让公共数据更进一步

从农药登记数据的 RESTful 网络服务到与应用无关的本体（扩展版）

Katharina Albrecht [1], Kristoffer Janis Schneider[1], Daniel Martini[1]

摘要：施用农药时，必须考虑间隔要求。不过，这些要求必须通过人工方式确定并纳入考量。为帮助农户，PAM 项目开发了一套决策支持系统。该系统的一部分是由 KTBL 提供的间隔限制服务。针对这项服务，开发了一套流程：从 REST 接口读取农药登记数据，借助 RML 映射将读取的信息转换为本体，并通过 SPARQL 端点使其能够以与应用无关的方式供人查询。

关键词：关联开放数据、本体、语义网、RML、SPARQL。

1 引言

农药（PSM）的机械化施用已是当前的技术标准；田块通过 GPS 引导进行作业，借助施用图可以定义与边缘带结构、地表水体或城市结构之间的距离，并可通过分段关闭有针对性地施用农药。

然而，这一过程并不会自动完成，农户仍需从德国联邦消费者保护与食品安全局（BVL）提供的登记数据库（以下简称 PSM-DB）中的限制条款文本里手动确定必须遵守的距离，并将其录入施用图，或在施用过程中加以遵守。此外，还必须定期检查并进行更新。这一流程不仅耗时，而且容易出错。

为支持农户，Pesticide Application Manager（农药施用管理器，PAM）项目及其后续项目 PAMrobust、PAM3D 和 PAM-M 开发了决策支持系统 [Sc16] 。在项目实施期间，

[1] 德国农业技术与建筑协会，数字技术，Bartningstraße 49, 64289 达姆施塔特，k.albrecht@ktbl.de, https://orcid.org/0000-0002-3400-8102, [illegible]hneider@ktbl.de, d.martini@ktbl.de

KTBL 建立了一套流程，将限制数据存储在本体中，并着重利用关联开放数据（LOD）和标准来提供这些数据。其成果——间隔限制服务——以机器可读且独立于应用程序的形式表示 PSM-DB 提供的间隔要求，因此可将其自动传输到施用图中。在此过程中，批准数据通过声明式映射从现有关系数据库转换为以资源描述框架（RDF）[BG04] 三元组表示的陈述。该方法基于 [Be09]中所述的流程。随后，三元组会存储在三元组存储库中，通过 SPARQL 查询进行检索，并经由网络服务提供。

BVL 于 2020年参与了 tech4germany[2] 倡议，此后便通过 RESTful 接口提供 PSM-DB。因此，借助合适的技术工具即可访问各个表。这是提升批准数据机器可读性的重要一步。

2 知识现状

在过去的 PAM 项目中，开发了间隔限制服务，并不断根据新的发展进行调整和改进。[Ma15] 展示了将关系型数据映射到本体的初步映射过程，[Ma18] 则描述了首个 PAM 项目期间形成的 PSM 映射过程。数据通过映射描述从关系型数据库转换到 LOD 服务并予以提供。鉴于存在大量媒体断层和结构调整，后续的 PAMrobust 项目特别重视采用标准并减少媒体断层。这一改进后的映射过程总结于 [AMS19] 。PAM3D 对数据结构和内容呈现进行了完善，并以此为基础开展了下文所述、在 PAM-M 项目期间完成的工作。

图 1 展示了目前的映射过程。首先，以关系数据库（本例中为 Access 数据库）作为数据基础。该数据库包含大量表，这些表通过映射工具 db2triples（该工具实现了映射语言 R2RML [DSC12] ）转换为 RDF 三元组。随后，这些三元组通过 Apache Jena Fuseki[3] 三元组存储库发布，从而可以通过 SPARQL 查询 [HS13] 访问数据。此外，相关信息还通过 KTBL 的网络服务发布。

[2] https://tech.4germany.org/project/psm/[3] https://jena.apache.org/documentation/fuseki2/

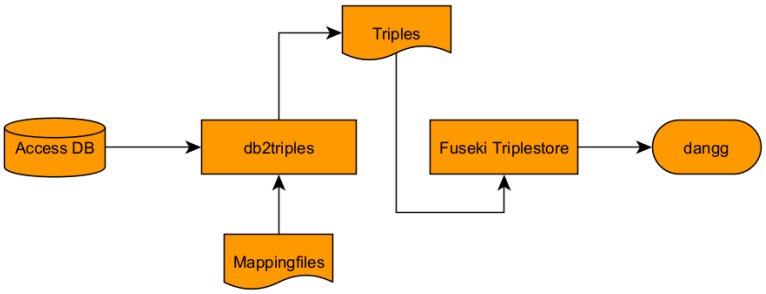


图 1：PSM 数据存储在 Access 数据库中，使用 db2triples 映射，并通过 Fuseki 三元组存储库经由 dangg 网络服务提供

迄今为止，所提供本体的时效性很大程度上依赖于 Access 数据库；该数据库需要人工提供、存储和处理。由于 PSM-DB 现已通过 REST 服务提供，这种依赖不复存在。因此，RESTful 提供将使未来能够完全自动化地读取数据。

对于 PAM 流程而言，这首先意味着需要改进现有流程。必须实施一个用于查询 REST API 的新流程，基于此构建的映射以及数据提供方式也必须满足新的要求。此外，db2triples 在 GitHub 上的源代码仓库已被原始提供者关闭，而且该工具目前显然不再继续开发或更新。因此，必须为映射寻找并采用一种替代方案。

3 PAM 流程

图 2 展示了新的 PAM 流程。PSM-DB 通过 RESTful 网络服务提供，由 PAM-DB 爬虫[4] 获取并存储在 SQLite 数据库中。相关内容将在第 3.1 节中说明。接下来的第 3.2 节将说明映射过程：使用映射文件和 rml.io 的映射工具 RMLMapper[5]，将 SQLite 数据库中的信息转换为 RDF 三元组。随后，第 3.3 节将说明如何通过 Fuseki 三元组存储提供数据，以及 KTBL 开发的网络服务。

[4] https://github.com/KTBL/PAM-DB-Crawler[5] https://github.com/RMLio/rmlmapper-java

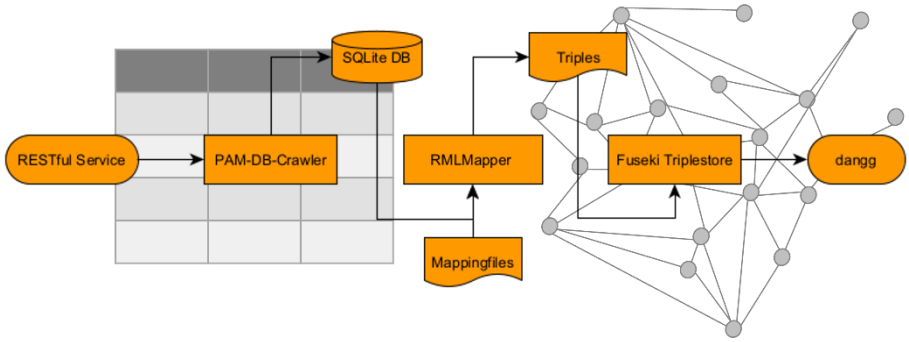


图 2：PSM 数据通过 RESTful 服务提供，可通过 PAM-DB 爬虫查询并存储在 SQLite 数据库

3.1 PAM-DB 爬虫

PSM-DB 的内容由 BVL 提供，可通过 REST API 查询。REST 服务[6] 的文档依据 OpenAPI 标准，通过 Swagger 提供。根据文档，该 REST 服务基于 Oracle REST Data Services 系统，可返回 JSON 数据格式的数据。

PSM-DB 中的每张表都提供一个格式为“/<表名>/”的接口。此外，还有“/stand/”接口，用于提供当前数据集发布日期等一般信息。所有接口都可以通过 HTTP GET 查询。不带其他参数时，会返回所有可用数据。根据表的不同，可以传入不同的参数，以查询单个条目或按特定条件进行筛选。由于大多数接口返回的数据量都很大（明显超过 100 个数据条目），因此采用了分页机制。这意味着每次调用 API 接口时，只会返回完整结果中的一部分。若要查询更多条目，可以在 URL 中传入经过编码的“offset”参数。默认情况下，每次最多返回 100 个条目。如有需要，可通过“limit”参数更改页面大小。由此，便可使用统一的系统查询所有 PSM 数据。此外，文档页面还定义了各接口所使用的数据模型。由于 REST 服务基于 Oracle 数据库，所使用的数据类型均声明为 Oracle 数据类型。

[6] https://psm-api.bvl.bund.de

为了继续使用此前为从旧 Access 数据库迁移数据而开发的 R2RML 映射，需要将 PSM 数据存储在关系型数据库中。借助 REST 服务，可以查询所有数据并将其存储在本地数据库中。随后，可通过 R2RML 映射将这些数据转换为本体。实现这一目标的一种方法是利用数据模型定义，手动创建合适的数据结构，通过 API 查询填充这些结构，并将查询到的数据存储在 SQLite 数据库中。因此，每个待查询表的实现都包括以下步骤：

- 实现数据结构 - 实现 API 查询，分别填充相应的数据结构 - 用于创建 SQLite 表的 SQL 语句 - 用于填充 SQLite 表的 SQL INSERT 语句

采用这种方法需要编写大量样板代码。虽然所有数据结构和 API 的构建与使用方式都相似，但由于内容各不相同，仍须分别实现各个元素。为减少样板代码，鉴于这些内容具有相似性，可以使用元编程生成器，而不是手动实现结构、SQL 语句和查询。此类生成器根据定义信息生成代码，涵盖上述从结构定义到 INSERT 语句的所有步骤。

针对这一方法，我们用 Python 实现了一个简单的生成器，它会生成 Python 代码。定义信息由 API 端点地址和模型定义（来自服务文档）组合而成。生成器据此生成数据结构、CREATE 和 INSERT 语句，以及用于查询 API 并保存已加载数据的代码。生成器以格式字符串为基础，其中包含作为文本的 Python 代码，并在相应位置插入特定于结构或 API 的信息。SQLite 表的类型信息取自文档中的模型定义。由于 Oracle 中的类型与 SQLite 类型不同，该项目为这两种类型体系提供了映射。生成的 Python 代码会保存为扩展名为“.py”的文本文件。另有一个不会自动生成的脚本，用作执行脚本来管理下载。该脚本会加载所有动态生成的结构定义，在必要时初始化数据库，然后启动数据加载流程。如果流程中断，当前状态会被保存，之后可以继续执行。

这种做法的一大优势是，可以快速应用对数据模型或接口地址的更改，并通过不同方式进行调整。一种方式是修改数据定义，然后

启动生成器，重新生成 Python 代码。另一种方式是手动调整已生成的 Python 代码，因为它以代码文本的形式存在。不过，采用后一种方式时，必须注意，重新启动生成器可能会覆盖所做的更改。

3.2 PSM映射

PAM-DB 爬虫运行后存储在 SQLite 数据库中的 PSM 数据，通过映射方法从关系型表示转换为 RDF三元组或本体。不过，迄今使用的 db2triples 映射工具已无法在 GitHub 上获取。因此，该工具不再提供更新或错误修复，有必要改用其他工具。经过深入调研，并考虑其是否符合标准以及是否可免费使用，我们将 rml.io 的 RMLMapper-java 确定为合适的工具。RMLMapper 是一个 Java库，可借助 RML 规则 [DHD20] 生成关联数据。RML 基于 R2RML 并对其进行了扩展。RMLMapper 支持多种本地和远程数据源，可将其用作映射的输入数据。不过，它并不支持 SQLite 或 Access 数据库。为实现连接，我们克隆了 RMLMapper 的 GitHub 代码仓库，并将特定的数据库连接功能集成到了源代码中。

为了测试使用 RMLMapper-java 进行映射，首先将现有的 Access 数据库用作输入。结果符合预期，因此得以实现 SQLite连接。下文将说明映射的各个步骤。

关系表示

图 3 展示了 PSM-DB 中两个表的部分内容，这些内容在 Access 数据库和 SQLite 数据库中的形式相同。AWG 表包含各项适应症的综合信息，并在映射过程中作为锚点，之后可通过它检索所有信息。其中，AWG_ID 是适应症的 ID。第二个表展示了 AWG_KULTUR表 的部分内容，即在相应适应症下应用农药的作物。该表是一个关联表，用于将适应症与作物对应起来。要从这两个表中生成一条陈述，即一个 RDF三元组，就必须对它们进行映射。为此，将采用以下映射。

AWG_ID	...
034028-60/07-004	...
034028-60/07-006	...
...	...

AWG_ID	KULTUR	...
034028-60/07-004	RUBID	...
034028-60/07-006	RUBFR	...
...	...	...

图 3 左：AWG 表的一个摘录；右：AWG_KULTUR 表的一个摘录

可识别性

映射的目标是创建一个 PSM 本体，其中包含适应症、药剂、作物、有害生物和间隔要求的所有必要信息。为此，必须确保各项信息都能被明确对应。

由于在关联数据领域，标识符还应当能够解析[7]，，正如 FAIR 原则 A1[8] 所述，因此无法解析的 URN（统一资源名称）无法使用。因此，本体通常使用 URI（统一资源标识符）。这些 URI 使映射过程中定义的各个资源都能够被唯一识别。唯一性还使得来自不同来源或映射的信息能够正确组合，并可任意扩展。反过来，也能够明确查询和获取信息。

PSM 本体对此前使用的 URI 系统进行了修订。现在，URI 与其所指内容的关联更加明确，也就是本例中的适应症和作物。本体的词汇表，即必要的类和属性，使用路径 http://srv.ktbl.de/data/psm/；具体实例则通过 http://srv.ktbl.de/data/psm/resources/ 定义，并根据类另外添加所谓的类前缀。在本例中，适用于 AWG_ID 或适应症的前缀是 ind_，适用于药剂所作用的作物的前缀是 crop_。部分 URI 目前已经可以解析。

[7] https://www.w3.org/wiki/LinkedData[8] https://www.go-fair.org/fair-principles/metadata-retrievable-identifier-standardised-communication-protocol/

Das 映射文件

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix rml: <http://semweb.mmlab.be/ns/rml#>.
@prefix psm: <http://srv.ktbl.de/data/psm/>.
@prefix ql: <http://semweb.mmlab.be/ns/ql#>.

@prefix d2rq:<http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#> .
<#DB_source> a d2rq:Database;
    d2rq:jdbcDSN "jdbc:sqlite://Pfad/Name;
    d2rq:jdbcDriver "org.sqlite.JDBC";
    d2rq:username "";
    d2rq:password "".

<#TriplesMap1> a rr:TriplesMap;
    rml:logicalSource [
        rml:source <#DB_source> ;
        rr:sqlVersion rr:SQL2008;
        rml:referenceFormulation ql:CSV;
        rml:query """ SELECT AWG_ID, REPLACE(AWG_ID, "/", "-") AS URLID FROM AWG
        """
    ];
    rr:subjectMap [
        rr:template "http://srv.ktbl.de/data/psm/resources/ind_{URLID}";
        rr:class psm:Indication;
    ];
    rr:predicateObjectMap [
        rr:predicate psm:appliedOnCrop;
        rr:objectMap [
            rr:parentTriplesMap <#TriplesMap2>;
            rr:joinCondition [
                rr:child "AWG_ID";
                rr:parent "AWG_ID";
            ];
        ];
    ].
    <#TriplesMap2> a rr:TriplesMap;
    rml:logicalSource [
        rml:source <#DB_source> ;
        rr:sqlVersion rr:SQL2008;
        rml:referenceFormulation ql:CSV;
        rml:query """ SELECT AWG_ID, KULTUR FROM AWG_KULTUR """
    ];
    rr:subjectMap [
        rr:template "http://srv.ktbl.de/data/psm/resources/crop_{KULTUR}";
        rr:class psm:Crop;
    ].
```

图 4：映射文件示例：Map_Crop.ttl 摘录

上述示例的映射如图 4 所示，其中摘录了 Map_Crop.ttl 的部分内容。使用 R2RML 或 RML 描述的映射文件内容本身构成一个采用 Turtle 语法的 RDF 图 [PC13]。PSM-DB 的完整映射文件可在 KTBL 的 GitHub 仓库 RML- Mapping[9] 中找到。

所有映射文件都包含一个文件头，其中列出了文件中使用的所有前缀。这些前缀使内容的呈现更加紧凑、易读。示例中使用了 RML 规定的前缀 rr、rml、ql 和 d2rq，以及前缀 psm；psm 用于所有与植物保护产品相关的内容。

接下来是声明部分，其中包含所谓的三元组映射。三元组映射可以单独、相互独立地定义，也可以彼此关联。为此，需要通过连接条件定义父子关系。

由于 RML 建立在 R2RML 之上，因此映射文件之间的差异通常很小。不过，与只能应用于关系数据库的 R2RML 映射不同，RML 可以使用多种数据格式作为输入文件。因此，需要“源标头”来指明输入文件的类型。在本例中， <#DB_source> 描述了所创建的 SQLite 数据库。此外，还会指定相应的 JDBC 驱动程序。此数据库无需用户名和密码。

<#TriplesMap*1*> 描述了第一个映射。rml:logicalSource 包含有关所用输入文件、SQL 版本（SQL 2008）和引用形式（CSV）的信息。由于需要访问数据库，因此还可以通过 rml:query 为每个三元组映射定义 SQL 查询。在映射过程中，还必须考虑 Access 和 SQLite 的 SQL 查询语言细节。由于一些间隔要求只存在于限制条款文本中，因此必须使用 SQL 查询。例如，可以借助正则表达式（RegEx）从限制条款文本中提取具体内容。这样便能整理 PSM 数据，使需要遵守的间隔以机器可识别的形式提供，也就是将数值与单位分开，仅提供数值。

在一个三元组映射中，必须始终定义一个 SubjectMap 和至少一个谓词对象映射。顾名思义，SubjectMap 表示 RDF 三元组的主语。在示例中，主语是适应症。由于 AWG_ID 字符串中的斜杠（‘/’）是 URL 编码中的保留字符，因此在 SQL 查询中会将其替换为连字符

[9] https://github.com/KTBL/RML-Mapping

。通过 template 可确定主语的 URI，并使用 rr:Class 将类 psm:Indication 指派给主语。

在 SubjectMap 之后定义 PredicateObjectMap。对象可以是资源、字面量或空白节点。谓词 psm:appliedOnCrop 表示，在该适应症中使用的植物保护产品会应用于特定作物。为此，使用连接条件，在两个表中查找指定表列（AWG_ID）对应的共同条目。

<#TriplesMap2> 展示了作物的创建方式。rml:logicalSource 的定义方式与<#TriplesMap1> 相同。再次通过 SQL 查询定义对作物表的访问。必须同时读取作物列和 AWG_ID，否则连接条件就无法建立三元组映射之间的关系。

映射结果

使用 java -jar rmlmapper-VERSION-all.jar -m Map Crop.ttl -o Res Crop.ttl -s turtle 执行映射，并将结果三元组保存在 Turtle 格式的文件 Res_Crop.ttl 中。随后将这些文件加载到三元组存储库中。上述示例的映射输出如下：

@prefix psm: <http://srv.ktbl.de/data/psm/> .

@prefix psmr: <http://srv.ktbl.de/data/psm/resources/> .

psmr:ind 034028-60-07-004 a psm:Indication;

psm:appliedOnCrop psmr:crop RUBID .

psmr:ind 034028-60-07-006 a psm:Indication;

psm:appliedOnCrop psmr:crop RUBFR .

psmr:crop RUBFR 一个 psm:Crop.

psmr:crop RUBID 一个 psm:Crop.

各行都构成主语-谓语-宾语形式的逻辑陈述。由此可见，psmr:ind_034028-60-07-004 是类 psm:Indication 的一个实例。分号表示下一行具有相同的主语，也就是说，psmr:ind_034028-60-07-004 应用于作物 psmr:crop_RUBID。对 psmr:crop_RUBFR 也作出了相同的陈述。反过来，psmr:crop_RUBID 和 psmr:crop_RUBFR 都是类 psm:Crop 的实例。

3.3 三元组存储库与部署

要使生成的本体可供访问，必须将这些三元组加载到三元组存储库中。该三元组存储库通过 SPARQL 端点提供访问，之后便可使用标准化查询语言 SPARQL 对其进行查询。

数据发布使用 Apache Jena Fuseki 服务器。Fuseki 可以运行在台式电脑、虚拟机（VM）或 Docker[10]容器中。最后一种方式对于系统的可移植性尤为重要，因为 Docker 容器独立于主系统的实际硬件。

SPARQL 查询既可以直接在 Fuseki 的用户界面中执行，也可以通过 HTTP 请求发送。图 5 展示了用户界面中的一个示例。示例中执行了一个 DESCRIBE 查询：DESCRIBE psmr:ind 034028-60-07-004。该查询会返回与所查适应症直接相关的所有信息¯；也就是说，所查资源是一种适应症，并且该适应症适用于 RUBID 培养物。除了查询结果外，还列出了本体中包含的所有前缀。

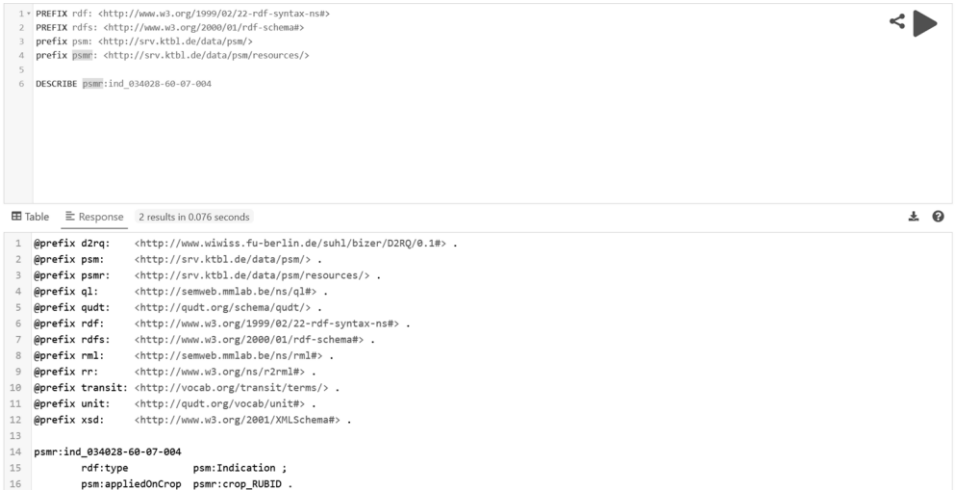


图 5：Fuseki 界面中 DESCRIBE 查询的返回示例

此外，还可以定义 SELECT 查询，以执行简单查询并获得类似 SQL 的表格输出；也可以定义 CONSTRUCT 查询，借助

[10] https://www.docker.com/

它们可以有选择地创建子图。如果端点的实现支持，也可以使用 UPDATE 查询。示例中使用 Turtle 作为返回格式，但还可以使用其他多种格式，包括 JSON-LD、N-Triples 或 XML。

4 结果与讨论

针对间隔限制服务，我们用 go[11]，实现了一个专用网络服务，名为“dangg”。该服务会以 JSON-LD[12] 格式返回某一特定适应症所需考虑的所有间隔要求等信息。图 6 展示了返回内容的一个节选。该节选显示了适应症 034028-60/07-004 及其对应的作物、病原体、药剂、施药时间、桶混、限制条件和间隔要求。诸如 _:b4 的引用表示空白节点，它们提供了其他内容。此表示形式由 Fuseki 提供。返回结果中的键值对包含各个资源的唯一标识符，因此，在完成映射后，可以查询有关药剂或作物/病原体的更多信息。


```
}, {
  "@id" : "psmr:ind_034028-60-07-004",
  "@type" : "psm:Indication",
  "agent" : "psmr:ex_034028-60",
  "applicationTime" : "psmr:at_BF",
  "appliedOnCrop" : "psmr:crop_RUBID",
  "appliedOnPest" : "psmr:pest_PHRARU",
  "considerBufferConstraint" : "_:b14",
  "constraint" : [ "psmr:cons_NW605", "psmr:cons_NT102", "psmr:cons_WW7091", "psmr:cons_NW606", "psmr:cons_NW701" ],
  "scopeOfApplication" : "psmr:foa_O",
  "usableApplicationRate" : "_:b6",
  "label" : "034028-60/07-004"
}, {
```


图 6：示例适应症的 JSON-LD 返回内容节选

通过 RESTful 接口提供 PSM-DB，显著简化了 PAM 流程。借助 PAM-DB 爬虫，可以灵活地将数据存储到 SQLite 数据库中并进行映射。RMLMapper 是 db2triples 的合适替代方案，也为进一步改进提供了空间，例如将其他格式的数据源纳入映射。使用 Fuseki 也证明是行之有效的。总体而言，PAM 流程如今提供了一种可靠的方式，可独立于具体应用，以机器可读的形式提供农药登记数据。

[11] https://go.dev/[12] https://json-ld.org/

5 展望

BVL API 使 PAM 流程能够实现完全自动化。PAM-DB 爬虫的开发是实现这一自动化的关键一步。今后可以通过批处理作业实现 PAM-DB 爬虫自动获取 PSM-DB 的每月更新，从而及时进行映射。不过，为避免冗余和错误信息，还需要制定合适的处理方案。RMLMapper 支持以多种格式保存三元组。除了易于人类阅读的 Turtle 格式外，还包括 ntriples[13] 和 nquads[14]等格式。这些表示形式有助于版本控制，因为它们支持逐行比较陈述。

此外，借助 RMLMapper，还可以通过 SPARQL 更新将创建的映射自动发布到 SPARQL 端点，从而在未来省去手动上传这一步骤。不过，为了确保稳定提供服务并保障安全运行，仍需进一步考虑如何构建稳定的基础设施。

目前使用的 Fuseki 版本为 4.3.2。。在下次更新后，JSON-LD 表示中将能够返回带前缀的键，从而实现与本体更紧密、更便捷的关联。

为尽可能减少对不同基础设施的依赖，整个流程将借助 Docker 实现。此外，这也便于其他用户交换或共享流程模块。

该项目（FKZ 321-06.01-28-RZ-5-105）的资助来自农业年金银行提供的药剂。项目由农业年金银行通过创新促进计划负责管理。

参考文献

[AMS19] Albrecht, K., Martini, D. 和 Schmitz, M.：《植物保护领域的关联开放数据——以机器可读形式提供适应症和间隔要求所需的信息》。载于：Meyer-Aurich, A., Gandorfer, M., Barta, N., Gronauer, A., Kantelhardt, J. 和 Floto, H.（编），第39届 GIL 年会，《小规模区域农业企业的数字化——自相矛盾？》。波恩：德国信息学会，第19-24页，2019年。[Be09] Berners-Lee, T.：《语义网上的关系数据库》。万维网联盟，2009年。http://www.w3.org/DesignIssues/RDB-RDF.html，访问日期：2022年10月12日。

[13] https://www.w3.org/TR/n-triples/[14] https://www.w3.org/TR/n-quads/

[DSC12] Das, S., Sundara S., Cyganiak, R.：《R2RML：关系数据库到 RDF 映射语言》，https://www.w3.org/TR/r2rml/，访问日期：2022年10月12日。[HS13] Harris, S., Seaborne, A.：《SPARQL 1.1 查询语言》。万维网联盟，2013年。http://www.w3.org/TR/sparql11-query/，访问日期：2022年10月12日。[Ma15] Martini, D. 等：《KTBL 规划数据迈向未来——通过关联开放数据提供》。载于：Ruckelshausen, A.; Schwarz, H.-P.; Theuvsen, B.（编），《2015年农业、林业与食品经济信息学》，盖森海姆。GI 版信息学讲义第238卷，波恩，第105-108页，2015年。[Ma18] Martini, D.：《异构数据集上的网络服务——以 KTBL 规划数据为例的实现方法》。载于：Ruckelshausen, A., Meyer-Aurich, A., Borchard, K., Hofacker, C., Loy, J.-P., Schwerdtfeger, R., Sundermeier, H.-H. F. 和 Theuvsen, B.（编），第38届 GIL 年会，《数字市场与平台》。波恩：德国信息学会，第155-158页，2018年。[DHD20] De Meester, B., Heyvaert, P., Delva, T.：《RDF 映射语言（RML）》。IDLab – imec – 根特大学，2020年10月6日。https://rml.io/specs/rml/，访问日期：2022年10月13日。[PC13] Prud’hommeaux, E., Carothers, G.：《Turtle——简洁 RDF 三元组语言》。万维网联盟，2014年2月25日。http://www.w3.org/TR/turtle/，访问日期：2022年10月12日。[Sc16] Scheiber, M. 等：《植物保护应用管理器（PAM）：自动纳入间隔要求。实际演示与田间测试结果。自动纳入间隔要求。实际演示与田间测试结果》。载于：Ruckelshausen, A., Meyer-Aurich, A., Rath, T., Recke, G. 和 Theuvsen, B.（编），《2016年农业、林业与食品经济信息学》。波恩：德国信息学会，第177-180页，2016年。