文件编号：**ORIENTX-V1.0-ARCHITECTURE**
版 本 号：**V1.0**
受控状态：受控
发 放 号：

OrientX 数据管理系统

中国人民大学 IDKE 实验室
<OrientX@ruc.edu.cn>

1.OrientX系统概述

1.1 OrientX 系统是什么系统？

OrientX 是一个 Native XML 数据库管理系统（Native XML DataBase Management System）。OrientX 是 **O**riginal **R**UC **I**DK**E** **N**a**t**ive **X**ML 的缩写，表示 OrientX 系统是中国人民大学 IDKE 实验室研发的 Native XML 数据管理系统。该系统的研发工作是从 2001 年至今，已经历时 3 年时间。

1.2 OrientX 系统的应用背景

随着因特网应用的发展，XML 逐渐成为数据描述和数据交换的标准，大量的 XML 文档出现在网络中；有效地存储 XML 数据并提供高效的 XML 数据查询，成为当前急需解决的问题。

最直接的处理办法就是，把 XML 数据存储到关系数据库中，并使用 SQL 查询作为数据存取的方法。这种方法可以利用现有成熟的关系数据库产品；它可以依据成熟完备的关系数据库理论来解决数据的存储和存取过程中的各种问题。

但是，用关系数据库来存放 XML 数据存在很多问题：将树状结构的 XML 数据转换成关系数据库的二维关系表形式时面临语义信息丢失的问题；XML 查询（例如 XPath 和 XQuery）等不能直接在关系数据库上执行，需要转换成 SQL 查询；而且其关系表形式的查询结果还必须得还原成树状形式的 XML 数据，这将会导致效率的严重下降。

因此，人们开始研究新的数据存取方式。OrientX 系统正是在这样的应用背景下产生的，它以 Native 方式存储 XML 数据，保留 XML 数据的树状模型，并支持 XPath 和 XQuery 等 XML 查询以读取数据。所谓的 XML 的 Native 存储方式，就是存储时保留数据的树形模式；根据一个结点可以直接找到其孩子结点、左右兄弟结点或父亲节点等。以 Native 方式存取 XML 数据，就无需进行数据模式的转换，也不需要进行查询语言的转换。

1.3 系统功能和系统特征

OrientX 是 Native XML 数据库管理系统，具有如下的**系统功能：**

1. **数据库的建立和维护**

   OrientX 系统的把具有相同的或相似的模式（schema）的 XML 文档集合存放到同一个数据库中（在 OrientX 中称为数据集 DataSet）。此功能包括 Schema 的导入和导出，数据库的建立和删除等。

2. **数据操纵.**

   此功能包括对数据库数据的检索、插入、修改和删除等基本操作。

   当前版本的 OrientX 系统的数据检索支持了 XPath；下一个版本将支持 XQuery。

   数据的插入，修改，删除等功能在 OrientX 系统 version1.0 中都是通过 API 接口来提供；预计下一个版本将提供类似于 SQL 中的 Update 语句的操作语言

3. **数据库运行管理**

OrientX 系统目前支持简单的数据库运行管理

4. 数据组织、存储和管理功能

对 XML 数据、模式信息、存取路径等数据进行分门别类地组织、存储和管理，确定以何种文件结构和存取方式物理地组织这些数据，如何实现数据之间的联系，以便提高存储空间利用率以及提高查找、增、删、改等操作的时间效率。

5. 数据通信接口

OrientX 系统提供一套程序开发的 API 接口;用户通过此接口可以开发自己的产品，实现数据的存储、检索和管理。

值得注意的是，与现在成熟的关系数据库管理系统产品相比,OrientX 系统还没有支持下列的功能：例如并发控制等数据库运行管理；数据库的转储与恢复、数据库的重组织与重构造、性能的监视与分析等系统维护功能。

OrientX 系统作为一个数据库管理系统，它具有如下的**系统特征：**

1. 它是基于模式（Schema based）的 Native XML 数据库管理系统。

Native XML DBMS 有是否基于模式（schema-based）之分。所谓基于模式，就是创建数据库时需要指定相应的模式信息；对数据的检索，更新等操作也需要指定其操作对象的模式。OrientX 系统是基于模式的，它为每个数据库保存了其相应的模式信息。因此查询数据或修改数据时只需要说明操作对象所在的数据库即可。

有了模式，就可以导入的 XML 文档是否符合指定的模式；在查询时，可以利用模式信息加速查询处理。

2. 多样化的数据组织和存储方式

所谓数据组织和存储方式，就是如何划分 XML 数据组成一个个的记录，如何存储这些记录的关系等。OrientX 系统支持四种存储策略：Element-based(EB)、Subtree-based(SB)、Clustering Storage (LC)、Clustering Element-based (CEB)。根据用户查询数据的需求和针对文档的不同特点，系统将会选择合适的存储策略。当文档比较小的时候，采用 EB 或者 CEB 方法；当文档比较大的时候，使用 LC 和 CEB 方法。对于文档性质比较强的文档，采用 EB 或者 CEB 方法；对于数据性质比较强的文档，采用 LC 或者 CEB 方法。

3. 数据存取路径

所谓数据存取路径，是指如何从数据库中读取数据，是顺序扫描、还是通过索引随机读取等？OrientX系统提供最常见的类似于DOM接口，可以根据当前XML数据结点导航遍历其父亲、孩子和左右兄弟结点；此外还支持三种索引：基于元素的索引（Elementary Index），路径索引（Path Index）和值索引（Value Index）。

所谓基于元素的索引（Elementary Index），是指对元素名称（element name）进行索引；有了此索引，就可以一下子读取到指定名称的所有元素结点在数据库中的地址（Element node）。

所谓路径索引（path index），就是对一个 XPath 路径进行索引；通过该索引，可以返回匹配指定路径的元素有序对。例如对路径 A//D 建立索引，通过此索引就可以返回一系列的(A,D)有序对，它指示着符合路径 A//D 的嵌套关系（父子或祖先后代关系）的结点对在数据库中相应的地址。

所谓值索引（Value Index），类似关系数据库中的索引，它可以直接定位指定的 text 值或 attribute 值的元素结点。例如，假设在 book/@price 上建立值索引了，那么我们通过该索引就可以定位 price=100 的元素结点。

4. **数据模型遵循XQuery 1.0 and XPath 2.0 Data Model标准。**
5. **数据检索支持 W3C 推荐的 XPath 查询标准**

2. OrientX 系统的设计

本节介绍 OrientX 系统的总体框架和各个组成模块的功能。至于各个模块的概要设计请参照其相应的概要设计说明书。

2.1 总体设计（Architecture Design）

描述 OrientX 系统的总体框架的最好方法是，通过检查它是如何导入一个 XML 文档并存储到数据库中去；然后在其上进行一个 XML 查询，看看它是如何实现的。

如图 1 所示，XML 文档通过 Data Manager 从逻辑上划分成多个记录；每个记录的信息通过存取模块(Access Manager)传送到存储模块（Storage manager）；存储模块（Storage manager）将多个的逻辑记录组装到一个物理页面，并把它存储到磁盘上（由文件管理模块 File Manager 负责）。当导入 XML 文档时，还需要指定文档要导入到哪个数据库中，也就是要指定该文档的模式（Schema）信息，（前面已经说过，OrientX 系统中每一个数据库对应于具有相同的模式 shema 的数据集 dataset）。当需要对数据进行查询检索时，一个 XML 查询（XPath 或者 XQuery 查询）以文本的形式传送到查询执行引擎（XPath&XQuery Execute Engine）；在查询执行引擎中，XML 查询将被分析(parse)成一个查询执行计划，此过程中从模式管理模块（Schema manager）读取相关信息，判断该查询是否存在语义错误，例如目标文档或数据库是否存在，XPath 路径中的结点在对应的模式中是否存在等问题；如果存在这样的错误，则系统就报告错误，查询不再往下执行。查询执行引擎还可以对查询计划进行优化；如果存在合适的索引可以优化查询执行效率，查询执行引擎就可以通过索引管理模块（index manager）直接访问数据库，而不需要通过数据管理模块（Data Manager）导航式地访问数据库。在访问数据库时，缓冲区管理模块（buffer manager）缓冲了最近访问过的数据块，从而减少了 I/O 次数,提高了执行效率。

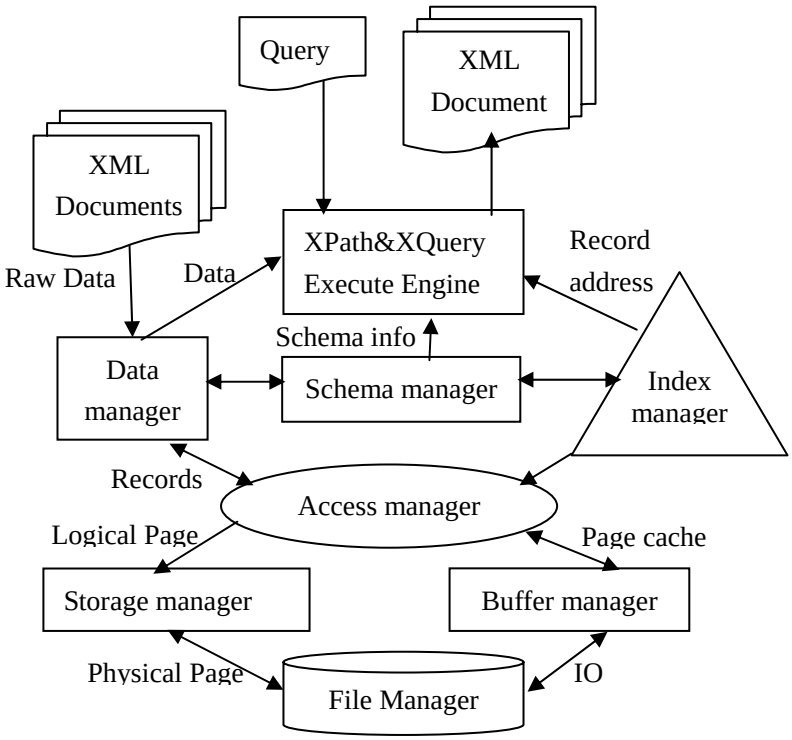

图 1　**系统框架图**

2.2 文件管理(File Manager)

OrientX 系统仍然是建立在操作系统之上，它是以文件的方式对磁盘的访问的。例如，OrientX 系统为每个数据库（数据集）向操作系统申请了几个大的文件的，该数据库中的数据（包括模式信息等）都是存放在这些文件上的。

文件管理模块（File manager）就是为整个系统负责管理所有的涉及磁盘文件创建、打开、删除和读写的操作，实现文件读写指针的共享，为上层存储模块提供了独立于操作系统的接口；这样，上层模块对磁盘的访问，就可以直接对“磁盘页面”进行读写删除等访问，而不需要知道这些磁盘块如何组织在文件中的。

这模块暂缺概要设计文档。

2.3 存储管理(Storage Manager)

本模块的主要功能是，提供统一的接口，使上层尽量独立于本部分的具体实现方式。

本模块提供的基本操作集是：对 DataSet 的建立和删除；对 DataSet 内 XML 文档的建立和删除；向上层模块提供一个物理页面为单位进行页面申请和读写操作的接口，管理系统物理存储空间的分配与回收。

此模块的概要设计请参照文档“Storager 概要设计”

2.4 缓冲区管理(Buffer Manager)

OrientX 系统为了减少 I/O 次数，缓冲了最近访问过的数据块。

上层的模块对缓冲管理模块的基本操作有：读一块物理页面到缓冲区、释放一块缓冲区、申请一块新的物理页面、释放一块物理页面（注意，所有对数据库的操作都通过缓冲管理模块，所以申请和释放物理页面也是通过缓冲模块，再由缓冲模块调用底层的存储模块的方法）。

页面淘汰算法将使用最近最少使用算法。

关于此模块的概要设计请参照文档“BufferManager 概要设计”和“缓冲管理模块设计”

2.5 存取管理(Access Manager)

此模块的功能主要是，对存储模块(Storage Manager)和缓冲区模块（Buffer Manager）进行包装，向上层数据管理模块提供统一的接口，使上层直接通过这个接口来存取数据。

关于此模块的概要设计请参照文档“数据存取模块概要设计说明书”

2.6 数据管理(Data Manager)

DataManager 将 Parser 后的 Dom 树转化为实际存贮的物理纪录（导入文档），同时提供接口，使得上层用户能够根据需要在已存储的 XML 文档节点间移动（导航式访问数据），并能够对已存在的 XML 文档节点进行增加、删除和修改；并能都重新生成 XML 文档文件（导出数据）。

向上层查询模块提供接口的是 DataManager 类。它通过调用 Schema 类实现模式文档的解析和数据集的建立，通过调用 ImportHandler 类实现 XML 格式的文档的导入，上层模块利用它的导航接口可以提取想要的数据对象。

此模块的概要设计请参照“数据管理模块概要设计”和“DataManager 概要设计”，后者是于 2002 年 5 月写的；后者是于 2004 年 3 月写的；该模块的设计在这期间作了一些调整。

2.7 模式管理(Schema Manager)

SchemaManager管理的对象是数据集的Schema。Schema类似与关系数据库中的数据字典。数据字典说明了数据库中的表结构、索引结构等信息。在OrientX中，一个数据集对应于一个Schema，每一个数据集由多个符合同样Shema定义的文档组成。Schema有两大作用：首先，Schema类似于DTD或者XMLSchema的作用，说明了文档的结构特征。其次，Schema还记录了关于数据集的索引、引用和被引用关系等。

Schema Manager 在查询中的作用是：当一个查询提交时，首先用 Schema 检查该查询是否合法，比如路径是否存在，数据类型是否匹配等；如果合法，则根据 Schema 把查询中的 TagName 转换成结点名内部表示 EID。

关于此模块的概要设计，请参照文档“MetaDataManager 设计概要”。

2.8 索引管理(Index manager)

与关系数据库一样，为了加速数据访问的速度，Native XML 数据库管理系统也可以提供索引。OrientX 系统支持的索引有：基于元素的索引（Elementary Index），路径索引（Path Index）和值索引（Value Index）。

根据基于元素的索引（Elementary Index），就可以一下子读取到指定名称的所有元素结点在数据库中的地址（Element node）。此索引通常用于嵌套连接（containment join）。通过路径索引（Path Index），可以返回匹配指定路径的元素有序对。例如对路径 A//D 建立索引，通过此索引就可以返回一系列的(A,D)有序对，它指示着符合路径 A//D 的嵌套关系（父子或祖先后代关系）的结点对在数据库中相应的地址。值索引（Value Index）可以直接定位指定的 text 值或 attribute 值的元素结点。例如，假设在 book/@price 上建立值索引了，那么我们通过该索引就可以定位 price=100 的元素结点。

关于索引管理模块，暂缺设计文档。

2.9 XML 查询引擎(XML Query Engine)

在 OrientX 中，查询引擎由三个模块构成：查询分解（Parse）、查询优化（Query Optimizer）和查询执行(Execution Engine)。OrientX version1.0 支持 XPath 查询语言。输入为 XPath 语言描述的查询语句，输出为 XML 形式的结果文档

2.9.1 查询分析(Query parser)

XML 查询是通过交互式或文件批处理方式提交的；无论是哪一种的提交方式，在查询执行引擎（Query execute engine）看来，所接收到的查询语句都看作是**字符流**。显然地，这是不能够直接让查询执行引擎进行处理；必须把字符流形式的查询语句转换成 Xpath 处理器的可处理的内部结构，同时检查其是否符合语法(也就是判断其是否合法的 Xpath 查询语句)，满足一定的语义要求。完成这个工作的就是 Xpath 的查询分析。

经过此模块的分析，XML 查询被转换成内部结构，传递到查询执行引擎中去。

关于此模块的概要设计请参照文档“Xpath 查询分析”，相关的文档还有“XQuery Syntax Parse 概要设计”，“XQuery 文法规则的改写”

2.9.2 查询优化和执行(Query optimization and execution)

Optimizer 是 XPath 处理流程中的第二步，采用逻辑优化的方法，制定一系列优化规则，把 Parse 得到的内部路径根据规则转化成优化的查询计划。同时吸收了物理优化的思想，在优化规则中利用模式、索引等信息进行查询优化。

其关键的思想是：先利用模式信息对查询路径进行预处理，再用最小分解的方法处理复杂路径，对每一个子路径，再利用索引信息进一步分解，以获得最优的执行方案。

查询执行，就是按照一定次序去完成执行计划上的操作。执行的结果是一棵 XML 子树，可供进一步操作，也可以打印成一个 XML 文档。

关于此模块的概要设计请参照文档“XPath 处理模块概要设计”。

3.总结和未来工作

OrientX 系统是一个 Native XML 数据库管理系统，可供管理 XML 数据，具体包括导入

模式（schema）文档创建数据库、删除数据库、导入存储 XML 文档、导出数据库或 XML 文档、支持 XPath 查询等功能

OrientX 系统为 XML 数据管理相关研究工作提供了一个实验平台的。

OrientX 系统 version1.0 提供了 Client-Server 的应用体系结构；用户可以在客户端通过命令进行数据管理（建立删除数据库、导入导出文档、数据查询等操作）

OrientX 系统还提供了一套 API 接口，支持 XML 数据管理的应用开发。

关于OrientX系统的功能演示，请登录我们的主页：http://idke.ruc.edu.cn

如果用户需要OrientX系统相关的程序包、源代码、说明文档，请向我们索取：OrientX@ruc.edu.cn 或登录我们的主页：http://idke.ruc.edu.cn

关于 **OrientX 系统的未来工作**，我们有这样的设想：

1. 对系统进行全面的功能测试和性能测试
2. 进一步完善 XML 数据更新的工作
3. 在 XML 数据查询方面，支持 W3C 推荐的 XQuery 查询。
4. 在数据安全方面，支持用户访问控制。
5. 提供 GUI 形式的客户端访问界面