



# Tamino

## XML Server

---

WHITE PAPER

## Contents

---

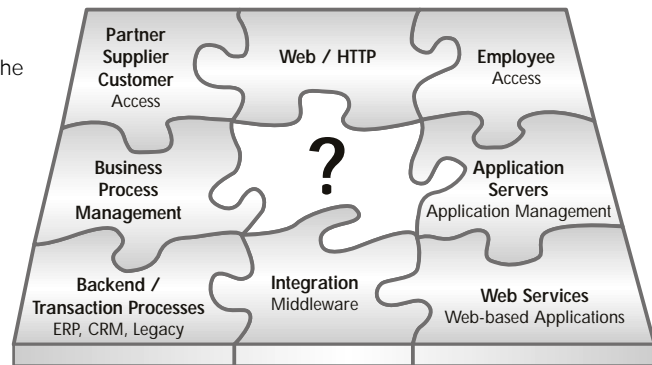
<b>Executive Summary</b>	<b>4</b>
<b>Introduction</b>	<b>5</b>
<b>XML Hype and XML Reality</b>	<b>5</b>
<b>Electronic Business and XML</b>	<b>6</b>
<b>EDI versus XML</b>	<b>6</b>
<b>XML Servers Assist Electronic Business</b>	<b>6</b>
<b>The Value of a Native XML Server</b>	<b>8</b>
Major mismatches between XML data and relational databases	9
Pure relational	9
Post-relational	9
Native XML	10
<b>Tamino XML Server</b>	<b>10</b>
<b>Core Services</b>	<b>11</b>
Storage service	11
X-Query service	11
Full-text retrieval service	12
XML Schema service	12
Extension service	12
<b>Enabling Services</b>	<b>12</b>
X-Node service	12
Integration service	12
UDDI service	12
BizTalk service	12
EJB service	12
Synchronization service	12
X-Application service	13
Tamino WebDAV Server	13
<b>Solutions</b>	<b>14</b>
<b>Tamino Developer Community</b>	<b>14</b>
<b>XML Server - Areas of Application</b>	<b>15</b>
Assembling dynamic content for Web applications	15
Content management	15
UDDI and Web services	16
Supporting wireless clients	17
<b>Conclusion: Tamino XML Server Value Proposition</b>	<b>18</b>

## Executive Summary

The IT departments of many medium and large enterprises face a significant challenge: they have invested heavily in a transaction processing and enterprise application infrastructure that does a good job of supporting their traditional business operations, but this infrastructure is not easily adaptable to support electronic business with customers, suppliers, and distributors over the Internet. Some enterprise software companies promote the strategy of ripping out the legacy infrastructure and replacing it with a modern suite of Web-enabled applications. Most sensible IT executives reject that strategy because it is expensive, risky, and - in uncertain economic times - simply impossible to fund. It makes much more sense to continue to leverage investments in the business infrastructure and invest in technologies that allow it to be Web-enabled, that is, integrated with newer and faster ways of conducting electronic business.

XML is the most important technology that has yet emerged to support Web-enabling a business infrastructure. The last few years have seen an explosion of XML technologies and specific formats that help bridge the gap between traditional ways of doing business and the newer means of electronic business. These include overall frameworks, such as ebXML and BizTalk, XML-formatted business documents from the Open Applications Group and numerous vertical industry organizations, and the Web services paradigm promoted by many large vendors and supported by the SOAP protocol, UDDI repositories, WSDL service descriptions, etc. All these are directed at the goal of allowing customers and business partners to get

Fig. 1:  
XML server - the  
missing piece of the  
IT puzzle



an up-to-date view of any company's offerings over the Web and to conduct business using simple, vendor-neutral formats and processes.

There is no shortage of visionaries and vendors offering their XML-based solutions to these problems. There is, however, a shortage of tools and technologies that address these problems in an industrial-strength manner suitable for serious businesses. Web-enablement projects are frequently challenged by problems of security, scalability, and performance in the middleware components. There is also a shortage of time, money and personnel required to implement and maintain these integrations. The Gartner Group has estimated that on average, IT organizations spend 35-40% of their budgets on the adaptation of heterogeneous system components to one another. Even more challenging, these efforts are aiming at a moving target as electronic business technology evolves. For example, as globalization, remote employees and outsourcing become more prevalent, Web-enablement will increasingly imply integration via the wireless Web to mobile devices such as mobile phones and wireless PDAs as well as to Web browsers running on wired PCs.

We at Software AG believe that an XML server provides a powerful architecture with which to provide industrial-strength solutions to these problems. XML is already regarded as the "lingua franca" for exchanging data between diverse systems. Exposing legacy information via XML protects IT infrastructure investments in a Web-enabled, wireless-enabled, and future-proof way. An XML server essentially functions as a virtual database<sup>1</sup>, providing robust DBMS services using XML data that is stored natively and/or dynamically produced from other databases and applications. Providing an XML representation of heterogeneous data allows IT shops to leverage the large and rapidly growing pool of XML expertise, tools and products to maximize time to market and minimize total cost of ownership of business integration applications.

### Tamino XML Server is

Software AG's product offering for storage, maintenance, publishing, and exchange of XML documents. In addition to highly efficient native XML storage and query capabilities, Tamino offers, in conjunction with Software AG's EntireX integration server, all functions required for total business integration. A wide range of services and solutions complement the core XML storage component.

<sup>1</sup> Term coined by IDC in 2000

## Introduction

XML and the technologies built on it are becoming pervasive in the IT world. Recent surveys indicate that more than half of all development shops will have XML projects underway by the end of the year 2001. Tamino XML Server from Software AG provides a solid technical infrastructure to exploit the power of XML to its full potential while leveraging existing IT technology investments. This white paper explores the reasons why most enterprises will soon need the services provided by an XML server and what the Tamino product line offers to meet those needs.

## XML Hype and XML Reality

XML has been hyped by so many software executives and visionaries that there is a danger that grossly inflated expectations have been raised: As CIO Magazine<sup>2</sup> jokingly put it:

"I hear it's going to cure cancer," says Tim Bray, its co-creator.

"It's going to do my dishes, I hear," says Anne-Marie Keane, Staples' Vice President of B2B e-commerce.

So what is the reality behind the hype?

In fact, XML is based on a deceptively simple idea - it is an easily understood set of conventions for defining text markup, represented by the tags within angle brackets and other XML syntax features, that allow virtually any type of data to be represented.

Let's illustrate this with a very simple XML example:

```
<?xml version="1.0"?>
<weather-report>
  <date>20010822</date>
  <time>09:00</time>
  <area>
    <city>Frankfurt</city>
    <country>Germany</country>
  </area>
  <measurements>
    <skies>rain</skies>
    <temp scale="C">28</temp>
  </measurements>
</weather-report>
```

What do we see here?

- First, it is a simple piece of text. This makes it easily exchanged across diverse hardware, software platforms, and human languages.
- The markup is self-describing. It describes the content to a human user or programmer. This clearly is a weather report for Frankfurt, Germany, describing the conditions at 9:00 am on August 22, 2001.
- It looks like the familiar HTML format. Many experienced Web users will recognize the overall format as looking much like HTML; in fact, HTML and XML are close relatives, both stemming from SGML, a generalized markup language format that has been an international standard (ISO-8879) since 1986.
- XML is easily readable by computers and by humans. It can be thought of either as a document containing human-readable information or data that would be exchanged between computers ... or both.

Some implications of having a simple, standard and powerful data format include:

- The simplicity and standardization make it easy and popular for different vendors to implement.
- This popularity promotes a network effect. As more and more vendors and consumers adopt XML, the more valuable it becomes to adopt.
- A community of experts, tools, examples, and associated specifications has grown up around XML, making it easy to build applications that exploit its power.
- Most importantly, additional industry-specific standards spring up around and on top of XML to increase its value both in horizontal applications and vertical industries. XML users can exploit the XML Schema specification as a data definition language, the XPath (and forthcoming XQuery) specifications for query languages, the XSLT, XSL-FO, SVG, CSS, and other XML technologies that specify display formats, the SOAP specification as a format for data exchanged between applications, and a rapidly-growing list of industry-specific data formats for common business documents and messages.

For these reasons, XML not only provides this simple system for defining markup languages, but has come to provide a substantial component of the infrastructure that underlies many electronic business applications and integration efforts.

<sup>2</sup> [http://www.cio.com/archive/051501/xml\\_content.html](http://www.cio.com/archive/051501/xml_content.html)

## Electronic Business and XML

Electronic business involves using Internet and Web technologies to facilitate links among internal IT applications (A2A), the IT systems of partners and suppliers (B2B), and the various devices used by consumers (B2C). XML is at the center of many, perhaps most A2A, B2B, and B2C technologies today, because it enables collaboration between businesses and business units by allowing systems to interface at a high level that can be understood by business people, not just at a low level understandable only by technologists. That is, XML lets organizations exchange information in the electronic equivalent to familiar documents that are understandable to humans, while still facilitating the underlying transactions that implement electronic business relationships.

More specifically,

- electronic versions of ordinary business documents, such as invoices, are being defined in XML format and exchanged between companies.
- XML formats based on traditional EDI messages are used to automate supply chain transactions.
- Web services.

XML is pervasive in electronic business largely because it is neutral with respect to platforms, vendors and programming languages. It leverages the existing Web infrastructure of servers, browsers and development environments that most enterprises have learned to support over the last few years. Again, these various advantages of XML for exchanging business documents and data lead to a network effect - the more popular it becomes, the more utility it provides, further increasing its popularity.

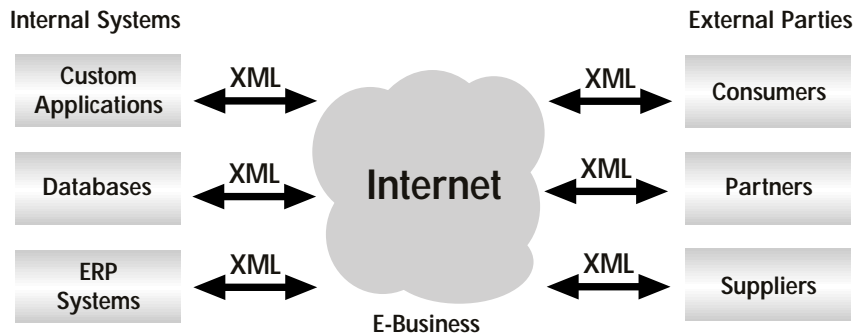


Fig. 2: Electronic business combines systems and markets

### EDI versus XML

What about Electronic Data Interchange (EDI) technologies that have been deployed for some time? XML for B2B is based on the same general idea as EDI, but fits into a different category. XML is simple, generic, pervasive, cheap, and easy to deploy. EDI solutions are much more specialized and customized, expensive to implement but optimized for performance and scalability, challenging to deploy but suitable for industrial-strength operations. Thus, XML is well suited to bring electronic business to small companies, whereas EDI is useful mainly for electronic business between large enterprises. This does not mean that XML will only be of benefit when it replaces EDI. Rather, XML can be used to complement an existing EDI infrastructure by extending its reach or putting an industrial-strength XML infrastructure in place throughout an enterprise.

### XML Servers Assist Electronic Business

XML itself provides a simple but powerful set of specifications upon which to build and interpret business documents and messages in a commonly understood electronic format. These characteristics have generated a strong network effect for XML, leading to a rapid growth in its adoption in electronic business. What should become of the XML messages and documents that are exchanged within and among businesses? Is XML just a format for transient messages?

It will become increasingly necessary to store even transitory XML data in a reliable manner so that it can be retrieved quickly, conveniently, and accurately.

There are several reasons for this:

- Enterprise applications that require XML to be stored persistently usually require more of the features supplied by a database management system. The value proposition for a DBMS is the same for XML data as for any other type of data - a DBMS provides services to an application that are too critical, too difficult or too costly to be left to application developers.

These include state-of-the-art transaction handling, efficient queries, scalability to databases larger than file systems can handle and utilities for reliably backing up/restoring the database without interrupting routine operation.

- A single Web service transaction can often generate multiple operations in different legacy business systems. For example, a Web service request may require consultation of a customer relationship management database, a credit check in an external service, an order in an ERP system, and an entry added to a workflow system. A clear "audit trail" of what happened and why could easily be maintained by logging the SOAP messages sent to the Web service in a native XML store. Auditing and analyzing the fragmented transactions in the separate back-end systems would be considerably more difficult.
- There can be a significant expense in representing XML data from an underlying system; caching it in XML format can be an effective way to maximize performance in situations where the same information is likely to be accessed repeatedly. Thus, an XML server can function as a staging server that presents a clean, easily usable view of a much more complex infrastructure.
- There are numerous tools and products that provide specialized XML processing capabilities that only work if the input data is in XML format (e.g. transformation to other formats or to publication-quality print formats, performing queries that exploit the structure of XML markup, generalized hyper-linking and mixed content capabilities).

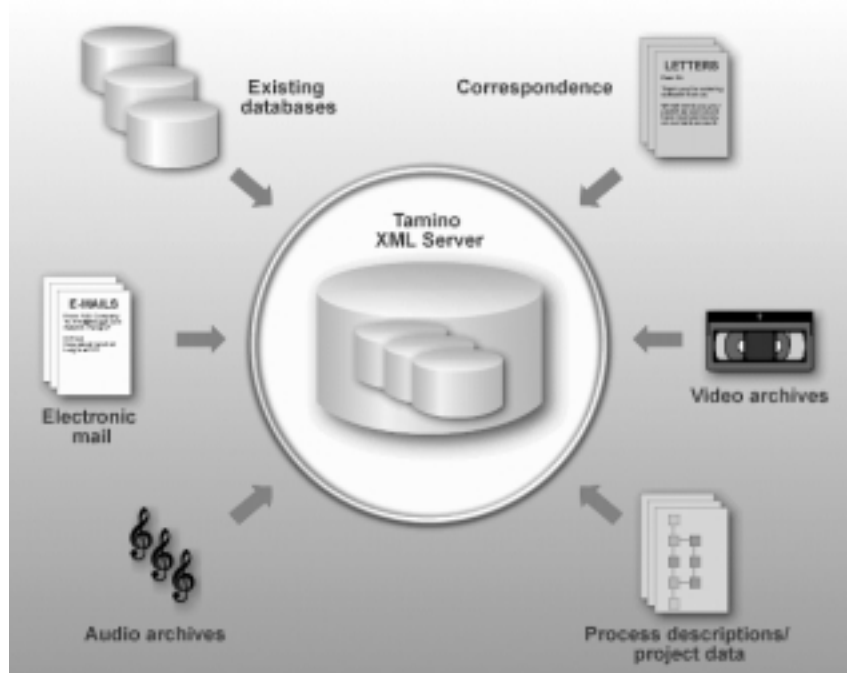


Fig. 3: Data storage in Tamino XML Server

ties, validation against the document schema / DTD defining the proper structure and content of an XML document).

- Web services that provide access to critical business processes via the XML-based SOAP protocol are sure to be a key component of most companies' Web-enablement strategy in the near future. A single SOAP request may trigger several back-office transactions, and translating back and forth from legacy format to XML could be tedious and expensive. Maintaining a single XML representation of the legacy systems in an XML server offers Web service developers more easily accessible data than the diverse legacy systems themselves.

The basic functions of an XML server are to **store** XML data from multiple sources in a reliable manner, including XML meta data about graphical, multimedia, or legacy format objects...

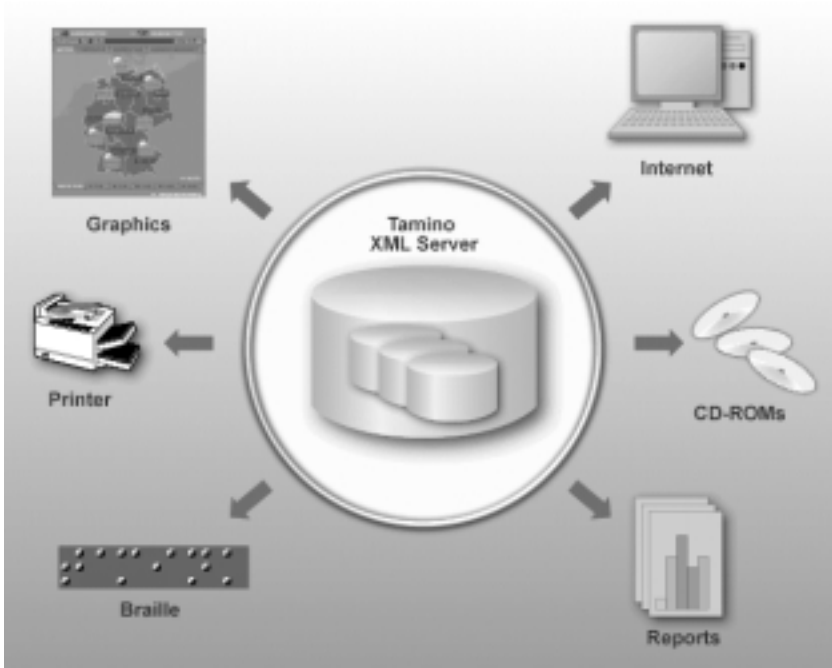


Fig. 4: Data formatting when retrieved from an XML server

... and to exploit XML's transformation capabilities to **publish** the stored data in a variety of formats, allowing the built-in XML data store to be used for the full range of devices in use today and in the future.

Thus, an XML server provides services supporting the storage, publication, and exchange of the XML documents that drive electronic business. XML servers offer an excellent business consolidation platform by providing persistence for XML data objects, a staging platform for building applications based on an XML view of underlying transactions, and an environment for developing Web services.

## The Value of a Native XML Server

At this point the reader may be wondering, "Why should I store all this information in an XML server? I have an RDBMS system in place that can store relational data, program objects and multimedia content, and the vendor promises to add XML support in the next release."

Our answer in a nutshell is: "An XML server can store XML data natively, which means without further conversion into other formats, and it can query and transform that data using the Web and XML specifications and interfaces. In addition, it integrates cleanly with Web servers and application servers in existing infrastructures. This provides clear benefits in time to market and total cost of ownership when an organization is challenged to deal with increasing volumes of XML data. Furthermore, Tamino XML Server does not try to replace existing relational databases for the tasks that RDBMS systems are well suited.

Let us analyze these assertions in more detail.

First, there are fundamental mismatches between the XML-structured data and the relational data model supported by virtually all mainstream DBMS products. Many books and articles have been written on this subject, so let us summarize the essential differences in table 1.

## MISMATCHES BETWEEN XML DATA AND RELATIONAL DATABASES

XML	RDBMS (normalized)
<ul style="list-style-type: none"> <li>• Data in single hierarchical structure</li> <li>• Nodes have element and/or attribute values</li> <li>• Elements can be nested</li> <li>• Elements are ordered</li> <li>• Elements can be recursive</li> <li>• Schema optional</li> <li>• Direct storage/retrieval of XML documents</li> <li>• Query with XML standards</li> </ul>	<ul style="list-style-type: none"> <li>• Data in multiple tables</li> <li>• Cells have a single value</li> <li>• Atomic cell values</li> <li>• Row/column order not defined</li> <li>• Schema required</li> <li>• Little support for recursive elements</li> <li>• Joins often necessary to retrieve XML documents</li> <li>• Query with SQL retrofitted for XML</li> </ul>

Table 1: Major mismatches between XML data and RDBMS

There are many ways to overcome these mismatches, and it might be useful to think of them as falling along a spectrum from the "pure relational" approach to the "post relational" approach - although most real-world applications use some combination.

### PURE RELATIONAL

The relational model of databases offers one answer: Normalize the XML data into rows and columns, with each cell containing an atomic text value. It is well established that any hierarchical or network data model can be translated into normalized relations, so in principle any XML document can be decomposed for relational storage. In practice, however, this is often far more expensive than it is worth. One analogy, however, seems to apply: Using RDBMSs for storing hierarchically structured XML documents is as efficient as disassembling your car for parking it in the garage in the evening and then re-assembling it every morning before leaving for work.

As a purely pragmatic matter for the customers of real-world RDBMS systems, normalizing XML structures into RDBMS relations can be

- fiendishly complex for designers
- time-consuming for programmers, and
- operationally inefficient for DBAs and end users.

As a general rule, the more document-like the data model - that is, when there are recursive elements, mixed content, and a generally less rigid structure - the more difficult it is to devise practical RDBMS models for XML data. Furthermore, there are well-known challenges in using SQL to effectively query normalized recursive data models such as a bill of materials.

### POST-RELATIONAL

Recognizing the practical difficulties of handling XML data in the pure relational model, RDBMS vendors have added features that can be used to simplify XML data management. Most fundamentally, RDBMS systems now support LOB (Large Object) data types that allow arbitrary types and amounts of data to be stored and retrieved in a single cell of a table. Similarly, RDBMS vendors (and the SQL standard) have added other post-relational features such as support for multidimensional data<sup>3</sup>, and the addition of full-text search capabilities, all to

make it easier for non-specialists to build effective database applications that don't fit the constraints of the pure relational model. It has been possible for the RDBMS-turned-object-relational vendors to add XML extensions to their products that employ the proprietary, post-relational, object-oriented text-retrieval features that have been added over the years. Additional utilities are provided to ease the burden of modeling XML hierarchies to work with the underlying relational and post-relational storage models. W3C recommendations for XML are primarily supported in external utilities rather than in the database itself; once records have been located by SQL or proprietary text-search extensions, the XML utilities supplied by the database vendor provide tools for representing the results as XML and manipulating the XML with DOM, XPath, etc.

XML-enabled RDBMSs require a considerable amount of programming before one can actually store, retrieve, or query XML documents. At present, XML-enabled database systems do not support a complete, seamless round-trip of arbitrary XML content into and out of a database, and none support the com-

<sup>3</sup> This post-relational feature has been supported in Software AG's pre-relational Adabas C DBMS for years!

plete XPath Specification, a language for addressing parts of an XML document stored as a file or inside of a database. Different vendors take different approaches, but all add the XML-enablement on top of existing features rather than as a fundamentally new storage model inside the database engine. One vendor, for example, added a "native XML SQL data type" to its system, which is essentially just a CLOB that supports some proprietary extensions to SQL for XML processing.

Perhaps most significantly, the XML-enabled RDBMS systems force their developers to use proprietary extensions to SQL rather than XML or Web standards to perform non-trivial operations and manipulations on the stored XML data. They are best characterized as conventional database systems that can load and save their data in XML format and not as true XML repositories.

#### **NATIVE XML**

An alternative approach, pioneered by Software AG and more recently adopted by a number of start-up companies, is to build a DBMS from the bottom up to easily store, retrieve, and query XML-structured data. This type of native XML server system exposes the data and the processing model via XML standards:

- an XML document is generally the fundamental unit of storage;
- XML DTDs or schemas rather than RDBMS schemas are used as the "data definition language" that defines the properties of document collections;

- XPath or another XML-specific query language is used to locate documents meeting some search criteria;

- and some products allow XML data to be processed with SAX, DOM, XSLT, etc., in the actual server engine as opposed to in an external utility.

As a practical matter, "XML-enabled RDBMS" and "native XML DBMS" strategies both have their place. At the risk of over-simplifying, it is generally acceptable to use XML-enabling RDBMS tools to expose existing data-centric systems as XML and to use native XML tools to capture documents that naturally occur in XML form. Today, however, these XML-enabled RDBMSs can provide an XML representation of existing relational data. These systems are not optimal if you have XML data that needs to be stored efficiently. The distinction between documents and data is a fuzzy one and is the source of considerable controversy in the XML community, but it is an important one:

Companies run on documents; the exchange of documents, such as catalogs, purchase orders, and invoices, is the foundation of all business, including electronic business; XML is well-suited to representing common business documents, and a native XML server is the easiest and most cost-effective repository for XML documents.

## **Tamino XML Server**

As described earlier in this paper, an XML server provides a common XML representation of both native XML and non-XML data. It allows application developers and systems

integrators to do their work on the clean XML representation of the data using standard XML tools.

Tamino XML Server builds on the foundation of its pioneering native XML server engine and XML services, which are covered later in this document. With its storage services (native XML data store), Tamino provides a consolidated representation of data in XML format, originating from Adabas or relational database management systems, or XML-enabled enterprise transaction systems. Tamino users can also write custom code to retrieve information from applications and embed it in any part of a returned XML document. Moreover, Tamino XML Server can handle not only XML but also non-XML data such as graphics and video files, so you have everything in one place. Since Tamino XML Server can work in conjunction with a variety of storage systems and database servers, it provides a consolidation platform that helps unite them.

Additionally, it works in conjunction with application servers - such as WebSphere, WebLogic, iPlanet, and the AppServer (formerly known as Total-e-Server) - which have become a key part of many companies' Web infrastructure to provide services such as:

- Authentication, access control, and encryption
- Transaction management
- Multitasking and load balancing
- Fail-over and crash recovery

Finally, Tamino XML Server can work in conjunction with middleware products, including integration products such as IBM MQSeries and especially Software AG's EntireX product line.

How Tamino XML Server fits into the server space opened up by application servers, integration middleware and database management systems is illustrated in figure 5.

Tamino XML Server presents a "virtual database" upon which developers can build Web and electronic business applications using standard XML development tools, schemas, and query languages. The actual reality behind the scenes consists of a combination of native XML data stores and other data sources mapped onto XML. This is significantly different from a post-relational universal database server because:

- Post-relational database products allow storage of relational, XML, and non-XML data, but present proprietary and/or SQL-based interfaces, rather than a standards-based XML representation of the underlying data.
- These products do not cleanly handle a heterogeneous mix of products in the back office, but generally make an all or nothing assumption that one product manages all the data.
- Tamino XML Server delivers services based on a standard XML representation of heterogeneous data.

Thus, no matter how the data physically exists in the background, using Tamino XML Server as a consolidation platform allows developers to build applications using XML standards such as DTDs and XML schemas to define data structures and constraints, the XPath (and soon XQuery) syntax to locate documents of interest, the DOM API to manipulate the data in application programs, and XSLT and XSL-FO to publish the data in a variety of formats.

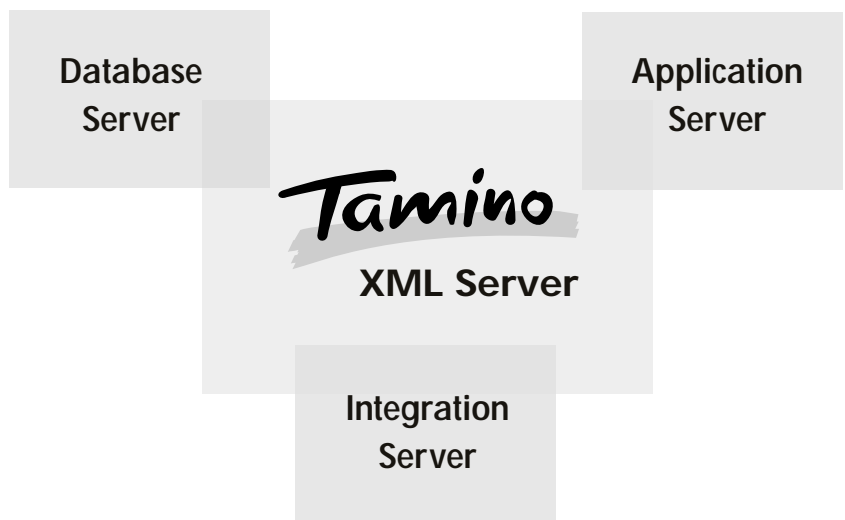


Fig. 5: The XML server space

## Core Services

The heart of Tamino XML Server is the server core. It comprises core services as key features and functionality.

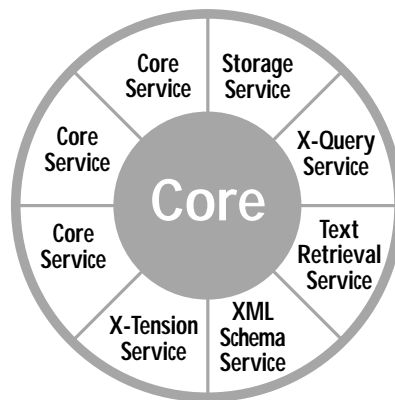


Fig. 6: Tamino XML Server's core and core services

### STORAGE SERVICE

The storage service stores XML in its native format. This direct storage of XML objects without further conversion to other data structures is one of the main explanations for Tamino's excellent performance in handling XML documents. In addition, this service can store non-XML formats, such as graphics, videos and so on, making Tamino XML Server an ideal choice for applica-

tions such as content management systems.

### X-QUERY SERVICE

The X-Query service provides a powerful mechanism for querying XML documents. The XML recommendation does not consider the handling of multiple XML documents at the same time. Currently, there is no standardized query language for XML (although the W3C XQuery specification is close to recommendation status). However, as part of the XSL recommendation, XPath was defined, which allows positioning within a single document. Tamino's X-Query implementation has extended XPath semantics to embrace the handling of sets of documents. If a query returns more than one document (or document fragment), the pure concatenation of these would not yield well-formed XML, because an XML document must have exactly one root. Therefore, Tamino wraps the result set in an artificial root element.

W3C XQuery describes a query language syntax with the power of SQL, but using concepts and operations that are appropriate to XML

structures rather than relational tables. XQuery is designed to be broadly applicable across all types of XML data sources. It is equally capable of querying data that is physically represented as XML, whether it be stored persistently in an XML data store or transiently passed between applications. Software AG is committed to supporting the up-and-coming XQuery specification in Tamino XML Server as it matures.

### FULL-TEXT RETRIEVAL SERVICE

While the navigation-based approach of XPath matches the needs of retrieval in data-centric environments quite well, the document-centric environments need a more content-based retrieval facility. Therefore, Tamino also supports full-text search over the content of attributes and elements (including their children, ignoring markup).

### XML SCHEMA SERVICE

The XML Schema service supports the W3C XML Schema. Tamino XML Server is very flexible in its handling of XML documents. It supports the storage of both well-formed XML (without an explicit schema definition) and valid XML (adhering to a schema), thus catering to varying project requirements. Documents stored in Tamino are grouped into collections (explicitly on insertion into the data store). Within a collection, several document types can be declared. For each document type, a common schema can be defined, but as an open-content model; for each document to be stored, Tamino assigns one of the document types. This assignment is based on the root element type of a document. The document must match the schema of the document type assigned, but might have additional elements/attributes

that are not modeled in the schema (open-content model). Thus, the structure of documents stored in a document type might differ considerably. If no document type with an appropriate root element type is found, the document is stored without it being matched to a particular schema.

### EXTENSION SERVICE (TAMINO X-TENSION)

The X-Tension service comprises an infrastructure for extending and customizing Tamino XML Server. Server extensions written using this infrastructure are user-defined function plug-ins of Tamino XML Server. A typical user-defined function is one that handles data in some specific way that cannot be anticipated by a standard function provided by Tamino. Once plugged in, these extensions are not distinguishable to the user from Tamino's standard functions. Extensions can be written in Java, C, C++.

### INTEGRATION SERVICE

The integration service unlocks and externalizes business assets from legacy systems via server extensions and integration technology such as Software AG's EntireX.

### UDDI SERVICE

Thanks to the native XML nature of Tamino XML Server, its design predisposes it to serve as either a public or private UDDI registry. It is also ideal as a repository for WSDL and additional Web service meta data (e.g. XML schemas, XSLT, etc.)

### BIZTALK SERVICE

If you are using Microsoft's BizTalk Server, you will probably find that tracking current and past business through BizTalk is not straightforward. This is because BizTalk Servers store XML documents in Microsoft SQL Server in hexadecimal format in a single SQL field.

Examining a document involves complex programming: You need to open every entry in the data store, convert it to XML, and search through the XML to retrieve document information. If, however, you use Tamino XML Server to capture XML documents as XML, complex information retrieval is then made easy via X-Query.

### EJB SERVICE

Tamino's EJB service provides an interface between Tamino XML Server and major application/EJB servers. Tamino transactions can be controlled by standard J2EE mechanisms.

### SYNCHRONIZATION SERVICE

Tamino's synchronization services provide data-loading mechanisms for the synchronization of content stored in XML-enabled databases on

## Enabling Services

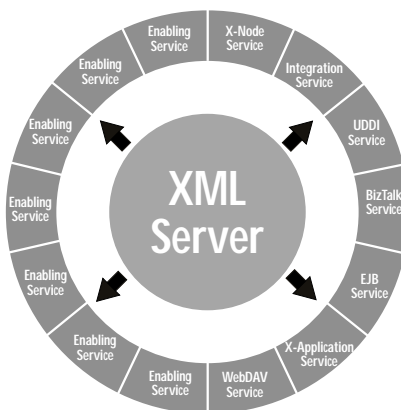


Fig. 7: Tamino XML Server's enabling services

### X-NODE SERVICE

The X-Node service provides convenient access to external data sources containing non-XML data, for example to Software AG's Adabas or an RDBMS. The X-Node service is of relevance as much of business data today is not in XML.

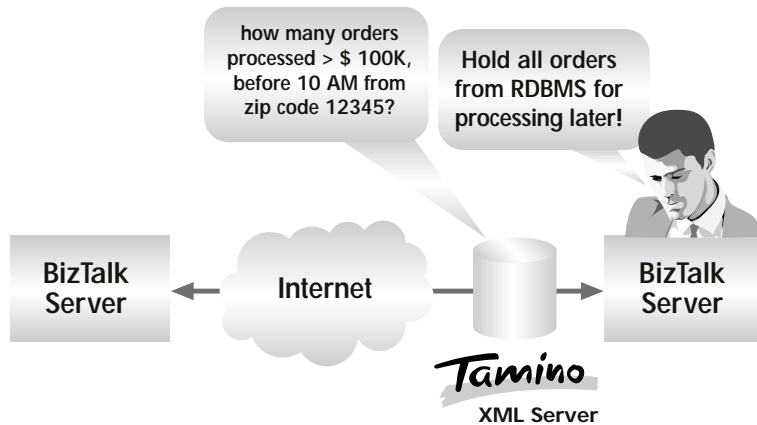


Fig. 8: Tamino XML Server - BizTalk service

remotely operated mobile devices, such as handheld PCs (PDAs), notebooks and other end-user devices. These services also allow the definition of preferences for the pre-selection of personally required data. Thus, end users always have location-independent online/near-online access to specific information they need, such as patient or customer data, hotel and restaurant directories, etc.

#### X-APPLICATION SERVICE

The X-Application service jump-starts Tamino application development, in particular the writing of Web front ends to Tamino XML Server. Tamino X-Application connects Web pages to Tamino with no programming involved. It provides JSP tags to embed access to Tamino XML Server in HTML pages. Tamino X-Application is implemented as generic, ready-to-run Java modules, and JSP tags make it possible to query, browse and maintain documents stored in Tamino. This hides the technical details associated with accessing Tamino XML Server. For more information about the purpose and functionality of this Web-application development framework, refer to Software AG's Tamino X-Application white paper.

#### TAMINO WEBDAV SERVER

Web-based Distributed Authoring and Versioning (WebDAV) is a standard for collaborative authoring of information resources over the Web. It is a set of extensions to the HTTP protocol and has been designed to make Web-based access to information resources easier than ever.

Server can be edited using any WebDAV-enabled tool, such as Word 2000 (these documents even appear in the list of recently used files), or an application such as XMLSpy. Tamino WebDAV Server can be downloaded from the Tamino Developer Community site ([www.softwareag.com/developer](http://www.softwareag.com/developer)),

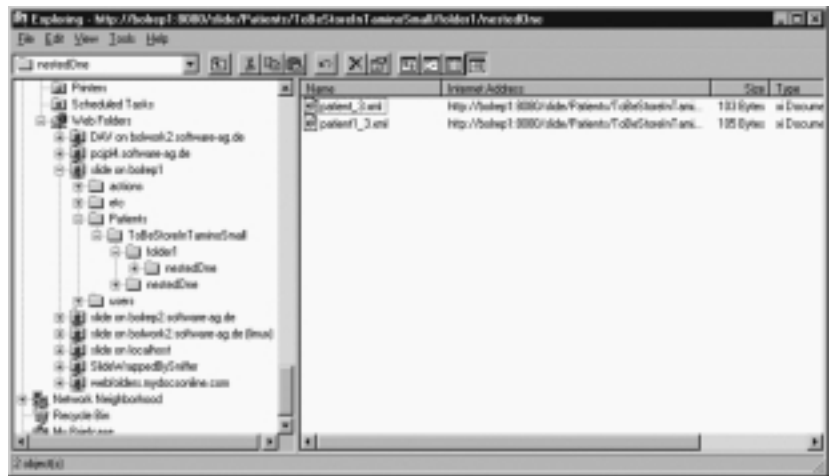


Fig. 9: Management of documents and other resources stored in Tamino XML Server is easy with WebDAV and Windows Explorer.

Tamino WebDAV Server is an implementation of the WebDAV specification that provides easy access to data in Tamino XML Server. With Tamino WebDAV Server, dragging and dropping any kind of document from/into Tamino XML Server is as easy as using Windows Explorer. Documents stored in Tamino XML

described later in this paper. Please refer to the Tamino WebDAV Server white paper for more information on the combination of Tamino WebDAV Server and Tamino XML Server or visit the Tamino Developer Community site.

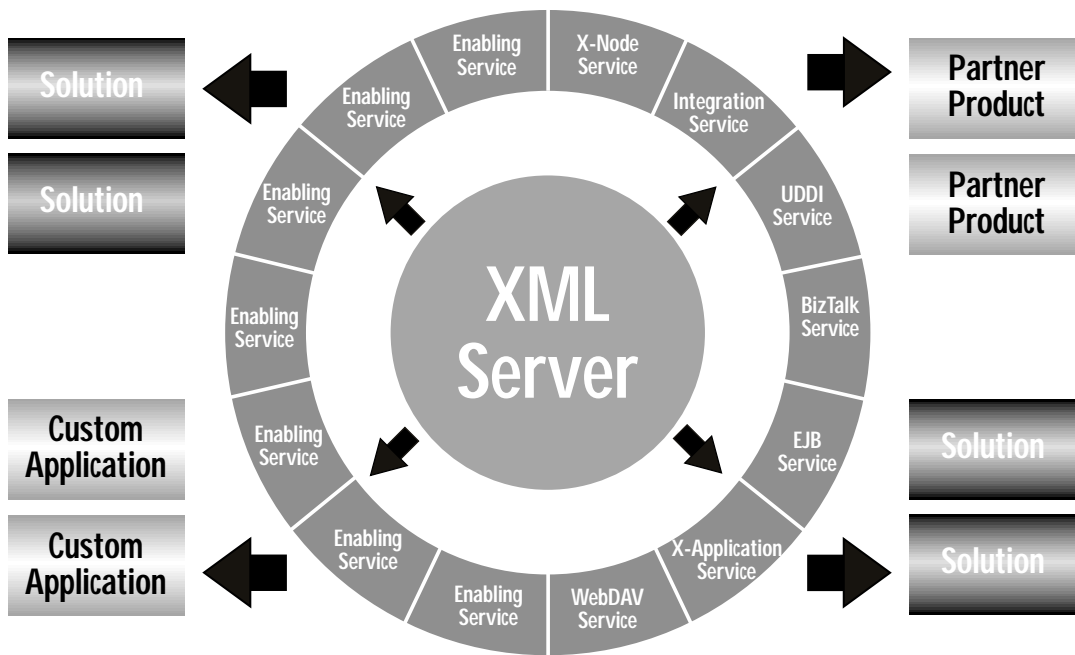


Fig. 10: Tamino XML Server's core, services, and solutions

## Solutions

The open standards-based architecture of Tamino XML Server makes it easy to add new enabling services. These services may be provided by Software AG or by partner companies, and even (when the licenses allow) through open-source development efforts. On top of the enabling services, new customer solutions and partner products form the third layer, as depicted in the figure above.

## Tamino Developer Community

It is important to note that the services offered by Tamino XML Server are not all hard-coded into the server itself. Tamino exploits Web and XML standards and common practices to provide an open architecture to which new services can be added - by Software AG, by the community of Tamino developers, or through custom coding at a particular installation - independently of the Tamino Server itself. In other words, Tamino allows services to be decoupled from the server itself. The Tamino Developer Community has

been set up online to provide a forum in which outside developers can communicate with Software AG staff, share suggestions with one another, offer additional services for download, and so forth. While Tamino XML Server itself is not an open-source product, some of its components, such as the Tamino X-Application Framework, will be offered via this shared source model. Figure 11 illustrates the relationship between the Tamino Developer Community and various other Tamino XML Server user groups, either providing or using the associated services or solutions.

Additionally, the Tamino Demo Zone offers complete sample applications for download, as well as a rich collection of links, presentations and other tutorial materials. In general, this material is freely available for research and development purposes and must be licensed only when ready to be commercially deployed.

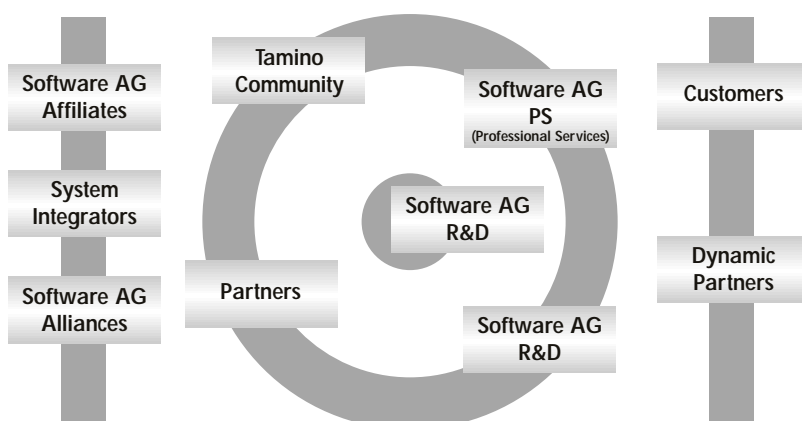


Fig. 11: Tamino Community

## XML Server - Areas of Application

Let us conclude by examining some concrete use cases that illustrate how Tamino XML Server simplifies the task of developing and deploying electronic business applications.

### ASSEMBLING DYNAMIC CONTENT FOR WEB APPLICATIONS

Web applications use HTTP as the underlying data exchange protocol. Typically, the user interface for Web applications is supplied by a Web browser, either on a PC or other Web-enabled devices, such as PDAs or pocket PCs. The following example is based upon a situation in which a company's Web-based applications must integrate data from a number of diverse back-office systems. An ordinary Web server will be the ultimate integration point in that the user will point a browser at the server and request some specific information - perhaps via an HTML form, or by requesting data via a URL. Finally, the Web server must somehow access the necessary data and return it to the browser. The conventional approach is to use a combination of HTML templates and some procedural code to access the back-end systems and fill in the specific values that were requested.

This poses a number of problems, with which Web developers have wrestled for years:

- It is difficult, even with the template languages, to maintain data independence between the desired output format, the underlying data representation, and the code that glues it together. Seemingly straightforward changes to the output format, or the requirement to support new formats, such as

XHTML or WML, can lead to considerable amounts of code being rewritten.

- The glue code is often very platform-specific. A requirement to support the increased traffic a successful site generates may mean a move to an operating system or development environment that is incompatible with the existing code.
- Systems created by cutting-edge Web developers when the project is new eventually have to be maintained by people with more conventional skills, or else the system must be rewritten. Performance often suffers in either case, because it is difficult to write server code that is secure, scalable, and maintainable.
- Interoperability with new back-end systems poses new challenges. If an additional system (within your enterprise or from a partner company) must be supported, someone has to figure out the remote system's APIs, protocols, data formats, etc., and either hack or rewrite the procedural code to handle them.

You often hear developers lament: "If we kept our back-office data in XML, we could do this much more easily with XML tools, but our existing systems work, and the executives won't take a risk on rewriting them just to make them easier to integrate and Web-enabled." Tamino XML Server can help you to have it both ways because it functions as a "virtual database" that provides an XML representation of the underlying data. Instead of writing procedural code to glue together the bits and pieces in the back office, you can use a vast array of XML tools,

techniques and products to manipulate the XML representation from any modern development environment. For example, if the back-end data can be presented as XML, one can use XSLT to specify what the mapping from source data to the output should be, rather than how to transform it. You can focus on what you do best, which for most of us is not designing optimized, scalable, multithreaded, robust server software. You then select an implementation of an XSLT engine that meets your needs ... and switch to another one as your needs change.

As output requirements change, or new output formats need to be supported in the XML server environment (e.g. for wireless devices, or for new XML data exchange standards as they emerge in specific industries), you mainly just have to write another stylesheet. XSLT editing tools are getting good enough that in the near future the job of adding and updating stylesheets will be handled by Web designers and not require the specialized skills of programmers.

Portability is also much enhanced by building on the basis of open XML standards. If a Tamino XML Server shop needs to switch to a new Web development or deployment environment, it is quite possible that some of the stylesheets will have to be tweaked to work in the new environment, but the XSLT stylesheets should work the same in both environments.

### CONTENT MANAGEMENT

Content management is one of the hottest topics in both business and IT terms. Why? To a large degree, company business depends on documents and the exchange of documents. These documents (content)

communicate product details, describe new innovations, carry bids and quotes, confirm terms, and handle payment. They teach customers how to use products, provide informative analysis, seal agreements between companies, and between companies and customers.

Actually, as the Meta group has stated, "Commerce and content are converging because the actual transaction is only 10% of the dialog one has with a customer. The other 90% ... is information."

So being successful in business, particularly on the Web, is all about managing content. And this is where XML comes in:

XML adds structure and meaning to document content, thus making it easier to find content. Separation of content from presentation facilitates content delivery in a manner suited to the user's needs, and suited to the transport medium (browser, cell phone, printer, etc.).

Tamino XML Server is thus ideally suited as an engine for content management systems. It provides for:

- Full native XML support, storing XML content as is and providing true XML retrieval capabilities based on the XPath and XQuery standards.
- Comprehensive full-text search capabilities (to obviate additional cost for 3rd-party offerings).
- Services for population and integration of external content from RDBMSs, ERP systems, file systems and others, via EntireX, Software AG's integration server.

- Transformation services to convert formats (schema and file types) from one format to another.
- Multi-channel output - one source, multiple target devices (paper, browser, PDA, etc.).
- Handling of structured and unstructured multimedia rich content (Tamino can store and retrieve multiple different file types).
- Easy integration with Microsoft Office and other WebDAV-enabled product suites (drag and drop to Tamino XML Server, edit XML documents with Word, etc.) via the WebDAV Server that comes with Tamino XML Server.

Tamino XML Server has already proven its capabilities in many content management projects. Major partners offer complete content management systems built on top of Tamino XML Server.

#### UDDI AND WEB SERVICES

Few would doubt that Web services will change the way enterprises conduct business today, and how new applications will be created. The Web services model offers an opportunity to make use of core competencies developed over years in new revenue-generating ways. Key to profitable implementation and usage of Web services are powerful integration capabilities and a strong commitment to open standards.

From a more technical perspective, Web services are a complementary model for creating dynamic, distributed applications built on an array of standard Internet technologies. Instead of relying on proprietary protocols including proprietary infrastructure, as in the past, XML and

HTTP for example are the foundation for easily connecting Web services among each other and to user front ends.

- XML is the basis for understanding. All protocols for Web services are based on the syntax of this meta language.

Web services run on all kinds of platforms and can be implemented in any programming language. The consumers of the services do not have to have any knowledge of the hardware or the operating system that the services are running on. Nor is it necessary to know anything about the object model or the programming language in which the service is implemented.

However, what will be required are more common standards so that the individual participants can communicate with one another. Examples of these standards include SOAP, WSDL, and WSFL. And there will certainly be more to follow, using the Web as an infrastructure.

- SOAP (Simple Object Access Protocol) uses XML to wrap remote method invocations and send them to the recipient. The recipient is usually an application.
- WSDL (Web Services Description Language) is used to describe services that are being offered. The service providers and the service requestors use WSDL as a contract language. It gives the service requestor specific information on the methods that are being offered, and therefore makes links to the providers possible.

- WSFL (Web Services Flow Language) is an XML language used to describe how Web services may be aggregated into new Web services. Business processes are supported by two existing types of Web service compositions that allow you to:

- specify the logic of a business process for achieving a particular business goal (usage pattern). The result is a description of the business process.

- define how participants in a business process can connect to each other's Web services offering (interactivity pattern). The result is a description of the overall partner interactions.

UDDI adds value to more general Web service scenarios by providing a registry with publishing, discovery and integration information about existing Web services.

- UDDI (Universal Discovery, Description and Integration) provides access to businesses worldwide and the descriptions of their products and services, which you may be looking for. Your own products and services can be registered for others to search and analyze as well.

Many organizations are creating internal (private) UDDI repositories as a way to acquaint themselves with and test the concept and technology of Web services and the subsequent integration with business enterprise applications.

Tamino as a native XML server provides the high speed search mechanisms that make it ideally suited to act as a UDDI registry and repository. EntireX provides "three-click"

transformation of business services into Web services using wrapper technology. Furthermore, EntireX - with its track record of eight-plus successful years in the integration field - supports publish and discovery for automated communication with UDDI registries.

#### **SUPPORTING WIRELESS CLIENTS**

Tamino is also ideally suited as a portal/staging server for wireless devices that must provide access to information maintained in an IT infrastructure. Consider an application running on a wireless PDA that allows a sales or service organization to schedule customer appointments and access customer information remotely. Basing the architecture on XML allows this to be a special case of an application integration problem - the PDA application and the enterprise infrastructure applications exchange data in a neutral XML format that conveniently wraps up the necessary information for exchange between the systems. Tamino XML Server supports a number of alternative architectures for such an application since it offers different XML services that a designer could exploit.

In one scenario, the PDA application could be written as a straightforward HTML or WML application that lets the PDA user fill out forms requesting specific information from the XML server. Tamino facilitates this architecture by its ability to provide a simple XML representation of the underlying infrastructure to the client, using its internal data map, custom extension code, and native XML repository. Similarly, Tamino's transformation services could be invoked to translate the XML schema that it presents to clients

into specific HTML or WML markup that can be directly displayed on the device.

In a variation of this scenario, the PDA application may not be built on XML technology, but employs the SOAP protocol in its back end to access Web services via Tamino.

Finally, the PDA application itself may not speak XML, but uses a lightweight database management system installed on the PDA to manage its data. For example, an XML-enabled mobile DBMS could then be frequently used in remote locations by employing Tamino XML Server's synchronization services to keep the schedule and customer data cached on the PDA in sync.

## Conclusion: Tamino XML Server Value Proposition

How can Tamino XML Server provide tangible benefits to those implementing electronic business applications on top of heterogeneous systems?

**Firstly**, Tamino XML Server enables customers to easily implement a flexible, low-cost data integration strategy based on XML.

- XML is the key to exchanging data across applications and enterprises.
- Tamino X-Node lets you access data from legacy databases as if it were in XML format.
- Tamino X-Tension provides an XML-enabled interface to legacy processes.
- Tamino XML Server can function as a convenient staging server in which to cache an XML representation of a back office; XML applications can be built on top of this clean XML representation of the underlying complexity.

This ease of use implies a shorter time to market for applications built on top of Tamino XML Server.

**Secondly**, Tamino XML Server can protect investments in core IT systems that need to be exposed to the outside world as XML. Taking advantage of the Internet and electronic business means opening up (externalizing) core IT processes, the so-called enterprise transaction systems, to the outside world. Writing whole new applications is simply not an alternative due to huge investments in existing technologies

and systems, which are generally doing the job they are supposed to do quite satisfactorily.

Tamino XML Server can access existing systems both at the data level and at the application level, and is particularly strong in conjunction with EntireX with its proven track record of integration success. The EntireX Orchestrator module complements Tamino because it provides a highly configurable, adaptable, and scalable XML routing hub for messages and documents generated by other XML-enabled applications.

**Thirdly**, Tamino XML Server helps minimize the total cost of ownership of integrated electronic business applications. For information that has already been put in XML format, Tamino XML Server stores XML documents, as is, in their native format, eliminating the need to map XML structures to relational or other structures. This is more efficient than the approach taken by post-relational "universal database servers" which must map XML onto SQL and full-text storage subsystems and back to XML upon retrieval. More importantly, maintaining these mappings involves a lot of work for system administrators as well as the programmers developing the initial system.

For information that does not arrive in native XML format, Tamino XML Server offers a flexible set of tools for mapping data from underlying databases and applications onto an easy-to-use XML representation. Whether XML data is stored natively in Tamino XML Server or generated dynamically from other applications, you can use easily obtained, widely understood tools to exploit the XML representation that Tamino provides.

Tamino XML Server is based throughout on industry standards. All interfaces between the components of the Server and external systems are thus easier to handle and more open to adapting to future needs.

This leads to savings in personnel costs. A major headache for IT managers is the difficulty in finding and retaining the right staff to develop and maintain systems. Since Tamino XML Server is built on standards, in particular those relating to XML and the Internet, the IT staff required to manage Tamino needs little knowledge other than that related to these standards. This makes personnel recruitment easier, and people working with XML standards know that their future prospects are rosy.

**Fourthly**, using Tamino XML Server as a staging platform can offer significant performance and scalability advantages over other integration strategies. The staging server concept means that access to core business processes, such as those in ERP systems, is not direct but indirect through the staging server. The advantage is that the operational back-end systems are not overloaded, so their response times are generally very good. In addition, the staging server can compile information from many sources, such as multiple online stores, etc., and present the information in a single view to the customer.

Tamino XML Server is an ideal engine for these implementations because of

- its native XML storage,
- its ability to communicate with back-end systems and
- its open, Web-oriented architecture.

For more information on Tamino XML Server:  
<http://www.softwareag.com/tamino>

Tamino Community:  
<http://www.softwareag.com/developer>

Download XML Starter Kit:  
<http://www.xmlstarterkit.com>

**Software AG**  
**Corporate Headquarters**  
Uhlandstraße 12  
64297 Darmstadt/Germany  
Tel: +49-61 51-92-0  
Fax: +49-61 51-92-11 91  
[www.softwareag.com](http://www.softwareag.com)

 **SOFTWARE AG**  
THE XML COMPANY