

# Repurposing Zachman Framework Principles for "Enterprise Model"-Driven Engineering

Alisa Harkai, Mihai Cinpoeru and Robert Andrei Buchmann

*Business Informatics Research Center, Babeş-Bolyai University, str. T. Mihali 58-60, Cluj Napoca, Romania*

**Keywords:** Enterprise Modelling, Enterprise Knowledge Engineering, Zachman Framework, Resource Description Framework.

**Abstract:** The paper proposes an agile modelling tool which implements a domain-specific modelling method. As a motivational starting point for the development of this modelling tool, we employ the Zachman Framework - an ontology which conceptualises an enterprise across a variety of abstractions and facets. We conducted our work with respect to the Zachman Framework in order to cover several of these facets and to suggest the possibility of further employing Agile Modelling Method Engineering to extend this coverage, with the tool providing the ability to create hyperlinks between models expressing different enterprise views. The agile modelling tool developed as a proof-of-concept is further coupled with semantic technology to make models available to semantic queries and machine reasoning in the context of model-driven software engineering.

## 1 INTRODUCTION

Motivated by project-based requirements (the EnterKnow project (Enter Know, 2017)), we analysed the Zachman Framework (ZF) as a motivational starting point for implementing an enterprise modelling method. Although ZF is commonly presented as an ontology, it is not one in the formal sense employed in ontology engineering (as promoted in the works of (Smith, 2004) or (Guarino, 1995)). Also, it is not truly a methodology (Sessions, 2007), so it cannot be considered a fully-fledged modelling method in the sense defined by (Karagiannis and Kühn, 2002). It is a bi-dimensional schema that can guide enterprise architects and enterprise architecture managers in making decisions based on definition, design and analysis of architectural information for enterprises and their information systems (Zachman, 1982).

Consequently, we aimed to investigate how ZF's principles can be grounded in more formal and actionable knowledge structures, with respect to software engineering needs. More precisely, we aim to derive machine-readable knowledge from a ZF-guided enterprise architecture design, in a way that can further support knowledge-driven software development processes for Enterprise Information Systems. Our investigation identified several

opportunities and paradigms developed during recent years, whose interplay supports our goal: (i) the *Multi-perspective Enterprise Modelling* paradigm which advocates the description of an enterprise through multiple models reflecting different facets of the same enterprise (Kingston and Macintosh, 2000) (Frank, 2000); (ii) the *Agile Modelling Method Engineering* (AMME) framework (Karagiannis, 2015) which enables the full customization and quick prototyping of multi-view modelling tools according to situation-specific requirements – see an example in (Bork, 2015); (iii) the *Linked Open Models* proposal (Karagiannis and Buchmann, 2016) which advocates the serialization of diagrammatic models in RDF knowledge graphs (W3C, 2017a) in order to make diagrammatic content available to running knowledge-based systems outside a modelling environment.

Based on this investigation, we advocate the possibility of coupling the design thinking suggested by ZF's conceptual frame with semantic technology that can expose it to semantic queries and machine reasoning. In support of this proposal, we developed a proof-of-concept agile enterprise modelling tool that reflects the multi-faceted nature of ZF while establishing machine-readable semantic links governed by an overarching meta-model to expose the heterogeneous customised models as machine-readable knowledge graphs. One current limitation

due to project requirements and limited scope is that not all of ZF's perspectives have been deployed in the proof-of-concept, only a selected subset correlated with the requirements for a model-driven software artefact. The software engineering method employed to develop model-driven software artefacts based on our tool is discussed in a different paper (Buchmann et al., 2018).

This paper is structured as follows: Section 2 provides background information about ZF, the AMME methodology and the RDF technological space. Section 3 comments on related works. Section 4 discusses design decisions. Section 5 presents the proof-of-concept agile enterprise modelling tool and the final Section 6 provides conclusions based on a SWOT analysis.

## 2 BACKGROUND, METHODOLOGY AND ENABLERS

### 2.1 Motivation: The Zachman Framework

ZF is an enterprise ontology prescribing a fundamental structure for Enterprise Architecture – i.e., a formal and structured way of viewing and defining an enterprise (Zachman, 2008). The representation of this ontology is often in the form of a 6x6 matrix, a two-dimensional schema used in order to represent and organise the architectural layers and artefacts comprising an enterprise system.

Figure 1 shows a representational form of this framework in which the columns represent the ZF facets and the rows represent the types of stakeholders: (i) execution perspective (contextual level); (ii) business management perspective (conceptual level); (iii) architect perspective (logical level); (iv) engineer perspective (physical level); (v) technician perspective (as built); (vi) enterprise perspective (functioning).

The core idea is that an asset can be described in different ways for different purposes using those levels of abstraction and views. This has also been recognised in multi-view enterprise modelling (Bork, 2015) – the principle can be transferred to the practice of Agile Modelling Method Engineering with the goal of producing modelling tools that are customised to capture, in a semantically integrated way, the facets of ZF or enterprise assets pertaining to those facets.

	What	How	Where	Who	When	Why
Contextual	Business Function Modeling External Requirements and Drivers					
Conceptual	Business Process Model Enterprise Model					
Logical	Logical Models Requirements Definition					
Physical	Technology Model Physical Models					
As built	As built Deployment					
Functioning	Enterprise Evaluation					

Figure 1: Zachman’s Framework abstraction overview.

### 2.2 Methodology: The Agile Modelling Method Engineering

Agile Modelling Method Engineering (Karagiannis, 2015) can be considered a Design Science (Peffer et al., 2007) approach that is specialised for the realisation of artefacts such as modelling methods and modelling tools, regardless of application domain and abstraction level. It advocates the ability of agilely evolving the modelling semantics, syntax or functionality with the help of meta-modelling environments such as ADOxx (BOC Group, 2017). In AMME, agility applies on two levels: (i) *artefact agility* – which means that a modelling method is decomposed in manageable building blocks of various granularities (i.e., modelling language, mechanisms and algorithms, modelling procedure) and (ii) *methodological agility* which refers to the actual engineering process - an incremental and iterative development cycle in 5 steps: Create, Design, Formalise, Develop and Deploy – see Figure 2, adapted from (Karagiannis, 2015) and subsequent presentations (OMiLAB, 2017).

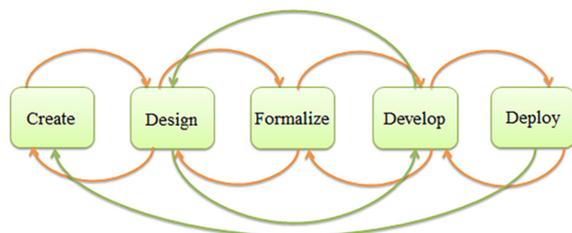


Figure 2: AMME lifecycle.

Nowadays, the notions of Agile Enterprise (Goldman et al. 1994) and Agile Knowledge Management (Levy and Hazzan, 2008) are commonly adopted in management practices and Enterprise

Architecture Management could embrace a certain level of agility in its model-based knowledge repositories, one that can be synchronised with software engineering processes. With the AMME framework, agile modelling methods reflect the fact that different purposes for different scopes must be addressed due to the diversity of domains and requirements. To this, the *Linked Open Models* vision proposed in (Karagiannis and Buchmann, 2016) adds the opportunity of exposing agile models to knowledge-driven information systems, with the help of the Resource Description Framework.

### 2.3 Technological Enablers: The Resource Description Framework

The Resource Description Framework (RDF) (W3C, 2017a) is a World Wide Web Consortium standard originally designed as a metadata model and evolved as a technological foundation for the Semantic Web. It can be used as a data model for conceptual descriptions of Web resources in Linked Data environments and coupled with ontologies.

The core idea of this data model is to capture knowledge as statements about Web resources, in the form of triples (subject-predicate-object). The subject represents the resource and the predicate asserts a relationship between the subject and the object or an attribute whose value is the object. A collection of RDF statements forms a directed multi-graph that can be managed with the help of graph databases – e.g. (Ontotext, 2017) – and queried through SPARQL queries (W3C, 2017b). The Linked Open Models vision (Karagiannis and Buchmann, 2016) introduced several patterns and a plug-in for converting diagrammatic models of arbitrary types to RDF – this is employed in the work at hand for the ZF-driven modelling tool hereby discussed.

## 3 RELATED WORKS

ZF is a well-established guide for structuring enterprise knowledge. It was originally used for the purpose of business system planning and it is discussed under multiple interpretations by managers, designers, programmers and other types of stakeholders. This framework, designed as a schema, prescribes facets and perspectives on enterprise descriptions – however it does not specify the means of bridging them in machine-oriented ways, and how to further expose those bridges to external processing.

During the '90s several frameworks extended ZF – e.g., Evernden, The Integrated Architecture

Framework (Schekkerman, 2003). ZF is also used to map various processes relevant to enterprise architectures – e.g., analysis of the Rational Unified Process (DJ de Villiers, 2001), Model-driven architecture (Frankel et. al., 2003), TOGAF (The Open Group, 2017a).

This paper's proposal tries to unify and combine ZF, AMME and Linked Open Models towards the goal of increasing the value and reusability of enterprise architecture knowledge, opening the possibility of interoperability with model-driven information systems. AMME is used to partition with appropriate granularity a custom-made modelling language following the guidance of ZF's conceptual frame and to agilely implement this in a modelling tool. Then the Linked Open Models approach is employed to make the resulting structure (and any models built on it) available to semantic information systems. The properties of enterprise artefacts conceptualised in models can be navigated through a dereferencing mechanism (originally presented in (Cinpoeru, 2017) with application to BPMN) or through SPARQL queries.

Enterprise knowledge often relies on RDF multi-graphs coupled with ontologies (Teyeb et. al., 2015). Modelling languages and conceptual patterns driven by domain-specific requirements are also becoming more prominent (Brambilla and Umuhoza, 2017) (Galkin et. al., 2017). However, there is little work in bridging all these paradigms towards a richer model-based semantic support for enterprise information systems.

## 4 DESIGN DECISIONS

The modelling tool to be described in this paper was developed to combine the benefits of ZF, AMME and RDF. By combining these, we want to show the possibility of making a multi-perspective conceptual frame available to semantic queries. Cross-perspective links are governed by a meta-model which enables the representation of enterprise knowledge as machine-readable knowledge graphs covering the facets of ZF that are deemed relevant for a particular purpose.

Figure 3 shows examples of mappings between ZF's facets and enterprise modelling symbols from two well-known languages – Archimate (Open Group, 2017b) and EEML (Carlsen, 1998).

The work at hand takes a similar mapping as a starting point and specifies granular semantic links across the ZF facets that can be exported as bridges between RDF multigraphs. To develop the modelling

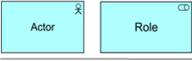
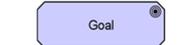
Symbol example (Archimate)	Symbol example (EEML)	Zachman facet
	 Task	How + When
	 Roles	Who
	 Artefacts	What
	 Goal	Why

Figure 3: Examples of diagrammatic symbols in modelling languages reflecting the ZF facets.

tool, we started from creating a class diagram which represents the meta-model governing the modelling language, which can be evolved by applying the AMME methodology as new requirements are adopted (see Figure 4 for a snapshot of the meta-model).

The meta-model was implemented using the ADOxx platform (BOC Group, 2017), which provides some abstract classes (e.g., `__D-construct__`, `__D-resource__`, `__D-container__` etc.) as a starting point for the multi-perspective modelling language. The class `__D-construct__` represents the root class for every meta-model and all the other abstract or non-abstract classes inherit from it. As predefined abstract classes, we also used the classes `__D-resource__`, `__D-container__` and `__D-aggregation__`. The meta-model contains so far three types of diagrams: (i) Model of courier (make-to-order) processes, (ii) Model of locations and (iii) Model of participants.

These three types of models/diagrams are linked to each other by hyperlinks, the tasks require parkings in cities or regions and also can have assigned roles or employees. With these three types of models we tried to cover the following ZF facets: (i) When/How, (ii) Where and (iii) Who, with symbols indicated in Figure 5.

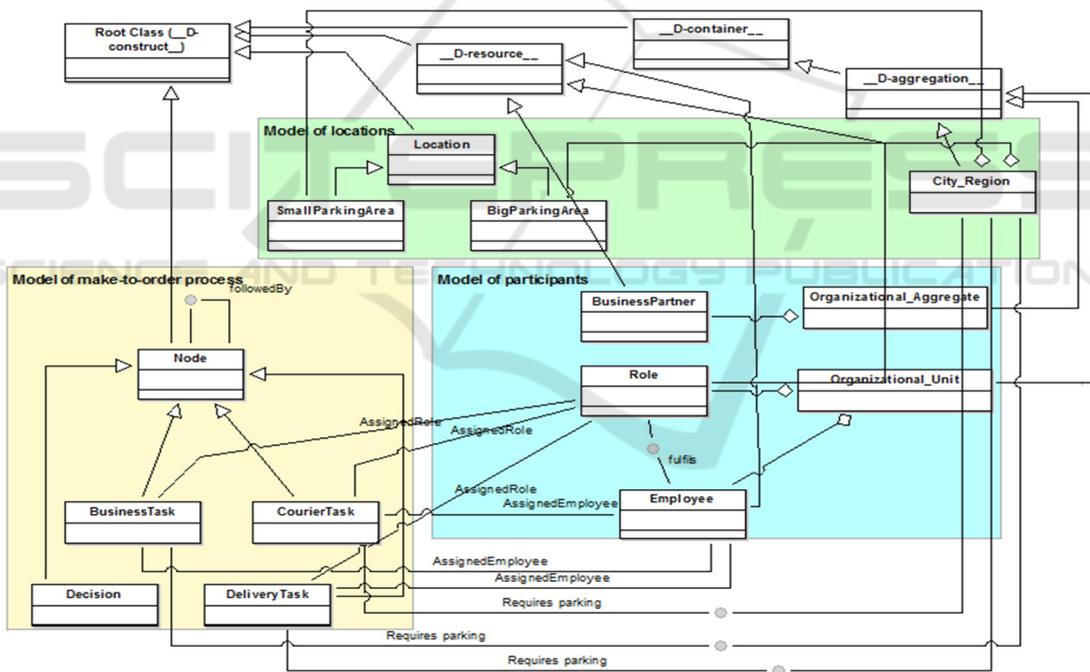


Figure 4: Meta-Model (as Class Diagram).

Because of the project's limited scope, not all of ZF's facets have yet been implemented, but this target could be reached with the help of AMME and ADOxx platform, which allows the addition of new types of models in order to extend the actual meta-model to cover all ZF facets. Across these facets, hyperlinks such as those highlighted in Figure 6 can be established, distinguished by their semantics

according to the metamodel. Different kinds of links are visible: The task *Deliver* has as assigned employee the instance *Jim* or, for convenience, the reverse link: *Jim* is responsible for doing the task *Deliver*; the employee *Jim* can fulfil the role *Big Car Driver* (roles can also be assigned to tasks, thus delegating the instance assignment to some external workflow management system); the task *Deliver*

Symbol	ZF facet
	How + When
	Who
	Where

Figure 5: Examples of symbols and corresponding ZF facets.

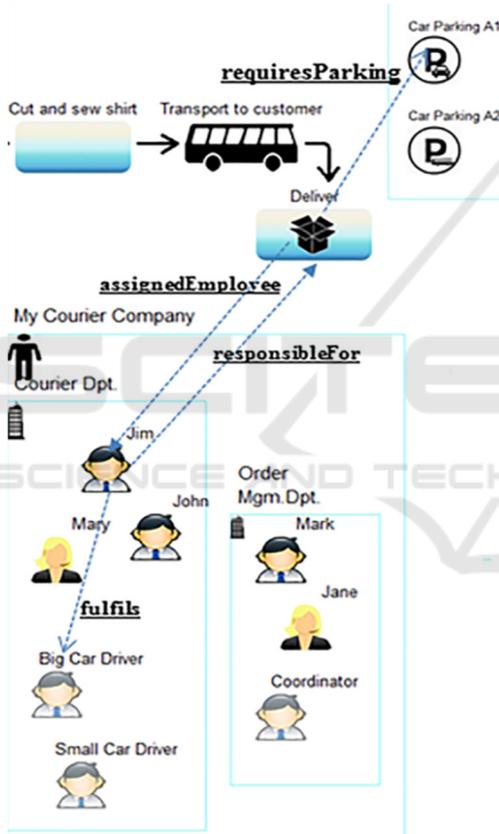


Figure 6: Linked model fragments.

requires parking in one of the cities, in this case *City A* which contains two parkings areas of different types.

All the models developed in the implemented modelling tool can be exported and interpreted as machine-readable content (RDF knowledge graphs). RDF multi-graphs are used to represent the knowledge, in this case the modelled enterprise knowledge, and can be written in different

serialization syntaxes, e.g.: TriG (W3C, 2017c); Turtle (W3C, 2017d); RDF/XML (W3C, 2017e). A graphical representation of the model fragments and links in Figure 6 is shown in Figure 7, including the types derived from the metamodel (Figure 4).

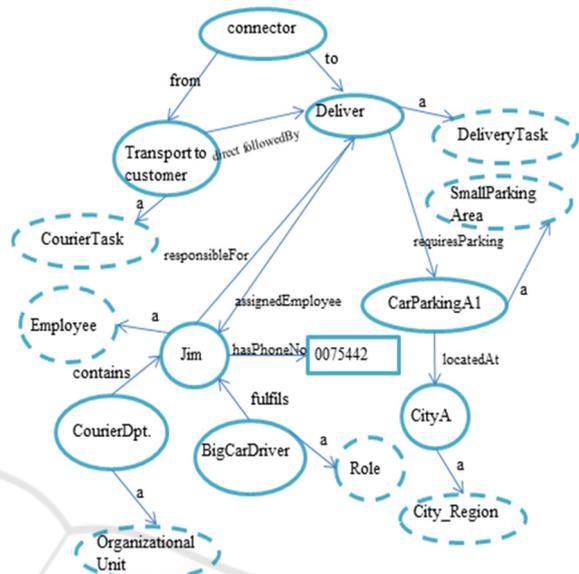


Figure 7: Machine-readable knowledge graph.

## 5 PROOF-OF-CONCEPT

For developing the agile modelling tool we adopted the case of a transport company that needs its courier processes mapped on human resources and geographical coverage, including instance processes whose tasks can be assigned in the modelling environment to instance responsables and locations. The human resources are described both in terms of roles and instance performers, grouped by departments or organisational units depicted as visual containers, e.g.: Production; Research/Development; Marketing; Finance; Resources. Figure 8 depicts examples of models developed so far in the implemented modelling tool: Model of make-to-order process (M1); Model of participants (M2); Model of locations (M3).

The first model contains a business process comprising three types of tasks and also decisions. The second type of diagram contains some departments of the transport enterprise (in which we have the employees) and also some business partners. Finally, the third model contains cities or regions with two types of parkings: small parking area and big parking area. All these types of models can be linked across the ZF facets they represent: (i) the tasks are

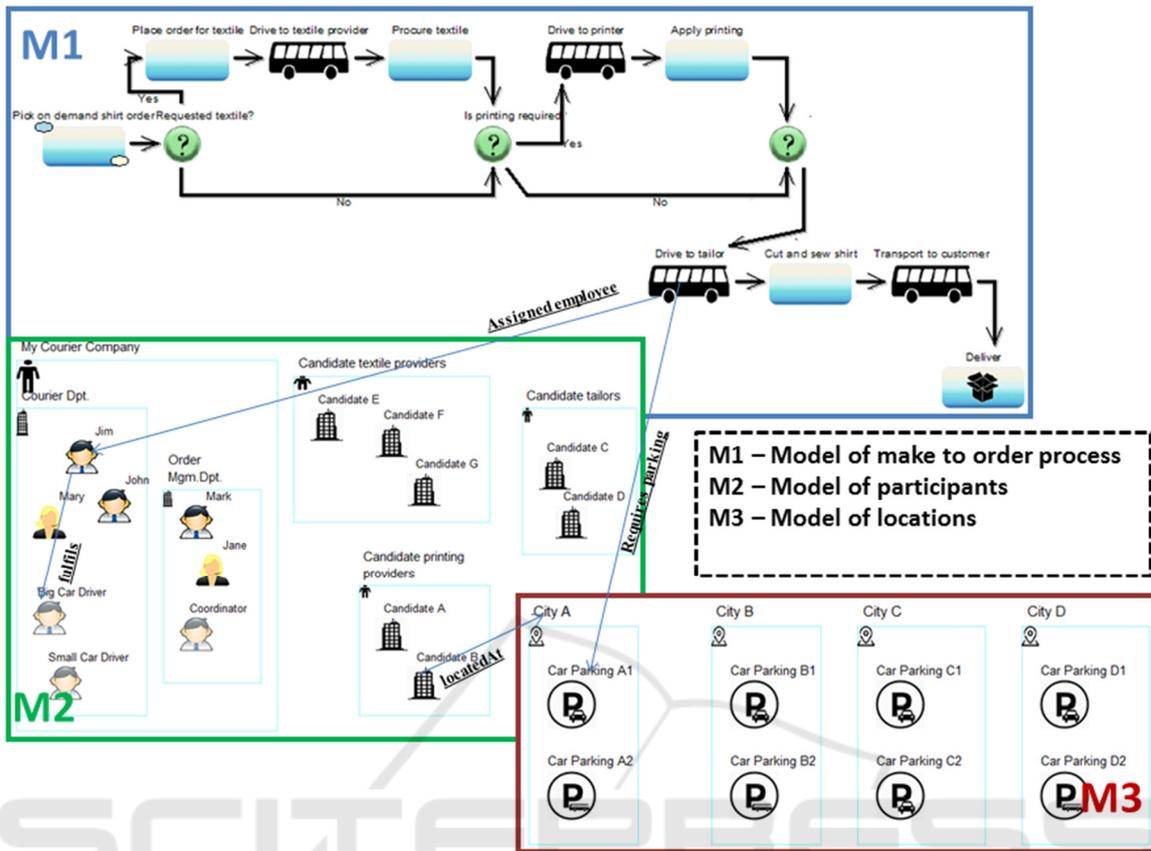


Figure 8: Samples of models in the multi-view modelling tool.

```

<http://www.example.org/DeliveryTask-49803-Deliver>
  <http://www.comvantage.eu/mm#described_in>
    <http://www.example.org/Business_Process_Diagram-Business_Process_Diagram_-_new> ;
  <http://www.example.org#Assigned_employee> <http://www.example.org/Employee-49815-Jim> .

<http://www.example.org/Employee-49815-Jim> <http://www.comvantage.eu/mm#described_in>
  <http://www.example.org/Resource_Diagram-Resource_Diagram_-_new> ;
  <http://www.example.org#Name> "Jim" ;
  <http://www.example.org#Position> "NODE x:1.5cm y:6cm index:1" ;
  <http://www.example.org#fulfills> <http://www.example.org/Role-49818-Driver> ;
  a <http://www.comvantage.eu/mm#Instance_class> , <http://www.example.org#Employee> .
    
```

Figure 9: RDF model serialisation sample.

performed by employees; (ii) the tasks require parking areas in cities.

In Figure 9 some fragments in RDF are presented, describing the link between the *Deliver* task and the *Jim* instance (including some positioning properties relevant in case serialised models are imported in another tool).

Based on these statements, we can run SPARQL queries from any programming environment, in order to access information from client applications (e.g., a workflow management system). Some examples of SPARQL queries are:

```

SELECT ?x WHERE
  {?x a :Employee.}
    
```

```

SELECT ?role (Count(?task) as ?count)
WHERE {
  ?task a :DeliveryTask.
  ?task :AssignedRole ?role.}
    
```

The first query returns all the employees that work in the given enterprise and the second query counts the tasks assigned to a role. Based on a similar example, paper (Buchmann et al., 2018) discussed the

overarching software engineering method enabled by our proposal, whereas (Karagiannis and Buchmann, 2018) discusses the OWL reasoning opportunities that are open by this proposal when using a graph database management system such as GraphDB (Ontotext, 2017).

## 6 CONCLUDING SWOT ANALYSIS

The paper repurposes ZF principles to create a new kind of enabler for enterprise model-driven engineering – agile modelling tools that expose ZF abstractions and their semantic links to knowledge-driven software development processes. We encapsulate the concluding discussion in a SWOT analysis:

**Strengths:** The proposal provides a use case for both (i) graph databases and (ii) agile modelling methods which can deviate from standards in order to fulfil enterprise-specific requirements. The ZF abstractions are connected in ways that can be exposed as RDF graphs and can be further complemented by ontologies with the goal of deriving hybrid knowledge bases – an aspect detailed in (Karagiannis and Buchmann, 2018).

**Weaknesses:** Evaluations are necessary for industry-oriented or standard languages; a client-side proof-of-concept of model-aware application benefitting from the proposed knowledge acquisition enabler is not discussed in this paper – details on the software engineering method that is enabled by the proposal are available in (Buchmann et al., 2018).

**Opportunities:** (i) The GeoSPARQL ontology (OGC, 2017) creates opportunities for geospatial inferences, which are considered for the future evolution of the tool; (ii) An enterprise should be modelled in a multi-perspective manner and the relations between those perspectives can be managed by combining ZF's prescriptive schema, AMME and RDF; (iii) Graph-like structures are more intuitive for humans as means of knowledge representation and the proposed tool could be evolved towards a graphical RDF editor.

**Threats:** So far the adoption of RDF and Semantic Web principles is still slow in enterprise applications.

## ACKNOWLEDGEMENT

This work is supported by the Romanian National Research Authority through UEFISCDI, under grant agreement PN-III-P2-2.1-PED-2016-1140.

## REFERENCES

- BOC-Group GmbH, 2017. *ADOxx platform page – official website*, <http://www.adoxx.org/live>, last accessed: 30<sup>th</sup> October 2017.
- Bork, D., 2015. *Using conceptual modelling for designing multi-view modelling tools* In: *Proceedings of the 21st Americas Conference on Information Systems*, Association for Information Systems.
- Brambilla, M., Umuhoza, E., 2017. *Model-driven Development of User Interfaces for IoT Systems via Domain-specific Components and Patterns*. In *Proceedings of the 19th International Conference on Enterprise Information Systems - Volume 2*, pp. 246-253, SCITEPRESS.
- Buchmann, R.A., Cinpoeru, M., Harkai, A., Karagiannis, D., 2018. *Model-aware software engineering: a knowledge-based approach to model-driven software engineering*. In *Proceedings of the 13<sup>th</sup> Int. Conf. on Evaluation of Novel Approaches to Software Engineering*, SCITEPRESS (in press).
- Carlsen, S., 1998. *Action port model: A mixed paradigm conceptual workflow modeling language*. In: *Proceedings of Third IFCIS Conference on Cooperative Information Systems*, ACM.
- Cinpoeru, M., 2017. *Dereferencing service for navigating enterprise knowledge structures from diagrammatic representations*. In *Proceedings of BIS 2017 Workshops*, LNBIP 303, pp.85-96, Springer.
- Enter Know, 2017. Project page <http://enterknow.granturi.ubbcluj.ro>, last accessed: 30<sup>th</sup> October 2017.
- Frank, U., 2000. *Multi-perspective enterprise models as a conceptual foundation for knowledge management* In: *Proceedings of the 33rd Hawaii Conf. on System Sciences*, IEEE.
- Frankel, D. S., Harmon, P., Mukerji, J., Odell, J., Owen, M., Rivitt, P., Rosen, M., Soley, 2003. *The Zachman Framework and the OMG's Model Driven Architecture*. White paper. Business Process Trends.
- Galkin, M., Auer, S., Vidal, M., Scerri S., 2017. *Enterprise Knowledge Graphs: A Semantic Approach for Knowledge Management in the Next Generation of Enterprise Information Systems*. In *Proceedings of the 19th International Conference on Enterprise Information Systems - Volume 2*, pp. 88-98, SCITEPRESS.
- Goldman, S., Naegel, R., Preiss, K. 1994. *Agile Competitors and Virtual Organizations: Strategies for Enriching the Customer*. Wiley

- Guarino, N., 1995. *Formal ontology, conceptual analysis and knowledge representation*. In *International Journal of Human-Computer Studies*, 43(5-6), pp. 625-640.
- Karagiannis, D., 2015. *Agile modeling method engineering*. In *Proceedings of the 19<sup>th</sup> Panhellenic Conf. on Informatics*, pp. 5-10, ACM.
- Karagiannis, D., Buchmann, R. A., 2018. *A proposal for deploying hybrid knowledge bases: the ADOxx-to-GraphDB interoperability case*. In *Proceedings of the 51<sup>st</sup> Hawaii Int. Conf. on System Sciences*, pp. 4055-4064, University of Hawaii.
- Karagiannis, D., Buchmann, R., 2016. *Linked Open Models: Extending Linked Open Data with conceptual model information*. In *Journal Information System*, 56, pp. 174-197.
- Karagiannis, D., Kühn, H., 2002. *Metamodelling platforms*. In *Proceedings of the Third International Conference EC-Web 2002 – DEXA 2002*. LNCS 2455, 182, Springer.
- Kingston, J., Macintosh, A., 2000. *Knowledge management through multi-perspective modelling: representing and distributing organizational memory*. In *Journal of Knowledge-based Systems*, 13(2-3), pp. 121-131.
- Levy, M., Hazzan, O., 2008. *Agile Knowledge Management*. In: *Encyclopedia of Information Science and Technology*, pp. 112-117, IGI Global
- OGC, 2017. *GeoSPARQL – A Geographic Query Language for RDF Data*, <http://www.opengeospatial.org/standards/geosparql>, last accessed: 30<sup>th</sup> October 2017.
- OMiLAB, 2017. NEMO Summer School series lectures, <http://nemo.omilab.org>, last accessed: 30<sup>th</sup> October 2017
- Ontotext, 2017. *GraphDB Downloads and Resources*, <http://graphdb.ontotext.com/>, last accessed: 30<sup>th</sup> October 2017.
- Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S., 2007. *A Design Science Research Methodology for Information Systems Research*. In *Journal of Management Information Systems*, 24(3), pp. 45-77.
- Schekkerman, J., 2003. *How to Survive in the Jungle of Enterprise Architecture Frameworks*. pp. 139-144, Trafford.
- Sessions, R., 2007. *A Comparison of the Top Four Enterprise Architecture Methodologies*. Microsoft Developer Network Architecture Center.
- Smith, B., 2004. *Beyond Concepts: Ontology as Reality Representation*. In *Proceedings of the Third International Conference on Formal Ontology in Information Systems*, pp. 73-84, IOS Press.
- Teyeb, J. R., Torjmen, K., M., Hernandez N., Haemmerle O., Ben J. M., 2015. *Semantic Annotation of Images Extracted from the Web using RDF Patterns and a Domain Ontology*. In *Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 2*, pp. 137-144, SCITEPRESS.
- The Open Group, 2017a. *ADM and the Zachman Framework*, <http://pubs.opengroup.org/architecture/togaf8-doc/arch/chap39.html>, last accessed: 30<sup>th</sup> October, 2017.
- The Open Group, 2017b. *The Archimate Enterprise Architecture Modelling Language*. <http://www.opengroup.org/subjectareas/enterprise/archimate-overview>, last accessed: 30<sup>th</sup> October, 2017.
- Villiers, D. J. de, 2001. *Using the Zachman Framework to Assess the Rational Unified Process*. In *The Rational Edge*, Rational Software.
- W3C, 2017a. *RDF - Semantic Web Standards*, <https://www.w3.org/RDF/>, last accessed: 30<sup>th</sup> October 2017.
- W3C, 2017b. *SPARQL 1.1 Query Language*, <http://www.w3.org/TR/sparql11-query>, last accessed: 30<sup>th</sup> October 2017.
- W3C, 2017c. *RDF 1.1 TriG*, <http://www.w3.org/TR/trig>, last accessed: 30<sup>th</sup> October
- W3C, 2017d. *RDF 1.1 Turtle*, <http://www.w3.org/TR/turtle>, last accessed: 30<sup>th</sup> October
- W3C, 2017e. *RDF 1.1 XML Syntax*, <https://www.w3.org/TR/rdf-syntax-grammar/>, last accessed: 30<sup>th</sup> October
- Zachman, J. A., 1982. *Business Systems Planning and Business Information Control Study: A comparison*. In *IBM Systems Journal*, 21, pp. 31-53.
- Zachman, J. A., 2008. *John Zachman's Concise Definition of the Zachman Framework*. <https://www.zachman.com/16-zachman/the-zachman-framework/35-the-concise-definition>, last accessed: 30<sup>th</sup> October 2017.