Supporting Transparent Information/Knowledge Federation in Collaborative Administrative Environments

Beibei Pang[1(✉)], Hamideh Afsarmanesh[2], Juanqiong Gou[1], and Wenxin Mu[1]

[1] School of Economics and Management, Beijing Jiaotong University,
Beijing, China
{16113125, jqgou, wxmu}@bjtu.edu.cn
[2] University of Amsterdam, Amsterdam, The Netherlands
h.afsarmanesh@uva.nl

Abstract. Leading edge ICT facilitates obtaining and interoperating information within collaborative networks (CNs), providing the base to tackle more advanced challenges. The paper addresses provision of transparent federated information/knowledge within administrative CNs. We introduce a methodology and mechanisms for incremental ontology development. The paper first identifies four typical sources of information/knowledge at the organizations involved in targeted emerging CNs, including: (i) database schemas, (ii) mission statements and main application scenarios, (iii) textual communications, and (iv) governance policies. It then introduces a systematic methodology to develop their meta-data and unify them into an ontology. This methodology consists of four semi-automated steps to gradually develop and enhance an ontology for the environment. The paper describes and exemplifies these steps and their mechanisms. An example real emerging case in the field of higher education administration in China is presented to serve as the proof of concept and verification of our proposed solution approach.

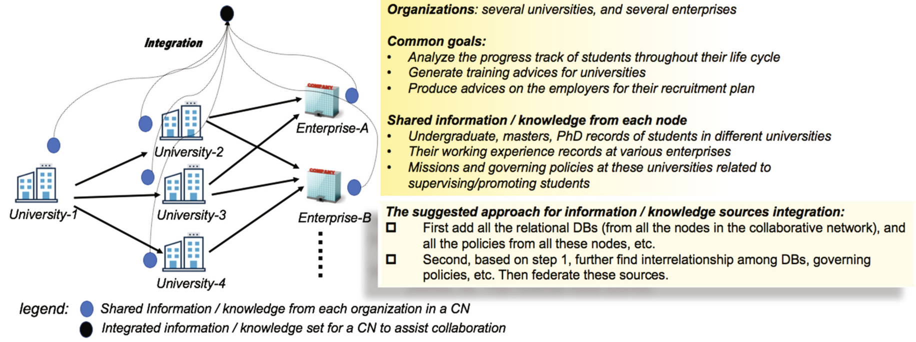
Keywords: Collaborative network · Knowledge federation · Unified ontology

1 Introduction

"A collaborative network (CN) is an alliance constituted by a variety of entities (e.g. organizations and people) that are largely autonomous, geographically distributed, and heterogeneous in terms of their operating environments, culture, values, and goals, but that collaborate to better achieve common or compatible goals, and whose interactions are achieved through computer networks [1]." Advanced ICT and emerging technologies provide the base to facilitate obtaining, sharing and exchange of various types of information/knowledge in collaborative networks [2]. Focusing on administrative CNs, we consider the application case of federating varied information/ knowledge sources as for instance illustrated in Fig. 1. Suppose that there are several organizations including some universities and enterprises that collaborate to achieve the following common goals: (i) obtaining the progress track of students throughout their life cycle;

Published by Springer Nature Switzerland AG 2019
L. M. Camarinha-Matos et al. (Eds.): PRO-VE 2019, IFIP AICT 568, pp. 205–219, 2019.
https://doi.org/10.1007/978-3-030-28464-0_19

(ii) generating training advices to universities on how to supervise and promote good students, and (iii) producing advices to employers for their recruitment plans, according to students' background experience and school performance. To achieve these common goals, it is necessary to first identify the information/ knowledge to be shared among these organizations, such as the undergraduate, masters, PhD records of students in different universities, their working experience records at various enterprises, as well as the missions and governing policies at these universities related to supervising/ promoting students. Furthermore, it is necessary to then integrate and federate these information/knowledge, generating both a unified and transparent pool that can be accessed by all actors in this environment, e.g. from the students and staff at universities to analysts and decision makers at enterprises, while supporting fair analysis of all students in this environment.


Integration
University-1
University-2
University-3
University-4
Enterprise-A
Enterprise-B
COMPANY
COMPANY
legend: ● Shared Information / knowledge from each organization in a CN
● Integrated information / knowledge set for a CN to assist collaboration

***Organizations**: several universities, and several enterprises*

Common goals:

- *Analyze the progress track of students throughout their life cycle*
- *Generate training advices for universities*
- *Produce advices on the employers for their recruitment plan*

Shared information / knowledge from each node

- *Undergraduate, masters, PhD records of students in different universities*
- *Their working experience records at various enterprises*
- *Missions and governing policies at these universities related to supervising/promoting students*

The suggested approach for information / knowledge sources integration:

□ First add all the relational DBs (from all the nodes in the collaborative network), and all the policies from all these nodes, etc.

□ Second, based on step 1, further find interrelationship among DBs, governing policies, etc. Then federate these sources.

Fig. 1. An example application case of CN in higher education environment

In this process, we first identify four main kinds of information/knowledge sources, including: (i) relational databases, (ii) existing mission statements and example application scenarios, usually characterized by their fragmented, lightweight and behavior-intensive features, (iii) textual communications among its stakeholders, mostly gathered through fragmented application cases and (iv) governance policies. Each of the above organizations, being a university or an enterprise, in this CN has one or more kinds of these four information/knowledge sources. In order to realize the federation of all these heterogeneous knowledge sources, we first analyze each data source, identify some inherent challenges, and define its related meta-data applying object-oriented principles, and second extract semantic relations among all these different pieces of information/knowledge. We then create a unified ontology for this collaborative environment, and formalize it using the OWL [3]. In our approach, we consider and apply current state of the art approaches for integrating databases, and for gathering governing policies [2] [8]. Thus, the paper does not address these aspects in details, and rather focuses on challenges that are not yet addressed. We specifically describe and tackle the following obstacles in this paper, since they are faced in achieving the information/knowledge federation goal in administrative CNs:

- For relational database schemas, knowledge is typically represented as tables, and attributes are classified as primary key, foreign key, etc. These are typically captured using the data definition language (DDL). The challenge faced here is to automate transforming relational schemas represented as DDL information, into the OWL elements for the unified ontology.
- For extracting meta-data from application scenarios, typically their knowledge from every source has completely different organizational structure, and the relationship between different knowledge pieces is not well expressed. There are also usually some knowledge overlaps with the knowledge presented through the relational schema. The challenge faced here is how to semi-automatically deal with resolving these problems.
- For textual data gathered from different communications in the environment, data is usually recorded together with some timestamps. The challenge faced here is first to convert these into structured formats, and then to automatically extract semantic information from text corpora, and generate their meta-data.
- For generating meta-data related to the governing policies of the environment, since their expression formats are quite flexible, only few studies have so far treated them. But in fact these represent an important knowledge source in administrative environments. This is especially the case for capturing the temporal data behavior that is usually hidden elsewhere, and only present in governing policies. Therefore, the challenges faced here are complex and at present we can only manually identify and formalize these temporal data behaviors in order to represent them in OWL format for the ontology.

Aiming to address the above-mentioned obstacles, we introduce our systematic methodology to knowledge source's meta-data unification that consists of four semi-automated steps, that gradually develop a unified ontology for the environment, formalized in OWL. This article is structured according to the following sections. Section 2 represents the related work. Sections 3 and 4 describe a methodology to facilitate identification and resolution of the main encountered typical inconsistencies among heterogeneous knowledge sources within collaborative environments. Section 5 concludes this research work and provides some perspectives for future plans.

2 Related Work

Influenced by [3], we define knowledge as the set of collected information together with their context, which could be understood, formatted, and shared without ambiguity by the environment stakeholders. In this paper, we aim at developing an ontology to support collaboration within administrative CNs. We address different kinds of information and/or knowledge that can be shared by the involved organizations. Please note that for simplicity reasons, in the remaining of this paper we mostly refer to the information/knowledge as "knowledge", and to the sources of information and/or knowledge as "knowledge sources". We then focus on generating the common/unified meta-data from all addressed sources. Our related research review focuses on three main challenging aspects: (i) unification of heterogeneous knowledge sources, (ii) specification and management of governing policies, and (iii) topic modeling for the content of textual communications.

2.1 Unification of Heterogeneous Knowledge Sources

Research areas related to unification of heterogeneous knowledge sources in collaborative networks either address the ontology based knowledge integration [4], or the ontology based data base integration [2, 5]. The unified ontology represents all types of data in uniform format and realizes intelligent analysis on integrated data sets [4]. It can also guide the integration of heterogeneous data bases [5]. State of the art in ontology based unification methods can be summarized as: first identifying different knowledge sources [2, 6], second formalizing and generating ontology for each source, and third integrating ontologies by using semantic similarity (i.e. graph based or content based) research methods [2, 7]. However, the state of the art research primarily focuses on addressing similar types of knowledge sources, such as research on integrating a number of databases [5], or research on integrating several XML documents [6]. In our research however, we design a methodology that handles four representative knowledge sources that are typically heterogeneous, and we aim to unify all these varied meta-data into one ontology. To the best of our knowledge, the current existing approaches have not yet addressed this problem area. We therefore propose a systematic approach to knowledge source unification for administrative CNs, through a methodology consisting of four semi-automated unification steps, gradually developing and enhancing the unified ontology for this environment.

2.2 Specification and Management of Governing Policies

A few studies capture and model the governance policies in the environment. one closely related research addresses enterprise modeling field [8], in which three relations are identified for business rules, namely the is-a relation, support relation, and hinder relation. For example, a governing rule states that: *"if the training plan of students that can be updated systematically every four years support our planned goal, then you must organize relevant revision work every four year".* Since policies are typically defined flexibly at every node in the network, they do not typically have a uniform format. In our research, we focus on extracting relevant semantics from governing policies in the environment, in order to integrate these with their related concepts in relational schemas or other meta-data. This in turn helps knowledge transfer between policy makers and executors, as well as benefiting the intelligent check whether these policies are being executed as expected. Semantics like temporal data behavior, complex causal relationship are identified by our methodology. In this paper however, we only address the temporal data behavior, and our approach is rooted in temporal data bases [9]. We will describe and formalize temporal data behavior patterns, reflected from governing policies in the environment.

2.3 Topic Modelling for the Content of Textual Communications

Topic models are commonly used to extract topics from texts, by simulating the human thinking process. Related topic models include Latent Semantic Analysis (LSA) [10], Probability Latent Semantic Analysis (PLSA) [11], and Latent Dirichlet Distribution (LDA) [12]. LSA breaks the previous thinking of text representation based on "dictionary space", and introduces a semantic dimension. However, the basis of the LSA methodology is derived from linear algebra, and the results of the operation are negative in many dimensions. Hofmann proposes a new method PLSA based on reliable probability statistics for the defects of LSA. But PLSA does not provide a probabilistic model at the document level, which leads to a linear increase in the number of parameters to be estimated in the model, depending on the size of the corpus. LDA has further extended the PLSA model by introducing a Dirichlet prior distribution. This approach overcomes the shortage of PLSA parameters as the document set grows linearly, thus forming a widely used probability topic model [13]. In our previous work, LDA model has been used in the first step of extracting domain knowledge in education field [14]. In this paper, LDA model will be used to enhance the ontology's data properties through parsing the content of textual communications.

3 Research Approach

We use the application case mentioned in Sect. 1 as the input and proof of concept for our approach. The considered four knowledge sources include: source #1: integrated relational schemas of databases from universities and enterprises; source #2: gathered meta-data from application scenarios, such as students take part in lectures information; source #3: gathered textual communications between students and education staff; source #4: gathered governing policies from universities.

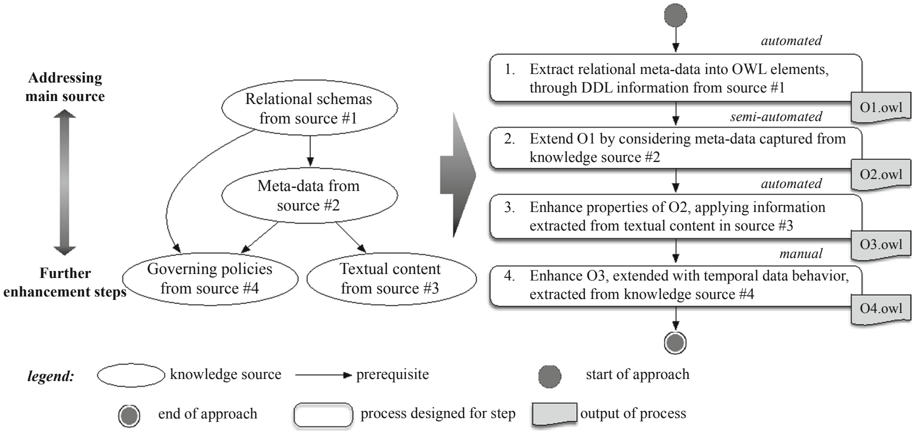

Addressing main source
Further enhancement steps
Relational schemas from source #1
Meta-data from source #2
Governing policies from source #4
Textual content from source #3
automated
1. Extract relational meta-data into OWL elements, through DDL information from source #1
O1.owl
semi-automated
2. Extend O1 by considering meta-data captured from knowledge source #2
O2.owl
automated
3. Enhance properties of O2, applying information extracted from textual content in source #3
O3.owl
manual
4. Enhance O3, extended with temporal data behavior, extracted from knowledge source #4
O4.owl
legend:
knowledge source
prerequisite
start of approach
end of approach
process designed for step
output of process

Fig. 2. Unification methodology flowchart

We introduce our step-wise knowledge federation methodology as depicted in Fig. 2. The methodology starts by first tackling the main source #1. This source contains the basic structure and conceptual model of the main entities in the environment, thus providing the tarp for our unification process. Second, the meta-data from source #2 will be generated from application scenarios, in order to extend the basic structure from source #1. Third, from source #3, some semantic information can be extracted to further extend the generated unified model. Furthermore, another important source is the governing policies. In our approach, temporal data behaviors are extracted from these policies. Since description objects related to governing policies are mainly defined on top of the knowledge sources mentioned above, it is necessary to first formalize the above knowledge and then integrate the governing policies. As shown on the right half of Fig. 2, in our proposed methodology, steps 1 and 3 are fully automated, while step 2 is semi-automated, and step 4 is manual. Every step enhances/extends the ontology generated in previous step, as shown by O1 to O4.

4 Detailed Meta-data Unification Methodology

In order to better explain the addressed knowledge types and their meta-data, from each of the four knowledge sources, we provide some simple and easy to understand examples for each discussed aspects. Please note that to help with better understanding of the examples in this section, a partial information/knowledge from each of the four sources are provided as annex at the end of this paper.

4.1 Step 1: Extract Relational Meta-data to OWL Through DDL Information

This step turns relational schemas into OWL. Some relevant examples are shown in Table 1. The Algorithm 1 in Fig. 3 represents this process and its three functions. Function F1.1 converts each table to a class. Function F1.2 converts attributes (if not foreign keys) to data properties, and adds the related class generated by function F1.1 as the domain class for each data property. Function F1.3 converts the relational foreign keys to object properties one by one, and specifies their respective domain and range classes according to the reference relation specified in the schema.

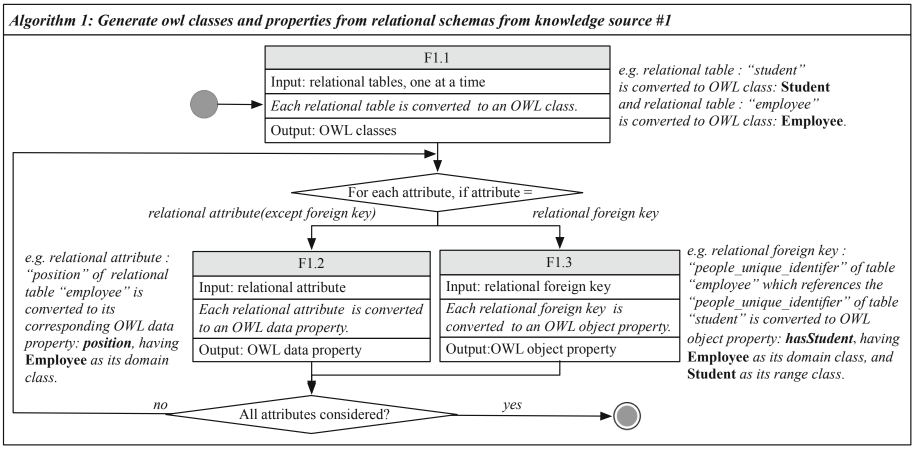

Algorithm 1: Generate owl classes and properties from relational schemas from knowledge source #1
F1.1
Input: relational tables, one at a time
Each relational table is converted to an OWL class.
Output: OWL classes
e.g. relational table : "student" is converted to OWL class: Student and relational table : "employee" is converted to OWL class: Employee.
For each attribute, if attribute =
relational attribute(except foreign key)
relational foreign key
F1.2
Input: relational attribute
Each relational attribute is converted to an OWL data property.
Output: OWL data property
F1.3
Input: relational foreign key
Each relational foreign key is converted to an OWL object property.
Output:OWL object property
e.g. relational attribute : "position" of relational table "employee" is converted to its corresponding OWL data property: position, having Employee as its domain class.
e.g. relational foreign key : "people_unique_identifer" of table "employee" which references the "people_unique_identifier" of table "student" is converted to OWL object property: hasStudent, having Employee as its domain class, and Student as its range class.
no
All attributes considered?
yes

Fig. 3. Algorithm 1: Generates OWL classes and properties from relational schemas

4.2 Step 2: Extend O1 Through Meta-data from Knowledge Source #2

Unlike the concise definition provided by relational schemas, typically there is no uniform specification for meta-data from knowledge source #2. Here, usually the meta-data only contains some tables and some definition of their related fields. So in this step, it is necessary to first manually analyze and organize these meta-data as explained below, and then to formalize them further in OWL. Some relevant examples are shown in Table 2. In our proposed method, fields of tables from the source #2 are specifically classified into the following four categories:

(a) *sameAs* relation

The *sameAs* relation means that the field is already addressed and exists either in O1 or in another already categorized extracted table from the source #2, e.g. *user_name* of table *take_extracurricular_lecture* is *sameAs student_name* of class *Student* in O1.

(b) user defined relation

This is defined in special situations of a field in a table when the domain class is not the class that corresponds to its own table in the meta-data. e.g. *mobile* field in table *take_extracurricular_lecture* is intended to describe the students' mobile contact information. Therefore, this field is not semantically related to taking extracurricular lectures. Rather, it will be converted to a data property of the class *Student* in OWL.

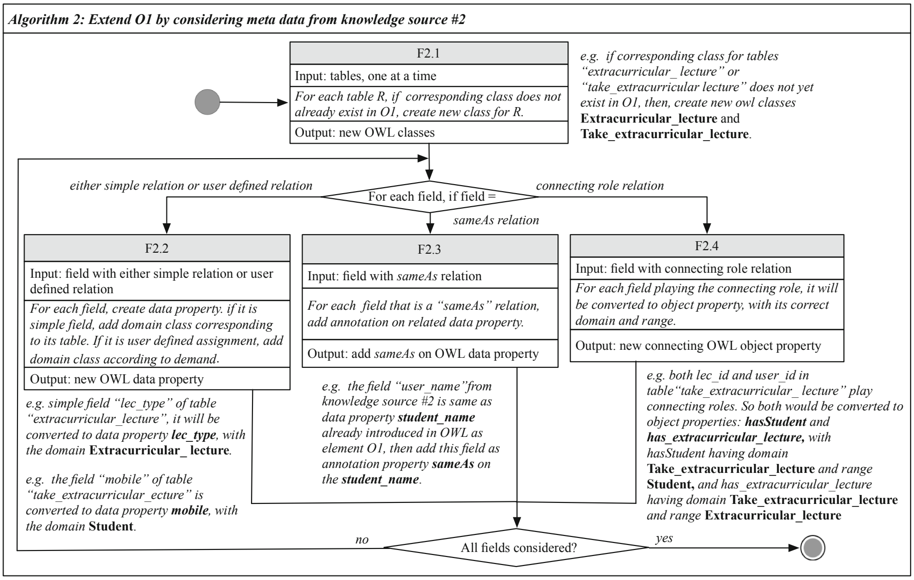

Algorithm 2: Extend O1 by considering meta data from knowledge source #2
F2.1
Input: tables, one at a time
For each table R, if corresponding class does not already exist in O1, create new class for R.
Output: new OWL classes
e.g. if corresponding class for tables "extracurricular_ lecture" or "take_extracurricular lecture" does not yet exist in O1, then, create new owl classes Extracurricular_lecture and Take_extracurricular_lecture.
either simple relation or user defined relation
For each field, if field =
connecting role relation
sameAs relation
F2.2
Input: field with either simple relation or user defined relation
For each field, create data property. if it is simple field, add domain class corresponding to its table. If it is user defined assignment, add domain class according to demand.
Output: new OWL data property
F2.3
Input: field with sameAs relation
For each field that is a "sameAs" relation, add annotation on related data property.
Output: add sameAs on OWL data property
F2.4
Input: field with connecting role relation
For each field playing the connecting role, it will be converted to object property, with its correct domain and range.
Output: new connecting OWL object property
e.g. simple field "lec_type" of table "extracurricular_lecture", it will be converted to data property lec_type, with the domain Extracurricular_ lecture.
e.g. the field "mobile" of table "take_extracurricular_ecture" is converted to data property mobile, with the domain Student.
e.g. the field "user_name"from knowledge source #2 is same as data property student_name already introduced in OWL as element O1, then add this field as annotation property sameAs on the student_name.
e.g. both lec_id and user_id in table"take_extracurricular_ lecture" play connecting roles. So both would be converted to object properties: hasStudent and has_extracurricular_lecture, with hasStudent having domain Take_extracurricular_lecture and range Student, and has_extracurricular_lecture having domain Take_extracurricular_lecture and range Extracurricular_lecture
no
All fields considered?
yes

Fig. 4. Algorithm 2: Extends O1 by considering meta-data from knowledge source #2

(c) connecting role relation

Some fields play the role of connecting two tables, or connecting a table with a class. e.g. consider if table *take_extracurricular_lecture* and table *extracurricular_lecture* both have the field *lec_id*, and the *lec_id* in table *take_extracurricular_lecture* is used to describe which lectures are considered as extra-curricular. So in OWL the *lec_id* in table *take_extracurricular_lecture* will be converted to an object property, in order to link the class *take_extracurricular_lecture* and the class *extracurricular_lecture* with each other, as created by function F2.1 in Fig. 4.

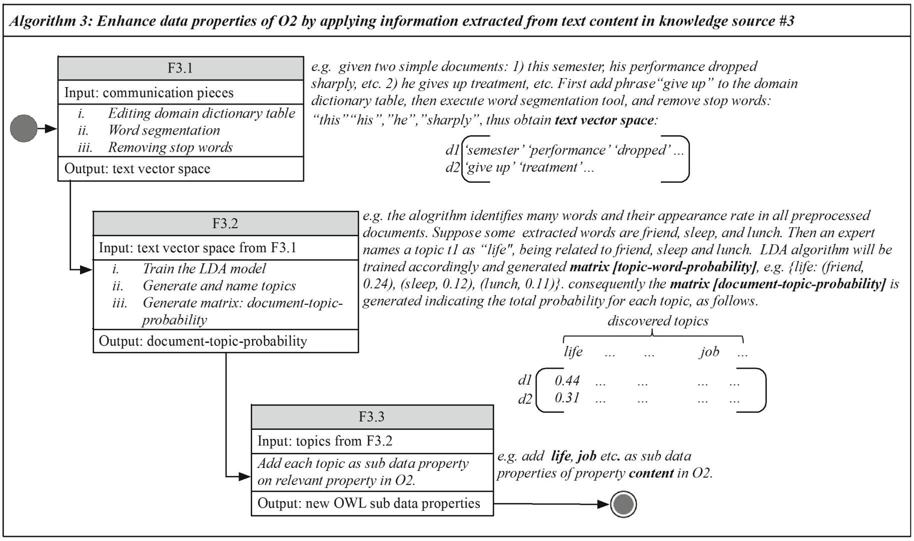
(d) simple relation

We call all other fields that do not satisfy the above three classifications simple fields, e.g. the *lec_type* of table *extracurricular_lecture.*

After manually sorting out the above four kinds of fields, our algorithm 2 will be executed to extend the output of step 1 of the methodology, as also shown in Fig. 4. There are four functions defined in algorithm 2. Function F2.1 converts tables one by one to classes, if their corresponding classes are not yet present in O1. Based on result of function F2.1, Function F2.2 handles the simple and user defined relations, and accordingly adds their suitable domain classes. In function F2.3, fields with *sameAs* relation will appear as annotations on their related data properties. In function F2.4, each field that plays a connecting role will be converted to an object property, and its domain and range classes will be defined according to its role.

4.3 Step 3: Extend Data Properties of O2 with Knowledge Source #3

Rooted in our earlier study [14], we introduce our approach in step 3. There are three main sub steps that generate new meta-data from the recorded text provided through this knowledge source, and further extend the O2 as sub data properties. We first briefly address how the gathered texts are preprocessed and then describe how semantics are extracted from them, in order to create new meta-data elements in OWL. Some relevant examples are shown in Table 3. More information about our approach to automate step 3, related to functions F3.1 and F3.2 are described below (Fig. 5).



Algorithm 3: Enhance data properties of O2 by applying information extracted from text content in knowledge source #3

F3.1
Input: communication pieces
i. Editing domain dictionary table
ii. Word segmentation
iii. Removing stop words
Output: text vector space

e.g. given two simple documents: 1) this semester, his performance dropped sharply, etc. 2) he gives up treatment, etc. First add phrase"give up" to the domain dictionary table, then execute word segmentation tool, and remove stop words: "this""his","he","sharply", thus obtain ***text vector space****:*

d1 'semester' 'performance' 'dropped' ...
d2 'give up' 'treatment'...
F3.2
Input: text vector space from F3.1
i. Train the LDA model
ii. Generate and name topics
iii. Generate matrix: document-topic-probability
Output: document-topic-probability

e.g. the alogrithm identifies many words and their appearance rate in all preprocessed documents. Suppose some extracted words are friend, sleep, and lunch. Then an expert names a topic t1 as "life", being related to friend, sleep and lunch. LDA algorithm will be trained accordingly and generated ***matrix [topic-word-probability]****, e.g. {life: (friend, 0.24), (sleep, 0.12), (lunch, 0.11)}. consequently the* ***matrix [document-topic-probability]*** *is generated indicating the total probability for each topic, as follows.*

e.g. add ***life, job etc.*** *as sub data properties of property* ***content*** *in O2.*

Fig. 5. Algorithm 3: Enhances data properties of O2, by applying information extracted from text content in knowledge source #3 (for more description of these examples, see [14])

4.3.1 Data Pre-processing by Function F3.1

Preprocessing of the acquired text data involves the following tasks. Each piece of recorded communication represents a “document” in this process. Pre-processing program first removes all carriage return characters in each document, then assembles all documents into one “integrated document”, while separating the original documents by adding carriage return characters. In the next step, the document set is word segmented. In order to improve the accuracy of word segmentation, a domain dictionary table and a stop-use dictionary table are manually constructed, as described below under (1) and (2). Clearly, the language for the content presented in the two tables must be with the same as the language for the text being processed.

(1) The domain dictionary table avoids incorrect segmentation of domain-specific words (e.g. in English, “give up” into “give” and “up”) by word segmentation tools.
(2) The stop-use dictionary keeps track of meaningless words such as “the” and “in” that appear in the document of word segmentation.

4.3.2 Topic Based Semantic Extraction by Function F3.2

We apply the LDA model [12], mentioned in Sect. 2.3, as follows:

(1) ***LDA topic modeling*** – “Topic” represents a concept and the conditional probability of a series of words. Each word in a document is characterized by the process of “selecting a topic t with a certain probability, and selecting a certain word w from the topic t with a certain probability.” So for a document d, the probability of each word appearing in it can be calculated by: $p(w|d = \sum_t p(w|t) \times p(t|d))$ In this formula, w is word, d is document, t is topic, and p is probability. For one document, this can be represented by $C = \Phi \times \Theta$, as in the following matrix (Fig. 6):

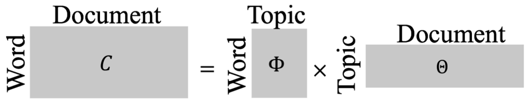


Fig. 6. Topic modeling theory

The “document-word” matrix represents the word appearance’s frequency in each document. The “topic-word” matrix represents the probability of occurrence of each word in each topic. The “document-topic” matrix represents the frequency of each topic appearing in each document. Given the pre-processed document set, the “document-word” matrix on the left can be obtained by segmenting different documents, and calculating the frequency of each word in each document.

Our topic model is then trained by learning from the matrix on the left, to derive the two matrices on the right. For this training, we apply the Dirichlet distribution [12], which identifies appropriate number of topics in document set. The basic idea there is to identify all topics when similarity between the topics is the smallest. Therefore, appropriate numbers of topics will be identified by the LDA method.

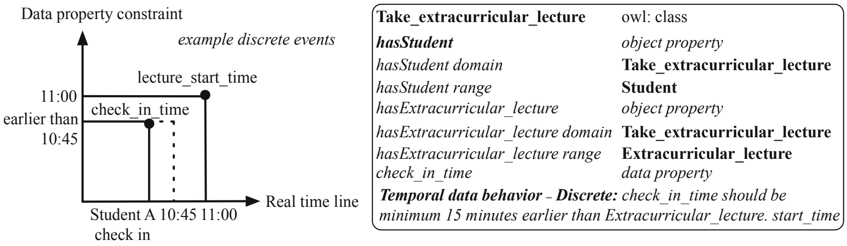
(2) ***Naming each topic*** – Here, we apply the LDA model and generate the set of (*topicID*, *word*, *probability*), that describe the distribution of words that are related to each topic. Combined with the domain knowledge, the topic names are defined.

(3) ***Generating topics distribution of each document*** – We apply the LDA topic model to generate the set of (*documentID*, *topicID*, *probability*), which represent the probability distribution of each document under each topic. Combined with the set generated in (2), we therefore produce: (*documentID*, *topicName*, *probability*).

4.4 Step 4: Extend O3 by Governing Policies from Knowledge Source #4

This step focuses on extraction of "temporal data behavior" concepts, e.g. related to environment policies (relevant examples see Table 4), as necessary regulation constraints can enhance the conceptual model of the collaborative environment. For the interest of this paper, three kinds of behavior for temporal entities are considered and classified, as described below:

(i) Discrete temporal behavior – Discrete temporal data properties represent events that can be recorded only at specific points in time, such as check in time of every student for a lecture or the lecture's start time, then for example, a discrete temporal behavior rule related to such a lecture will state that the value of check in time for a student should be minimum 15 min earlier than the value of start time. This time behavior can be modeled as a time constraint on data property on the defined values (Fig. 7).



Take_extracurricular_lecture	owl: class
hasStudent	*object property*
hasStudent domain	**Take_extracurricular_lecture**
hasStudent range	**Student**
hasExtracurricular_lecture	*object property*
hasExtracurricular_lecture domain	**Take_extracurricular_lecture**
hasExtracurricular_lecture range	**Extracurricular_lecture**
check_in_time	*data property*

Temporal data behavior – Discrete: *check_in_time should be minimum 15 minutes earlier than Extracurricular_lecture. start_time*

Fig. 7. Example of temporal rule on discrete behavior

(ii) Stepwise & constant temporal behavior – Here we mainly consider two specific cases of stepwise & constant temporal behavior, as addressed below.

(a) Situation1: constant step duration – As an example of a time constraint on class instances, suppose that since a decade ago the definition of the students' training plan changes once every 4 years. Given this policy, the behavior of every instance in class “Training plan” while being constant, changes with steps of exactly 4 years (Fig. 8).

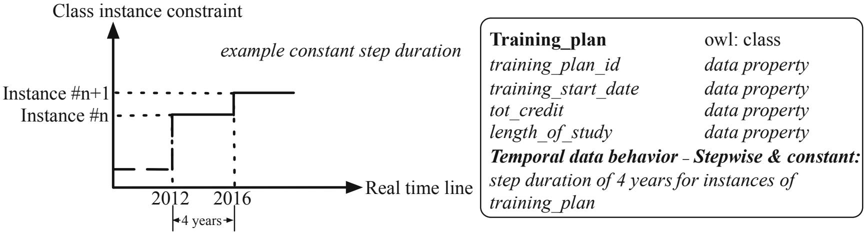

Class instance constraint
example constant step duration
Instance #n+1
Instance #n
2012
2016
4 years
Real time line
Training_plan owl: class
training_plan_id data property
training_start_date data property
tot_credit data property
length_of_study data property
Temporal data behavior – Stepwise & constant:
step duration of 4 years for instances of training_plan

Fig. 8. Example of temporal rule on constant step duration behavior

(b) Situation2: variable step duration – As an example of a time constraint on data properties, consider different governing policies related to the minimum required outcomes to pass a postgraduate innovation project at a school. Suppose that between Sept.2013 and Sept.2017, the minimum required outcomes were two papers, but that after Sept.2018, the required outcomes are either 1 high impact publication or 3 papers. Valid time for the requirements can be represented by the interval: (t1, t2), where t1 and t2 correspond to the start and end date of the period respectively. In order to also support the case when the end time is not known, we introduce ** symbol that indicates the expiration date would be the date of next potential start time (Fig. 9).

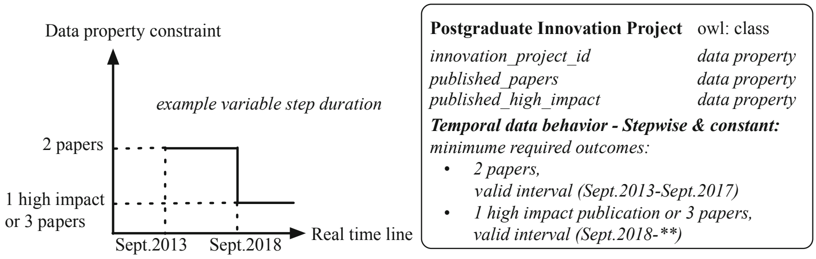
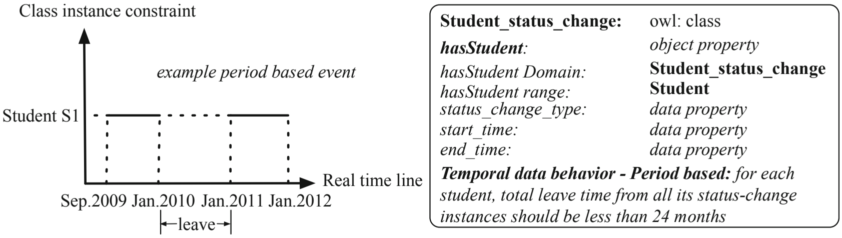

Data property constraint
example variable step duration
2 papers
1 high impact
or 3 papers
Sept.2013
Sept.2018
Real time line
Postgraduate Innovation Project owl: class
innovation_project_id data property
published_papers data property
published_high_impact data property
Temporal data behavior - Stepwise & constant:
minimume required outcomes:
• 2 papers,
valid interval (Sept.2013-Sept.2017)
• 1 high impact publication or 3 papers,
valid interval (Sept.2018-**)

Fig. 9. Example of temporal rule on variable step duration behavior

(iii) Period based temporal behavior – As another example of time constraint on class instances, consider a period-based governing policy to capture the behavior of events that may occur on each entity (e.g. a student) in an environment, over a period of time (e.g. study in a program). For example, a policy can state that for each student, the total number of months of leave from school cannot go over 24 months (Fig. 10).

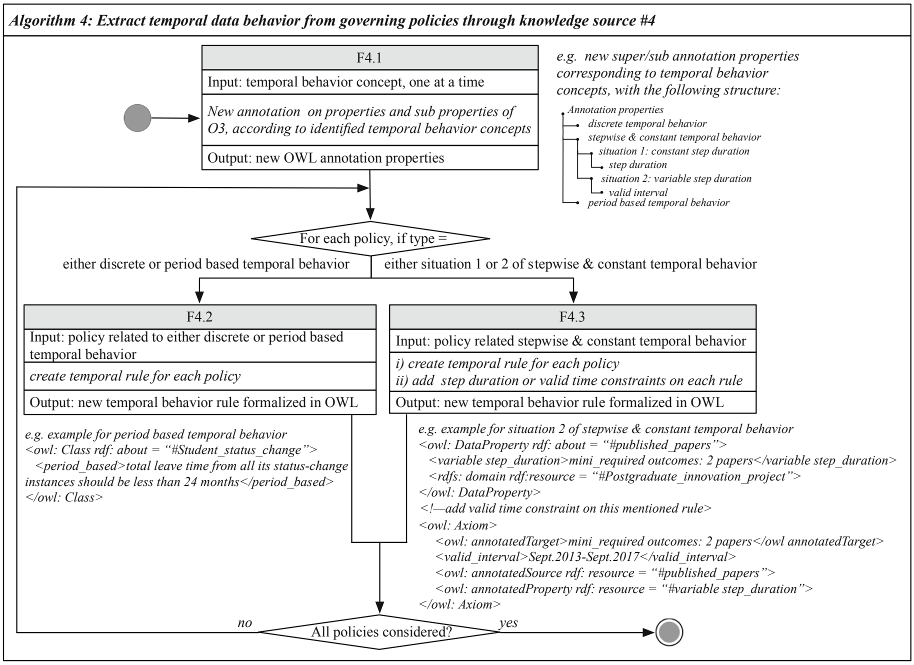


Student_status_change:	owl: class
***hasStudent**:*	*object property*
hasStudent Domain:	**Student_status_change**
hasStudent range:	**Student**
status_change_type:	*data property*
start_time:	*data property*
end_time:	*data property*

***Temporal data behavior - Period based:** for each student, total leave time from all its status-change instances should be less than 24 months*

Fig. 10. Example of temporal rule on period based behavior

The Algorithm 4 and its three functions in Fig. 11 represents the process we introduce for step 4 of our approach. In function F4.1, temporal behavior concepts mentioned above are added into O3 as annotation properties. Function F4.2 extracts discrete or period based temporal behavior from governing policies. In function F4.3, two situations of stepwise & constant temporal behavior are considered, compared with function F4.2, an important process here is adding of step duration or valid time constraints on each related rule. Therefore, O4 is generated as the output of this step, as well as the final output of our knowledge federation approach.


Algorithm 4: Extract temporal data behavior from governing policies through knowledge source #4
F4.1
Input: temporal behavior concept, one at a time
New annotation on properties and sub properties of O3, according to identified temporal behavior concepts
Output: new OWL annotation properties
e.g. new super/sub annotation properties corresponding to temporal behavior concepts, with the following structure:
Annotation properties
discrete temporal behavior
stepwise & constant temporal behavior
situation 1: constant step duration
step duration
situation 2: variable step duration
valid interval
period based temporal behavior
For each policy, if type =
either discrete or period based temporal behavior
either situation 1 or 2 of stepwise & constant temporal behavior
F4.2
Input: policy related to either discrete or period based temporal behavior
create temporal rule for each policy
Output: new temporal behavior rule formalized in OWL
F4.3
Input: policy related stepwise & constant temporal behavior
i) create temporal rule for each policy
ii) add step duration or valid time constraints on each rule
Output: new temporal behavior rule formalized in OWL
no
All policies considered?
yes

e.g. example for period based temporal behavior

```
<owl: Class rdf: about = “#Student_status_change”>
  <period_based>total leave time from all its status-change
instances should be less than 24 months</period_based>
</owl: Class>
```

e.g. example for situation 2 of stepwise & constant temporal behavior

```
<owl: DataProperty rdf: about = “#published_papers”>
  <variable step_duration>mini_required outcomes: 2 papers</variable step_duration>
  <rdfs: domain rdf:resource = “#Postgraduate_innovation_project”>
</owl: DataProperty>
<!—add valid time constraint on this mentioned rule>
<owl: Axiom>
   <owl: annotatedTarget>mini_required outcomes: 2 papers</owl annotatedTarget>
   <valid_interval>Sept.2013-Sept.2017</valid_interval>
   <owl: annotatedSource rdf: resource = “#published_papers”>
   <owl: annotatedProperty rdf: resource = “#variable step_duration”>
</owl: Axiom>
```

Fig. 11. Algorithm 4: Extracts temporal data behavior from governing policies through knowledge source #4

4.5 Discussion

This section provides details of our proposed methodology. We extract/federate meta-data from four kinds of knowledge sources, as mentioned in Sect. 1. The meta-data extracted from integrated relational schemas provide the tarp for our unification approach, while information from other sources are used to extract semantics to further enhance the unified ontology. The other considered sources include: (i) meta-data generated from mission statements and application scenarios, (ii) meta-data extracted from textual communications, through LDA model that are converted to sub data properties used to enhance already generated data property, and (iii) meta-data generated from governing policies used to enhance conceptual models by adding time constraints on class instances or on data property values. Through our approach, these four kinds of heterogeneous knowledge sources are unified, and in turn effectively supporting knowledge interoperability in administrative CNs. Nevertheless, our proposed methodology is relatively general and applicable to other relevant administrative CN cases. A point of caution for the current approach is related to data privacy. In other words, the approach to providing information/knowledge transparency needs to be carefully adjusted to the type of environment stakeholders. This means that not all users, e.g. student, administrative staff, etc., can see everything transparently, rather the interface accessible to every kind of user, must concisely correspond to the user's level of information/ knowledge visibility.

5 Conclusion and Future Works

To realize federation of varied knowledge sources in administrative CNs, we propose a systematic methodology and a set of mechanisms for federation of four typical types of knowledge sources shared within collaborative environments. Our introduced mechanisms support semi-automation of the methodology steps and incremental generation of a unified ontology capturing all shared knowledge from heterogeneous sources. As a proof of concept, our approach is exemplified for a real emerging case in higher education administration environment. As the next steps of our research, we intend to address knowledge unification for several other types of sources that we identify in collaborative administration networks. These include: entity relationship diagrams (ERD), data dictionaries accessible from relational data bases, standard regulation documents, and data captured through cyber physical devices. We also intend to further tackle other kinds of temporal data behaviors, including those with complex causal relations, as well as addressing interactive application scenarios and information communicated among the environment stakeholders.

We are currently in the process of developing mostly automated mechanisms to handle knowledge unification, which will be addressed in forthcoming publications.

Acknowledgments. The present research has been partially supported by the Fundamental Research Funds for the Central Universities of China (2017YJS081) and Research on Behavioral Intelligence Based on Collaborative Scenarios in High Education Environment (2018A02008). This research is also partially supported by the ARCON-ACM project at the University of Amsterdam.

Annexed Tables

Table 1. Example of integrated relational schemas related to knowledge source #1

student (people_unique_identifier (PK), name, birth_date)
student_degree (student_id(PK), people_unique_identifier (FK), discipline_id(FK), entry_year, university_name, degree_type)
discipline(discipline_id(PK), title, start_time, end_time, teaching_language)
graduate_innovation_project (innovation_project_id (PK), project_manager_id(FK), published_papers, published_high_impact)
training plan (training_plan_id(PK), discipline_id(FK), training_start_date, tot_credit, length_of_study)
student_status_change (student_status_change_id(PK), student_id(FK), status_change_type, start_time, end_time)
employee (employee_id (PK), people_unique_identifier (FK), position, enterprise_name, salary_level, entry_date)

Table 2. Example of gathered meta-data related to knowledge source #2

extracurricular_lecture		take_extracurricular_lecture		communication_information	
lec_id	end_time	lec_id	check_in_time	staff_id	end_time
lec_title	total_nmu	user_id	mobile	stu_id	content
start_time	lec_type	user_name		start_time	

Table 3. Example of communication's content related to knowledge source #3

Example of one document:
Yesterday, I had a conversation with Student A. He just received an intern offer from a company ranked in Fortune 500. This is an opportunity that many students dream of. However, since he is going to take an important National Civil Servant Examination, it seems hard for him to balance both, and he looked really anxious.

Table 4. Example of gathered governing policies related to knowledge source #4

a. Between Sept.2013 and Sept.2017, the minimum required outcomes to pass a postgraduate innovation project were two papers, but that after Sept.2018, the required outcomes are either 1 high impact publication or 3 papers. (*from university-1*)
b. The value of check in time for a student should be minimum 15 minutes earlier than the value of extracurricular lecture's start time. (*from university-2*)
c. The definition of the students' training plan systematically changes once every 4 years. (*from university-3*)
d. For each student, total leave time from all its status-change instances should be less than 24 months. (*from university-1, university-2 and university-3*)

References

1. Camarinha-Matos, L.M., Afsarmanesh, H.: On reference models for collaborative networked organizations. Int. J. Prod. Res. **46**(9), 2453–2469 (2008)
2. Unal, O., Afsarmanesh, H.: Semi-automated schema integration with SASMINT. Knowl. Inf. Syst. **23**(1), 99–128 (2010)
3. Ekaterina, E.: Management of information in virtual organizations breeding environments. Ph.D. dissertation, University of Amsterdam, Netherlands (2014)
4. García, M.D.M.R., García-Nieto, J., Aldana-Montes, J.F.: An ontology-based data integration approach for web analytics in e-commerce. Expert Syst. **63**, 20–34 (2016)
5. Li, L., Wei, Y., Tian, F.: A Framework for ontology-based top-K global schema generation. J. Data Semant. **6**(1), 31–53 (2017)
6. Tekli, J., Charbel, N., Chbeir, R.: Building semantic trees from XML documents. Web Semant. **37**, 1–24 (2016)
7. Zhu, G., Iglesias, C.A.: Computing semantic similarity of concepts in knowledge graphs. IEEE Trans. Knowl. Data Eng. **29**(1), 72–85 (2017)
8. Sandkuhl, K., Stirna, J., Persson, A., Wißotzki, M.: Enterprise Modeling: Tackling Business Challenges with the 4EM Method. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43725-4
9. Date, C.J., Darwen, H., Lorentzos, N.: Temporal Data & the Relational Model. Morgan Kaufmann, San Francisco (2002)
10. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse Process. **25**(2–3), 259–284 (1998)
11. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, pp. 289–296. Morgan Kaufmann, San Francisco (1999)
12. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
13. Della Rocca, P., Senatore, S., Loia, V.: A semantic-grained perspective of latent knowledge modeling. Inf. Fusion. **36**, 52–67 (2017)
14. Pang, B., Gou, J., Mu, W.: Extracting topics and their relationship from college student mentoring. Data Anal. Knowl. Discov. **2**(6), 92–101 (2018). (in Chinese)