Matisse
The Object Developer's Database

Data Sheet

"Matisse allowed us to deliver the carrier class reliability our customers demanded and met our time-to-market requirements".

Dave Hecht, Development Manager
Threshold Networks

Matisse is the ideal DBMS for network infrastructure developers who need to rapidly develop and deploy scalable object-based applications and services. Matisse's proven architecture, and patent pending technology, uniquely combines native support for objects, XML, and SQL with carrier class reliability, a small footprint, high performance and scalability, and minimal administration.

The Matisse Advantage

- Reduces application code up to 70% by eliminating O-R mapping
- Native object, XML, and SQL support simplifies development
- Carrier class reliability protects mission critical applications
- Unique versioning engine significantly improves performance
- Dynamic schema evolution eliminates schema modification downtime
- Minimal administration with no dedicated DBA
- Comprehensive language support future proofs application development
- Small footprint is ideal for embedded applications

Reduces Application Code up to 70%

Object applications are based on the use of complex programming language objects, such as Java classes, that rely on powerful mechanisms like inheritance and hierarchy, and XML which is itself a complex, hierarchical data format. Using relational databases in conjunction with object programming languages like Java leads to what is commonly referred to as an "impedance mismatch." The only solution for impedance mismatch is to map objects to relational tables, referred to as "O-R mapping." Often as much as 30% to 70% of the code in a Java (or C# or C++) application must be devoted to O-R Mapping. Matisse's native object support eliminates the overhead and added complexity of object-relational (O-R) mapping.

Native Objects, XML, and SQL

The Matisse DBMS is a natural database for handling XML documents since XML hierarchical elements can be mapped directly to object structures. Matisse provides an object API to manipulate XML documents, as well as an XML utility that provides automatic loading and generation of XML documents through SQL queries. The utility maps XML documents to a pre-defined schema, performs batch loading into the database, or exporting from the database, and validates the structure of the XML documents against the object data schema. Matisse is unique in its native object, XML, and SQL support – the key elements of applications with diverse data management requirements.

Best Price / Performance Ratio

Matisse customers often enjoy 100X price/performance improvements when switching from relational and other database technologies. Matisse's unique versioning engine can yield significant improvements in performance while maintaining database integrity for a broad class of applications. Matisse is designed to maximize reading while writing, making it possible to load/update large volumes of data while concurrently serving queries or other processing tasks. When an object is updated, the versioning engine creates a new version of the object in a new location, instead of updating the old version in place. Applications are able to read the previous version of the object while the new version is being created, and because all references are left intact, applications see a consistent view of the database at all times. If a transaction fails, requests simply continue to be directed to the previous version; there is no need to incur downtime by rolling back the database in order to achieve a consistent state.

Minimal Administration

Matisse is designed to run without a dedicated DBA, only requiring minimal administration once in operation. Many of the tasks that typically require downtime, such as adding or removing disks, modifying schema, or performing bulk loads are done concurrently with database activity, incurring no significant performance degradation or downtime. Matisse's versioning engine guarantees database recovery, but it does not use a transaction log file like most other databases. Thus, Matisse eliminates all of the burdensome tasks associated with administering log files – making Matisse the ideal database for embedded or remote systems, or sites with limited IT staff, where administrator intervention is either difficult or impossible. ►

Download a developer's version of Matisse 6.0 at **www.matisse.com.**

Carrier Class Reliability

Inherently reliable, Matisse utilizes automatic disk mirroring in order to provide uninterrupted service in the event of disk crashes. The system automatically reconfigures itself; so no system administration is required to implement this capability. Server based replication also allows a Matisse database to be automatically replicated to one or more systems, ensuring continued high availability during system failures. Beyond simply surviving systems failures, reliability also means being able to provide predictable response times and service levels by meeting the daily usage challenges encountered in today's dynamic computing environments. To meet this additional reliability challenge, Matisse tunes itself by automatically adjusting to changes in the environment, such as disk load imbalance and changing usage patterns.

Dynamic Schema Evolution

Matisse is one of the few database management systems that enables database schema to be modified during processing. This feature, termed Dynamic Schema Evolution, improves availability and is of critical importance in constantly changing environments where the data model changes as new services and storage is added or removed. Dynamic schema evolution allows for the addition or removal of classes, or properties, while Matisse is online.

Future Proofed Applications

Matisse is language independent, so data created using one programming language can be accessed by applications using other object-oriented or scripting languages. Matisse provides support for most popular programming languages: SQL, Java, C++, C#, Perl, Python, PHP, Eiffel and C in addition to ODL and DDL. Matisse supports the JDBC/ODBC standard for SQL access from applications in Java and other languages, and supports the EJB standard and the J2EE and .NET frameworks. Developers using native programming languages to manipulate persistent data can build and deploy applications quickly and easily, as there is no proprietary language learning curve.

Platform support & Hardware requirements

Platform	OS version	Minimum RAM	Disk consumption
Sparc/Solaris 32 bit	2.7+	128 MB	24 MB
Sparc/Solaris 64 bit	2.7+	128 MB	24 MB
Intel/Solaris	2.8+	128 MB	31 MB
Intel/Windows	NT4.0	128 MB	13 MB
Intel/Windows	2000	128 MB	13 MB
Intel/Linux	Redhat 6.2+	128 MB	17 MB
Intel/FreeBSD	4.2+	128 MB	16 MB

Database features

Native object support

J2EE and .NET support

Database replication

Object persistence

Multiple inheritance

Inter-object references

Encapsulation

Polymorphism

Unique object identifiers

Online backup

Full transaction support, locking

Server -side ANSI SQL 99 support

Stored procedures, triggers, referential integrity

Versioning engine

Automatic generation and control of object versions

No transaction log file

No "dirty reads"

Scalability and performance

Server side method execution (SQL stored procedures)

Scales linearly with CPUs for SMP architectures

Eliminates processing-intensive joins and denormalization

Language support

Java

C, C++

SQL

PHP

ODL

Eiffel

Python

Perl

VB, VBS, ADO, COM, C#