
DATA PUBLISHING ARCHITECTURE FOR THE EXTENDED ENTERPRISE

AN INSIGHTREPORT.COM WHITE PAPER

“The dual pressures of e-business productivity potential and challenging global economies are pushing executives for more rapid decision-making and hands-on analysis both inside and far beyond corporate firewalls. This being the case, a data publishing architecture allowing managers to view, create and collaborate on Web reports and data analysis has a necessary place in today’s extended enterprise.”

AUTHOR: MIKE KELLY

Mike Kelly is co-founder and Chief Technology Officer of Databeacon Inc. He brings a Ph.D., Electrical Engineering and more than ten years of related experience in the computing industry to the company. Much of this experience comes from the Computing Research Laboratory at Nortel Inc., creating innovative products utilizing the emerging technologies from industry and academia, with a focus on the design and development of client-server applications, network-based programming, and the design and creation of Java-based software applications.

mkelly@databeacon.com
Databeacon Inc.
1565 Carling Ave., Suite 300
Ottawa, Ontario, Canada, K1Z 8R1
Phone: (888) 921-8360, ext. 224
Fax: (613) 729-6711

EXECUTIVE SUMMARY

This paper describes the growing opportunity for mass deployments of data beyond traditional static reporting in today's business and governmental environments. Readers should be familiar with I.T.-centralized reporting implementations, online analytical processing (OLAP) software, as well as relational OLAP (ROLAP) and multidimensional OLAP (MOLAP) data-cube-based systems used to present structured time-series data. The paper is intended for those interested in exploring the possible benefits of more decentralized, self-service architectures that share report creation and analysis functionality to a wide user community across the extended enterprise.

The key to this type of architecture is utilizing today's Web-top and Internet Protocol (I.P.) infrastructures, including the ability to publish interactive data as a content type that bundles both the presentation and analysis layers of technology, as well as MOLAP data.

As content, these elements are downloaded into the browser in a highly compressed, self-contained environment that then scales to thousands of users. The architecture also supports mobile users who then wish to disconnect from the server entirely and take their data, self-serve reports and self-customized analysis sessions with them in their work.

**A HIGHLY
COMPRESSED,
SELF-CONTAINED
ENVIRONMENT
THAT THEN
SCALES TO
THOUSANDS OF
USERS**

TABLE OF CONTENTS

STATE OF THE ANALYTICS MARKETPLACE	1
THE CASE(S) FOR MASS DEPLOYMENT	2
THE MASS DEPLOYMENT CONUNDRUM	3
THE DATA PUBLISHING ARCHITECTURE	3
COMPARISON OF DATA PUBLISHING TO CLIENT/SERVER ARCHITECTURES	4
EXTENSIONS OF THE BASIC DATA PUBLISHING ARCHITECTURE	5
SCHEDULED CUBE BUILDING	
CUBES ON DEMAND	
DYNAMIC CUBE DEFINITION	
NETWORK SERVICES THAT EXTEND ANALYSIS	6
SOURCE DATA RETRIEVAL	
SHARED REPORTS	
CUBE-TO-CUBE NAVIGATION	
ARBITRARY LINK NAVIGATION	
COLLABORATION	8
CONCLUSION	8

INTRODUCTION

Where small numbers of experts are responsible for analyzing large amounts of data, large server-based static reporting and interactive analysis systems have clearly shown their usefulness in the enterprise. Users of such systems have access to reports and data in the same virtual space, and can perform a wide variety of analysis operations on that data to satisfy a wide variety of needs. While powerful, this architecture does not scale well to large numbers of users without significant investments in both back end server systems and end-user software. To keep I.T. costs under control while servicing enterprise needs, the answer has been to provide multi-dimensional online analytical processing (MOLAP) capabilities to a relative few “expert users”, with data analysis results distributed to wide user populations via static reports.

Where large numbers of users of different levels are to be provided with not only static reports, but the ability to analyze the data they receive, analysis computation must be transferred to client machines, which is the basis of data publishing architecture. Multi-dimensional cubes of data are accessed by the user over the Web as a form of reporting content, much like clicking on a PDF or an MP3 file. But unlike these content types, a Java-based client applet is also shipped to the user’s PC (and can then be cached for subsequent sessions) with one click on the Web report. While the user experiences the virtual data space as it is explored, he or she has full analysis capabilities on the Web-top. So rather than being *Web-enabled* with a system that requires calculations and queries to revert back to the server for processing, the user in a data publishing architecture is truly *Web-based*.

The dual pressures of e-business productivity potential and challenging global economies are pushing executives for more rapid decision-making and hands-on analysis both inside and far beyond corporate firewalls. This being the case, a data publishing architecture allowing managers to view, create and collaborate on

Web reports and data analysis has a necessary place in today’s extended enterprise. This paper will outline the operational components and data path in this architecture, going from raw data in a database to presentation and analysis by a user over the Web. It will also describe how the combination of desktop – and network – awareness of the software is used to provide additional functionality including saving and sharing reports between sessions, navigating a network of MOLAP cubes, and reaching back through to source data while doing analysis on the client.

On the server side, a design application is used to sample data and enables the data publisher to quickly create the desired multi-dimensional data structures – dimensions, measures, etc. A builder application then runs in batch mode or on request, converting data into highly compressed multi-dimensional files that are hosted on the Web server. Static or dynamic Web pages refer to the data and analysis software and can be requested by the user’s browser as Web content. In sophisticated installations, design definitions can be generated dynamically by application logic (by querying the user for data and presentation parameters) and a custom on-demand data file is delivered to the user.

STATE OF ANALYTICS/ BI MARKETPLACE

Business efficiency and business model change are two major factors in driving innovation in technology and software architecture, so it’s worth noting that the Analytics marketplace is still a healthy one relative to overall declines in I.T. spending, with Gartner Dataquest tracking 2001 growth at 7%.¹

The reason for that growth is that in tough economic times operationally-oriented, ROI-led I.T. projects get priority over “nice-to-have” (and often big-ticket) infrastructure investments. An Oct. IDC study states that

1. Biscotti, Fabrizio, and Hostmann, Bill Hostmann. ‘Business Intelligence: Software Titans Upset the Market’, Gartner Dataquest Research Brief, Aug. 15, 2002, 2-3

RATHER THAN BEING WEB-ENABLED WITH A SYSTEM THAT REVERTS BACK TO THE SERVER FOR PROCESSING, THE USER IN A DATA PUBLISHING ARCHITECTURE IS TRULY WEB-BASED

organizations that have successfully implemented and utilized analytic applications have realized returns ranging from 17% to more than 2000% with a median return on investment of 112%.² As the enterprise extends outside the corporate firewall in pursuit of better efficiencies, analytical applications are extending as well. Established vertical software vendors are talking about “information democracy”, and mass-market software companies are now offering their first analytics solutions. The realities of the extended enterprise, and of a challenging economy mean more managers are expected to deliver more and better insights to not just corporate analysts, but also to suppliers, customers, regulators and contract employees who all comprise the decision-making DNA of today’s organizations.

Today it seems, everyone has an urgent need to know more about the metrics driving their businesses. Indeed, an August I.T. survey by A.T. Kearney/Line56 Research revealed that customer analytics and content management ranked as the highest priority CRM initiative in 2003.³ The challenge is to find an analytics technology architecture that can embrace not just dozens of experts, but large populations comprised of thousands of unnamed ad hoc information consumers who increasingly understand the value and potential of analytics to help them achieve their goals.

THE CASE(S) FOR MASS DEPLOYMENT

Analytics, especially MOLAP, has traditionally been deployed to very small audiences of specialists. In most organizations, the analytics (and the data itself) is managed and accessible only to an exclusive group of users called upon to produce reports and analysis for various members of that organization. Driven from that “all-controlled-in-one-place” environment, analytics has concentrated on

working with ever larger amounts of data – only a small portion of which would ever be needed for any given task – and ever longer lists of features to simplify producing a very wide variety of reports. But as any CEO running a Web-enabled extended enterprise will attest, analytics is too valuable to keep in the hands of a select few, or to be bottlenecked by a structure that funnels everything through a small number of experts.

Mass deployment of MOLAP analytics is a natural for many organizations where data is moving broadly internally, and especially where it must be made available to a remote or external audience. E-billing applications are a good corporate example, but there are many more inside and outside the private sector:

- 1) Financial institutions providing portfolio information, market statistics, etc. to their clients;
- 2) Data services companies, market analysis companies, and a plethora of organizations whose product is data, make all kinds available (for a price) to client audiences;
- 3) Health/Medicare organizations looking to disseminate prescription cost and tracking data to users ranging from individual doctors right up to state health bureaucrats;
- 4) Governments mandated to make various types of regulatory and economic data available to citizens over the Internet.

What is common about these applications?

- 1) User populations can be very large, in the thousands or hundreds of thousands;
- 2) Users are interested in a particular slice of data, which may be their own (in the case of e-billing, for example), or may be of particular interest to them (in the case of competitive intelligence data);

2. IDC vendor-sponsored study. ‘The Financial Impact of Business Analytics’, IDC Press Release, Nov 7, 2002

3. Clare, Gary. ‘The New Three Cs of E-Business’, Line56.com, Sept 2, 2002

- 3) The users are PC-enabled and online;
- 4) Analyzing reports and data is a part-time interest to these users. They need access to reports and data analysis, but the prospect of complex software installations, training classes or consulting manuals doesn't interest them. They expect one-click gratification (as they get with other Web content from MP3s to PDFs).

THE MASS DEPLOYMENT CONUNDRUM

Web reporting and data analysis software has followed the trends of most I.T. applications. Before the desktop revolution, users accessed central computers through dumb terminals. As personal computers became more common, applications migrated to the desktop – where Online Analytical Processing (OLAP) as we know it today was really born in the early 90s. When data warehousing became popular, that data became the target of analysis applications, but it required moving the application back to the centralized computer, and so the client-server age of OLAP was born.

Web protocols have replaced proprietary client-server protocols in the current implementations of these applications, but otherwise, they have essentially stayed the same.

As is well known, scaling the number of users in a client-server environment stresses the shared resource – the server. The traditional solution is to move the server application to a bigger machine, or to layer onto it elaborate

load-balancing algorithms for running on clusters of machines. But this only pushes the boundary back incrementally when what's really needed is an architecture where computational resources grow as the demand for it grows – tied to the growth in the number of users.

THE DATA PUBLISHING ARCHITECTURE

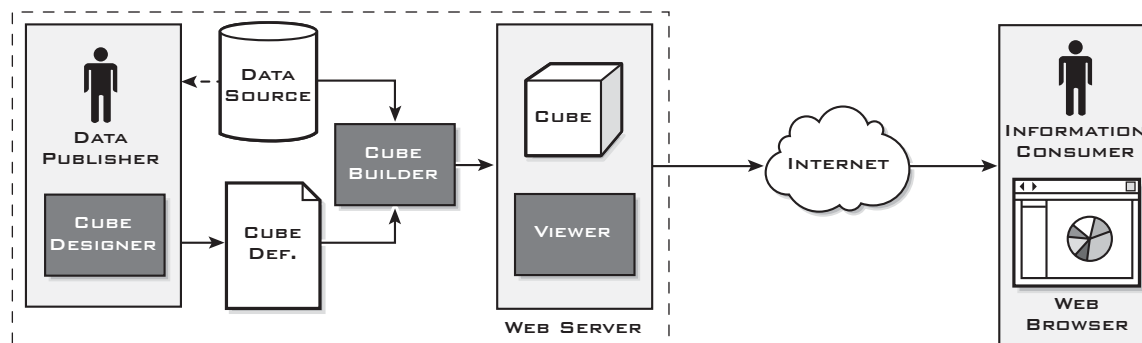
The data publishing architecture – a true Web-based architecture – is based on the premise that the work of analytics is best done on the end-user's desktop. A server should be used to deliver Web reporting and data analysis software *as content*; the server must not be a shared computational element or effective scaling is simply not possible.

The second key driver in shaping such an architecture is the nature of the Web as a means to deliver content. The data and its analytics application should reach a user in the same way – and as effortlessly – as any other Web content. It should be as easy to view as static content like text and images, and as easy to manipulate as active content like forms or video clips.

The primary components in the architecture are the designer, the builder, and the Web reporting and analysis viewer. In the simplest scenario, an administrator uses the designer to prepare a profile or definition, runs the builder, and links the resulting MOLAP cube into a Web site. The flow of data through this process is shown in figure 1.

WHAT'S REALLY NEEDED IS AN ARCHITECTURE WHERE COMPUTATIONAL RESOURCES GROW AS THE DEMAND FOR IT GROWS, THAT IS, AS THE NUMBER OF USERS GROWS.

FIGURE 1: BASIC PUBLISHING ARCHITECTURE



CUBES ARE BUILT ON DEMAND, USING PARAMETERS SPECIFIC TO THE USER WHEN THE REQUEST IS SUBMITTED

In a standard environment, published data and the Web reporting and analysis viewer are delivered to the client as would any other Web content — and causing no greater load on the server than any other Web content. Whatever the number of users, the computing power they require grows with them since the analysis is done on each user's PC.

Let's see what this means in practical terms. In total, with a Web reporting and data analysis product such as Databeacon, the Web server is responsible for transferring the 600 KB Java applet (the viewer) to the client, along with a highly compressed cube of perhaps 500 KB — the size of an Excel or Word e-mail attachment. (For an understanding of what is meant by "highly compressed" a 500 KB cube may represent 5 million source data records.)

The user now has the data in the browser and can, in fact, save it locally and launch further reporting and analysis sessions when disconnected. This contrasts with traditional client/server architectures where the data always resides on the server, and each click on a Web report or analysis operation requires fetching a new report view from the server.

In such a system, the user suffers network latency every time they interact with the data — not to mention waiting for computing time on the shared server — and cannot access or analyze data and reports when disconnected.

In a data publishing architecture the server is not used as a computing resource, but it is, of course, still available for other services that may extend the value of a Web reporting and analysis session. The user may, for example, wish to drill through to the source records behind a presented report, or share a particular view of the data with others for further exploration within a self-defined workgroup. Or the data publisher/administrator may wish to provide links from the data to other parts of the web site — possibly other MOLAP cubes — or to pages on other web sites.

COMPARISON OF DATA PUBLISHING TO CLIENT/SERVER ARCHITECTURES

With variations, meaningful Web reporting and data analysis relies on one of two basic strategies: relational OLAP, or ROLAP, and multidimensional OLAP, or MOLAP.

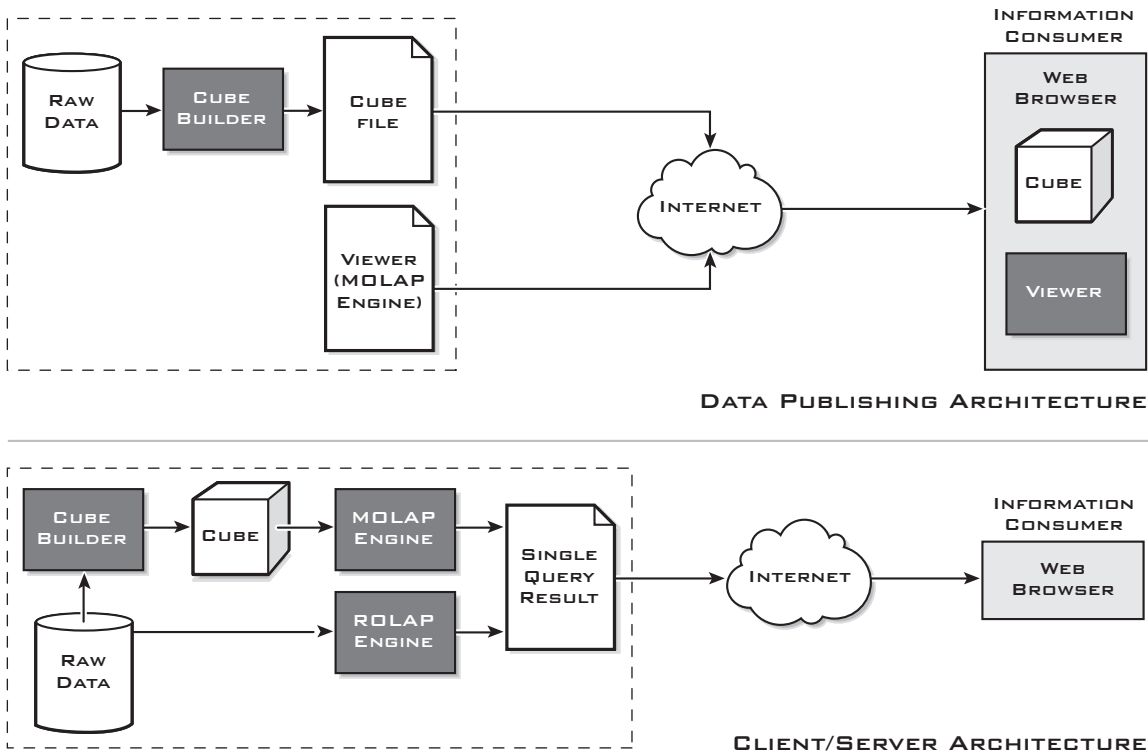
A ROLAP engine uses relational data tables directly to produce Web reports and data analysis results to satisfy a user's request. The request is converted into one or more SQL statements that are submitted to the relational sources that respond with result tables. These results are combined and formatted into the form the user expects.

In the MOLAP case, multidimensional cubes are built from relational tables based on cube definitions, and the engine uses those cubes to compute results in response to user requests.

ROLAP has the advantage of always providing completely up to date access to all available data. However, since it is always producing results from raw data, a ROLAP engine will generally be slower in responding to requests, varying from a few seconds to several minutes depending on the complexity of the search request. Also, ROLAP solutions are only available to remote users as client/server applications because it is impractical to move large databases to end users.

MOLAP has a speed and predictability of performance advantage over ROLAP, but requires cubes be periodically rebuilt to keep them as up to date as the application requires. It also doesn't scale to large data sets as well as ROLAP solutions. In its basic form in the data publishing architecture, MOLAP data will contain aggregations of raw data to the level that records can be distinguished in the MOLAP cube's virtual space. (For example, the raw data may contain several records for transactions occurring on the same day, but the cube would only have one aggregate record if its date dimension only distinguishes to the day.)

**FIGURE 2:
COMPARISON OF DATA PUBLISHING AND CLIENT/SERVER ARCHITECTURES**



But further aggregations of data may be stored in a MOLAP cube to improve performance. (From the example, the cube may also contain aggregate records for monthly values to speed accesses requiring monthly results.) But that performance gain comes at a price: The larger the data set, the higher will be the number of intermediate sums it must contain to respond well, regardless of the nature of a user request. The number of combinations of dimensions and members grows exponentially with the numbers of those elements in the cube, and so the cube can suffer a “data explosion” of intermediate sums. Hence, MOLAP solutions can become impractical for very large data sets, and where they can be used, are only available to remote users as client/server applications because it is impractical to move large data cubes to end users.

The data publishing architecture essentially offers a MOLAP solution where cube size is kept small by building cubes with subsets of

data for specific problems — those that a single user would concentrate on at any given time. It takes advantage of the compression inherent in cube formation due to record aggregation, without suffering the data explosion experienced with large data sets. The cubes can also be further compressed to reduce the cost of transporting them to the client. Commonly, megabytes of raw data are compressed into a cube of tens or hundreds of kilobytes, suitable for access using standard Web-top/IP technology.

Figure 2 shows a comparison of data publishing architecture to ROLAP and MOLAP client/server architectures.

EXTENSIONS OF THE BASIC DATA PUBLISHING ARCHITECTURE

This section describes a few data publishing architecture variations on the simple Web reporting and data analysis deployment scenario described earlier.

IN TRADITIONAL CLIENT/SERVER ARCHITECTURES THE USER SUFFERS NETWORK LATENCY EVERY TIME THEY INTERACT WITH THE DATA

SCHEDULED CUBE BUILDING

The simple scenario dealt with a monolithic block of static data. In many applications, the data changes over time, and the data is subset for different purposes. An example would be e-government demographic data which naturally changes over time, and would be subset different ways, depending on the goals of the user.

The administrator may define a small set of cube definitions, representing the different types of problems users are interested in solving. Those cube definitions can be applied to different subsets of data for different time periods, for example, or for different geographic regions. In such a case, the administrator would set up scheduled cube builds for each cube definition, with parameters for each build to cause it to use the desired subset of data.

CUBES ON DEMAND

While the scenario above makes it easy to generate the data cubes for a Web site on a fixed schedule, it may be that there are so many possible cubes that generating and storing them is unattractive. On the other hand, it may be that only an unpredictable subset of the possible cubes would be accessed during a generation cycle, so it would be wasteful to generate them all each time.

A more appropriate setup for cube building in such an environment would

have OLAP cubes built on demand, using parameters specific to (or provided by) the user when the request is submitted. This makes the transaction with the server more than just a file retrieval, but the computations involved in analyzing the data are still performed on the user's machine, maintaining the scalability of the architecture.

DYNAMIC CUBE DEFINITION

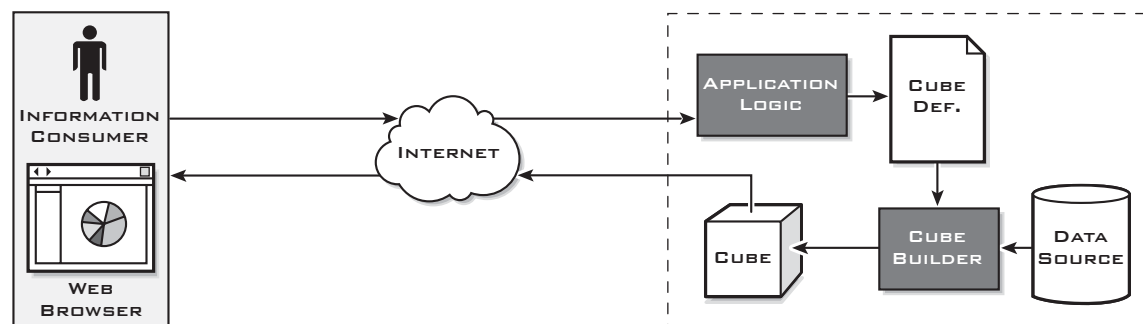
Cube definitions can be created using a scripting language, suitable for programmatic generation. Where cube requests may be so varied that it would be difficult or inefficient to create static cube definitions, they can be dynamically generated and used to create a cube on demand.

In this scenario, when a user follows a link to a cube containing Web reports and data for analysis, parameters for that cube are passed with the request. A server component (part of the Web application) generates a cube definition, submits it to the cube builder, and responds with the freshly minted OLAP cube. The parameters for the cube may be any combination of implicitly collected data as the user interacts with the Web application, explicitly collected data which the user provides in a form, and user-specific data maintained by the Web application. Figure 3 shows this scenario for dynamic cube definition and building.

NETWORK SERVICES THAT EXTEND ANALYSIS

There are a number of reasons a user may want to interact with networked components while analyzing MOLAP data within a data publishing architecture:

**FIGURE 3:
DELIVERING CUBES USING DYNAMIC CUBE DEFINITION**



- 1) Retrieve a subset of the source data for the cube being analyzed;
- 2) Save a report (the end-product of an analytics session) for another user to make use of;
- 3) Navigate to a related cube of data — maybe a more detailed view of particular element;
- 4) Navigate to a related Web page — maybe a document describing the data or the portion of it the user indicates.

SOURCE DATA RETRIEVAL

A user may want to view the source data behind a particular element presented in a Web reporting and analysis interface: a pie slice or table cell, for example. Since the source data resides on the server, this implies a server call to retrieve the data, format it, and return it to the user.

This is a natural operation in a truly Web-centric design because the Web reporting and data analysis software is aware of its network source, and can call back to it with appropriate parameters to retrieve source data. The parameters sent by the viewer applet to the server identify the cube and the user, and detailed coordinate information about the data element the user selected for the call. With this information, a server component can refer back to the original data source with selection parameters to narrow the request and user parameters so that access control can be applied. The server then has the opportunity to format the data as HTML, or Excel format and return it to the user. In the HTML case, the result can be paged back to the user, using a link on each page to request the next — keeping each request independent.

SHARED REPORTS

An important aspect of Web reporting and data analysis is the saving of views of the data as self-customized reports. A report view may be made up of a particular arrangement of dimensions on axes, sorts and filters of the data, and inserted

computations. If these reports are saved on the server, they are not only available to the user in later sessions, but can be shared with other users.

In a Web-aware application, a self-customized report formed on the client machine as the user analyzes their data cube can be named and sent to the server for storage there. That report may be marked as private to the user that created it, or public for use by all. Any user launching a Web reporting and data analysis session on the given cube — or one similar enough for given reports to apply — will retrieve previously stored reports from the server and immediately have different analysis states to choose from.

In the most common case, public views are created by an administrator, and each user has their own set of self-customized reports.

CUBE-TO-CUBE NAVIGATION

Similar to the source retrieval scenario, a user may ask to see more detail for a particular element in a Web reporting and data analysis session — but want the report result returned as a cube for further analysis. He or she may, on the other hand, wish to jump to a cube in another location in the data space altogether.

From the user's point of view, they are making a choice in the Web reporting and data analysis software interface to move to another location in the data space. The logic on the server turns the user's request into a particular cube selection/definition and returns that cube to the user. What the user sees, is a new browser window opened with the requested cube presented.

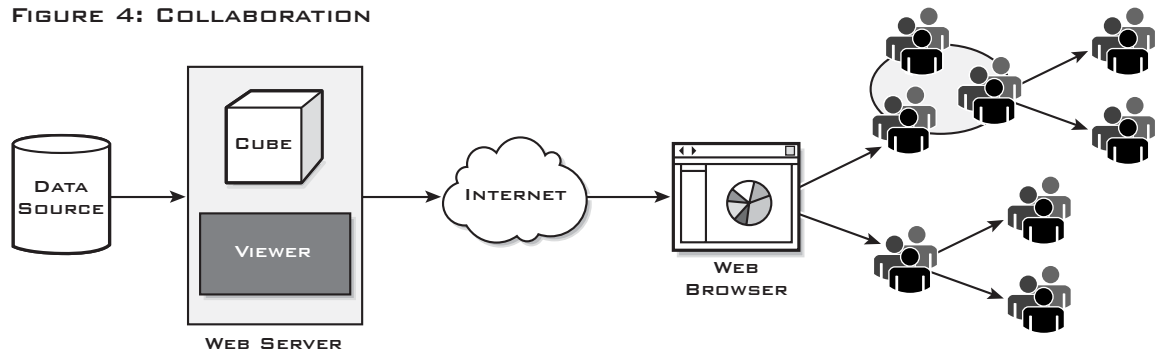
ARBITRARY LINK NAVIGATION

As a further extension to the source data retrieval and cube-to-cube capabilities,

AN IMPORTANT ASPECT OF WEB REPORTING AND DATA ANALYSIS IS THE SAVING OF VIEWS OF THE DATA AS SELF-CUSTOMIZED REPORTS

WITH NOTHING MORE THAN A BROWSER INFORMATION CONSUMERS CAN HAVE ONE-CLICK ACCESS TO ROBUST WEB REPORTING AND DATA ANALYSIS

FIGURE 4: COLLABORATION



server components can be designed to reach arbitrary services on the user's behalf. In this case, the cube and user information along with the user's coordinates in the analysis space can be used to jump them to arbitrary Web addresses (under an administrator's control), with or without the data referred to in the request. Such services could look up documents associated with the data, retrieve images or other media associated with the user's session, or generate standard reports from the requested data. The Web's the limit!

COLLABORATION

As discussed above, an important aspect of Web reporting and data analysis is the saving of self-customized reports derived from online analysis sessions. Once saved on the server, these reports are not only available to the user in later sessions, but can be shared. And because Web reporting data and analysis are delivered as content in a format similar in size to many email attachments, it's entirely viable to email a URL to that analytic content within a spontaneously created

workgroup. Not only that, if an administrator permits it, members of the workgroup can then save their own self-customized Web reports on the same OLAP cube. This makes Web reporting and data analysis an integral part of workgroup collaboration projects — the new team sport in e-business. See Figure 4.

CONCLUSION

The data publishing architecture described in this paper meets the needs of a Web-enabled extended enterprise seeking to drive decision-making throughout and beyond the corporate firewall. Equipped with nothing more than a browser and connection to the Internet, large populations of ad hoc information consumers can have one-click access to Web reporting and data analysis results that are made available as highly efficient Web content files. After that click, they can then create their own self-customized reports that can in turn be shared and modified within spontaneously created workgroups. For financial, health, data services and e-government organizations seeking the documented ROI of analytics but unable to afford the hardware infrastructure or named user license schemes of traditional client-server analytics systems, data publishing architecture is a breakthrough worthy of serious investigation.

WWW.INSIGHTREPORT.COM

© Copyright 1995-2003 Databeacon Inc.

Content from this white paper is the property of Databeacon Inc. Any reprint or reproduction in any format without permission is strictly prohibited.

Get Insight Out and Information Outreach are trademarks of Databeacon Inc. All other company and product names are trademarks or registered trademarks of their respective companies.