一种基于对象的多媒体存储系统优化策略*)

邹 强[1,2] 冯 丹[1,2] 曹 炬[3] 田 磊[1,2] 曾令仿[1,2]

(华中科技大学计算机科学与技术学院[1] 武汉光电国家实验室[2] 华中科技大学数学系[3] 武汉 430074)

摘 要 设计一个实用的分布式多媒体服务系统,存储子系统是其中的一个研究重点。根据多媒体系统的存储特点,从全新的视角提出一种存储子系统设计方案——基于对象的多媒体存储系统(Multimedia Object-based Storage System,MOSS),并分析了MOSS的系统架构和工作流程。与传统的多媒体存储技术相比,对象存储不仅能有效地改善系统I/O性能,还具有较好的安全性。以对象存储原型系统为基础,讨论了热点对象文件的分布和均匀分块优化策略。通过建立与系统相对应的数学模型,对数传率、I/O请求丢失率等性能指标作了定性分析。该研究工作对消除多媒体服务系统I/O瓶颈、进行存储子系统设计具有实际的指导意义。

关键词 多媒体系统,对象存储,I/O请求

Optimization Policy for Object-based Distributed Multimedia Storage System

ZOU Qiang[1,2] FENG Dan[1,2] CAO Ju[3] TIAN Lei[1,2] ZENG Ling-fang[1,2]

(School of Computer, Huazhong University of Science & Technology[1], Wuhan National Laboratory for Optoelectronics[2], Department Mathematic, Huazhong University of Science & Technology[3], Wuhan 430074, China)

Abstract Storage subsystem is one of the pivots to design an applied distributed multimedia system. This paper set up an object-based distributed multimedia storage system and discussed a optimization storage policy about the hot object file. Comparing with the traditional network storage technology, the object-based distributed multimedia storage system is not only able to improve the I/O performance of system, but also make the system more safe. By setting up the mathematical model corresponding to the system, this paper qualitatively examined some performance classes such as the loss rate of program access etc. This research is significant to remove the I/O bottleneck and design the distributed multimedia storage subsystem.

Keywords Multimedia system, OBS, I/O request

1 引言

多媒体系统是一个非常复杂的系统,对服务器性能有很高的要求。为了满足大规模多媒体应用和服务的需要,多媒体服务器除了需要有快速的数据流吞吐能力、尽可能小的响应时间,还需要具有海量存储能力,以存储大量多媒体文件并尽可能支持更多的并发访问用户数,尤其是热门节目源的并发访问数。由此看来,存储系统对多媒体服务器的性能具有极其重要的影响。一般情况下,MPEG-II节目的标准速率为4M/s,一个1小时的MPEG-II节目至少需要2GB的存储空间;而MPEG-II高清电视节目流的基本速率更是高达20～25Mbps,因而需要更高的存储能力。一个多媒体系统,至少需要数十个甚至数百个节目供用户选择。因此,即使每个节目在系统中只有一个拷贝,多媒体点播系统也需要数百GB的存储容量。虽然现在计算机的网络带宽、存储速度以及计算能力均有了非常大的提高,要实现支持大量用户的多媒体系统仍旧存在很多困难。

多媒体存储通常具有以下要求[1-3]:首先系统响应要具有实时性。客户端对多媒体数据的访问,一般都有QoS(Quality of Service)要求。如果在给定时间内请求没有得到响应,会导致服务器的QoS降低,用户的播放或显示就会出现抖动;其次是I/O请求量大,客户端对多媒体的访问请求尺度大且要求速度高[4];另外,通常情况下数据访问以读为主。尽管多媒体负载包括读和写两类,但是写请求不是经常发生的,并且没有实时性要求,服务器可以离线完成写请求。

因为廉价且有较高的性能,磁盘成为了多媒体服务器主要的外存设备。在多媒体数据存储中,以空间获取时间是比较简单有效的方法,传统的方法有Disk Mirror(disk shadow)[1,5]和Disk Striping[5]等,其中Disk Mirror在分布式多媒体服务器中被广泛采用[6],有效地改善了I/O性能,同时通过数据或文件的复制提高了分布式多媒体系统的可用性。而在大范围的多媒体并行服务中,节目的流行度则会使系统产生负载平衡方面的问题[7,16]。另外,多媒体系统中的海量存储数据要求存储系统安全、稳定,具有一定的容错能力,且易于管理,能使服务器集群共享所有存储数据。文献[8]探讨了采用RAID并行预取技术从软件系统角度增加数据的安全性和提高系统性能的方法。

本文引入一种全新的网络存储技术——对象存储(Object Storage,OS),构建了基于对象的多媒体存储系统(MOSS)。MOSS的关键思想在于应用智能化的对象存储设备对存储数据进行优化管理,向媒体服务器集群的应用程序提供对象级访问接口,存储的位置、迁移、性能和规模对多媒

*)基金项目:国家重点基础研究发展规划(973)项目(No. 2004CB318201),国家自然科学基金(No. 60603048)。邹 强 博士生,主要研究方向为I/O流量特征和网络存储系统;冯 丹 教授,博士生导师,主要研究方向为海量信息存储。

体服务器集群的访问具有透明性，并且允许多媒体服务器集群对对象存储设备并行读取数据，极大地提高了存储 I/O 效率，充分发挥了集群服务器的并发性能。本文以对象存储原型系统为基础，讨论了多媒体服务系统中热点对象文件的存储优化策略。通过建立与系统相对应的数学模型，本文对数传率、I/O 请求丢失率等性能指标作了定性分析。分析结果对消除系统 I/O 瓶颈、设计分布式多媒体存储子系统具有实际的指导意义。

2 对象存储系统及其实现

对象存储 OBS 的研究最早可追溯到 1980 年，麻省理工学院(MIT)第一次在他们的 SWALLOW 项目中实现了分布式对象存储，成为该领域的先驱。OBS 有 5 个主要组成部分[9,12]：客户机、对象存储设备(Object-based Storage Device，OBSD)、分布式文件系统、元数据服务器(Metadata Server，MDS)和互联网络(如图 1 所示)。对象存储设备的控制器称之为对象控制器(Object Storage Controller，OSC)，在多媒体服务中起存储服务器的作用。OBS 兼有网络附加存储(Network Attached Storage，NAS)和存储区域网(Storage Area Network，SAN)的优点，不仅具有良好的可扩展性、可用性、可靠性，还具有较好的安全性。OBS 与传统存储技术的存储特性对比见表 1。

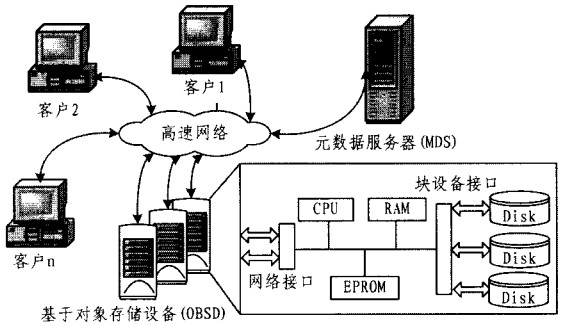

图 1 OBS 体系结构图

OBS 采用了富有表现力的对象接口，在数据共享、安全性及智能化方面得到突破。

(1) 数据共享

OBS 的数据共享得益于对象级的接口和对象具有描述数据性质的属性。基于对象的接口与文件系统的接口类似，可以像文件一样对对象进行删除、读、写、查询属性等操作。“文件级”的接口易于理解和标准化，为跨平台共享提供了便利。元数据服务器只须保存较少的全局性的元数据，易于保持一致性，为数据共享提供了方便。

(2) 安全性

正因为对象具有属性，使得 OBS 可以在以对象为基本单位的基础上，建立灵活的安全机制。正如传统文件服务器中的每个文件都具有不同的属性一样，OBS 可以对整个设备、一组对象、单个对象、甚至对象内的部分数据赋予不同的安全属性，分别实施认证访问。OBS 还可以对整个设备进行分区，每个分区的安全属性及访问规则由存储应用决定。另外，OBS 还可对每一次 I/O 操作进行授权。

OBS 将安全机制的实施(在 OBSD 中)从安全策略(在元数据服务器)中分离出来，OBSD 不必维护与用户(客户机)有关的安全信息，使得 OBSD 可以独立于客户机进行扩展，而在网络上的客户机也不必假设为可信任的。

(3) 智能化

对象存储控制器 OSC 自主地管理存储在其内部的对象，对数据块之间的逻辑关系不再一无所知，这使得存储设备进行自管理、自配置、自保护、自优化、自治愈和自组织成为可能，从而可以更好地服务于各类应用。

我们设计实现了基于对象的存储设备、元数据服务器和满足 POXIS 文件请求的客户端三方通讯的对象存储原型系统，并就数据组织、属性管理、基于对象存储系统的负载均衡等进行了研究[13-15]。本文以对象存储原型系统为基础，进一步结合多媒体应用探讨了一种基于对象的分布式多媒体存储系统架构，并进行了相关性能分析。

3 基于对象的多媒体存储系统体系结构

由于 OBS 在数据共享、安全性及智能化方面具有优势，本文将 OBS 引入多媒体服务系统，构建了基于对象的分布式多媒体存储系统，支持多用户访问。

在基于对象存储的分布式多媒体服务系统中，元数据服务器负责用户点播行为的认证和系统数据管理。节目文件的数据存放在 OBSD 中，OSC 充当存储服务器，数据分布的方式由元数据服务器(MDS)确定。

元数据服务器是该分布式系统的中心，它可以和点播服务器运行于同一硬件平台上，其功能分为用户点播和系统管理两部分。用户进行多媒体点播的过程如下：

①用户通过浏览器，向点播服务器发送节目点播请求。

②点播服务器验证用户信息、处理用户请求后向元数据服务器发出读取多媒体对象的请求。

③元数据服务器向点播服务器返回对象逻辑视图和访问信任状。其中，对象逻辑视图包括每个 OBSD 中存储的媒体对象组件的信息(如 OSC 的负载情况、节目情况和与点播用户的距离等)和一个合适的 OSC 地址(被访问的数据存放在对应的 OBSD 上)。一方面，当媒体文件足够大时，每个文件的条带宽度仅和 OBSD 的数量有关；另一方面，拥有对象逻辑视图是点播服务器自治地直接访问 OBSD 的前提条件。访问信任状，类似于安全令牌，允许特定的点播服务器在特定时限内通过特定指令读取数据；元数据服务器同时设置了回叫命令，在任何点播服务器修改文件时通知集群中其他点播服务器同步更改数据，确保了点播服务器中的缓存一致。

④点播服务器将访问信任状和读取数据指令打包后直接发送给 OBSD，请求多媒体数据。

⑤OBSD 验证访问信任状后向点播服务器并行传输被请求的多媒体数据，同时并行等待接受点播服务器的下一个请求指令。

⑥点播服务器通过适当调度策略将媒体文件传输至用户。

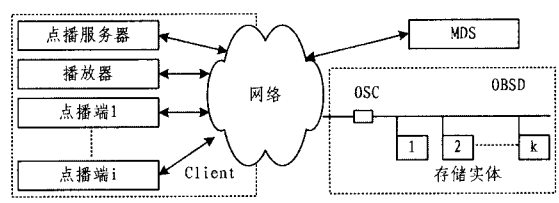

图 2 单 OBSD 多媒体服务系统结构拓扑图

点播服务器、播放器和若干个点播端构成一个基本点播单元，简称为客户端(Client)。各个 Client 通过网络与元数据

服务器和存储服务器连接。单个 OBSD 的多媒体服务系统结构如图 2 所示：由网络与点播端构成的虚框为 Client；OBSD 上挂有 $k(k\geqslant 1)$ 个存储实体（每个存储实体可以分别是单个的磁盘或磁盘阵列等）。OBSD 包括 CPU，Memory，网络接口以及块设备接口，管理对象存储空间的分配与数据组织，负责将对象映射为块设备上的块。OBSD 具有集成处理能力，可实现设备对对象数据的智能化自主管理。

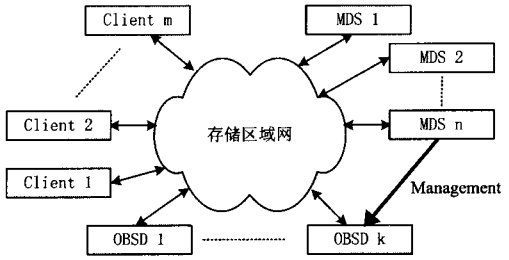

图 3　多 OBSD 多媒体服务系统结构拓扑图

基于对象的分布式多媒体存储系统通常由多个 OBSD 组成，其体系结构如图 3 所示。在系统中，每个节目文件通常仅存储在一个 OBSD 内，在每个 OBSD 中可以存储多个文件；存储服务器(OSC)具有良好的安全性、并行性和可扩展性，且每个对象存储设备都具有一定的智能，能够根据存储策略自动管理其存储的数据。

4　存储优化策略与系统性能分析

4.1　热点对象文件均分策略

由于在多媒体应用中，对象文件访问具有高度局部性的特征，比如在服务器中，不同对象文件的访问频率不同，即使同一个对象文件内部，不同的数据块的负载也是不同的。用户对对象文件的访问倾向性，使得热点对象文件被频繁访问，造成了系统热点瓶颈问题。本文以对象存储原型系统为基础，介绍了一种基于热点对象文件的均匀分块优化策略。将热点数据分段存储在 OBSD 上既能改善性能，又能减少文件复制所需的巨大容量开销。

在 OBSS 中，热点对象文件均匀分块策略是由 MDS 和 OBSD 共同完成的。以近期内对象文件被点播次数（即对象被点播频率）为准则判断热点对象是否存在，如果某对象文件近期内的被点播频率达到一个预先设定的阈值，则认为该对象文件为热点对象文件。为了保证系统的 QoS，通过系统监控程序统计用户的服务请求，当 OBSD 发现某对象文件的被点播频率达到阈值时，由 OBSD 发起实施热点对象文件均分策略，将热点对象文件均匀分块（分块大小由 MDS 来决定），分别存放到不同的 OBSD 上，同时淘汰那些服务器上点播频率低的对象文件。MDS 完成初始对象的均匀放置，对分块对象产生均匀访问；在系统运行过程中由 OBSD 发现产生的热门对象，并由所有的 OBSD 协同完成分块存储；OBSD 则在分块存储过程结束后向 MDS 报告对象存储位置的变更。在设计分布式多媒体对象存储系统时，可以使用只容纳部分对象文件的存储服务器，这些对象文件为该系统中的热点对象文件。

均匀分块策略是一种实时策略，建立在高带宽低延迟的高速网络之上。随着网络技术的飞速发展，实时的均匀分块策略是可行的。虽然将热点对象数据分块存储到不同的 OBSD 上，会造成额外的系统开销，但由于实施均匀分块策略后，各个 OBSD 将并行传输数据，因此，本文将在各 OBSD 上由于分块存储策略所引起的额外开销记为 $\Delta t(i)$，其均值为 $E[\Delta t(i)]$。另外，在客户端对数据进行整合的开销可忽略不计。

对于热点对象文件均匀分块策略在改善系统各项性能指标方面的有效性，可以通过建立相应的数学模型来进行定性分析。本文将着重研究热点对象文件均匀分块策略对数传率和 I/O 请求丢失率带来的影响。

4.2　系统性能分析

由于基于对象存储的多媒体服务系统固有的复杂性，为了简化系统，不妨先假定基于对象的多媒体存储系统中只有一个支持 N 个用户的 OBSD，里面仅存储一个传输平均时长为 T 的对象文件，且客户端以泊松流随机地请求服务。不妨设第 i 个客户端的请求时刻为 t_i，$i=1,2,\cdots$，则前 N 个用户发出请求后，便能立刻得到服务，服务平均时长为 T。若在第 $N+1$ 个用户请求到达前，第一个用户已经服务完毕。即 $t_1+T\leqslant t_{N+1}$，则第 $N+1$ 个请求可以马上得到响应；否则，即 $t_1+T>t_{N+1}$，(t_N,t_1+T) 时间区间内的请求被拒绝。t_1+T 时刻后，第一个用户离去，系统可以响应新的请求。以此类推，与系统相对应的时空图如图 4 所示。由图 4 可知，对象访问由连续的周期组成，每一周期时长为 t_1+T，且由接受阶段和拒绝阶段组成，其中接受阶段的时长为 t_N，拒绝阶段的时长为 t_1+T-t_N。

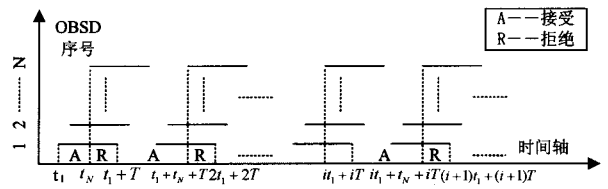

图 4　多媒体服务系统时空图

约定：OFS 表示对象文件大小；THR 表示数传率，也就是带宽；NRR 表示被拒绝的访问数量；TNR 表示访问的总数量；LRP 表示请求被拒绝的时间；TTL 表示总的访问时间。根据泊松流的性质，在 t_i 时间长度内，请求的平均数为 λt_i，t_i 的平均值为 i/λ。因此，对象文件的 I/O 请求拒绝率可表示如下：

$$\eta_{LOR}=\frac{NRR}{TNR}=\frac{\lambda\cdot LRP}{\lambda\cdot TTL}=1-\frac{N\cdot THR}{THR+\lambda\cdot OFS} \tag{1}$$

因此，要减小 I/O 请求被拒绝的概率，必须减小 $T(T=OFS/THR)$，即减少 OFS 或提高 THR。在基于对象的分布式多媒体存储子系统中，每个对象文件的 I/O 请求拒绝率也可用(1)式来表示。当热点对象文件出现时，实施均分策略，将对象文件分为 k 段，每段分别存放于不同的 OBSD 内，每段的数据分为若干个块，块大小由 MDS 决定。假设第 i 段的传输时长为 $T(i)$，$i=1,2,\cdots,k$，而由均分策略引起的额外开销分别为 $\Delta t(i)$，记 $E[\Delta t(i)]=t$，则系统 I/O 请求丢失率为

$$\eta_{loss}=1-\frac{N}{1+\lambda kt+\lambda\cdot\max[T(i)]},i=1,2,\cdots,k \tag{2}$$

当 $T(i)=T(i+1)=T/k$，$i=1,2,\cdots,k-1$ 时，有

$$\eta_{loss}=1-\frac{N}{1+\lambda kt+\frac{\lambda T}{k}}$$

$$=1-\frac{k\cdot N\cdot THR}{k\cdot THR+k^2\cdot THR\cdot\lambda t+\lambda\cdot OFS} \tag{3}$$

由(3)式可知，I/O请求丢失率受数传率THR、实施策略要用到的OBSD个数 k 和系统开销 $\Delta t(i)$ 等因素的影响。由 $\lambda kt+\frac{\lambda T}{k}\geqslant 2\lambda\sqrt{Tt}$ 知，当 $\lambda kt+\frac{\lambda T}{k}$ 取最小值时，η_{loss} 最小。与(3)式相对应的走势图如图5所示。其中横轴表示实施对象文件均分策略所用到的OBSD个数 k，纵轴表示I/O请求丢失率 η_{loss}。显然，越是热点对象文件，其服务请求到达率就越高，其相应的I/O请求丢失率也就越高。此外，由图5可知，图中的每条曲线都存在一个很明显的拐点，η_{loss} 在拐点处取得最小值。由此可见，在实施均匀分块策略的过程中，并不是将热点对象文件分块越多，I/O请求丢失率的改善效果越好，而是存在一个最优的分块数 $k_{opt}=\sqrt{\frac{T}{t}}$，使得系统开销的增加对系统性能改善影响最小，从而使得 η_{loss} 降到最低。另外，(3)式还反映了 η_{loss} 与THR的紧密关系，在对象文件大小OFS一定，且其他因素不变的情况下，THR越大，其相应的 η_{loss} 越小。这说明 η_{loss} 的变化与THR息息相关。

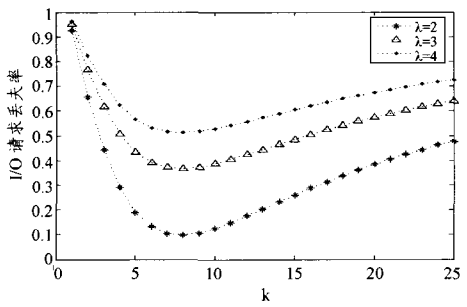

图5 各个相关指标关系图

结束语 本文将一种新的网络存储架构——对象存储应用于多媒体服务系统，提出并构建了MOSS，这将是发展下一代多媒体服务系统，尤其是建设超大规模多媒体应用的一个方向。此外，本文还讨论了热点对象文件的存储优化策略，通过建立与系统相对应的数学模型，对数传率、I/O请求丢失率等性能指标作了定性分析。该策略对消除系统I/O瓶颈、设计分布式多媒体服务系统具有实际的指导意义。

参 考 文 献

[1] Bitton D, Gray J. Disk shadowing// The 34th IEEE COMPCON. San Francisco, CA, 1998

[2] Ghandeharizadeh S, et al. On disk scheduling and data placement for video servers. Tech Rep, USC-CS-97-650. USC, USA, 1997

[3] Shenoy P J, Vin H M. Efficient striping techniques for variable bit rate continuous media file servers. Performance Evaluation, 1999, 38(3/4): 175-199

[4] Schindler J, et al. Track-aligned extents: Matching access patterns to disk drive characteristics// Conf. File and Storage Technologies (FAST). Monterey, CA, USA, 2002

[5] Yu Xiang, Gum Benjamin, et al. Trading capacity for performance in a disk array// Symposium in a disk array. Symposium on Operating Systems Design and Implementation. San Diego, CA, USA, 2000

[6] On G, Zink M, et al. Replication for a distributed multimedia system// The 8th Int'l Conf. Parallel and Distributed Systems. Kyongju City, Korea, 2001

[7] Wu Song, Jin Hai. Symmetrical pair scheme: A load balancing strategy to solve intra-movie skewness for parallel video servers // The Int'l Parallel and Distributed Processing Symposium. Marriott Marina, Fort Lauderdale, Florida, 2002

[8] 李宇，张江陵，冯丹. 一种面向视频播放系统的RAID并行预取技术及实现. 计算机研究与发展，2002，39(11)：1526-1530

[9] Mesnier M, et al. Object-Based Storage. IEEE Communications Magazine, August 2003

[10] 李钢江，杨士强. 分布式多媒体点播系统的结构设计. 小型微型计算机系统，2001，22(3)：257-260

[11] 孟玉柯. 排队论基础及应用. 上海：同济大学出版社，1989

[12] Factor M, et al. Object storage: the future building block for storage systems. IBM Haifa Research Laboratories, 2005

[13] Zeng Ling-fang, Feng Dan, et al. Object replication and migration policy based OSS// Proceedings of the Fourth International Conference on Machine Learning and Cybernetics. Guangzhou, 2005

[14] Zeng Ling-fang, Feng Dan, et al. A strategy of load balancing in object storage system // Proceedings of the Fifth International Conference on Computer and Information Technology (CIT'05). 2005

[15] 覃灵军，冯丹，曾令仿，等. 基于对象存储系统的动态负载均衡算法. 计算机科学，2006，33(5)：88-91

[16] Little T, Venkatesh D. Popularity-based assignment of movies to storage devices in a video-on-demand system. Multimedia Systems, 1995, 2(8): 280-287

[17] 靳超，郑纬民，张悠慧. 主动存储系统结构. 计算机学报，2005，28(6)：1013-1020

[18] 陈鑫林. 现代通信中的排队论. 电子工业出版社，1999

(上接第75页)

[2] RFC 2779. Instant Messaging / Presence Protocol Requirements. http://www.ietf.org/rfc/rfc2779.txt, 2000

[3] RFC3920. Extensible Messaging and Presence Protocol (XMPP): Core. http://www.ietf.org/rfc/rfc3920.txt, 2004

[4] RFC 3921. Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence. http://www.ietf.org/rfc/rfc3920.txt, 2004

[5] RFC 3922. Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging. http://www.ietf.org/rfc/rfc3922.txt, 2004

[6] RFC 3923. End-to-End Object Encryption in the Extensible Messaging and Presence Protocol (XMPP). http://www.ietf.org/rfc/rfc3923.txt, 2004

[7] SIP for Instant Messaging and Presence Leveraging Extensions (SIMPLE). http://www.ietf.org/html.charters/simple-charter.html, 2007

[8] Zhang Yun-chuan. Standardized Instant Messaging Protocols: Comparative Analysis of SIMPLE and XMPP. Journal of Wuhan University of Science and Technology (Natural Science Edition), 2005, 4

[9] Petrack S. SIMPLE Aims for IM Interoperability. Network World, 2004, 21(3): 39

[10] Hildebrand J. XMPP Transports Presence Data. Network World, 2004, 21(10): 31

[11] van der Vlist E, Ayers D, Bruchez E. Professional Web 2.0 Programming, Wrox Press, 2007

[12] Gómez-Pérez A. The Semantic Web: Research and Applications. Springer Press, 2004

[13] OMG's CORBA. http://www.corba.org/

[14] Mobile Agent. http://en.wikipedia.org/wiki/Mobile_agent

[15] Tiecke S. Open grid services infrastructure (OGSI) Version 1.0. http://www.globus.org/toolkit/draft-ggf-ogsi-gridservice-33_2003-06-27.pdf