software AG

Tamino

Concepts

Version 9.5 SP1

November 2013

tamino XML Server

This document applies to Tamino Version 9.5 SP1.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: INS-CONCEPTS-95SP1-20131030

Table of Contents

1 Introducing Tamino 1
2 Background 3
    What is Tamino? 4
    Why is Tamino Based on XML? 4
    Why Use Native XML Storage? 5
3 Concepts 7
4 General Architecture 9
    The Tamino XML Server 11
5 Tamino Product Components 19
    Tamino Schema Editor 20
    Tamino Interactive Interface 20
    Tamino X-Plorer 20
    Application Programming Interfaces 21
6 Working with Tamino 23
7 Conclusion: Advantages of Tamino XML Server in a Nutshell 25
Index 27

1 Introducing Tamino

Tamino XML Server is a high performance data management platform based on XML standards. Tamino:

- stores XML documents natively, that is in their original format;
- stores non-XML documents such as Microsoft Office documents or PDF files;
- exposes information residing in various external XML or non-XML sources (legacy data) or applications to the outside world in XML format; and
- enables you to search effectively on the information to which it has access.

This paper introduces you to Tamino and provides a technical overview under the following headings:

Background
Concepts
General Architecture
Tamino Product Components
Working with Tamino
Conclusion: Advantages in a Nutshell

2 Background

- What is Tamino? 4
- Why is Tamino Based on XML? 4
- Why Use Native XML Storage? 5

This chapter gives you brief answers to the following questions about Tamino:

What is Tamino?

Tamino XML Server is a high performance data management platform. Tamino is based upon XML and other open standard internet technologies and helps with finding and managing any type of content across the enterprise. Using Tamino XML Server is of importance for companies implementing platform-independent business-to-business collaboration strategies in mission-critical environments. Its performance, its query capabilities and its flexibility save a considerable amount of time as well as development and operational cost that would otherwise be spent for adapting traditional, yet XML-enabled solutions (RDBMS) to work effectively in an XML environment.

Why is Tamino Based on XML?

XML is the most important technology for Web-enabled infrastructures. XML's world wide acceptance comes from various important roles it plays for the Internet:

- It is a universal standard for structured data.
- XML is the underlying technology for all web service implementations and SOA (service oriented architecture) systems.
- It supports “MOM” capabilities (messaging oriented middleware) based on XML's inherent extensibility and robustness. XML's MOM features allow for easy adaptation to changing business needs and flexible exchange of information between diverse IT systems.
- Through XML's “POP” capabilities (presentation oriented publishing) content is separated from presentation. Thus, XML is ideally suited for creating and maintaining information only once, yet presenting it dynamically in a multiplicity of output formats and on a variety of display devices.

As electronic business evolves, XML is playing a further important role for businesses that have to cope with increasing volumes of exchanged data and business documents.

What is required is obviously an efficient storage format (native XML) for all information coded in XML. Tamino XML Server is the first commercial server providing these highly efficient native XML storage capabilities.

Why Use Native XML Storage?

Storing XML data natively has an enormous advantage over relational database management systems (RDBMSs), because no extra data conversion layer is required as, for example, needed for XML-enabled RDBMSs and the document structure is kept intact. Relational database management systems (RDBMS) may appear to be a possible choice to facilitate the exchange of XML objects. But the table-based data model of the RDBMS does not suit the hierarchical and interconnected nature of XML objects. An RDBMS would need to break an XML document down into a multitude of interrelated tables. A query against this database would result in many relational retrieval and join operations, requiring high processing power to overcome a considerable degradation of performance.

In addition, RDBMSs and more advanced DBMSs, such as multi-dimensional relational databases or object-oriented databases, cannot handle data with dynamic structure, which is the key to XML's extensibility. A native XML data store must be able to store and retrieve any well-formed XML document, even if schema information (DTD or XML Schema) of the document is not available. An RDBMS, however, needs schema definitions for each table, so a document with an unknown tag would require a change request for a new schema definition, to be built and approved before it can be put into production. Likewise, in an object-oriented database a new class definition would be necessary, which is too time-consuming for Internet applications.

Native XML storage is the essential method to avoid these performance limitations that are a crucial factor in the evolving mission-critical high-speed world of e-business. Where data is subject to change or exhibits a complex structure, storage in a native XML Server will give better performance and greater flexibility. Some of the applications that demonstrate these characteristics are:

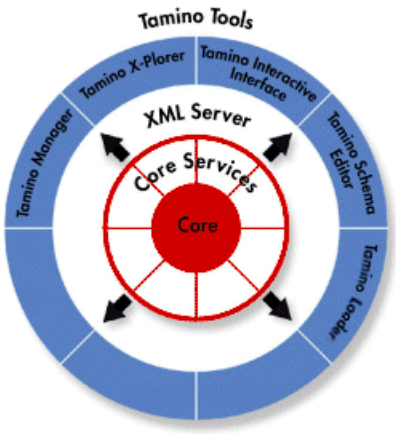
- *Content Management*: Tamino has been embedded into content management solutions to offer native XML storage. This makes it easy to publish content in different formats to a wide variety of devices.
- *E-Commerce Audit*: As more and more transactions become wholly electronic, there is a need to store an accurate representation of these transactions, the vast majority of which will be XML documents.
- *Staging Server*: A staging server provides an XML representation of information held in a back-office system. It prevents the back-office system from potential overload and from untrusted access, while making the data accessible via the Web to customers, suppliers, and trading partners.
- *Enterprise Portals*: Easy access to content is a vital part of a portal solution, and Tamino makes it easier to deliver content in a form that is best suited to the user's requirements.

For further information on using Tamino XML Server, see the "XML Data Management" pages at **http://www.softwareag.com/Corporate/products/wm/tamino/default.asp** as part of the webMethods product suite on Software AG's corporate Internet site.

3 Concepts

The Tamino XML Server concept is based on a simple equation:

```
Tamino XML Server = Core Services + Enabling Services (Tamino Tools) + Solutions
```



Core Services

The heart of Tamino XML Server is a compact, high-performance server core providing mechanisms and functionality for:

- managing and maintaining business documents (data)
- storing these documents in native XML and non-XML formats
- configuring one or more Tamino systems
- browsing
- fast querying on internally or externally stored data with minimal effort
- customizing the server functionality.

Enabling Services (Tamino Tools)

Enabling services provide a broad range of tools and components necessary for productive development of XML-based solutions focusing on Web-enablement of existing disparate corporate IT infrastructure, as well as effectively publishing and exchanging electronic documents over the Internet.

Based on enabling services, Tamino XML Server supports and easily integrates with application servers and external data sources.

Enabling services also allow for

- describing and accessing information from within and outside the enterprise without compromising security, availability or performance and include Tools for
- jumpstarting development of XML-based applications, even with no programming involved and tools that allow for using Tamino XML Server as if it were a writable web file system.

Solutions

New custom-built solutions and products from independent software vendors form the versatile and fast growing outer application-specific layer of Tamino XML Server. Since Tamino XML Server is open standards based and providing an open architecture, it is easy to add new enabling services and to build powerful solutions quickly, leveraging all the services mentioned.

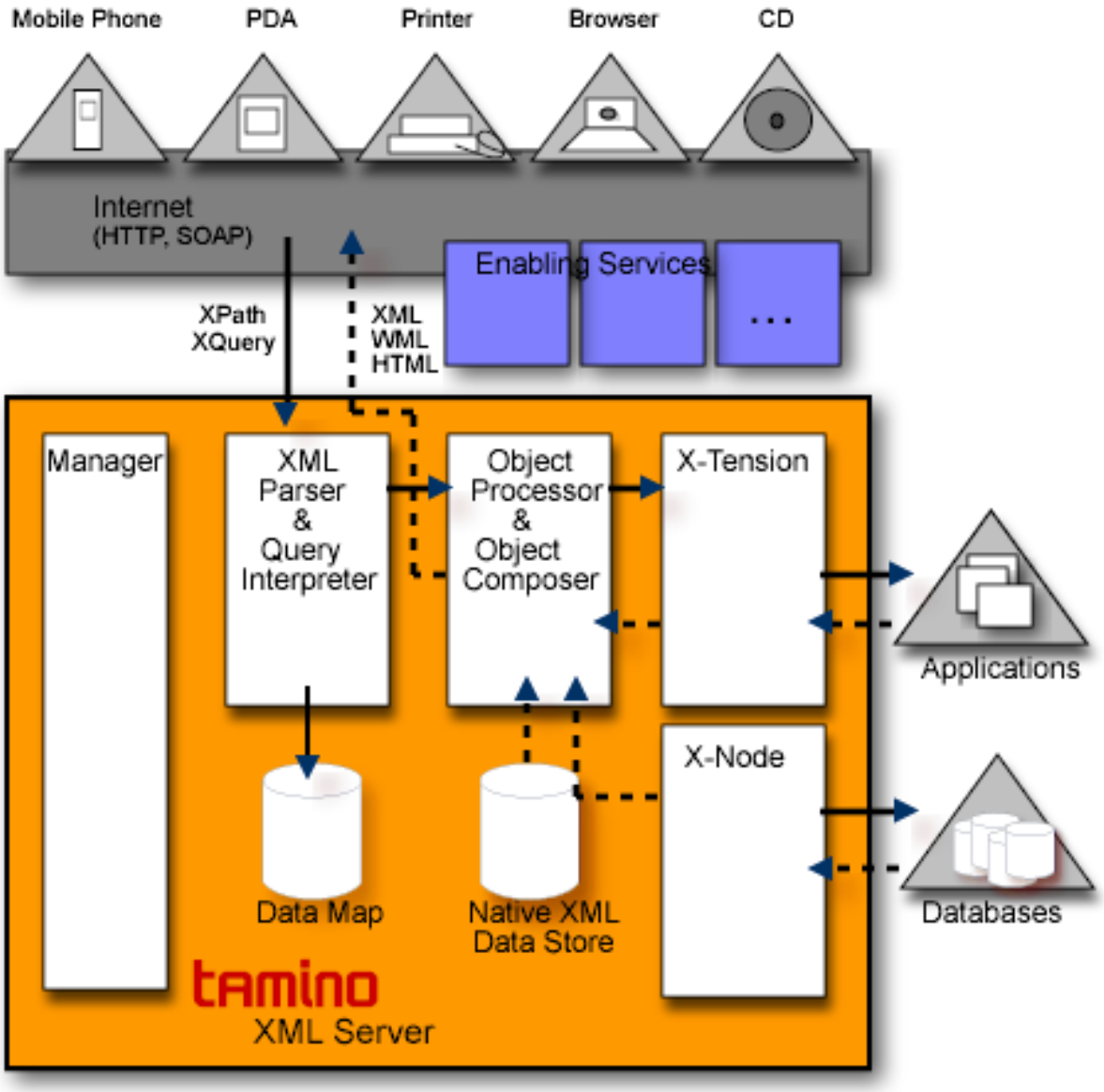
Tamino Services are not all hard-coded into the server itself. Tamino allows services to be decoupled from the server, thus exploiting Web- and XML-standards and common practices to provide an open architecture in which new services can be added – by Software AG, or by custom code at a particular installation – independently of the Tamino Server itself.

4 General Architecture

- The Tamino XML Server 11

This topic illustrates how Tamino XML Server's technology is implemented. It describes Tamino's general architecture by giving an overview of the salient components and a description of how they interact. In essence, Tamino consists of two main parts, the Tamino XML Server and the product components (enabling services), which are single units being able to work as standalone components.

The following graphic illustrates the complete Tamino installation:



Architecture of the Tamino XML Server

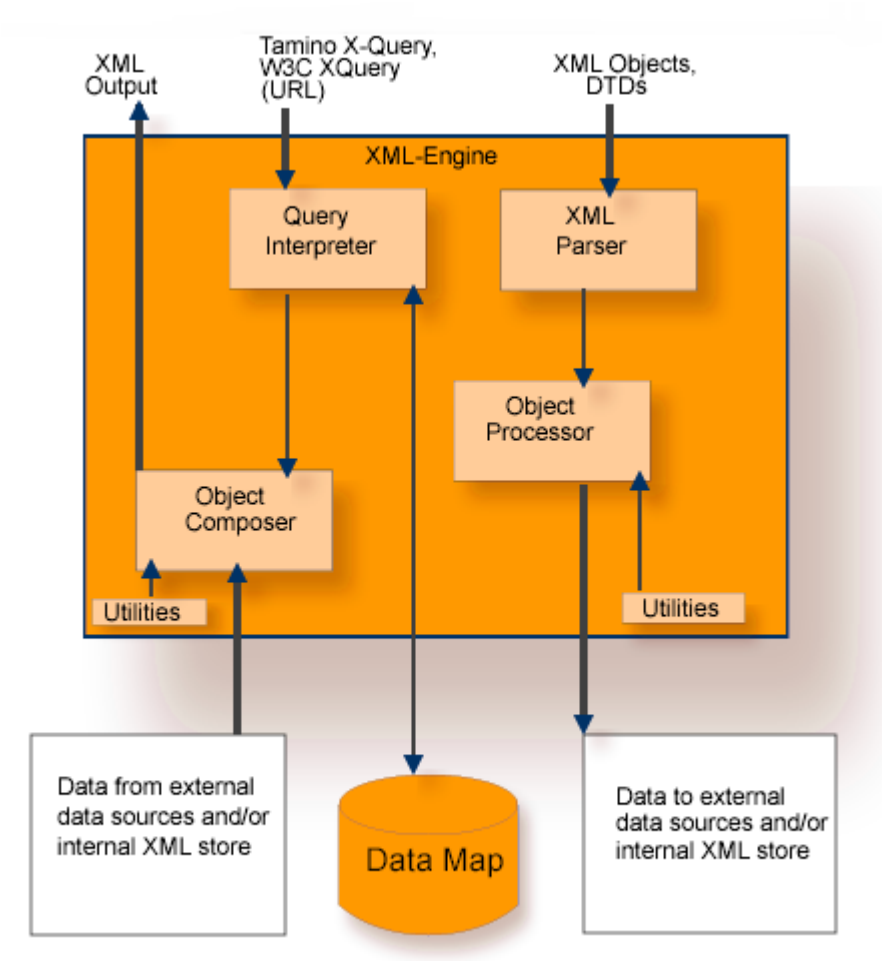
The Tamino XML Server

The Tamino XML Server is not just a data store. It is comprised of the following major parts that make up a system for XML storage and retrieval. The five major parts are:

- **The Native XML Data Store, including the XML-Engine**
- **The Data Map**
- **The X-Node**
- **The X-Tension**
- **The Tamino Manager**

The Native XML Data Store plus XML-Engine

The native XML Data Store plus the XML-Engine (also referred to as X-Machine) are the central and most powerful components in the Tamino XML Server architecture. Their high performance and robustness are the basis for many Tamino core services such as highly efficient storage, querying and retrieval of XML documents. These core services include Tamino X-Query as well as W3C-conform XQuery, and full-text retrieval functionality. They are based on major building blocks such as an integrated XML parser, a query interpreter, and an integrated native XML data store. It is the direct storage of XML objects without further conversion to other data structures that is a main reason for Tamino's excellent performance. It is also capable of storing arbitrary non-XML objects.



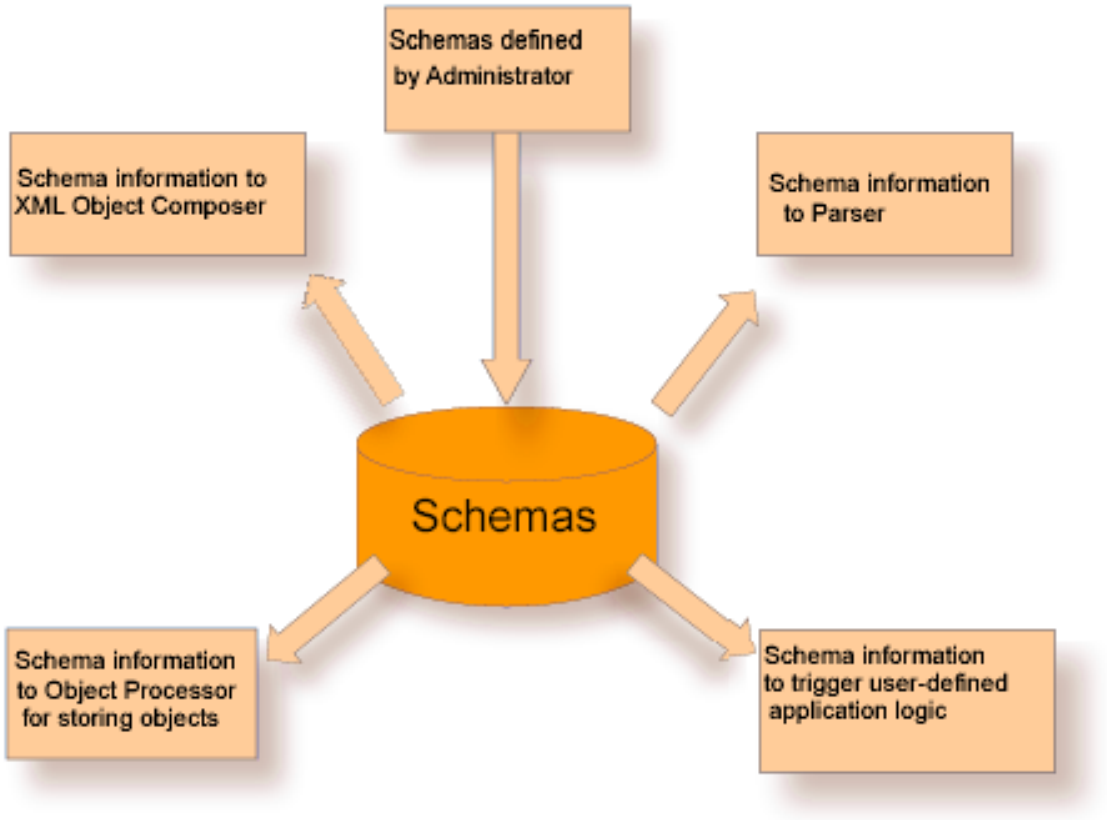
XML Engine and Native XML Data Store

- **XML Parser:**
  XML objects to be stored by the X-Machine are described by their schema stored in Tamino's Data Map. The X-Machine's internal XML parser checks syntactical correctness of the schemas and ensures that incoming XML objects are well-formed. It also validates data, if there is a schema.
- **Object Processor:**
  The Object Processor is used when storing objects in the native XML store. Support of external data sources is provided by the Tamino X-Node and X-Tension.
- **Query Interpreter:**
  Tamino supports two query languages: Tamino X-Query, based on the XPath standard, and the standard query language XQuery as recommended by the W3C. The Query Interpreter consists of the Query Compiler and the Query Executor. It optimizes the query along the given schema for resolving requests and checks whether indexes are available to accelerate query execution. It interacts with the Object Composer to retrieve XML objects according to the schemas stored in the Data Map.

- **Object Composer:**
  The Object Composer is used when the XML information sets have to be composed. Using the storage and retrieval rules defined in the Data Map, the Object Composer constructs the information objects and returns them as XML documents. The simplest case will be retrieving an object stored natively as XML. In more complex cases, communication with X-Node and X-Tension is required to compose an XML object from non-XML data sources.

The Data Map

The Data Map is the knowledge base of Tamino's server core. It contains XML metadata: the Tamino schemas, defining the rules according to which XML objects are stored and composed. The Tamino schema determines how XML objects, embedded in XML documents, will be mapped to physical database structures, whether they reside natively or externally (for example, legacy databases), and whether indexes are to be built for faster retrieval. This way, the Data Map allows existing databases to be enabled for XML technology and the Web:



The Data Map contains the information required for the following functions:

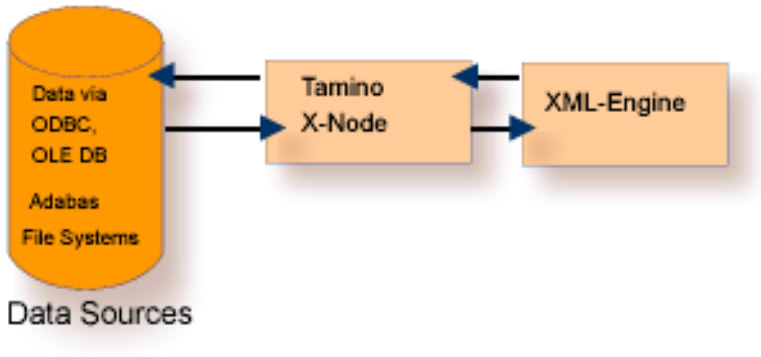
- validation against logical schema
- storage and indexing of XML objects within Tamino
- mapping of data to different data structures to enable the integration of existing data
- mapping of data to existing databases
- executing user-defined application logic using a **Server Extension Function** associated with an object

The definition of the schemas in the Data Map is supported by a graphical tool (the **Tamino Schema Editor**) that guarantees the creation of error-free XML syntax and provides some default specifications.

Tamino XML Server supports W3C's XML Schema based on the XML-Engine's and the Data Map's capabilities. Therefore, Tamino is very flexible in its handling of XML documents and supports the storage of both well-formed XML (without an explicit schema definition) and valid XML (adhering to a schema).

The X-Node

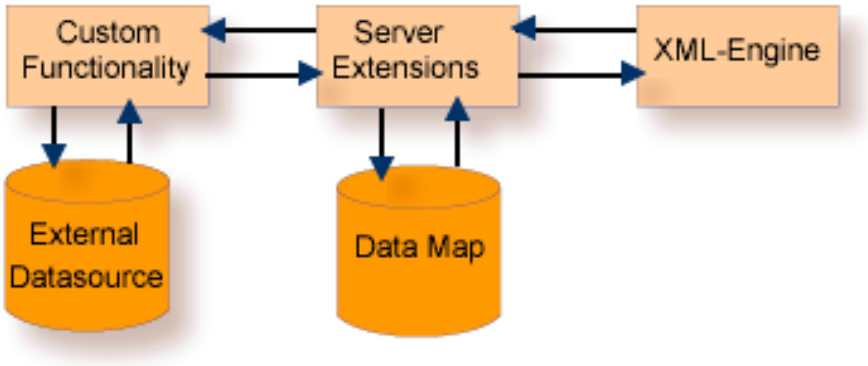
The X-Node is Tamino's integration component with external data storage systems:



Tamino X-Node provides access to existing Adabas databases with traditional data structures. Tamino X-Node maps this data to XML structures, providing continued usability of existing database infrastructures and thus protecting legacy IT investments. With the help of Tamino XML Server's mapping mechanism, Tamino X-Node allows the presentation of disparate corporate data to the client application as if it were obtained from a single database (single server view). This gives Tamino XML Server the power to act as virtual DBMS, meaning a central server for existing databases over the Web and for Web-oriented applications.

X-Tension

Tamino's X-Tension component allows calls to user-defined functions, so-called Server Extensions:

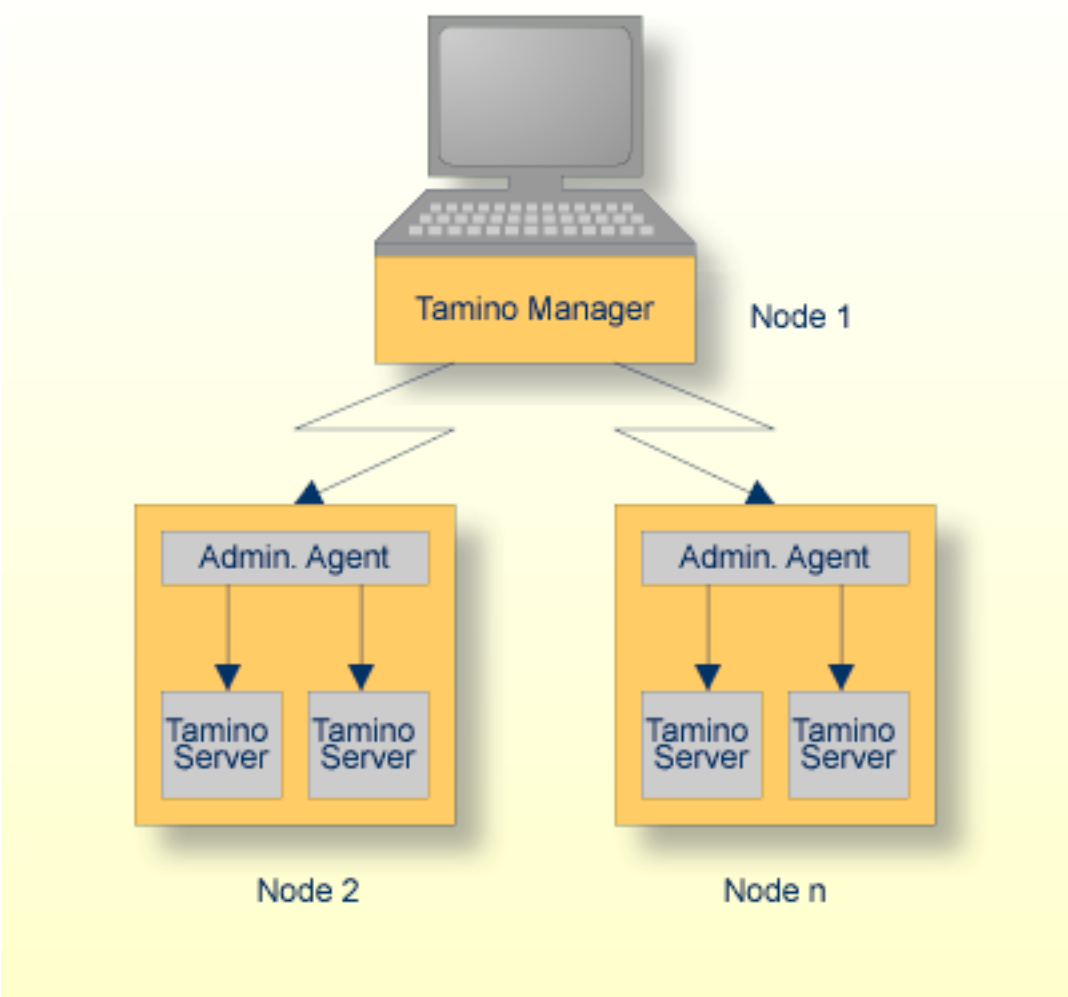


Server Extensions allow for access to various external applications and for writing custom functionality enabling Tamino XML Server to meet application specific needs. These user-defined function plug-ins of Tamino XML Server can be written in Java, C, C++ and any COM-enabled language. Tools (for Java, C, C++ and Natural) and an add-in for Microsoft Visual Studio (for C++) support the implementation of extension functions.

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 for X-Query/XQuery or mapping. Based on the schema definitions, an incoming XML object can be mapped to a user-defined function, which is then executed. Thus, Tamino X-Tension provides a built-in XML-enabled interface for access to legacy processes and data sources, using Software AG's EntireX package for the integration layer. Also, action triggers can be associated to nodes of the schema.

The Tamino Manager

The Tamino Manager is Tamino's administration tool.



It is implemented as a client-server application and is integrated into the System Management Hub, Software AG's multi-platform environment for the unified management of Software AG products. As such, it is Tamino XML Server's point of central administration. It provides a graphical user interface running on standard Web browsers, and a command line interface. Tamino Manager allows the Tamino administrator to manage the entire system over the Web (that is, create database, start/stop server, back up, restore, load, etc.). Tamino Manager allows for the installation of Tamino X-Tension server extensions for greater flexibility.

5 Tamino Product Components

- Tamino Schema Editor 20
- Tamino Interactive Interface 20
- Tamino X-Plorer 20
- Application Programming Interfaces 21

Besides the Tamino Server, there are several runtime and development components (enabling services) for ease of use and for application development with Tamino. The single Tamino XML Server product contains all you need to set up a running Tamino system on a server, including the necessary product components. They are:

- **Tamino Schema Editor**
- **Tamino Interactive Interface**
- **Tamino X-Plorer**
- **Application Programming Interfaces (APIs)**

Tamino Schema Editor

The Tamino Schema Editor supports you in creating Tamino schemas. It shields you from having to type in schema language syntax, thus making schema creation much faster and less error-prone. The Tamino Schema Editor also shields you from the complexity of the XML Schema standard by offering dialogs specifically for the creation of Tamino schemas. Schema constructs required by the XML Schema standard are added automatically to ensure that valid schemas are generated.

Tamino Interactive Interface

The Tamino Interactive Interface is a simple browser-based interface to Tamino XML Server. It allows you to define collections within a schema, load XML instances of a schema into a database, and delete XML instances, schemas and collections from a database. You can also submit query language requests to a database, using the W3C standard query language XQuery *or* the Tamino query language X-Query.

Tamino X-Plorer

From its graphical user interface, the Tamino X-Plorer conveniently displays the contents of Tamino XML Server databases in a navigation tree, thus allowing you to intuitively explore and manipulate its contents.

Summarizing the Tamino X-Plorer's features you can:

- Explore a Tamino XML Server's database
- Query a Tamino XML Server's database
- Maintain the structure and contents of a Tamino XML Server
- Display and edit objects in a Tamino XML Server

- Invoke tools which support you in developing Tamino applications
- View externally maintained data such as security data, or server extensions

Application Programming Interfaces

There are several application programming interfaces (APIs) that are delivered along with the Tamino XML Server. These are:

- **Tamino API for Java**
  This is a very flexible object-oriented API. It offers the Java programmer a comfortable access to data stored in Tamino by using different object models (DOM, SAX, JDOM) or stream-based access.
- **Tamino API for .NET**
  The Tamino API for .NET provides an object-oriented programming interface to the Tamino XML Server for .NET applications. The API, which is completely written in C#, supports the .NET XML classes for processing XML documents in a Tamino database.
- **Tamino API for C**
  This API offers a variety of methods and properties that enable any application that can call C functions to access and manipulate documents in a Tamino database. The Tamino API for C allows client applications to access a Tamino XML Server without going through a web server. This distinguishes it from most other Tamino APIs.
- **HTTP Client API for ActiveX (Windows only)**
  An API for Windows platforms that uses the DOM object model to access and manipulate data stored in Tamino XML Server. It can be used for applications written for example in C++ and VisualBasic.
- **HTTP Client API for JScript (Windows only)**
  An API that uses the DOM object model to access and manipulate data stored in Tamino XML Server. It can be used for applications within the Microsoft Internet Explorer, Version 5.0 and above.

6 Working with Tamino

Tamino XML Server and its components offer a wide range of solutions for developing and deploying Electronic Business applications. Here is a typical scenario: A company's Web-based applications (that is, those in which the user interface is supplied by a Web browser, either on a PC or some other Web-enabled device such as a PDA) must integrate data from a number of diverse back office systems. An ordinary Web server will be the ultimate integration point - the user will point a browser at the server, request (perhaps via an HTML form, or by requesting data via a URL) some specific information, and the Web server must somehow access the necessary data and prepare it for display in the browser. The Tamino XML Server functions as a "virtual database" that presents an XML view of the underlying data. You can use a vast array of XML tools, techniques, and products to manipulate the XML view from any modern development environment. For example, if the back-end data can be presented as XML, you can use XSLT (eXtensible Stylesheet Language Transformations) to specify the output format.

Another scenario involves the use of web services in an SOA (service oriented architecture) environment. Suppose that you wish to expose parts of your business operations to partners or customers as a service offered in the Web, that is, you wish to allow access to your services to those who use the SOAP protocol, i.e., an XML-formatted message requesting some operations to be performed and the result returned in XML format. Many of the advantages of using the Tamino XML Server to build Web applications that humans can read apply when a computer ultimately "reads" the data. Building Web services on an XML virtual database view is easier, more portable, and cheaper than doing this with ordinary programming.

But in essence, whether you wish to store XML data in Tamino, to use Tamino to update external databases, to integrate existing data sources for retrieval, to build Web applications or to produce different output formats, the basic procedure for working with Tamino is the same in each case. Steps one and two (see below) are optional, since a native XML store such as Tamino can store and retrieve *any* well-formed XML document, even if the DTD of the document is not available. In addition, it is possible to store non-XML documents, e.g. graphic files, not needing a DTD or schema.

1. Describe the structure of your XML data using a schema. Tamino accepts XML Document Type Definitions (DTDs) via its Schema Editor, or XML Schema and converts them into Tamino schemas (in a separate step).
2. Define a Tamino schema. Tamino is based on XML Schema, thus providing a schema language recommended by the W3C that describes the rules according to which data are stored and can be retrieved. You can either write a Tamino schema using a text editor, or use Tamino's graphical Schema Editor.
3. Once a Tamino schema is defined, you can store instances of the schema. Tamino stores and retrieves information according to the rules specified in the schema.
4. Use the standard query language *XQuery* (based on the W3C recommendation) to retrieve your data. You can query your data via your favorite browser, with the Tamino Interactive Interface, or with the Tamino X-Plorer.
5. Build an application with the data returned, using Tamino's vast array of tools and services.

Tamino's possibilities are manifold. For an introductory description of the single steps including examples, refer to the Tamino document Getting Started.

Detailed and advanced information about working with Tamino can be found in the documentation. Also, take a look at the **Tamino Developer Community** for questions and answers and important news.

7 Conclusion: Advantages of Tamino XML Server in a Nutshell

Let's briefly summarize how the Tamino XML Server can provide tangible benefits to those implementing Electronic Business applications on heterogeneous systems.

First, *the Tamino XML Server enables you 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 stores XML data *natively*.
- Tamino X-Node lets you view legacy databases in XML format.
- Tamino X-Tension provide an XML-enabled interface to legacy processes.
- Tamino XML Server can function as a convenient server in which to cache an XML representation of a "back office"; XML applications can be built on top of this clean XML view of the underlying complexity.

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

Second, the *Tamino XML Server protects investments* in core IT systems that need to be exposed to the outside as XML. Taking advantage of the Internet and e-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 integrate existing systems both at the data level and at the application level, and is particularly strong in conjunction with Software AG's EntireX and its proven track record of integration success.

Third, *Tamino XML Server minimizes the Total Cost of Ownership* of integrated electronic business applications. For information that has already been put in XML format, the 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 any other approach taken by post-relational "universal databases", 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 systems 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 easily usable XML view.

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 the Tamino XML Server is built throughout on standards, in particular those relating to XML and the Internet, the IT staff required to manage Tamino need little knowledge other than that related to these standards. This makes personnel recruitment easier.

Fourth, using the *Tamino XML Server as a staging platform offers significant performance and scalability advantages* over other data storage and management 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 remain unaffected. In addition, the staging server can "compile" information from many sources (for example, multiple online stores) and present the information in a single view to the customer. Tamino XML Server with its native XML storage, its ability to communicate with back-end systems and its open, Web-oriented architecture is an ideal engine for such implementations.

Index

A

access to external datasources, 16
administration, 16
Application Programming Interface (API)
    for .NET, 21
    for Java, 21
    HTTP Client API for ActiveX, 21
    HTTP Client API for Java, 21
    HTTP Client API for JScript, 21
architecture
    Tamino, 9

C

components of Tamino, 19
Content Management, 5

D

Data Map, 13
Data Store, 11
DBMS (Database Management System), 5

E

E-Commerce, 5
Enterprise Portals, 5

I

Interactive Interface, 20

N

native XML, 5

O

Object Composer, 11
Object Processor, 11

P

product components of Tamino, 19

Q

Query Interpreter, 11

R

RDBMS (Relational Database Management System), 5
relational database, 5

S

schema
    in Data Map, 13
Schema Editor, 20
server extensions, 16
staging server, 5
System Management Hub, 16

T

Tamino Manager, 16
Tamino product components, 19
Tamino XML Server, 4
    major parts, 11

U

user-defined function, 16

X

X-Machine, 11
X-Node, 15
X-Plorer, 20
X-Tension, 16
XML Parser, 11
XML-Engine, 11