Paper 268-2009

The interactive data warehouse

Introducing transactional data marts and “smart applications” to interact with data warehouse data

Stein Arve Finnestad, Capgemini Norge AS, Stavanger, Norway
Pål Navestad, ConocoPhillips Norge AS, Stavanger, Norway
Gisle Karlsen, ConocoPhillips Norge AS, Stavanger, Norway
Rune Lekve, SAS Institute AS, Stavanger, Norway
Odd Jarle Tednes, Capgemini Norge AS, Stavanger, Norway
Terje Strømstad, Capgemini Norge AS, Stavanger, Norway

ABSTRACT

Traditional data warehouse literature describes four maturity levels for a data warehouse, the highest being the “Integrated data warehouse”, where triggers in the operational systems keep the data warehouse immediately updated, and new transactions from the data warehouse are immediately written back to operational systems.
But the number of different operational systems, the absence of proper legacy system API's, and the difference in transactional level between the data warehouse and operational systems make this unattainable for most enterprises.
Building on the foundations of an offline SAS data warehouse, the project team at ConocoPhillips Norge has developed the “Interactive data warehouse”; an “intermediate” maturity level: We introduce the “transactional data mart”, a database acting as both a data mart and an operational database, and a three-tier application using this as its database. The application interfaces with SAS Stored Processes to run simulations, reporting and so on. New aggregated transactions are written back to the data warehouse instead of back to operational systems, typically by scheduled SAS DI Studio ® jobs.
Precedence rules between original source systems and the transactional data mart are essential to the success of this concept.
The paper will explain these topics in detail, using examples from Integrated Planning in the oil industry.

INTRODUCTION

- Traditional data warehouse literature describes four maturity levels for a data warehouse:
- Offline Operational Databases
- Offline Data Warehouse (periodically updated using scheduled, automated ETL jobs)
- Real Time Data Warehouse (immediately updated using triggers in operational systems)
- Integrated Data Warehouse (Real Time data warehouse, and triggered write back from data marts to operational systems)

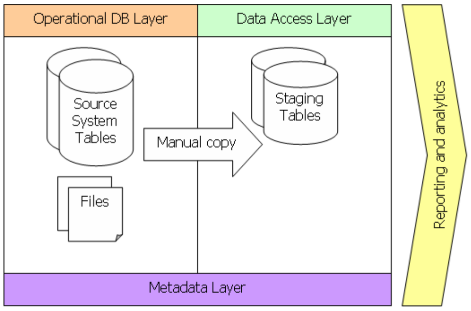

Figure 1. Offline Operational Databases

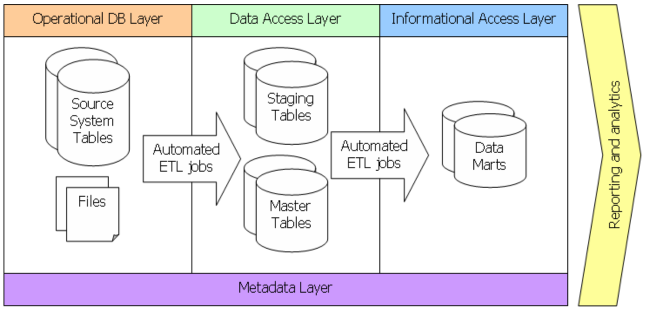

Figure 2. Offline Data Warehouse

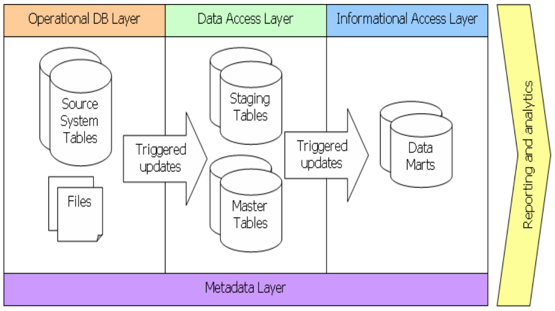

Figure 3. Real Time Data Warehouse

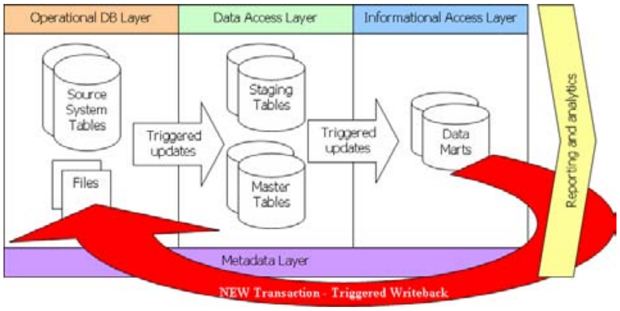

Figure 4. Integrated Data Warehouse

The integrated data warehouse is described as the highest and hence most "advanced" maturity level. But enterprises with a lot of old legacy operational systems cannot get to this maturity level without a disproportionate cost. Most enterprises also have more than one operational system evolving over time, each with its own technology and API, adding complexity to developing a fully integrated solution. Operational systems such as ERP solutions are often tightly connected to business processes that may not be compatible with transactions based on aggregated data warehouse data. Moreover, aggregation processes may introduce logical errors that may be impossible to fix if source system data is overwritten.

There is also another, more structural problem: The transactional level is often higher in the data warehouse than in the operational systems, since the data warehouse works on aggregated data which cannot be written back to the operational system.

These are all good reasons why most data warehouse implementations today are offline, or at best, for some selected data sources, real-time.

At ConocoPhillips Norge, we have discovered 2 "intermediate" data warehouse maturity levels, in addition to the four "traditional" ones:

- Semi-Integrated Data Warehouse: An offline data warehouse with periodically (not triggered) updates from the data marts to the operational systems using scheduled, automated ETL jobs.
- Interactive Data Warehouse: An offline data warehouse from which a (sub-) dataset is replicated to an external online relational database that in turn is used as a source for the data warehouse. Updates will typically be scheduled, but can also be triggered.

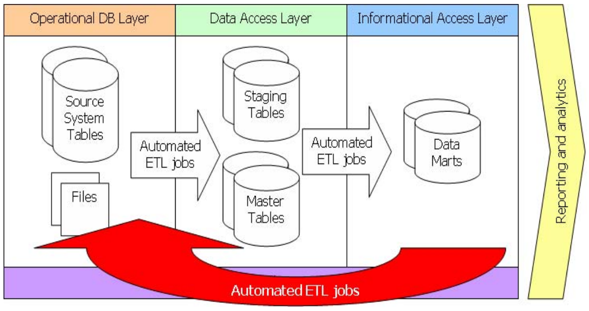

Figure 5. Semi-integrated Data Warehouse

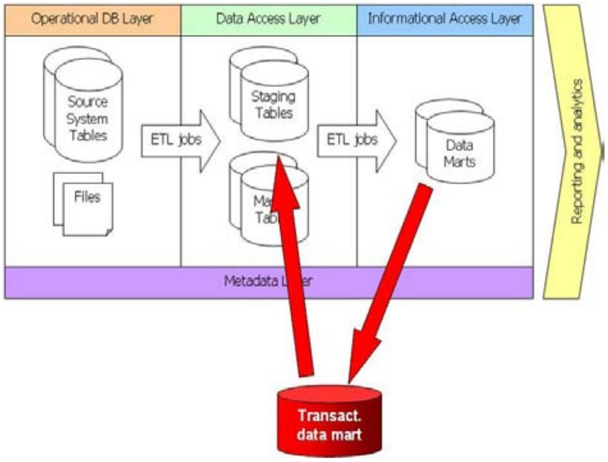

Figure 6. Interactive Data Warehouse

The interactive data warehouse is an attractive approach for businesses that need to work on integrated and aggregated data in data marts, but cannot take on the cost and risk of interfacing existing legacy operating systems.

In order to implement the interactive data warehouse, we suggest an external online database which contains a (subset of) the data mart and at the same time acts as a source system for the data warehouse, as well as the database server for a "smart application". The smart application operates on aggregated and/or detailed data, and interacts with the Reporting and Analytics Layer and Information Access Layer of the data warehouse, particularly through interaction with SAS Stored Processes.

CASE: THE DATA WAREHOUSE AT CONOCOPHILLIPS NORGE

The offline data warehouse at ConocoPhillips Norge is designed to facilitate the operations at Greater Ekofisk[1]. It contains scores of information for different domains, such as pressure/temperature data from production, data about safety incidents, daily and finance adjusted production volumes, maintenance and project work, personnel on board, etc.

The data warehouse has been gradually built through a period of 10 years. The original aim was to do analysis of safety related maintenance. As the data warehouse has evolved and more ideas have been explored, it has become possible to use it as a basis for doing Integrated Operations, including Integrated Planning.

INTEGRATED PLANNING

In a large company there will always be several different plans, with different owners, possibly created in different tools, and with different focus and level of detail. The budget and long term investment plans e.g., are plans with very few details, while the detailed maintenance plan has a high amount of details.

Integrated Planning is a process that coordinates all long and short-term activities to produce one overall approved and prioritized plan and schedule. In other words: Integrated planning is combining all plans from different sources into one large master plan. This master plan is then analysed, optimised and tweaked, and can be viewed in different time horizons spanning from tomorrow’s work list to forecasting what is believed to happen in 20-30 years time.

The source for planning data is initiation of new work. New work transactions usually appear after somebody has found a fault or an improvement area. The transactions are relatively few, but complex, stay open for a very long time, and are changed at irregular intervals, very unlike e.g. bank transactions, which are very simple, but come in large volumes. These discontinuous data has to be combined with continuous production and plant data and this is not simple. Nevertheless we have found great synergies from working with these diverse data at the same time.

There are always some common problems and common solutions. We also find that information which we at first can’t see any relation between actually turns out to be related after all. As a consequence of this we have found combinations of data that are not possible to replicate outside the data warehouse. This information has become business critical, and is used to create alarms and manage operational issues.

Integrated Planning at ConocoPhillips Norge has been implemented in 2 phases. The first phase created the data mart and metadata model, ETL jobs to load and integrate from different sources, and several reports, e.g. Gantt charts and work lists. The second phase, initiated from the need to interact on the integrated data, is where the interactive data warehouse and the smart application evolved.

THE ARCHITECTURE OF THE INTERACTIVE DATA WAREHOUSE

One of the most common IT problems is that for all processes with some complexity, the data organisation for reporting and analytics is highly different from the data organisation for interaction. A de-normalised (star schema) structure is often the most effective structure for reporting and analytics, while a normalised entity-relationship structure usually is preferred for manipulation by user applications. At ConocoPhillips Norge we have created a data mart for integrated planning that extracts data from several data sources, and organised the data in ways suitable for e.g. reporting and analytics. The resulting data organisation is difficult to use for interaction.

THE TRANSACTIONAL DATA MART

Because of this we do all the analytical work in the data warehouse, and introduce a new relational structure to be used for the interaction application: A database which is both a data mart and a (new) operational system as seen from the data warehouse, as well as the database server for a “smart application”. ETL jobs extract a subset from the data mart, transform the data to normalised form and load them into the transactional data mart. The smart application operates on aggregated and/or detailed data, and interacts with the Analytic Layer and Information Access Layer of the data warehouse, particularly through interaction with SAS Stored Processes. Other ETL jobs

[1] The Greater Ekofisk area is comprised of four producing fields: Ekofisk, Eldfisk, Embla and Tor, and consists of a total of 29 installations. Approx. 1,300 people are situated at these installations at any time. The Ekofisk complex is located 200 miles offshore Stavanger, Norway. Since first production in 1971, technology has been used to increase production and extend the economic lifetime of the field. More details can be found here: http://www.conocophillips.com/about/worldwide_ops/country/europe/norway.htm

extract changed data from the transactional data mart, transform them into de-normalised form and loads them into the data warehouse, similar to what is being done with changes coming from other source systems.

One of the most obvious advantages with the transactional data mart is that we get a database with a predictable size. The number of activities will increase in the data warehouse, which is still used for all analytical and statistical reporting. The transactional data mart may be scoped using a transient time-window, so the database will always be approximately at the same size. Even all the open planning data for a very large site like Greater Ekofisk is at a manageable size with less than 100,000 observations.

LOADING FROM THE EXISTING DATA MARTS INTO THE TRANSACTIONAL DATA MART

One of the first problems we faced after creating the transactional data mart was the two ways to update data; either from the source system or from transactional data mart. At least 20 % of the project has been spent on making certain that we are able to honour the right data source. From an operational perspective it may be a problem that there is a lag from when transactions are modified in the source system to when they are updated in the data warehouse. In our case the update is daily. But another problem is how to handle bad or missing data. Ideally this should be fixed immediately, e.g. in a planning meeting. In such cases we allow direct fixes in the transactional data mart. But what happens next? Should the data warehouse too be updated with the new (presumably more correct) value? This illustrates the need for clear precedence rules, which will be outlined in the next section.

LOADING FROM THE TRANSACTIONAL DATA MART INTO THE DATA WAREHOUSE – PRECEDENCE RULES

Writing directly back to the source systems is not always feasible. The source systems feeding the ConocoPhillips Norge data warehouse include systems operated by third party contractors, and there are both technical and contractual reasons why it’s not possible to write data back to their original systems. Most of the data entry in the Smart Application is done at an aggregate level, on dependencies between data from different sources, or even at levels that do not exist in any single source system. In some cases it may be necessary to overwrite or manipulate data at the source level. In a planning application this becomes evident for progress reporting. Especially in situations with high activity levels there is a need to be able to update progress very fast. However, the main purpose of the most important source systems, is to plan the individual job, and allocate time and cost. Typically this is done through the individuals’ time sheets, which (at best) is updated weekly. In an ordinary operational mode this is acceptable, but in times when the pace is faster we need updated progress information much earlier. Consequently we have made it possible to override this in the transactional data mart using the smart application.

Write back into the data warehouse puts another problem into play: There is a potential risk that the data in the smart application overwrites data from the source application in a way that creates undesired effects in the data warehouse. Care must always be taken to avoid this. One of the provisions we have built in is for how long the transactional data mart should have precedence over the source data.

Columns present only in the data warehouse. Simple to maintain but the columns need initial or default values when source is the transactional data mart.

Columns present only in the transactional data mart. Simple to maintain but the columns need initial or default values when loading into the transactional data mart.

Columns present in both the data warehouse and the transactional data mart:

- The source system is responsible, i.e. keep the value in the data warehouse. Changes are not supposed to happen in the transactional data mart, neither by any applications nor by automatic batch jobs etc. The smart application will prevent such columns from being updated. This should be the normal situation.
- The transactional data mart is responsible, i.e. overwrite the value in the data warehouse. Should generally be avoided, but can be used in cases where data quality from the source is bad.
- The source system is responsible, but it is practical to perform a quick update in the transactional data mart. The smart application permits changes, but warns the user that he/she needs to go back to the source to perform the permanent change. For the batch update job, this means keep the value in the data warehouse. This rule should be avoided if possible.
- Rule controlled by flag or timestamp or another column’s value. Row level precedence. The batch job examines the content of certain columns in the record to determine which system is responsible. Examples:
    - If source system is ’XXX’ then smart application is responsible (because this column is lacking or have bad quality in that particular source system), else the source system is responsible.

- If ‘allow edit flag’ is ’Y’ then smart application is responsible, else the source system is responsible. The ‘allow edit flag’ can be changed from the smart application by an authorised user. Gives users the possibility to ‘take control’ over a record. Gives users the possibility to take control.
- If ‘last report date’ > (today – n days) then smart application is responsible, else source system is responsible. Gives users the possibility to take control for a limited period, and automatically give precedence back when the source system is expected to be updated again
- If column value is ‘XXX’ then smart application is responsible, else source system is responsible. Can e.g. be used to mark records which are subject to special attention: Column ‘severity’ says something about how severe a record is, and the source system selects from a number of codes when filling this. Introduce one new code not found in the source system, e.g. ‘Manually monitored’.
- Etc.

BATCH JOBS – MASS CALCULATIONS AND UPDATES ON THE TRANSACTIONAL DATA MART

The data mart that we use for Integrated Planning gets data from different source systems, as shown in Figure 7.

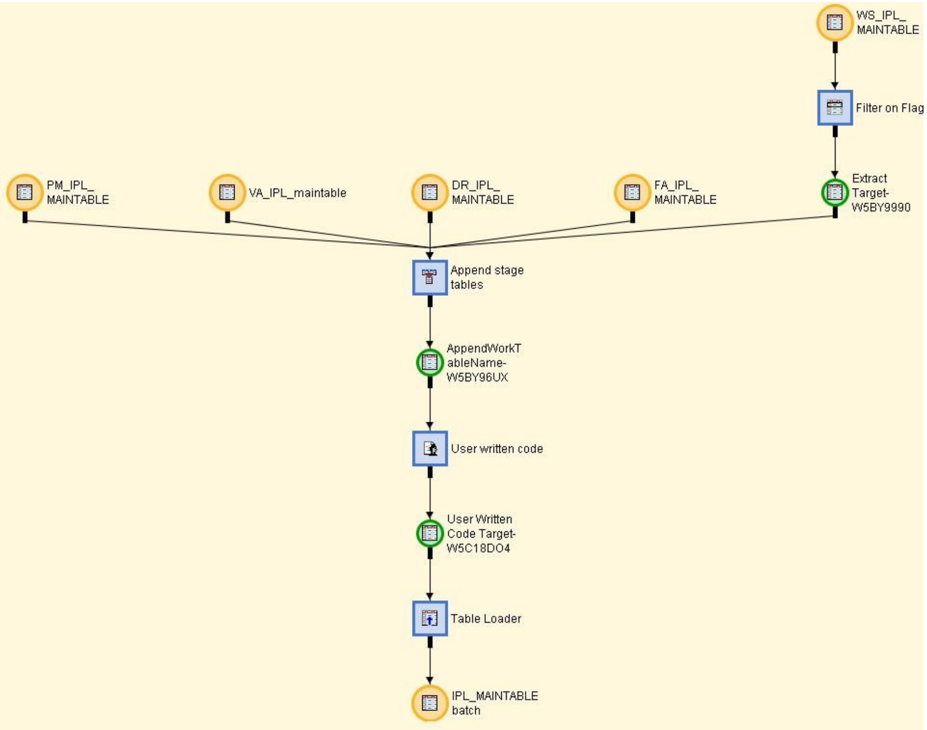

Figure 7. Data Integration flow loading the Integrated Planning data mart from several source systems.

An addition to the source planning systems the transactional data mart has been introduced as a data source in the Data Integration flow. The original design of the integrated planning data mart is such that every source system delivers data that is structurally fitting into the data mart. Data from the transactional data mart are inherently structurally fit, so its integration becomes pretty easy because of this. However, there are some considerations to be made.

Even though the data is delivered from the source in a structurally consistent manner there are a lot of business rules that only have meaning after integrating all of the data sources. With the added complexity of precedence rules from the transactional data mart it is necessary to have really good control of the business rules and how these are applied.

We have used SAS DI Studio ® and the SAS Enterprise DI ® server to manage all of these rules. Especially the Impact and Reverse Impact analyses capabilities of the DI Studio ® product have proven to be extremely useful in this work. However the most important feature is that we are really utilizing the Metadata controls and possibilities that are built into the product. Great care is taken when designing the jobs, and there is a lot of peer discussion and design around the jobs. The overall guiding principle is that it must always be possible to follow the metadata flow completely.

When designing the jobs there is always a compromise between ease of programming and maintenance and execution speed. Introducing the transactional data mart has made it easier to separate the “slow” nightly batch updates from the data that has to be updated immediately.

All DI jobs in the data warehouse are self documenting. This has proven to be a success, and programmers have been able to quickly fix faults and correct jobs they’ve never touched before, in stressed situations. It is also a good selling point for programmers that we don’t write documentation. We have good experience in on boarding new programmers and consultants using this approach since everything contains its own documentation. One must be aware that this approach requires that everybody conforms to the agreed standards.

THE SMART APPLICATION

All operational systems that depend on human interaction need an application to be able to manipulate the data. For the planning application we looked at several commercial available products for planning, for instance PrimaVera® or SAP®. We found that it was very hard to adapt these user interfaces to our requirements, and decided to build our own application.

The application has been built using agile techniques, without a clear scope. Since Integrated Planning is a new concept, we have developed business processes and IT tools at the same time. To do this you need a multi skilled team. In our case we have had very close cooperation between the business representatives, the Java programmers and the SAS expertise. Most of the persons on the team have had several of these skills.

Even though we did not have a clear scope of the application and how it would look like, we had some very clear ideas of what we wanted to achieve:

- Get data from the data warehouse and apply general business rules
- Add knowledge and business rules that can not be generalized
- Plan using several work breakdown structures
- Override general business rules when appropriate
- Have the opportunity to update integrated data quickly as opposed to waiting for slow nightly batch jobs

The application has been designed as a composite of a pure transactional system and a reporting system. This has also affected the database: Some tables were designed from the start in a way that makes reporting more efficient, while at the same time obtaining reasonable speed for transactional work. An example of this is the table that stores approved offshore bed assignments: The smart application displays one line with start date and end date, while the database stores this as one record per day, which is much easier to report on.

The application was written from the beginning as a hybrid for manipulating data, reporting, advanced analytics and optimisation, and it has consequently been possible to use the best concepts from each world. We used traditional 3-tier architecture, as shown in Figure 8, with the transactional data mart as database.

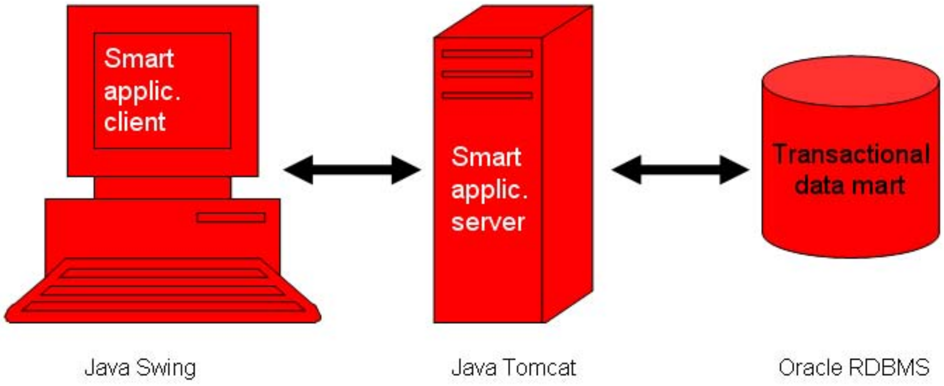

Figure 8. Three-tier application architecture.

The main advantage is the possibility to create functionality specifically for the business problem at hand. An ERP system like SAP will always be a compromise between different requirements, for instance you want to do single job planning, control the material flow, record and allocate costs and do overall planning at the same time. Some of these requirements are contradictory, and even though SAP has done a good job it is more or less impossible to fulfil all requirements. The focus of the smart application has been solely on integrating plans and making the interactions. We do not replace functionality which works well in the original application.

Project planning systems such as PrimaVera are mainly meant for planning projects and adding tasks. Consequently, there are no functions for integrating many different planning sources. We have been able to utilize the original systems' functions fully because integration happens in an intermediate layer of general business rules and transformations.

Finally, we've developed new functionality not found in the source systems by enhancing data in the transactional data mart and utilise the analytical and optimisation powers that exist in SAS. For instance, in planning, we've introduced more sets of dates in addition to the traditional sets such as early/late and schedule, to give more flexibility to how we optimise a schedule run.

ACCESS CONTROL TO DATA MANIPULATION

The smart application has incorporated a lot of access rules to grant or deny users the possibility to manipulate data. To a large extent, these rules follow the precedence rules discussed earlier. If a piece of information is governed by the source system, it's normally not editable in the smart application. Sometimes you can change a flag to indicate that you override the source for this particular column value. Experience has shown that access control rules in the smart application follows closely the precedence rules in the EL jobs.

SIMPLE INTERFACING WITH SAS STORED PROCESSES

We developed a module in the Smart Application to facilitate the process of approving/rejecting requests for offshore beds. The user (the approver) gets a list of requests, and can press a button to launch a report (histogram) where he/she can control overbooking etc... The report is implemented as a SAS Stored Process in a web browser window. Only after this report button has been pressed, the button to approve requests is enabled. Very simple interaction, and there's no guarantee that the user chooses to not look at the report, but this way we guide the approvers to review the consequences before approving.

ADVANCED INTERFACING WITH SAS STORED PROCESSES

A SAS Stored Process can do much more than just generate and display a report.

One example is 'what if'- simulations in integrated planning: After changing data in the application, one may launch a stored process which:

- extracts data to a temporary data mart
- runs a schedule using PROC PM
- displays a report comparing the new schedule with the “official” version in the data warehouse
- gives the user the opportunity to save or discard the new schedule

Even though there is no direct feedback to the smart application from the stored process, there is feedback on the data level. The possibility to launch and interact with SAS Stored Processes from the smart application enables us to combine the speed and user interface of a transactional system with SAS' full range of powerful statistical and analytical computing functions.

CONCLUSION

Implementing an integrated data warehouse can be extremely complex and time consuming, and there are a lot of pitfalls to be overcome. At ConocoPhillips Norge we've developed an “intermediate” maturity level which we've called the “Interactive data warehouse”. The interactive data warehouse is in effect an offline data warehouse, with ETL batch loading into an online relational database that we've named the “transactional data mart”, which in turn is a source to the data warehouse, giving the opportunity to create and alter transactions on levels not possible in the different source systems.

The transactional data mart is manipulated by a “smart application” as well as several batch jobs for optimisation, mass calculation etc. The possibility to interface the smart application with SAS Stored Processes has been a very important enabler, giving access to the vast function richness available in the SAS environment.

Precedence rules between original source systems and the transactional data mart are essential to the success of this concept. The same precedence rules should be recognised as access rules in the smart application (if a column is governed by the source system, it should not be editable in the smart application). It is possible to control the precedence/access rules down to single row/column level, and change them on the fly when needed, e.g. by flags.

The interactive data warehouse and the smart application has given the users at ConocoPhillips the best of two worlds; the calculation, analytical and reporting power of a data warehouse combined with the flexibility and user friendliness of an operational application.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Stein Arve Finnestad
Enterprise:	Capgemini Norge AS
Address:	Maskinveien 24
City, State ZIP:	Stavanger, NO-4033
Work Phone:	+47 – 922 75 607
Fax:	+47 – 51 57 92 16
E-mail:	stein-arve.finnestad@capgemini.com
Web:	http://www.no.capgemini.com