH. What this does is that it shifts the "informal power structure" from those in command to those in control of information gathering skills. This can upset existing hierarchies, and trigger significant resistance.

2. Every DWH Starts With A Business Case

Sometimes data warehouses are needed to meet regulatory requirements. One could argue that your DWH then is a *conditio sine qua non*, a ticket to market. All other cases should definitely be founded on a solid business case. Not only to "justify" investments being made, but certainly also to provide guidance on setting priorities within the project.

When the going gets tough, your business case reminds you *why* you got started in the first place. It provides a tangible manifestation of successful execution of business strategy (see also tip# 5), justifying management attention and corporate sponsorship. *Every* DWH project runs into setbacks, in particular during the extract, transform, load (ETL) phase (see also tip# 4). When you're having second thoughts, or the project needs to "compete" for resources, the business case will see you through.

3. Deliver Your DWH in Increments

For all the (largely pointless) controversy between Inmon and Kimball, one thing is clear: a DWH needs to be delivered incrementally. Inmon fully agrees but his approach to enterprise data warehouses (EDW's) appears much less suited for an incremental, bottom-up implementation. For an EDW, however, Kimball's architecture has some major challenges. Conforming departmental data marts (DM's) is difficult enough as it is, but changes in grain (the minimal level of detail available) wreak havoc and invariably cause major scrap and rework.

What is required to make incremental delivery of a DWH robust, is a top-down architecture in combination with bottom-up implementation. It's the only way you can gracefully deal with (unavoidable) change. Since 2/3 of TCO will go to maintenance (itself, largely building in changes) the architecture should account for this.

4. Test For (Acceptable) Data Quality First

As you are mapping information requirements onto the existing data systems in the enterprise, some sources may prove to be more "mission critical" for the DWH than others. Unless data quality programs and a (working) master data management (MDM) solution are in place, you can never take sufficient data quality for granted. Therefore do preliminary data profiling as early as possible, to give management a "feel" for what data quality is like, and work these findings into your project estimates.

One company we worked with wanted to learn from their DWH who their best customers were. However, they implemented a sales force automation tool that had failed spectacularly. Many reps were not providing data at all, and those that did, provided irreconcilable naming. The same client could go under a dozen names, or more! There was no way the DWH could enable tallying up cost and revenue

per client. Unwelcome as it may be, you're better off learning these things sooner than later!

5. Profile Your Source Systems To Plan Your Project

About 60% or more of DWH projects goes to the ETL phase. Incidentally, this is also where the greatest risks for project overruns lie, and certainly *not only* because it is such a large chunk of the work. Poor data quality, missing, incorrect, or out of data source specifications, a shortage of business domain expertise, transformation complexities, and many other things can go wrong here. Kimball (2008): "Due to all the unknown data realities hidden in your source system data, data staging processes have a well-earned reputation of being nearly impossible to estimate and deliver in time."

The one way to gather objective information to help you plan this stage better is in-depth profiling of source systems. This also alleviates any hiatus in meta data available about these systems, which will tremendously help ETL programmers do a better job the first time around. Remember that data staging is a "classic" software development task (see also next month's newsletter on software testing) where testing and trial deployment are bound to be followed by multiple iterations of fixing. If you only planned for development and initial testing, you will drift from the plan – and potentially get lost!

6. Tying Your DWH To Corporate Strategy Is Mandatory

Given the central role a DWH will play in the organization, it is adamant that you closely align it to corporate strategy. If you're pursuing a product strategy, make sure the right KPI's are supported. Likewise for operational excellence: defect rates, wing-to-wing time, cost measures, scrap & rework, etc., all should be available.

The best way to surface an organization's strategy is not to read documentation or corporate promotion on "strategy." These PR blurbs tend to be overflowing with platitudes, and the strategy as stated there may or may not coincide with practice at all. You need to investigate what "excellent performance" means. Having the latest and greatest technology (product strategy)? Zero out-of-stock positions (operational excellence)? Everybody wants to satisfy their customers these days, but that does not necessarily imply customer intimacy, etc. You infer the 'true' corporate strategy by finding out what

managers need to accomplish in order to earn an excellent performance review.

7. If It's Not An EDW, You're Creating A (Another?) Silo

There is nothing wrong with creating a DWH to meet specific departmental needs. In particular if this is where you can make a profitable business case. If data driven decision making is to become the norm, though, realize that at some point you will want to create horizontal integration of information.

However, somewhere down the line, someone *will* raise the question how the comptroller's numbers are related to marketing, how marketing campaigns relate to activity in the call or service center, etc. "We can't say" because of some concocted, technical explanation doesn't impress. Certainly not from the DWH team. Yet these questions are only natural and legitimate! Only EDW's allow you to answer cross-departmental queries. If you embarked on a departmental DWH, managing these expectations is a full-time job.

8. DWH Architecture Is A Genuine Profession

Although data warehousing is still relatively new, enough experience and practices have been established to merit a new and unique role: the data warehouse architect. Like with "traditional" architecture, you need to negotiate (business) needs and wants with technical possibilities. The outcome can take the form of a blueprint (architectural drawings), a plan (construction project), set of components (chosen building materials), or principles (legislation or guidelines). All too often, there is confusion about what exactly is meant by "architecture."

Unfortunately, tradition has it that when developers have been working in the field from 3-5 years on, they get "promoted" to architect status. This doesn't necessarily mean they have any formal training in architecture, it usually just means you're dealing with a developer who's got hands-on experience, and who has probably seen several mistakes made more than once. Some developers are just very good (and maybe experienced) developers, and some people are more suited for other work. Developing and architecture require completely different people skills, the kinds of (almost innate) talents that don't necessarily evolve with experience.

9. Consider The *Business Case For Real-Time Data*

Integrated data are used ever more widely to support operational decision making. This may happen in your call-center, where optimal cross-sell suggestions may be used that were derived via data analytics. Or in your warehouse when last-minute changes to shipments are based on the latest data on availability. The end result is that the pressure on your DWH to deliver (more) timely data increases. Monthly refreshes used to be the norm, then we went to weekly, but many data warehouses are now updated every day, or even more often. Hardware is becoming ever cheaper, and your architecture has a tremendous bearing on the cost to increase data throughput, and update frequencies. So technically, a lot has become possible, but faster (or the holy grail: "real time") data always come at a (substantial) price.

Most operational data must absolutely be up-to-date for a flawless customer experience. Like contact data in your CRM application, or warehouse and shipment data. But must your cross-sell suggestions be real-time? How often will you make "the wrong" offer if your next best offers are calculated only once per week or month? Everybody *always* wants their data faster, but has the value of better decision making really been quantified?

10. Data Warehousing Should Be A *Programme, Not A Project*

Building a data warehouse from the ground up is such a gargantuan effort, requiring specialist skills, that outside help is just about always required. To fence off these consultant resources "some" project needs to be defined, preferably around one or more initial increments (see also tip# 3).

You can not expect your data warehouse journey to *ever* end. There is never a fixed end date, and maybe more or less of a start date (the kick-off party?). Kimball (2008): "... each data warehouse is continuously evolving and dynamic."

Since you want as smooth a transition as possible from building to deployment, this is best accomplished by making the effort part of a programme, right from the start. You'll typically encompass multiple parallel efforts like data profiling, overhaul of source systems, cataloguing of meta data, data quality efforts, etc. Demand from your consultants that they actively help you transition from a project mode to in-house continuation of DWH operation (if that's your chosen maintenance model).