关联数据能力指标

Competency Index for Linked Data

引 言

关联数据能力指标（Competency Index, CI）是由关联数据专业实践所需的知识、技能与思维习惯（方式）等指标因素构成。能力指标的结构如下：

1.主题簇

1.1 主题

1.1.1. 能力：知识、技能与思维习惯的主张（140 个字符的 Tweet 长度）

1.1.1.1.基准：达成相关能力的行动

LD4PE 网站的学习资源页面完全按照此结构展开［1］。在 CI 编辑委员会指导下，CI 的未来发展希望得到广大专家学者的群策群力、共同推进。

基于 RDF 建模的 CI 主要目标是，提供关联数据能力指标以及相对应的学习资源描述。针对特定学习需求，促进对学习资源的发现、识别、选择、获取与探索。在探索学习资源页面显示的能力指标中［1］，能力/基准节点后紧跟一个数字，该数字表示学习资源描述对应的能力/基准序号。

CI 的模型与表征参考了图书馆领域的成就标准网络（Achievement Standards Network, ASN）能力指标［2］。LD4PE 项目其他信息和进度，参见 LD4PE 网站中简报［3］的 **LD4PE 概述**［4］。

［1］ http://explore.dublincore.net/explore-learning-resources-by-competency/
［2］ http://asn.jesandco.org/resources/D25275893
［3］ http://explore.dublincore.net/theory/briefing-papers/
［4］ http://explore.dublincore.net/theory/briefing-papers/ld4peoverview/

关联数据能力指标

1. RDF（资源描述框架）基础

1.1 RDF 中的识别

1.1.1. **知道**在 RDF 中所有东西（thing）通过唯一资源识别符 URIs 进行命名，例如，代理、地点、事件、物件与概念等。

1.1.2. **了解**现实世界中的事物可能需要通过 URI 来命名，这与关于事物的信息的 URI 并不是一回事。

1.1.3. **辨认** URIs 的归属，通过互联网域名的所有者来判断。

1.1.4. **知道**唯一资源识别符 URIs 包括唯一资源定位符 URLs（用来定位网页），以及与位置无关的物理资源、概念资源与网络资源识别符。

1.2 RDF 数据建模

1.2.1. **知道**一个三元组（triple）由“主语－谓语－宾语”组成。

1.2.2. **了解** URIs 和文本串（literal）代表现实的、想象的、概念化的事物。

1.2.3. **了解**文本串资源（literal）与非文本串（non-literal）资源之间的差异。

1.2.4. **了解**资源（resource）是通过 rdf:type 属性来声明为某类（class）的成员（实例 instance）。

1.2.5. **了解**文本串的数据类型与语种标签的使用。

1.2.6. **了解**空节点（blank nodes）及其使用方法。

1.2.7. **了解** QNames 为长 URIs 定义了简易前缀。

1.2.8. **了解**命名图（named graph）是图集合的一种，构成一个 RDF 数据集。在该数据集中图的名称是唯一的。

1.2.9. **了解** RDF 中的命名空间（namespace），非正式地用于命名空间 URI 或 RDF 词汇。这与面向对象中类的数据属性与功能（方法）的命名空间存在本质区别。

1.2.9.1. 在 RDF 规范文本（specification）和 RDF 数据中为 URIs 使用前缀。

1.2.10. **阐明** RDF 抽象数据模型与 XML 模型、关系模型的区别。

1.2.11. **了解** RDF 抽象数据模型是一个有向标记图。

1.2.12. **知道** RDF 模型的图形化绘制惯例。

1.2.12.1. 能够使用绘图或建模软件与他人分享 RDF 模型。

1.3. 相关的数据模型

1.3.1. **掌握**语法验证用途（XML）和推理用途（RDF schema）在模式（schemas）上存在的本质区别。

1.3.2. **区分**层级式文档模型（XML）与图模型（RDF）。

1.3.3. **了解** RDF 的类（事物命名集合）与面向对象编程的类的本质区别。面向对象编程的类定义了对象的类型，封装了“状态”（数据值的属性）与“行为”（对状态进行操作的函数）。

1.4. RDF 序列

1.4.1. **区分** RDF 抽象数据模型与 RDF 数据的序列化表示。

1.4.1.1. 表征 RDF 序列数据（serialization），例如，RDF/XML、N-Triples、Turtle、N3、Trig、JSON-LD 及 RDFa 等格式。

1.4.2. **了解** RDF 序列作为给定三元组集合（RDF 图）可互换的编码格式。

1.4.2.1. 会用工具实现 RDF 数据不同序列格式之间的转换。

2. 关联数据基础

2.1 Web 技术

2.1.1. **知道**万维网（World Wide Web, 1989）的起源，其本身是基于互联网的非线性交互系统或多媒体系统。

2.1.2. **了解**关联数据（Linked Data, 2006）将文档的网络扩展为细粒度数据的网络（关联数据云图）。

2.1.3. **知道**超文本标记语言 HTML（1991+）是作为网页内容与多媒体组件的标记化语言。

2.1.4. **知道** HTML 的最新版本 HTML5（2014）扩展了对复杂网络与移动应用的支持。

2.1.5. **知道**超文本传输协议 HTTP（1991+）是用于解析万维网上的超链接与传输数据的基本技术。

2.1.6. **知道**表征状态传输（Representational State Transfer, REST）(2000）是一种软件架构风格，通过浏览器与网络服务器交换数据，通常基于大家所熟悉的 HTTP 操作。

2.2 关联数据原则

2.2.1. **知道** Tim Berners-Lee 提出的关联数据原则：使用 URIs 命名事物，使用 HTTP URIs 解析出有用的信息，与其他事物的 URIs 建立链接。

2.2.2. **知道**开放数据 5 星标准：数据上网，优先使用结构化非专属数据格式，使用 URIs 对事物命名，链接其他数据。

2.3 关联数据策略与最佳实践

2.3.1. **知道**关联数据标准化相关的主要组织机构。

2.3.1.1 参与标准制定和相关组织（如 W3C）的最佳实践。

2.4 非 RDF 关联数据

3. RDF 词汇与应用纲要

3.1. 发现 RDF 词汇

3.1.1. **知道**如何通过数据门户与注册系统找到 RDF 词汇。

3.1.1.1. 检索<关联开放词汇注册系统>LOV 中注册的属性(properties)与类(classes)，了解它们的版本与依赖关系。

3.2. 设计 RDF 词汇

3.2.1. **会用** RDF Schema 表达词汇表中的语义关系。

3.2.1.1. 正确使用子类关系，以支持推理。

3.2.1.2. 正确使用子属性关系，以支持推理。

3.2.2. **知道** RDF 属性与类的命名惯例。

3.2.3. **重用**已发布的可用属性与类。

3.2.4. **创建**（必要的）新属性与类的命名空间 URIs，视需求而定。

3.2.4.1. 制定属性与类的 URIs 创建规则。

3.2.4.2. 选择基于# 号（hash）或斜线（slash）的 URI 模式，视需求而定。

3.2.5. **知道**网络本体语言 OWL（2004）作为一套 RDF 词汇，扩展了属性与类，以支持表征型（expressive）数据建模与自动推理。

3.2.6. **知道**“本体”一词是模糊的，与 RDF 词汇有关，但更多是在特定领域中支持推理的 OWL 类与属性集合。

3.2.7. **知道**简单知识组织系统 SKOS（2009），一套对自然语言中标记的概念进行表达的 RDF 词汇，将概念组织成非正式的层级，形成概念体系。

3.2.8. **知道** SKOS 的词汇扩展 SKOS-XL（2009），增加了用于描述与关联词汇标签（Label 类的实例）的属性集合。

3.2.9. **了解** SKOS 概念（concept）不是 RDF 类（class），而是 RDF 实例（instance）。因此，SKOS 概念并不与实例集合（类的扩展）建立正式关联。

3.2.10. **了解** SKOS 能灵活表达概念的相关关系，不需要启用更严格的自动推理。基于类的 OWL 本体通常需要严格指定推理规则。

3.2.11. **了解** OWL 子类链与 SKOS 概念层级的区别。SKOS 概念层级的设计初衷不是层级之间自动形成传递机制，因为这并不是人们思考与组织信息的方式。

3.3. RDF 词汇的维护

3.3.1. **了解**持久性保障的策略选项。

3.3.1.1. 起草持久性策略文件。

3.4. RDF 词汇的版本控制

3.4.1. **知道**版本的格式、内容与粒度的相关技术选项。

3.4.2. **了解**如何在 RDF 词汇以定期版本号发布与持续增量发布之间做出权衡。

3.4.2.1. 能够撰写与评判一份版本策略文件。

3.5. RDF 词汇的发布

3.5.1 **了解**“解引用”（derefencable），即通过 URI 应该能检索到它所代表的资源的一种表征。

3.5.1.1. 确保 Web 浏览器解引用时，URI 返回的资源表征形式是用户可读的 HTML。

3.5.1.2. 确保 RDF 应用解引用时，URI 返回的资源表征形式是被请求的 RDF 序列语法。

3.5.2. **了解** RDF 词汇的常用发布格式及其相对优点。

3.5.3. **了解** RDF 词汇通过内容协商发布多种数据格式的目的。

3.6. RDF 词汇映射

3.6.1. **了解** RDF 词汇的层级包含属性（rdfs:subProperty 子属性和 rdfs:subClassOf 子类）可用于表达词汇之间的映射关系。

3.6.2. **了解** owl:equivalentProperty 等价属性与 owl:equivalentClasss 等价类分别适用于属性与类的精确等价关系。

3.6.3. **辨认** owl:sameAs 全等是常见的一种映射属性，带有强大的形式化语义，会引发意想不到的推论。

3.7. RDF 应用纲要

3.7.1. **识别**应用领域内真实世界的实体，作为 RDF 类的备选。

3.7.2. **识别**领域实体间的资源属性与关系，作为 RDF 属性的备选。

3.7.3. **调查**相同或相似应用领域的 RDF 建模方法。

*3.7.3.1. 以文字与图表方式**交流**领域模型。*

*3.7.3.2. **参与**元数据应用纲要的社会化过程。*

4. RDF 数据的生成与转换

4.1. 管理标识符（URIs）

4.1.1. **了解**持久性的含义，即 URI 必须是稳定的、具有记录含义的，能够永久指向特定资源。

4.1.2. **了解**如何在“不透明”的 URIs 与那些使用版本号、服务器名、日期、应用程序特定文件扩展、查询字符串及其他过时的 URIs 之间做出权衡。

4.1.3. **意识到**一个发布的命名空间的政策宣告意味着对重要 URIs 的持久性与语义稳定性作出机构承诺。

4.2. RDF 数据的生成

4.2.1. 基于非 RDF 资源**生成** RDF 数据。

4.2.2. **知道**从表格数据（如 CSV，逗号分隔值）生成 RDF 数据的方法。

4.2.3. **知道**关系型数据直接映射为 RDF 的方法（2012），从关系模型（键、值、行、列与表）数据转换为 RDF 图数据。

4.3. RDF 数据的版本管理

4.4. RDF 数据溯源

4.5. RDF 数据清理与一致化

4.5.1. 清理数据集，包括纠错、去重以及移除不需要的数据。

4.6. RDF 数据映射与丰富化

4.6.1 将可用资源**用于**命名实体识别、抽取与一致性化。

5. 与 RDF 数据的交互

5.1. 找寻 RDF 数据

5.1.1. **知道**如何查找已有的关联数据集相关资源。

5.1.1.1. 检索与访问“开放网络”中的 RDF 数据。

5.1.2. **监控与更新** SPARQL Endpoint 的状态列表。

5.1.3. **使用**可用的数据集描述词汇，以促进数据集的发现。

5.1.4. **注册**数据集及相关服务，以促进发现。

5.2. 使用编程语言处理 RDF 数据

5.2.1. **了解** RDF 数据模型的各个组件（数据集、图、陈述及各种节点类型）如何使用特定编程语言的 RDF 库进行表达。例如，构造为面向对象的类。

5.2.1.1. 使用 RDF 编程库支持的语法对 RDF 数据进行序列化。

5.2.1.2. 使用 RDF 编程方法迭代 RDF 数据组件。

5.2.1.3. 使用 RDF 编程库的特定方法表征常见的 RDF 词汇，例如 RDF、DC 与 SKOS。

5.2.1.4. 以编程方式在前缀中加入相关的命名空间，主要用于 RDF 序列生成和解析 SPARQL 查询两种情况。

5.2.1.5. 使用 RDF 编程库，从 CSV 文件、数据库或网页中提取 RDF 数据。

5.2.1.6. 使用 RDF 编程库，在内存、硬盘或与 RDF 数据库（Triplestore）交互时持久地存储三元组。

5.2.1.7. 以编程方式通过自定义的函数或方法对三元组进行推理。

5.2.2. **了解** SPARQL 查询匹配的模式能够使用 RDF 编程库中功能等价的构造方法表达。

5.2.2.1. 使用 RDF 编程方法查询 RDF 数据，存储查询结果用于后续处理。

5.2.2.2.使用工具与便利的功能为经常使用的模式设置快捷方式。例如，匹配真实数据中的多个标签属性。

5.2.2.3. 使用 RDF 编程库处理各种类型的 SPARQL 查询结果。

5.3. 查询 RDF 数据

5.3.1. **了解** SPARQL 查询是由常量与变量组成的三元组模式，匹配结果为一个 RDF 图。

5.3.2. **了解** SPARQL 查询的基本语法。

5.3.2.1. 使用尖括号来界定 URIs。

5.3.2.2. 使用问号表示变量。

5.3.2.3. 对基地址（base URIs）使用前缀（PREFIX）。

5.3.3. **示范** SPARQL 结果集的格式与用法的操作命令（SELECT、CONSTRUCT、DESCRIBE 与 ASK)。

5.3.3.1. 使用 SELECT 子句识别查询结果表中出现的变量。

5.3.3.2. 使用 WHERE 子句提供的图模式匹配相应的图数据。

5.3.3.3. 在 SELECT 与 WHERE 子句中通过变量获得一张查询结果表。

5.3.3.4. 使用 ASK 对查询模式的匹配进行真/假测试。

5.3.3.5. 使用 DESCRIBE 提取单个图，其包含相应资源的 RDF 数据。

5.3.3.6.根据指定的图模板，使用 CONSTRUCT 抽取结果并将其转换为单个 RDF 图。

5.3.3.7. 使用 FROM 构造 URLs 查询与本地文件查询。

5.3.4. **了解**如何使用操作符（如 UNION、OPTIONAL、FILTER 与 MINUS）对图模式进行组合与过滤。

5.3.4.1 使用 UNION 构造多种可能的图模式查询。

5.3.4.2. 使用 OPTIONAL 构造查询，返回可用的可选变量的值。

5.3.4.3. 使用 FILTER 构造查询，用于结果集过滤。

5.3.4.4. 使用 NOT EXISTS 判断数据中是否存在特定的图模式。

5.3.4.5. 使用 MINUS 删除两个模式评估之后的结果匹配。

5.3.4.6. 使用 NOT IN 限定一个变量不出现在既定的值集合。

5.3.5. **了解** SPARQL 结果集的主要修饰符，例如，对结果限制数量或排序，或返回不重复的结果。

5.3.5.1. 使用 ORDER BY 根据变量、函数调用或表达式来定义排序条件。

5.3.5.2. 使用 DISTINCT 确保查询结果的取值是唯一的。

5.3.5.3. 使用 OFFSET 控制从整体查询结果的哪条记录（基准点）开始处理。

5.3.5.4. 使用 LIMIT 限制查询结果的返回个数。

5.3.5.5. 使用投影将结果序列变换为只保留其中一些变量的新结果序列。

5.3.6. **了解** SPARQL 函数与操作符的用法。

5.3.6.1. 使用正则表达式（regex()）匹配字符串。

5.3.6.2. 使用聚集函数对结果进行分组、投影以及过滤等。

5.3.6.3. 使用 lang()函数返回 RDF 文本串的语种标签。

5.3.6.4. 使用 langMatches()函数匹配语言范围的语种标签。

5.3.6.5. 使用 xsd:decimal(expn)函数将表达式转换为数值。

5.3.6.6. 使用 GROUP BY 子句对结果集进行转换，每个分组变量集只显示一行。

5.3.6.7. 使用 HAVING 子句在分组之后对结果集进行过滤。

5.3.7. **区分**默认图与命名图，使用 GRAPH 子句构造查询。

5.3.7.1. 在本地数据上使用 FROM NAMED 与 GRAPH 构造高级查询。

5.3.7.2. 在远程数据上使用 FROM NAMED 构造高级查询。

5.3.7.3. 在包含空白节点的数据上构造高级查询。

5.3.7.4. 使用子查询构造高级查询。

5.3.8. **使用**临时变量扩展查询。

5.3.9. **了解**属性路径的作用，以及如何通过类似正则表达式的操作符谓词组合而成。

5.3.10. **了解**联合搜索的概念。

5.3.10.1. 使用 SERVICE 指令在远程 SPARQL Endpoint 上构造高级查询。

5.3.10.2. 使用联合查询对本地图数据库或多个 SPARQL Endpoint 进行查询。

5.3.10.3. 使用 SERVICE 指令从不同的 SPARQL Endpoint 上获取数据。

5.3.11. 根据第三方工具与 APIs 的具体格式需要，**转换/操作** SPARQL 查询结果输出（RDF-XML、JSON）。

5.3.12. **阅读与了解**数据集的类与属性的高级描述，以便于编写查询。

5.3.13. 使用可用的工具、服务器与 Endpoint，对数据集进行查询。

5.3.13.1. 使用 Jena ARQ 命令行工具执行 SPARQL 查询。

5.3.13.2. 使用 ARQ 查询多个本地数据文件。

5.3.13.3. 使用 ARQ 评估本地数据上的查询。

5.3.13.4. 使用 Fuseki 服务器评估数据集上的查询。

5.3.13.5. 使用 Fuseki 查询多个数据文件。

5.3.13.6. 访问 DBpedia 的 SNORQL/SPARQL Endpoint，提交简单的查询。

5.4. 可视化 RDF 数据

5.4.1. **使用**公开可用的工具可视化数据。

5.4.1.1. 使用 Google 的 Fusion Tables 生成地图与图表。

5.4.2. 从大型数据集中**提取**数据，以用户友好的方式可视化。

5.4.3. 根据第三方工具与 APIs 的具体格式需要，**转换/操作** SPARQL 查询结果输出（RDF-XML、JSON）。

5.5. 基于 RDF 数据进行推理

5.5.1. **了解**推理的原理与实践。

5.5.1.1. 使用常见的蕴涵机制并了解它们的用法。

5.5.2. **了解**推理中形式化声明的定义域与值域的作用。

5.5.3. **了解**如何在集成的多元化数据集上进行推理。

5.5.4. **知道**网络本体语言（Web Ontology Language, OWL）在表达能力、推理方面以及涉及数据库与业务规则的应用中具有较好的适用性。

5.5.5. **了解** OWL Full 支持所有可用的构造，若不考虑推理性能问题的话，则 OWL Full 是最佳选择。

5.6. 评估 RDF 数据质量

5.7. RDF 数据分析

5.7.1. **会用**可获取的本体浏览工具，探索特定数据集中的本体。

5.8. RDF 数据处理

5.8.1. **知道** SPARQL 1.1 更新语言，对图数据库中的 RDF 图进行更改、创建与删除。

5.8.1.1. 使用 INSERT/DELETE 更改三元组。

5.8.1.2. 执行 INSERT/DELETE 操作之前，使用 CONSTRUCT 查询提前预览变化。

5.8.2. **知道** SPARQL 1.1 图数据库以 RESTful 风格，通过 HTTP 协议更新网络服务器上的图。

5.8.2.1. 使用 GET 从默认图或命名图中检索三元组。

5.8.2.2. 使用 PUT 向新图或替代已有图插入一个新的三元组集合。

5.8.2.3. 使用 DELETE 删除图。

5.8.2.4. 使用 POST 向已有图添加三元组。

5.8.2.5. 针对具体媒介类型使用恰当的语法，例如，Turtle。

5.8.3. **了解** SQL 查询语言（操作的是数据库表）与 SPARQL 查询语言（操作的是 RDF 图）之间的差异。

6. 关联数据应用的开发

6.1. 存储 RDF 数据