Ontology Modelling and Standardized Information Exchange with Content Delivery Applications in Technical Communication

Daniel Nägele[1*], *Patricia* Vobl[1**]

[1]Karlsruhe University of Applied Sciences, Faculty of Information Management and Media, 76133 Karlsruhe, Germany

Abstract. Ontologies are a technology recently used in technical communication (TC) to model information into a multidimensional net. They expand the modelling by taxonomy of metadata in TC. Any kind of relation between multiple classes and instances can be established. These ontologies can appear in the form of semantic correlation rules (SCR), which represent the connection between the metadata of the objects. SCR are used in connection with component content management systems (CCMS), semantic modelling systems (SMS) and content delivery portals (CDP) to deliver the appropriate amount of content in a more precise manner to the end user. In general, Ontology tools, CCMS and CDP are not based on the same ecosystem and therefore, they do not always work together effortlessly. A solution to this problem are exchange formats like the intelligent information Request and Delivery Standard (iiRDS), which enable a standardized information exchange between supported systems. Another solution would be compound information systems (CIS) like ONTOLIS, which combine a CCMS, CDP and SMS all in one. This paper aims to investigate the effect of SCR in the CDP of a CIS like ONTOLIS and to evaluate the use of exchange formats like iiRDS.

1 Introduction

The complexity of products and their variants has increased significantly in the last decades due to digitalization and globalization. In the digital age, accessing information is easier than ever, which can lead to an information overflow and harm the user experience. From the user's perspective, it can be difficult to find information in the right amount and context as quickly as possible. Usually, an entire document would provide too much content, but a single topic is probably not enough to fulfil the user's needs [1]. Therefore, requirements on documentations are higher, especially due to the new possibilities introduced by digitalization.

Additionally, many different types of information systems are used nowadays in TC to meet the increased requirements on documentation. The information systems can derive from different vendors, which can hinder the compatibility between them and, therefore, impede the data exchange. Consequently, different systems may not work together flawlessly, which decreases the quality and efficiency of documentation.

We attempt to improve the compatibility of variant systems using two different approaches: implementing exchange formats or employing a CIS. Furthermore, we explore SCR as a method to deliver the amount of content needed in the context necessary to the user. In our case the technological context and background was the field of Smart Technologies with a focus on Smart Homes.

2 Ontologies

An ontology defines all relevant objects and their possible interactions with each other within a certain field. They consist of object classes, their instances and properties and their relations between each other. The amount of all ontology objects varies depending on the product complexity and use cases. These ontologies build the base in product modelling and serve as structured central knowledge bases [2]. The result is usually displayed as a multidimensional semantic network.

Ontologies gained popularity recently in TC because they can help to improve model-based content creation processes or effectivity of search and delivery processes [3].

3 Semantic correlation rules (SCR)

SCR can be viewed as an (ultra-)lightweight ontology. Consequently, SCR have less semantic expressiveness than ontologies. Therefore, SCR can only replace ontologies, if the underlying use cases, CDP functions and content and product structure are not too complex. SCR are rulesets that consist of inrules connected to several outrules. These inrules and outrules describe correlations between information objects based on their metadata in a standardized and formalized way. For further specific information on SCR see reference [4].

* Daniel Nägele: nada1015@hs-karlsruhe.de
** Patricia Vobl: vopa1017@hs-karlsruhe.de

4 Standardized information exchange using the example of iiRDS

In the following, we will describe and explain the information exchange between information systems using the intelligent information Request and Delivery Standard (iiRDS).

4.1 Introduction to iiRDS

The intelligent information Request and Delivery Standard was developed by tekom [5], which is the largest European association for TC. The development started in 2016 and version 1.0 was released in 2018. The current version is 1.1, which we used in this project.

iiRDS aims to transform content from XML-, HTML- or PDF-formats into iiRDS packages, similar to compressing several files into a Zip archive file format. The iiRDS package cannot only contain content, but also metadata and ontology relations. It provides a standardized set of metadata, which can also be expanded to integrate existing customized metadata.

This standardized set of metadata can be found in the iiRDS specification [6]. The specification also contains all information necessary to work with iiRDS, including examples. The standardized set of metadata is modelled after the PI-Class model developed by Prof. Dr. Ziegler [7]. Therefore, it is best suited for content in the field of mechanical engineering. Additionally, there are two supplementary extensions: the Machinery Domain and the Software Domain. They extend the core with additional metadata for machinery or software documentation. In this project, we focused on the iiRDS core without domain extensions, because they did not fit our field and use cases.

The reason for the standardized metadata set is to ensure, that every compatible information system can process the metadata accordingly. It also ensures, that technical writers understand the meaning behind it and utilize the metadata correctly. The metadata consists of classes and instances. Classes can also be subclasses of other classes. Classes are categories of metadata and have corresponding instances, which are arranged hierarchically below the classes. The class ‘Information Unit’ has, for example, the instances ‘Document’, ‘Topic’ and ‘Fragment’.

Ideally, information from CCMS can be used in CDP and SMS with the help of iiRDS. Moreover, the goal is to maximize the efficiency of information exchange between the already mentioned systems.

4.2 Procedure of implementing iiRDS

The goal of this approach is to get content and its metadata from a CCMS into a CDP using an exchange format like iiRDS. For the CCMS, we used Smart Media Creator by EC-Systems [8] and for our CDP we used i-Views by Empolis Intelligent Views [9].

The first step of implementing iiRDS occurs in the CCMS. There, all relevant content is stored with its associated metadata. In an optimal scenario, the metadata structure in the CMS should already match the standardized structure of the iiRDS metadata set. In this case, a simple iiRDS export would be sufficient to produce an adequate package. This package can be imported and used in the CDP afterwards.

Unfortunately, our testing showed, that many of the processes involved in the information exchange with iiRDS do not work flawlessly.

4.2.1 Export from the CCMS

Currently, Smart Media Creator has a direct iiRDS export in development. At the time of this project, this feature was not available. Instead, an Extensible Markup Language (XML) export was used, and every topic was exported individually as an XML file.

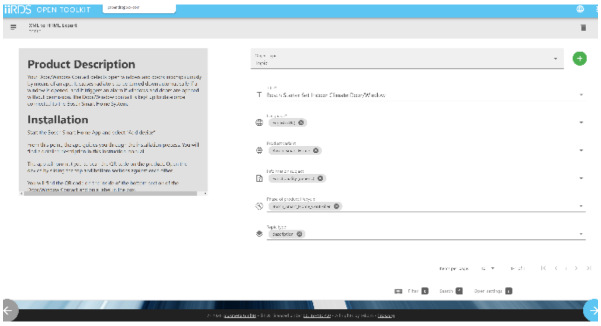
The generated XML files then needed to be converted into Hypertext Markup Language (HTML) files. At this point in the process, Extensible Stylesheet Language Transformation (XSLT) was applied. In a batch file the XML file, the stylesheet and the processor Saxon were linked together, resulting in the generation of an HTML file upon execution. The stylesheet transformed every XML element into its equivalent HTML element. In the end, all topics from the CCMS were represented by an HTML file each.

4.2.2 Integrating metadata and generating iiRDS packages

Our generated HTML files contained all of our content, but still lacked metadata. The CCMS metadata was produced without iiRDS metadata in mind and was only based on our observed products and created use cases. Before metadata can be assigned to produce an iiRDS package, the CCMS metadata must be matched to the standardized set of metadata which the iiRDS specification provides.

For this purpose, an Excel table was created and filled with the CCMS metadata in a hierarchical structure. As our project’s use cases and metadata are not from the field of mechanical engineering, the matching was not always straightforward. For example, we could match our ‘I-Class’ CCMS metadata class directly with the iiRDS class ‘TopicType’. For our class ‘physical_parameters’, there was no logical equivalent from iiRDS, so we matched it with the unused ‘Supply’ class.

After matching every class of CCMS metadata to an iiRDS class, the content is ready to be packaged. For this task, tekom offers a web-based software called iiRDS Open Toolkit [10]. We focused on using the iiRDS Open Toolkit because it is – at the current state of the iiRDS implementation – an easy way to produce iiRDS packages. The Open Toolkit enables users to upload their content in a variety of file formats like HTML or docx. Afterwards, iiRDS metadata classes can be assigned and their instances can be chosen from drop-down lists. Conveniently, proprietary instances can be added to these classes. Proprietary classes, however, cannot be added. Not all classes from the iiRDS core are included in the Open Toolkit, which decreases its flexibility. Due to the missing proprietary classes, we were not able to reproduce our hierarchical structure when assigning metadata.



Start the Bosch Smart Home App and select "Add device"

Fig. 1. Assigning metadata in the iiRDS Open Toolkit.

In the next step the Open Toolkit processes the data and generates an iiRDS package containing the uploaded content and assigned metadata. The finished package can then be downloaded.

4.2.3 Editing iiRDS packages

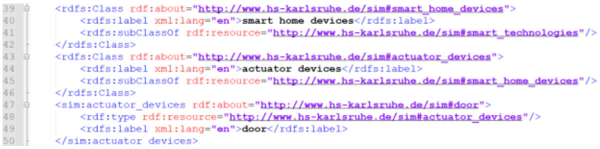
The iiRDS package is not finalized and can still be edited. It can be opened with any file archiver, like 7-Zip, to edit the files inside of it. Every iiRDS package contains a content folder and a metadata folder. The content folder named 'CONTENT' contains the content file, in our case the HTML topic. The metadata folder named 'META-INF' contains a metadata.rdf file in which all metadata is stored.

This metadata.rdf file can be edited to add custom code and, therefore, more metadata or ontology data. Unfortunately, ontology data cannot be added through the Open Toolkit and must be added afterwards. Exchanging ontology data with iiRDS was not in the scope of this project.

The metadata.rdf file can be edited in a text editor to add proprietary metadata. With the help of the correct syntax, proprietary classes and instances can be added. Instructions for this process can be found in the iiRDS specification [4]. Due to the inability of the iiRDS Open Toolkit to add proprietary classes, we added them manually.

To create and identify our own metadata, we first defined a namespace in the metadata.rdf file. We did this with an xmlns attribute within the <rdf:RDF> tag. We chose the prefix 'sim' and named the namespace 'http://www.hs-karlsruhe.de/sim#'. After adding the namespace, the first proprietary class can be added using the <rdfs:class> tag.

Within this tag, the defined namespace is used to identify the new proprietary class. <rdfs:class> also contains two subelements: <rdfs:label> for the display name of this class and <rdfs:subClassOf> to indicate that this class is a subclass of another class.



```
39      <rdfs:Class rdf:about="http://www.hs-karlsruhe.de/sim#smart_home_devices">
40          <rdfs:label xml:lang="en">smart home devices</rdfs:label>
41          <rdfs:subClassOf rdf:resource="http://www.hs-karlsruhe.de/sim#smart_technologies"/>
42      </rdfs:Class>
43      <rdfs:Class rdf:about="http://www.hs-karlsruhe.de/sim#actuator_devices">
44          <rdfs:label xml:lang="en">actuator devices</rdfs:label>
45          <rdfs:subClassOf rdf:resource="http://www.hs-karlsruhe.de/sim#smart_home_devices"/>
46      </rdfs:Class>
47      <sim:actuator_devices rdf:about="http://www.hs-karlsruhe.de/sim#door">
48          <rdf:type rdf:resource="http://www.hs-karlsruhe.de/sim#actuator_devices"/>
49          <rdfs:label xml:lang="en">door</rdfs:label>
50      </sim:actuator devices>
```

Fig. 2. Editing metadata in the metadata.rdf file.

In this example, we added the proprietary classes 'smart home devices' and 'actuator devices' and the proprietary instance 'door'. Using the <rdfs:subClassOf> tag, it was possible to replicate our hierarchical metadata structure. In line 47 in Fig. 3, the proprietary instance 'door' is added as an instance of the class 'actuator devices'.

Due to the limited timeframe of this project, the metadata of only one iiRDS package was edited completely following this procedure. For the remaining packages, metadata was added exclusively through the Open Toolkit.

4.2.4 Importing iiRDS packages

After generating and potentially editing the iiRDS packages, they can be imported into a CDP, in our case i-Views by Empolis Intelligent Views. Since one of the goals of iiRDS is to enable an easy information exchange, we focused on testing the import of iiRDS packages from the Open Toolkit without editing.

The import itself was done by another group in this project. For more information on the testing procedure see reference [11]. As a result, the information exchange itself worked out, but the metadata could not be imported properly. The metadata had to be linked to the corresponding content afterwards in the CDP.

4.3 Result

In theory, iiRDS provides a simple solution to the problem of information exchange between different information systems. In practice, we encountered a few problems.

Firstly, iiRDS is a relatively new standard, therefore, it is not supported natively by all information systems yet. Even if information systems support iiRDS, levels of compatibility can vary. This was the case in the CMS and the CDP. Thankfully, these problems could be solved using workarounds. Nevertheless, the workflow with iiRDS and the information systems was not as smooth as we hoped.

Secondly, there was more coding required than expected. We anticipated that the implementation workflow and information exchange could be done without coding. Due to the limitation of the Open Toolkit, our metadata structure and impaired compatibility of information systems, coding was a necessity for our testing.

Thirdly, the iiRDS Open Toolkit has some limitations which made it more tedious to use. Not all iiRDS metadata in the specification is existent in the Open Toolkit. This exacerbates the matching of existing metadata to the iiRDS metadata. Less metadata makes the Open Toolkit less flexible in general.

5 Compound information system

In the following, we will describe and explain the functionality of a CIS and the implementation of SCR into it, using the example of the software ONTOLIS. A CIS is a system, that includes several types of information systems in one. Primarily, three types of information systems are part of CIS: CCMS, CDP, and SMS.

5.1 Introduction to ONTOLIS

ONTOLIS, by the company ONTOLIS GmbH, is a proprietary web-based compound software. The main scope of this software is to create ontologies according to the customers' request and, therefore, to serve as a SMS. In addition to ontology modelling, the software can be used as a CCMS, CDP and as a delivery platform for metadata. It can also serve as a terminology and localization manager, which we did not include, because it was not part of the project. Even though ONTOLIS is a compound software and, therefore, can be utilized as a CCMS, SMS and CDP all in one, it can also be used as only one of the already mentioned information systems depending on the project’s scope. The functionality behind the software is based on three different pillars: the semantic network, which is visualized as tree structures in this specific software, the models and QIRA (Query Integrate Reasoning Automate). Within the semantic tree structures, the object classes and relations between these classes are determined. Inside of the models, the instances of the object classes are defined. QIRA is the proprietary graphic programming language of ONTOLIS and determines how the models can be used and displayed in the CDP [2].

5.2 Procedure of implementing the SCR in ONTOLIS

In the following, the pre-conditions and steps for the SCR implementation will be described in detail. These steps do not need to be executed in a chronological order but can overlap and influence each other.

The goal of this procedure is to verify if the SCR implementation only is sufficient to make the CDP work correctly or if the entire ontology with its multiple relations is needed. Theoretically, an ontology in ONTOLIS can display much more than only SCR. The software is optimal for keeping track of thousands of products and many complex relations in between them with its typical semantic tree structures. However, in our specific case, it was not necessary to use all the functions and the complexity of which the software is capable of.

5.2.1 Implementing object classes

As the first step, the object classes, which were defined by our project group at the beginning, were implemented into the software. This serves as a pre-condition for every other step. The object classes and their subclasses were mapped in the form of tree structures in the so-called “Smart Technologies Schema”, on which the models will be assembled on. The metadata, content, SCR and the QIRA depend on this schema.

In the screenshot below, a typical object class with its relations, properties and operations is shown. As already mentioned, the tree structure, which can be observed below, is the typical visualization in ONTOLIS.

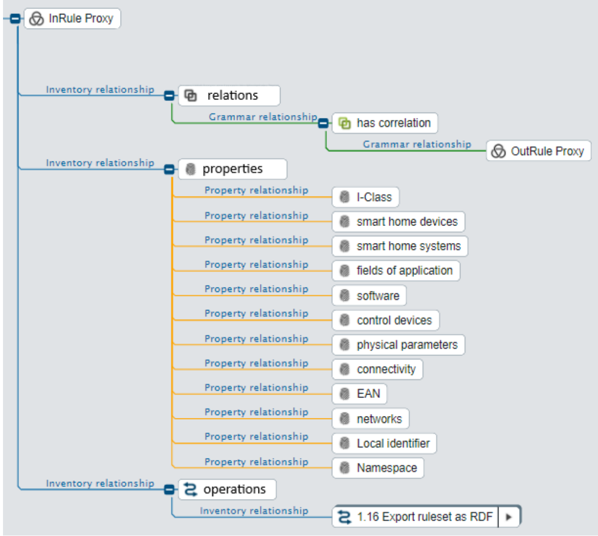


Fig. 3. Object class in ONTOLIS

Every object class and subclass inside of the schema was connected specifically to their relations and properties. In ONTOLIS, all relations and properties are defined centrally at a specific space inside of the semantic tree structure and they are then assigned to the specific object classes through drag and drop. The properties are important to assign metadata to the object classes in the models. The relations define, which object class is connected to one another. Ultimately, operations were connected to the object classes. The operations are not necessary for the functioning of SCR. They enable additional QIRA processes like RDF export.

5.2.2 Implementing metadata

In general, metadata enables tagging content specifically and, therefore, serves as the pre-condition for displaying related content in the CDP and SCR modelling. Consequently, the metadata can be considered as the interface between content, the CDP, and the SCR.

For implementing metadata into the software, models based upon the schema with its object classes must be created. It is pre-defined in the schema, which relations the metadata can have in the models.

In the following screenshot, an example of a metadata model is shown. The object class “smart home devices” has different hierarchical instances, which are displayed below.

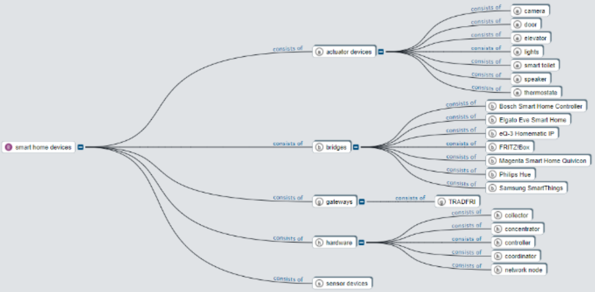


Fig. 4. Metadata model in ONTOLIS

This model only serves as an example. In fact, metadata models for all object classes are available.

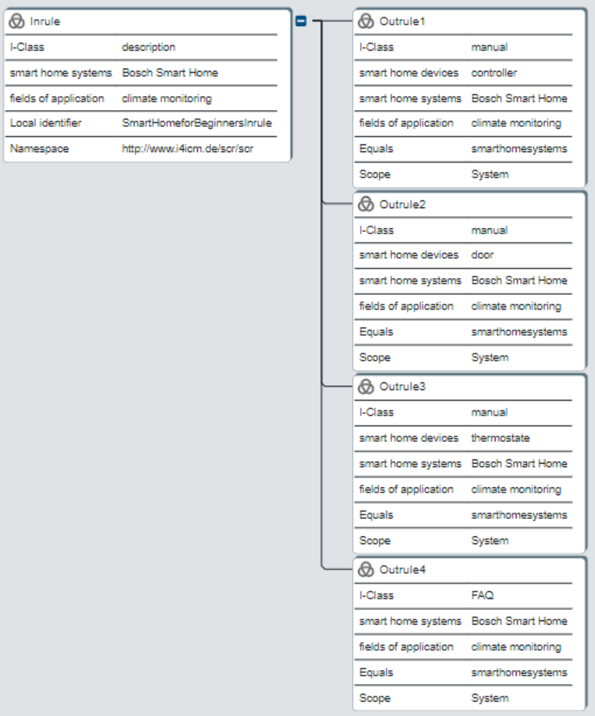
5.2.3 Implementing content

The content, consisting of different PDFs regarding the field smart technologies, was implemented afterwards. Every PDF was inserted in a separate model and was tagged with appropriate metadata. Inserting XML data or text nodes would be possible too. Although, it must be defined in QIRA beforehand which formats can be displayed afterwards in the CDP.

5.2.4 Implementing SCR

Four different use cases, which are valid for our entire project group, were defined as the base for the SCR. Every use case is built out of one inrule and various outrules. Together, they are entitled as a ruleset. Afterwards, the inrules and outrules were tagged with metadata. The metadata represent the validities for every rule. For example, one ruleset could be tagged as only valid for Bosch Smart Home. For further information on SCR and inrules and outrules see reference [4].

In the screenshot below, an example of a proper ruleset is shown. It consists of one inrule und four different outrules. The equals can be implemented in outrules to reference the same instance in the inrule. The ruleset below represents use case 1 "Smart Home for Beginners": A mother, who has five children and a full-time job has no time for studying the details of the whole manual. She bought a BOSCH smart home system and aims to implement the climate monitoring.



Inrule	
I-Class	description
smart home systems	Bosch Smart Home
fields of application	climate monitoring
Local identifier	SmartHomeforBeginnersInrule
Namespace	http://www.i4icm.de/scr/scr

Outrule1	
I-Class	manual
smart home devices	controller
smart home systems	Bosch Smart Home
fields of application	climate monitoring
Equals	smarthomesystems
Scope	System

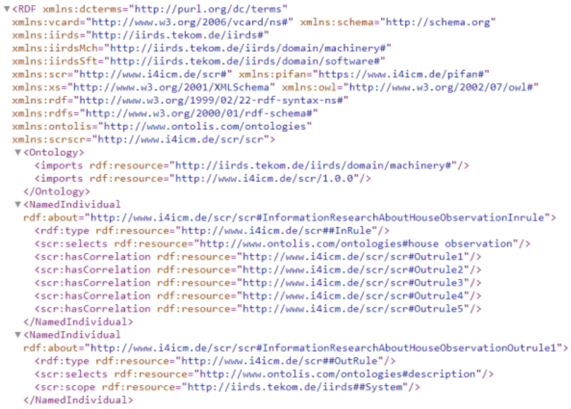
Outrule2	
I-Class	manual
smart home devices	door
smart home systems	Bosch Smart Home
fields of application	climate monitoring
Equals	smarthomesystems
Scope	System

Outrule3	
I-Class	manual
smart home devices	thermostate
smart home systems	Bosch Smart Home
fields of application	climate monitoring
Equals	smarthomesystems
Scope	System

Outrule4	
I-Class	FAQ
smart home systems	Bosch Smart Home
fields of application	climate monitoring
Equals	smarthomesystems
Scope	System

Fig. 5. SCR in ONTOLIS

There is also the possibility to combine a CIS such as Ontolis with standardized exchange formats like resource description frameworks (RDF) through a QIRA procedure. This enables the data exchange between a CIS and another software system. For starting the export process, the button which appears when selecting the inrule model must be clicked. When clicked, the system generates an RDF file. Theoretically, an RDF import would also be possible, but must be defined in QIRA first. An example for an RDF file is shown in the screenshot below.



```
▼<RDF xmlns:dcterms="http://purl.org/dc/terms"
  xmlns:vcard="http://www.w3.org/2006/vcard/ns#" xmlns:schema="http://schema.org"
  xmlns:iirds="http://iirds.tekom.de/iirds#"
  xmlns:iirdsMch="http://iirds.tekom.de/iirds/domain/machinery#"
  xmlns:iirdsSft="http://iirds.tekom.de/iirds/domain/software#"
  xmlns:scr="http://www.i4icm.de/scr#" xmlns:pifan="https://www.i4icm.de/pifan#"
  xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:ontolis="http://www.ontolis.com/ontologies"
  xmlns:scrscr="http://www.i4icm.de/scr/scr">
  ▼<Ontology>
     <imports rdf:resource="http://iirds.tekom.de/iirds/domain/machinery#"/>
     <imports rdf:resource="http://www.i4icm.de/scr/1.0.0"/>
   </Ontology>
  ▼<NamedIndividual
   rdf:about="http://www.i4icm.de/scr/scr#InformationResearchAboutHouseObservationInrule">
     <rdf:type rdf:resource="http://www.i4icm.de/scr##InRule"/>
     <scr:selects rdf:resource="http://www.ontolis.com/ontologies#house observation"/>
     <scr:hasCorrelation rdf:resource="http://www.i4icm.de/scr/scr#Outrule1"/>
     <scr:hasCorrelation rdf:resource="http://www.i4icm.de/scr/scr#Outrule2"/>
     <scr:hasCorrelation rdf:resource="http://www.i4icm.de/scr/scr#Outrule3"/>
     <scr:hasCorrelation rdf:resource="http://www.i4icm.de/scr/scr#Outrule4"/>
     <scr:hasCorrelation rdf:resource="http://www.i4icm.de/scr/scr#Outrule5"/>
   </NamedIndividual>
  ▼<NamedIndividual
   rdf:about="http://www.i4icm.de/scr/scr#InformationResearchAboutHouseObservationOutrule1">
     <rdf:type rdf:resource="http://www.i4icm.de/scr##OutRule"/>
     <scr:selects rdf:resource="http://www.ontolis.com/ontologies#description"/>
     <scr:scope rdf:resource="http://iirds.tekom.de/iirds##System"/>
   </NamedIndividual>
```

Fig. 6. RDF file

Usually, lightweight, and less complex ontology representations are using the syntax of RDF and RDF-schemas [3].

5.2.5 Implementing QIRA procedures

As the last step, the QIRA procedures were inserted into the Smart Technologies Schema. The metadata was adapted inside of these procedures. QIRA enables the content folders to be displayed in the CDP and connects it to the SCR. These procedures run in the background of this proprietary software. Therefore, there will be no further technical information about this topic, because it would go beyond the scope of this paper.

5.3 Result of the SCR in the CDP

After implementing the object classes, the metadata, the content, the SCR and the QIRA procedures, the effect of realizing all these steps, was tested in the CDP.

In the screenshot below, the specific CDP of ONTOLIS is shown.

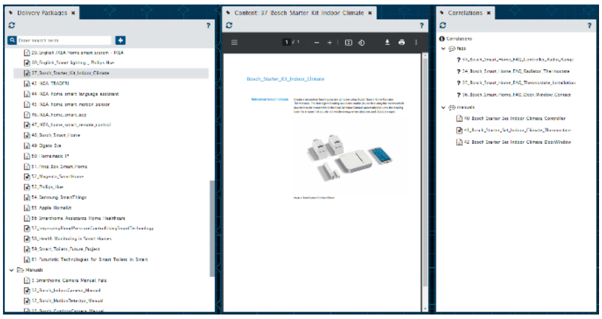

Delivery Packages
Content: 37 Bosch Starter Kit Indoor Climate
Correlations
29_English_IKEA Home smart system - IKEA
37_Bosch_Starter_Kit_Indoor_Climate
43 IKEA TRADFRI
44 IKEA home smart language assistant
45 IKEA home smart motion sensor
46_IKEA_home_smart_app
47_IKEA_home_smart_remote_control
48_Bosch_Smart_Home
49 Digano Dve
50 Homematic IP
51_Fritz.Box.Smart.Home
52_Magenta_SmartHome
53_Philips_Hue
54 Samsung SmartThings
55 Apple HomeKit
56 Smarthome Assistants Home Healthcare
57_ImprovingBloodPressureControlUsingSmartTechnology
58_Health Monitoring in Smart Homes
59_Smart_Toilets_Future_Project
61 Futuristic Technologies for Smart Toilets in Smart
Manuals
3 Smarthome Camera Manual Yale
Bosch_Starter_Kit_Indoor_Climate
faqs
34_Bosch_Smart_Home_FAQ_Radiator_Thermostate
35_Bosch_Smart_Home_FAQ_Thermostate_Installation
36_Bosch_Smart_Home_FAQ_Door_Window_Contact
manuals
40 Bosch Starter Set Indoor Climate Controller
41_Bosch_Starter_Set_Indoor_Climate_Thermostate
42 Bosch Starter Set Indoor Climate DoorWindow

Fig. 7. CDP of ONTOLIS

As displayed above, the CDP consists of three different functions, which are called 'viewlets'. Every viewlet has its own QIRA procedure, which begins to run when the CDP is opened.

The example in the screenshot shows the effect of use case 1 in the CDP: If topic 37 is selected in the delivery packages viewlet on the left, the document is displayed in the content viewer in the center. In addition, the related content of this specific topic is shown in the correlations viewlet on the right, which consists of topic 33, 34, 35, 36, 40, 41, and 42. This list is generated dynamically based on the correlating inrules and outrules of the selected topic. The testing of the remaining use cases can be found in reference [12].

The SCR become relevant when looking at the correlations viewlet. Through the SCR, the content units are correlated with one another, which provides additional context in the correlations viewlet instead of only displaying a single topic in the content viewer. Simultaneously, unnecessary context is being avoided by not delivering an entire manual to the user. Consequently, SCR implementation is sufficient for displaying related content correctly. There is no need for implementing an entire ontology. SCR can be viewed as an (ultra-)lightweight ontology. Therefore, the modelling is quite simple in comparison to ontology modelling, which is more sophisticated since expertise and major product knowledge is needed. For modelling SCR, all that is needed is information about the metadata and use cases and basic knowledge in the field of semantic modelling. Even though SCR modelling has this advantage of being simple and easy to implement, there is also a great disadvantage in comparison to ontologies: SCR have less semantic expressiveness than ontologies. In our case, SCR were sufficient for displaying related content in the CDP since the use cases and metadata were not complex. But if there are more precise and more complex relations, validities and use cases and a higher need for automatization, there is a need for implementing an ontology to make the CDP work.

6 Conclusion

To conclude, the two solutions for the incompatibility of different information systems, which were mentioned in the beginning, will be compared.

There are less difficulties with a CIS like ONTOLIS, which combines CCMS, CDP and SMS, because no external information exchange is needed. On one hand, there is more adaptability in individualizing the data in a CIS since there is no need to follow a standard. On the other hand, the flexibility with a standardized information exchange is higher because there is no restraint to one vendor. Due to that, different strengths from different systems can be utilized.

Theoretically, no programming skills are needed to use standardized information exchange like iiRDS. CIS are software tools, therefore, they need to be learned first for an effective use. In comparison, a standardized exchange format does not have to be learned before implementing it. However, iiRDS is only very intuitive in theory. In practice, coding and workarounds for the import and export into certain systems can be required.

Consequently, if there is no system already in use, implementing a CIS might be an appropriate choice since it combines all relevant functions for information management. If there is already a CCMS in use that supports iiRDS, a CDP that also supports it might be recommendable as this will allow an easy exchange of information between them. SCR can be implemented in both scenarios to deliver the appropriate amount of content in a more precise manner to the end user.

Special thanks to the Faculty of Information Management and Media for the financial support and Nicole Fabricius of ONTOLIS GmbH for the continuous technical support and system access.

References:

1. Ziegler, W. (2019). "Delivery zwischen Kontext und Content". technische kommunikation, (2019) H. 6, S. 58-61
2. Ontolis, Think It – Model It, https://www.ontolis.com/
3. Ziegler, W. (2020). Extending intelligent content delivery in technical communication by semantics: microdocuments and content services. Proceedings of the ETLTC ACM Chapter International Conference. Aizuwakamatsu, Japan. (2020) https://doi.org/10.1051/shsconf/20207703009
4. Ziegler, W. (2021). Semantic Correlation Rules as a Logic Layer between Content Management and Content Delivery. Proceedings of the ETLTC ACM Chapter International Conference. Aizuwakamatsu, Japan. (2021)
5. tekom EUROPE, tekom Europe e.V., https://www.technical-communication.org/
6. iiRDS, tekom iiRDS Standard, *https://iirds.org/fileadmin/iiRDS_specification/20201103-1.1-release/index.html#metadata-location*
7. Ziegler, W. (2016). Content management systems tested: Introducing the PI-Fan, tcworld e-magazine 10/2016
8. EC-Systems, The Smart Media Creator, https://www.ec-systems.de/
9. I-views, The Smart Data Engine, https://i-views.com/en/
10. iiRDS, iiRDS Open Toolkit, *https://iirds-ot.plusmeta.de/iirds-open-toolkit/start*
11. Baro, D. / Neundörfer, L. (2021). Contrasting Ontology Modeling with Correlation Rules for Delivery Applications. Proceedings of the ETLTC ACM Chapter International Conference. Aizuwakamatsu, Japan. (2021)
12. Bischoff, S. / Fuchs, J. (2021). Creating Content and Use Cases for Testing Semantic Correlation Rules in Content Delivery. Proceedings of the ETLTC ACM Chapter International Conference. Aizuwakamatsu, Japan. (2021)