Off-the-shelf CRM with Drupal
A case study of documenting decorated papers

Athanasios Velios · Aurelie Martin

Received: date / Accepted: date

Abstract We present a method of setting up a website using the Drupal CMS in order to publish CRM data. Our setup requires basic technical expertise by researchers who are then able to publish their records in both a human accessible way through HTML and a machine friendly format through RDFa. We begin by examining previous work on Drupal and the CRM and identifying useful patterns. We present the Drupal modules that are required by our setup and we explain why these are sustainable. We continue by giving guidelines for setting up Drupal to serve CRM data easily and we describe a specific installation for our case study which is related to decorated papers alongside our CRM mapping. We finish with highlighting the benefits of our method (i.e. speed and user-friendliness) and we refer to a number of issues which require further work (i.e. automatic validation, UI improvements and the provision for SPARQL endpoints).

Keywords Drupal · CIDOC-CRM · decorated paper

A. Velios
Ligatus Research Centre
CCW Graduate School
University of the Arts London
16 John Islip Street
London, SW1P 4JU
U.K.
Tel.: +44 (0)20 7514 6432
E-mail: a.velios@arts.ac.uk

A. Martin
Ligatus Research Centre
CCW Graduate School
University of the Arts London
16 John Islip Street
London, SW1P 4JU
U.K.
Tel.: +44 (0)20 7514 6432

1 Introduction

This paper describes a method for collecting and publishing data from cultural heritage projects based on CRM mappings and using popular software which does not require expert technical knowledge. We believe that a method based on easy to use tools is critical for the wider adoption of the CRM.

The benefits of publishing data using the CRM are well-known to the readers of this special issue. They have been outlined many times in the past (for example [2, 4]) and more recently (for example [15, 14]). The CRM is a mature standard for information exchange and integration of data from cultural heritage projects. Despite these strengths and its long history, the CRM has not yet enjoyed the popularity it deserves. The global platform for sharing data is the Semantic Web and the Linked Open Data (LoD) initiative. One of the reasons why the CRM is not being used more is that the Semantic Web does not yet offer efficient tools to accommodate CRM data easily and as such technical implementations for the CRM are often a barrier for researchers. Humanities researchers without a background in computer science may take the time to understand the CRM concepts, they may even attempt to map their records to these concepts, but they rarely make the extra step to implement their mapping and publish their data online. This extra step often requires the use of unfamiliar software and typically involves complex setups which are difficult for a non-expert and prone to frustration and errors. The mapping process has recently been made accessible to non-technical researchers with the development of a new tool called 3M (Mapping Memory Manager) which can receive data from a database and export CRM data[5]. Exported

data can then be uploaded onto a triple store [1] and queried/browsed often through a standard web interface. This is not a trivial task and many humanities projects do not have the technical support to undertake it. Moreover the resulting online resource lacks identity since it is primarily designed for the consumption of data by other software and it looks inaccessible to humans.

In this paper we propose the use of popular off-the-shelf software which can enable researchers to undertake the otherwise technically complicated last step in order to publish their project data both on a human-friendly website and also as Linked Data. We test our setup on the study of decorated papers. This is a representative example of the use of the CRM for modeling descriptions of historical objects. Decorated papers are decorative objects used as wallpapers, box and furniture linings or as book covers or endleaves where a modest number of techniques and materials can be used to produce a huge variety of results. The study of decorated papers is current. They have been a subject of growing interest as suggested by recent publications (e.g. [8, 9, 11, 10, 12, 16]) and therefore we hope that our case study will be directly applicable to current research in the field.

1.1 Off-the-shelf CRM

The CRM has been showcased by flagship projects funded by, among others, European and American agencies. These projects have produced software which proved that the CRM is an ideal solution for interoperability of data in cultural heritage but they have created an expectation that implementing projects based on the CRM will be expensive, due to the necessity of technical expertise which the humanities researchers do not always have. However, the CRM community has recognised that in order for the CRM to become more widely adopted, researchers need to be empowered to undertake projects themselves following guidance by the CRM community.

At the same time researchers have become familiar with web applications because of the popularity of online research resources as well as social media etc. Researchers are confident using point-and-click web interfaces and switching from one resource management system to another requires a not unreasonable amount of effort on their behalf often without technical support. In that respect empowerment of researchers has already started at that level.

By adopting an off-the-shelf system for our method we ensure that it can be implemented using the same popular tools by other researchers following a do-it-yourself approach with support when required. A number of systems could be considered as suitable options for a popular off-the-shelf system, including Wordpress[2] and MediaWiki[3]. We have focused on Drupal [4], a user-friendly Content Management System (CMS[5]) not only because it provides a generic framework for publishing data online but also because our technical team already provides support for Drupal installations on our servers. Previous work has reported on the use of the CRM in other systems (e.g. [17]). Comparing the suitability of popular systems for implementing CRM is beyond the scope of this text.

Drupal requires basic technical knowledge and has been extremely popular over the years[6], including in humanities projects. Drupal is an open source project with a strong community of users and developers. It is a modular system with each module offering part of its functionality. Drupal is a collection of so-called *core modules*, but its functionality can be extended using contributed modules. At the time of writing drupal.org reports 11,476 contributed modules for Drupal version 7.0. In the U.K., Drupal hosting costs as low as about £3.00 per month. The low cost and user-friendliness of Drupal makes it an attractive option for researchers who wish to publish their data online. This paper shows how Drupal can be set up by configuring a small number of popular modules to publish CRM data as Linked Data (http://www.w3.org/standards/semanticweb/data) by non-experts. This method can help the wider adoption and publication of CRM data. We begin by explaining the current state of the study of decorated papers and our new objectives which dictate the requirements for our Drupal setup.

1.2 The study of decorated papers

The study of decorated papers is difficult because of the lack of useful tools to identify them quickly, describe them accurately and help attribute their provenance.

[1] https://en.wikipedia.org/wiki/Triplestore

[2] https://wordpress.org/

[3] https://www.mediawiki.org/

[4] http://drupal.org/

[5] https://en.wikipedia.org/wiki/Content_management_system

[6] It is difficult to source accurate statistics about market share for Drupal but it regularly appears in the top 3 most popular CMSs in relevant listings (e.g. http://www.opensourcecms.com/general/cms-marketshare.php, http://w3techs.com/technologies/overview/content_management/all, http://www.datanyze.com/market-share/CMS/).

1.2.1 Limitations in the study of decorated papers

The main way of communicating results in the study of decorated papers is through the publication of monographs referring to historical analysis and to typical examples of papers (e.g. [8, 9]). The format of the monograph poses limitations:

- Printed volumes restrict the number of pictures reproduced. Only typical examples of papers are included which therefore prevents the study of the large number of variations.
- The structure of a monograph requires a consistent but not standardised terminology. As the effort towards a standardised terminology in the description of decorated papers is a recent development, authors still use different terms to describe similar papers, which makes their identification and comparison complicated. A recent publication of a manual to describe decorated papers in three languages [16] is a great tool, however the proposed terms are still not widely used by authors who prefer to continue using their own terminology.

A few decorated paper databases have been published online, but many suffer from problems such as:

- They are not interoperable and retrieval of records is difficult because they do not rely on controlled vocabularies and/or standardised terminologies.
- The terms used often mix different concepts together, such as the materials used with the techniques or resulting patterns. This makes semantic interoperability impossible.
- Information about the production of decorated papers is regularly omitted or recorded inaccurately, thereby preventing accurate attribution of provenance. For example, a paper may be tagged with the terms 'French' and '18th century', but it is unclear what these tags refer to i.e. the paper itself or the book that the paper is part of? Evidence of why the item has been tagged as such are also missing. Such unreliable data may lead to incorrect conclusions.

A resource with a solid semantic structure to express this data has yet to be published and our project makes a good first step towards that aim by adopting a thesaurus of terms (based on the Simple Knowledge Organisation System - SKOS [13]) and by identifying the important events in the history of a decorated paper which need to be recorded separately in order for provenance to be established.

1.2.2 Case-study objectives

The aim of this case study is to allow research on the provenance of decorated papers based on evidence captured in photographs and described explicitly as separate data. We augment these descriptions by building semantic relationships across data, which clarifies much of the ambiguous tagging of other resources. The number of photographs we can publish is only limited by the amount of disk space available on our server. We welcome submissions from external parties and therefore we will rely on customisations of the user-friendly Drupal interface to encourage submissions. We intend to describe the papers with terms from controlled vocabularies to create a resource that is interoperable and machine-searchable (see next section). We rely on the flexibility of SKOS to provide alternative labels for concepts to avoid issues such as the use of different terms for identical concepts.

1.3 Terminology based on the CRM

Like in many other humanities fields, issues concerning terminology have been central to the study of decorated papers. The recent publication of the Language of Bindings (LoB) thesaurus (http://www.ligatus.org.uk/lob) aims to address some of these issues. LoB is a thesaurus based on SKOS and the core idea of broader and narrower terms. LoB concepts are arranged in hierarchies. The top terms of these hierarchies match CRM entities. For example one hierachy's top term is *materials* and any concept in that hierarchy can be used as a CRM *E57 Material*. Another is *techniques* and can be used as a CRM *E29 Design or Procedure*. Matching thesaurus top terms with CRM entities is beneficial because it allows easy integration of lookup fields inside a schema from distinct hierarchies of the thesaurus. In this case study we make extensive use of LoB thus addressing the issue of terminology standardisation. In other cases other thesauri where concepts/terms are identified with URIs can be used.

2 CRM in Drupal

Drupal was developed as a CMS for the publication of web-pages to be viewed by humans. Recent developments in Drupal allow the publication of data within HTML web-pages in RDFa [7] which is a flavour of the Resource Description Framework (RDF) [3] that can be embedded in HTML and therefore suitable for machine consumption.

Drupal is a reasonable choice for an off-the-shelf system for CRM data because:

- It features familiar tools which empower non-technical humanities researchers to publish their data without expensive technical support.
- It can publish structured data as well as nice looking web-pages which means that it is complete in the sense that one dataset can be used and maintained for both uses of the data thus avoiding periodical exports from one database (e.g. XML db) to another (e.g. triple store).

Following some introductory notes on the basic concepts of Drupal we describe the modules required for our setup and we offer some generic guidelines which may be applicable to other projects.

2.1 How Drupal works

A good starting point for understanding Drupal's structure is the documentation pages[7] on drupal.org. Drupal includes a range of standard tools for user management, content feeds, aggregation, theming etc. Over here we will only outline the ones that are relevant to our method:

- A *Drupal entity* is an abstract way to refer to any of the structures that modules in Drupal produce (see next).
- A *node* is a piece of content which receives a separate identifier in the Drupal database. Typically a node can be rendered as a web-page. The web-page can be found in a URL which is produced based on the database identifier.
- Drupal does not make any assumptions on the type of content users publish. Nodes have *content types* which can be specified by the user. We could consider the content types as parts of a schema. A user produces a new node by filling in a form which has been produced based on the content type specification.
- The specification of content types involves the definition of *fields* which may belong to more than one content types. Fields can be considered as schema elements. Fields can hold data and there is a range of different types of fields available (text, number, file etc.).
- A *taxonomy vocabulary* is a way to group special content called *taxonomy terms*. A *taxonomy term* is content which can be used to organise other content. Typically a content type can have a *term reference field* which links a taxonomy term to a node (this functionality is provided by the core *Taxonomy* module).

Figure 1 shows how the above Drupal concepts are related. It shows two content types based on which we can produce two sets of nodes. Each item in these sets receives an ID which can be used to form the URL for this item. Fields 1 and 2 are term reference fields which store links to the vocabulary terms.

Drupal features flexible tools for the customisation of the appearance of a web-page (theming tools). We can confirm that Drupal can produce a variety of aesthetic result in the way that records can be presented but referring to the theming tools is beyond the scope of this paper. More information can be found here: https://www.drupal.org/documentation/theme.

2.2 RDF in Drupal

With version 7.x, Drupal has embedded RDF technologies in its core modules enabling websites to publish Linked Data. Current documentation on RDF in Drupal can be found on https://www.drupal.org/node/1089804. Corlosquet et al. [1] explained the principles of this development which can be summarised here:

- Every Drupal node has a unique URL. These could be considered as resolvable HTTP URIs as required for Linked Data.
- When defining a content type Drupal requires the user to indicate an associated semantic class. This class will be assigned to the nodes produced based on this content type. Every new node receives an `rdf:type` triple defining that class. For example the `rdf:type` of a Drupal user account page can be `foaf:person` (http://xmlns.com/foaf/spec/#term_Person).
- When defining a field within a content type Drupal requires the user to indicate an associated semantic property. The domain of this property is the content type class, while the range of this property is the data held in the field. For example a node `foaf:person` can have a field with a property `foaf:name` (http://xmlns.com/foaf/spec/#term_name) which for a specific node can hold the value `"Martin Doerr"`.

Figure 2 shows these basic Drupal concepts and their consideration in relation to Linked Data and RDF triples.

As mentioned above the RDF triples resulting from each field are expressed as RDFa within Drupal's HTML pages. The underlying data used to produce these triples are stored in Drupal's relational database. Currently Drupal does not feature an in-built triple store. Drupal's standard installation comes with two content types

[7] https://www.drupal.org/node/19828

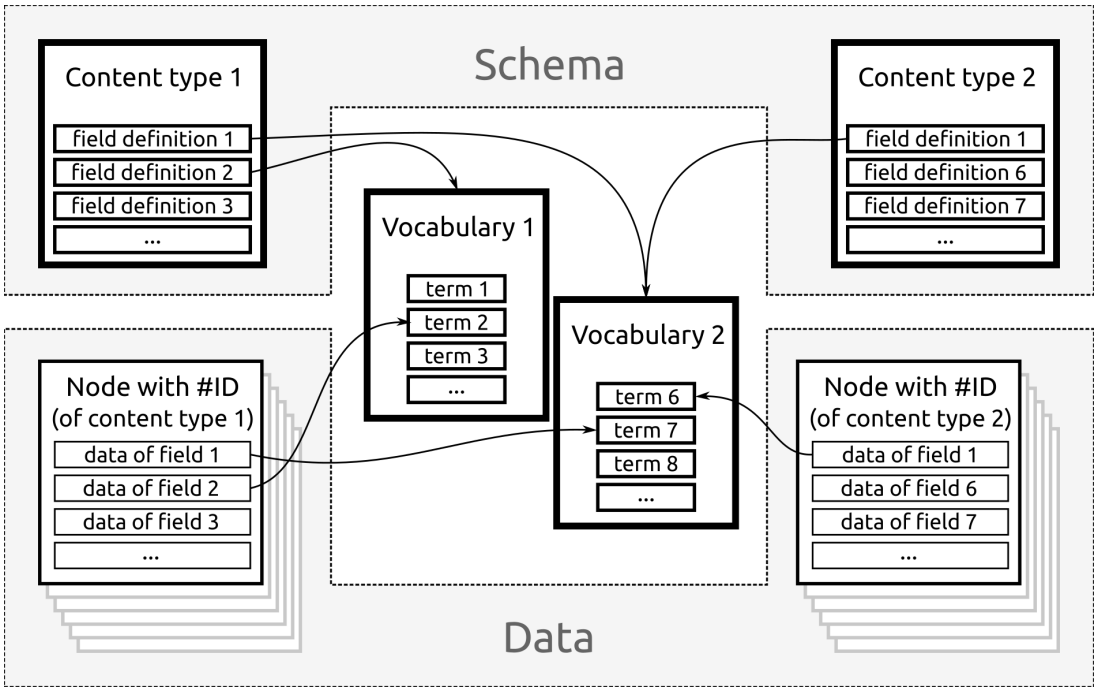

Fig. 1 Content, content types and relationships to vocabularies in Drupal.

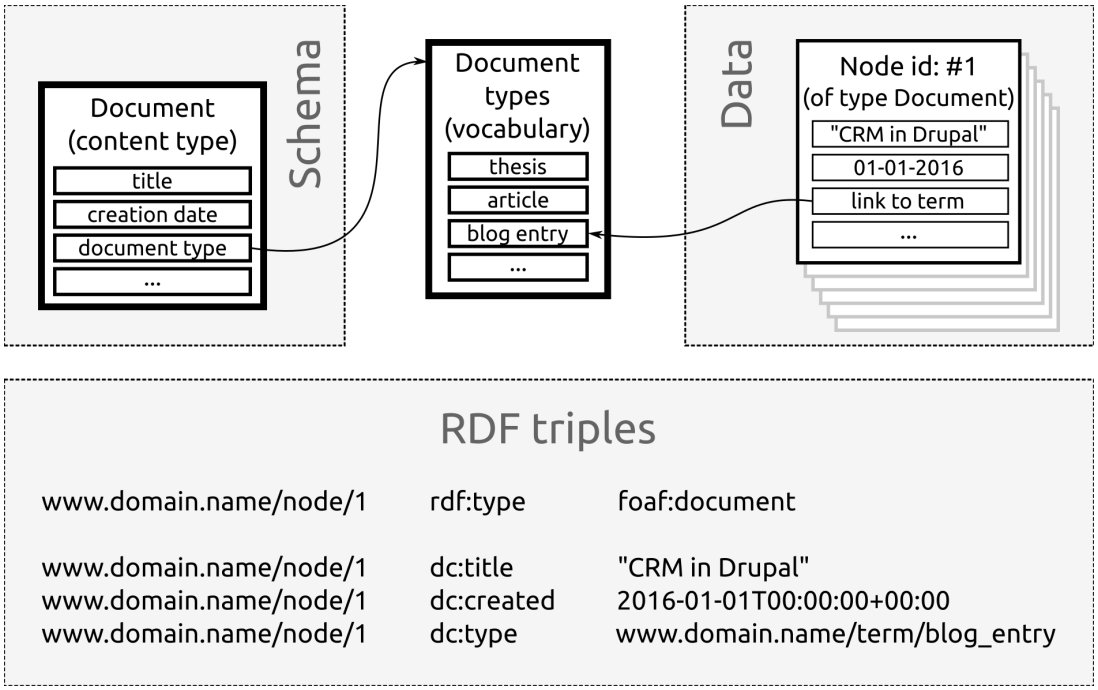

Fig. 2 Content, content types and relationships to vocabularies in Drupal.

with pre-defined fields and mappings to several ontologies including Dublin Core (DC), Friend of A Friend (FOAF) and Semantically-Interlinked Online Communities (SIOC). In the next section we examine other attempts to provide similar functionality for publishing CRM data in Drupal and we explain in which cases these solutions are useful.

2.3 Previous CRM implementations based on Drupal

2.3.1 WissKI

WissKI (http://wiss-ki.eu/) is a collaborative project aiming to produce a virtual research environment for cultural heritage using the CRM [18]. WissKI software

is based on Drupal and the WissKI team have produced a number of contributed modules to implement the system. Three important ones are:

- The WissKI Core module allows the production of content types and nodes based on semantic rules defined in the CRM. This assigns a CRM entity to each Drupal content type and produces the possible fields associated with this entity which may include relationships with other entities. Expressing such relationships in the CRM may involve non-trivial mappings to CRM entities and the production of a number of intermediate nodes. WissKi provides functionality to do this automatically by examining the CRM semantic rules as provided by the Erlangen CRM project (http://erlangen-crm.org/) thus making sure that all produced statements are semantically correct. This greatly simplifies the use of the CRM for researchers without technical expertise.
- The WissKI SALZ (Store Abstraction Layer Zero) module allows project data to be stored on a triple store, in addition to Drupal's own relational database. This allows the additional benefits of using a triple store for querying CRM data including any inference engines.
- The WissKI Name Authority API module allows the integration of any SKOS vocabulary with WissKI.

WissKI is an implementation of the CRM which significantly extends Drupal's functionality. Apart from the three core modules described above, WissKI's rich functionality requires about 30 contributed modules out of which about 20 are provided by the WissKI team. WissKI installations can customise Drupal to a great extent. In our project we decided not to adopt WissKI because of its heavy customisations and the dependency on many modules. In some cases these may lead to difficulties with upgrading to newer versions of Drupal and also with integrating with other Drupal modules which are not aware of the WissKI customisations. WissKI provides a usable system for the average humanities researcher without compromising on the benefits researchers get from using the CRM in terms of semantic descriptions and semantic searching and as such it is a highly recommended option for CRM implementations. In our case, we have opted for a simpler system with fewer and popular modules/dependencies which is an advantage when upgrading and integrating, but at the cost of reduced functionality and in some cases reduced usability.

2.3.2 Relation module

Guillem et al. [6] describe another way of implementing the CRM in Drupal for their project *Behindthe-Lines*. They are using the *Relation* contributed module (https://www.drupal.org/project/relation) to re-create the semantic relationships of the CRM within the Drupal database. The Relation module allows linking two Drupal entities (called endpoints) and typically these links can be considered as CRM properties and RDF predicates. *Relation* can be used to replicate the core RDF structures of Drupal since it can model links between Drupal entities in the same way as RDF statements (subject - predicate - object). *Relation* offers additional functionality. For example, it allows each link/statement to be a Drupal entity itself and therefore become an endpoint of another link/statement. This allows statements about statements, i.e. the evaluation of statements which was a core objective in Behindthe-Lines. The Relation module has been received with enthusiasm from the Drupal community and Guillem et al. report positively about it.

One concern about this module is that it replicates Drupal's core mechanism for expressing RDF and if the module's extra functionality is not required then it could be considered redundant. Another concern is the extra work required to map the Relation entities to the core Drupal RDF so that they can be used as Linked Data (a relevant 'issue' to do that automatically has been pending for a number of years https://www.drupal.org/node/1132724). This indicates more clearly why the Relation module could be considered redundant in simple use case scenarios. In our case, we do not require the extra functionality offered by Relation but recognise that this may be useful elsewhere.

2.4 Off-the-shelf approach for CRM in Drupal

We emphasise here that our proposed solution is based on standard core Drupal functionality alongside some essential contributed modules which are well-supported and almost guaranteed in the long term. These are currently maintained by leading Drupal community members.

2.4.1 Important contributed modules

Pre-requisite modules *Entity API* (https://www.drupal.org/project/entity) and *Chaos tools* (https://www.drupal.org/project/ctools) are pre-requisite modules to a number of contributed modules because they offer convenient development tools. They are extremely popular modules with 3.5 and 5.6 million downloads respec-

tively and hundreds of thousands of working installations as of December 2015. Our solution depends on these but their popularity ensures they are sustainable alongside Drupal core modules.

RDFx (https://www.drupal.org/project/rdfx) The *RDFx* module offers a number of additional functions to a Drupal installation including serialisation of RDF data and importing RDF ontologies for use within Drupal. Within our solution we use this module because it offers a UI for manually mapping content types and their fields to ontologies (unlike WissKI which offers tools for automating this process). This means that in addition to the standard Drupal mappings to FOAF, SIOC and DC we can map Drupal entities to CRM entities. In our solution RDFx is only required to provide a mapping UI and it does not alter in any way the default structure of RDF in Drupal.

Web taxonomy (https://www.drupal.org/project/web_taxonomy) The *Web taxonomy* module allows querying remote vocabularies and thesauri from within a Drupal field. This is useful for storing references to Linked Data hosted on remote resources, but if remote querying of data is not required, this module can be ignored. As of December 2015 there are plugins for the Web taxonomy module which allow querying DBpedia entries as well as the Getty vocabularies and, for our case study, the Language of Bindings Thesaurus.

Entity reference (https://www.drupal.org/project/entityreference) The *Entity reference* module is another extremely popular module with 1.3 million downloads and as of December 2015 with 257,894 active installations. This module allows linking pairs of entities in Drupal. For example Drupal core offers linking node entities to term entities, but Entity reference allows linking of nodes to other nodes or users. This module is essential for our solution as it offers a way of building links between content types.

Taking into account the functionality offered by the aforementioned modules, in the next section we provide some guidelines for producing CRM Linked Data using Drupal.

2.4.2 Guidelines for Drupal schema

Both the WissKI and BehindtheLines projects adopt the idea that content types can correspond to CRM entities. Our solution also adopts this position. Naming a content type can be done based on the specific business case and it does not have to follow CRM naming conventions. The mapped CRM entity can be considered as `rdf:type` for every node of that content type. For example in our case study a content type named *Paper* refers to separate objects in a collection and therefore the associated records will have `rdf:type` of *E24 Physical Man-Made Thing*. Naming content types with non-CRM names helps researchers without CRM expertise to become familiar with the application quickly.

Given that the Drupal core RDF model requires that the fields of a content type are implemented as RDF predicates, we propose that direct and inherited CRM properties are mapped to these fields and that other properties are not included. CRM properties which connect entities with types are easily accommodated using the core Taxonomy module or the contributed Web Taxonomy module. Properties that point to literals can be accommodated by Drupal's default field functionality with a range of data types supported. For example, a typical piece of information recorded for a decorated paper is the type of material it is made of. A field called *paper by origin* has the property `crm:P45.consists_of` and can store references (using the Web taxonomy module) to the Language of Bindings Thesaurus.

When CRM properties link two CRM entities, the corresponding association in Drupal should be links between nodes. This can be done by creating a field in one content type and defining it as an *Entity reference* field. This functionality is offered by the Entity reference module. For example, a content type *Paper production* which is of `rdf:type` *E12 Production* has an Entity reference field called *paper* with `crm:P108.has_produced` which links to a content type called *Paper* with `rdf:type` *E24 Physical Man-Made Thing*.

Drupal requires the definition of at least one field for a content type. By default this field is text and is called *title*. In our solution *title* can be considered as an identifier and rules for the consistent production of standardised identifiers for this field can be defined easily (this identifier may be different to the internal database ID for each node).

All definitions of mappings can be done using the UI provided by the RDFx module.

3 Case study

As mentioned before our main objective is to provide a resource for research in the provenance of decorated papers based on a solid semantic foundation. We explain here our main considerations on the kind of data which we are recording.

3.1 Decorated papers model

We consider some critical events in the history of a decorated paper that must be recorded in order for researchers to establish its provenance.

- Paper production: the event of creating the decorated paper. Some decorated papers are produced through elaborate techniques which take a long time and require actions at different periods of time. We consider the whole production of the paper as one event because we think that describing the process step by step will be impossible for most papers.
- Incorporation into an object: the event of adding the paper to another object. This is important because from this point onward the two objects share their histories. This event typically involves attaching decorated papers to a book. A typical misinterpretation in the field is assuming that the date and place of production of the book is the date and place of production of the decorated paper.

Our system includes the possibility of recording the physical description of papers by indicating details such as the materials and techniques, the colours, the production tools, etc. These details are linked to the correct event or object and therefore can offer better granularity for searching and semantically accurate statements.

3.2 CRM mapping

Table 1 shows some of the Drupal content types and fields alongside their corresponding CRM entities and properties for the decorated papers case study as well as the Drupal modules that can be used to implement them. We have two fields which are not included in the table but are common to all content types:

- Title: as mentioned in section 2.4.2 every content type requires a title field which is mapped as an identifier (*P1 is identified by*).
- Notes: a free text field mapped using the CRM property *P3 has note*.

As mentioned in section 2.4.1 the Entity reference module can create links between Drupal nodes, from the node that hosts the Entity reference field (domain) to another node (range), but the inverse link is not established explicitly (the Relation module allows for such functionality). In our mapping, when two entities need to be linked from range to domain we use the concept of the CRM inverse properties, e.g. *P108i was produced by* instead of *P108 produced*.

A complete mapping is included in figure 3. Each table corresponds to a Drupal content type and each row of a table corresponds to a Drupal field. The entity and taxonomy references are indicated with an arrow and the thesauri referenced are enclosed in boxes.

4 Additional modules

Having described the essential modules for our solution we now refer to some additional modules which may improve the user experience when producing and searching records on the Drupal system.

4.1 Data inputting

Automatic Entity labels (https://www.drupal.org/project/auto_entitylabel) It is useful to specify rules for the production of node identifiers which are meant to be stored in Drupal's default title field. We recommend the use of the *Automatic Entity Labels* module for this task. An example of such rule for the *Paper* content type is this:

```
paper - [node:nid]
```

resulting identifiers like:

```
paper - 1
paper - 2
paper - 3
paper - 4
```

where the word *paper* is accompanied by the internal Drupal database ID for the specific node.

Entity connect (https://www.drupal.org/project/entityconnect) The interface offered by the Entity reference module requires that the node to connect to (range) has been created already. When producing records which span multiple content types the range nodes may not have been created for every domain node. This would typically mean that the domain node has to be saved incomplete, a range node has to be created and then the domain node has to be edited for the link to be established. This is a time consuming process which may slow down and frustrate a user. We recommend the use of the *Entity connect* module which allows the creation of a range node from within the domain node and without the domain node having to be saved in an incomplete state.

4.2 Searching

Drupal offers a number of modules which can be used for manual searching of the records including a standard indexing and searching core module. It is beyond

Table 1 Indicative examples of CRM mapping for the decorated papers case study

Drupal entity	Drupal entity type	CRM entity/property	Drupal module
Paper	Content type	E24 Physical Man-Made Thing	Node (core)
paper production	Entity reference field	P108i was produced by	Field (core), Entity reference
paper by decoration	Web taxonomy field	P45 consists of	Field (core), Web taxonomy
paper by origin	Web taxonomy field	P45 consists of	Field (core), Web taxonomy
paper by manufacture	Web taxonomy field	P45 consists of	Field (core), Web taxonomy
located at	Entity reference field	P53 has former or current location	Field (core), Entity reference
decoration features	Entity reference field	P56 bears feature	Field (core), Entity reference
Paper production	Content type	E12 Production	Node (core)
technique	Web taxonomy field	P32 used general technique	Field (core), Web taxonomy
tool	Web taxonomy field	P125 used object of type	Field (core), Web taxonomy
date	Field	P4 has time-span	Field (core)
place	Field	P7 took place at	Field (core)
maker	Field	P14 carried out by	Field (core), Taxonomy (core)
Part addition	Content type	E79 Part Addition	Node (core)
paper	Entity reference field	P111 added	Field (core), Entity reference
object	Entity reference field	P110 augmented	Field (core), Entity reference
date	Field	P4 has time-span	Field (core)
place	Field	P7 took place at	Field (core)
maker	Field	P14 carried out by	Field (core), Taxonomy (core)

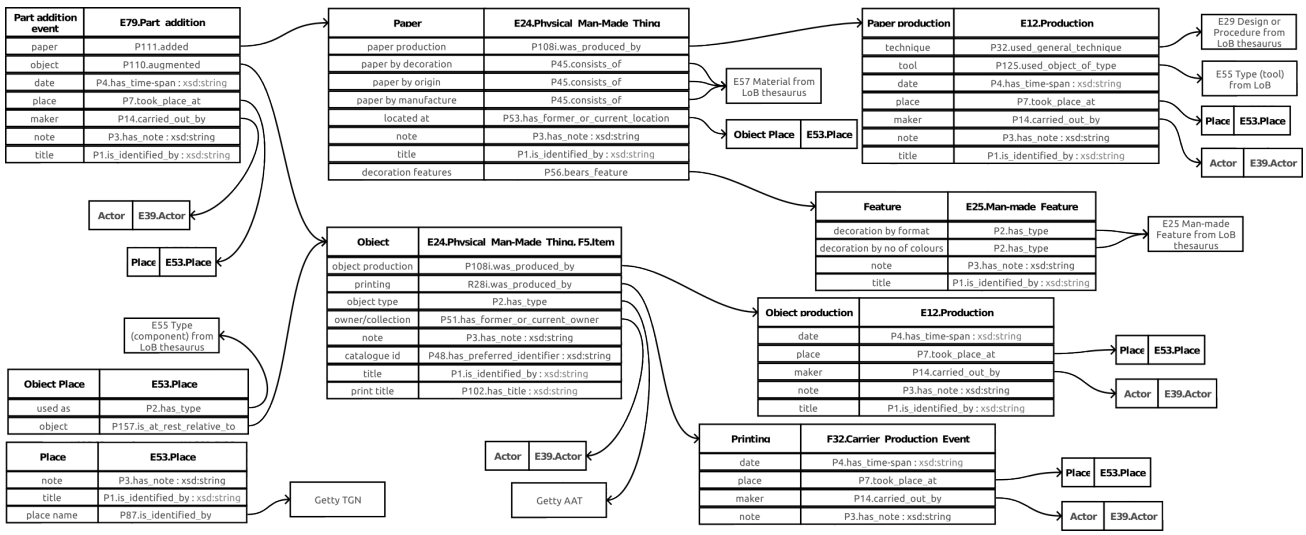

Fig. 3 Content, content types and relationships to vocabularies in Drupal.

the scope of this paper to make recommendations about data retrieval given that requirements will vary according to research objectives and available resources. But we highlight some important modules which we have tested and found useful:

- *Views* (https://www.drupal.org/project/views) is a popular module for building highly customisable queries to retrieve data from the database of a Drupal installation.
- *Apache Solr Search* (https://www.drupal.org/project/apachesolr) integrates with the popular indexing and querying application Apache Solr (https://lucene.apache.org/solr/).
- *Facet API* (https://www.drupal.org/project/facetapi) allows using faceted searching in Drupal. In our setup Facet API can be combined with the Entity reference module to allow faceted searching based on CRM properties.

It is worth making a special reference to the Views module: it is widely supported by other modules and in our case both the Entity reference and Taxonomy links can be queried using Views. An example of a Views query can be seen on the decorated papers website front page (http://www.ligatus.org.uk/decoratedpapers/). Fields from different content types have been combined in one search page by using their corresponding Entity reference links. For example, *decorated paper type* is a field in content type *Paper*, *colour* is a field in content type *Paper feature* and *technique* is a field in content type *Paper production*, yet all of them can be combined to act as criteria for filtering in one query.

4.3 Drupal as SPARQL endpoint

Several discussions about the development of a module for offering Drupal RDF data through a SPARQL endpoint have taken place. These discussions have not led to a defacto module with that functionality. Following the project *SPARQL* (https://www.drupal.org/project/sparql), the current recommendation for SPARQL endpoints in Drupal is to use the *RDF indexer* module (https://www.drupal.org/project/rdf_indexer) combined with the *ARC2 store* module (https://www.drupal.org/project/arc2_store). These are useful for testing but not recommended for a production server. The newest version of Drupal (8.x) includes the required functionality for our proposed setup but again a clear path for a sustainable module to offer RDF mappings and a SPARQL endpoint is not set.

5 Discussion and conclusions

We demonstrated a method for recording and publishing CRM data by researchers with little technical knowledge. Our solution uses Drupal - a popular and user friendly CMS and a small number of popular and well-supported contributed modules ensuring long-term sustainability. Our solution can be implemented cheaply using standard hosting packages. Modern CMSs such as Drupal only require a basic understanding of familiar controls similar to any web application. We have offered guidelines which should allow a researcher to set up a project website for creating and publishing CRM data in a matter of hours. The resulting system takes advantage of Drupal's capacity to render human-friendly webpages and at the same time provide underlying data in standard RDF format thus requiring the maintenance of only one resource as opposed to maintaining multiple formats of data. We believe that our method will enable more researchers to share their CRM data online.

We have confirmed that our method works with a case study for the description of decorated papers. Next, we describe areas where we believe further work is required.

Content spanning multiple content types In our case study data about an entry (e.g. a specific decorated paper) is stored in nodes from different content types. As explained in section 4.1 building links across nodes requires that the range node has been created already. We recommended the Entity connect module to work around this problem. In our case we discovered that if data inputting starts from the content type *Part addition event* (see figure 3) then all other nodes can be created in a single process through the Entity connect links without having to save intermediate and incomplete records. I.e. Part addition → Paper → Paper production → Paper features → Object → Object production etc. However, in other cases it may not be possible to identify a single path and therefore incomplete records will need to be saved. We intend to investigate this further and confirm when a single path is possible with the use of inverse properties and will report on our findings in the future.

SPARQL querying RDF is a popular way of publishing CRM data. Drupal supports the publication of data as RDFa as part of the rendered HTML pages. We have investigated a number of possibilities for making this data available as a SPARQL endpoint but our conclusion is that none of them are mature enough to be used in a production server. We will continue to monitor the developments in this area. It is possible to export Drupal data in a range of formats including various RDF serialisations and then import this data to an external triple store which features tools for SPARQL querying. However, this would require a much higher level of technical expertise and would compromise one of our objectives which is the publication of CRM data without any technical expertise. We are confident that implementing our solution on Drupal does not prevent data migration to other systems given the wide range of exporting modules available. For the time being, indexing of RDFa data can be done by external tools. We have done some preliminary testing of our RDFa data with validation tools such as the Google Structured Data Testing Tool (https://developers.google.com/structured-data/testing-tool/) and the Structured Data Linter (http://linter.structured-data.org/) which have successfully identified the CRM properties and entities. We will continue to investigate options for indexing RDFa data.

Field validation based on Enlargen CRM Unlike the solution offered by WissKI, we specify the fields of our content types manually. This process is prone to errors and may lead to incorrect mapping of fields to the CRM. The specification of fields through validation with Enlargen CRM would be useful functionality to offer to a researcher without much experience of the CRM. Such functionality has been demonstrated for the schema.org ontology by the *RDFUI* module https://www.drupal.org/project/rdfui which is currently only available for Drupal 8.x. Generalising this functionality to support any ontology should be relatively easy and a relevant 'issue' has already been submitted (https://www.drupal.org/node/2386779). We expect that with the gradual migration from Drupal 7.x to Drupal 8.x we will be able to take advantage of automatic validation of fields using the Enlargen CRM.

Online documentation We are planning to offer online documentation for researchers who wish to use Drupal to publish their data. This will be in the form of a point-and-click how-to explaining the functionality of the different modules and will be published within http://www.ligatus.org.uk/. In preparation for migration to Drupal 8.x we will also provide a configuration for our proposed setup since the latest version of Drupal supports exporting and importing such configuration out of the box.

We believe that these resources will allow more researchers to publish their CRM data easily and we will continue to improve our setup as the development of the related modules continues.

Acknowledgements We would like to thank Maria Theodoridou and George Bruseker from FORTH for their comments on Drupal and the CRM mapping. We would also like to thank members of the Arbeitskreis Buntpapier: Henk Porck (Koninklijke Bibliotheek), Susanne Krause (Hamburger Buntpapier - Buntpapierverlag), Matthias Hageböck (Herzogin Anna Amalie Bibliothek), Julia Rinck (Deutsches Buch- und Schriftmuseum der Deutschen Nationalbibliothek) for their advice with the decorated papers terminology. Special thanks to Nicholas Pickwoad for his continuous support of this project.

References

1. Corlosquet S, Delbru R, Clark T, Polleres A, Decker S (2009) Produce and Consume Linked Data with Drupal! In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Bernstein A, Karger DR, Heath T, Feigenbaum L, Maynard D, Motta E, Thirunarayan K (eds) The Semantic Web - ISWC 2009, Springer Berlin Heidelberg, Berlin, Heidelberg, vol 5823, pp 763–778, URL http://data.semanticweb.org/conference/iswc/2009/paper/inuse/101/html
2. Crofts N, Doerr M (1999) Electronic Esperanto: The Role of the Object Oriented CIDOC Reference Model. In: Cultural Heritage Informatics: Selected papers from ICHIM99: the International Cultural Heritage Informatics Meeting, Archives & Museum Informatics, Washington, DC, USA, URL http://www.archimuse.com/publishing/ichim99/doerr.pdf
3. Cyganiak R, Wood D, Lanthaler M, Klyne G, Carroll JJ, McBride B (2014) RDF 1.1 Concepts and Abstract Syntax. URL http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/
4. Doerr M (2003) The CIDOC Conceptual Reference Module: An Ontological Approach to Semantic Interoperability of Metadata. AI Magazine 24(3):75, DOI 10.1609/aimag.v24i3.1720, URL http://www.aaai.org/ojs/index.php/aimagazine/article/view/1720
5. Doerr M, Theodoridou M, Aspöck E, Masur A (2015) Mapping archaeological databases to CIDOC-CRM. Sienna
6. Guillem A, Bruseker G, Zarnic R (2015) Building an Argumentation Platform for 3d Reconstruction using CIDOC-CRM and Drupal. In: Proceedings of the 2015 Digital Heritage International Congress, Grenada, Spain
7. Herman I, Adida B, Sporny M, Birbeck M (2013) RDFa 1.1 Primer - Second Edition. URL http://www.w3.org/TR/xhtml-rdfa-primer/
8. Jammes A (2010) Papiers dominotés: trait d'union entre l'imagerie populaire et les papiers peints (France, 1750-1820). Aux Editions des cendres, [Paris]
9. Kopylov CF (2012) Papiers dorés d'Allemagne au siècle des Lumières: suivis de quelques autres papiers décorés (Bilderbogen, Kattunpapiere & Herrnhutpapiere) : 1680-1830. Éditions des Cendres, [Paris]
10. Kopylov M (2012) Papiers dominotés italiens: un univers de couleurs, de fantaisie et d'invention : 1750-1850. Éditions des Cendres, [Paris]
11. Kopylov M, Jammes A (2012) Papiers dominotés français ou L'art de revêtir d'éphémères couvertures colorées livres et brochures entre 1750 et 1820. Éd. des Cendres, Paris
12. Marks PJM (2015) An anthology of decorated papers. A sourcebook for designers. Thames & Hudson Ltd., Farnborough
13. Miles A, Bechhofer S (2009) SKOS Simple Knowledge Organization System Reference. URL http://www.w3.org/TR/2009/REC-skos-reference-20090818/
14. Myers D, Dalgity A, Avramides I, Wuthrich D (2012) Arches: An Open Source GIS for the Inventory and Management of Immovable Cultural Heritage. In: Ioannides M, Fritsch D, Leissner J, Davies R, Remondino F, Caffo R (eds) Progress in Cultural Heritage Preservation, no. 7616 in Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp 817–824, URL http://link.springer.com/chapter/10.1007/978-3-642-34234-9_87, dOI: 10.1007/978-3-642-34234-9_87
15. Oldman D, Doerr M, Jong G, Norton B, Wikman T (2014) Realizing Lessons of the Last 20 Years: A Manifesto for Data Provisioning and Aggregation Services for the Digital Humanities (A Position Paper) System. D-Lib Magazine 20(7/8),

DOI 10.1045/july2014-oldman, URL http://www.dlib.org/dlib/july14/oldman/07oldman.html
16. Porck H, Schrijver I, Gerard J (2009) Buntpapier - ein Bestimmungsbuch = Decorated paper - a guide book = Sierpapier - een gids. Buntpapierverl., Hamburg
17. Ribaud V (2010) End-user storytelling with a CIDOC CRM - based semantic wiki. pp 1 –22, URL http://hal.univ-brest.fr/hal-00630580/document
18. Scholz M, Goerz G (2012) WissKI: A virtual research environment for cultural heritage. Front Artif Intell Appl Frontiers in Artificial Intelligence and Applications 242:1017–1018