



A Tale of Two Suits

A Discussion on the Commercial Aspects of Developing a Data Warehouse

*Part of the series of the Insource **Commercial Aspects of BI** discussion papers*

Hazel Markou
10th August 2008

Contents

Introduction.....	3
Objective.....	3
Background	3
Management Summary.....	4
Building a Data Warehouse – A Non-Technical View	5
Options for Building a SQL Data Warehouse	6
Use of Framework and tools	8
Conclusion.....	10
About the Author	10
About Insource.....	11

Insource is part of the Insource Group, Data Management & Business Intelligence specialists

© Insource IT Consultancy Ltd 2009. All rights reserved. Insource Data Academy® is either a registered trademark or trademark of Insource IT Consultancy Ltd. Microsoft is either a registered trademark or trademark of the Microsoft Corporation in the United States and/or other countries

Insource

**Insource House, 4 Southern Court, South
Street, Reading, Berkshire RG1 4QS**

Telephone: +44 (0) 118 921 1555
Facsimile: +44 (0) 118 921 1565
eMail: contact@insource.co.uk
Web: www.insource.co.uk

Introduction

Objective

There's a myriad of technical white papers covering Business Intelligence, data warehousing and information management. It matters little what tickles your technical fancy, there's likely to be a document that covers it for you. However, you can't say the same for commercial papers; for some reason these seem to be rather less common. That's something we're aiming to begin addressing via this white paper.

So why write these papers? The simple answer is that investment in BI solutions (and the data warehouse needed to support BI) is a business decision. The objective of BI in the enterprise is to develop a commercial advantage and it's a good idea to have some business documentation to support the decision. So we decided to create a series of papers which focus on the business issues surrounding data warehousing - which for this white paper will be the factors involved in deciding whether to adopt a bespoke code or framework/toolset approach.

Background

Getting meaningful information out of a software application was historically a mind-numbing, resource intensive task. This pushed the development of new concepts in organising and accessing data by luminaries including EF Codd, which in turn led to the development of tools, standards and technologies, such as the relational database, to make the process easier and less onerous. However, this also created a new problem; what to do with all the seemingly related or even duplicate information then flying around the organisation.

By the late Eighties the first concepts of data warehousing were being laid down from an article by Barry Devlin and Paul Murphy titled *An Architecture for a Business and Information Systems* which appeared in the IBM Systems Journal. With existing systems unable to cope with the new demands placed upon them by this new concept, data warehouses were built as separate systems that brought in data from the disparate sources within the organisation and integrated the information into a single platform.

Management Summary

This is a short discussion paper looking at a particular aspect of developing a data warehouse; the choice between building with manual coding or introducing software tools to automate some or all of the process.

The document commences with a simple explanation of the terms Data, Information and Intelligence, and from there aims to take a balanced view of either methodology and thus attempts to state both benefits and drawbacks. For the hand-built approach the extent that the data warehouse can be customised to a best fit objective was noted, whilst also pointing out that such a method is time, resource and budget hungry and vulnerable to staff turnover, a lack of coding standards and documentation can be quickly rendered obsolete and can present support issues whilst in operation.

The framework/tools choice on the other hand can address many of the issues that a hand-built approach will pose. It is quicker, will normally work to a standard, can be updated more easily, will have a pool of knowledge in existence from other customers and will benefit from comprehensive documentation and support. There is a concern that the level of customisation may not be as high as the hand-built option, but I would conclude that such developments can be polished after the majority of the work has been done by the tools – thus presenting the organisation with the best of both worlds.

The paper concludes with the finding that whilst either approach has a strong case, the framework and toolset derived route is potentially the most advantageous to the organisation; it can be less budget and resource hungry for development and support, while also having the benefit of consistent maintainability and support for growth in the future.

Building a Data Warehouse – A Non-Technical View

Before diving in and discussing the minutiae of Data Warehousing I'm aware there is still some confusion over terminology, so I will clarify some of the expressions in use. To start, let's explore "data", "information" and "intelligence" – each term has its own position within the discipline and so here's a good way to describe each term.

Data is a simple, raw fact which here I'll express as *I am wearing a skirt, jacket and blouse.*

Information goes further, by relating data together. So for example you could now say *my skirt and jacket are blue and the blouse is pink.*

Intelligence moves information along by bringing it together from different sources to create a more comprehensive picture. So for example, my skirt is 2 months older than my jacket and the colours are all of the same tone and coordinate with each other.

Good data warehousing is the culmination of a lot of planning, research and in many cases some degree of change management within the organisation to deliver good practice in information management. As we're looking at data here I've noted some considerations that need to be made when creating a data warehouse.

Data needs to be captured from a variety of source systems. A typical medium or large organisation will have a variety of platforms and applications using different programming languages. They all have a valid contribution to make to the data warehouse as the information they contain will help build the pictures that allow for more confident decision making.

Data Quality needs to be checked. However, the same data that has been drawn from all those systems needs to be thoroughly checked for its quality – there's little point in retaining information that is obsolete or wrongly recorded for example, or if it duplicated in other pools of information being brought into the same warehouse.

Information needs to be derived from data if it doesn't exist in raw format. Some data is only available by deriving it from existing data. For example your system may hold units purchased and unit cost – but not a record of line item value. However, incorporating a simple x multiplied by y calculation will present the order value for any relevant report.

Data needs to be manipulated to get it ready for reporting. Having the data all sitting in a single place is one thing. Being able to report on it is quite another. The data has to be correctly structured so that a report can be drawn from it with sub-second response times. I have seen many examples of reports from transactional systems taking hours, if not days, to produce a response.

Reporting tools need to be able to access the data to let people see it and ask further questions of it. The reporting tools used in effect present the data collected together; a commonly used term "single version of the truth" is a holy grail that many organisations strive for. Then the time currently spent debating which set of figures are correct can instead be devoted to more productive and proactive management of the business. This essentially is the justification of having a data warehouse – it provides the means for confident analysis of business performance and therefore decisive decision making.

Options for Building a SQL Data Warehouse

Code from the ground up - Advantages

Customisation

Without doubt a bespoke solution (coding from the ground up) can provide a more tailored solution than an off-the-shelf package. Assuming there is a clear understanding between the commercial and the technical functions involved, the ground up approach can be customised to the “nth” degree, providing every conceivable piece of functionality to meet all ‘potential’ requirements.

Evolving to Meet Business Need

Bespoke development provides a greater opportunity for existing processes to dictate the way in which the application would work – meaning the organisations doesn’t have to change to suit the needs of the software. There is a caveat to this however; Business Intelligence is more than just technology. It involves people and processes too, and sometimes these have to be modified or changed in order to improve specific areas or functions. The bespoke route should also provide for more flexibility and openness to organisational change – which fits in well with the very nature of business intelligence - a constantly evolving strategy that is designed to allow organisations to make business decisions to effect change.

An organisation that has a bespoke application effectively owns and directs the development path. So features, functions and releases are undertaken at the speed and frequency to suit the organisation. It also means the software will only contain the features actually required and the more ‘tinsel’ type add-ons aren’t there. As the organisation owns the application it cannot be subject to external problems such as the software developer ceasing development, going bust or ending support. It can also be quite valuable IP although correctly valuing the software can be open to some debate.

Disadvantages

Time to Build

A ground-up approach means starting with a blank piece of paper. Every element of the data warehouse needs to be specified - and paid for. Then every element of that specification needs to be hard coded - and paid for. Each element of the build then needs to be tested - and paid for. You may have noticed there’s a lot of paying going on before any results start to show.

I like to use an analogy for hard coding. You want to build a house; you have a budget and you have chosen your plot. At this point would you then dig your own footing and set your own foundations? I doubt it. You’d employ a builder who specialises in that area. You then need to get the bricks to build the house. Do you have bricks made to your own specification or do you buy ready made and certified bricks that conform to a standard? Unless you have a very unusual reason - and an unlimited budget - you buy correct specification bricks. For each stage of the house build you go to a supplier who has already made that specific element; windows, doors, flooring etc. You may have a custom kitchen fitted, you may have certain elements of certain rooms customised to your own personal preference but most people certainly don’t build each individual element to have a fully bespoke house. Why? Because 99% of us can’t afford it!

Testing

Each element of a bespoke solution needs to go through the relevant stages of testing (Test, Fix and Re-test). Without a framework or defined methodology to adhere too this makes the process a much more arduous task. Each line of code needs to be tested (it hasn't been written before), increasing the cost before any benefits can be realised.

Maintenance and support

While a bespoke solution does possess a fundamentally powerful reason for adoption, this is also a major weakness. In attempting to provide comprehensive functionality, it leaves the developer to approach the development process in whichever way they see fit – and that could spell trouble later when maintaining the solution. A huge amount of time is wasted in unravelling the methods applied by the original developer. Therefore the only answer is a strict adherence to internal development processes, standards and procedures. This is of course extremely difficult and expensive to enforce. Which you pay for.

With regards ongoing maintenance, remember that the bespoke solution has only been built once – for you. Any changes or maintenance to the solution also needs to be bespoke. No other user/organisations to ask, potentially no comprehensive documentation to refer to, no forums, bulletin boards or partner groups to post questions to. Maintaining a one-off solution is an expensive task involving a time-consuming knowledge transfer to new staff or even re-engaging with the third party consultancy who built the original solution.

Key Staff Dependency

Bespoke software often requires large teams of professionals possessing particular skills such as analysts, programmers, hardware and software specialists and technical writers. It's very unusual for any organisation to already employ the people and the knowledge required to successfully pull off such a challenge. They may be expert application developers but are they experienced Data Warehouse developers? If they do have previous experience, what happens when these specialists are no longer available? Invariably they have been contracted into the organisation to perform a specific task and afterwards they move on. I have no doubt that the individuals involved are more than capable of coding a perfectly suitable solution based on the agreed specification; I also have no doubt that each of these individuals would code a SQL Data Warehouse slightly differently.

Business Continuity

If your Data Warehouse was coded by some long since parted developers; who provides the baton for the next evolution of the warehouse? I would very much hope you've been left with very detailed documentation of your pride and joy. If not you will be left with the difficult task of unravelling the code structure used in the original build, all contributing to additional time and cost. And yes, you will pay for it.

Solution Life Cycle

One issue that can sneak up on a lot of projects is around the life expectancy of a solution. Technology moves on; over time the bespoke solution can lose certain features due to obsolescence or performance issues. A more certain development can be a technology upgrade. Over time this raises its own issues around security or performance patches, support and of course compliance with legislative drivers. With a bespoke solution overcoming these sort of issues isn't straightforward as it requires a re-assessment of the technology in place and subsequent development to repair the area at fault. Pay for it? That's pretty much a given now.

Use of Framework and tools

Advantages

Tried and Tested

Solutions are available that will generate standard SQL code enabling all levels of development staff to maximize their input into this approach to building a SQL Data Warehouse. The code created has been tried and tested (and in most cases even certified by other parties too), so it is an industry standard users can be comfortable that it works. There are no unknowns; you are not the first to be implementing the code!

Time, resource and cost savings

Pareto's 80:20 principle can be applied here. A good SQL DW toolset will provide 80% of the functionality needed to build a robust, fault tolerant DW. Significant impacts on development time and the level of resource required can be achieved (and therefore a significant reduction in cost), with 80% of DW functionality contained with a toolset this negates the need to build from scratch the fundamental elements of the DW, as importantly it also negates around 80% of the need for the 'high-level' resource needed to write this code.

Customization

A new breed of toolsets are now available that will allow customisation of the code generated. Some regard this approach as the 'perfect fit' solution. The majority of the DW written in industry standard, tried and tested code, generated in a fraction of the time it takes to hand script these elements, opening up the resources able to develop and test. The remaining 20% of the requirements (against the specification) can then be hand-crafted to provide the 'tailored-fit' needed for specific business requirements. Remember the house build!

Maintenance and Support

A Data Warehouse is typically used to underpin a Business Intelligence (BI) solution. The nature of any BI solution is one of change. Inevitably this leads to a need for change within the data warehouse that underpins the solution. Change within traditional bespoke Data Warehouse architectures can be extremely costly. By providing a simple to use toolset, more resources can easily undertake these changes and implement them with the minimum of fuss and complexity.

Disadvantages

By utilising a toolset the developer or organisation may be closely restricted to following a certain path which may ultimately turn out to be out of step with other commercial or technical development within the organisation. All toolsets provide their own methodology which affectively restricts you to the toolset's own way of doing things. I will say that in some cases that is not such a bad idea; it can keep a focus on reaching core objectives for example without wandering off on feature tangents. However, it is the longer term life cycle of the toolset that should be scrutinised. Does it provide the opportunity for incremental improvements and developments over time, or does it very quickly reach a development ceiling that cannot be broken through?

Some frameworks and toolsets are 'black box' technology and have no open access to the code they generate. This will mean that an organisation is tied into using ongoing consultancy from the provider which may prove an expensive commitment, although it does hopefully provide a guarantee of standards.

Careful scrutiny of the licensing and other terms and conditions is always recommended. Is support or development restricted by a contractual obligation? What are the ongoing costs involved in using the application? What commitments are made by the application provider in terms of support, updates, patching and life cycle?

Conclusion

As you can see from the points made in this paper there are valid arguments to support either approach. Like a bespoke suit, the manual coding approach will always give the most precise and tailored fit for an organisation. However, it is the most time consuming, resource hungry and expensive approach – as well as presenting the biggest risk to your organisation. Will the organisation ever reach the objectives set at the start of the development project? How would the organisation deal with development staff leaving or moving roles? Are the people involved in the development assigned full time or shared with other projects or duties? What would be the level of documentation? These are just a few questions that would have to be answered and some very tough decisions reached before committing to what could potentially be a high level of capital expenditure.

The application derived route will never quite match the tailoring of a manually coded data warehouse, but the trade-off is potentially a faster completion date and access to a wider community if required for support and development. Note here however that I haven't said it would be more budget-friendly; some applications can display a strikingly similar appetite for your money to the manual coding approach. However, you do have the assurance of joining a community of fellow users, and there is nothing better than the comfort of knowing that you can go and speak to peers about your experiences and ideas.

Food for thought I hope, and if this paper promotes a healthy debate before you commit to your chosen route then I will be very happy indeed. If you have any comments to add I would love to hear from you – please email me at hazelm@insource.co.uk.

About the Author

Hazel Markou has spent 10 years involved with the “non-technical” aspects of delivering software solutions. Many of these have been fully bespoke solutions and Hazel also had responsibility for the testing teams. This makes her fully aware of the power of bespoke solution development and the fact that however skilled and experienced the development teams were, the unit testing, regression testing and first-pass user testing took significant amounts of time as the iterations of Test, Fix, Re-test were actioned. This not only increased project costs but also used to prove a logistical challenge to ensure both developers and testers were fully utilized at all times.

In recent years her experience of delivering projects based on software frameworks tools and code generators means she has seen significantly reduced testing overheads and has allowed expert focus to concentrate on the customizable elements. In situations where a business has to conform to a process dictated by “off the shelf” software Hazel believes the negatives should be realistically measured against the positives and not just ignored. However, Hazel is a great proponent of software having a place in supporting business processes as long as there are true business benefits being realised.

About Insource

A Microsoft® Gold Partner founded in 1994, Insource provides powerful Data Management, Business Intelligence, Data Warehousing and Reporting solutions. Through the effective combination of our software products, proven implementation processes and knowledge transfer we obtain a clear understanding of your organisation's business goals and then provide an end-to-end and/or modular DM and BI solutions.

Data Academy

Insource Data Academy® is a data warehouse builder which leverages the functionality of Microsoft® SQL Server™ 2005 to build robust data warehouses and marts in a fraction of the time it would take using native SQL. This is achieved through the use of intuitive, browser-based screens and data warehousing-specific modules. The resultant SQL data warehouse has full administrative functionality, it can be added to at any time; and because it develops Microsoft SQL native code it has the potential for unlimited further enhancement and customisation.

Professional Services

Business Intelligence is more than just technology. From Performance Management to KPI, data analysis, in fact just about any aspect of the organisation, it touches on everyone's business life. It involves people, both to bring BI to life and to work within it and around it. It means using the best skills available, taking the knowledge and experience from previous efforts and applying it to your situation, your systems and your procedures.

Insource Professional Services can bring the knowledge and experience from working with some of the most demanding organisations to ensure your BI objectives are met handsomely. From basic help and support to strategic advice and leadership, Insource Professional Services deliver core elements of a project throughout its lifetime - leaving you to concentrate on planning for your new, bright future.

Insource IT Consultancy Ltd

4 Southern Court, South Street, Reading, Berkshire RG1 4QS

Tel: +44 (0) 118 921 1555

Web: www.insource.co.uk

eMail: contact@insource.co.uk



Business Process and Integration
Data Management Solutions
ISV/Software Solutions