Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 81 (2019) 139–144

www.elsevier.com/locate/procedia

52nd CIRP Conference on Manufacturing Systems

Combining Channel Theory and Semantic Web Technology to build up a Production Capability Matching Framework

Andreas Bildstein[a,b,]*, Junkang Feng[a], Thomas Bauernhansl[b,c]

[a]*University of the West of Scotland, School of Computing, Engineering and Physical Sciences, Paisley PA1 2BE, UK*
[b]*Fraunhofer IPA, Nobelstraße 12, Stuttgart 70569, Germany*
[c]*University of Stuttgart, Institute of Industrial Manufacturing and Management, Allmandring 35, Stuttgart 70569, Germany*

* Corresponding author. Tel.: +49-711-970-1255; Mobile: +49-173-3161104. *E-mail address:* andreas.bildstein@ipa.fraunhofer.de

Abstract

One of the main theories behind the worldwide 'Factories of the Future' movement is the idea that machines and systems, and thus the production system, should reach a certain level of self-organisation. We support this self-organisation of the production system by introducing a production capability matching framework that is based on an extended application of Channel Theory combined with the extensive use of Semantic Web technologies like OWL ontologies and rules based on SWRL. This approach shows that the combination of Channel Theory and Semantic Web technology is able to lay the groundwork for an automated equipment assignment process in global and local value chains.

Keywords: Channel Theory; Capability Matching Framework; Semantic Web Technology;

1. Introduction

One of the main theories behind the worldwide 'Factory of the Future' movement (also known as smart manufacturing in the US or Industrie 4.0 in Germany) is the idea that products, production equipment, and production IT systems are getting far more interconnected as they are at the moment to finally reach a certain level of self-organization and autonomy.

An example of this self-organization is a production system, which decides without human intervention on which production equipment should be used to conduct a specific production step. Caused by globalisation and customer requirements for more customised and even personalised products, many manufacturing companies have to react to an increasing number of product variants while the number of sold products per product variant declines. This results in an increasing need for flexible production systems [1], which are able to handle a rising variety of different combinations of equipment and tools able to conduct a growing number of different and new products and thus production steps. To manage this increasing complexity in the decision on which equipment should be used to conduct a specific production step, both in local and global value chains, an automated equipment assignment solution might be desirable.

For this automated equipment assignment process, we propose the usage of a matching framework, which is based on an extended application of the theory of Information Flow (IF), also known as Channel Theory [2]. We introduced this extended application of IF in [3] and will now focus within this paper on building up the proposed production capability matching framework by combining the components for the application of Channel Theory in complex environments [3], [4] with an implementation approach based on Semantic Web technologies like OWL ontologies [5] and SWRL rules.

In the literature, Channel Theory and Information Flow are typically used synonymously – however, we will use Channel

10.1016/j.procir.2019.03.025

Theory when the theoretical underpinnings of the production capability matching framework are discussed and will use Information Flow when the components of Channel Theory are applied in the specific application scenario.

The proposed production capability matching framework compares the specifications for a specific production step with the capabilities of the available machinery and selects those machines that are able to conduct the production step. While matching the production steps with the production equipment, the production capability matching framework makes its decisions based on former successful pairings between production steps and machines, but also based on the characteristics of so far unknown specifications from new products and the properties of the available equipment.

In our application scenario, the production steps and the production equipment describe two different sets of things, which are however related to each other within the context of our application scenario. This informational relationship between two or more different sets of things qualifies our application scenario as a distributed system as it is described in [2].

We use Channel Theory as the theoretical groundwork for our matching framework, because Channel Theory provides us with the mathematical tools that help us to describe the flow of information within a distributed system and because Channel Theory has been successfully applied in a series of different scenarios where the relationships of two or more sets of things have to be determined [6], [7].

The application of IF as it is used within this paper stands in the tradition of these applications of Channel Theory as it is for example shown in detail in the work of Kalfoglou and Schorlemmer, e.g. in [8], or in the work of Yang and Feng, e.g. shown in [9], as well as in the work of other researchers as summarized in [7]. However, we enhance these approaches for the application of IF [3] and combine them with the implementation of an Information Flow-based capability matching framework based on Semantic Web technology.

2. Components for the application of Channel Theory

The main components for the application of Channel Theory are classifications, infomorphisms, correspondences and constraints. Those components are used to construct an IF channel and will be introduced within this chapter.

2.1. Channel Theory, classifications

In Channel Theory, a component of a distributed system is modelled with the help of a classification. This classification represents the context of this component and consists of particulars (objects) and attributes that help to describe those particulars. In Channel Theory, the particulars are named *tokens* and the attributes that classify those tokens are named *types*.

By classifying the tokens with types, a relation between the particulars and the attributes within a component is given. The individual components are modelled with the help of a simple mathematical structure that is called *classification*. Definition of a classification [1, p.69]:

"A classification A = $\langle \text{tok}(A), \text{typ}(A), \models_A \rangle$ consists of a set, tok(A), of objects to be classified, called the *tokens of A*, a set, typ(A), of objects used to classify the tokens, called the types of A, and a binary relation, $\models_A$, between tok(A) and typ(A)."

We depict a classification in Fig. 1:

Fig. 1. Classification A.

So, within a classification A, the binary relation classifies the tokens a_i of A to the types α_i of A in the form that the binary relation $\models_A$ is a subset of the Cartesian product between the types and the tokens of A, $\models_A \subseteq \text{tok}(A) \text{ X } \text{typ}(A)$.

Such a classification might be used, for example, to build up a context $ET_{Drilling}$, which consists of a set of different drilling machines tok($ET_{Drilling}$) and a set of production capabilities typ($ET_{Drilling}$), which those drilling machines provide.

2.2. Channel Theory, infomorphisms

In Channel Theory, classifications are connected with each other via so-called *infomorphisms*. The definition of an infomorphism can been found in [1, p.72]:

"An infomorphism f: $A \rightleftarrows B$ from A to B consists of two classifications $\langle A, B \rangle$ and a contravariant pair $f = \langle f^\wedge, f^\vee \rangle$ of functions between A and B, satisfying the following fundamental property of infomorphisms:

$$f^\vee(b) \models_A \alpha \quad \text{iff} \quad b \models_B f^\wedge(\alpha)$$

for each token b ∈ tok(B) and each type α ∈ typ(A)."

With the help of these infomorphisms, information can be carried back and forth between the component classifications. The information that is being carried is the fact that a specific token a is classified to a specific type α, meaning the information that a is being of type α. We can depict an infomorphism as in Fig. 2:

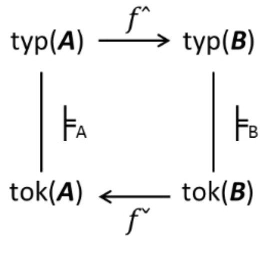

Fig. 2. Infomorphism f: A ⇄ B.

Between a component and the system as a whole, there is at least one such infomorphism. An infomorphism is a pair of functions each of which captures correspondences between types or between tokens of two classifications that comply with the above fundamental property. Moreover, through the system as a whole (which would be represented as the 'core' of the information channel, which will be discussed shortly), relationships between components are also captured.

With the help of such infomorphisms, relationships between the tokens or between the types of two components of an information channel are represented. An infomorphism on token-level, for example, can represent the information that a specific production step x ∈ tok($DD_{makingHole}$) can be conducted with machine y ∈ tok($ET_{Drilling}$).

2.3. Channel theory, correspondences

We have shown that an infomorphism is a pair of functions, which respectively represent the *correspondences* between the types and the tokens of classifications within a distributed system. The correspondences themselves represent relationships between types and between tokens of the involved classifications.

As such, so far known correspondences are an important starting point when we have to match between the different components of a distributed system. Those initial correspondences are the result of a priori knowledge or other kinds of heuristics, but can be also the result of a posteriori knowledge when we feed back our experience to this initial partial alignment. Thus, correspondences give us the information, what we already know about the relationships between the tokens or between the types of two contexts. In our application scenario, this might be, for example, the knowledge that the production step x of type $PS_{makingHole}$ can be conducted in the setting of a specific production environment with the manufacturing processes MP = {drilling, milling, turning, punching}.

In Information Flow, a correspondence is a pair of elements, which contains either two tokens or two types from the corresponding IF classifications and describes a particular relationship between the two component classifications. This pair of elements is then be used to build up a token t_x = <$t_{i,Ax}$;$t_{j,Bx}$> within the core classification of the IF channel and this token t_x is described by the types of the involved tokens from the IF classifications *A* and *B* that build up this token t_x.

2.4. Channel theory, constraints

According to the first principle of information flow [1, p.8], the flow of information heavily depends on regularities in the distributed system. The more random a distributed system is the less information is able to flow between the components of this distributed system. Thus, the aim is to find as much regularities as possible in a distributed system to reach a stable alignment framework between the different components, namely classifications, within the distributed system. In Channel Theory those regularities are called *constraints* and are defined as follows [1, p.29]:

"Let *A* be a classification and let $\langle \Gamma \mid \Delta \rangle$ be a sequent of *A*. A token *a* of *A* satisfies $\langle \Gamma \mid \Delta \rangle$ provided that if *a* is of type α for every $\alpha \in \Gamma$ then *a* is of type α for some $\alpha \in \Delta$. We say that Γ *entails* Δ in *A*, written $\Gamma \vdash_A \Delta$, if every token *a* of *A* satisfies $\langle \Gamma \mid \Delta \rangle$. If $\Gamma \vdash_A \Delta$ then the pair $\langle \Gamma \mid \Delta \rangle$ is called a *constraint* supported by the classification *A*."

According to the above definition of constraints, a sequent is a pair $\langle \Gamma \mid \Delta \rangle$ of sets of types from a classification. Following that, constraints provide regularities on type level in a classification. Together with the infomorphisms there are now mechanisms available that help us to align classifications from the different components in a distributed system with the help of an IF channel and based on regularities derived from some initial correspondences. The regularities within a distributed system are necessary to successfully establish a matching framework for this distributed system.

2.5. Channel theory, channels

The main aim in the application of the IF theory is the construction of a so-called *IF channel*. A channel is defined like follows [1, p.76]:

"A channel *C* is an indexed family $\{f_i: A_i \rightleftarrows C\}_{i \in I}$ of infomorphisms with a common codomain *C*, called the *core* of *C*. The tokens of *C* are called connections; a connection *c* is said to connect the tokens $f_i(c)$ for i ∈ I."

This definition from Barwise and Seligman is a general definition for an n-ary channel with an index set {0,…,n-1}. However, most of the examples in the literature about the application of IF are dealing with two components only and this is exactly what we need for our application scenario. So we are talking about a binary channel and for the case of a binary channel, we can stick to the following channel definition that is given by Schorlemmer and Kalfoglou [10]:

"An IF channel consists of two IF classifications A_1 and A_2 connected through a core IF classification *C* via two infomorphisms f_1 and f_2."

A binary channel can be depicted like this:

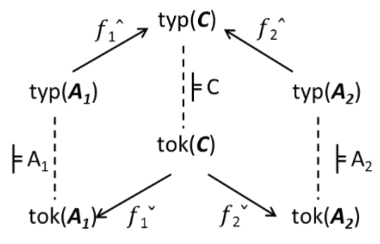

Fig. 3. Binary channel.

According to this description, an IF channel consists of a *core classification C*, two component classifications *A* and *B* and corresponding infomorphisms that connect the core classification with the component classifications. Additionally, the core of the IF channel is a classification whose tokens are connections between the tokens from the component classifications and whose types are the disjoint union of the types from the component classifications that are involved in the IF channel.

2.6. Channel theory, constructing the channel

The introduction showed that various researchers already successfully applied IF in a series of different scenarios where the relationships of two or more sets of things have to be determined in environments that can be seen as distributed systems. All those application scenarios have in common that for the construction of the IF channel they follow the approach that is described in the diverse work of Kalfoglou and Schorlemmer, e.g. in [11].

In the examples from the literature, the construction of the IF channel normally starts at the baseline of a set of initial correspondences as a partial alignment between the IF classifications of the distributed system. Those initial correspondences either represent known relationships between the tokens of the involved classifications or relationships between the types of the classifications of a distributed system.

3. Overview of the capability matching framework

The proposed production capability matching framework for our application scenario is concerned with the question of which machine is able to conduct a specific production step. Thus, the capability matching framework has to cover at least two contexts, namely one context for the production steps and their specifications that have to be conducted and one context for the production equipment that is provided by the production system to conduct these production steps. We design these two contexts as the components of the distributed system of our application scenario that are related to IF classifications in the application of IF. As our application scenario shows a level of complexity, which does not allow to match between production steps and suitable production equipment on one single step, we have to use a layered approach for the overall matching process [3], [4].

3.1. Production step repository

The building block for the context production steps is named *production step repository*. This building block is constituted by a component PS (see Fig. 4), which consists of a set of tokens tok(PS) and a set of types typ(PS). The tokens tok(PS) represent the individual production steps. The types typ(PS) are used to classify the tokens to a specific type of production step like making hole, trimming workpiece or make complex structure.

For the production steps, a detailed description of the production step specifications is needed as well. As these production step specifications may differentiate between the different production steps and also in combination with the manufacturing processes that might be used for one production step, also the detailed description of these production specifications may differ. We have to take care of these detailed descriptions in our system design and provide a separate component *detailed description* (DD) for them (see Fig. 4). The component DD consists of a set of tok(DD) and a set of types typ(DD). The tokens tok(DD) represent the production steps of a specific type, e.g. making hole, and the types typ(DD) are used to describe the characteristics of a specific production step, e.g. hole diameter, material, et al.

3.2. Production technology repository

The building block for the context equipment and tooling is named *production technology repository*. This building block is constituted by a component MP (see Fig. 4), which consists of a set of tokens tok(MP) and a set of types typ(MP). The tokens tok(MP) represent the specific manufacturing processes that are available, and each of these tokens is categorised to a manufacturing process, e.g. drilling, milling or turning. The production technology repository also encompasses the component ET_x (see Fig. 4) for the equipment and tooling related to the manufacturing processes from component MP. The component ET_x consists of a set of tokens tok(ET_x) and a set of types typ(ET_x). The tokens tok(ET_x) represent the different machines, which are set up with different tools and tooling related to the individual manufacturing processes $x \in$ typ(MP), and the types typ(ET_x) represent the different capabilities of tok(ET_x)—the machine characteristics and the available tooling—related to the individual manufacturing processes $x \in$ typ(MP) and categorise the tokens tok(ET_x).

3.3. Implementation of the capability matching framework

A high-level system overview of the IF capability matching framework is represented in Fig. 4. It can be seen in this drawing that the capability matching service builds the heart of the production capability matching framework and that the components of the framework are split into separate parts according to the layered approach for the matchmaking presented in [3].

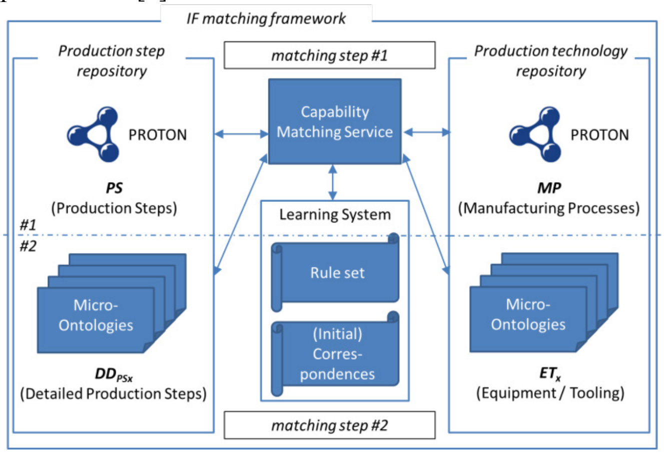

Fig. 4. IF matching framework, component overview.

Furthermore, Fig. 4 shows the IF classifications for the capability matching framework, namely the classification PS for the modelling of the production steps and the classification MP for the modelling of the manufacturing processes on layer 1 for matching step #1. Additionally, for the matching step #2 on layer 2, the classifications DD_{PSx} for the detailed description of the production steps and the classifications ET_x for the modelling of the production capabilities of the equipment and tooling are shown.

The heart of the IF matching framework, the capability matching service, "resides" between the two components production step repository and production technology repository and is also interconnected with the learning system component. The learning system component provides the capability matching service with the known relationships between the types and between the tokens of the related IF classifications, namely the constraints and initial correspondences of constructed channels on the different layers of the overall matching process. The capability matching service enquires these rules when it operates the user input to find suitable pairings during the matching process.

The IF classifications of the production step repository and the IF classifications of the production technology repository are all consolidated in one ontology, which is named PROTON. PROTON is a recursive acronym representing the PROduction Technology ONtology—an ontology, which was built up for the implementation of the proposed capability matching framework according to a systematic approach. This systematic approach and the reason why ontologies are used for the representation of IF classifications within this application scenario are introduced in [5].

In the capability matching framework, the production technology repository and the production step repository are represented in OWL. In [5], the advantages of using OWL for the representation of IF classifications is introduced. The Semantic Web Rule Language (SWRL) is an extension of OWL and offers capabilities to define complex rules in an OWL ontology. SWRL (pronounced swirl) provides an abstract syntax for the definition of Horn-like rules, which can be combined with an OWL knowledge base (TBox and ABox). With the help of SWRL and SQWRL queries, a dynamic construction and execution of core components for the application of Information Flow like IF channels, infomorphisms or constraints is possible.

3.4. Overall system integration

Fig. 5 shows the proposed architecture, which was used for the implementation of the production capability matching framework. The proposed system architecture provides a frontend component with the graphical user interface for the end users based on a web application. The backend provides the OWL ontology, an RDF/XML serialisation of the OWL ontology PROTON, which was developed with Protégé and is saved as a single file. Between the frontend and the backend, there is a middleware component providing a set of interfaces as connection hooks to allow the frontend getting access to the backend.

The frontend component offers the user interface where the user can provide input about the details and requirements of the production steps, which have to be conducted, and returns the results of the matching process to the user and thus is the interface to the capability matching service component. The OWL ontology PROTON is the backend component of the capability matching framework. Here, the complete knowledge base—including class hierarchy (TBox), instance relations (ABox), and rules (RBox)—is stored. The contents of this ontology can be accessed and manipulated via two application programming interfaces (APIs) provided by the middleware component.

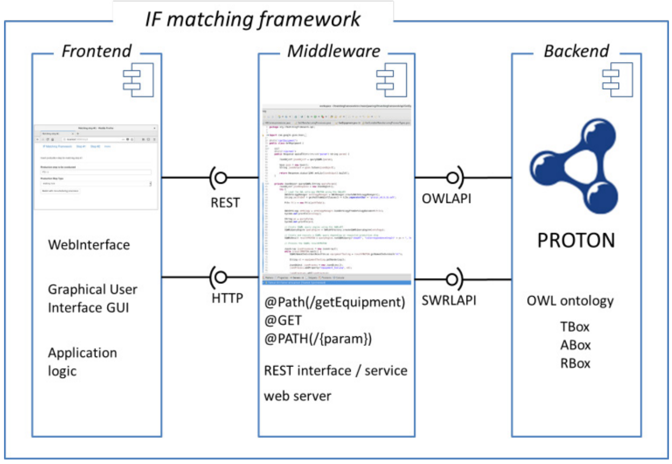

Fig. 5. Overview of overall system integration.

The middleware component connects the backend and frontend components of the IF matching framework. This component provides the web service interfaces needed by the web application to get access to the so-called resources of the middleware component in the capability matching service component. These resources receive the requests from the frontend client, including the input data from the user, process them, and return the result of the request processing back to the frontend client. This communication is done via the HTTP protocol, which offers the functionalities on which the REST web services rely on. Furthermore, the middleware component provides access to the OWL ontology via the OWLAPI and the SWRLAPI.

4. Use Case and validation of the framework

The application scenario of additive manufacturing was chosen for the evaluation of the proposed IF matching framework. The evaluation aims to see whether real-world data can be handled adequately within the IF matching framework and how the main IF components for the construction of an IF channel can be reproduced within the IF matching framework with the help of the Semantic Web technologies.

A set of different 3D printers and a set of different products were chosen for the evaluation. All 3D printers and all products are real-world entities, which are installed in a real environment or were produced with one of the selected 3D printers, respectively. The specifications of the products and the capabilities of the 3D printers were put into PROTON, the OWL ontology.

The 3D printers are put into the terminology hierarchy of PROTON by adding a class 3D printer under the equipment class. With the help of object properties, the specific 3D printer instance is related to the kinds of material, which the 3D printer is able to process, and to the printing technologies, which this specific 3D printer instance is offering. Specific characteristics of a 3D printer are given by data properties. In this example, a minimal and maximum layer thickness provided by the printer is given, and the maximum travel distances for the three axis, which the 3D printer operates. The layer thickness gives an indication of the surface quality of the printed product, and the maximum travel distances show the possible working space the 3D printer is offering. These capabilities and specifications of the 3D printers can be easily enhanced when additional requirements are needed; for this enhancement new object properties or data properties just have to be added to PROTON.

The products are put in the class *ProductionStep_PS* of PROTON. They are classified via the object property *isOfProductionStepType* under the production step type *MakingComplexStructure*. Additional specifications of the product are given by adding further object properties, e.g. material (*hasMaterial*) or known participation in initial correspondences on layer 1 (*fitsTo*) and layer2 (*isCorrespondenceStep2*). Data properties give information about requirements regarding layer thickness (*hasLayerThickness*), length (*hasLength*), width (*hasWidth*), or height (*hasHeight*) of the product. Again, necessary additional requirements and specifications can be easily added by new object or data properties.

To assign object and data properties to instances is a very straightforward and flexible way to describe properties of instances, to classify, and to relate these instances to one another or to literals. However, some kind of expert knowledge is necessary to model the contexts within an ontology adequately.

The evaluation shows that the structure of the OWL ontology PROTON is flexible enough to integrate real-world data of products and production steps or of manufacturing processes and the production capabilities of the manufacturing equipment. Furthermore, the usage of object and data properties within the OWL ontologies allows a flexible modelling of new relationships between types (constraints) and tokens (initial correspondences) and provides a means for assigning specific data values to separate instances (type-token relations).

Furthermore, the results of the evaluation show that the use of Semantic Web technologies helps to reproduce the IF components, which are needed to construct an IF channel successfully. We can use the functionality of the ontology language OWL to model IF classifications by relating instances to one another (via object properties) or to data values (via data properties) and to establish relationships of interest between types (instances or data properties) or between tokens (instances). Furthermore, with the help of the semantic web rule language SWRL, we can establish rules to describe the regularities, which govern the relationships of interest between the types or between the tokens within our application domain, and we can use these rules to infer new relationships of interest within the IF matching framework.

5. Conclusions

This paper shows that with the help of OWL an ontology, named PROTON in this work, can be built to represent the IF classifications, which are necessary to model the contexts of the distributed system in the discussed application scenario of the equipment assignment process within in a complex production system. For the time of this writing, PROTON was developed with a focus on machining manufacturing processes and additive manufacturing processes, especially 3D printing technologies. However, even with that focus, PROTON is far away of being a complete ontology for manufacturing but shows how Semantic Web Technology like OWL and SWRL in combination with Channel Theory might be used to develop a production capability matching framework. Such a framework might then be used to automate the search for suitable equipment for a specific production step and thus simplify the equipment assignment process in global or local value chains.

References

[1] Y. Koren, The Global Manufacturing Revolution – Product-Process-Business Integration and Reconfigurable Systems. Hoboken, New Jersey: John Wiley & Sons, Ltd, 2010.

[2] J. Barwise and J. Seligman, Information Flow: The Logic of Distributed Systems, Reprinted. New York: Cambridge University Press, 1997.

[3] A. Bildstein and J. Feng, “A Channel Theory based 2-step Approach to Semantic Alignment in a Complex Environment,” Int. J. Mod. Educ. Comput. Sci., vol. 9, no. 9, pp. 1–12, 2017.

[4] A. Bildstein and J. Feng, “Identifying Relationships of Interest in Complex Environments by Using Channel Theory,” in Smart Systems for Complex Problems - Abstracts from IIMA/ICITED Joint Conference 2017, 2017.

[5] A. Bildstein and J. Feng, “Using Ontologies for Representing Classifications of an Information Flow Based Matching Framework for Smart Manufacturing,” in 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), 2018, pp. 1–8.

[6] A. Bildstein, J. Feng, and T. Bauernhansl, “Information Flow-based Capability Matching Service for Smart Manufacturing,” Procedia CIRP, vol. 72, pp. 1015–1021, 2018.

[7] A. Bildstein, “Iterative Channel Theoretic Analysis for the Identification of Relationships of Interest in Complex Environments,” University of the West of Scotland, 2018.

[8] Y. Kalfoglou and M. Schorlemmer, “IF-Map : An Ontology-Mapping Method based on Information-Flow Theory,” J. Data Semant., vol. 1, no. 1, pp. 98–127, 2003.

[9] G. Yang and J. Feng, “Database Semantic Interoperability based on Information Flow Theory and Formal Concept Analysis,” Int. J. Inf. Technol. Comput. Sci., vol. 4, no. 7, pp. 33–42, Jul. 2012.

[10] M. Schorlemmer and Y. Kalfoglou, “Progressive ontology alignment for meaning coordination: An Information-Theoretic Foundation,” in Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems - AAMAS ’05, 2005, pp. 737–744.

[11] M. Schorlemmer and Y. Kalfoglou, Using Information-Flow Theory to Enable Semantic Interoperability. Amsterdam: IOS Press, 2003.