2013 年第 2 期（总第 17 期）

数字图书馆标准规范
跟踪扫描

主办单位：中国科学院国家科学图书馆

2013 年 3 月

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

为传播科学知识，促进业界交流，特编译《标准规范跟踪扫描》，仅供个人学习、研究使用。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

目　录

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

【标准规范报道】

1、NISO 推出开放获取元数据和指标开发标准的新举措

2013 年 2 月 7 日——巴尔的摩，马里兰州——美国国家信息标准协会（NISO）有表决权的成员已核准了一项开发标准化书目元数据和可视指标来描述期刊论文的可访问性及潜在描述条目如何“开放”的新项目。许多产品可以以开放获取（OA）、增加访问、公共存取或其他描述的名义从出版商处获取；在某些情况下，出版商基于作者的资助单位提供不同的条款。许多出版商也提供混合选项，其中一些文章是“开放的”，而期刊其他内容仅能通过订阅或许可来获取。目前没有提供某一特定文章是否可随意阅读以及读者可获得什么重用权的信息的标准化书目元数据。出版商甚至来自同一出版商的期刊间，指示一篇文章的开放性的可视指标或图标的设计和使用都是不一致的。

“NISO OA 元数据和指标项目将补充当前正在进行中的其他相关工作，”Nettie Lagace（NISO 项目副主任）说道。“这样的项目包括 CrossMark，CrossRef 的更新识别服务；它如何开放？（How Open Is It?），由公共科学图书馆（PLoS）、学术出版与学术资源联盟（SPARC）以及开放获取学术出版商联合会（OASPA）开发的指南；OA 词汇（V4OA），JISC/UKOLN 项目；ONIX-PL，EDItEUR 开发的沟通许可条款的规范；链接内容联盟；NISO 的开放发现计划。与这些项目协调和沟通将是 NISO 工作组努力的一个重要方面。”

“拥有标准化 OA 元数据和指标的益处应该对学术交流链中的许多参与者有积极的影响 ” Todd Carpenter（NISO 执行董事）解释道。“实施 OA 授权的资助者将有一种机制来决定某一特定文章或研究者是否符合其政策。混合期刊的出版商将得益于拥有一个标识在该模型下出版的文章的 OA 状态的简单机制。作者可以更容易地判断他们所选择的分配选项是否被尊重以及是否能够按照资助者要求进行记录。读者可以更加容易地从检索结果中探知他们是否能够免费或付费阅读一篇文章，并且更易遵守出版商已制定的条款。汇集者和发现服务提供者将拥有一种改进的机制，以编程的方式收集和揭示社区中可用的 OA 文章。”

NISO 所发起的该项目将最初聚焦于描述与某一 OA 文章相关联的读者权限的元数据元素。具体来说，NISO 工作组将决定用于描述和传递权限的最佳机制，拥有该权限的任意用户可访问来自任意互联网连接点的特定文章。推荐将包括一种以机器可读格式分配和聚集此元数据的手段。该组还将考虑整合重用权限信息的可行性以及就那种数据传输达成一致意见的可能性。

(编译自：NISO Launches New Initiative to Develop Standard for Open Access Meta-data and Indicators.
http://www.niso.org/news/pr/view?item_key=d2e5f409bc6af6b7f504a10edf0329203ffec

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

6f9. [2013-02-07])

（岳增慧编译，吴贝贝校对）

2、电子书：厚望网络标准

电子书与开放网络平台的 W3C 研讨会

2013 年 2 月 11-12 日，纽约，美国

W3C 与 IDPF（国际数字出版论坛）、BISG（书业研究集团）在美国纽约于 2013 年 2 月 11、12 日一起举行了电子书与开放网络平台的研讨会，名为“电子书：厚望网络标准”。

该研讨会的技术讨论集中于目前用于电子书的开放网络平台技术以及这些技术用于未来数字出版物所需的改进。参与者在会议上及休息时段讨论了如何拉近出版业与网络技术的发展以保证顺畅的合作等包罗万象的问题。

执行摘要

今天的电子书市场是动态的、快速变化的和强大的。电子书与印刷版本竞争，电子书读者可使用许多硬件和软件。然而，出版商在该市场中面临着重大的商业和技术挑战，其中一些可以通过标准化进行减少或删除。

国际数字出版论坛（IDPF）已定义了 EPUB 标准（最新版本 3.0），该标准主要建立在 W3C 开放网络平台技术之上。OWP 也越来越多地被用于独立桌面和移动应用程序的核心。然而，可以并且应该采取更多措施来解决出版业的具体问题和要求，应用于电子书的网络标准、网站的发布或内容应用。

W3C 寻求支持数字出版环境下网络技术的广泛使用。因此，网络和出版团体有必要围绕定义良好的技术问题加强合作。该研讨会是第一步，将利益相关者聚集到一起，共享他们自己的观点、要求和思想，以确保新兴的全球技术标准符合数字出版业的需求。研讨会已确认了 W3C 在未来几年中能够且应该一起研究解决的许多技术问题 。

研讨会的参与者按照主题列表的顺序开始讨论，如描述、布局、字体或可访问性。作为下一步，W3C 工作人员将与利益相关者（如 IDPF 和 BISG）在数字出版生态系统进行协作，以识别与出版标准相关工作的机会。

提交的意见书有 43 份，注册参与者有 89 个。在一天半的研讨会中，有 5 个会议，24 个演讲。

征稿主题为：

应用：

- 标准化问题，包括当前和未来 W3C 标准（如 HTML、SVG、MathML、Web API-s、元数据等）的关系。
- 布局定义与控制（固定和自适应布局、高质量的排版、字体定义和管理等）

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

- 可访问性（编写可访问性指导方针，包括图形编辑、后备）
- 语音控制

描述：

- 色彩管理和转换
- 设备描述
- 小工具定义、标准化
- 一致性（定义、要求、测试方法、认证）
- 功能分化

管理：

- DRM 管理(包括可互操作的、开放的 DRM 系统、社会 DRM 等)
- 电子书唯一识别
- 外联和部署
- 包装
- 元数据存储和词汇表

研讨会的参与者达成了一个广泛共识,开放网络平台中的技术为电子书提供了一个引人注目的基础，但需要进一步的工作。在描述问题、可访问性、测试及其他议题的工作上都有强大的支持。总结会议概要中给出了完整的列表。

(编译自：eBooks: Great Expectations for Web Standards. http://www.w3.org/2012/08/electronic-books/rapportebook.html. [2013-03-04])

（岳增慧编译，吴贝贝校对）

3、标准日——关于欧洲标准化的开放信息日

时间：2013 年 4 月 23—24 日

地点：比利时布鲁吉亚市 Marnix17,1000 大道 CEN-CENELEC 会议中心

登记：开放

培训目标

标准日将对欧洲标准化系统、CEN（欧洲标准化委员会）和 CENELEC（欧洲电工标准化委员会）产品和过程、以及潜在的利益相关者参与的好处做出明确的结构化概述。

格式

标准日是一个为期两天的信息会议，包含一个介绍性的全体会议，它说明了 CEN 和 CENELEC 的不同方面与任务（第一天），以及互动、深入的讨论会（第二天）。这种安排能使参与者自身有机会熟悉整个欧洲标准化系统（第一天），然后利用 CEN 和 CENELEC 的专业知识来获得进一步的阐述与说明（第二天）。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

参与对象

标准日是针对所有对欧洲标准化系统感兴趣的参与者：工业、用户、管理部门与政府、欧洲机构、研究中心、国家标准机构、学术界等。

参与

参加是免费的。

(编译自：StandarDays - open info days on European Standardization. http://www.cencenelec.eu/News/Events/Pages/EV-2013-05.aspx. [2013-03-02])

（王妍编译，岳增慧校对）

【标准规范推介】

一、关联数据平台 1.0

一系列最佳实践和一个读写关联数据结构的简单方法，基于 HTTP 能够获取使用 RDF 描述状态的网络资源。

1 简介

这个文档描述了使用HTTP获取、更新、创建和删除资源，这些资源来自将资源揭示为关联数据的服务器。它提供了一些新的规则和说明以及关联数据四原则的扩展。[LINKED-DATA]

1. 使用 URI 作为任何事物的标识名称；
2. 使用 HTTP URI 使任何人都可以访问这些标识名称；
3. 当有人访问某个标识名称时，提供有用的信息；
4. 包含对其他 URI 的链接，以使人们可以发现更多的事物。

包含在这个文档中的最佳实践和反模式是：

- 资源：当构建读取和写入关联数据的客户端和服务器时，你应该使用 HTTP 和 RDF 标准技术还有最佳实践的概要，应该避免反模式。
- 容器：定义允许使用 HTTP POST 创建新资源和使用 HTTP GET 发现已存在资源的资源。

此外，这个文档还旨在使得其他的规则和规则的分层群组成为可能，例如其他的规范。范围上有意缩窄的目的是提供一系列读写关联数据的主要规则。其他的大部分规范，即使不是所有的，都将会依赖于这些主要规则，在实施方面也会支持这些规则。

2 术语

术语基于万维网的W3C结构和超文本传输协议。[HTTP11]

关联

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

当一种资源（表达）参考其他资源，两种资源之间的关系用 URI 的方式表现。

关联数据

如Tim Berners-Lee所定义的。[LINKED-DATA]

关联数据平台资源（LDPR）

在这个文档中符合简单生命周期模式和协议的 HTTP 资源。

关联数据平台容器（LDPC）

在这个文档中管理全体成员的符合其他模式和协议的 LDP 资源。

客户端

一个为了发送请求而建立联系的程序。[HTTP11]

服务器

一个接收通过发送响应接收联系来服务请求的应用程序。任何给定的程序都可能同时作为一个客户端和一个服务器。这些术语的使用仅涉及为一个特殊的关联而设计的程序所扮演的角色，而不涉及一般性的程序能力。同样地，任何服务器可以作为一个原始服务器、代理、路径或通道活动，并根据每个请求的性质改变行为。[HTTP11]

问题八

CLOSED 更好地定义或不仅仅使用“Basic profile”术语。

2.1 这个文档使用的协议

样本资源表达以text/turtle格式提供。[TURTLE]

一般使用命名空间前缀：

```
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#>.
@prefix ldp:      <http://www.w3.org/ns/ldp#>.
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#>.
```

3 一致性

和标识为非标准的部分一样，此规范中所有的编写指南、图表、举例和注释都是非标准的。这个规范中的其他内容是标准化的。

这个规范中的关键词 must、must not、required、should、should not、recommended、may 和 optional 都会在 RFC2119 中得到解释。

4 关联数据平台资源

关联数据平台资源（LDPRs）是与这部分的简单模式和协议相符合的 HTTP 资源。获取、修改、创建和删除 LDPRs 的 HTTP 请求可以通过 LDPR 服务器获取和处理。大部分 LDPRs 是特定领域的资源，这些资源包含关于一些领域实体（可以是商业、政府、科技、宗教或其他）的数据。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

这个文档中定义的一些规则提供了基本关联数据规则的说明和提炼，其他的规则处理另外的需求。

关联数据平台资源的规则致力于解决基本问题，例如：

- 应使用什么样的资源格式？
- 应使用什么样的文本值类型？
- 存在一些应当被复用的典型词表吗？
- 如何处理乐观的冲突检测来为更新服务？
- 客户端期望应该如何适应关联资源的改变，例如类型的变化？
- 服务器应该如何减轻资源创建约束的负担？

以下部分定义了 LDPRs 使用的规则和指南。

4.1 概要

4.1.1 LDPR 服务器必须至少是与 HTTP/1.1 相符合的服务器。

4.1.2 LDPR 服务器必须提供一个关于 LDPRs 的 RDF 表达，主语通常是 LDPR 本身。

4.1.3 LDPR 服务器可以掌控一个 LDPRs 和非 LDPRs 的混合体。例如，对 LDPR 服务器来说，需要控制没有有效 RDF 表达的二进制的或文本的资源是普遍的。

4.1.4 客户端可以使用复合 URLs 来获取一个 LDPR（例如，当域名系统失真时）。一个 LDPR 服务器必须响应每个使用一致的规范的 URL 的请求，因为 LDPR 可能存在于 Location header 的响应中，也可能潜在地存在于 LDPR 的表达中。客户端应该使用规范的 URL 来标识 LDPR。

4.1.5 LDPR 谓语在可能的时候应该使用标准的词表，例如 DC、RDF 和 RDF 模式。LDPRs 应该复用已存在的词表，而不应该创建自己重复的词表术语。

4.1.6 LDPR 谓语必须使用在“4.8 Common Properties”部分中被定义的知名的 RDF 词表（当谓语的意思符合词表中的某一个的时候）。

4.1.6.1 LDPRs 必须使用谓语 rdf:type 来表示类型的概念。不鼓励使用非标准的类型谓语还有 dcterms:type。

4.1.7 LDPR 表示应当明确地有至少一个 rdf:type。这使 得 LDPR 表达对不支持推理的客户端程序更为有效。

4.1.8 在 LDPR 表达中使用的谓语 URLs 应该是 HTTP URLs。这些谓语 URIs 必须标识 LDPRs，它的表达是可检索的。LDPR 服务器应该提供这些谓语的 RDF 模式表示。

问题 9

在 LDPR 表达中使用的属性应当是 LDPRs 吗？

4.1.9 LDPR表达必须使用以下标准的数据类型。RDF自己没有定义文本属性值的数据类型，因此，一系列基于[XMLSCHEMA11-2] 和[RDF-PRIMER]的标准数据类型被使用：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

URI	Description
http://www.w3.org/2001/XMLSchema#boolean	Boolean type as specified by XSD Boolean
http://www.w3.org/2001/XMLSchema#date	Date type as specified by XSD date
http://www.w3.org/2001/XMLSchema#dateTime	Date and Time type as specified by XSD dateTime
http://www.w3.org/2001/XMLSchema#decimal	Decimal number type as specified by XSD Decimal
http://www.w3.org/2001/XMLSchema#double	Double floating-point number type as specified by XSD Double
http://www.w3.org/2001/XMLSchema#float	Floating-point number type as specified by XSD Float
http://www.w3.org/2001/XMLSchema#integer	Integer number type as specified by XSD Integer
http://www.w3.org/2001/XMLSchema#string	String type as specified by XSD String
http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral	Literal XML value as specified by RDF

问题 6

LDBP 应该说任何类型用户定义的简单数据类型都不被允许吗？

4.1.10 LDPRs 必须至少使用一个 RDF 三元组来表达与其他资源的一个关联（关系）。也就是说，在三元组表达的关联中使来源资源 URI 作为主语，使目标资源的 URI 作为宾语是足够的，不要求创建一个媒介关联资源来描述关系。

4.1.11 LDPR服务器可以支持其他标准的表达。这些可以是其他的RDF格式，例如N3 或是NTriples，但是非RDF格式，例如HTML[HTML401]和JSON[RFC4627]可能是常见的。

问题 22

需要规范地参考和推荐 JSON-LD。

4.1.12 LDPRs 可以使用没有在此文档中定义的方法来创建、更新和删除，例如通过特定程序的方式 SPARQL UPDATE 等。

4.1.13 LDPR 服务器响应必须包含准确的响应 ETag 标题值。

问题 10

包含 ETags 的说明和指导。

问题 2

LDPR 版本以系统的可发现的方式被管理吗？

问题 15

共享二进制资源和元数据。

问题 16

非信息资源重定向至 LDPRs。

问题 19

处理更多的错误案例。

4.2 HTTP GET

4.2.1 LDPR 服务器必须支持对 LDPRs 的 HTTP GET 方法。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

4.2.2 LDPR 服务器必须提供被请求的 LDPR 的一个 text/turtle 表达。

4.2.3 LDPR 服务器应该提供被请求的 LDPR 的一个 application/rdf+xml 表达。

问题 23

作为一种需要移除 application/rdf+xml。

4.2.4 在程序或值域的特殊信息缺乏的情况下，LDPR 客户端必须假定任意的 LDPR 拥有对 rdf:type 的多样化的值。

4.2.5 在程序或值域的特殊信息缺乏的情况下，LDPR 客户端必须假定一个给定的 LDPR 的 rdf:type 值可以随着时间而改变。

4.3 HTTP POST

对于 LDPRs 的 HTTP POST 没有附加要求。

4.4 HTTP PUT

4.4.1 如果一个 HTTP PUT 在一个已存在的资源中被执行，LDPR 服务器必须用请求体中的实体表达来取代被标识资源的整个持续状态。唯一公认的例外是 dcterms:modified 属性和 dcterms:creator 属性,它们不受客户端控制—LDPR 服务器必须忽略由客户端提供的这些属性的任意值。任何希望支持一个更复杂的数据合并的 LDPR 服务器（由存储在一个资源服务器中已存在的客户端提供）必须使用 HTTP PATCH，而不是 HTTP PUT。

问题 11

我们需要定义管理服务器的属性还是将其留给程序管理？

4.4.2 LDPR 服务器应该使用 HTTP if-match 标题和 HTTP ETags 来确保它没有修改一个自从客户端上一次获取其表达后改变了的资源。LDPR 服务器应该要求 HTTP if-match 标题和 HTTP ETags 检测冲突。在请求没有其他错误的情况下如果 ETags 不能匹配，则 LDPR 服务器必须以状态编码 402（状态失败）作为响应。

4.4.3 LDPR 客户端应该总是假定任意服务器的一个特定类型资源的谓语是开放的，在这种意义上，相同类型的不同资源可能不会在它们的三元组中拥有相同的谓语，并且在一个资源的状态中使用的谓语不受任何先前定义的谓语的约束。

4.4.4 LDPR 客户端应该假定一个 LDPR 服务器能够丢弃其服务器无法识别或无法选择谓语的三元组。也就是说，LDPR 服务器可以将它们自己限制到一个已知的谓语系列中，但是如果客户端的目的是执行一个稍后的 HTTP PUT 来更新资源时，LDPR 客户端禁止将他们自己限制到一个已知的谓语系列中。

4.4.5 一个 LDPR 客户端必须使用 HTTP GET 保存所有取得的三元组,当它的目的是使用 HTTP PUT 执行一个更新时，不论它是否理解谓语它都不会改变。用 HTTP PATCH 而不是 HTTP PUT 更新为客户端避免了这种负担。

4.4.6 LDPR 服务器可以选择允许使用 HTTP PUT 来创建新资源。

4.4.7 LDPR 服务器应该允许客户端在没有特定服务器约束的详细信息时也可以更新资源。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

对 LDPR 服务器来说，对表达进行限制是普遍的，例如，rdf:type 的范围、谓语的数据类型和三元组中谓语出现的数量，但是服务器应当使那些限制最小化。也就是说，LDPR 服务器需要对 LDPRs 进行最简单的修饰。更为复杂限制的执行将会极大地限制可以修饰资源的客户端的类型。对一些服务器程序来说，可能需要对资源修改进行过多的限制。

4.5 HTTP DELETE

4.5.1 LDPR 服务器必须移除由 Request-URI 标识的资源。在一个成功的 HTTP DELETE 之后，相同的 Request-URI 上一个接续的 HTTP GET 必须产生一个 404（没找到）或 410（丢失）状态编码，直到有相同 Request-URI 的资源被创建或关联。

4.5.2 LDPR 服务器可能因为一个 HTTP DELETE 的请求改变其它资源的状态。例如，对服务器来说从其它资源（主语或宾语是被删除的资源）中移除三元组是可接受的。对 LDPR 服务器来说，不这样做也是合理的普遍的——行为是服务器程序特定的。

问题 24

DETELD 资源应该保持删除状态吗？

4.6 HTTP HEAD

4.6.1 LDPR 服务器必须支持 HTTP HEAD 方法。

4.6.2 LDPR 服务器必须通过响应一个 HTTP HEAD 请求来表明对 HTTP Methods 的支持。

4.7 HTTP PATCH

4.7.1 LDPR 服务器可以执行 HTTP PATCH 来允许对它们资源的修改，特别是部分替代。这个文档中没有要求修改文档格式的最小集合。

4.7.2 LDPR 服务器应该允许客户端在没有要求特定服务约束的详细信息时更新资源。对 LDPR 服务器来说，对表达进行限制是普遍的，例如，rdf:type 的范围、谓语的数据类型和三元组中谓语出现的数量——但是服务器详细的、特定值域约束的执行将极大地限制可以更新资源的客户端的类型。

问题 12

HTTP PATCH 可以被用于资源创建吗？

问题 17

变更集合作为一个推荐的 PATCH 格式。

4.8 通用属性

这一部分总结了一些应用在关联数据平台资源中的知名的 RDF 词表（当一个资源需要使用含义符合词表的谓语时）。例如，如果一个 BP 资源有一个描述且那个描述的应用语义与 dcterms:description 是相容的，那么就必须使用 dcterms:description。如果需要的话，其它特殊程序的谓语可以被使用。基于 BP 的一个领域的规范可能会需要一个特定资源类型中的一个或多个属性。

下表中 Range 栏标识了关于属性的推荐的 rdfs:range。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

4.8.1 来自 DC

URI：http://purl.org/dc/terms/。

Property	Range/DataType	Comment
dcterms:contributor	dcterms:Agent	
dcterms:creator	dcterms:Agent	
dcterms:created	xsd:dateTime	
dcterms:description	rdf:XMLLiteral	Descriptive text about the resource represented as rich text in XHTML format. SHOULD include only content that is valid and suitable inside an XHTML element.
dcterms:identifier	rdfs:Literal	
dcterms:modified	xsd:dateTime	
dcterms:relation	rdfs:Resource	The HTTP URI of a related resource. This is the predicate to use when you don't know what else to use. If you know more specifically what sort of relationship it is, use a more specific predicate.
dcterms:subject	rdfs:Resource	
dcterms:title	rdf:XMLLiteral	A name given to the resource. Represented as rich text in XHTML format. SHOULD include only content that is valid inside an XHTML element.

不应该使用谓语 dcterms:type，应该使用 rdf:type。

4.8.2 来自 RDF

URI：http://www.w3.org/1999/02/22-rdf-syntax-ns#。

Property	Range	Comment
rdf:type	rdfs:Class	The type or types of the resource

4.8.3 来自 RDF 模式

URI：http://www.w3.org/2000/01/rdf-schema#。

Property	Range	Comment
rdfs:member	rdfs:Resource	
rdfs:label	rdfs:Literal	Only use this in vocabulary documents, to define the name of the vocabulary term.

5 关联数据平台容器

5.1 信息性的

这部分是非标准的。

很多 HTTP 程序和网站组织了将资源整体划分成更小的容器的概念。博客群组被分成 blogs，维基网页被分成 wikis,产品被分成目录。每种在程序或网站中创建的资源可以在像容器一样的实体的实例中创建，用户可以在一个实例中罗列出已存在的人工制品。容器回答下列基本问题：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

1. 我可以发到哪些 URLs 上来创建新资源？
2. 我从哪里得到已存在资源的列表？
3. 容器入口的顺序是如何表达的？
4. 我如何得到关于成员和容器的信息？
5. 我如何得到被分成网页的一个大的容器的入口？
6. 我如何确保资源数据能够容易地询问？

这个文档定义了解释这些问题的容器的表达和行为。一个容器的成员集合由称为全体成员三元组的表达中的三元组集合来定义。一个容器的全体成员三元组都拥有相同的主语和谓语——全体成员三元组的宾语定义了容器的成员。全体成员三元组的主语被称为成员主语，谓语被称为成员谓语。在最简单的案例中，成员主语将是 LDPC 资源本身，但这不是一定的。成员谓语也是可变的，通常是来自服务器程序词表的谓语或是 rdfs:member 谓语。

这个文档包括了使用 POST 创建新资源并将它们添加到容器成员列表中的一系列指导方针。这个文档也解释了如何包含关于容器表达的每个成员的信息以及在容器变得太大时如何为容器的表达标记页数。

以下举例说明了一个只有 3 个成员的简单容器和一些关于容器的信息（事实上，它是一个容器和一个简洁的标题）：

EXAMPLE 1

```
# The following is the representation of
#    http://example.org/container1
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.

<http://example.org/container1>
   a ldp:Container;
   dcterms:title "A very simple container";
   rdfs:member
      <http://example.org/container1/member1>,
      <http://example.org/container1/member2>,
      <http://example.org/container1/member3>.
```

这个例子是非常简明的，成员谓语是 rdfs:member，成员主语是容器自身。这个容器的 POST 将创建一个新资源，并通过添加一个新的全体成员三元组到容器中来将容器添加到成员列表。

有时用一个主语而不是容器自身作为成员主语更有效，同样的，用一个谓语而不是 rdfs:member 作为成员谓语更有效。以下举例说明：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

EXAMPLE 2

```
# The following is the representation of
#   http://example.org/netWorth/nw1/assetContainer
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.
@prefix o: <http://example.org/ontology/>.

<http://example.org/netWorth/nw1/assetContainer>
   a ldp:Container;
   ldp:membershipSubject <http://example.org/netWorth/nw1>;
   ldp:membershipPredicate o:asset.

<http://example.org/netWorth/nw1>
   a o:NetWorth;
   o:asset
      <http://example.org/netWorth/nw1/assetContainer/a1>,
      <http://example.org/netWorth/nw1/assetContainer/a2>.
```

容器的基本结构是相同的，但在这个例子中，成员主语不是容器自身——它是一个分离的网状价值资源。成员谓语是 o:asset——来自值域模型的一个谓语。这个容器的一个 POST 将创建一个新的资产并通过将一个新的全体成员三元组添加到容器中来将容器添加到成员列表。你可能想知道为什么我们不仅仅使 http://example.org/netWorth/nw1 作为一个容器并直接发送新的资产到那里。如果 http://example.org/netWorth/nw1 有唯一的资产，这将是个好的设计，但如果它对资产和负债有单独的谓语，那个设计就没有发挥作用，因为它没有指明 POST 应该添加一个全体成员三元组到哪个谓语。拥有单独的 http://example.org/netWorth/nw1/assetContainer
和 http://example.org/netWorth/nw1/liabilityContainer 容器资源允许资产和负债都能被创建。

在这个例子中，客户不能简单地猜测哪个资源是成员主语，哪个谓语是成员谓语，所以这个例子在主语是 LDPC 资源本身的三元组中包含了这项信息。

5.1.1 容器成员信息

这一部分是非标准的。

在很多（可能是大部分）包括容器的程序中，客户端想要能够得到关于每一个容器成员的信息，而不用 在每个容器中做一个 GET。LDPC 允许服务器在容器的表达中直接包含这项信息。服务器决定了每个所提供成员的数据的数量。一些通用的战略包含提供一个标准属性的固定数量或提供每一个成员资源的全部的 RDF 表达或什么也不提供。服务器程序领域和它的应用案例将决定需要多少信息。

从网状价值例子开始，在表达中将有成员资源（资产）的其它三元组：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

EXAMPLE 3

```
# The following is the representation of
#        http://example.org/netWorth/nw1/assetContainer
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#>.
@prefix ldp:      <http://www.w3.org/ns/ldp#>.
@prefix o:       <http://example.org/ontology/>.

<http://example.org/netWorth/nw1/assetContainer>
   a ldp:Container;
   dcterms:title "The assets of JohnZSmith";
   ldp:membershipSubject <http://example.org/netWorth/nw1>;
   ldp:membershipPredicate o:asset.

<http://example.org/netWorth/nw1>
   a o:NetWorth;
   o:asset
      <http://example.org/netWorth/nw1/assetContainer/a1>,
      <http://example.org/netWorth/nw1/assetContainer/a3>,
      <http://example.org/netWorth/nw1/assetContainer/a2>.

<http://example.org/netWorth/nw1/assetContainer/a1>
   a o:Stock;
   o:value 10000.
<http://example.org/netWorth/nw1/assetContainer/a2>
   a o:Bond;
   o:value 20000.
<http://example.org/netWorth/nw1/assetContainer/a3>
   a o:RealEstateHolding;
   o:value 300000.
```

5.1.2 仅获取非成员属性

这一部分是非标准的。

拥有许多成员的容器的表达是巨大的。有一些重要的案例，在这些案例中客户端仅需要获取容器的非成员属性。因为对客户端来说，获取整个容器表达来得到这些信息可能是艰巨的，并且对服务器产生不必要的负担。我们希望定义一种方法来仅获得非成员属性值。对每一个 LDPC 通信资源（称为非成员资源，其状态为容器状态的子集）进行定义，做了上述工作。

这里列举的例子只是展示了一个简单的案例，仅仅获取了一些简单的非成员属性。在现实环境中更复杂的案例是有可能的，例如那些添加其它谓语到容器中的，例如提供验证信息和 SPARQL 端点关联。

这是一个要求一个容器的非成员属性的例子，由 http://example.org/container1 标识并添加查询字符串?non-member-properties：

请求：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

EXAMPLE 4

```
GET /container1?non-member-properties HTTP/1.1
Host: example.org
Accept: text/turtle; charset=UTF-8
```

响应：

EXAMPLE 5

```
HTTP/1.1 200 OK
Content-Type: text/turtle; chartset=UTF-8
ETag: "_87e52ce291112"
Content-Length: 325

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.

<http://example.org/container1>
   a ldp:Container;
   dcterms:title "A Linked Data Platform Container of Acme Resources";
   ldp:membershipPredicate rdfs:member;
   dcterms:publisher <http://acme.com/>.
```

5.1.3 标页码

这一部分是非标准的。

有时候一个容器太大了而不能合理地在一个单一的 HTTP 响应中传送它的表达。当容器表达包含很多来自其成员表达的三元组时，上述情况将是真实的。一个客户端可能预期一个容器将是很巨大的，例如一个获取缺陷的客户端工具可能假定单个缺陷通常是限制大小的，它能立刻对请求产生作用，但是所有被创建的缺陷的容器通常都很大。或者，一个服务器可能识别出已被请求的容器太大了而不能使用单一的信息返回。

为了解释这个问题，LDPCs 可以支持一个叫做 paging 的技术。Paging 可以用一个简单的 RDF 模式来获取。对每个容器资源<containerURL>，我们定义一个新的资源<containerURL>?firstPage 。在<containerURL>?firstPage 表达的三元组是<containerURL>三元组的一个子集——有相同的主语、谓语和宾语。

LDPC 服务器通过将客户端转向首页资源——使用了一个 303“See Other”，转向指向网页资源的真实 URL 来响应容器的请求。

从来自JohnZSmith网状价值实例的成员信息继续，我们将通过两个页面分别响应。客户端请求首页http://example.org/netWorth/nw1/assetContainer?firstPage：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

EXAMPLE 6

```
# The following is the representation of
#    http://example.org/netWorth/nw1/assetContainer?firstPage
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.
@prefix o: <http://example.org/ontology/>.

<http://example.org/netWorth/nw1/assetContainer>
   a ldp:Container;
   dcterms:title "The assets of JohnZSmith";
   ldp:membershipSubject <http://example.org/netWorth/nw1>;
   ldp:membershipPredicate o:asset.

<http://example.org/netWorth/nw1/assetContainer?firstPage>
   a ldp:Page;
   ldp:pageOf <http://example.org/netWorth/nw1/assetContainer>;
   ldp:nextPage <http://example.org/netWorth/nw1/assetContainer?p=2>.

<http://example.org/netWorth/nw1>
    a o:NetWorth;
        o:asset
        <http://example.org/netWorth/nw1/assetContainer/a1>,
        <http://example.org/netWorth/nw1/assetContainer/a4>,
        <http://example.org/netWorth/nw1/assetContainer/a3>,
        <http://example.org/netWorth/nw1/assetContainer/a2>.

<http://example.org/netWorth/nw1/assetContainer/a1>
   a o:Stock;
   o:value 100.00.
<http://example.org/netWorth/nw1/assetContainer/a2>
   a o:Cash;
   o:value 50.00.
# server initially supplied no data for a3 and a4 in this response
```

下面的例子是获取下一页表达的结果：

EXAMPLE 7

```
# The following is the representation of
#  http://example.org/netWorth/nw1/assetContainer?p=2
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.
@prefix o: <http://example.org/ontology/>.

<http://example.org/netWorth/nw1/assetContainer>
   a ldp:Container;
   dcterms:title "The assets of JohnZSmith";
   ldp:membershipSubject <http://example.org/netWorth/nw1>;
   ldp:membershipPredicate o:asset.

<http://example.org/netWorth/nw1/assetContainer?p=2>
   a ldp:Page;
   ldp:pageOf <http://example.org/netWorth/nw1/assetContainer>;
   ldp:nextPage rdf:nil.

<http://example.org/netWorth/nw1>
   a o:NetWorth;
   o:asset
      <http://example.org/netWorth/nw1/assetContainer/a5>.

<http://example.org/netWorth/nw1/assetContainer/a5>
   a o:Stock;
   dcterms:title "Big Co.";
   o:value 200.02.
```


本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

在这个例子中，在最后页面仅有容器中的一个成员。为了指明这是最后的页面，使用一个 rdf:nil 值用作网页资源 ldp:nextPage 谓语。

LDPC 保证关于成员的任何和所有三元组将在相同的页面上作为成员的全体成员三元组。

5.1.4 排序

这一部分是非标准的。

有很多容器成员的顺序很重要的案例。LDPC 没有对容器中成员的服务器排序提供任何特殊的支持，因为任何客户端都可以基于可获得的成员的属性值来选择一种方法对成员进行排序。在下面的例子中，o:value 谓语的值对每个成员都是存在的，所以客户端可以通过那个属性值轻易地对成员进行排序。通过这种方法，LDPC 避免了 RDF 构造（如 Seq 和表达顺序的 List）的使用。

当容器被标页码的时候，顺序才会对 LDPC 服务器重要。如果服务器在构建页面的时候没有注意排序，客户端在成员分类前就要被迫去获取所有的页面，这将导致标页码目的的失败。在排序重要的案例中，一个 LDPC 服务器将所有的成员显示在一个页面上，比前一页的所有成员有更高的分类顺序，比下一页的所有成员有更低的分类顺序。LDPC 规范提供了一个谓语——ldp:containerSortPredicates——服务器可以使用这个谓语与客户端交流，此客户端使用谓语来进行页面排序。多样的谓语值可以用来分类，因此这个谓语的值是一个有序的列表。

这是一个之前描述过的示例容器，拥有对资源进行排序的表达：

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

EXAMPLE 8

```
# The following is the ordered representation of
#   http://example.org/netWorth/nw1/assetContainer
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.
@prefix o: <http://example.org/ontology/>.

<http://example.org/netWorth/nw1/assetContainer>
   a ldp:Container;
   dcterms:title "The assets of JohnZSmith";
   ldp:membershipSubject <http://example.org/netWorth/nw1>;
   ldp:membershipPredicate o:asset.

<http://example.org/netWorth/nw1/assetContainer?firstPage>
   a ldp:Page;
   ldp:pageOf <http://example.org/netWorth/nw1/assetContainer>;
   ldp:containerSortPredicates (o:value).

<http://example.org/netWorth/nw1>
   a o:NetWorth;
   o:asset
      <http://example.org/netWorth/nw1/assetContainer/a1>,
      <http://example.org/netWorth/nw1/assetContainer/a3>,
      <http://example.org/netWorth/nw1/assetContainer/a2>.

<http://example.org/netWorth/nw1/assetContainer/a1>
   a o:Stock;
   o:value 100.00.
<http://example.org/netWorth/nw1/assetContainer/a2>
   a o:Cash;
   o:value 50.00.
<http://example.org/netWorth/nw1/assetContainer/a3>
   a o:RealEstateHolding;
   o:value 300000.
```

正如你所看到的，通过 ldp:containerSortPredicates 谓语的增加，o:value 谓语被用来定义结果的顺序。由值域模型和服务器决定适合的谓语来表明同一页面里的资源顺序，由获得这个表达的客户端决定以任何合适的方法使用那个顺序，例如在一个用户界面中对先于表达的数据进行分类。

5.2 概要

这一部分是非标准的。

5.2.1 LDPC 服务器也必须与 LDPR 服务器相符合。一个关联数据平台容器是一种是关联数据平台资源的资源。

5.2.2 同样的资源可以是多样的 LDPCs 的一个成员。这通常发生在如果一个容器是一个更大的容器的概览时。

5.2.3 一个 LDPC 的表达包含关于哪些资源是其成员的信息。一个容器的一系列成员是称为全体成员三元组的表达中的三元组集合。一个容器的全体成员三元组有相同的主语和谓语——全体成员三元组的宾语定义了容器的成员。全体成员三元组的主语被称为成员主语，谓语被称为成员谓语。在最简单的案例中，成员主语将是 LDPC 资源本身，但这不是一定的。成员谓语也是可变的，通常是来自服务器程序词表的谓语。如果没有来自服务器程序词表的明显谓语可以使用，LDPC 服务器应该使用 rdfs:member 谓语。

问题 21

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

容器容量

5.2.4 一个 LDPC 必须包含一个 ldp:membershipSubject 谓语的三元组，当容器主语不是 LDPC 自身时用以指明全体成员三元组的主语。

5.2.5 一个 LDPC 必须包含一个 ldp:membershipPredicate 谓语的三元组，当容器谓语不是 rdfs:member 时用以指明全体成员三元组的谓语。

5.2.6 一个 LDPC 表达可以包含任意数量的其他三元组，这些三元组的主语是容器的成员，或是来自成员表达（如果它们有 RDF 表达）。这允许一个 LDPC 服务器提供成员的信息给客户端而不需要客户端对每个成员单独做一个 GET。更多的信息看 5.1.1 Container Member Information 部分。

问题 13

包含含有成员三元组的 LDPC 表达的说明。

5.2.7 一个 LDPC 表达必须有 ldp:Container 的 rdf:type，但它也可以有其它的 rdf:types。

5.2.8 LDPCs 不应该使用 RDF 容器类型 rdf:Bag，rdf:Seq 和 rdf:List。

问题 3

LDPC 版本是以系统的可发现的方法被管理吗？

5.3 HTTP GET

5.3.1 一个 LDPC 表达必须包括一系列拥有一致主语和谓语的三元组，它们的宾语指向容器成员。三元组的主语可以是容器自身或是其它资源（如上述例子提到的）。参见 5.2.3。

5.3.2 LDPC 服务器应该通过 LDPC URL 查询组件“non-member-properties”的存在来支持关于一个已知的 LDPC 信息的请求，而不需要获取包含所有成员的一个完整表达。例如，如果有一个 LDPC URL <containerURL>,请求非成员关系属性的 URL 将是<containerURL>? non-member-properties。更多信息参见 5.1.2 Retrieving Non-member Properties 部分。一个通过<containerURL>? non-member-properties 的 Request-URI 不支持获取非成员资源属性的 LDPC 服务器，必须返回一个 HTTP 状态编码 404（没找到）。

5.3.3 一个通过作为<containerURL>已知的 LDPC 不支持获取首页资源表达的 LDPC 服务器，必须返回一个 HTTP 状态编码 404（没找到）。

5.3.4 LDPC 服务器必须支持将大的 LDPCs 分割成多个页面的请求，这些页面由客户端通过提供对 LDPC URL 的查询组件 firstPage 来指明。例如，如果有一个 LDPC URL<containerURL>，请求首页的 URL 将会是<containerURL>? firstPage。任意页面的表达，包含首页，将包含下一页面的 URL。更多信息参见 5.1.3 titled “Paging”部分。

5.3.5 LDPC 服务器可以分割一个 LDPC 的响应表达，而不管客户端请求的是什么。（例如当一个客户端省略一个请求 URL 的查询组件“firstPage”时）。这也可以当作服务器发起的标页码得知。更多信息参见 5.1.3 Paging。

5.3.5.1 发起标页码的 LDPC 服务器应该通过将客户端转向网页资源（通过一个 303“See

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

Other”转向网页资源的真实的 URL）来对一个 LDPC 的请求做出响应。

5.3.6 支持标页码的 LDPC 服务器必须包含在页面表达中，例如一个对于 LDPC 的表达：

5.3.6.1 页面资源表达必须有一个三元组来指明它的类型，这个三元组的主语是页面的 URL，谓语是 rdf:type，宾语是 ldp:Page，它也应该有一个三元组来指明正在标页码的容器，其主语是页面的 URL，谓语是 ldp:pageOf，宾语是 LDPC 的 URL。

5.3.6.2 页面资源表达必须有一个三元组，其主语是页面，谓语是 ldp:nextPage,宾语是接下来页面的 URL。

5.3.6.3 最后一页资源表达必须有一个三元组，其主语是最后一页，谓语是 ldp:nextPage，宾语是 rdf:nil。

问题 18

容器成员和稳固的分页。

5.3.7 LDPC 服务器可以以连续的顺序表达被标页的 LDPC 的成员。顺序必须是详细的，使用 ldp:containerSortPredicates 谓语，其主语是页面，宾语是 LDPC 序数谓语的列表。默认排序是上升的。文本数据类型支持的唯一的序数谓语是由 SPARQL SELECT 的 ORDER BY 语句定义的。

问题 14

包含关于在 LDPC 表达中排序的说明。

5.3.7.1 ldp:containerSortPredicates 的宾语，指明排序使用的谓语，禁止在接下来的页面之间改变。如果改变了，跨越页面的容器成员间的排序是非定义的。更多信息参见 5.1.4 Ordering。

关联数据平台没有定义客户端如何发现 LDPCs。

5.4 HTTP POST

5.4.1 LDPC 客户端应该通过将作为 HTTP POST 实体的一个表达传送给一个已知的 LDPC 来创建资源。LDPC 服务器必须通过状态编码 202（已创建的）和新资源 URL 的 Location 标题进行响应。

5.4.2 在对一个 LDPC 成功的 HTTP POST 请求后，新资源必须作为 LDPC 的一个成员出现，直到新资源被删除或通过其它方法被移除。LDPC 也可以容纳通过其它方式添加的资源，例如通过实现 LDPC 网站的用户界面。

5.4.3 LDPC 服务器可以接收非 RDF 表达的 HTTP POST 来支持任意类型资源的创建，例如二进制资源。

5.4.4 对于支持创建的服务器，LDPC 服务器必须在请求实体中创建一个来自 RDF 表达的 LDPR。

5.4.5 LDPC 服务器不应该包含在一个 201 响应实体中被创建的资源表达。换句话说，客户端不应该在一个 201 响应的实体中期望任何表达。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

5.4.6 对 LDPCs 来说，服务器必须接收带有一个 text/turtle 内容类型的请求实体。

5.4.7 对 LDPCs 来说，LDPR 服务器应该接收一个带有 application/rdf+xml 内容类型的请求。

5.4.8 对 RDF 表达来说，当提到请求主体的实体时，LDPC 服务器必须解释在 LDPR 表达中关于三元组主语的无效但相关的 URI。通常，那个实体是“to be created”的 LDPR 的模型，因此主语是无效相关 URI 的三元组将产生被创建资源（其主语是被创建的资源）的三元组。

问题 20

标识和命名 POSTed 资源。

5.4.9 LDPC 服务器应该用服务器程序特定规则为将被创建的资源分配主语 URI。

5.4.10 LDPC 服务器应该允许客户端不需要知道特定程序约束的详细内容就能创建新资源。一些 LDPC 服务器对表达进行约束，例 rdf:type 的范围，谓语的数据类型还有三元组中谓语出现的数量。但是服务器详细的特定值域的强制约束将极大地限制可以创建资源的客户端类型，因此是不被鼓励的。

5.5 HTTP PUT

5.5.1 LDPC 服务器不应该允许 HTTP PUT 更新 LDPC 的成员，如果服务器接收到这样一个请求，它应该返回 409（冲突）状态编码。

5.5.2 LDPC 服务器可以允许通过使用<containerURL>?non-member-properties 的 HTTP PUT 来更新 LDPC 非成员属性，它可以排除服务器的管理属性，例如 ldp:membershipSubject 和 ldp:membershipPredicate。

5.6 HTTP DELETE

5.6.1 如果一个 LDPC 服务器支持对 LDPC 的删除，服务器也可以删除参考其内容的资源。

5.6.2 当包含在一个 LDPC 中的资源（例如，由全体成员三元组进行参考的）被删除时，服务器也必须从用来创建它的 LDPC 中移除它，而且应该从任何引用它的容器中移除它。

5.7 HTTP HEAD

对于 HTTP HEAD 没有另外的要求。

5.8 HTTP PATCH

5.8.1 LDPC 服务器支持 HTTP PATCH 作为优先的方法来更新 LDPC 非成员关系属性。

问题 7

允许对容器进行什么操作，它们是如何被调用的？

(编译自：Linked Data Platform 1.0. http://www.w3.org/TR/2012/WD-ldp-20121025/. [2012-10-25])

（吴贝贝编译，王妍校对）

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

二、IMS 学习工具互操作性（LTI）消息框架版本 2.0 公共草案

1 概述

此文件是一个实施指南，它解释了消息是如何从工具使用者通过工具提供者发送到消息处理者的。它为那些消息的传输和接收定义了一个行为模型，并且描述了作为一个 HTTP 请求绑定一个消息的步骤。

1.1 概念性介绍

消息产生于工具使用者并且发送到工具提供者中的消息处理器，如图 1.1 所示。实际上，消息也许并不直接来自于工具使用者。例如，对于一个工具启动请求，消息也许来自于用户的浏览器。然而，即使在这种情况下，消息是在工具使用者中被构建并传递给浏览器。浏览器只是转发消息到消息处理器。因此，在所有情况下，消息确实起源于工具使用者。

Figure 1.1□ Messaging from Tool Consumer to Tool Provider.

在 LTI 标准中，每种类型的消息都被建模为一个不同的类。重要的是要记住我们把这些实体说成是类纯粹是为了说明的便利。实现者不需要在一些编程语言（如 Java 或 C#）中实现这些类。事实上，正如在第 4 部分讨论的那样，只有 LTI 标准所需要的消息绑定是基于 HTTP 请求的协议而编码的。其他绑定和其他协议会在之后的标准版本中介绍，但是在目前的版本中 LTI 消息仅仅是 HTTP 请求。LTI 标准并没有为消息处理器定义任何的绑定。他们也许通过使用任何能处理 HTTP 请求的可用技术来实现，例如 Java Servlets、动态服务器页面、PHP 等。

1.2 术语

LTI 规范中经常使用的词汇的列表和定义包含在 LTI 实现指南文件中[LTI ,12IMG]。

1.3 文件的结构

此文件的结构是：

2. 关键概念
3. 行为模型
4. 绑定和实现指南

2 关键概念

2.1 消息类型

LTI 定义了三种不同类型的消息：

1. 基本 LTI 启动请求

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

2. 工具代理注册请求

3. 工具代理再注册请求

每种消息类型都有一个在工具资料中出现的符号名，在这里工具提供者声明它所支持的消息处理器的类型。符号名也在工具使用者资料中出现，在这里工具使用者声明它能发出消息的类型。三种支持的消息类型的符号名分别是：（1）balsc-lti-launch-request，（2）ToolProxyRegistrationRequest，（3）ToolProxyReregistrationRequest。

这些消息是一个工具启动请求的示例，即一个从用户的浏览器发布的一个 HTTP 请求。当用户跟随一个链接或点击工具使用者系统中的一个按钮并导航至工具提供者系统时，这个提交发生。

2.2 工具启动服务

工具使用者实现了负责生成工具启动请求的工具启动服务。启动一个外部工具的请求被提交到工具使用者中的工具启动服务。启动服务产生一个封装了工具启动请求的 HTML 表单，并且它在 HTTP 响应中给用户的浏览器发送此表单。浏览器自动地把表单内容发布给工具提供者中的合适的消息处理器。这个序列在使用案例 LTIv1-25 中被描述。可参见 4.1 节关于消息的 HTML 表单绑定的讨论。

2.3 传输方法

LTI 标准要求所有的 LTI 消息必须以 HTTP(S) 请求被传递。使用“application/x-www-form-urlencoded”编码通过 HTTP(S)POST 或 GET 来发送消息。

2.4 消息参数

每一个具体的消息类型都被一个类正式地定义，该类从被称作 mixins 基类的集合中继承属性，如图 2.1 所示。具体的类和它的 mixins 定义了与给定消息类型相关的参数标准集。第 4 部分解释了类模型中的属性是如何映射到 HTML 表单消息编码中的参数。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

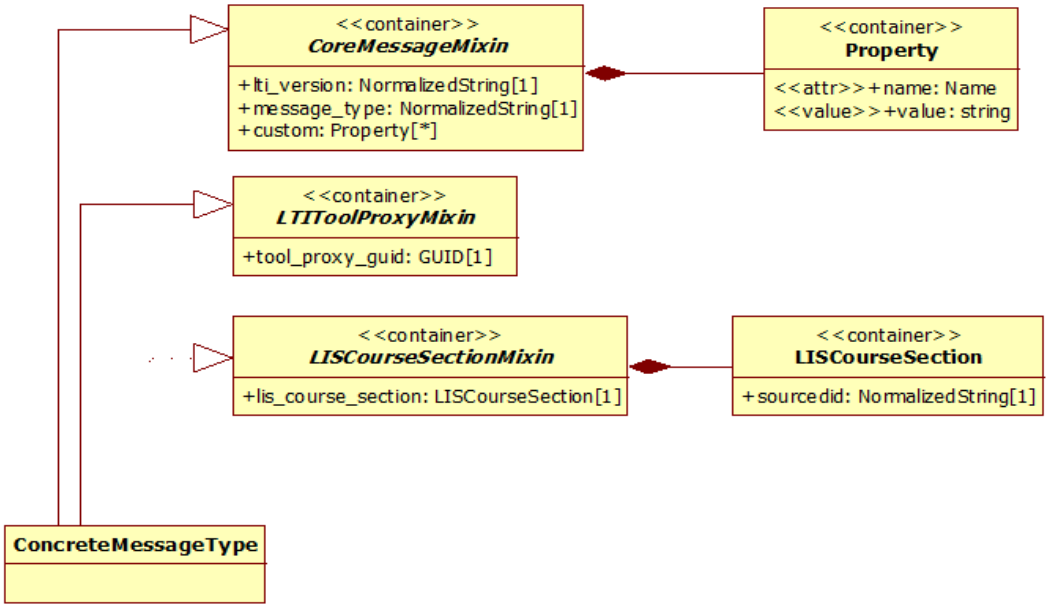

Figure 2.1□ A fictitious concrete message type illustrating multiple inheritance

2.4.1 自定义参数

除了在类模型中指定的标准参数外，每个消息也包含一个自定义参数集。这些自定义参数被建模为 CoreMessageMixing 中的 Property 对象集（见图 2.1）。注意每个 Property 仅仅是任意名称/值对。

自定义参数的值有两个不同的来源：

- 工具资料中的参数定义
- 链接的创建者增添的参数

工具资料中的参数定义在 2.5 节中讨论。

LTI 标准没有指定一个特定的用户界面来编辑链接。每个工具使用者可实现自己的用户界面，但必须使链接创建者可以为任何链接添加任意的名称/值对。

当一个自定义参数映射到 HTTP 请求中的 POST 参数时（参见 4.1 节），参数名 以“custom_”为前缀来呈现。这种做法可以确保类模型中定义的标准参数名不会发生冲突。例如，如果链接的创建者增加一个名为“learning_objective”的自定义参数，POST 参数会命名为“custom_learning_objective”。

2.5 消息处理器声明

每个工具向工具使用者注册它能处理的消息类型。这些消息处理器定义出现在工具资料中。正如图 2.3 所示，消息处理器定义包括工具提供者系统中的消息类型的符号名称和消息处理器的端点 URI。端点定义是一个路径，该路径与工具资料中的工具信息部分声明的基本 URL 相关。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

```
□
□ "message" : [
□□□ {
□□□□□ "message_type" : "basic-lti-launch-request" ,
□□□□□ "path" : "resource/homework" ,
□□□□□ "parameter" : [
□□□□□□□ { "name" □ : "discipline" ,
□□□□□□□□□ "fixed" : "chemistry"
□□□□□□□ },
□□□□□□□ { "name" □□□□ : "lis_person_name_given" ,
□□□□□□□□□ "variable" : "$Person.name.given"
□□□□□□□ },
□□□□□□□ { "name" : "textbook_isbn" },
□□□□□□□ { "name" : "chapter" }
□□□□□]
□□□ }
□ ]
□
```

Figure 2.2 Message Handler definition in Tool Profile.

工具使用者在工具代理的部署过程中发现这些消息处理器。当定义自定义参数时，工具提供者可选择该参数分配一个固定值或一个变量值，如图 2.3。

在这个例子中，当工具使用者在指定的路径发布一种 basic-lti-launch-request 类型的消息给工具提供者时，它会以下列名称传递四个自定义参数。

- custom_discipline
- custom_lis_person_name_given
- custom_textbook_isbn
- custom_chapter

参数 custom_discipline 的值将总是“化学”，因为这个参数是以固定值声明的。

参数 custom_lis_person_name_given 的值取决于点击链接的特定用户，因为这个参数是以变量值声明的。有关变量的更多信息，参见[LTI,12IMG]。

工具资料没有指明参数 custom_textbook_isbn 和 custom_chapter 的值。链接创建者必须通过工具使用者显示的用户接口来显式地提供这些值。

2.6 消息传递功能

一个给定的工具使用者系统也许能够发出某些类型的消息而不是其他消息。工具使用者必须声明在工具使用者资料的功能部分所支持的消息的特定类型。例如，可以发出 basic-lti-launch-request 和 ToolProxyReregistrationRequest 的工具使用者要声明图 2.5 所示的功能。另外，工具使用者也必须声明它支持的变量的功能。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

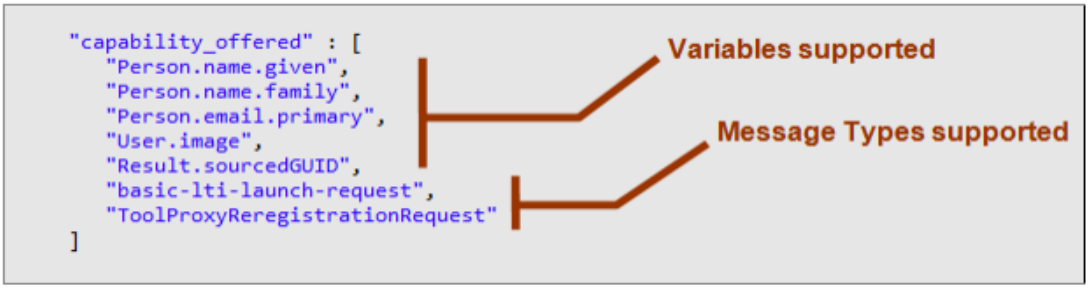

Figure 2.5 Capabilities declared in Tool Consumer Profile

3 行为模型

3.1 行为模型概述

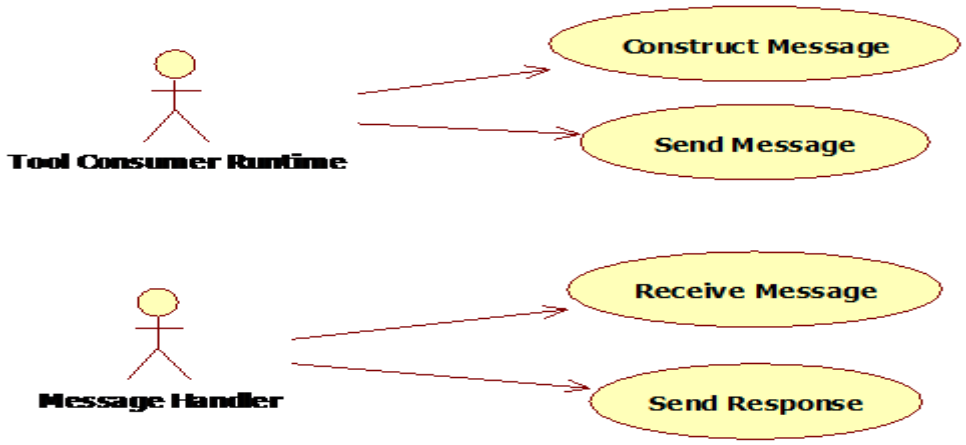

Figure 3.1 Diagram depicting the Messaging Framework behavioral model.

消息框架的行为模型是直接的。工具使用者运行负责构建和发送消息。工具提供者中的消息处理器接收消息并发送响应。

此部分解释了工具使用者运行如何构建信息。正如万维网联盟（http：//www.w3.org/Protocols）定义的，使用标准 HTTP 协议来发送消息。

消息处理器可以接收使用任何可用技术的消息来处理 HTTP 请求。

3.2 使用案例

此部分描述了从工具使用者向工具提供者传递信息的使用案例。在 LTI 2.0 中，只有一种消息使用案例。LTI 的未来版本也许会介绍额外的使用案例，即在工具提供者中工具使用者运行调用 HTTP 消息处理器。

3.2.1 从工具提供者向用户提供服务内容

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

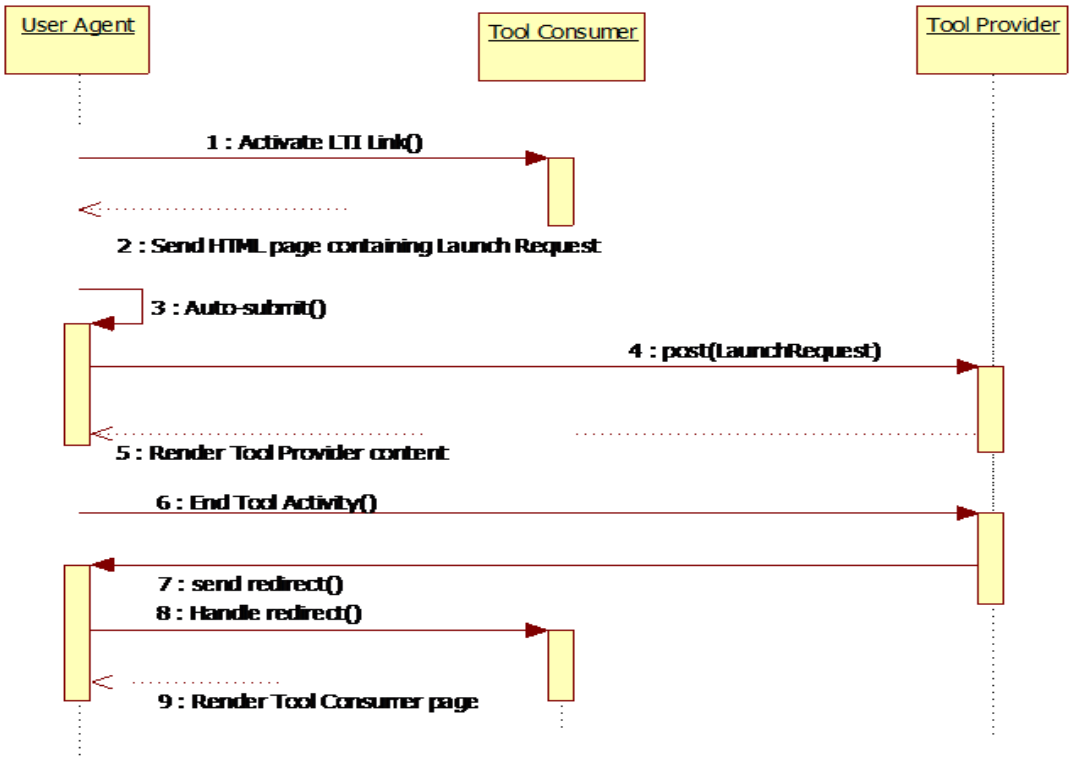

Figure 3.2 Sequence diagram illustrating content being served to the user from the Tool Provider.

使用案例名称	**从工具提供者向用户提供服务内容**
使用案例本地 ID	LTIv1-25
简要说明	在此使用案例中，用户最初在工具使用者的用户界面中导航，并转移到工具提供者的用户界面。这个序列包含一个单点登录协议以便于工具提供者可以使基于工具使用者签署证书的用户可信。从工具使用者到工具提供者的启动请求包括关于学习环境的足够信息来确保一致的用户体验。
级别	概要
角色	工具使用者运行时间、工具提供者消息处理器、用户
先决条件	在工具使用者中部署和启用工具代理。 用户登录到工具使用者。
基本事件流	1. 激活 LTI 链接。用户在他们的浏览器中点击由工具使用者呈现的链接，并且在工具使用者中将 HTTP 请求发送给工具启动服务。 2. 发送包含启动请求的 HTML 页面。工具启动服务生成一个包含表单的 HTML 页面，表单的“方法”属性是“POST”。表单字段对应于 LTI 启动请求消息的序列化。 3. 自动提交。HTML 页面包含 JavaScript 代码来自动提交表单。表单的目标应该是工具提供者系统中适当的消息处理器的 URL，正如在工具资

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

	料中定义的（即 URL 由基本 URL 和消息“路径”结合来构建）。 4. 发布启动请求。用户的浏览器将包含有启动请求的表单发布给工具提供者的服务器。 5. 呈递工具提供者的内容。工具提供者按照在工具部署过程中协商的安全概要来验证启动请求。除了其他方面，启动请求包含一个“返回 URL”参数。工具提供者保存这个 URL，以便当用户完成与工具的交互时它能够将用户的浏览器重新定向到工具使用者系统（见步骤 7）。最后，工具提供者通过将请求内容发送到用户的浏览器来响应启动请求。 6. 终止工具活动。用户与工具交互，最终执行一些操作来“关闭”工具。 7. 发送重定向。当工具提供者收到请求来关闭工具时，它将用户的浏览器重新定向到在步骤 4 中启动工具时接收的“返回 URL”。HTTP 301 重新定向的消息应该发送到和工具启动时相同的窗口或框架。 8. 处理重定向。工具使用者接收重定向请求。 9. 呈递工具使用者的页面。工具使用者恢复对用户体验的控制。
备用路径	**A. 打开新窗口或 IFRAME** 基本流程处理的情况是在持有原始 LTI 链接的相同窗口打开工具。然而，链接也会在一个新窗口或 IFRAME 中被配置为打开。 **B. 禁用 JavaScript** 步骤 2 中传输的 HTML 表单应该包含一个嵌套在<noscript>标签中的提交按钮。因此，如果 JavaScript 被禁用，用户将能够通过提交按钮来手动提交请求而不用依赖于步骤 3 中的自动提交行为。 **C. 基本 LTI** 对于基本 LTI 链接，基本流程有一个微小变化。 ● 端点 URL 与基本 LTI 直接相关。与工具资料相关的工具代理和完整的 LTI 声明消息处理器（参见基本流程中的步骤 3）。然而，基本 LTI 链接并没有一个相关的工具资料。而每个基本 LTI 链接与链接被创建的消息处理器 URL 直接相关。 ● 安全性配置文件是不可转让的。基本 LTI 链接经常使用 lti_oauth_hash_message_security 配置文件。详细情况见[LTI ,12SEC]。 **D. 发送 Cookies** 当工具提供者响应步骤 5 的启动请求时，它也要发送 cookies。例如工具提供者会选择发送会话 cookie 给用户的浏览器。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

E. 交换用户资料工具使用者

大多的工具启动请求包含一个唯一标识用户的参数。工具提供者将不会知道此用户第一次尝试获取工具的任何事情。因此，在接到来自给定用户的第一个请求时，工具提供者可能需要阅读来自工具使用者的用户资料。这个交换可以通过调用由工具使用者服务器提供的 LIS 个人管理者的网络服务来完成。工具提供者通过检查在工具部署过程中接收的工具使用者资料来发现此网络服务的端点 URL。

另外，工具提供者可能会通过变量参数直接将用户属性嵌入到启动请求中。与个人实体相关的变量参数列表，参见[LTI, 12IMG]。

F. 交换课程部分的细节

如果工具在工具使用者系统中的特定课程部分被启动，启动请求会包含作为 context_id 参数部分的标识符。同样，工具提供者会从 LIS 个人管理者中读取用户资料，工具提供者也可以通过调用 LIS 课程部分的管理者来读取课程部分的详情。

另外，工具提供者可能会通过变量参数将用户属性直接嵌入到启动请求中。与个人实体相关的变量参数列表，参见[LTI ,12IMG]。

G. 直接返回到工具使用者

工具提供者的用户接口包含一个正常的 HTML 链接，它的 href 直接指向工具使用者系统中的返回 URL，而不是步骤 7 中的发送一个 HTTP 301 重定向。因此，当用户点击这样一个链接时，浏览器会直接导航到工具使用者系统。

H. 工具使用者关闭窗口

如果工具在一个新窗口打开，工具使用者要负责在处理步骤 8 的重定向之后要关闭窗口。工具使用者可能会选择在窗口中展示消息并提供允许用户手动关闭窗口的控制。或者工具使用者会自动关闭新窗口并且将焦点转移到“更开放”的窗口。

I. 突然关闭窗口

如果工具在一个新窗口被打开，用户很可能突然关闭那个窗口而没有在工具中正式地结束当前活动。也就是说，用户可能在基本流程的步骤 6 之前强行关闭了浏览器窗口。在这个案例中，工具使用者不会收到“返回 URL”的重定向。

J. 会话管理

当用户离开步骤 7 的工具提供者系统时，工具提供者选择保持用户的对话是存在的，期待用户可能会返回。如果用户没有返回，工具提供者通

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

	过一个合适的超时限制最终应该终止会话。LTI 版本 2.0 没有定义一个单一签署的协议。 另外，工具提供者可在他或她离开工具提供者系统之后会选择终止用户的会话。在这个案例里，用户会每个启动请求都获得一个新会话。

4 绑定与实现指南

4.1 HTML 表单消息绑定

工具提供者的消息处理器接收消息作为 HTTP 请求，该请求是由具有如下特点的 HTML 表单的提交而产生的：

- ‘enctype’属性有“application/x-www-form-urlencoded”值。
- 表单的‘action’属性是 LTI 消息的目标 URL。此 URL 是在工具资料中被定义为基本 URL 和适当“信息”元素的路径属性的组合。
- 方法（GET 或 POST）遵守被给定的消息类型所指定的公约。看每种消息类型的文档来确定适当的 HTTP 方法。
- 表单字段对应于 LTI 信息模型中的属性。然而 HTML 表单包含了一个输入字段的平铺序列，信息模型趋向于更加结构化。出现在 HTML 表单中的字段集被每一种消息类型显式地指定为每一种消息类型文档的一部分。在文档集中定义的字段名按照以下规定从信息模型中派生出来：
  1. 对于作为消息类的属性或它的 mixins 之一出现的每种简单的类型，用相同的名字添加一个隐藏的输入字段。
  2. 对于每一种复杂的类型，添加属性作为隐藏字段，每个隐藏字段的名称通过复杂属性的名称与复杂类型中的属性名称的连接而形成，由下划线分隔。例如，ContextMixin 类包含一个实体属性命名为“context”，并且 Context 实体反过来包含一个简单的属性命名为“id”，因此 HTML 表单中的字段名是“context_id”。
  3. 对于在工具配置文件中声明的每种适用的工具设置和每种适用的参数，添加一个字段，它的名称与用 custom 作为前缀的给定属性名相匹配。
  4. 添加消息类型的安全合同所规定的字段。

4.1.1 生成 HTML 页面的指南

当生成包含了返回到用户浏览器的启动请求的 HTML 页面时，此部分为工具使用者提供指南。

HTML 页面应该包含自动提交表单的 JavaScript 代码。

表单应该包括一个控制，例如一个按钮，以便于如果 JavaScript 被禁用，则用户可以手动提交表单。

图 4.1 说明了这些指南。

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

```
<form
  action="URL_FOR_MESSAGE_HANDLER_AS_DEFINED_IN_TOOL_PROFILE"
  name="ltiLaunchForm"
  method="post"
  enctype="application/x-www-form-urlencoded">

<!-security parameters -->
<input type="hidden" name="oauth_signature"
value="Xddn2A2BjzgIYkvigaK"/>
<input type="hidden" name="oauth_signature_method" value="HMAC-
SHA1"/>
<input type="hidden" name="oauth_timestamp" value="1244834250"/>
<input type="hidden" name="oauth_nonce"    value="43589328293941"/>
<input type="hidden" name="oauth_version"   value="1.0"/>
<input type="hidden" name="oauth_callback"  value="about:blank"/>

<!-lti message parameters -->
<input type="hidden" name="some_lti_message_attribute"
value="some_value"/>
…
<noscript>
<input type="submit" value="Press to launch external tool"/>
</noscript>
</form>

<script language="javascript">
   document.ltiLaunchForm.submit();
</script>
```

Figure 4.1 Sample HTML showing the structure of an HTML Form Binding for an LTI message

注意，表单包含了一个消息参数集和一个安全参数集。在工具部署过程中被工具使用者和工具提供者一致认可的安全性配置文件规定了必须存在的特定安全参数集。这里所示的例子中，oauth 安全性配置文件规定了安全参数。

大多数安全性配置文件包含一个数字签名或者消息身份验证代码（MAC）。数字签名是由所有的 POST 参数连接形成的散列字符串。对于 oauth 安全性配置文件，签名是由第八行所示的 oauth_signature 参数提供的。

4.1.2 通过 POST 自动提交 HTML 表单的好处

通过 HTTP POST 方法使用一个自动提交的 HTML 表单有超过 HTTP 重定向的以下好处：

- 包含在 HTTP POST 中的数据量是无限的，而 HTTP 重定向中的数据量要受到特定浏览器对 URL 长度的限制。
- 在 HTTP 请求的 POST 体中发送 LTI 消息而不是 URL 编码的查询字符串限制 LTI 消息属性的公开。例如，可能被传输为消息属性的个人身份信息将在使用 HTTPS 协议的 POST 中不可见。

4.1.3 非 ASCII 字符

如 http://www.w3.org/TR/html40/appendix/notes.html#non-ascii-chars 中所描述的，应避开参数名或值中的非 ASCII 字符。

(编译自：IMS Learning Tools Interoperability (LTI) Messaging Framework Version 2.0 Public Draft.

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。

http://www.imsglobal.org/lti/v2p0pd/ltiMSFv2p0pd.html. [2012-11-01])

（王妍编译，岳增慧校对）

本作品采用知识共享署名-非商业性使用-禁止演绎 2.5 中国大陆许可协议进行许可。