



# Enabling New Business Paradigms with the Xpiori XMS

by Chris Brandin

Release 1.1

**Xpiori, LLC**  
2864 S. Circle Dr.  
Ste. 1200  
Colorado Springs, CO 80906  
(719) 527-1315  
[www.xpiori.com](http://www.xpiori.com)

© 2007 by Xpiori, LLC. All rights reserved.

**Version 1.1**

**Copyright Xpiori, LLC All Rights Reserved**

**Xpiori technology is protected by the following patents:**

**US Patent #5,742,611 (21 Apr 98)**

**US Patent #5,942,002 (8 Aug 99)**

**US Patent #6,157,617 (5 Dec 00)**

**US Patent #6,167,400 (26 Dec 00)**

**US Patent #6,324,636 (27 Nov 01)**

**US Patent #6,493,813 (10 Dec 02)**

**US Patent #6,792,428 (14 Sept 04)**

**Other U.S. and international patents pending.**

**The information in this white paper has been provided by Xpiori, LLC. To the best knowledge of Xpiori, it contains information concerning the current state of information processing technology. Xpiori, LLC disclaims any and all liabilities for and makes no warranties, expressed or implied, with respect to products described in this paper, including, without limitation, the implied warranties of merchantability and fitness for a particular purpose. No specific reliance should be made on the material provided herein without thorough investigation of the technology and its proposed application to specific circumstances. Product and technology information is subject to change without notice.**

## Introduction

XML has garnered almost universal acceptance because it offers a standard for the information infrastructure that supports the increasingly dynamic ways business is being conducted. XML promises an unprecedented degree of flexibility, and the world of eCommerce is responding.

The Xpiori XML Information server was designed, from the ground up, to extend the flexibility of XML to the storage and management of information. This paper endeavors to describe architectural features of the Xpiori XML server from a business solutions perspective.

## The Nature of Information

The words “data” and “information” are often used interchangeably. In fact, the word “data” describes a subset of information – the half devoid of context. If I simply say the word “blue” I have given you data, but no information - because you don’t know what I mean. If I say “my car is blue”, I have given you information because the data is accompanied by context (or “metadata”). Information is data and metadata combined.

XML is the first widely adopted standard for expressing information, as opposed to just data. In fact, XML requires that all data elements be accompanied with tags (metadata). This precipitated the immediate acceptance of XML as the preferred transport mechanism for data interchange. Also, system designers have begun to create applications that capitalize on the flexibility of XML in order to accommodate new ways of doing business. Unfortunately, once all this XML information has to be stored and managed somewhere, flexibility evaporates - the product of the system-wide least common denominator (usually the database management system). What is needed is a true information management system that fully supports the flexibility promised by XML.

## The Xpiori XML Information Base Server

Traditional Databases were designed to manage data by creating a static framework to contain dynamic data. That is, data elements can be managed as long as all metadata has been defined in advance. This solution falls short of true information management by half. The Xpiori solution was designed to manage metadata in exactly the same dynamic manner that was heretofore only possible for the data component of information. This offers two significant (if not profound) advantages to the system designer: First, constraints on the use of dynamic data types are removed, offering application developers unlimited flexibility. Second, the processes of database design and configuration (usually the most labor and time consuming aspect of system design) are reduced to practically nothing.

The Xpiori XML server was designed to achieve the following goals:

- **Dynamic management of metadata.** Traditional DBMSs are ill suited for this because they are, at their core, static frameworks of metadata for dynamic data. Data elements are arranged in predefined rows and columns and indexed according to predefined rules. Even object relational systems require data objects to be arranged and indexed according to predefined parameters. Xpiori took an entirely novel approach, drawing on pattern recognition techniques rather than traditional notions of data management. All information is represented in an internal format called “Information Couplets” – pairings of data and complete metadata. Information Couplets are treated as patterns, and those patterns can be arbitrary. Consequently, there are no rows or columns, and there is no need to predefine indices. Also, patterns can be as common or

as unique as desired. A piece of information, for example, can be associated with a single fragment of information without having to be predefined, pre-allocated, or added to another information fragment. This also means that an entirely new data type can be added to an application at any time without having to do anything to a database at all.

- **Immediate availability of information.** Information is “indexed” as soon as it is posted. Actually, there is no separate indexing process because there is no need to specify what should be indexed. The Xpiori XML server automatically generates data patterns that allow information to be retrieved based on fully, or partially qualified queries. Complex queries are accommodated through a process called “Vector Annealing”, which quickly converges sets of individual pattern matches to locate information fragments based on multiple criteria. The Vector Annealing process operates on any information that has been posted, requiring no predefinition of any sort.
- **An information-centric view.** Traditional DBMSs offer a data-centric view of information; that is, they do not dynamically manage metadata. “Native XML” (a complete misnomer) products offer a “document-centric” view of information. These are descriptions of limitations, not features. Both models impose serious obstacles to managing heterogeneous information in a dynamic way. The disadvantages of data-centric solutions have already been discussed. For many applications (like workflow management, or customer relations management), document-centric solutions can be even worse. Systems that adhere to this model always deal with XML documents in their totality, making sub-document manipulations extremely inefficient. For example; in order to change a data element, the entire XML document is loaded into a DOM (Document Object Model), modified, and then re-posted in its entirety, replacing the previous version of the document. This amounts to throwing the baby out with the bathwater. Arguably, one should be able to do four basic things with any data repository mechanism: put information in, get it back out, get rid of it, and change it. Fantastically, document-centric systems can’t really achieve these four basic functions because, for all practical purposes, they cannot insert, change, or delete information fragments. The Xpiori XML server treats documents as aggregations of information. No restrictions on how much information is returned or modified are imposed. New information can be inserted or deleted at will without any re-processing of documents. It was considered critically important that the Xpiori XML server be suitable for all information management tasks that lend themselves to XML.
- **Schema independence.** The Xpiori XML server was designed to be oblivious to schemas or DTDs. This is a more important feature than it may seem at first blush. Most XML data management systems require that all XML information be described by a schema or DTD for data mapping reasons. Some products claim schema independence; but performance degrades to unacceptable levels in that mode of operation. Another problem schema dependence imposes is that it destroys the ability to have heterogeneous data within similar document types. For example, suppose you want to add a new field to some, but not all, documents of the same type. How will the database know which schema applies unless a new schema is supplied? How will it know whether all the other documents of the same type need to be changed, or whether they should be treated as different document types? How will other applications know about this new document type? Ultimately, schema dependence makes it virtually impossible to use XML’s most attractive feature – its extensibility. Another advantage of schema independence is that there is no database design that needs to be done, and no penalty is imposed for change. Schema independence was considered critically important for the Xpiori XML server. To do otherwise would have substantially defeated the purpose, and advantage, of XML.

- **Scalability.** There are many XML solutions for processing small, single documents. The Xpriori XML server was designed to manage huge repositories (terabytes) of XML documents of all types. This was achieved by treating XML documents as aggregations of information. The Xpriori XML server is aware of, but functionally oblivious to, the document-centric structure of XML information. XML data management systems are notorious for not scaling well – both as document size increases, and as the number of documents increases. The Xpriori XML server was designed to seamlessly scale to very large information management requirements, exhibiting remarkably flat performance as system size increases. Individual document size has no bearing on performance
- **Information structure independence.** This is an issue that most system designers don't consider until it's too late. For example, consider the following: There are two groups of XML documents containing the same information – telephone number listings. One is a single document containing all listings, and the other consists of separate XML documents for each listing. Which should be used? With document-centric systems, performance degrades considerably, and storage space increases, if the single document is used. Furthermore, because modifications are made on a document level, maintenance tasks become utterly unwieldy. In other words, the behavior of the database is profoundly affected by the nature of information, irrespective of whether it has been characterized by a schema or not. There may be many reasons to choose ways to structure information, but having to accommodate a database system should not be one of them. The Xpriori XML server was designed to treat data structure indifferently – it makes no difference whatsoever.
- **Efficient use of storage.** The use of Information Couplets and Vector Annealing makes it possible to store XML information very efficiently. Information Couplets map directly into XML, thereby eliminating inefficiencies associated with mapping between information domains. Information Couplets represent a fundamental means of storing and managing information, which constitutes a general case means for containing information of which XML is a subset. Digital Pattern Processing (DPP) and symbolic processing are used to maximize efficiency. The upshot is that the amount of storage used, for everything combined, adds up to between 1x and 2x the size of the XML documents alone. This includes the documents, all indices, access control information – everything. This compares very favorably with all other methods of managing information, requiring less than half (or less) the storage of any DBMS, and a tiny fraction of the space used by DOMs.
- **Speed.** The Xpriori XML server was designed, from the onset, to be very fast. This was largely achieved through the use of DPP, Information Couplets, and Vector Annealing; rather than by attempting to use techniques that were developed before XML existed. By focusing on the nature of XML, instead of prior methods of managing data, Xpriori was able to tailor their server precisely to the challenges posed by the nature of self describing, heterogeneous information – a very different paradigm than existed when older data management methods were developed. The result is a server that outruns any networks' ability to transport data - by a wide margin. A RDBMS, for example, can process in the range of 1-10 XML transactions per second. Typical "Native" XML databases can process, in the best-case scenario, about 25 XML transactions per second. A fast Ethernet connection saturates at about 1000 simple XML transactions per second. The Xpriori XML server can process tens-of-thousands of XML transactions per second, meaning that it will be able to keep up with any networks' ability to keep it busy – now, and in the future.

- **Ease of use.** Because the Xpiori XML server is schema independent, no “database design” is necessary. All that’s left is deciding how big the data repository should be and where you want to put it. One would be hard pressed to make things any easier. When contrasted to the often-monumental project designing, configuring and changing a database management system can be, the ease of using the Xpiori XML server becomes one of its most attractive feature.
- **Flexibility.** All the features discussed above lead to flexibility. The Xpiori XML server allows applications to be developed directly to achieve a business objective. Rather than having to consider the limitations imposed by data management systems, the system designer has the flexibility to build the best possible solution – the freedom to use XML to its full potential and change things when needed. By now, it should be apparent that this kind of flexibility would be unachievable using old methods of information management. The key to developing a server to accommodate these features was to correctly identify them in the first place. Imagine if computers had been used to build increasingly effective ways to control linotype machines. Instead, better ways to set and print newspapers were devised. Although DBMSs remain an effective way to manage data, they are ill suited to managing XML information.

### Cooperation with legacy systems

Although the Xpiori XML server is an ideal means for managing XML information, the fact remains that conventional DBMSs contain nearly all of the worlds’ information. For new information systems, data may only need to be imported. For other systems, conventional DBMSs will be a part of a larger, XML based solution. Xpiori has a product that is used to map, import, and export data from DBMSs. Also, triggers can be defined to cause data to be synchronized with DBMSs based on user-defined events. This allows the Xpiori XML server to be used in a variety of ways:

- The Xpiori XML server can be used as the “front-line” repository, periodically synchronizing with DBMSs.
- The Xpiori XML server can act as a cache, synchronizing with DBMSs in real-time.
- The Xpiori XML server can synchronize with DBMSs with data being managed by them, and manage other information (not currently in the DBMSs) itself. Later, when the new information has been characterized for the DBMSs, it can be transferred.
- The Xpiori XML server can manage all information at first - when it is dynamic, and post summaries, infrequently used data, or archive data to DBMSs.

## Conclusion

The overriding objective Xpiori had while designing its XML server was to thoroughly understand the meaning of XML, the effect it would have on the way business would be transacted, and to build an information management system specifically tailored to XML. A tremendous amount of time and effort has been expended in the industry to migrate to XML, only to be stalled by the inadequacy of data management systems. The woes suffered by many early providers of XML-based ecommerce solutions are prototypical of the conundrum later adopters of XML will find themselves in as they deploy their XML-based products. Incredibly, the industry has moved forward to adopt XML without regard to whether there is a viable way to contain, and persist all the XML information being produced. Xpiori has, from the onset, endeavored to offer a solution to this problem; while others attempt to find a way to make old methods fit. In the future, Xpiori will certainly have competition, but not from those attempting to apply old technologies to XML. Currently, Xpiori has the only product that offers features that allow XML to be used to its full potential.

# # #