·开放获取·

提高学术知识库的互操作性

Jeroen Bekaert，Herbert Van de Sompel 著　　林颖　编译

（林颖，北京师范大学图书馆，100875）

2006 年 4 月 20-21 日，微软公司、美国梅隆基金会、网络信息联盟、数字图书馆联盟以及英国联合信息系统委员会在美国纽约共同主办了一次关于促进学术知识库间互操作性的会议。

1、会议内容

越来越多的组织机构都将构建知识库将其作为存放和分享丰富的数字对象的方式。但是，每个组织机构在数字资源的知识库的设计和管理上都持有不同的观点。因为知识库系统对不同的用户群体会有不同的服务，对于采集哪些资源、使用哪种存储机制和认证机制、长期保存采取哪些策略等等也都有不同的政策。然而，在特定资源或应用领域中，知识库系统的设计和实施很可能会应用各种不同的技术，这就不可避免地导致知识库之间缺乏跨组织和跨知识库的互操作。因此，到目前为止，依然还没有真正实现一个内部互连的数字知识环境，它以异构的知识库为核心，能够灵活地利用或再利用异构知识库中的数字对象。此次会议的主题是对这方面的探讨以及对互操作技术的解决方案达成基本认识，包括不同的目标和做法是否能够最终实现互操作？这次会议的目的是达成一个最低限度的共识。

在设计互操作框架的过程中，每个成员都应该了解各种可能出现的问题，Herbert Van de Sompel 在会议上着重对这方面提出自己的见解，他认为，存在两种主要类型的跨知识库价值链促进跨知识库互操作性的需求：

- 能够覆盖多个知识库的更丰富的跨知识库服务（例如，发现服务、虚拟馆藏服务）。
- 跨知识库的学术交流流程（例如，学术交流流程的主体是数字知识库中存贮的数字对象，并且可以在不同的环境下利用和再利用）。

价值链实例：

- 为数字对象提供面向发现的服务以便将知识库中数字对象（含部分）传递给相关成员。一个“化学搜索引擎”就可以搜索数字对象中计算机可处理的化学结构。而且可以从多个分布式知识库中获取数字对象。
- 为数字对象提供增值服务以便将知识库中数字对象（含部分）传递给相关成员。比如，一个编辑可以从各个不同的机构知识库中收集学术论文，并且通过加工将它们形成一个完整的期刊（例如，增加元数据或增加对文章的评论）。
- 引入机器可读和机器可操作的书目引文。
- 各个不同知识库的数据集的再利用是建立新数据集或用于发行的基础，而且知识库要能够存取和再揭示新创建的数字对象。
- 在各知识库间镜像数字对象以是保证资源的备份。
- 数字对象虚拟馆藏的建立。一个知识库应该要表现一个连贯内容的馆藏，尽管事实上，它是由多个不同的知识库构成。在这种情况下，虚拟馆藏中的各部分不应该在需要的时候才被显现出来。我们也不应该在虚拟馆藏中复制数字对象，而应该采取某种形式的重新定位。”

我们已经实现了上述某些目标，然而只是以特定的或手工的方式。通常情况下，我们还忽视了数字对象源（它存储在某个知识库中）与源于其它数字对象创建的数字对象之间的联

系。

2、会议预期目标

此次会议主要关注能够实现上述异构知识库所需要的某种层次的互操作：

- 就一定量的建立基于协议的知识库接口（REST-full 或基于 SOAP 的 web 服务）的核心的本质和特征达成共识，以便下层应用系统以更有效和一致的方式与异构知识库相互作用。
- 为全面制定、确认和实施这样的知识库接口确定相关具体工作。
- 为全面制定、确认和执行这样的知识库接口确定工作时间表。

Cliff Lynch 在他的发言中认为：“我不期望能够完成规范说明、协议测试以及真正的实用，但是我希望对互操作方式和如何在这些方式上执行达成某种一致。经验表明，这种工作需要长时间的讨论和在试验中不断的完善，我希望能够在此次会议结束前能够提出一个未来一段时间内我们所需要采取的试验方法。”

3、什么是数字知识库？

探索数字知识库的相关内容还需要了解数字知识库的本质及其所存储的复合资源的类型。此次会议上 Herbert Van de Sompel 提出知识库需要完善相关定义：

知识库是一个能对相关的数字对象提供服务网络系统。通常包括以下几种类型：机构知识库、出版商知识库、数据集知识库、学习对象知识库、文化遗产知识库等。

Cliff Lynch 也对此作出了进一步的解释：“目前，我们还没有真正对知识库的定义达成共识。我们缺少技术规范用于测试知识库在互操作环境下的一致性。可能我认同的知识库系统但是你却有可能不认同。目前，我们只能达成这样的共识，如果一个系统假定为知识库，但是它并不支持 OAI 元数据收割协议，那么它就不是一个性能优越的数字知识库。在这次会议中，我们致力于提出知识库的各种技术和表现特性。”

4、什么是数字对象？

会议还认为：一个数字对象是一个必须包含数字数据和关键元数据的数据结构。数字数据可以是一个数据流或数字对象，即，一个数字对象可能会由一个或多个其它的数字对象构成。关键元数据必须包括这个数字对象的标识符。这里对数字对象的概念雷同于 Kahn/Wilensky 数字对象。

在过去的几年中，很多学术研究领域都在关注（复合或复杂）数字对象：例如，e-learning（数字学习）对象、艺术对象、博物馆对象、e-science（数字科研）数据集、化学数据库、电子图书、期刊、复合论文，而且，每种数字对象都可能包含多个数据流，或者以某种方式与其它数字对象相关联。因此，我们需要组合、分解和结构化这些数字对象。一个数字对象的一部分要素会存储在一个地方，而其余部分则会要求保存在另外的地方等。而且，大多数学术研究领域对这些复合的数字对象的依赖在飞速增长。

5、构建一个跨知识库的互操作层

会议在 Pathways、aDORe、CORDRA、中国 DSpace 联盟项目、各种的 JISC 项目以及 OAI 等基础上，提出了下列增进互操作的方式：

- 支持跨知识库数字对象的数据模型。数据模型是数字对象的抽象或更高层次的概要抽象，每个数字对象即为数据模型所定义的实例。数据模型是要对一些异构的数字仓储的数字对象提供一个普遍的表示。
- 支持通过代理延续数字对象与数据模型的一致性。事实上，代理就是数字对象连续

信息的外在表现形式，它用于来回传输和使用数字对象。常见的数字连续技术主要有 XML 和 RDF/XML。

- 支持一个性能优越的数字知识库所必须的一系列核心服务和接口：
  1) 获取接口：它是一个支持各数字对象服务请求的知识库接口。它的特性至少表现在两个接口层次上：a）一个能够获得指定数字对象代理的获取接口。每个知识库都必须支持这类接口。b）一个能够获得知识库数字对象（或数据流）特殊服务的获取接口。各个独立的知识库都应该能够支持这类接口，而且能够作为知识库整体执行。
  2) 收割接口:它是一个通过揭示数字代理表现形式用于数字对象外部采集和收割的知识库接口。
  3) 存放接口：它是一个向知识库提交一个或多个代理的知识库接口。

6、会议讨论概要

在会议讨论中，提出了大量的互操作技术和方案， Pathways 项目甚至已经能够进行原型系统演示。在这些讨论的基础上，还应该形成一个概要总结，以下就是相关的一些问题概述。

6.1 数据模型与代理

数据模型的概念非常重要，它从更高层次上对数字对象进行抽象概括，具有更长的时间延续性。因为数字连续技术是会随着时间因为技术的进步而改变。更高层次的数字对象抽象也是学术知识库对于长期保存的期望。而且，数字对象抽象具备一定程度的灵活性，它能够与多种数字连续（数据模型一致）的框架并存。

Carl Lagoze 介绍了 Pathways 的核心数据模型。Pathways 数据模型作为一个被大会提议的互操作数据模型，其重点就是要适应跨知识库服务和数据流。它允许以相同的知识库中间件方式传送和获取数字对象的属性和要素。

Pathways 的核心是实体元素和数据流元素。实体元素表现数字对象的抽象属性。实体元素具有嵌套递归属性，它能够将一个数字对象（含部分）的任何内容抽象为一个数字对象集。实体元素的主要属性包括 hasIdentifier、hasProviderInfo、 hasLineage、hasProviderPersistence 和 hasSemantic。这些属性在后继的篇幅中，还有详细的解释和讨论。

数据流元素表现数字对象的具体属性，它表现比特流层次的属性。实体可能会有多个的数据流。Pathways核心中定义了两种数据流特性：hasLocation传递内含比特流的URI；hasFormat传递数比特流的数字格式。

Pathways核心数据模型能够通过多种方式进行实例化，此次会议的演示系统使用的是RDF语法。

6.1.1 Pathways 核心树型结构

在Pathways核心中，数字对象或复合数字对象的树型结构的根部首先衍生的是hanEntity和hasDatastream属性。数字对象由1-0实体和1-0个比特流构成。实体元素表现数字对象的抽象属性并且可以嵌套递归，而数据流元素则表现具体的比特流。嵌套特性允许数字对象（含部分）包含其它数字对象，而且也允许共享内容的联合。

这种“包含”概念来自描述数字对象数据结构的 Kahn/Wilensky 框架，这个框架提出数据由多个比特序列和多个数字对象构成。一个包含其它数字对象的数字对象通常被认为是一个复合对象。非复合数字对象就被认为是单数字对象。

这个总体模型看起来不易引起争议而且结构完善，事实也表明此次会议对此也鲜有争议。

6.1.2 修饰 Pathways 核心树型结构

会上，Jim Gray 指出，还要进一步修饰 Pathways 核心数据模型树结构的其它特性节点。下文就是对这些特性的总体概述。

（1）数字对象标识符

在 Pathways 核心中，数字对象的标识符通过 **hasIdentifier** 属性传递。这是一个关于标识符类型的中立特性，它主要用于识别数字对象。这样，数据模型就可应用于更多类型的知识库系统。因为文化、政策、组织或技术等原因会存在不同的标识符框架，但最终一个（修正过的）数字对象是否需要一个新的标识符还是生成一个新版本的指示符，是由知识库的策略应用所决定。

（2）ProviderInfo:获得数字对象代理

在 Pathways 核心中，**providerInfo** 记录了获取数字对象代理格式所必需的信息。这些信息主要包括三个方面：a）用于揭示数字代理的知识库标识符；b）数字对象的第一标识符（preferredIdentifier）；c）可选的版本关键字。通过在实体结点末端增加 providerInfo，就能够在附加服务中获得和重复使用该结点的数字连续流，以及数字对象重复使用的尺度。这种获取数字对象代理的过程如下所述：

首先，在服务注册列表中使用知识库标识符（也就是提供者的标识符）。注册后会返回这个知识库其它可用服务的信息（类似返回服务接口位置）。其中“获取”服务就是：允许通过数字对象的 preferredIdentifier 获取其代理格式。其次，使用数字对象的 preferredIdentifier（以及可选的版本关键字）从获取接口中获得被标示的数字对象的代理格式。

（3）使用 ProviderInfo 持续获得数字对象的代理

实体元素的hasProviderPersistence属性表现提供者对数字对象的延续性，即通过实体的providerInfo特性返回的代理是否能满足将来的需求。这种延续性也会随着知识库策略、资源类型等等而变化。

会上，关于这种属性展开了多方面的讨论，Cliff Lynch对这些不同的观点作了最后的总结：“我们需要特别阐明代理、代理永久性、基本资源的永久性之间的关系。首先必须要了解相关的策略是什么、在哪里才能正确应用这种技术机制，但同时还要强调当前实践的不完整性。尽管我们认为要保证观点的永久性，但是目前存在着多种类型的出版商，他们对已经存在的数字对象鲜有修正。每个出版商都可以对数字对象采用不同的策略。尽管通过目前的机制无法修正，但是我们可以重新决定其中的一些领域。”

（4）继承：获取代理的工作流

hasLineage属性传送数字对象的历史信息，它记录了数字代理的发展过程。实际上，继承属性包含的信息是providerInfo的复制：它包含的信息是获取数字对象的数字代理所需要的信息，这个信息在工作流的输入中被使用。在数字代理中加入继承属性，可以更准确地揭示代理格式的工作流。

（5）数字对象的格式和语义及其构成

在 Pathways 核心中，hasSemantic 属性传送数字对象的“类型”或构成。HasFormat 属性传送关于破译或解析数据流的比特结构信息。不论类型还是格式，它们都会促进更深层次的服务联合。例如，Jane Hunter 的 PANIC 工作就是基于服务驱动 MIME 的一个典型示例。

通常，格式习惯于表示为 MIME 类型，但 MIME 也有较大的局限。例如，MIME 格式“application/pdf”并没有指出这个 PDF 文件的版本信息，不同的 PDF 版本会有不同的特性和表现形式。因此，专家建议、，类型和格式有必要通过全球唯一标识符来揭示。而且每个标识符都应该与注册的某个类型和格式条目相一致。目前格式注册相关的有 PRONOM 系统

和 Global digital Format Registry 系统。

6.2 核心服务

这部分阐述一个性能优越的知识库所应该具备的最核心的服务。Andy Powell, Herbert Van de Sompel 和 achel Heery 均分别阐述了收割、获取和存放之间的区别。

6.2.1 收割接口

收割接口是通过揭示数字代理表现形式用于数字对象外部采集和收割的知识库接口。其中最重要的概念就是可选择的收割方式，应用这种收割方式可以仅仅收割在一段时间内产生或改变的数字资源的数字代理。以下就是收割的相关描述：

- 聚合器。这是一种能够聚集并揭示不同知识库的数字代理的服务，其它应用或服务可以基于此进行数据收割。
- 文摘和索引。这是一种能够聚集不同学术知识库中的数字代理并且通过获取接口可以访问相关的数据流和书目元数据的应用。通常将这些信息综合在一起后，还需要对可用的数字对象进行索引、内部连接以及排序。
- 存档。这是一种使用收割接口和获取接口将数字对象的备份迁移到一个用于长期保存的存档存储的服务。

OAI-PMH（开放存档计划元数据收割协议）是一个性能优越的收割数字资源的框架，OAI-PMH 主要是用于收割基于 XML 的描述性元数据记录，例如 DC 或 MARCXML。从模型的扩展角度出发，OAI-PMH 应该被认为是收割数字代理的一种方法。很显然，数字代理还应该与之前描述的数据模型相一致。

OAI-PMH从一开始就受到了数字图书馆团体的高度关注，尤其它可用于交换大量各不相同的数字仓储中的描述性元数据。因此，越来越多的人有兴趣使用揭示描述性元数据的OAI-PMH促进对内容收割接口更加广泛地采用。

如果使用OAI-PMH收割数字代理，那么还要注意以下事项：

- 应该根据 OAI-PMH 的资源标识符进行基于数字对象标识符的收割请求服务，但是目前还都是根据 OAI-PMH 条目标识符进行收割请求。
- 同样，OAI-PMH 的时间戳应该保存在元数据记录中而不是数字对象中。因此，对大量的复合数字对象而言，还需要一些研究工作来规范 OAI-PMH 时间戳的含义，同时还必明确时间戳和数字代理、数据流和数字对象之间的不同关系。
- 另外，OAI-PMH 并不应该是一个永久可靠的解决方案。OAI-PMH 规范只能基于 HTTP 协议和 XML 语法传送和包装收割的记录。OAI-PMH 最大的用处还在于它对收割接口的抽象定义。

其它收割技术还有RSS2.0协议和Atom出版协议，RSS是类似微软简单分享扩展的内容联合插件。就目前而言，后者更为流行。

6.2.2 获取接口

获取接口是支持独立数字对象（包括复合数据流）请求服务的知识库接口，它至少在两个接口层次上存在特性。此次会议的重点仅仅讨论它在第一层（相对简单）的一致性。

- 一致性的简单层。这一层能够获取一个被标识的数字对象的外在表示。例如，访问一个数字对象的 pathways 核心的数字代理。
- 一致性的高级层。这一层能够请求一个指定数字对象包括其数据流的服务。例如，获取指定格式的数据流，它可以是数据流的 PDF 格式也可以是 MS Word 格式。不过这次会议并没有讨论与这一层相关的内容。

NISO 用于情景敏感服务的 OpenURL 框架可作为获取接口的基础。OpenURL 框架提出

了一种定义服务环境的构想，在这个环境中信息包能够进行网络传递。这些信息包包含一个与数字对象相连的标识符，它们通过获取被指引对象的情景敏感服务进行传送。为了使这些信息包的接收者传递这样的一种情景敏感服务，每个信息包都要描述被指引的对象、服务的类型、指引对象的网络环境以及请求服务发生的环境。尤其值得强调它的两个方面特征：

- OpenURL 框架允许知识库或智能代理包含在获取请求需要的情景敏感服务。这对数字代理的访问可能并不非常重要，但起码证明了对数据流的请求服务而言是最基本的。
- 以抽象的方式定义 OpenURL 框架：OpenURL 概念的抽象定义和真实表现可能会有明显的区别。它支持任何命名空间的标识符，也不要求一定要应用指定的标识符技术。

6.2.3 存放接口

存放接口是向知识库提交一个或多个数字代理的接口，从而来促进知识库数字对象馆藏的增加。通常有以下几种情形：

- 存放接口能使用户更快更有效地组织知识库的数字资源。例如，作者可以从一个桌面写作系统中将完成的报告直接存储到机构知识库中。
- 存放接口能够使知识库更有目的地交换数据。例如，一个机构知识库可以提交一个学习对象至一个联合学习对象知识库中。
- 在实验数据知识库中，分光计输出的实验数据要另存为一个文件，并同时生成这个文件的元数据。对实验数据知识库而言，首先要调用数据获取服务，然后存放这次实验的文件与必需的元数据。

6.3 其它服务接口

此次会议还讨论了能够促进互操作发展的搜索接口和出版订阅服务。总的来说，工作组成员认为这两种接口都非常基础，尽管它们不是核心服务，但还是应该被视为知识库的自主服务，应该在与诸如收割、获取等真正的核心知识库接口交互过程中创建。

6.3.1 搜索接口

6.3.2 订阅接口

出版订阅接口是一个基于学科和事件驱动的服务，一旦有符合某个用户需求的新的内容或数字对象被创建，这个接口就能及时通知该用户。

6.4基础设施构成

会议的另一个主题是Jeremy Frumkin关于注册和基础设施的介绍，他认为当前提议的互操作框架需要一些基础设施的支持：

- 服务注册就是其中一种基本的需求，它能促进框架中各个知识库核心接口（获取、收割以及存放）的查找。服务注册将知识库的标识符作为其关键字，并且记录服务和定位服务接口。
- 格式注册和类型注册是另一种类型的需求，分别记录了 hasFormat 和 hasSemantic 应用中所需要的信息。前者将媒体格式的标识符作为其关键字，记录媒体格式的各种属性。后者将语义类型作为其关键字，记录其中的各种属性。

编译自：Jeroen Bekaert, Herbert Van de Sompel.Augmenting Interoperability across Scholarly Repositories.http://msc.mellon.org/Meetings/Interop/FinalReport. [2006-10-26]