

Oracle9 iAS 高速缓存解决方案

Oracle 技术白皮书

2001 年 12 月

ORACLE

Oracle9iAS 高速缓存解决方案

概要.....	3
实现高速缓存.....	3
创建“可高速缓存 (Cache Aware)”的 Web 应用程序.....	4
高速缓存解决方案概述.....	6
浏览器高速缓存.....	6
代理高速缓存.....	7
内容传递网络服务.....	7
服务器加速器.....	8
应用服务器的演变.....	8
Oracle9iAS Web Cache.....	9
EDGE SIDE INCLUDES FOR JAVA (JESI).....	10
JESI 标记库.....	10
使用 JESI 标记库.....	10
应用程序层高速缓存.....	13
Java 对象高速缓存.....	14
Web 对象高速缓存.....	14
Web 对象高速缓存标记库.....	15
Oracle9iAS Web cache vs. Java-level Web Object Cache.....	16
部署可高速缓存的 WEB 应用程序.....	18
总结.....	20
参考文献.....	21

概要

因为 Web 站点将为数以千计的并发用户所使用，所以他们典型地对于速度和可伸缩性具有迫切的需求。Web 应用程序在不降低响应时间的条件下必须经常是全球可访问的。高速缓存是提高 Web 应用程序的速度和可伸缩性的有效方法。Web 站点也要求他们的内容对于特定的用户群体和区域是高度动态的和可定制的。因为传统的高速缓存是重复地重用相同的静态内容，所以动态内容的生成从根本上是与传统的高速缓存技术相冲突的。

在本文中，我们研究了由 Oracle9i 应用服务器 (Oracle9iAS) 针对这一难题所提供的具有创新性的高速缓存解决方案。解决方案本质上主要存在两方面的问题：

- 实现满足需求的高速缓存
- 使用高速缓存解决方案来创建“可高速缓存的” Web 应用程序

实现高速缓存

Web 高速缓存解决方案随着 Web 从浏览场所到交易场所的变化也在逐渐地发展。早期的解决方案通过在浏览器中进行高速缓存来试图使浏览更有效。当大部分的内容是静态的并且缺省状态下浏览器运行在台式机上时，浏览器高速缓存提供了有限的帮助。随着 Internet 服务提供商的出现和 Intranet 逐渐成为业界标准，因而 Web 用户显著地增加，所以也临时采用了在服务器端进行高速缓存的代理高速缓存技术。高速缓存或者自身就可满足请求，或者作为浏览器的代理将请求传送到服务器。当 Web 开始随处可见时，例如内容传递网络 (CDNs) 这样的更新的解决方案就出现在市场上，以设法满足全球分布的 Web 用户对于可伸缩性和响应时间的需求。另一种高速缓存策略的演变方案是服务器加速器 (或相反的代理高速缓存)，它代表一或多个特定的 Web 服务器而不是代表一组浏览器用户来工作。

随着 Web 内容变成高度动态的，变化内容的高速缓存也提出了更大的挑战。结合了多种高速缓存技术的新的解决方案已经开发出来，它在保证动态内容刷新的同时也能够满足全球分布的或大范围区域的 Web 用户群的性能需求。

创建“可高速缓存（Cache Aware）”的 Web 应用程序

为了从高速缓存解决方案中获得最大的效益，开发人员必须创建能够充分利用特定解决方案的应用程序。一个强大的技术结合就是使用 CDNs 和服务器加速器。这一结合提供了发送在 Internet 中生成的动态内容的能力。服务器加速器也逐渐地用于在企业的 intranet 内建立企业的 CDNs，它用于分配应用程序的交付到分支办公室。为了帮助开发人员创建“可高速缓存的”动态应用程序，一种称为 Edge Side Include (ESI) 的业界标准标记语言已经开发出来。ESI 遵循 World Wide Web 联盟 (W3C) 的标准，它使得 Web 应用程序的开发人员能够划分可缓存的页片段并且在网络上动态地将它们组合到整个 Web 页面中。适用于 ESI 标准的 Java API 以称为 JESI (ESI for Java) 标记的 Java 服务器页标记库的形式也可以获得。JESI 标记成为由 Java Community Process JSR 128 制定的 J2EE 标准的工作也正在进行中。

同 ESI 相比，更好的优化可以在 Java 应用程序层采用可编程的高速缓存技术来实现。在某些情况下，当 Web 开发人员能够重用或后处理被高速缓存的内容，他们可以使用对应用程序的深入认识的知识来直接提高性能。

本文提供了关于 Oracle9iAS 应用程序开发人员能够部署在他们的应用程序中的不同的高速缓存解决方案和技术的概述。它介绍了即使在用户数增长的情况下，为了迎接高速缓存动态内容和快速提供最新内容的挑战，Oracle9i 应用服务器发行版 2 所提供的高速缓存的特性和工具。

简介

电子商务模型对 Web 站点提出了新的性能需求。为了取得成功，Web 上的商务活动必须有很短的快速响应时间并且能够防止由于高峰负荷所造成的系统运行中断。Web 站点必须足够快以使 Internet 用户满足。传送内容的延迟可能会使潜在的用户转向竞争者一面。多个市场调查的研究都强调了响应时间对于电子商务的重要性¹。

尤其是在经济困难时期，电子商务需要具有创新性地来维持收益能力。虽然保持较低的基础设施的成本是必需的，但是在不降低性能的条件下写出能够充分利用应用服务器的多种性能来满足全球客户需要的应用程序也是非常重要的。

应用程序开发人员必须部署能够缩短应用程序开发时间、实现可伸缩性、可用性和具有快速响应时间的技术从而具有竞争性。今天，简单地提供动态内容是不够的；Web 应用程序必须也支持个性化。应用程序开发人员必须满足如上的要求并且还要保证他们 Web 应用程序的设计是可扩展的。

对大多数的 Web 应用程序的开发人员而言，主要的挑战在于创建可伸缩的和快速的应用程序。从来没有客户抱怨过：“我的应用程序运行得太快了。”

为了满足可伸缩性和响应时间的需求，一个简单的解决方案是添加更多的硬件资源。然而，尤其是对于每秒钟要处理数千次访问的 Web 站点而言，这是一个昂贵的解决方案。限制应用程序为静态的页面也不是一个可行的选择。今天的 Web 应用程序被定义为是动态的并且在许多情况下是事务性的。通过提供静态和动态的内容作为整个 Web 页或从高速缓存的片段组装整个页这种创造性的方法来使用高速缓存有助于满足可伸缩性和响应时间的需求。

本文提供了关于 Oracle9i 应用程序开发人员能够部署在他们的应用程序中的不同的高速缓存解决方案和技术的概述。本文分为六节：

- 第一节提供了不同的高速缓存解决方案的概述
- 第二节说明了应用服务器功能的扩展以包括早期的例如内核中的高速缓存这样的特定领域的特性
- 第三节介绍了 JESI 的功能，包括概念和标记库
- 第四节清晰地说明了 Java 应用程序层高速缓存的必要性并且介绍了应用程序层高速缓存 API 和相关的标记库

- 第五节比较了 JESI 和应用程序层高速缓存技术并且说明了怎样在单个的 Web 应用程序中采用有效的方式将他们结合起来
- 第六节提供了部署可高速缓存的应用程序到 Oracle9i 应用服务器中的步骤。

高速缓存解决方案概述

长久以来，高速缓存已经被应用到包括操作系统和数据库在内的许多计算领域以提高性能。高速缓存是有望降低计算和经济成本的关键技术。当应用到 Web 应用程序时，高速缓存作为一项必要的技术以用来在更接近于浏览器的存储器中存储部分或整个 Web 页（静态和动态的）从而能够显著地解决 Web 站点访问速度慢的问题。几乎所有的应用程序都受益于将 Web 的内容高速缓存于在搜索内容的消费者和内容源本身（也称为源服务器）之间的结点上。

一个实际的高速缓存解决方案必须满足如下的需求：

- 提供保证动态内容刷新的功能
- 以持续高速的吞吐量来处理数以千计的并发用户
- 提供快速响应时间
- 支持本地和全球部署
- 因为不存在单一的解决方案能够满足所有的需求，所以要与其它高速缓存技术集成
- 后处理高速缓存内容
- 使用低成本的基础设施提供高收益

高速缓存解决方案可在不同层次上部署。每个解决方案都针对于特定的层次提供特定的性能。重要的是，必须注意响应时间是在用户的体系结构中访问不同层次时间的累加。应用程序开发人员和系统设计师必须在不同的层次上努力减少访问时间。最通常的情况下，一个完整的解决方案是一或多个高速缓存解决方案的组合。这些高速缓存解决方案包括：

- 浏览器高速缓存
- 代理高速缓存
- 内容传递网络服务
- 服务器加速器

浏览器高速缓存

高速缓存是市场上的所有 Web 浏览器都支持的特性。大部分的浏览器都能够将用户的硬盘目录中存储例如用户在 Web 上访问过的图像这样的静态对象。浏览器被配置成为实现这一目的分配一定数量的硬盘空间。浏览器高速缓存可加速包含被高速缓存对象的页面的显示。

这一方法是有局限性的，因为他只有助于加速传递静态页元素。同时，它的有效性也是有限的，因为内容提供者经常在 HTTP 响应头中使用 **nocache Pragma** 来禁止对即使是静态生成的内容进行高速缓存。内容提供者这样做是为了保持对内容的控制，尤其是在内容经常变化的情况下。

此外，一些小型 Internet 设备可能没有足够的本地存储空间以支持存储有效数量的高速缓存内容。

代理高速缓存

同为客户机上进行存储的浏览器高速缓存相比，*代理高速缓存*是一个基于服务器的解决方案。代理高速缓存服务器部署在大量的浏览器客户—例如拨号 ISP 用户或企业 Intranet 用户—和公共 Internet 之间。当用户试图访问 URL 时，客户浏览器发送 HTTP 请求到代理服务器。代理服务器为请求的对象检查它的本地高速缓存。如果“失败”（不可得到）的话，代理服务器从源 Web 服务器请求对象，将对象存储到高速缓存中并传送到客户。如果“找到”的话，也就是说发现对象在将来才会过期，那么代理服务器发送带有 If-Modified-Since 头的 GET 请求到源 Web 服务器。于是源 Web 服务器返回一个 Last-Modified 响应头。代理服务器对时间戳进行比较：如果是新的，被高速缓存的对象直接发送到客户；如果是旧的，就从源 Web 服务器请求对象并传送到客户。

代理服务器的主要目的是减少带宽的支出，尽管在某些情况下响应时间也缩短了。通过代理在本地为更多的请求提供服务，就有相对较少的请求需要通过长时间的、昂贵的网络传输到源 Web 服务器。代理高速缓存在企业的 Intranet 中也是有效的，它们作为保护 intranet 服务器免受攻击的防火墙。

来自 Internet 的某些情况下代理配置的缺点在于每个浏览器必须显式地配置来使用它。同时，代理高速缓存自身也能成为系统发生故障的一个环节。

内容传递网络服务

内容传递网络服务 (CDN) 设计为利用终端用户的地理位置来工作。第一代 CDN 主要支持静态内容。当客户浏览器从源 Web 服务器请求内容时，服务器返回带有嵌入的引用 CDN 服务域的 HREF 链接。于是客户浏览器从 CDN 中请求引用的对象。CDN 使用分布式 DNS 机制和当前的 Internet 上的通信状况来选择最优的高速缓存服务器对请求作出响应。

更新的 CDN 服务,例如 Akamai EdgeSuite², 提供了传递网上动态生成内容的能力。内容提供者必须为他们的 Web 站点创建一个 DNS 别名作为由 CDN 服务提供者管理的主机名。浏览器对动态生成的 HTML 和嵌入的静态内容的请求现在完全通过 CDN 所管理的边缘服务器网络来传输。Edge Side Includes (ESI) 功能使得 CDNs 能够聚合各部分的动态 Web 页面并且为单独的用户快速地重新组装这些页面。ESI 是由 Oracle 公司和 Akamai Technologies 联合开发的,它也被提议作为 W3C 和 JCP (JSR #128) 中的一个开放式标准。

关于规范的更多信息和其他细节请参见 <http://www.esi.org/CDN> 服务非常适合于全球分布的、基于 Internet 的用户群。然而,当 Web 站点存在频繁的和大量的内容更新时,CDN 也存在着一些不足。此外,这些服务非常昂贵,这也使得它们不能够被许多电子商务所采纳³。

服务器加速器

服务器加速器是一种高速缓存,它代表着一或多个特定的 Web 服务器而不是代表着一组浏览器用户。一个服务器加速器高速缓存(或“相反的代理”高速缓存)截取所有到 Web 服务器的请求,高速缓存所提供对象的拷贝,而后当它下一次接收到对它们的请求时,再提供这些对象。随着服务器加速器高速缓存的增加,他自身能够为更多的请求服务,这也为应用服务器和数据库中的其它任务释放出了更多的处理资源。服务器加速器也有助于降低成本,它摆脱了较昂贵的后端服务器,只是在廉价的 PC 平台上来实现。

在接下来的几节中,我们会探讨不同的高速缓存解决方案以满足前面所列的需求。

应用服务器的演变

随着应用服务器的发展,用户期待着早期的例如门户、商务智能、无线能力、安全等这样的特定领域特性能成为应用服务器的集成部分。今天,带有开发工具的高速缓存有望成为应用服务器的固有的部分。Oracle9iAS 满足了这些需求并且包括一个高性能的 HTTP 层高速缓存—Oracle9iAS Web 高速缓存,它能够高速缓存静态和动态的数据。

Oracle9iAS Web Cache

Oracle9iAS Web Cache 是一个 *可识别内容* 的服务器加速器，它在普通硬件平台上提供了每秒钟几千次请求的吞吐速率。

Oracle9iAS Web 通过在内存中存储经常访问的页来减轻繁忙的 Web 服务器的负载，这样也不必在中间层应用服务器和数据库中对这些页的请求重复进行处理。由于 Oracle9iAS Web 高速缓存是基础设施的一部分，所以高速缓存成为内容的传递工具，这也使应用服务器和数据库解脱出来能够处理更新和生成新的内容。通过将内容的传递和生成相分离，Oracle9iAS Web 高速缓存提供了一种廉价的方式来实现可伸缩性和高性能。

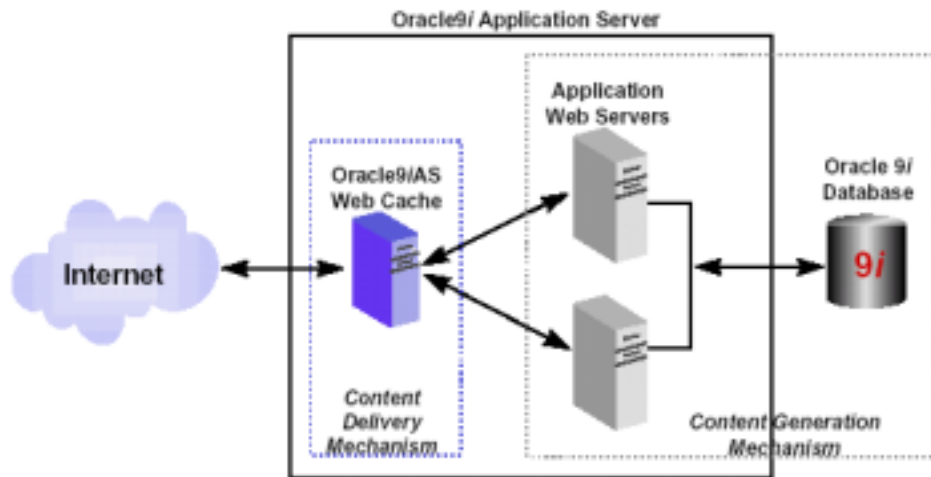


图1. 使用 Oracle9iAS Containers for J2EE 和 Oracle9iAS Web Cache - 分离内容的生成和内容的传递

Oracle9iAS Web Cache 位于例如 Oracle9iAS Containers for J2EE (OC4J) 这样的应用 Web 服务器之前，它高速缓存它们的内容并且为请求内容的 Web 浏览器提供这些内容。Oracle9iAS Web Cache 能够识别 HTTP 头并且能够在用户定义的可高速缓存性原则的基础上能够作出高速缓存和路由的决定。当 Web 浏览器访问 Web 站点时，它们发送 HTTP 协议或 HTTPS 协议请求到 Oracle9iAS Web 高速缓存，它也反过来作为应用服务器的一个虚拟服务器。如果请求的内容已过期或是无效的或是不能够再访问，Oracle9iAS Web 高速缓存会从应用服务器中检索新的内容。Oracle9iAS Web 高速缓存不仅能够处理大容量的频繁的数据更新，而且它能够在沉重负载的情况下保持同数据源的一致性。

Oracle9iAS Web 高速缓存包括一个支持在应用 Web 服务器响应中使用的 Edge 端包括标记语言的 ESI 处理器。ESI 是一种简单的标记语言，应用程序开发人员使用它可为在网络上的动态高速缓存和页面组装进行内容片段的识别。

EDGE SIDE INCLUDES FOR JAVA (JESI)

除了 ESI 之外,Oracle公司和 Akamai Technologies 也为使用 Java 和 J2EE API 的 Web 应用程序开发人员引入了 Edge Side Includes for Java (JESI)。Oracle9iAS Containers for J2EE (OC4J)提供了 JESI 标记作为与 ESI 标记和 Edge Side Includes 的Web高速缓存功能的一个方便的接口。尽管 ESI 标记能够在任何 Web 应用程序中直接使用, JESI 标记在 JavaServer Pages环境中也提供了额外的便利的使用方法。JESI 的一些好处包括:

- 标准的 JSP 框架
用户可通过页相关的或应用程序相关的语法代替完整的 URL 或文件路径来引用包含的页。—一个方便的特性
- JESI 快捷语法
JESI 标记支持方便的语法和标记属性来指定元数据信息(例如高速缓存页的截止日期)、适当地显式使页面失效以及为个性化页面使用 cookie 信息。
- 应用程序层配置文件
JESI 标记库能够使用应用程序层配置文件来指定适合于特定用户环境的部署时间参数和缺省设置。

JESI 标记库

Oracle 实现的 JESI 标记库是一个 *标准的*、在标准的 ESI 框架之上的 JavaServer Pages 标记库。JESI 标记可同标准的 JSP 1.1 或更高版本的容器以及同 ESI 1.0 或更高版本的规范相兼容的可 ESI 的高速缓存环境一起使用。关于 JESI 和 ESI的更多信息,请参见: <http://www.esi.org/>。

下表总结了不同的 JESI 标记:

标 记	目 的
<jesi:include>	在“模板”页中使用以指示 ESI 处理器怎样组装片段(标记生成<esi:include> 标记。
<jesi:control>	为模板和片段指定属性(例如,截止日期)
<jesi:template>	用于在它的正文中包含整个 JSP 容器页的内容。
<jesi:fragment>	在 JSP 页中封装单独的内容片段。
<jesi:invalidate>	显式地删除和/或使 ESI 处理器中被选择的高速缓存对象过期。
<jesi:personalize>	插入个性化的内容到一个页面中,在此页面中内容是放置在 cookies 中的并由 ESI 处理器插入到页中。

使用 JESI 标记库

JESI 标记允许将 JSP 页中的动态元素分解为可高速缓存的组件或片段。通常一个 Web 页是由多个片段或部分页组成的。例如,一个 Web 页可包括每 4 小时更新的本地天气、每 15 分钟更新的股票行情收录机、每 30 秒钟更新的新闻标题和每周更新的促销信息。每一条信息都是一个拥有例如生存周期这样的自身属性的可高速缓存的组件。

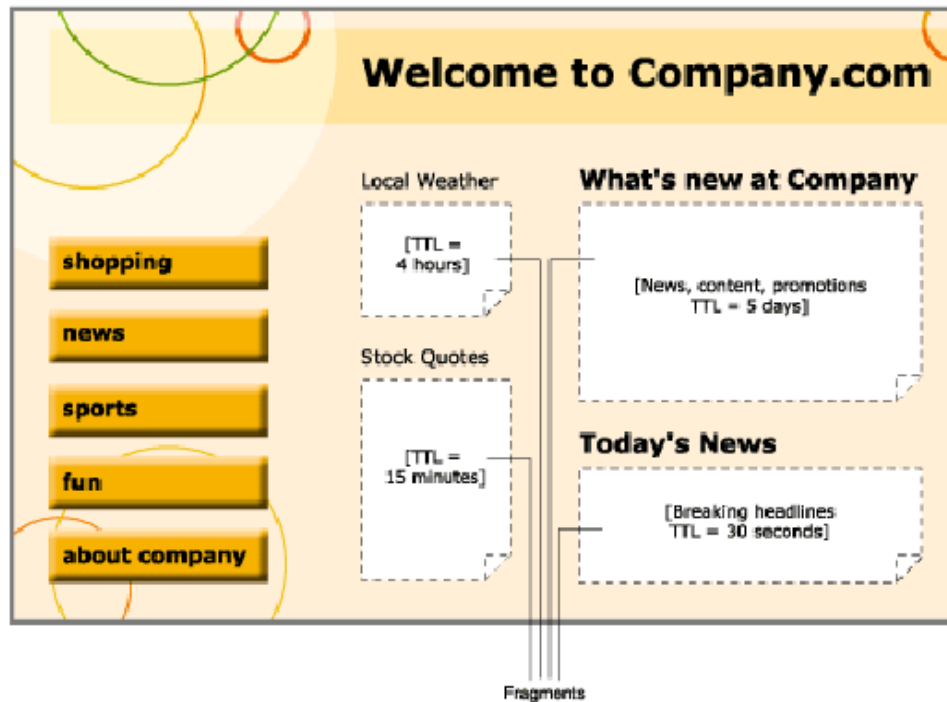


图 2: 一个典型的 Web 页布局是由许多片段组成, 每个片段都有自身的生存周期

有两种使用 JESI 标记来定义 JSP 页的设置和可高速缓存内容的方法:

- 控制/包含模型
- 模板/片段模型

控制/包含模型

控制/包含模型是一个组合的方法, 由此方法可高速缓存的内容使用 `jesi:include` 标记可包含进顶层的页中。对于每一个包含的页的内容都在 `jesi:include` 语句处合并到顶层页的 HTTP 输出中。顶层页和包含页都可选地带有一个 `jesi:control` 标记以用来设置例如截止时间这样的高速缓存参数以防缺省设置不合适。

下面是使用控制/包含 JESI 标记的多种方法之一。在例 1 中, `jesi:control` 标记用来为 `maxRemovalDelay` 和 `expiration` 指定非缺省的高速缓存设置。

- `maxRemovalDelay` — 指定了 ESI 处理器在被高速缓存对象过期后还能够存储它的以秒为单位的最大时间。缺省值为 0。
- `expiration` — 指定了以秒为单位的高速缓存对象的生存时间。缺省值是页永远不过期。

万一 order.jsp 不能够被检索, jesi:include 标记指定了一个供替换的页 alt.jsp。此外还要注意设置 ignoreError="true" 允许 ESI 处理器即使在 order.jsp 和 alt.jsp 都不能被检索时也能继续进行处理。第二个 jesi:include 既没有设置供替换的页也没有设置 ignoreError 为 true, 一旦 commit.jsp 不能够被检索, 处理就会停止。

例 1 — 控制/包含:

```
<%@ taglib uri="/WEB-INF/jesitaglib.tld" prefix="jesi" %>
<jesi:control maxRemovalDelay="1000" expiration="300" cache="yes"/>
<jesi:include page="order.jsp" alt="alt.jsp" ignoreError="true"/>
<jesi:include page="commit.jsp" />
```

模板/片段模型

在模板/片段模型中, 一个合成的 Web 页的内容被分割成相互分离的可高速缓存的片段。jesi:template 片段用来封装合计的所有的在片段内和片段外的可高速缓存的内容。The jesi:fragment 标记用来在合计的页内定义片段。每个片段都是一个带有自身高速缓存参数的单独的可高速缓存的实体。jesi:template 用来设置在片段外的内容的高速缓存参数。如果 jesi:template 或 jesi:fragment 的高速缓存参数没有设置, 就会使用缺省值。为了使用 JESI 模板/片段是推荐的用来转换已有的 JSP 页的方法。i

例 2 — 模板/片段模型

```
<%@ taglib uri="/WEB-INF/jesitaglib.tld" prefix="jesi" %>
<jesi:template expiration="3600">
...HTML block #1...
<jesi:fragment expiration="60">
...JSP code block #1...
</jesi:fragment>
...HTML block #2...
<jesi:fragment>
...JSP code block #2...
</jesi:fragment>
...HTML block #3...
<jesi:fragment expiration="600">
...JSP code block #3...
</jesi:fragment>
...HTML block #4...
</jesi:template>
```

例 2 说明了模板/片段方法。每一个 JSP 块都被高速缓存为由周围的 jesi:fragment 标记所划分的单独的对象。所有的顶层 HTML 块 (#1 - #4) 一起构成了一个单独的对象, 它按照 jesi:template 中设置的参数来进行高速缓存。注意, 模板和大部分的片段设置了高速缓存的截止时间, 其他的属性取缺省值。第二个片段包括截止时间在内的所有属性都取缺省值。

其他的 JESI 功能

有些情况下，必须显式地使高速缓存的内容失效。货物需求或供应的变化都要求显式地使高速缓存的内容失效。此外，应用程序中 JSP 页的执行也可能使与另一页相应的高速缓存的对象失效。数据库中底层数据的变化也会引起失效。

`jesi:invalidate` 标记以及它的子标记允许用户使页失效。失效也可能由于 URI 或 URI 的前缀而发生。此外，用户能够选择性地提供 cookie 名字-值对或 HTTP/1.1 请求头名字-值对。

JESI 标记也支持有限的个性化。The `jesi:personalize` 标记使用 cookie 信息来裁剪到每一个单独用户的输出。这一标记指示 ESI 处理器在保持高速缓存页的共享性的同时使用来自于发送进来的 cookies 中的动态字符串来取代高速缓存页中的位置持有者。`jesi:personalize` 标记的一个例子：

```
<jesi:personalize name="user_id" value="guest" />
```

在此 ESI 处理器查找名字为 `user_id` 的 cookie 并检索它的值。如果找不到 cookie，它就使用缺省值 `guest`。Oracle9iAS 发行版 2 也包括提供了更加丰富和高级特性的个性化标记。（更详细的情况参见《*Oracle9iAS Containers for J2EE JSP 标记库和工具参考书*》）

OC4J 中的 JSP 容器基本上打包在两个 jar 文件中，名字为 `ojjsp.jar` 和 `ojsputil.jar`。发行版也包括适用于不同标记库的 tld 文件，它包括 `jesitaglib.tld` 和一些演示程序。

ESI 也可以通过生成必要的作为输出页一部分的 ESI 标记从而能够在 servlet 中使用。

存在多种 Web 应用程序开发人员可部署的组合和属性的设置。本文提供了几个说明性的例子。有关所有的属性和使用情况的信息，请参见《*JavaServer Pages OracleJSP 支持开发人员指南和参考*》和《*J2EE JSP Oracle9iAS 容器标记库和工具参考书*》。

应用程序层高速缓存

至此我们已经了解了用户如何使用高速的 HTTP 层的 Oracle9iAS Web 高速缓存以及使用 JESI 标记的可 ESI 的 CDNs。然而，Web 应用程序开发人员会遇到这样的情况，应用程序对象不是 HTML 或 XML 片段，而是例如 XML DOM 对象或者可串行化的 Java 对象。此外，可能会有这样的需求，要求能够重用或后处理被高速缓存的内容，或者保留中间结果。Oracle9iAS 发行版 2 包含了如下的内容来完成应用程序层高速缓存：

- Java 对象高速缓存 - 低层对象高速缓存 API，为 Java 对象提供高速缓存服务。
- Web 对象高速缓存 - 使用 Java 对象高速缓存作为库来创建 Web 应用程序级高速缓存设备。

Java 对象高速缓存

Java 对象高速缓存是一种底层对象高速缓存 API，支持普通对象类型，例如存储器对象，磁盘对象，存储池对象和流访问对象。它使用分布式对象管理来协调 Java 对象的更新和失效。关于 Java 对象高速缓存的详细信息，请参见：
《Oracle9i 应用服务器 Java 高速缓存开发人员指南》发行版 2 (v9.0.2)。

当应用服务器使用 Java 程序来提供内容时，Java 对象高速缓存可以为昂贵的且频繁使用的 Java 对象提供高速缓存。被高速缓存的 Java 对象可能包含产生的页面，也可能在程序里提供支持对象来帮助创建新的对象。Java 对象高速缓存能够自动加载并更新 Java 应用程序所指定的对象，还包含管理被高速缓存的 Java 对象的 API。尽管它在很多方面都灵活并且强大，但它不提供例如定制的 JSP 标记这样更高层的抽象并且缺少同 HTTP 环境的集成。Java 对象高速缓存的一般特性使得它对于更高级的高速缓存，例如 Web 对象高速缓存（下面要讨论的），成为一种理想的高速缓存库，可以减轻开发工作并且降低复杂度。

Web 对象高速缓存

Web 对象高速缓存通过向应用程序开发人员提供精细的控制来编程实现高速缓存，或使用 API 调用（对于 servlet），或定制的常规标记库（对于 JSP）来推动 WEB 应用程序级的高速缓存。Web 对象高速缓存允许用 Java 所写的 Web 应用程序捕获、存储、重用、后处理和保留由动态 Web 页产生的局部和中间结果。Web 对象高速缓存工作在 Java 层并且与 JSP 和 servlet 应用程序的 HTTP 环境紧密集成。被高速缓存的对象可能由 HTML 或 XML 片段、XML DOM 对象或者可串行化的 Java 对象组成。

应用程序能够动态的在被高速缓存的 XML DOM 对象上使用 XSLT 样式表为不同的用户群或设备生成不同的页面。被高速缓存的对象还能够使用 SMTP 通过 e-mail 发送给多个用户，对于电子商务应用程序来说它是一项常见的任务。

Web 对象高速缓存在下列情况下最有效:

- 对被高速缓存数据对象进行的后处理, 例如 XSLT 或 XML DOM 操作
- 非 HTTP 环境中的数据共享, 例如重用被高速缓存的 XML 数据或 Java 对象, 并通过 SMTP、JMS、AQ 或 SOAP 将这些数据发往别处。
- 特殊的存储需要, 例如把被高速缓存的数据存储在一个文件系统或是数据库中, 以持久稳固的存储长生命周期的数据
- 应用程序相关的权限和安全机制, 允许不同的用户对于不同的数据项有不同的访问权限, 例如基于 Web 的组件应用程序

Web 对象高速缓存由两部分组成:

i) 高速缓存库

ii) 编程界面高速缓存容器是一种负责数据存储、数据发布和高速缓存失效的组件。

由于 Java 对象高速缓存是一种通用的、灵活的高速缓存服务, 并有设计好的 API 可供 Java 应用程序使用, 所以 Web 对象高速缓存使用 Java 对象高速缓存作为它的默认缓存库。尽管并不推荐, 但也有其他的实现方案, 例如一个简单的文件系统。

Web 对象高速缓存编程界面包括 Web 对象高速缓存 JSP 标记库和 Web 对象高速缓存 API。本文我们讨论的范围只限于 Web 对象高速缓存 JSP 标记库。关于 Web 对象高速缓存 API 的信息, 请参见: 《*JavaServer Pages OracleJSP 支持开发人员指南和参考*》

Web 对象高速缓存标记库

下面的表格总结了 Web 对象高速缓存标记。

标 记	目 的
<ojsp:cache>	这个面向于一般的高速缓存 (对比于高速缓存)
<ojsp:cacheXMLObj>	用于高速缓存 XML 对象
<ojsp:cacheXMLObj>	高速缓存并后处理 XML 文档
<ojsp:useCacheObj>	用于高速缓存任意的 Java 可连载对象
<ojsp:cacheInclude>	综合上述高速缓存标记和标准 JSP 包含标记的功能
<ojsp:invalidatecache>	通过编程逻辑显式地使一个高速缓存块失效

尽管 Web 对象高速缓存能够高速缓存普通的 HTML 和 XML 片段，但它更适宜于高速缓存 XML DOM 或者 Java 可串行化对象。

`ojsp:cacheXMLObj` 标记能够高速缓存 XML 对象。大多数的属性可以被设置以获得期望的行为。这些属性包括：**scope**、**TTL(time-to-live)**、**writeThrough**、**fromXMLObjname**、**toXMLObjName**，等。

标记 `ojsp:useCacheObj` 用来高速缓存任意 Java 可串行化对象。

标记 `ojsp:useCacheObj` 还包括一个丰富的属性集用来控制被高速缓存的 Java 对象的行为。下面的这个例子说明了如何使用 `ojsp:useCacheObj`。

例 3 : `ojsp:useCacheObj`

```
<ojsp:useCacheObj id="a2" policy="/WEB-INF/test-policy.cpd"
type="examples.RStrArray" >
<%
// create a temp writeable array
WStrArray tmpa2=new WStrArray(3);
tmpa2.setStr(2,request.getParameter("testing4"));
tmpa2.setStr(1,"def");
tmpa2.setStr(0,(new java.util.Date()).toString());
// create a readonly copy for the cache
a2=new RStrArray(tmpa2);
// storing the a2 into pagecontext
// so useCacheObj tag can pick it up
pageContext.setAttribute("a2",a2);
%>
</ojsp:useCacheObj>
```

位于 OC4J 中的 JSP 容器预先配置为使用 Java 对象高速缓存作为库。Web 对象高速缓存标记的功能打包为 `ojsputil.jar` 和 `jwcache.tld` 的一部分，OC4J/Oracle9iAS 发行版 2 的发布也包含了这两种类型的高速缓存演示程序。

Oracle9iAS Web cache vs. Java-level Web Object Cache

最初，Oracle9iAS Web 高速缓存和 Web 对象高速缓存在某些方面看起来很相似。但是，他们为获取最大效益而所使用的方法却迥然不同。

Oracle9iAS Web Cache 是作为“前线/frontline”高速缓存来使用的。它直接为 HTTP 客户提供高速缓存页。Oracle9iAS Web 高速缓存能够以快速的响应时间来处理大量的 HTTP 流量—满足了大多数 Web 站点的要求—并且可以存储全部或部分的动态 Web 页。被高速缓存的 Web 页在送往客户前，还能够进行一定程度的定制。例如，定制包括 cookie 替换和页片段连接的定制。推荐 Oracle9iAS Web 高速缓存作为用户主要的 Web 高速缓存机制来使用。Oracle9iAS Web 高速缓存不仅能够显著地缩短响应时间，它还能够减轻应用服务器和后端数据库的负荷。此外，Oracle9iAS Web Cache 运行于经济的硬件上仍具有很好的性价比。

Web 对象高速缓存并不打算用作主要的“前端”高速缓存。它是嵌入在 Java Web 应用程序中并由其来维护的应用程序层的高速缓存。如前所述, Java Web 应用程序开发人员可以使用 Web 对象高速缓存来捕捉 JSP 和 servlet 执行产生的中间结果, 并随后在 Java 应用程序逻辑的其他部分重用它们。对于 Web 对象高速缓存中被高速缓存的结果, 检索路径包括 JVM 和 JSP/Servlet 容器, 同 Oracle9iAS Web 高速缓存相比, 通常花费更长的时间来提供一个页。Web 对象高速缓存是一种辅助的高速缓存, 仅仅有益于在向客户提供页之前, 对被高速缓存的内容进行大量后处理的 Web 应用程序, Web 对象高速缓存可用的情况在前面的章节已有论述。对于其他的情况, 推荐使用 Oracle9iAS Web 高速缓存, 因为它可以使速度能以数量级的飞越提高。

尽管 Oracle9iAS Web 高速缓存和 Web 对象高速缓存的目的迥然不同, 但在某些情况下他们却以一种有趣的方式彼此互补。某些情况下用户想对被高速缓存的内容作后处理并想使用 Oracle9iAS Web 高速缓存, 此时就可同时使用 JESI 和 Web 对象高速缓存标记,

例 4 : 使用 JESI 标记、Web 对象高速缓存标记和其他的定制标记

```
<jesi:template cache="no">
<% String userStyleLoc="style/rowset.xml"; %>
<h3>Transform DBQuery Tag Example</h3>
<h4>Current Time=<%= new java.util.Date() %></h4>
<jesi:fragment expiration="60">
<!-- You can cache HTML in Oracle Web Cache with JESI
or you can cache it in Oracle Web Object Cache -->
<h4>Cached Time=<%= new java.util.Date() %></h4>
<sql:dbOpen connId="conn1" URL="<%= connStr %>"
user="scott" password="tiger" />
<jml:transform href="<%= userStyleLoc %>" >
<!-- The XML DOM object is produced by dbQuery
And, the DOM object is cached in OracleJSP Web Object Cache.
XSLT is performed on the cached object. -->
<ojsp:cacheXMLObj TTL="60" toWriter="false">
<sql:dbQuery connId="conn1" output="xml" queryId="myquery" >
select ENAME, EMPNO from EMP
</sql:dbQuery>
</ojsp:cacheXMLObj>
</jml:transform>
<sql:dbCloseQuery queryId="myquery" />
<sql:dbClose connId="con1" />
</jesi:fragment>
</jesi:template>
```

一个昂贵的数据库查询的结果可以被高速缓存为 Web 对象高速缓存中的一个片段, 通过 Oracle9iAS Web 高速缓存还能对其进行后处理 (例如使用样式表) 和发送。例 4 说明 Web 对象高速缓存标记和 JESI 标记的结合。此外, 在保持 JSP

程序简单性的同时，还应注意其他的标记，例如 SQL 和转换标记，是怎样被应用于一个简单的应用程序，以发挥定制标记的作用。

例 4 说明各种标签如何有效地在一起使用。这里我们使用 `sql:dbquery` 标记来执行一个简单的 SQL 语句。输出属性设置为“XML”。使用 `ojsp:cacheXMLObj` 将查询结果作为一个 XML DOM 对象存储在 Web 对象高速缓存中。使用 `jml:transform` 标记，在高速缓存对象上应用 XSLT 样式表。使用 JESI **模板/片段** 标记来高速缓存被转换的 XML DOM 对象和一些 HTML，并通过 Oracle9iAS Web 高速缓存提供。注意在 `jesi:fragment` 标记和 `jesi:template` 标记中的 HTML 内容。

部署可高速缓存的 WEB 应用程序

在 Oracle9iAS 上部署可高速缓存的 Web 应用程序的步骤同常规的 Web 应用程序是相同的。在准备部署前，首先要启动 Oracle9iAS Web 高速缓存。同时可方便地使用称为 Web Cache Manager 的 GUI 工具来管理和配置 Oracle9iAS Web 高速缓存。此外，用户可查询 *Oracle9iAS Web 高速缓存管理与开发指南* 获得详细的信息。

一旦用户安装了 Oracle9iAS Web 高速缓存并且选择了用户想要运行的配置，用户就能使用 JDeveloper9i 的内建工具部署可高速缓存的 Web 应用程序，或按照如下说明的步骤手工进行部署：

步骤 1：将应用程序打包为 EAR 文件。EAR 遵循一个 J2EE 应用程序必须遵守的标准结构。目录结构如图 4 所示：

```
<appname>
|-----META-INF/
| \-----application.xml
|
|-----EJB JAR file
|-----WEB WAR file
\-----Client JAR file
```

图 4. EAR 文件结构

```

myapp
|___web
|___WEB-INF
|   |___web.xml
|   |___orion-web.xml
|   |___classes
|   |___myServlet.class
|___lib
|___libs, jars
|___<taglibs>.tld
|___add.jsp
|___delete.jsp
|___edit.jsp
|___index.html

```

图 5. WAR 文件结构

此 application.xml 作为说明文件提供了应用程序名、组成应用程序的不同模块和 Web 应用程序的上下文根等更多的信息。每一个 EAR 组件，例如 JAR 文件和 WAR 文件，都遵循一个标准结构。Web application 打包到 WAR 文件中。确保用户已经包含了应用程序所需的 JSP、servlet、例如 ojsputil.jar 的 jar、，例如 jesitaglib.tld 的 TLD 文件和其他的相关文件。The WAR 文件结构如图 5 所示。

为了部署 Web 应用程序，用户必须创建 EAR 文件。如果只使用 servlet 和 JSP，EAR 文件的其他组件可为空。

步骤 2:使用 admin.jar 工具部署打包的 EAR 文件。

示例:

```

java -jar $J2EE_HOME/admin.jar ormi://localhost admin
<passwd> -deploy -file ./app/HRapp.ear -
deploymentName HRweb-app

```

HRapp.ear 使用用户定义的Hrweb-app 来部署。这在 server.xml中生成一个新的条目。

```

<application name= "HRweb-app"
path= "../applications/HRapp.ear" /> ]

```

步骤 3:绑定 Web 应用程序

示例:

```

java -jar $J2EE_HOME/admin.jar ormi://localhost admin
<passwd> -bindWebApp HRweb-app emp-web default-website
/employees

```

Hrweb-app Web 应用程序绑定到特定的Web 站点（在本例中为缺省的 Web 站点）和上下文根（/employees）。这在 default-web-app.xml 中生成了一个新的条目。

```
<web-app application=“HRweb-app” name=“emp-web” root=“/employees” ]
```

用户的应用程序已准备好访问 <http://<host>:<port>/employees>

总结

总之，由于 Web 站点的需求变得复杂并且 Web 的内容也大部分变成动态的，所以需要使用结合不同的高速缓存解决方案的创新性的技术。重要的是应当有一个能够优化性能的高速缓存解决方案。同等重要的是创建能够充分利用所采用的解决方案的 Web 应用程序。使用 JESI 标记并使企业和 Internet 的 CDNs 同例如 Oracle9iAS Web 高速缓存这样的服务器加速器相结合，就可在网络上完成动态 Web 内容的即时组装和发送。ESI 和 JESI 都是与标准兼容的技术。Web 应用程序开发人员在某些情况下也能使用 Java 应用程序层的高速缓存。JESI 和应用程序层高速缓存可以以有趣的方式一起使用来最大限度地提高性能。

参考文献

¹ 1999 Zona 研究报告，题为 *The Need for Speed*. 报告在
<http://www.zonaresearch.com>

² Akamai EdgeSuite 参见
http://www.akamai.com/html/en/sv/content_delivery.html

³ 典型的 CDN 提供者每秒传送的每兆位内容的成本从 1400 美元到 2000 美元：
<http://www.nwfusion.com/news/2001/0223peer.html> 内容在
<http://www.caching.com>

Oracle9iAS Web 高速缓存文档和相关内容在
http://otn.oracle.com/products/ias/web_cache/

OracleJSP Support for JavaServer Pages 开发人员指南和参考书
<http://otn.oracle.com/>

Oracle9iAS Containers for J2EE JSP 标记库和工具参考书
<http://otn.oracle.com/>



Oracle9iAS 高速缓存解决方案
2001年12月
作者: Prasad Shiva

Oracle 公司
全球总部
500 Oracle Parkway
Redwood Shores, CA 94065
U. S. A.

全球咨询热线:
电话: +1. 650. 506. 7000
传真: +1. 650. 506. 7200
www.oracle.com

Oracle Corporation provides the software that powers the internet.

Oracle 是 Oracle 公司的注册商标。此处引用的各种产品和服务名称可能是 Oracle 公司的商标。提及的所有其他产品和服务名称可能是它们相应所有者的商标。

版权所有 © 2000-2001 Oracle 公司
保留所有权利。