

Enhancing BPMN 2.0 Informational Perspective to Support Interoperability for Cross-Organizational Business Processes

Marija Jankovic, Miroslav Ljubcic, Nenad Anicic, and Zoran Marjanovic

Faculty of Organizational Sciences, University of Belgrade,
Jove Ilica 154, 11000 Belgrade, Serbia
{jankovic.marija, ljubcic.miroslav, anicic.nenad, marjanovic.zoran}@fon.bg.ac.rs

Abstract. Business Process Modeling Notation (BPMN) is being adopted as one of the industry standards for modeling cross-organizational business processes (CBPs). BPMN analyzes a business process as a set of interrelated activities, focusing primarily on the functional perspective of the process. However, for successful CBP modeling, an informational perspective is important. Although BPMN 2.0 supports information flow design, existing representations of data/information elements are not sufficient to support CBP modeling requirements. In this light, the paper proposes an approach for formal modeling and specification of information requirements used and generated in the CBPs. A UML View Profile is introduced to specify information requirements as views over the common reference ontology. A BPMN 2.0 extension is introduced to connect the defined views and the corresponding process activities. Ultimately, the proposed information requirements specification enables generation of the message instance and its transformation at the implementation level.

Keywords: BPMN, UML, interoperability, view, CBP.

1. Introduction

Business processes are often executed across multiple independent partners crossing organizational boundaries. Modeling of cross-organizational business processes (CBPs) focuses on defining process views describing the interaction between two or more business partners [1]. Typically, a three-level approach is applied for a comprehensive CBPs modeling [2]:

Business level: Business processes: This level specifies a computational independent view of the cooperation and the interaction expected between the partners. The CBPs modeled at this level may contain physical activities and additional information that is relevant to the perspective of the business analyst.

Business level: Technical processes: This level provides complete control flow of the CBP, modeled in a platform independent manner in order to support model reuse. However, all activities in the model should be implementable within Information and Communication Technology (ICT) system. For instance, physical activities are not included in the model.

Execution level: Executable processes: The CBP on this level is modeled in an actual language of the execution engine and contains system specific information e.g. data formats.

Currently, BPMN 2.0 could support all three levels, due to executable modeling that has been introduced as a brand new capability [3]. Executable details are fully captured in the BPMN standard attributes. Additional advantage of BPMN 2.0 is the capability to represent four important process modeling perspectives: *functional* (what activities are being performed), *behavioral* (when and how activities are performed), *organizational* (where and by whom activities are performed) and *informational* (informational entities/data produced or manipulated by a process) [4].

The problem is that in addition to specifying the CBP process flow, it is also necessary to define the detailed information requirements associated with that flow. In BPMN1.x it was not possible to define the process semantics for informational elements such as data or data flow. These elements were classified as artifacts; e.g., simple annotations of the diagram. In BPMN 2.0, data has been upgraded to a process variable, but only a small part of the information specified by semantic model is represented in the diagram; e.g., text label, data, and data store icon. Instead, BPMN designates XML Schema as its default data structure [3]. A significant disadvantage of the way that data structure is expressed in XML Schema is the lack of the clear graphical representation. Clear graphical representation should include only the constructs used to describe data semantics not including any constructs used to define syntax rules, such as choice or sequence constructs in the case of XML Schema.

The descriptions of document types - the informational and message models, and especially descriptions of their relationships - should be an integral part of the business processes' informational aspect. The BPMN 2.0 notation is not meant to allow data modeling and the breakdown of data information in specific data models [3]. Instead, it provides extension points to accommodate diverse technologies. Therefore, we propose an approach for formal modeling and specification of information requirements used and generated in the CBPs.

Our approach is based on the idea that information requirements should be specified in terms of a common, reference ontology. In the context of this work, a *reference ontology* is used as an unambiguous and formal representation of a set of business concepts and their relationships, for a particular CBP environment. That ontology provides a shared vocabulary and a conceptual model for communication between the collaborating business partners [5]. We will introduce a UML View Profile to specify information requirements as views over the common reference ontology. A BPMN 2.0 extension is introduced to enable the association of the defined views and the corresponding process activities. Finally, the proposed information requirements specification enables generation of the message instances and their transformation at the implementation level.

The remainder of the paper is structured as follows. The following section discusses the problem statement and gives essential background information. The third section discusses related work. The fourth section proposes the approach to solving the identified problems. The next section demonstrates the approach on an illustrative example. The final section concludes the paper.

2. Problem Statement and Background

This section presents a brief discussion of the problem statement that motivated our research. It also includes the essential background information on ontologies, the BPMN informational perspective, and available extension mechanisms that are relevant to this paper.

2.1. Problem Statement

The major problem addressed in this paper is the weaknesses of BPMN2.0 modeling notation. Specifically, we focus on the lack of support for modeling of informational perspective in the context of joint, cross-organizational, business processes. First, we present requirements for modeling of CBPs and discuss the importance of information flow specification on business, technical and execution level. Next, we identify problems during the modeling of information flow using BPMN 2.0 notation, and propose extensions necessary to address those problems.

CBPs Modeling Requirements. One of the first steps in designing CBPs is to identify and document modeling requirements. Different aspects and classification frameworks of CBPs requirements are proposed in [1, 2, 6]. We consider the following top-level requirements: support of a common reference ontology, information requirements formalization, and information requirements granularity.

Common reference ontology: The successful modeling of CBPs requires the inclusion of multiple domains and the interoperation with stakeholders' public and private business process models [7]. Using heterogeneous information models and domain business vocabularies raises the important research question of modeling the cross-organizational business processes and the corresponding information flows in CBPs [7]. Lippe et. al. point out that the information flow within the CBP has to be represented [2]. Moreover, they argue that global business information schema, which provides a common reference of interchanged business messages in CBPs should also be supported [2]. The usage of such a reference ontology to facilitate such interchanges is a broadly accepted approach to reconciling semantic mismatches between heterogeneous information models [5, 8, 9, 10, 11]. In a similar manner, to address aforementioned issues, we propose the specification of the information requirements in terms of a common reference ontology. Such an ontology will provide the unambiguous interpretation required by all stakeholders in the business process.

Information requirements formalization: Barnickel et.al. [12] point out that one of the common problems in designing CBPs information flows design is the lack of formalization. For example, they indicate that business process experts usually use business-oriented, high-level descriptions of information entities that are informal or semi-formal and expressed using a natural language. In that same paper, Barnickel et. al. argue that such descriptions increase the designated business-IT gap since the used terms are not explicitly linked to existing information or data models of the organization.

Information requirements granularity: Barkmeyer and Denno [13] point out that information requirements should have a fine-grained form, down to a property that is an

information unit of the entity. According to them, the information requirement arises when an agent uses a property of an entity or relationship in conducting a modeled activity. Here, the agent is an actor involved in the execution of the business process activity. The entity is a business entity defined within the information model such as database model, messaging standard or reference ontology.

The information model should provide a detailed description of the related business entities covering all information that might be used in several business processes. Hence, the business entities from the reference ontology have a general nature including a wide range of properties that are used across various business processes. Therefore, the information requirements of the activity should be defined as a subset of the business entity properties including only those properties involved in the realization of the particular activity. However, the business entities may contain other properties that are not relevant for given process activity. Consequently, a formal mechanism for specifying the information requirements as a subset of the business entity properties is needed.

BPMN 2.0 Shortcomings. BPMN 2.0 diagrams are not adequate for the discussed information requirements. BPMN 2.0 cannot address any resolution finer than the entity, although only a few modeled properties are used in many cases. Information requirements needed for activity execution are specified as Data Inputs while data that is produced is captured using Data Outputs [3]. The structure of Data Input/Output elements is not visible on the diagram; but, it can be defined using XML schema. However, XML schemas are difficult to create and understand by the business process experts who are responsible for defining the information flow on a conceptual level. A challenging issue is to specify information requirements in a suitable form for both, business analysts and IT experts.

We propose (1) a UML View Profile to solve the discussed shortcomings and (2) a BPMN 2.0 metamodel extension to include the information requirements specification based on that Profile. In designing both, we had to overcome three important problems: how to specify information requirements for the activity, how to represent needed associations between an activity and the requirements, and, how to exchange messages/documents during the activity realization. In addition to providing solutions to these problems, our approach has an additional advantage. It enhances the possibility to implement a generic transformation that supports specified information requirements automatically from messages exchanged at runtime during the process execution.

2.2. Background

Ontologies. In this section, we describe the use of ontologies in the context of our work. For a detailed introduction and a valuable overview of the ontologies see work by [14 – 17]. Gruber [18] has defined an ontology as “formal, explicit specification of a shared conceptualization”. In this paper, we use a common reference ontology that specifies, formalizes, and explicates the domain business concepts and their relationships involved in a particular CBP scenario [8]. The formal specification of business domain concepts,

given in the reference ontology, is important in order to provide a basis for unambiguous interpretation of data-exchange artifacts in CBPs [11].

As stated in [19, 20, 21], various languages can be used for the construction of the reference ontology. In our work, we use reference ontologies to support systems interoperability where we use UML to represent ontologies. The reasons for choosing UML for ontology representation are:

- UML Class diagram supports visual modeling of important ontology elements (e.g. class/sub-class hierarchies, relationships between classes, class attributes). This facilitates easier understanding by business analysts and end-users.
- UML is an open standard and has a standard mechanism for defining extensions, e.g. Profiles.
- OCL is a powerful mechanism for defining additional constraints; e.g. attribute values or possible instances of the relationships.
- UML is widely accepted in industry and has a large user community. For example, UML is most frequently used for visual representation of integration standards that are based on XML Schema, e.g. OAGIS, RosettaNet, Universal Business Language (UBL).
- Core UML concepts map appropriately to OWL concepts, as it is defined in [22].

Detailed specification of the UML reference ontology model used in our example, along with corresponding formal OWL representation can be found in [23].

A variety of different research project have been applying UML for ontology representation either directly or as graphical front-end for ontology languages that don't have visualization capabilities [21,24]. In their work, Baclawski et. al. [21] implemented tools for ontology development based on UML. They indicate that UML is not convenient for visualization of complex ontologies only, but for managing ontology development process as well. Cranfield and Purvis have investigated the use of UML class diagrams for representing ontologies [24,25].

BPMN Informational Perspective. In this section, we provide an overview of BPMN Informational Perspective that is relevant to the problem statement. Information flow plays a crucial role in CBP modeling, although BPMN focusses on the control flow aspects [26, 27]. The flow of informational entities (e.g., data, artifacts, products) between process elements is decoupled from the Sequence flow to allow modeling flexibility [26][27].

A primary construct for modeling all kinds of informational entities regardless of their physical nature (e.g. paper or electronic documents) in BPMN is a Data Object [3]. In BPMN 2.0, Data Objects are upgraded to first-class, semantic elements and defined as additional Data Categories aside from flow objects, connecting objects, swim-lanes, and artifacts. This is a big change, having in mind that in BPMN1.2 Data Objects were considered artifacts, simple annotation without any semantics [28]. Bruce Silver in [28] points out that Data Objects are programming constructs, a temporary data stored in the process instance. Data object elements are visually presented on a Process diagram (see Fig.1), and can be referenced by DataObjectReference that specifies different states of the same DataObject (e.g., <DataObject Name>[DataObjectReferenceState]). The structure of the Data Object is not visible on the diagram, but it can be defined by its associated itemDefinition element that specifies an XML schema.

For the purpose of representing persistent data (e.g., database records) BPMN2.0 introduces a new concept – Data Store. Additional elements of Data Category are Data Inputs, Data Outputs and Properties [3]. Collections of Data Objects, Data Inputs and Data Outputs are represented by Data Object Collection, InputSet and OutputSet, respectively. Property elements have no visual representation in the diagram, and they are relevant for process execution. Graphical representations of the Data Category elements are represented in the Fig.1.

Name	Data Object	Data Object Collection	Data Input	Data Input Collection	Data Output	Data Output Collection	Data Store
BPMN Shape	 Label	 Label	 Label	 Label	 Label	 Label	 Label

Fig. 1. Data Category elements (adapted from [3])

Besides simple, non-directional Association that is still used to link text annotations in the diagram, in BPMN2.0 Data Association is introduced [28]. Elements of Data Category are connected to other model elements (e.g., activities or events) through directional Data Association. This association represents a mapping between a Data Object and Data Input or Data Output [28]. When source and target of data flow are unambiguous, non-directional data associations to a sequence flow are allowed [28]. Data Objects are no longer used to represent the information content of a message between different pools or external entity; a new message symbol, an envelope icon, is introduced in BPMN 2.0.

BPMN Extension Mechanism. The BPMN is designed to be extensible by a standard extension mechanism that can be used by modelers to define new concepts with needed semantics. The BPMN extension mechanism consists of a set of extension elements that allow the attachment of additional elements and attributes to standard and existing BPMN elements [3]. These extension elements are: ExtensionDefinition, ExtensionAttributeDefinition, ExtensionAttributeValue and Extension [3]. Extension element is used to bind a BPMN model with an extension whose structure is defined using ExtensionDefinition element. ExtensionDefinition element groups additional attributes used to extend the BPMN model by attaching them to any BPMN element. The definition of each attribute includes the name and type of the attribute; and, it is given by corresponding ExtensionAttributeDefinition element. Within an extended BPMN element, ExtensionAttributeValue element is used to assign a value to a particular extension's attribute that has been defined previously within ExtensionDefinition using ExtensionAttributeDefinition element.

3. Related Work

The approach proposed by Barkmayer and Denno [13] relates to ours since they introduce a common, reference-ontology-based specification of needed information requirements in a joint business process. The term joint process denotes a shared viewpoint of the joint actions between collaborating partners. Their methodology includes three major components: a reference ontology for the business entities, a formal specification of the joint process, and a binding between process elements and business entities. They introduce concepts for the improvement of information flows through introducing message structures into the diagram itself. They offered an interesting conceptual solution that motivated our research. As the authors themselves mention “while the proposed concepts are simple, the actual representations of user and provider flows may be complex”. They do not propose any notation for information requirements specification. The authors do not give a formal definition of the view over the reference ontology, whereby their concept of the view represents a simple filter over the predefined entities of the reference ontology without the possibility to define more complex rules of execution (such as model traversing or calculations), which is possible in our approach.

Barnickel et. al. demonstrated a mediated, business-process-modeling approach for incorporating semantic bridges to implement information flow design [7]. They provide a complete end-to-end solution, from specification to implementation. Their approach is based on semantic bridges, which are applied to the domain ontology-based information entities in order to overcome semantic heterogeneities. For the purpose of better understanding and visualization, they propose a BPMN extension of Data Object category using a semantic sub-graphs. The paper highlights the ontological representation of information flows by the application of RDF and OWL. It does not offer a description of the implementation of information requirements, as parts of the entities from the reference ontology, they are rather used unchanged and complete. Consequently, there is a need for the semantic reconciliation of different reference ontologies used by different parties. The work in this article differs from their approach as we propose the use of common reference ontology, assuming that each party has a formalized ontological model that comprises, or is mapped to elements of a publicly available common ontological library.

Another interesting extension of BPMN 2.0 using semantic ontologies is presented in [29]. Gao et. al. state that BPMN 2.0 should be described in more details with respect to functional, data, organizational and control ARIS views. For our work is relevant Data View BPMN 2.0 extension using Linked Data Principle. They propose that BPMN ItemAwareElements should be annotated with concepts from domain-specific ontologies that are specified in RDFS or OWL using StructuredWebResource (SWR) framework. The authors argue that the proposed approach can make improvements not only in the execution phase, but also during other phases of the BPM lifecycle. However, in our opinion, graphical UML representation is more convenient for business process specification at conceptual level. The BPMN 2.0 extension proposed in our solution is more general since their solution is aligned with ARIS specific views.

The idea of Semantic Business Process Management (SBPM) is introduced in [30]. Hepp et.al. propose to combine Semantic Web services frameworks, ontology infrastructure, and BPM to create one consolidated technology. Representational requirements of SBPM are discussed in [31]. SBPM approaches are focused mainly on

ontology-based process flow annotation. Our work is, however, concerned with ontology-based information flow specification.

Various approaches highlight the importance of data flow modeling in business process languages. Deutch and Milo notice that business process flow affects data and vice versa [32]. According to [32], an important aspect of business process modeling is capturing the data manipulation and transformations performed by the process. On the other hand, approaches in [33, 34, 35] suggest to include process perspective into data management practice. Magnani and Montesi [36], identify and address shortcomings of data modeling using BPMN 1.2 modeling notation. They define BPMN extension called BPDMN (Business Process and Data Modeling Notation) in order to enhance visual data capabilities. While extending BPMN 1.2, their extension implies direct changes in the BPMN metamodel. In our work, the extension of BPMN is given through the use of a currently default and formally defined extension mechanism that is part of the BPMN 2.0 standard.

Unlike any mentioned approach, which makes use of reference ontology documents the way they are, we offer the possibility to define a view over reference ontology documents without the need to use complete document structures. All mentioned works describe the process on the technical level without specifying the implementation. Our work proposes a formalized definition of the view, associated with the process through the BPMN extension, and a described algorithm for obtaining a view instance on the implementation level, at the time of the process execution. Also, all mentioned works are oriented towards the semantic reconciliation of ontologies without defining the way in which this will affect the implementation itself. Our approach does not solve the problem of semantic reconciliation, it rather focuses on the method to enable executable specifications, in the sense to define how the specified information flows are realized on the implementation level.

4. Details of the Approach

The approach is based on the idea that the reference ontology is a shared definition of the types, properties and interrelationships of the business entities that are used to construct the messages exchanged between the collaborating business partners. We propose the use of information requirements defined in terms of the reference ontology as the basis for the sound design of information flows in CBPs. The idea is that during modeling of business processes, the reference ontology will enable unambiguous interpretation of specified information requirements. It does this by supporting common procedures for deriving the information requirements from interoperable, ontology-based, message exchange.

To specify information requirements as a subset of the business entity properties (attributes and relationships), we propose UML View Profile: a UML extension defined using the UML profile mechanism. This approach is similar to the concept of a database view. Business entities from a reference ontology correspond to tables of a database schema. The model defined using UML View Profile corresponds to the database view (in the rest of the paper this model is referenced as view model). The view model contains the definition of the information requirements of the activity including the mapping rules to the business entity properties. The mapping rules are used to derive

defined information requirements from the reference ontology model at runtime. They are defined using Object Constraint Language (OCL) [37]. To include our view model into BPMN model, we propose a BPMN extension based on the BPMN 2.0 extension mechanism.

The procedure for specifying the information requirements comprises the following steps.

- a. *Reference Ontology Development.* The reference ontology for a specific business domain is created, or an existing reference ontology is selected. Ontology specification at the conceptual level is presented using the UML class diagram.
- b. *Business Process Model Development.* The BPMN model of cross-organizational business process is created.
- c. *Annotation and Association of Information Requirements.* The BPMN model is annotated and enriched using concepts defined in the BPMN extension.
- d. *View Model Specification.* Detailed specification of the information requirements of the process activities is created by defining view models using the UML View Profile.

The UML View Profile, BPMN extension and model transformation process are described in the following sections.

4.1. UML View Profile

This section lays out a UML profile proposal, called the UML View profile, as a formal mechanism for identifying the information requirements of the process activities. Using the proposed UML View Profile, information requirements are defined as a subgroup of properties of the appropriate business entities from the reference ontology model. The defined stereotypes of the UML View profile are described as follows.

Stereotype: ViewPackage

Base Class: Package

Description: Represents a package that contains view model definition.

Constraints: The package members must be one of the stereotypes: ViewClass, ViewAssociation or basedOn.

Tagged Values: expressionLanguage - the language used to define the expressions and derivation rules within the package members.

Stereotype: ViewClass

Base Class: Class

Description: Represents a class defined within the view model definition, based on the reference ontology class. Contains ViewProperty properties that define subgroup of properties of corresponding reference ontology class.

Constraints: It must contain at least one property with the ViewProperty stereotype. It must be based on the reference ontology class (represented by the dependency relationship with the basedOn stereotype).

Tagged Values: isEntryPoint - signifies whether ViewClass is the entry point of the view, i.e. the initial point for the transformation execution.

Stereotype: ViewProperty*Base Class:* Property*Description:* Represents a property defined within ViewClass whose value is determined by an expression defined over the properties of the reference ontology class.*Constraints:* It must have a defined value for the tagged value expression.*Tagged Values:* expression - the expression that defines the mapping of the ViewProperty to one or more properties of the reference ontology class (derivation rule).**Stereotype: ViewAssociation***Base Class:* Association*Description:* Represents an association that connects two ViewClasses.*Constraints:* The association ends owner must be ViewClass or ViewAssociation itself (depending on the navigability of the association end). It must have a defined value for the tagged value refinementExpression.*Tagged Values:* refinementExpression - the expression that defines the condition for additional filtration of the set of ViewClass objects at the ViewAssociation end.**Stereotype: basedOn***Base Class:* Dependency*Description:* Dependency relationship of this stereotype defines the dependency of ViewClass from the reference ontology class, i.e. it defines the reference ontology class whose properties are subsetted by ViewProperties of the ViewClass.*Constraints:* The basedOn dependency source must be ViewClass while the target must be Class.**Stereotype: Key***Base Class:* Property*Description:* Represents the ViewClass identifier.*Constraints:* It must be applied to ViewProperty.**4.2. BPMN Extension**

We used the BPMN 2.0 extension mechanism to define the BPMN metamodel extension depicted in Fig. 2. Proposed extension enables inclusion of the ontology document model definition (i.e., part of the reference ontology model corresponding to the message exchanged) and view model definition into the BPMN process. *ExtensionDefinition* and *ExtensionAttributeDefinition* elements are used to define the structure of the proposed extension. In Fig. 2, they are represented as stereotypes, using the same name as the related elements. Original BPMN metamodel elements are marked with the stereotype *BPMN*.

The BPMN metamodel elements relevant for the association of the ontology document/view model definitions are *DataObject*, *DataInput* and *DataOutput*. They are subclasses of *ItemAwareElement*, selected as the BPMN metamodel concept being extended. The *ItemAwareElement* is extended either by the *OntologyElement* or the *ViewElement* extension definition. In Fig. 2 this is illustrated by the {xor} constraint. An *ItemAwareElement* (e.g. *DataObject*) can practically contain either the reference ontology document (represented by the *OntologyElement*) or the view model defined

over the reference ontology document (represented by the *ViewElement*). The original BPMN elements used in a general case to specify the data structures contained by an *ItemAwareElement* are *ItemDefinition* and *Import* elements. Likewise, within our proposed extension, these BPMN elements are utilized to import data structures of the reference ontology document model or the view model. If unspecified, each data structure is by default serialized in XML Schema format.

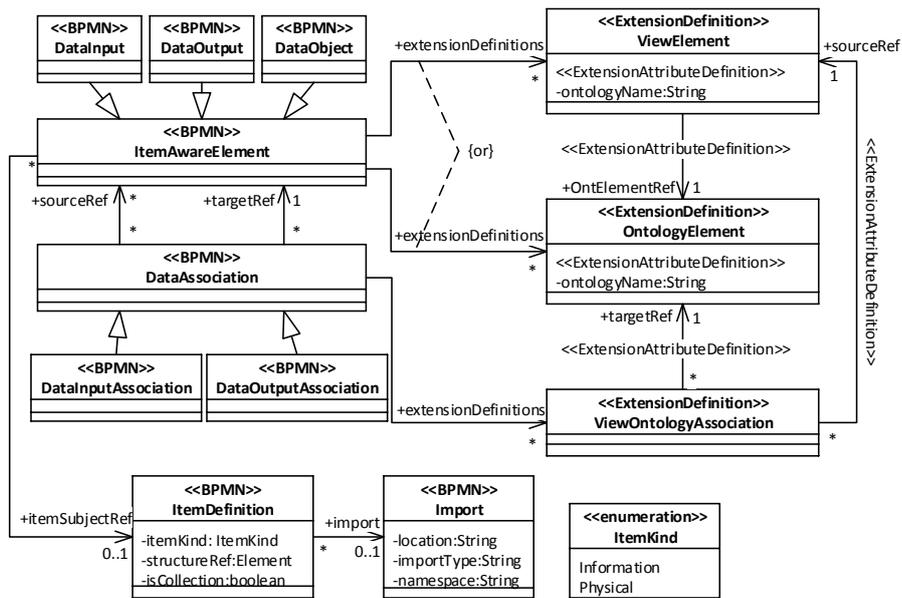


Fig. 2. BPMN Extension

ViewElement has the *OntElementRef* property referencing the *OntologyElement* on which it depends. For example, it defines the reference ontology document model to which the view model is to be applied at runtime to derive the information requirements from exchanged document/message. For the visual representation of this dependency, the *ViewOntologyAssociation* extension is defined, extending the original BPMN element *DataAssociation*. This extension limits *DataAssociation* by defining the *ViewElement* (*sourceRef* property) as an association source and *OntologyElement* (*targetRef* property) as the association target. When the *ViewOntologyAssociation* extension is used within a *DataAssociation* element, *sourceRef* and *targetRef* properties of the *DataAssociation*, if included, must have the same values as the respective properties of the *ViewOntologyAssociation* extension. Fig. 3 shows an illustrative example using the concepts defined in the BPMN extension (the extension concepts are marked with the appropriate stereotypes).

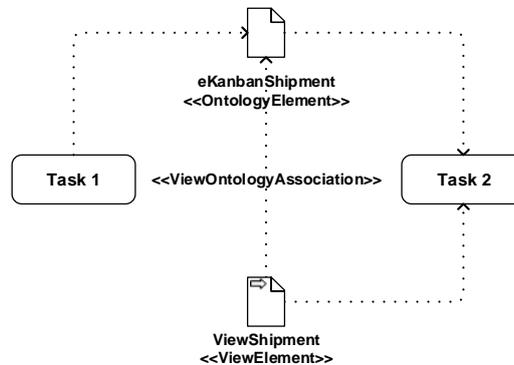


Fig. 3. Sample BPMN process with extension elements

4.3. Model Transformation

As already stated, the UML View Profile is used to specify semantic mapping rules between the view model and the reference ontology model (shown in Fig. 4 at M1 meta-layer of the four-layered metamodel architecture). These rules are contained within the view model definition. Based on them we can generate the transformation rules of the reference ontology model instance (i.e. message exchanged within the business process) to the instance of the view model. These instances are shown at M0 meta-layer in Fig. 4. There are several ways in which the transformation of the models can be defined. Query/View/Transformation (QVT) specification is one of the standard ways provided by Object Management Group (OMG) [39]. XML transformation languages can be used as well (e.g., XQUERY, Extensible Stylesheet Language Transformations (XSLT)). Our approach uses QVT; note, the XML-based transformations can be generated from them if necessary.

Transformation rules for the instances of the reference ontology model can be generated automatically based on the view model. This is possible because the transformation definition itself can be presented as a model. Specifically, result of a QVT transformation can be QVT transformation itself. To execute the result of a QVT transformation as a new QVT transformation, QVT specification defines *'asTransformation'* operation. This is used to invoke on-the-fly transformations definitions created dynamically. This QVT feature is used in our approach as illustrated in Fig.4 with the *Transformation Generation* node. In this step, the QVT transformation definitions are generated dynamically based on the rules defined within the view model. The generation algorithm relies on the fact that both source and target models are instances of the same metamodel; i.e. the UML metamodel. Since the model entry point is given for each view model definition using *entryPoint* tagged value of the *ViewClass* stereotype, the algorithm relies on the definition of the entry point *ViewClass* for further processing. The rules for the generation of the QVT transformations from the OCL expressions are applied primarily to the *ViewProperties* and *ViewAssociations* of the entry point *ViewClass*. Thereafter, they are successively applied to other *ViewClasses*

and their *ViewProperties* and *ViewAssociations*. In the next step, the generated QVT transformations (represented in Fig. 4 with the *Data Transformation* node) are executed.

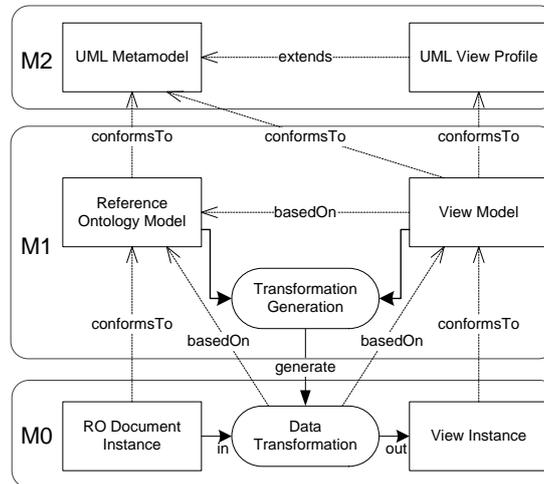


Fig. 4. Data Model Transformation

The transformation of data for the validation of the proposed approach is done within the Eclipse Modeling Framework (EMF) with the application of QVT implementation.

5. Example

Let us present our approach by an example. The example models a generic eKanban scenario [39].

5.1. Reference Ontology Development

In the first step, we create the reference ontology. The ontology definition represents a key part of the architecture and contains information about business concepts and the connections between them. It also contains the contextual description, which describes in what way the information entities (whether basic or aggregating) can be used in a specific business scenario. Alternatively, an existing reference ontology can be chosen instead of creating a new one. At this step, the eKanban reference ontology [23] was selected to illustrate our approach.

5.2. Business Process Model Development

The second step creates a formal specification for the collaborative business process identifying all activities and shared information exchanged within the process. An example of a collaborative shipping business process is given in Fig. 5, focusing solely on *Supplier* participant's activities.

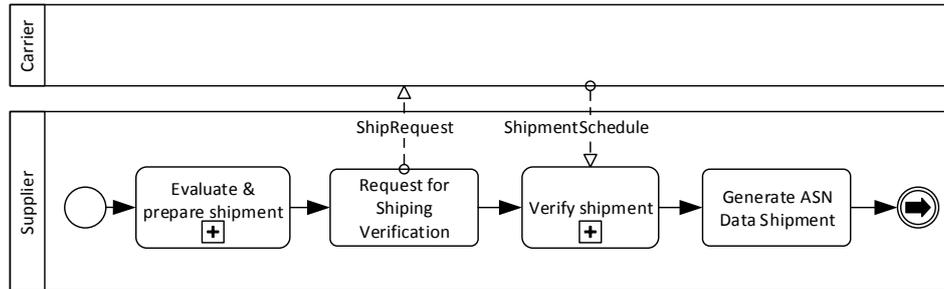


Fig. 5. Sample Shipping Process

5.3. Annotation and Association of Information Requirements

In this step, we associate the information requirements, defined as a view model, with the appropriate BPMN elements by annotating them in accordance with the proposed BPMN extension.

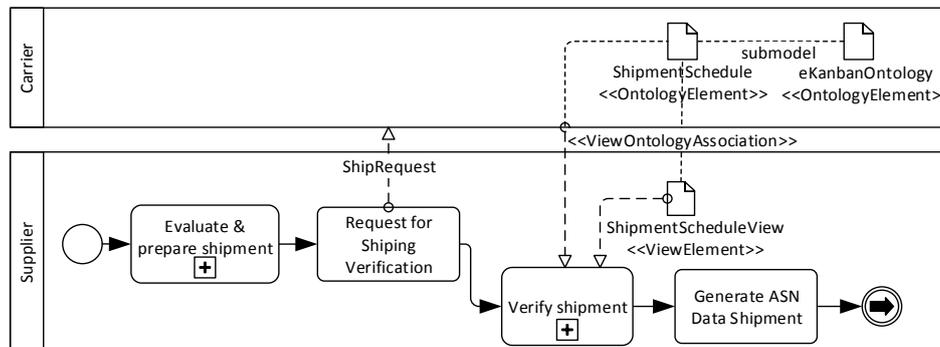


Fig. 6. Annotated Shipping Process

Fig. 6 illustrates annotation and association of information requirements for the Verify Shipment activity. Different types of data objects associated with the activity are annotated using appropriate stereotypes from the proposed BPMN extension. The *OntologyElement* stereotype is applied to *DataObject* representing the *ShipmentSchedule*

document from eKanban reference ontology exchanged between *Carrier* and *Supplier*. The *ViewElement* stereotype is applied to *DataObject* representing view model, named *ShipmentScheduleView*, which defines the activity information requirements as a view over *ShipmentSchedule* document model. Their mutual inter-dependency is shown explicitly by the dependency relationship annotated with *ViewOntologyAssociation* stereotype.

5.4. View Model Specification

In the final step, we create the information requirements specification using UML View Profile. Fig. 7 depicts the *ShipmentScheduleView* definition.

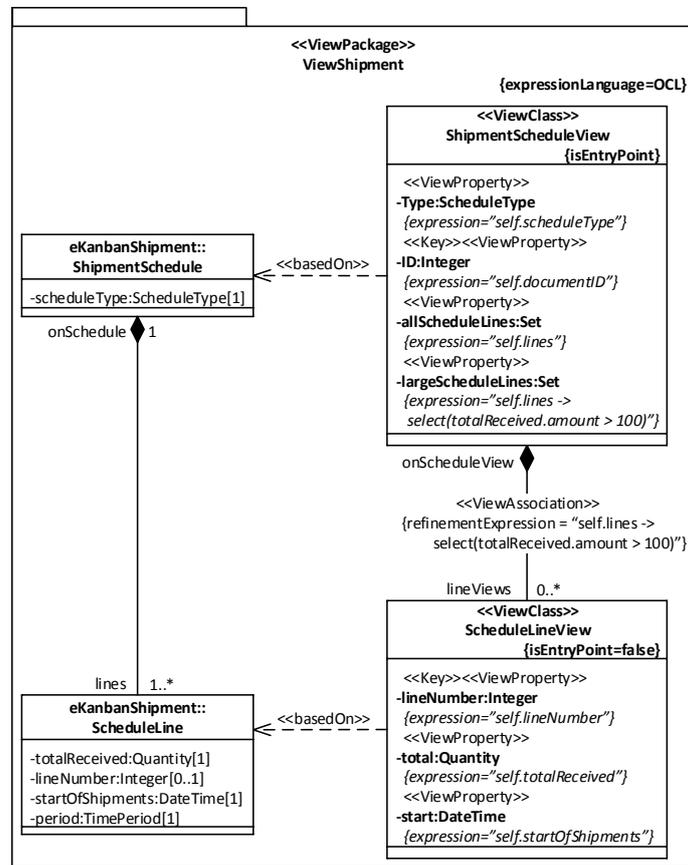


Fig. 7. ShipmentScheduleView Definition

The view model is defined over the *ShipmentSchedule* document model of the eKanban reference ontology. For the purpose of clarity, only the elements of the *ShipmentSchedule* document model relevant for the definition of the view model are

shown (*ShipmentSchedule* and *ScheduleLine* classes from the *eKanbanShipment* package). The view model is defined within the *ViewShipment* package with the *ShipmentScheduleView* ViewClass as the entry point of the transformation. The *ShipmentScheduleView* is mapped to the *ShipmentSchedule* class of the eKanban ontology. This is defined using *basedOn* dependency. ViewProperties of *ShipmentScheduleView* are mapped to the properties of the *ShipmentSchedule* class. These mappings are defined using OCL expressions given within the expression tagged value of each ViewProperty. The OCL mapping expressions are defined in the form suitable for direct execution against the appropriate reference ontology concept. The *self* keyword within the OCL expressions marks the reference ontology class to which the ViewClass, owner of the ViewProperty, is mapped. For example, ViewProperty *Type* is defined by the "*self.scheduleType*" expression which is executed against the *ShipmentSchedule* instance and results in the value of its *scheduleType* property.

A ViewClass can also define its ViewProperties over the related classes of the mapped reference ontology class and their properties. In line with the aforementioned, the *ShipmentScheduleView* contains *allScheduleLines* ViewProperty representing the Set of all *ScheduleLine* objects of *ShipmentSchedule*. Similarly, it contains *largeScheduleLines* ViewProperty representing the Set of *ScheduleLine* objects with amount greater than 100. In both cases, the Set will contain "full" *ScheduleLine* objects; i.e. objects having all properties of the *ScheduleLine* class. If it is necessary to use only the subset of properties of *ScheduleLine* class, a new ViewClass would have to be defined (*ScheduleLineView* in Fig. 7). Additionally, a new ViewAssociation with appropriate refinement expression have to be defined as well (ViewAssociation between *ShipmentScheduleView* and *ScheduleLineView* in Fig. 7). It should be noted that now *ShipmentScheduleView* has ViewProperty *largeScheduleLines* and ViewAssociation, both defined using the same expression (*self.lines->select (totalReceived.amount > 100)*), but resulting in sets of different objects. ViewProperty *largeScheduleLines* will contain the Set of *ScheduleLine* objects while the Set obtained through ViewAssociation will contain *ScheduleLineView* objects (that contains the subset of *ScheduleLine* properties relevant for the view definition).

Fig. 8 depicts this by an example of the *ShipmentSchedule* document instance (Fig. 8 a) and the appropriate *ShipmentScheduleView* instance obtained as the result of the transformation process (Fig. 8 b).

In summary, in the first step, eKanban reference ontology is used as a common specification of business domain concepts and their relationships. In the second step, a business process model is developed (see Fig.5). Next, that business process model is enhanced using proposed BPMN extension elements. For example, Fig. 6 illustrates information requirements for *Verify Shipment* activity, represented by *ShipmentScheduleView* element and its association with eKanban *ShipmentSchedule* document. Finally, detailed definition of information requirements for *Verify Shipment* activity is created using UML View Profile, which results in view model shown in Fig. 7. Defined view model contains mapping rules of specified information requirements to eKanban business entities from *ShipmentSchedule* document. This provides support for automatic document model instance transformation at runtime.

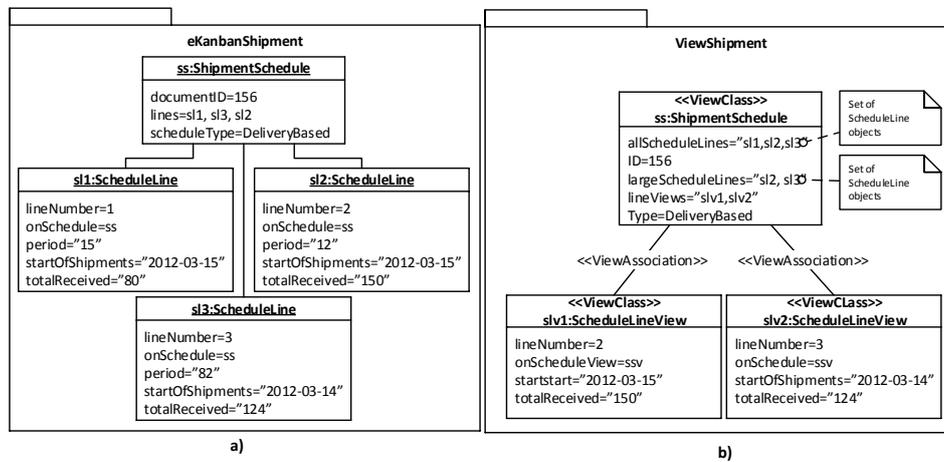


Fig. 8. Ontology document instance and view instance examples

6. Conclusion

This paper addresses two major topics. First, it presents an approach to formalize the informational aspect of cross-organizational business processes. Second, it promotes the possibility of automating the implementation of that formalization.

The key contributions of this paper are:

- the definition of the UML View Profile as a mechanism to specify the information requirements in terms of the reference ontology
- the definition of the BPMN extension to allow association of the information requirements to the BPMN model activities
- the definition of the role for and requirements for QVT transformations enabling the automation of the model instance transformation for the purpose of their easier implementation.

We believe that our proposed approach is sufficiently general and flexible to describe cross-organizational business processes that include a detailed specification of the informational content.

In the future, we plan to design tools to support the proposed manner of describing processes. Such tools will (1) facilitate the application of the steps of that approach and (2) make the application of the presented transformations possible.

References

1. D.A2.1: Cross-Organisational Business Process requirements and the State of the Art in Research, Technology and Standards Version 2. ATHENA Project No. (507849). (2005)

2. Lippe, S., Greiner, U., Barros, A.: A survey on state of the art to facilitate modelling of cross-organisational business processes. In Proceedings of the 2nd GI-Workshop XML4BPM 2005. Gesellschaft für Informatik Bonn, Germany, 7-22. (2005)
3. Business Process Model and Notation (BPMN) Version 2.0. OMG (2011). [Online]. Available: <http://www.omg.org/spec/BPMN/2.0/PDF/> (current April 2015)
4. Curtis, B., Kellner, M., Over, J.: Process modeling. *Communication of the ACM*, Vol. 35, No.9, 75-90. (1992)
5. Vujasinovic, M., Barkmeyer, E., Ivezic, N., Marjanovic, Z.: Interoperable Supply-Chain Applications: Message Metamodel-based Semantic Reconciliation of B2B Messages. *International Journal of Cooperative Information Systems*, World Scientific Publishing Co Pte Ltd, Vol. 19, No. 1-2, 31-69. (2010)
6. Dori, D., Beimel, D., Toch, E.: OPCATeam—collaborative business process modeling with OPM. *Business Process Management, Lecture Notes in Computer Science*, Springer Berlin Heidelberg, Vol. 3080, 66–81. (2004)
7. Barnickel, N., Bottcher, J., Paschke, A.: Incorporating semantic bridges into information flow of cross-organizational business process models. In Proceedings of the 6th International Conference on Semantic Systems (I-SEMANTICS '10), ACM, New York, NY, USA, Article 17, 1-9. (2010)
8. Vujasinovic, M., Ivezic, N., Kulvatunyou, B., Barkmeyer, E., Missikoff, M., Taglino, F., Marjanovic, Z., Miletic, I.: Semantic mediation for standard-based B2B interoperability. *Internet Computing, IEEE*, Vol.14, No.1, 52-63. (2010)
9. Yarimagan, Y., Dogac, A.: A semantic-based solution for UBL schema interoperability. *Internet Computing, IEEE*, Vol.13, No.3, 64-71. (2009)
10. Anicic, N., Ivezic, N., Jones, A.: An architecture for semantic enterprise application integration standards. *Interoperability of Enterprise Software and Applications*, Springer-Verlag, London, 25-34. (2006)
11. Vujasinovic, M., Ivezic, N., Barkmeyer, E., Marjanovic, Z.: Semantic B2B-integration using an Ontological Message Metamodel. *Concurrent Engineering, IEEE*, Vol.18, 219-232. (2010)
12. Barnickel, N., Bottcher, J., Paschke, A.: Semantic Mediation of Information flow in Cross-Organizational Business Process Modeling. In Proceedings of the 5th International Workshop on Semantic Business Process Management collocated with the Extended Semantic Web Conference, ACM, Heraklion, Greece, 21-28. (2010)
13. Barkmeyer, E. J., Denno, P.: On capturing information requirements in process specifications. *Enterprise Interoperability II*, Springer, London, 365-376. (2007)
14. Bunge, M.: *Treatise on basic philosophy. III. Ontology: The future of the world*. Reidel, Dordrecht, Holland.(1977)
15. Bunge, M.: *Treatise on basic philosophy. III. Ontology: A word of systems*. Reidel, Dordrecht, Holland. (1979)
16. Sowa, J.: *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks/Cole Publishing Co., Pacific Grove, CA. (2000)
17. Guarino, N.: Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human-Computer Studies*, Vol. 43, No. 5-6, 625-640. (1995)
18. Gruber, T. R.: Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, Vol. 43, No. 5–6, 907-928. (1995)
19. Gomez-Perez A., Fernandez-Lopez M., Corcho, O.: *Ontological Engineering with Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web*. Series: Advanced Information and Knowledge Processing. Springer-Verlag, New York, Inc., Secaucus, USA. (2007)
20. Borgo, S.: How formal ontology can help civil engineers. In *Ontologies for Urban Development, Studies in Computational Intelligence*, Vol. 61, Springer Berlin Heidelberg. (2007)

21. Baclawski K., Mieczyslaw K., Kokar K., Kogut P., Lewis H., Smith J., Letkowski J., Emery, P.: Extending the Unified Modeling Language for ontology development. *Software and Systems Modeling*, Springer-Verlag, Vol. 1, No. 2, 142-156. (2002)
22. Ontology Definition Metamodel Version 1.1. OMG (2014). [Online]. Available: <http://www.omg.org/spec/ODM/1.1/PDF/> (current April 2015)
23. Barkmeyer E. J., Kulvatunyou, B.: An Ontology for the e-Kanban Business Process. NIST Internal Report 7404, National Institute of Standards and Technology (NIST). (2007). [Online]. Available: http://www.nist.gov/customcf/get_pdf.cfm?pub_id=822708 (current April 2015)
24. Kogut, P., Cranfield, S., Hart, L., Dutra, A., Baclawski, K., Kokar, M., Smith, J.: UML for ontology development. *The Knowledge Engineering Review*, Cambridge University Press, Vol. 17, No.01, 61-64. (2002)
25. Cranefield, S., Purvis, M.: UML as an Ontology Modeling Language. In *Proceedings of the Workshop on Intelligent Information Integration, 16th Int. Joint Conference on AI (IJCAI-99)*, Germany, 46-53. (1999)
26. White, S., Miers, D.: *BPMN modeling and reference guide*. Future Strategies Inc., Lighthouse Point, Fla. (2008).
27. Freund, J., Rucker, B.: *Real-Life BPMN: Using BPMN 2.0 to Analyze, Improve and Automate Processes in Your Company*. Camunda, Lexington, KY. (2012)
28. Silver, B.: *BPMN Method and Style, 2nd Edition, with BPMN Implementer's Guide*. Cody-Cassidy Press, Aptos (2011)
29. Gao, F., Derguech, W., Zaremba, M.: Extending BPMN 2.0 to Enable Links between Process Models and ARIS Views Modeled with Linked Data. *Business Information Systems Workshops, BIS 2011 International Workshops and BPSC International Conference*, Poznan, Poland, 41-52. (2011)
30. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic business process management: a vision towards using semantic Web services for business process management. *IEEE International Conference on e-Business Engineering (ICEBE'05)*, 535-540. (2005)
31. Hepp, M., Roman, D.: An Ontology Framework for Semantic Business Process Management. In *Proceedings of Wirtschaftsinformatik, Karlsruhe*, 423-440. (2007)
32. Deutsch D., Milo, T.: *Business Processes: A Database Perspective. Synthesis Lectures on Data Management*, Morgan & Claypool Publishers, San Rafael, Calif., 1-103. (2012)
33. Beerl, C., Eyal, A., Milo, T., Pilberg, A.: Monitoring business processes with queries. In *Proceedings of the 33rd Int. Conf. on Very Large Data Bases*, Vienna, Austria, 603-614. (2007)
34. Deutsch, A., Marcus, M., Sui, L., Vianu, V., Zhou, D.: A verifier for interactive, data-driven web applications. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data - SIGMOD '05*, Baltimore, MD, USA, 539-550.(2005).
35. Fritz, C., Hull, R., Su, J.: Automatic construction of simple artifact-based business processes. In *Proceedings of the 12th International Conference on Database Theory - ICDT '09*, Saint-Petersburg, Russia, 225-238. (2009).
36. Magnani, M., Montesi, D.: *BPDMN: A Conservative Extension of BPMN with Enhanced Data Representation Capabilities*. arXiv:0907.1978v1[cs.SE] (2009)
37. Object Constraint Language (OCL) Version 2.3.1. OMG (2012). [Online]. Available: <http://www.omg.org/spec/OCL/2.3.1/PDF/> (current April 2015)
38. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Version 1.1. OMG (2011). [Online]. Available: <http://www.omg.org/spec/QVT/1.1> (current April 2015)
39. IBP-2 Inventory Visibility & Interoperability Electronic Kanban Business Process Version 1. Automotive Industry Action Group (AIAG). (2006)

Marija Jankovic is a teaching assistant in the Department of Information System and Technologies at the Faculty of Organizational Sciences, University of Belgrade. She received the M.Sc. degree in Information Systems from University of Belgrade, Serbia in 2007. Her research interests include business process modeling, information systems development methodologies, enterprise application interoperability, contemporary software architecture and database systems.

Miroslav Ljubicic is a teaching assistant in the Department of Information System and Technologies at the Faculty of Organizational Sciences, University of Belgrade. He received the M.Sc. degree in Information Systems from University of Belgrade, Serbia in 2011. His research interests include information systems development methodologies, databases, model driven development, service-oriented architecture, semantic technologies, and enterprise application interoperability.

Nenad Anicic is an associate professor in the Department of Information System and Technologies at the Faculty of Organizational Sciences, University of Belgrade. He received the M.Sc. and Ph.D. degrees in Information Systems from University of Belgrade, Serbia in 2001 and 2006, respectively. His research interests include information systems development methodologies, model driven development, semantic technologies, and interoperable application systems.

Zoran Marjanovic is a full professor at Faculty of Organizational Sciences, University of Belgrade, and a founder and president of the Breza Software Engineering company. His research interests are information systems development methodologies, databases, and semantic enterprise application interoperability. Professor Marjanovic is a lead on several on-going projects with government and commercial entities that address design, deployment, and testing of enterprise resource planning and other business systems. He received his MS and PhD degrees in Information Systems from University of Belgrade. He is a member of ACM and IEEE.

Received: November 12, 2014; Accepted: April 14, 2015.