2013 年第 9 期（总第 24 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 11 月

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。

目　录

【标准规范报道】

1、NISO 发布信息标准在开放获取方面的数据管理主题季刊

2013 年 10 月 17 日，国家信息标准组织（NISO）宣布 2013 信息标准秋季刊（ISQ）的发布，其特定主题是数据管理。由于许多政府和资助机构已命令公共资助的研究必须使研究开放（不仅包括在期刊论文中发布的结果，也包括隐含数据），人们对数据管理的兴趣已大大提高。因此，围绕着确保数据能够进行重用、长期保存的流程和工具正在进行许多讨论和工作。

“如果我能够总结一次又一次出现的主题，不仅是本刊的文章中，实现数字管理的标准很有必要，”英国大气数据中心的研究科学家和项目经理、刊物特邀内容编辑 Sarah Callaghan 说，“哪种类型的数据被管理根本不重要，比如，有关研究项目、出版物、灰色文献的元数据，实验室工作的方法与结果、或长期观察任务的测度等。有一件事是肯定的，创建数据的速度大大增加，数据管理的唯一途径是使其自动化，而且，实现它唯一的方法是有标准化的结构和本体。”

由 Colin L. Bire, Cerys Willoughby, Simon J. Coles 和 Jeremy G. Frey 写的专题文章，讨论了在化学科学中的数据管理问题，特别指出化学家每天在实验室和计算机的活动中都重视管理的重要性。作者强调，管理过程中的基本要素是元数据，特别是在数据和信息被创建的时候，这被表述为“管理源头。”

三个“在实践中”的文章提供了在欧洲学术团体中数据如何被管理的案例研究，特别是在考古学和地球科学领域。Jochen Schirrwagen 和合著者描述了在 OpenAIRE 学术交流基础设施、欧盟倡议——开放获取基础设施（获取欧洲资助项目的研究成果和机构、学科知识库网络中的开放获取内容）中的数据管理。Ray Moore 和 Tim Evans 讨论了保存灰色文献爆炸：考古数据服务（ADS）中的 PDF/A 和数字档案，其特别强调使用 PDF 档案格式作为保存标准的利弊。Esther Conway 和她的合著者分析了通过数据管理和保存，确保地球科学数据长期影响方面的挑战，因为大部分的地球科学数据发生于自然现象，且是无法复制的。他们指出保存这些数据用于某些领域的社会利益， 比如灾难管理、人类健康、可持续能源、气候变化、水质量和可用性、生态系统保护和农业管理。

“NISO 逐渐地参与到与数据管理相关的讨论和工作中，”NISO 执行董事 Todd Carpenter 说，“正如本刊中文章展示的那样，标准最初被开发，是为了管理电子期刊，例如数字对象标识符和 PDF/A，现在也被应用到了数据中。文章也指出了标准工作仍然需要的许多领域，例如数据管理、元数据、保存格式、指标等。”

信息标准季刊在NISO网站上以电子格式开放获取。ISQ的整个 2013 秋季数据管理看和单独文章均可免费下载。获取免费电子版本，请访

问：www.niso.org/publications/isq/2013/v25no3/。

关于信息标准季刊

信息标准季刊（ISQ）是NISO的印本和电子杂志，目的是交流在图书馆、出版和信息技术，特别是三个领域重合方面的技术（基于标准）和最佳实践。ISQ报导积极的进展以及能够潜在地显示可复制成果的实施、案例研究和最佳实践。更多信息，请访

问：www.niso.org/publications/isq/

关于 NISO

NISO促进标准的开发与维护，该标准有利于信息的创建、持久管理与有效交互，以便于它能够被信任地用于研究与学习。为了完成这个任务，NISO聚集了图书馆、出版商、信息聚合者以及其他组织，它们通过知识的创建、组织和管理来支持学习、研究和学术成就。NISO与交叉的兴趣团体共同工作，并跨越信息标准的整个生命周期。NISO是一个非盈利性协会，受到美国国家信息协会（ANSI）的认可。关于NISO的更多信息在网站：www.niso.org上。

（编译自：NISO Publishes Data Curation Themed Issue of Information Standards Quarterly in Open Accesshttp://www.niso.org/news/pr/view?item_key=dd448efd0f210acab8b1c6dce1efda94d0347aaf [2013-10-17]）

（王妍编译，吴贝贝校对）

2、NISO 和 UKSG 发布知识库和相关工具（KBART）公共评论的推荐修订草案

2013 年 9 月 5 日，国家信息标准组织和 UKSG 宣布发布一个知识库和相关工具（KBART）推荐实践修订版的公共评论草案。2010 年发布的最初推荐实践为信息供应链中的各方提供了有关数据格式的直接指导，目的是确保内容提供商和知识库开发者之间准确的元数据交换。以最初的推荐为基础，修订草案聚焦于在元数据供应中引起难题的更加细粒度的、复杂的问题，包括特定联盟元数据传输、开放获取出版物的元数据传输，以及电子书和会议文集的元数据传输。

“自从首个推荐实践发布后，超过 50 个出版商和内容提供商已经认可了 KBART，并表明他们的承诺——提供质量好的元数据，”Jisc Collections 的数据经理、KBART 工作组的联合主席 Magaly Bascones 声明说。“批准过程需要提交一个示例文件和 KBART 工作组的验证。有了这个批准就能够给用户一个保证，即提供商的元数据是可信的，而且具有所需的粒度级别，而免去了逐标题检查的繁重任务。”

“认可出版商的经历以及来自图书馆和联盟的调查反馈标识了这个扩展 KBART 版本所关注的重点领域，数字集合和数字资源图书馆员主任、KBART 工作组联合主席 Chad Hutchens 说。“在公共评论期之后，KBART 工作组将做出任何需要的修订，并完成推荐出版。在 NISO

网站的 KBART 信息中心也是可用的，该网站提供了关于 KBART 的支持材料，包括 KBART 术语表、认证信息、知识库供应链联系人的注册表以及 OpenURL 和知识库的背景信息。”

“在 KBART 二期推荐实践出版之后，项目将会转移到 NISO 中的常委会状态，”NISO 项目副主任 Nettie Lagace 声明说。“这个委员会将负责管理认证过程，提供持续 KBART 教育与推广，并维护信息中心”。

KBART的二期草案将在 2013 年 10 月 4 日起开放，进行公共评论。要下载草案或提交在线评论的人，请访问KBART信息中心：www.niso.org/workrooms/kbart。

关于 UKSG

UKSG 存在的目的是连接信息群体并鼓励有关学术交流的思想交换。它跨越了扩展的学术信息群体（图书馆员、出版商、中介机构和技术供应商）广泛的兴趣爱好以及活动。在动态环境中，UKSG 的目的是：

- 促进群体集成、网络、讨论和交换思想
- 提高学术信息部门的成员知识并支持技能发展
- 刺激研究和合作倡议，鼓励创新并促进良好实践的标准。
- 传播新闻、信息和出版物，并提高支持学术信息部门的服务意识。

（编译自：NISO and UKSG Release Draft Revised Recommendations for Knowledge Bases and Related Tools (KBART) for Public Comment http://www.niso.org/news/pr/view?item_key=cdf0874485c7edadb85e38ceb0fb2a393de44d6b [2013-09-05]）

（王妍编译，吴贝贝校对）

3、NISO 开放获取元数据与指标

开放获取元数据与指标项目将会开发标准化的书目元数据与直观的指标来表述期刊论文的可获取性，并潜在地描述项目是如何“开放”的。许多内容都可以在开放获取（OA）、增强获取、公共获取或其他描述的旗帜下从出版商处获得；在某些情况下，出版商和基于作者资助机构的出版商之间提供的条款有所不同。再加上许多潜在的误解，大量的出版商也提供复合选项，即一些文章是“开放”的，而期刊内容的剩余部分仅通过订阅或许可来获得。目前，标准化的书目元数据没有提供有关一篇特定文章是否免费可读或读者能使用哪些复用权限等方面的信息。指示一篇文章开放性的直观指标或图标在出版商或甚至是来自同一出版商的期刊在设计和使用方面是不一致的。

该项目最初关注于描述与一篇OA文章相联系的读者权利的元数据元素。具体来说，NISO工作组将确定最佳机制来描述和传递权利，如果有的话，一个任意的用户必须从任何因特网连接点获得一篇特定的文章。建议将包含一个机读形式元数据分布与集成的方法。工作组也将考虑融入有关复用权利的可行性以及达成数据传输协议的可行性。

（编译自：NISO Open Access Metadata and Indicators http://www.niso.org/workrooms/oami/[2013-11-05]）

（王妍编译，吴贝贝校对）

4、开放获取词表

有很多不同的利益相关者对开放获取学术交流有兴趣。同时，开放获取生态系统有很多不同的信息标准、词表和“规范列表”。这个项目将这两者结合起来，促进共享的理解、共识和合作。

背景：

对资助者、出版商、研究者和其他人来说为开放获取研究产出开发一个一致的元数据方案有越来越明显的重要价值。

这样一个方案的应用案例在转移到一个更开放的学术出版的节奏中变得越来越重要。这样的应用案例包括了解获取和复用学术产出水平的能力，通过一个知识库的 OpenURL 解析器将用户传递给一个合适的复件；通过搜索和目前的认知服务对资源进行排名。

同样地，由于影响、监管 OA 政策以及审计目的，对资助者和研究者也需要能够追踪、管理和查询开放学术产出。

已经有很多重要的项目和行动来指导这个项目，特别地有 PEER project，以及 OpenAire 和 DRIVER guidelines。

这项工作是 Open Access Implementation Group (OAIG)的直接成果，为各种各样的利益相关者的合作标识了关键目标。在 OAIG 代表和出版商贸易个体的会议中达成了一个优先项，即标识元数据领域/标准，这使 HEIs、资助者和出版商能揭示人机可读的数据，从而达到开放获取的强制规范和一般性的开放自动聚集。

目的：

工作的目标是建立一个持续发展的进程，在标准词表被用于 OA 情境的特殊环境方面达成一致，创建一个网络资源作为这些一致的中心点。

目标：

- 利益相关者在词表应该被用在哪些情境或环境中取得共识。
- 与发展中的 RIOXX 指导方针相协调。
- 建立一个持续发展的资源作为利益相关者进一步合作的焦点。

成果：

- 在这个领域中当前的实践和新兴工作的回顾。
- 在 OA 学术交流空间中关于合适的词表、应用程序配置文件和规范文档的推荐。
- 在参考和管理方面建立一个利益相关者小组。

- 一个广泛的兴趣社区。
- 一个设计良好的网络呈现，有足够的资源允许成果的明确交流和持续参与的路线。

（编译自：Vocabularies for Open Access (V4OA http://www.jisc.ac.uk/whatwedo/topics/digitallibraries/pals-group/v4oa.aspx [2013-11-05]）

（吴贝贝编译，王妍校对）

【标准规范推介】

一、关联数据平台应用案例和需求

1. 范围和动力

Tim Berners-Lee 定义了关联数据的原则：

1 使用 URI 作为事物的标识。

2 使用 HTTP URIs，这样人们就可以定位并查找（解引用）这些名称。

3 当 URI 被解引用时，提供资源相关的有用信息。

4 在发现的数据中包含链向其它相关 URI 的链接，使人们发现更多的信息。

这四条原则在指导和激励人们在网上发布关联数据时被证实是非常有效的。数据的数量，尤其是可以在网络中获取的开放数据的数量增长迅速，使用这些数据作为结果已经创造出了大量的具有创造性的并且有用的“mashups”。

关联数据平台的目标是定义一个所需的规范来允许一个可写的关联数据 API 的定义，这个 API 与现在使用 Atom 发布协议（Atom Publishing Protocol，APP）在网络上写的简单应用程序 APIs 是相同的。APP 与关联数据拥有一些共同的特点，如使用 HTTP 和 URLs，但依赖基于 RDF 的灵活数据模型，该模型允许多样表达。

2. 这个文档的组织

这个文档组织如下：

- **用户描述**从一个用户或应用程序的角度获取关于系统需求的声明。
- **应用案例**用来获取和建模功能性需求。应用案例描述不同状况下系统的行为，为系统做了什么、出于什么目的做编目，但不考虑系统设计或实现。
- **应用场景**用来为一个应用案例的功能性需求做模型，关注于发挥作用的一个应用案例。应用场景范围从轻量的陈述到正式定义的互动图解。
- **需求**列出功能性的和非功能性的或质量需求，还有它们源自的应用案例。

3. 用户描述

3.1 维护社会联系信息

我们很多人拥有多个电子邮箱账户，这些账户包含我们与之互动的个人和组织信息---名称、邮箱地址、电话号码、即时信息身份等。当某人的电子邮箱地址或电话号码改变时（或他们获取新的邮箱地址或号码时），如果我们能够在一个地点更新这个信息并且它所有的复件都能被自动更新，那么我们的生活将更加简便。换句话说，那些复件都被关联到“the contact”的定义。维护一个“contract”的本地复件可能也有好的理由（如离线邮箱寻址），但理想状态下，任何复件仍然被关联到某个中心“master”。

然而，仅同意一个“contact”的格式是不够的。即使所有的电子邮箱供应商都同意使用同一种联系的格式，我们仍然需要使用每一个提供商的交互界面来更新或取代提供商的复件，或者我们不得不赞同一种方式，每一个电子邮箱提供者都关联到“master”。如果我们跳出我们自己的个人兴趣向外看，如果人或机构透露他们自己的联系信息使我们能关联它的话，这将会更加有用。

在任一个案例中发挥作用的是对资源的一个普遍意义上的理解，所需的一些格式和这些资源的获取指南。这个将支持如何获取一个联系的链接、如何使用那些链接来与一个联系（包括阅读、更新和删除）互动、以及如何能够简单地创造一个新的联系并添加到我的联系中、何时删除一个联系、如何将它从我的联系列表中移除。能够添加一些关于联系的特定应用程序数据（原始设计没考虑到的）也是很好的。理想状态下，我们想要消除联系的多个复件，可能还会有关于我的联系的其它有价值信息，它们被存储在不同的服务器上，需要一种简单的方法将这个信息关联到我的联系。不管一个联系集是不是我自己的，是否被一个机构共享，还是一个电子邮箱提供商所知的所有联系，如果它们都以同一种方式工作，那将是很好的。

3.2 跟踪个人和商业关系

在我们每天生活中，我们在很多不同的关系中处理很多不同的组织，他们每一个都有关于我们的数据。然而，他们中的任何一个都不可能拥有我们的所有信息。他们常常给我们获取信息的途径，很多都是通过我们使用一些字符串——一个账号、用户 ID 等唯一标识身份的网站获取。我们不得不使用他们的应用程序来与关于我们的数据交互，并且我们不得不使用他们给我们的标识符。

那些数据中网络可寻址的部分可以被持续关联，而不用维护不同格式的各类标识符，也不用手动地为每一个通讯的应用程序提供标识符，我们可以建立一个书签集合，这不是更简便吗？当我们想要检查或交换他们的内容时，拥有一个单独的、持续的、他们都支持的应用程序界面不是更简便吗？

由任何单独机构所持有的信息可能是简单数据和其它数据集的混杂，例如，一个银行账户和一个历史交易的集合，我们的银行可以很容易地拥有一个账户集，这个账户集是每一个客户成员的集合。

3.3 系统和软件开发工具集成

系统和软件开发工具通常来自供应商的一个不同集合，并建立在不同的基础构架和技术

之上。这些工具被构建的目的是用于满足一个特定值域场景的需求。通常，工具提供商将与其它工具的集成行为视为不道德的，而不是为他们的终端用户提供附加价值。一个事后想法更多的是这些工具的数据如何集成并联系到合作的和外部的应用程序。这个问题可以通过对小部分工具进行标准化来分解，但这是很少发生的，如果发生也是在小组织内发生。伴随着这些组织在规模和复杂程度上的增长，他们需要与外包发展和多样化的内部组织联合起来开发他们自己的工具。这需要对更完整的商业流程有一个更好的支持。这个需求已存在很多年了，工具提供商也尝试了很多不同架构的方法来解决这个问题：

- 为每一个应用程序执行一个 API，然后，在每一个应用程序中，执行“glue code”，即利用其它应用程序的 API 来将它们链接在一起。
- 设计一个单独的数据库来存储多种多样的应用程序数据，执行反对这个数据库的每个应用程序。
- 执行一个中心“hub”或“bus”，通过利用之前描述的 APIs 来编制更广的商业流程。

尽管每一个方法有它的拥护者并获得一些成功，但是它们都不能完全地令人满意。使用关联数据作为应用程序集成是一个强烈的呼吁。

3.4 图书馆关联数据

W3C 图书馆关联数据工作小组在他们的应用案例报告中拥有大量的应用案例。这些应用案例关注于从全然不同的来源中抽取和关联图书馆数据的需求。这些应用案例的变体可以提供一致的格式还有改进或更新数据的方法，这促使使用简化的方法来有效地分享这些数据，同时产生增量更新。

“数字对象簇”包含很多相关的应用案例：

- 分组
- 强化
- 浏览
- 复用

集合簇也包含很多相关的应用案例：

- 集成层级的描述
- 集合发现
- 社区信息服务

3.5 自治运转监测

跨越不同的城市、乡镇、农村和不同的自治区，有大量的由自治区管理和运行的服务，它们产生和使用大量的信息。这些信息用于监测服务、预测 难题和处理物流。为了有效地收集、生产和分析这些数据，需要一个宽松的标准数据源的基础集合。需要一个简单的、低成本的方法来从监测服务的不同集合中揭示数据，这个方法可以简单地集成到其它系统的自治区。所有服务都有链接，都依赖于其它数据和服务，所以拥有一个简单的可扩展的关联模型

是关键。

3.6 医疗卫生

医生分析、诊断和建议治疗需要大量复杂的、变化的并增长的知识。这个知识需要来自大量的来源，包括医生自己的专业知识、与其他医疗专家的网络咨询、公共卫生来源、食品和药物监管机构，还有其它的医疗研究和推荐仓储。

为了诊断一个病人的状况需要有关病人药物和医疗历史的当前数据。除此之外，最近的关于药物的配药建议要关联到病人的数据。如果病人经历了来自药物的不同作用，这些医生需要将信息发布到一个合适的监管源。其他医疗专家需要获取这个药物已经被确认的和刚出现的作用。同样地，如果围绕一次疾病暴发有地理格局，需要人们了解新症状和疗法，那么这些信息需要很快地到达一个分布式的、多样的医疗信息系统集合。并且，根据疾病暴发的新病发者发给这些监管机构的报告，包括症状和起因的其它信息，对未来事件发明最有效的疗法是非常重要的。

3.7 广播节目中的元数据强化

当广播员对元数据强化产生兴趣时，有很多不同的应用案例：

- 通过关联事实、事件、地点和人物来丰富档案或新闻元数据。
- 由自动抽取工具如人物标识符产生的丰富元数据。
- 在分类体系或列表中丰富术语的定义。

这个支持更有效的信息管理和数据/内容挖掘。

然而，需要解决方法来促进与其它数据源的链接，处理例如发现、自动化、消除歧义等方面的问题。广播员面临的其它问题是关联数据的编辑质量，它的持续性和使用权利。

3.8 基础架构数据的聚合和混搭

对基础架构管理来说，提供环境是重要的，在这个环境中来自不同源的信息能够有效地被聚合、过滤、和可视化。特别地，以下应用案例应该被纳入考虑：

- 当一些数据源是基于关联数据，而其它的不基于时，聚合和混搭应该跨越不同的来源。
- 数据源的消费者和聚合/过滤的数据流不需要执行关联数据本身。
- 这个场景的简单版本是基于拉式的，即数据从数据源被请求。在更先进的设置里，基础架构没有一个主要的改变，它应该可能移到一个推式的互动模型。

在这个场景中，关键的因素是有允许简单聚合和过滤的抽象概念，独立于来源（被结合）的内在数据模型，可以被用于拉式的交互和推式的交互。

3.9 在低端设备中分享 RDF 数据的有效负荷

围绕降低语义网规模观点的一些项目需要跨越一个给定网络节点成员运输 RDF 的有效负荷。转换在一个强制性的情境下被完成。在一个 P2P 类型里，每一个节点有能力担任一个数据消费者或一个数据提供者，服务它自己的数据或作为一个接力传递给其它数据。

RDF 数据一个任意有效负荷的转换通过一个容器机制来实现，增加或移除 RDF 三元组的集合。目前，SemanticXO 项目使用命名的图表和图表存储协议来跨节点创建/删除/复制图表。不幸的是，三元组存储非常需要软件，而它在有限的硬件上不是经常可用的。一些支持轻量栏目存储的 REST-like 交互会是一个更好的办法。

3.10 共享二进制资源和元数据

当发布关于恒星的数据集时，一个人可能想要发布那些恒星图片的链接，这也可能需要他们自己发布图片。

如果关联数据包含以非 RDF 格式描述的资源信息时，这些非 RDF 格式应该像发布到关联数据服务器那样简单地来发布 RDF 关系。一个关联数据服务器应该允许非关联数据资源的发布，使发布和编辑那些资源的元数据变得简单。

资源来自两个部分——图片和关于图片的信息。关于图片的信息是重要的。它是一个图片数据和其它数据的混合项目。

3.11 数据目录

Asset Description Metadata Schema（ADMS）提供数据模型来描述语义资产仓储内容，但是这给构建这些仓储的联盟（目的是服务资产复用的需求）带来了挑战。这些包括获取和查询个人仓储和高效地检索更新的内容而不用检索全部内容。数据仓储集成方法允许我们处理来源技术的异质性并从最优化性能中收益。有了数据仓储，联盟需要：

- 理解数据
- 从不同的仓储中无缝地交换语义资产
- 保持自身更新

仓储拥有者可以为他们的仓储描述进行解除引用 URIs 的维护，以一个关联数据兼容的方式包含资产。ADMS 提供了必要的数据模型来实现有意义的数据交换。然而，这给有效获取没有完全编址的数据带来了挑战。

3.12 受约束的设备和网络

来自资源受限设备的信息在很多领域被标识为一个主要的驱动力，从智能城市到环境监测到实时追踪。由这些设备产生的信息的数量正迅速增长，需要以一种系统的、标准化的、低成本的方式获取和聚合。通过在网络中使用相同的标准，与应用程序的聚合将被简化并在资源受限的设备间进行更高水平的交互。即将出现的 IoT/WoT 标准如‘6LowPAN’——对资源受限设备的 IPV6——和受限应用协议（COAP）在 UDP 的顶端提供了一个降低规模的 HTTP 版本来使用受限设备，已经进入了成熟的状态。下一步是在资源受限设备上支持 RESTful 界面，依附于关联数据原则。由于受限资源可以在设备和网络中得到利用，一个基于 SPARQL Update 的解决方法在目前不被认为是有效的或可行的。一个基于 HTTP-CoAP 映射的方法可以使受限设备直接加入到一个基于关联数据的环境中。

3.13 支持科学流程的服务

现在很多科学领域包含生物信息学密集数据方法的分支，例如生物信息学、天文学。为了支持这些新的方法，我们期待超越由科学工作流系统提供的既定平台，通过本地可信的分享、分析、传播和复用来实现实验记录的获取、帮助和保存完整的生命周期。

- 聚合：特别地，研究对象（ROs）在服务和客户端之间交换。
- 我们需要轻量的可以被简单地聚合、跨软件和数据范围的服务。

  基础服务为了存储、修改、探索和复用而收集和揭示 ROs。

  给 ROs 提供附加值的服务。

3.14 项目成员信息

关于人员和项目的信息伴随着角色、组织和联系信息的改变而改变。找到一个项目目前的状态对于人们联系到正确角色的正确人物很重要。对于回溯在过去谁扮演了什么角色也是有用的。

关联数据平台的一个应用可以把管理这些信息的责任交给项目组本身，而不需要来自一个中心网站管理者要求的更新。

这个可以通过以下实现：

- 针对每个人和项目的资源描述
- 在项目中描述角色/成员的一个容器资源

为了获得项目的历史，一个资源的旧版本，包括容器资源，应当被保存，那样如果有需要处理特定的项目，也可以有一个“current”的注释。

获取信息有两个方面：

- 获取“current”的状态，而不管资源描述的版本。
- 获取历史状态，通过获取资源描述的特定版本。

3.15 云基础架构管理

云操作者通过提供 API 支持来帮助消费者远程获取云基础架构管理（IaaS）。基础架构包括系统、计算机、网络等。整个结构可以被看作是有层级的，跨链接被执行。

IaaS 场景对生命周期管理和发现做出特定的需求，同时处理非即时的变化、历史获取和查询：

- 与生命周期相关联的云基础架构的众多方面。
- 通常，获取一个新的生命周期状态是不及时的。客户端需要一个通用的机制来监测这些变化。
- 在一种资源的生命周期中检索所有事件的设备是有用的。
- 查询提供了在 API 后询问资源的一种方式，同时找到应用程序新的检索入口。

基础架构管理可以被看作资源潜在图表的操作。

4．应用案例

以下的应用案例来自上述的一个或多个用户描述。这些使用案例通过场景的开发被详细

的探索，每一个通过一个单一用户描述的关键点而被促进。它们包含的例子单纯地作为描绘性的目的，不应该被规范性地解释。

4.1 UC1:组成资源

大量的用户描述介绍了容器在应用程序情境下作为构成资源机制的一种想法。一个组成可以在它自己的权利中被 URI 标识。它的属性可以表达应用程序的功能可见性，让客户端决定它们可以执行的其它操作。这些操作可能包括特定服务应用程序的描述。

4.1.1 基本的场景：创建一个容器

在 LDP 服务器内创建一个新的容器资源。在支撑科学流程的服务中，研究对象在语义上是资源的丰富聚合。创建一个基本的工作流研究对象来聚合科学的工作流和它们的产物。这些产物通过项目的项目生命周期被添加到研究对象中。

下面的 RDF 描述采集了研究对象最初的状态。为了达到例子的目的，我们包含了创建的时间。它是通过以下 RDF 可以检索到的 URI 来处理一个关联数据资源。不相关的 URL<>应当被理解为对研究对象的自我参考。

EXAMPLE 1

```
@prefix ro:  <http://purl.org/wf4ever/ro#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix ore: <http://www.openarchives.org/ore/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<> a ro:ResearchObject, ore:Aggregation ;
    dct:created "2012-12-01"^^xsd:dateTime .
```

（见功能需求 F1.1）

4.1.2 可选择的场景：创建一个嵌套容器

嵌套容器的动力来自系统和软件开发工具聚合用户描述。OSLC 改变管理词表允许错误报告来拥有被成员谓语 oslc_cm:attachment 参考的联系。这个可以被看做嵌套容器。高层级的容器包含问题，每一个问题是联系的一个容器本身。在例子中，issue1234 是容器 top-level-container 的一个成员。反过来 attachment324 和 attachment251 在 issue1234 中是相联系的。把这些当作容器会使让更加简单地管理它们的自我控制单元。

EXAMPLE 2

```
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix oslc_cm: <http://open-services.net/ns/cm#>.
@prefix : <http://example.org/>.

:top-level-container rdfs:member :issue1234 .

:issue1234 a oslc_cm:ChangeRequest;
      dcterms:identifier "1234";
      dcterms:type "a bug";
      oslc_cm:attachments :attachments.

:attachments a oslc_cm:AttachmentList;
      oslc_cm:attachment :attachment324, :attachment251.
```

（见功能需求 F1.2）

4.1.3 可选择的场景：删除一个容器

如果一个容器可以被删除，那么任何受控资源和嵌套容器也应该能被删除才是正常的。（见功能需求 F1.3）

4.2 UC2：管理资源生命周期

这个应用案例处理一个资源被管理的生命周期，与资源所有者有关。管理资源的责任属于他们的容器。

- NF2.1：资源的非副本：“删除多重副本”，从维护社会联系信息的一个单独地方表达资源。
- NF2.2：资源的分配：关联数据可以存储在维护社会联系信息中的分离服务器中。
- NF2.3：一致的、全球的命名：资源应当被一致地关联…而不是以不同的格式来维护不同的标识符。

4.2.1 基本场景：创建资源

资源通过在一个容器中被创建而开始生命。从用户描述——维护社会联系信息中，“容易地创建一个新的联系并把它加入到我的联系”应该是可能的。这表明了资源创建紧密地跟应用程序情境关联。新资源在表达“我的联系”的容器内被创建。资源的生命周期被关联到容器的生命周期。

联系细节通过 RDF 描述和它的属性获取。描述可能包括非标准的 RDF，“原始设计没有考虑到我的联系的数据”。以下 RDF 使用 FOAF 词表来描述联系信息。一个联系用 foaf:PersonalProfileDocument 表达，定义可以被创建并更新为一个独立单元的资源，即使它可以描述附属资源，例如 foaf:Person。

EXAMPLE 3

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

<> a foaf:PersonalProfileDocument;
        foaf:PrimaryTopic [
                a foaf:Person;
                foaf:name "Timothy Berners-Lee";
                foaf:title "Sir";
                foaf:firstName "Timothy";
                foaf:surname "Berners-Lee";
                foaf:nick "TimBL", "timbl";
                foaf:homepage <http://www.w3.org/People/Berners-Lee/>;
                foaf:weblog <http://dig.csail.mit.edu/breadcrumbs/blog/4>;
                foaf:mbox <mailto:timbl@w3.org>;
                foaf:workplaceHomepage <http://www.w3.org/>.
        ]
```

（见功能需求 F2.1）

4.2.2 可选场景：删除资源

删除一个资源和它所有的属性。如果资源属于一个容器，它将被从那个容器中移除，然而其它的到被删除资源的链接可能被留下。在资源是容器的案例里，服务器也可以删除任何或所有被包含的资源。在一般性的实践中，一个被删除的资源不能恢复。然而，在受限的取消删除的边缘情况中，资源恢复是可以的。最佳实践表明"Cool URIs don't change"，暗示被删除的 URIs 不应该被回收利用。

（见功能需求 F2.2）

4.2.3 可选场景：移动包含的资源

在一个容器中，资源有超越他们成员的生命之外的价值。这表明了增加参考来修改容器成员的方法。所有者的变化可能或不可能表明 URI 的变化。既然不鼓励分配一个新的 URI 到一个资源，那么指明一个资源伴随一个合适的 HTTP 响应而移动是可能的。

4.3 UC3：检索资源描述

获取一个资源当前的描述，包括那个资源的属性和到相关资源的链接。表达可能包括相关资源（不能被直接获取）的描述。根据应用程序，一个服务器可能通过额外的三元组丰富被检索的 RDF。例子包括增加引入的链接，owl:sameAs 闭合和 rdf:type 闭合。HTTP 响应应当包含版本控制信息，以便接下来的更新可以确保它们被应用到正确的版本中。

NF3.1：使用一个合适的标准词汇使来自维护社会联系信息的一个对资源的通用理解成为可能。

NF3.2：一个来自“自治运转监测“的可扩展关联模型是关键。

4.3.1 基本场景：检索资源描述

用户故事 Project Membership Information 讨论了关于人和项目信息的表达。它要求对每一个人和项目的资源描述，允许项目组审查关于这些资源的信息。下面的例子解释了关

于基于 Epimorphics 组织本体的组织框架的各种信息。

例子 4 和 5 定义了基址在 http://example.com/被当作 LDP 服务器的两种资源。例子 4 的表达描述 http://example.com/member1，例子 5 的表达描述 http://example.com/role。一个读取例子 4 的客户端将单独地检索例子 5，目的 是获取角色信息，比如它的描述性标签。

注意，这些资源的表达可以包含相关资源的表达，如 http://www.w3.org/。

EXAMPLE 4

```
@prefix org: <http://www.w3.org/ns/org#> .
@prefix owltime: <http://www.w3.org/2006/time> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@base <http://example.com/> .

<member1> a org:Membership ;
        org:member <http://www.w3.org/People/Berners-Lee/card#i> ;
        org:organization http://www.w3.org/> ;
        org:role <director> ;
        org:memberDuring [a owltime:Interval; owltime:hasBeginning [
                owltime:inXSDDateTime "1994-10-01T00:00:00Z"^^xsd:dateTime]] .

<http://www.w3.org/> a org:FormalOrganization ;
        skos:prefLabel "The World Wide Web Consortium"@en ;
        skos:altLabel "W3C" .
```

EXAMPLE 5

```
@prefix org: <http://www.w3.org/ns/org#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://example.com/> .

<director> a org:Role ;
        rdfs:label "Director" .
```

（见功能需求 F3.1）

4.3.2 可选场景：检索一个非文档资源（hash URI）的描述

在很多案例中，有趣的事件并不总是可解析的事件。下面的例子验证了一个 FOAF 配置文件如何被用来区分人物和文件，前一个是后一个的主题。在片段被定义与基础相关的例子中，包含片段的 URL 可以用来获取表达包含文件的资源。HTTP 协议要求片段部分在从服务器请求一个 URI 之前被剥去。客户端可以从被检索的描述中读取 hash URI<#i>的属性。

EXAMPLE 6

EXAMPLE 6

```
@base <http://www.w3.org/People/Berners-Lee/card>
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix dc: <http://purl.org/dc/elements/1.1/>.

<> a foaf:PersonalProfileDocument ;
        dc:title "Tim Berners-Lee's FOAF file" ;
        foaf:homepage <http://www.w3.org/People/Berners-Lee/> ;
        foaf:primaryTopic <#i> .
```

（见功能需求 F3.2）

4.4 UC4：更新现存的资源

改变一个 LDP 资源的 RDF 描述，潜在地移除或重写现存的数据。这个允许应用程序通过添加额外的到其它资源的链接来丰富一个资源的表达。

NF4.1：不受限制的词表

4.4.1 基本场景：强化

这个与用户描述 Metadata Enrichment in Broadcasting（广播节目中的元数据强化）有关，且它基于 BBC 体育本体。关联数据的以资源为中心的观点提供了替换或重写一个资源和它的数据的一个自然粒度。最简单的更新将用一个新的表达来取代目前了解一个资源的内容。

在下面的例子中有两种明显的资源：一个体育事件和一个相关的奖励。资源的粒度将允许一个用户替换关于奖励的信息而不干扰事件的信息。

EXAMPLE 7

```
@prefix : <http://example.com/>.
@prefix sport: <http://www.bbc.co.uk/ontologies/sport/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

 :mens_sprint a sport:MultiStageCompetition;
    rdfs:label "Men's Sprint";
    sport:award <#gold_medal> .

<#gold_medal> a sport:Award .
```

描述可以被丰富为事件发展，通过用下面的替换上面的描述增加一个到金牌获得者的链接。

EXAMPLE 8

```
@prefix : <http://example.com/>.
@prefix sport: <http://www.bbc.co.uk/ontologies/sport/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

 :mens_sprint a sport:MultiStageCompetition;
    rdfs:label "Men's Sprint";
    sport:award <#gold_medal> .
<#gold_medal> a sport:Award;
    sport:awarded_to [
        a foaf:Agent ;
        foaf:name "Chris Hoy" .
    ] .
```

（见功能需求 F4.1）

4.4.2 可选场景：一个资源的选择性更新

这个与用户描述 Data Catalogs（数据目录）有关，一个目录通过下面的 RDF 模型进行描述，它基于数据目录词表（提供表达元数据的标准格式）。

EXAMPLE 9

```
@prefix : <http://example.com/>.
@prefix dcat: <http://www.w3.org/ns/dcat#> .
@prefix dcterms: <http://purl.org/dc/terms/> .

 :catalog a dcat:Catalog ;
    dcat:dataset :dataset/001;
    dcterms:issued "2012-12-11"^^xsd:date.
```

一个目录可以包含多个数据集，所以当链接到新的数据集时，只增加新的数据集链接是更简单、更可取的。例如，一个改变集可以被用来增加一个新的 dc:title 到数据集。下面的更新将指向目录来增加一个额外的数据集。

EXAMPLE 10

```
@prefix : <http://example.com/>.
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix cs: <http://purl.org/vocab/changeset/schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.

<change1>
  a cs:ChangeSet ;
  cs:subjectOfChange :catalog ;
  cs:createdDate "2012-01-01T00:00:00Z" ;
  cs:changeReason "Update catalog datasets" ;
  cs:addition [
    a rdf:Statement ;
    rdf:subject :catalog ;
    rdf:predicate dcat:dataset ;
    rdf:object :dataset/002 .
  ] .
```

（见功能需求 F4.2）

4.5 UC5:确定一个资源是否改变

检索关于一种资源的版本控制信息而不用下载资源的一个表达应该是可能的。这个信息与之前的信息作对比来确定资源是否被改变。这个版本控制信息也可以被用在后续的条件式请求中来确保它们只在版本无变化时被应用。

NF5.1：一种资源的多重改变。

4.5.1 基本场景：确定一个资源是否改变

基于用户描述 Constrained Devices and Networks（受约束的设备和网络），一个 LDP 服务器可以被配置为一个 CoAP 的代理服务器。作为 CoAP 资源的一个观测者，LDP 服务器注册它的兴趣，这样无论何时传感器读到变化它都能被通知。LDP 的客户端可以询问服务器来确定状态是否变化。

在这个例子中，关于传感器和通讯传感器解读的信息可以表达为 RDF 资源。下面的第一个资源表达了一个使用语义传感网络本体描述的传感器。

EXAMPLE 11

```
@prefix : <http://example.com/energy-management/>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix ssn: <http://purl.oclc.org/NET/ssnx/ssn#> .

<> a :MainsFrequencySensor;
  rdfs:comment "Sense grid load based on mains frequency";
  ssn:hasMeasurementCapability [
        a :FrequencyMeasurementCapability;
        ssn:hasMeasurementProperty <#property_1> .
  ] .
```

LDP 客户端可以询问下面的资源来确定它是否改变，而不需要下载 RDF 表达。由于不同的传感器属性被独立地表达，它们可以独立地变化。

EXAMPLE 12

```
@prefix : <http://example.com/energy-management/> .
@prefix ssn: <http://purl.oclc.org/NET/ssnx/ssn#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://example.com/energy-management#property_1> :hasMeasurementPropertyValue <
<> a :FrequencyValue;
        :hasQuantityValue "50"^^xsd:float.
```

（见功能需求 F5.1）

4.6 UC6：聚合资源

有一个能够管理资源集合的要求。一个集合的概念有重合但是又不同于一个容器的概念。这些集合是汇集，与资源的生命周期管理没有关系，不同于一种资源和容器之间的所有者。然而，一个容器的组成可以作为一个集合来支持容器和它内容的导航。需要通过增加和删除单独的成员属性来创建集合。资源可以属于多重集合，也可以不属于任何集合。

NF6.1：资源描述是简单数据和集合的混合。

NF6.2：相关联的 URIs：对一个集合来说，承担 RDF 的有效负荷应该是可能的，而不打破来自 Constrained Devices and Networks 的内部链接。

4.6.1 基本场景：增加一个资源到一个集合

这个例子来自 Library Linked Data 和 LLD-UC，特别是主题查找。

在 http://example.com/concept-scheme/subject-heading 中有一个现存的集合，定义了主题标题的一个集合。这个集合定被义为 skos:ConceptScheme，通过一个 skos:inScheme 链接关联到集合。在下面的例子中，一个新的主题标题"outer space exploration" 被添加到 scheme:subject-heading 集合中。下面的 RDF 描述了集合的（项目级别的）描述，也验证了父和子资源之间的关系可以在一个看起来与正常预期相反的方向中运行，从子到父。

EXAMPLE 13

```
@prefix scheme : <http://example.com/concept-scheme/>.
@prefix concept : <http://example.com/concept/>.
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .

scheme:subject-heading a skos:ConceptScheme.

concept:Outer+space+Exploration skos:inScheme scheme:subject-heading.
```

（见功能需求 F6.1）

4.6.2 可选场景：添加一个资源到多重集合

逻辑上，一个资源应该被一个以上的容器拥有。然而，它可以是多重集合的一员，定义一个更弱的汇集格式。因为这是一个集合的 RDF 描述的一个简单操作，增加相同的资源到多重集合应该是可能的。

作为一个机器可读的医学术语集合，SNOMED CT 本体在用户描述 Healthcare（医疗卫生）中是非常重要的。在下面的例子中，SNOMED 概念被当作更窄概念的集合。概念:TissueSpecimenFromHeart 属于两个父集合，因为它既是:TissueSpecimen 也是:SpecimenFromHeart。这个例子也验证了组成和汇集如何支持不同的场景。

EXAMPLE 14

```
@prefix : <http://example.com/snomed/>.
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .

:TissueSpecimen a skos:Concept ;
        :conceptID "119376003";
        skos:prefLabel "Tissue specimen"
        skos:narrowerTransitive :TissueSpecimenFromHeart.

:SpecimenFromHeart a skos:Concept ;
        :conceptID "127462005";
        skos:prefLabel "Specimen from heart"
        skos:narrowerTransitive :TissueSpecimenFromHeart.

:TissueSpecimenFromHeart a skos:Concept;
        :conceptID "128166000";
        rdfs:label "Tissue specimen from heart".
```

（见功能需求 F6.2）

4.7 UC7：过滤资源描述

这个应用案例通过排除特定的属性来扩展检索一种资源 RDF 描述的一般行为。对于容器来说，能读取一个集合或排除容器成员的项目级别描述是令人满意的。

4.7.1 基本场景：检索集合水平的描述

这个场景基于 Library Linked Data，使用 DC-COLLECTIONS（都柏林核心元数据倡议集合级别描述）。一个集合可以指任何物理的或数字项目的汇集。这个场景包含了客户端请求一个集合级别描述的案例，而不需要在集合内下载一个完整的项目列表。

EXAMPLE 15

```
@prefix dc: <http://purl.org/dc/elements/1.1/>.
@prefix : <http://example.org/bookshelf/>.
@prefix dcmitype: <http://purl.org/dc/dcmitype/>.
@prefix cld: <http://purl.org/cld/terms/>.
@prefix dcterms: <http://purl.org/dc/terms/>.

<> dc:type dcmitype:Collection ;
        dc:title "Directory of organizations working with Linked Data" ;
        dcterms:abstract "This is a directory of organisations specializing in Linked Data."
        cld:isLocatedAt <http://dir.w3.org>
        cld:isAccessedVia <http://dir.w3.org/directory/pages/landing-page.xhtml?view>
```

（见功能需求 F7.1）

4.7.2 可选场景：检索一个集合的项目级别描述

这个应用案例也是基于 Library Linked Data，关注于获取通过一个集合汇集资源的一个项目级别描述。下面的例子使用 rdfs:member。

基于以下例子，项目级别的描述应该包含一个 rdfs:member 关系的最小值。不需要包含集合的其它属性，不需要成员的额外属性。

EXAMPLE 16

```
@prefix frbr: <http://purl.org/vocab/frbr/core#>.
@prefix dc: <http://purl.org/dc/elements/1.1/>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<> rdfs:member <#ebooks97>, <#ebooks21279>.

<#work97> a frbr:LiteraryWork;
    dc:title "Flatland: a romance of many dimensions" ;
        frbr:creator <#Abbott_Edwin>;
        frbr:manifestation <ebook97>.

<#work21279> a frbr:LiteraryWork;
        dc:title "2 B R 0 2 B" ;
        frbr:creator <#Vonnegut_Kurt>;
        frbr:manifestation <ebook21279>.
```

（见功能需求 F7.2）

4.8 UC8：在多个部分检索一个大的资源描述

这个应用案例用“资源为中心”的方法解决与 RDF 数据交互的难题。问题是一些资源加入了大量的三元组，因此一个资源为中心的粒度导致在一个独立的 HTTP 请求里，资源描述太大而无法被处理。这些应用案例适用于所有资源，不只是容器。

4.8.1 基本场景：标页码

在用户描述 Maintaining Social Contact Information（维护社会联系信息）中，用户有很多的联系是普遍的。这导致了大量的资源描述，尤其也包含联系的基本信息。这个表达的大小可能非常大以至于在单一的 HTTP 请求里检索是不现实的。

在这个例子中第一个请求的响应包含了一个到 next 资源的参考。为了实现例子的目的，我们利用 XHTML Metainformation Vocabulary 定义的 next 属性。

EXAMPLE 17

```
@prefix : <http://example.com/people/>.
@prefix xhv: <http://www.w3.org/1999/xhtml/vocab#>.

:alice a foaf:Person;
   rdfs:label "Alice";
   foaf:mbox <mailto:alice@example.com>.

<> xhv:next <http://example.com/1234567890>.
```

当客户端请求被标识为 next 的资源时，响应包括额外的内容，该内容可以与早期的数

据合并来来构建原始请求资源的更完整模型。它也可能包含更多的 next 链接。

下面的表达是对被标识为 next 的资源的响应，同时完成联系列表。

EXAMPLE 18

```
@prefix : <http://example.com/people/>.

:bob a foaf:Person;
   rdfs:label "Bob";
   foaf:mbox <mailto:bob@example.com>.
```

（见功能需求 F8.1）

4.9 UC9：管理二进制资源

很容易地添加非 RDF 的二进制资源到接收它们的容器中应该是可能的。二进制资源可以在容器的生命周期中被更新和移除。

4.9.1 基本场景：获取二进制资源

从用户描述 Sharing Binary Resources and Metadata（共享二进制资源和元数据）中，很容易地添加非 RDF 的二进制资源到接收它们的容器中应该是可能的。客户端以被容器接受的一种媒体类型将一个非 RDF 表达提交到容器中。容器创建一个 URI 来表达这个媒体资源，并在容器中创建一个链接到新的 URI。二进制资源伴随着一个易于理解的 RDF 描述。应该可能找到关于这样一种资源的元数据并以通常的方式来获取和编辑它。

EXAMPLE 19

```
@prefix ma: <http://www.w3.org/ns/ma-ont#> .

<dataset> a ma:Collection ;
        ma:hasMember <dataset/image1.jpg>

<dataset/image1.jpg> a ma:MediaResource ;
        ma:hasFormat "image/jpeg" .
```

（见功能需求 F9.1）

4.9.2 可选场景：媒体资源附件

一种资源可能有多种表达形式。例如，你可以用 PDF 和 JEPG 来表达同一件事物。一个用户试图创建一个伴随着附加图片的作品顺序。对于用户和作品顺序系统，这两个人工行为被作为一个集合管理。一个单独的请求可以自动创建作品顺序、附件以及它们之间的关系。当用户检索作品顺序时，他们期望一个单一的默认请求可以检索作品顺序以及所有的附件。当用户更新作品顺序后，如标识作品已完成，他们只想更新作品顺序，而不是他的附件。用户可以在作品生命周期中添加/移除/替换附件。

（见功能需求 F9.2）

5. 需求

这部分列出了来自用户案例的功能性和非功能性的需求。

5.1 功能性需求

F1.1：

系统应该提供创建组成资源的容器的功能。

F1.2：

系统应该提供创建嵌套容器的功能。

F1.3:

在删除一个容器时，系统应该删除任何包含的资源和嵌套容器。

F2.1：

系统应该提供在一个容器中创建资源的功能。

F2.2：

系统应该提供删除资源的功能。

F3.1:

系统应该提供检索资源描述的功能。

F3.2

系统应该使客户端能够检索一个 hash URI 的描述。

F4.1：

系统应该提供通过替换来更新现存资源的功能。

F4.2：

系统应该提供执行一个资源可选更新的功能。

F5.1：

系统应该提供确定一种资源是否改变的功能。

F6.1:

系统应该提供聚合资源的功能。

F6.2：

系统应该支持一种资源的附加物。

F7.1：

系统应该提供检索一个集合级别的描述的功能。

F7.2：

系统应该提供检索一个项目级别的描述的功能。

F8.1：

系统应该提供检索一个标页的描述的功能。

F9.1：

系统应该提供存储和获取媒体资源的功能。

5.2 非功能性需求

NF1.1：

系统应该提供资源的获取指南。

NF2.1：

系统应该鼓励资源的非复件。

NF2.2：

系统应该支持资源的分配。

NF2.3：

系统应该支持一致的、全球性的命名。

NF3.1：

系统应该在适当的时候支持标准词表的使用。

NF3.2：

系统应该提供一个可扩展的关联模型。

NF4.1：

系统应该允许非限制性词表。

NF5.1：

LDP 应该在同步获取一种资源的案例中确保持续的获取。

NF6.1：

系统应该允许资源描述是一个简单数据和集合的混合。

NF6.2：

系统应该支持相关的 URIs 来实现集合的共享。

（编译自：Linked Data Platform Use Cases and Requirements http://www.w3.org/TR/ldp-ucr/[2013-10-31]）

（吴贝贝编译，王妍校对）