Evolve'2001: Pervasive Business, Sydney, Australia, 6-9 May 2001, DSTC.

Supporting Pervasive Business via Virtual Database Aggregation

John Blackham[1], Peter Grundeman[1], John Grundy[2], John Hosking[2] and Rick Mugridge[2]

[1]XSol Ltd, Parnel
Auckland, New Zealand
{ajb|peter}@xsol.co.nz
www.xsol.com

[2]Department of Computer Science, University of Auckland
Private Bag 92019, Auckland, New Zealand
{john-g|john|rick|@cs.auckland.ac.nz
www.cs.auckland.ac.nz

Abstract

Pervasive business requires information brokers that support customer/supplier enterprise system interactions in sensible ways. We present a summary of our model for pervasive business: a virtual database used to aggregate information from parts of multiple enterprise systems. Aggregated data is typically managed by a brokering enterprise in a high performance main-memory database. Replicated information from multiple customer and supplier enterprise systems is managed by the broker. This data is acquired and updated seamlessly by the broker's Enterprise Systems Logic™, using a variety of technologies to acquire the data, translate it into a canonical form and apply updates to the originating system when appropriate. We present the motivation for this novel pervasive business architecture, a review of related systems and technologies and our work to date on this project.

1. Introduction

E-commerce systems have become increasingly used by businesses to improve customer service, improve product and service delivery schedules, improve business efficiency and open up new business opportunities [24]. While Business-to-Customer (B2C) styles of E-commerce have been widely practiced in recent years, there is an increasing trend towards Business-to-Business interactions (B2B) [1, 24]. B2B E-commerce enables a business to co-operate more effectively with other businesses via the exchange of business information and co-ordination of inter-related inter-business processes.

B2B solutions typically integrate multiple business data models, IT functions and workflows across organisational boundaries [10]. Key aims in any B2B integration project include the reduction of costs associated with inter-business transactions, improvement of information accuracy and timeliness between co-operating businesses, provision of better time-to-market for business products and services, and, increasingly, the building of "virtual organisations" (loose collections of organisations co-operating to produce products and services) [1, 5, 24]. An increasing trend in B2B marketplaces is the use of "broker" organisations to link customers and suppliers of products and services. Broker enterprise systems provide single-point links between other organisations and typically have a unified business information model but dynamic business logic.

However, enabling B2B E-commerce is non-trivial [28, 10, 5]. This is for a number of reasons, chief amongst them being the highly disparate information systems used by different businesses. These system disparities occur at many levels:

- Data models: the representations of business data may be very different, and mappings between data in different organisation's systems are necessary to facilitate B2B E-commerce.
- IT functions: many data processing functions may be supported, constraining data usage in various ways. Again mapping between (some of) these is necessary to allow B2B cross-organisational workflow to be supported.
- Business workflow or logic: co-ordination of business activities can be very different across organisations, due to widely varying business models (logic) in use. Co-ordination of multiple organisations' business processes is necessary to realise effective B2B E-commerce.
- Implementation technologies: systems are built using a huge number of (often-incompatible) technologies. Data and control integration between these systems must be provided to support their effective inter-operation. This may range from very low-level, constrained data import/export to a wide range of distributed object interfaces.
- Deployment constraints: Business systems operate under constraints on their performance, security, integrity, quality of service and scalability. B2B solutions must not violate inherent constraints on system operation, but often bring additional new constraints that must be satisfied by the whole cross-organisational system.

Due to the increasing demands for B2B E-commerce, many tools and technologies have been developed in recent years to aid in such in business integration. These include database aggregation technologies, EDI messaging, XML-based information exchange, distributed object computing, inter-organisational workflow integration and Enterprise Application Integration systems [10, 1, 29, 25, 28]. However, current approaches to B2B integration are insufficient for effective cross-organisational, brokered E-commerce. In fact, as a business wants to integrate its data, functions and processes with more and more co-operating organisations, many existing approaches tend to compound the problems. Some of the key problems that need to be overcome include:

- Technology-level integration issues. Actually making computer systems talk to each other in the ways required by B2B is a challenge. Many business software systems were never written to inter-operate in the ways required by B2B E-commerce. Even when built with "open" standards, like CORBA, XML, EAI, EDI and application integration tools, a challenging amount of complex integration software must typically be built. The more business systems there are to integrate, the more difficult this problem is.
- Complexity of mapping data, control and processes across systems: Most technologies used to implement B2B integration lack general ways of specifying the mapping of data, function and business workflows between systems. Those that do, such as EDI and XSLT (data only), Virtuoso (data and limited database-stored functions), BuinessWare (data and some functions), and eXcelon (data and some functions) provide only limited tools and techniques, which only address part of the integration problem. Techniques focusing on workflow and function integration, which in pervasive business are often highly dynamic, require modification whenever integrated business systems are modified.
- Performance overheads: Many approaches to B2B integration are too slow or lack adequate quality of service characteristics to be feasible. Advanced technologies like object database caches and main-memory databases can aid performance, but don't support data and function mapping or other required technology integration facilities. Most distributed systems, especially those employing distributed transaction models, are susceptible to poor performance and quality of service (i.e. are not sufficiently fault tolerant).

We describe a project to develop a virtual database for aggregating multiple enterprise systems which avoids or at least reduces some of these problems. Typically a brokering organisation uses a high-performance main-memory database to record its own and selected parts of client/supplier business information models. The broker provides dynamically adjustable business logic, including workflow and data warehousing facilities. Data is sourced from client and supplier enterprise systems by the broker, and brokered information is accessible by clients and supplier systems. Replicated data which is updated is pushed back to the originating system, with conflict resolution strategies using broker business transactions. Configuration tools for virtual database aggregation are aimed at the business analyst, with technology-level system integration components configured and reused by the high-level tools.

2. Database Aggregation for E-commerce

Consider a simple scenario of a group of book publishers, a publishing broker and set of book customers (organisations). The broker collects publishing data in order to form a comprehensive catalogue, made available to customers and publishers. Customers place orders, made available to publishers along with customer credit. Figure 1 outlines this system.

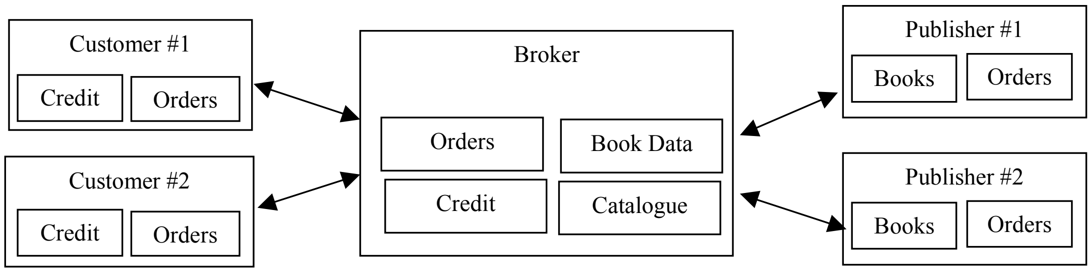


Figure 1. Simple B2B E-commerce Example.

The XSol system provides a set of facilities for defining and running business transactions based on the concept of Enterprise Systems Logic™ (ESL). Key components of ESL and hence an XSol system include:

- Business transactions. These are controlled by workflow stage specifications and specify the steps involved in carrying out input (customer->supplier) and ouput (supplier->customer) style business transaction logic. For example, this would include the steps in building orders and catalogue entries in the above example.
- Resources. These are base data used by transactions e.g. products (books), customer information, supplier information and so on.
- Transaction events. These record resources used and associated business transaction information from each stage in a business transaction e.g. quantity of product ordered, date ordered, transactional grouping e.g. multiple product order lines and so on.
- Accumulators. These are data warehousing facilities e.g. number of products sold to each customer by month, credit and product commitments and movement, and so on.

Conceptually, an XSol system has its own information management (database) containing the ESL-related data. In order to facilitate B2B E-commerce, this data is augmented by integration with multiple local and inter-organisational enterprise systems via a "virtual database". Figure 2 illustrates this systems integration concept. The virtual database may interact with other enterprise systems via database APIs, EAI/EDI messages or APIs, XML messages, distributed objects or even its own "proxy" remote agents which hide the details of the interaction mechanisms.

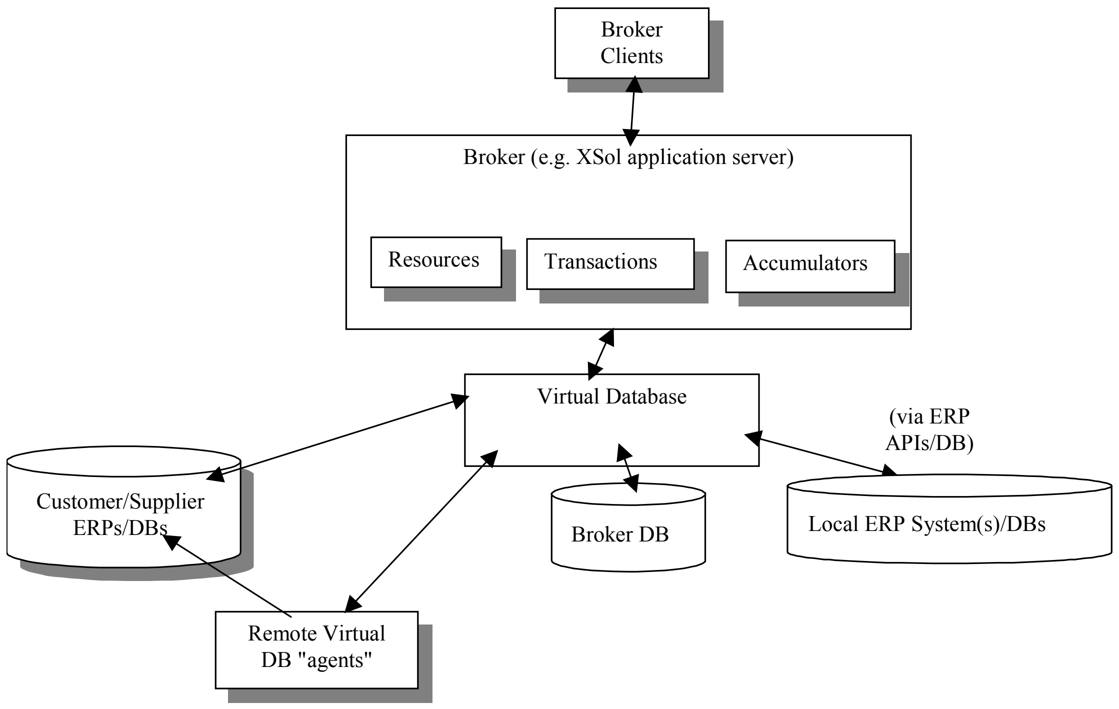


Figure 2. Enterprise Systems Integration via Virtual Database Aggregation.

We use the concept of an aggregated "virtual database" to facilitate B2B E-commerce as data exchange is the crux of brokered E-commerce. A key problem with alternative approaches, such as workflow and/or processing function integration, is that enterprise systems almost always have different workflow and processing functions, usually much more dynamic than their data. Data tends to be much more static and with clearer, simpler translations. Thus trying to integrate enterprise system workflows and/or functional logic is fraught with complexity and the potential for an explosion of inter-organisation workflow/function translations, subject to high frequency of change. With data-oriented integration, the key issues are ones of data acquiring, data translation and controlled data updates, with limited data structure and type changes to deal with.

Conceptually, a brokering enterprise system accesses client and supplier information, as well as its own, and clients and suppliers access brokered information as needed. By supporting pervasive business in this manner, client and supplier enterprise systems interact by accessing and modifying data, which is exchanged with each other via the broker. Ideally they are unaware that the "external" data they access or update, or their data that is accessed or updated, is not done by their own clients. By using an XSol system as the broker, we use ESL as a standardised mechanism for representing data, specifying data warehousing facilities and specifying broker business workflow.

In our publishing example, the XSol system is the broker with data, such as custromer and supplier information, books, orders and credit information, drawn from multiple publisher and customer ERPs/corporate databases. Data changes in customer/supplier systems are pushed to the broker e.g. when publishers change their information or add/update books and fulfill orders, and when customers place/pay for orders. Broker data, such as the catalogue, is pushed to customer and supplier systems as it is modified. Changes to replicated customer and supplier data in the broker are also pushed to customer/supplier systems as appropriate, such as placed orders, credit commitments and so on.

This approach provides for pervasive E-business in that customer, supplier and broker enterprise systems access and manipulate data with side-effects of data changes being exchanged to facilitate B2B E-commerce. This provides for seamless, data-oriented integration of enterprise systems. It also allows both the addition of new enterprises without affecting existing data aggregation relations and the seamless extension of the degree data aggregation between systems.

Key requirements for such a virtual database system that we have identified include:

- Fast (real-time) access to local (broker) and remote (customer/supplier) data by broker. This is essential to ensure efficient, high volume B2B interactions. The virtual database should give the appearance of 'live" data to the broker as well as customers and suppliers. Replication of external system data by the virtual database and the use of asynchronous external system data updates and change notifications is necessary to achieve this.
- Access to and update of customer/supplier enterprise system data via various technologies. Depending on the different technologies used to build these, this might be via database connectivity APIs, via ERP/EAI APIs, via EDI or XML-based messaging or via remote object interfaces. In some situations, virtual database agents may facilitate complex interactions, making it appear to the broker's virtual database that simple data transfers are occurring, but in actuality complex API calls are being used to achieve the effects of broker replicated data updates in the external systems. Ideally the virtual database can leverage 3rd party mapping tools effectively, especially for data access, transfer and updates.
- Synchronisation of data access and updates and suitable exception handling. Pushing replicated data updates to external enterprise systems might fail i.e. the broker has "updated" the data but subsequent, asynchronous updates on the external system fail. To resolve this, appropriate broker business transactions must be run to resolve conflicts by reversing previous transactions or running new ones to restore the broker data to a consistent state. Similarly, external systems pushing updates to broker data may be rejected by the broker enterprise system, in which case appropriate external system business transactions must be enacted by the virtual database to resolve inconsistencies.
- The XSol product allows business analysts, rather than programmers, to specify Enterprise Systems Logic (transactions, resources, data accumulations etc). Ideally business analysts will be able to specify via suitable high level tools the external systems, external data, external data mappings and conflict resolution transactions, which then deploy appropriate technologies to acquire and update enterprise systems data, to facilitate external systems integration.

The project we have embarked on aims to address the requirments discussed above, either by itself, or in combination with the XSol business modelling products being developed in parallel. The overall aim of the project is to develop the “XSol Virtual Database" (XVD). This will provide a mechanism by which data from disparate databases can be presented as a single real time dataset for use by an enterprise system and subsequently update the information in the originating databases. Key steps required involve:

- Finding a means of maintaining data integrity in data drawn from independent databases
- Finding a data access facility that enables real-time data updates to be applied back and forth between the source database and the dataset
- Finding a generic solution to these issues so that any enterprise database format can be accommodated without the need for custom programming
- Finding a means of coping with real-time dataset changes due to changes in data field format and basic data structure changes

Our approach is the use of memory based ‘virtual datasets’ whereby the information that an enterprise needs to support its business activity is derived directly from its customers’ and suppliers’ databases and presented to users as if it were in fact on their own database. This approach separates the user’s interaction with the data from the process of accessing and updating the large number and variety of customer and supplier databases. User response times will be fast and data management conflicts, such as through two users trying to update the same data field, will be managed by the secondary process, thereby buffering the user from the problem.

3. Related Work

We briefly overview systems attempting to address some of the issues related to this project. The similarities and differences to our work are discussed. Where appropriate the possibilities for using existing software to support some of this project's aims are highlighted. However, to our knowledge no system currently exists that offers the range of functionality of the XVD: high speed data management with seamlessly integrated external data, specified and maintained using abstract techniques.

Database Integration Technologies

Various technologies have been developed to assist heterogeneous database integration. Commonly used are database connection technologies, such as ODBC and ADO. These are available on many platforms and allow access to data from a wide range of sources (RDBMS, flat file, OODBMS). They are, however, very low-level, programmatic techniques. These approaches are often used to support multi-database integration, but only provide basic connectivity. Some RDBMS (and OODBMS) servers support multiple database connectivity via a single entry point e.g. MS SQL Server, Oracle and ObjectStore [6, 18]. Again, these are only basic connectivity mechanisms, with the programmers left to manage distributed transactions and data mapping themselves.

Some database integration techniques make use of distributed object computing technologies like DCOM and CORBA, or "agent-based" techniques [19, 22, 25]. All of these approaches are inherently programmatic and relatively low-level: programmers must design and implement code to connect to distributed systems, exchange data, map data into different forms, invoke operations, and so on. Evolution of systems, a necessary requirement for flexible B2B solutions, then becomes difficult. Heterogeneous database integration techniques and federated database systems attempt to make multiple databases look and behave as one. However this is usually restricted to databases under complete control of one organisation [33, 15, 16]. For example, Virtuoso [10] allows developers to integrate multiple, distributed databases and access them through their native or ODBC interfaces as it they were one database. While limited caching is employed together with data migration strategies, the external data is still external in terms of its ultimate look and feel as it is slower than local data and subject to remote database constraints.

While these basic programming techniques allow an application to access multiple data sources, they provide no support for solving high-level B2B integration problems. Access is usually slow and often infeasible across organisations, due to the layering of business logic on top of low-level data sources and the slow nature of distributed transaction models. It is almost certain the XVD will leverage some of these technologies to achieve basic sourcing of data and invocation of external functions. The XVD will provide multiple layers on top to allow users to:

- specify external system data sources
- locally replicate external data
- translate local updates into remote updates and vice-versa, and
- specify data (and operation) mappings in abstract ways.

A key XVD feature is the common look-and-feel of the integrated and local data. External data aggregated using the preceding approaches either is explicitly "different" to local data, or behaves differently when used (eg slower to access/update or is read-only). This makes the development and evolution of sophisticated B2B systems much more challenging but importantly, makes the performance of such systems unacceptable. Replicated external data in the XVD also supports a high degree of fault tolerance, allowing an organisation to continue carrying out business functions even if some partners are off-line. Most systems described above are pessimistic and avoid data replication, ensuring integrity at the expense of performance and quality of service. The XVD will use import/export-based approaches to exchange data with external systems. Replication of data will ensure performance and quality of service. An optimistic approach will be used to manage data integrity, using import/export conflict resolution and exception handling.

Data Exchange Formats

Many data exchange formats and technologies have been developed to assist systems integration. EDI is an umbrella term for a wide range of interchange formats ranging from specialised, e.g. HL7 [9] for Health Informatics, to EXPRESS [11] for object-based data exchanges, and to XML [31] for very general purpose data exchange. These provide various technologies (APIs and support languages/tools) and data encoding approaches (transport-level encoding, record-based, or object-based) to assist with complex systems integration at the data level. However, they are usually very slow across organisations (and within organisations). Data translations are complex to specify, and are often done programmatically. Some work has been done to provide more abstract, declarative data translation schemes, such as XSLT [32]. These are limited in the range of data translations they support, usually to mapping data between quite similar structures. Most existing data integration approaches tend to better-suit batch-style integration rather than "real-time" systems integration.

The XVD is likely to use many of these technologies and we aim to leverage where possible third-party tools to facilitate data exchange. However, the XVD will hide the particular external data format/nature from local data and will replicate the external data for high-speed local access. Developers will be provided with abstract external data specification and sourcing tools, used to define external data (and object/API) structures, external system data import/export functions and transaction models. These are then used by the XVD to maintain local forms of external business data synchronised with their external form via tailorable mechanisms.

Distributed Object Systems

A variety of distributed systems integration standards and technologies have been developed, due to the need for interoperation of multiple systems above the data exchange level. These include EAI approaches [28], COM and MTS [25], CORBA and OTS [19], Enterprise Java Beans [12], and CORBA services and components [20]. Various systems integration projects have used these technologies to assist with distributed systems integration [2, 22, 33]. However, these techniques are programmatic i.e. developers build software to provide remote system interfaces that are invoked to exchange data and control. EJBs hide transaction, security and other distributed system management features from component developers, but use an OTS transaction model and ODBC-based data integration model. Distributed transaction models like MTS and OTS are "pessimistic" in that they obtain local transactions on each distributed system and commit data across these systems using a (typically very) slow 2-phase commit protocol. Performance analysis of MTS and OTS-based distributed systems [8] show enormous performance degradation when only 2-3 machines (on high-speed local area networks) are used. Inter-organisational data exchange using such approaches would perform several orders of magnitude worse again. While such techniques have been used to successfully integrate data from disparate sources [22], they provide far less-than-ideal integration platforms for B2B systems inter-operation.

While the XVD will make use of some of these technologies, particularly DCOM and MTS to support "traditional" distributed transactions, it will 1) hide such distributed data and function access from the local application server; and 2) use an optimistic data management approach. This will focus on export/import of aggregated external system data rather than single external data accesses and updates using a distributed transaction model. The XVD will need to combine data mapping techniques with (limited) distributed object integration techniques to support integration of the XVD with distributed object-based systems. We envisage that a single, seamless external system-mapping environment will be provided to users of the XVD to achieve this integration. This will emphasise declarative specifications, rather than programmatic object integration or limited, declarative data-only translations. Declarative transaction demarcation, as used in the Enterprise Java Beans deployment descriptors, perhaps combined with workflow stage identification, will be used to co-ordinate external system data import/export.

Object and Main-memory Databases

Alternatives to traditional Relational Database (RDBMS) solutions exist, including:

- Entity-relationship (ER) databases (eg PCTE [30] and ERDB [3]);
- High-performance local object stores and persistent languages (eg PSE, PJava and Napier);
- OODBMSs (eg Versant, ObjectStore [6] and GemStone [7]); and
- Main-memory databases (MMDBs) (eg ERDB [3] and TimesTen [27]).

Object stores and databases provide object-based data management mechanisms, and often allow developers to manage persistent objects in efficient ways (via object caching and close-to-memory representations) that relational techniques support less-well [7, 21]. It is possible the XVD will leverage an object store or database to manage local data and replicated data efficiently. MMDBs hold all data in memory, with a secondary storage mirror [14, 26]. They utilise memory-optimised indexing structures for very high performance [17]. The XVD will incorporate a MMDB for high speed data access and update. Unfortunately all commercial MMDBs we are aware of are enormously expensive and the XVD will aim to provide a simple yet powerful MMDB without making the product as a whole unaffordable for small and medium-size businesses. Some interesting techniques have been developed to unify data management, such as with PCTE, CORBA and MultiView for software tools [4, 13]. These use a "canonical" form of data with multiple views to facilitate distributed data access and management. Some of these basic ideas may be utilised in the XVD.

It should be noted that while performance of the XVD-based data is important, in order to make feasible enterprise-wide utilisation of multiple external organistions' data, it is secondary to the convenience of the XVD's support for external data specification, translation and organisation facilities. Thus while the XVD will utilise main-memory database techniques to provide high-speed access and update of localised data, with incremental import/export-based integration of external data, the apparent seamless integration of the disparate data sources to the local enterprise application server will be its primary achievement.

ERP and B2B Integration Solutions

Packaged software i.e. ERP systems aim to obviate the need to integrate enterprise software by providing all an organisation needs in terms of IT functionality. They also typically provide various integration mechanisms for accessing and exporting data. These mechanisms are one (or more) of the preceeding approaches: ODBC/ADO database access; XML/EDI data exchange formats; EAI-based API calls, distributed object models; and/or various data import/export facilities. In addition, several products have been developed to integrate distributed systems in the B2B model. Examples include the eXcelon Integration Server [5] and the Vitria BusinessWare integration solution [28]. It is unclear exactly what degree of systems integration such packages provide, and how much is programmatic, but all claim to include XML/EDI, EAI and distributed objects (COM/CORBA) and transactions (MTS/OTS). Some, like eXcelon, provide tools to enable cross-organisational workflow and basic data translations to be specified. Virtual database products like Virtuoso [] focus on transport-level database integration. The XVD may well utilise such products to provide data sourcing/export. However it must provide higher-level data and function mapping specifications for ease of use and system evolution, memory-managed and replicated external data for performance and quality of service, and customisable conflict resolution strategies for distributed data integrity.

The XVD will provide high-level data mapping specification tools but aim to abstract such specifications further from underlying technologies and data representation techniques than do competitive products. It will also use main-memory data management on local and remote systems to provide high performance, optimistic data access and update strategies. It is likely the XVD will leverage similar technology-level facilities to other products, however, and may be able to make use of one or more of these products to facilitate its database aggregation. However, it should be noted again that these are typically extremely expensive products (in their most general forms) which are out of reach of most small-to-medium businesses. Where a corporate database to be integrated with the XVD is an ERP system with its own custom integration facilities, these will be leveraged by the XVD to import/export XVD-managed data. The XVD is designed to leverage existing (and new) database integration, distributed object, EAI/EDI and other integration technologies rather than provide its own custom techniques or implementations.

Summary of XVD Contributions

No complete B2B solution exists that is fast (near-real time data management), seamless (all data, whether local or remote, looks and behaves the same to users), abstractly specified (avoiding programming-level constructs for users) and non-proprietary (not tied to particular vendor databases, APIs, ERPs etc). The key differences of the XVD concept to existing systems include:

- A main-memory database holding local and remote, replicated data will be used for optimum performance and seamless data access and update.
- High-level data and business constraint mapping tools will isolate users from particular technologies and provide easily extensible, non-programmatic B2B integration facilities.
- A variety of transport-level, interface-level and distributed transaction support models will provide abstract data management to actual data representation integration. It is planned to leverage existing techniques and products where ever possible, and to ensure the XVD has an open architecture for integrating new transport-level integration services.
- External systems will run an XVD "proxy" with its own MMDB, managing external data sourcing and update exchange
- Local and distributed systems will use MMDB to MMDB data exchange for optimum performance.
- An emphasis is on optimistic transaction processing via import/export mechanisms with external systems vs pessimistic distributed transactions. High-level specification of conflict resolution strategies will be supported in the XVD, and suitable APIs provided to application servers using the XVD to detect and manage such conflicts.

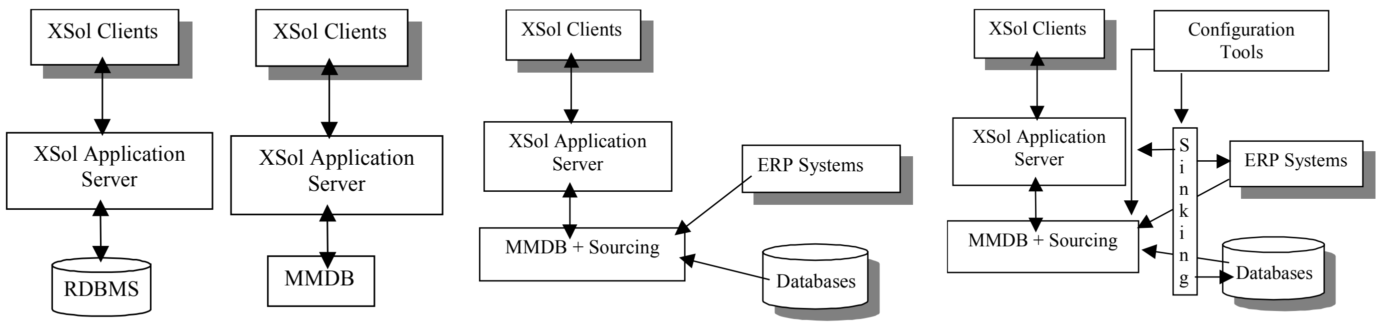
Where possible, the XVD will incorporate existing technologies and tools, but be "generic" in the sense that the widest possible range of solutions from different vendors will be supported.

4. XSol Virtual Database Development

We have identified four key steps in our XVD development, incorporating:

1. A MMDB for high speed broker data management, including external systems replicated data management
2. Data acquisition via various 3rd party technologies from external systems, including basic data and data operation mapping to and from the broker data representational format
3. External system data update and exception handling. This will include more sophisticated data acquisition and translation facilities.
4. Various high-level tools for business analysts to use to specify and configure systems integration with our virtual database technology.

While we are initially using the XSol application server as a "first client" of the XVD, we also aim to generalise it for use by 3rd party application servers. Figure 3 shows the steps in developing the XVD for XSol applications.



(a) XSol System (b) Level 1 XVD (c) Level 2 XVD (d) Level 3 XVD

Figure 3. Steps in Developing the XVD Integration System.

To date we have developed a high-performance MMDB prototype for the XSol application server as a "Level 1" XVD. This provides a 20-30 times performance speed-up on using a relational database management system for local data access and update. This MMDB was integrated with the XSol application server via an object-to-relational mapping layer, isolating the application server from the change in data management. We are extending the mapping layer and MMDB to facilitate ACID transaction management, multi-threaded XSol application server accesses, and tracking of externally-sourced data.

We are currently designing a "Level 2" XVD which incorporates a basic "data pump"-style facility. This will provide the ability to:

- Access data stored in external enterprise systems using a variety of 3rd party data access methods
- Perform simple data translations on external system data to map it into XSol's ESL-based representational formats
- Detect and push updates made to external system data into the XSol XVD, using a variety of batch and incremental techniques

In conjunction with this Level 2 XVD we are developing basic tools to specify external system technology characteristics, data formats, data translations to XVD data representation and external data change detection and refresh facilities. These will make use of various 3rd party data access technologies but be configurable by business analysts (the typical users of XSol's specification tools) to enable them to specify and modify external data sourcing.

Subsquently we will extend the XVD to "Level 3", initially pushing data into external enterprise systems for access in limited ways (e.g. for data analysis). This will be extended to pushing data changes to data owned by the external systems (i.e. is accessed/updated within the external systems). At the simplest level this may run basic database-style updates but at more complex levels this will involve external agents translating data changes into API calls, message formats etc. via complex interactions with the external enterprise systems. Conflicts that may occur e.g. failed updates or rejected updates, will be detected and the XVD will notify the XSol application server that resolution business transactions need to be run to reverse or correct previously made transactions. Tools for business analysts to use to specify this data exporting and conflict resolution will be developed.

5. Summary

This project aims to develop a new product suite to facilitate pervasive Business-to-Business E-commerce. This uses a combination of synthesis of existing technologies, such as open systems integration, main-memory databases and data synchronisation algorithms, along with new basic research into abstract data and data operation mapping techniques and tools. Success of the project will be measured by a combination of new enabling technologies for B2B E-commerce, along with new product lines realising this enterprise systems integration. Progress to date has included establishing the key characteristics of such an enabling technology, the Xsol Virtual Database, identifying strengths and weaknesses in current B2B integration products, developing a high-speed local virtual database, and design of data acquisition and synchronisation mechanisms. We are working on implementing these mechanisms, developing external data updating mechanisms, and developing tool support for developers and business analysts to expedite enterprise systems integration via our virtual database aggregation technology.

Acknowledgements

Funding for parts of this work from the New Zealand New Economy Research Fund is gratefully acknowledged.

References

1. Alonso G, Fiedler U, Hagen C, Lazcano A, Schuldt H, Weiler N. WISE: business to business e-commerce. Proceedings Ninth International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises. RIDE-VE'99. IEEE CS Press, 1999, pp.132-9. Los Alamitos, CA, USA.
2. Aleksy M, Schader M, Tapper C. Interoperability and interchangeability of middleware components in a three-tier CORBA-environment-state of the art. Proceedings Third International Enterprise Distributed Object Computing. Conference, IEEE CS Press. 1999, pp.204-13. Piscataway, NJ, USA.
3. Automated Technology Europe Inc, ERDB Entity-Relational Database, www.entity-relationship.com.
4. Emmerich, W., "CORBA and ODBMSs in Viewpoint Development Environment Architectures.," in Proceedings of the 4th International Conference on Object-Oriented Information Systems, Springer Verlag, 1997, pp. 347-360.
5. eXcelon Corp, eXcelon B2B Integration Server White Paper, www.exceloncorp.com.
6. eXcelon, ObjectStore white Paper, www.excelon.com.
7. GemStone, GemStone/J Application Server Persistent Caching, www.gemstone.com.

8. Gorton, I. Et al. Enterprise Middleware Evaluation Reports, Advanced Distributed Systems and technologies group, CSIRO, http://www.cmis.csiro.au/adsat/publications.htm.
9. HL7 Organisation, HL7 Standard Interchange Format, www.hl7.org.
10. Idenhen, K., Introducing OpenLink Virtuoso: Universal Data Access Without Boundaries, White paper, www.openlinksw.com.
11. ISO/TC184, Part 11: The EXPRESS Language Reference Manual in Industrial automation systems and integration - Product data representation and exchange, Draft International Standard, ISO-IEC, Geneva, Switzerland, ISO DIS 10303-11, August, (1992).
12. JavaSoft, Enterprise JavaBeans Technology, www.javasoft.com.
13. Kelter, U., Monecke, M. and Platz, D. Constructing Distributed SDEs using an Active Repository, in Proceedings of the 1st International Symposium on Constructing Software Engineering Tools, Los Angeles, 17-18 May 1999, University of South Australia, Australia, pp. 149-157.
14. Kim, G.B. Hae-Ybung Bae. Design and implementation of a query processor for real-time main memory database systems. Journal of Korean Information Sciences Society, vol.6, no.2, April 2000, pp.113-19. Publisher: Korea Inf. Sci. Soc, South Korea.
15. Li, W.S. Clifton C, Shu-Yao Liu. Database integration using neural networks: Implementation and experiences. Knowledge & Information Systems, vol.2, no.1, 2000, pp.73-96. Publisher: Springer-Verlag, UK.
16. Lim, E.P.Chiang RHL. The integration of relationship instances from heterogeneous databases. Decision Support Systems, vol.29, no.2, Aug. 2000, pp.153-67. Publisher: Elsevier, Netherlands.
17. Lu, H., Yuet Yeung Ng, Zengping Tian. T-tree or B-tree: main memory database index structure revisited. Proceedings 11th Australasian Database Conference. ADC 2000, IEEE CS Press, 1999, pp.65-73. Los Alamitos, CA, USA.
18. Microsoft Corp., Microsoft Backoffice: SQL Server 7 White Paper, www.microsoft.com.
19. Mowbray, T.J., Ruh, W.A. Inside Corba: Distributed Object Standards and Applications, Addison-Wesley, 1997.
20. OMG Organisation, CORBA: Common Object Services, www.omg.org.
21. PowerTier, White paper: PowerTier 5 for C++, www.persistence.com.
22. Purvis, M. et al, New Zealand Distributed Information Systems Project, www.dis.otago.ac.nz.
23. Secant Corp, Secant Extreme Persistent Object Service White Paper, www.secant.com.
24. Senn JA. The evolution of business-to-business commerce models: the influence of new information technology models. Proceedings of International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems. IEEE CS Press, 1999, pp.153-8. Piscataway, NJ, USA.
25. Sessions, R. COM and DCOM: Microsoft's vision for distributed objects, John Wiley & Sons 1998.
26. Song, E.M., Young-Keol Kim, Chanho Ryu, Mi-Sun Choi, Young-Kuk Kim, Seong-Il Jin, Mi-Kyong Han, Wan Choi. No-log recovery mechanism using stable memory for real-time main memory database systems. Proceedings Sixth International Conference on Real-Time Computing Systems and Applications, IEEE CS Press, 1999, pp.428-31. Los Alamitos, CA, USA.
27. TimesTen Inc, TimesTen Main Memory Database White Paper, www.timesten.com.
28. Vitria Technolgy Inc, Vitria BusinessWare White Paper, www.vitria.com.
29. Uchoa EMA, Melo RN. HEROS: a framework for heterogeneous database systems integration. Database and Expert Systems Applications. 10th International Conference, DEXA'99, Lecture Notes in Computer Science Vol.1677, Springer-Verlag. 1999, pp.656-67. Berlin, Germany.
30. Wakeman, L. and Jowett, J. *PCTE the standard for open repositories*, Prentice-Hall, 1993.
31. WC3 Consortium, XML: eXtensible Markup Language format, www.xml.org
32. WC3 Consortium, XSLT: XML Stylesheet LanguageTranslations, www.xml.org.
33. Wu, E.. A CORBA-based architecture for integrating distributed and heterogeneous databases. Proceedings Fifth IEEE International Conference on Engineering of Complex Computer Systems, IEEE CS Press, 1999, pp.143-52. Los Alamitos, CA, USA.