Automated REA (AREA): a software toolset for a machine-readable resource-event-agent (REA) ontology specification

FALLON, Richard and POLOVINA, Simon

Published version

FALLON, Richard and POLOVINA, Simon (2016). Automated REA (AREA): a software toolset for a machine-readable resource-event-agent (REA) ontology specification. CEUR Workshop Proceedings, 1637, 10-18.

Automated REA (AREA): A Software Toolset for a Machine-readable Resource-Event-Agent (REA) Ontology Specification

Richard L. Fallon

Dr. Simon Polovina

Sheffield Hallam University, Sheffield, United Kingdom

Richard.L.Fallon@student.shu.ac.uk, s.polovina@shu.ac.uk

Abstract. This paper demonstrates a toolset developed by the authors to enable a machine-readable REA ontology specification. Information modelling techniques are used to provide a unified enterprise ontology by capturing the business semantics using Conceptual Graphs (CGs) using Common Logic (CL) and the Conceptual Graph Interchange Format (CGIF) dialect for information exchange and transmission. Formal Concept Analysis (FCA) is used for model verification, knowledge discovery and extraction. The enterprise design follows the Open Groups definition of the TOGAF Architecture Development Method (ADM) to define the system architecture and subsequently provide a method for defining and automating the (REA) design models for; Business Architecture, Information System Architecture and Technology Architecture.

Keywords: REA, ADM, Conceptual Graph, Ontology, FCA, Data integration, Semantics, Protégé

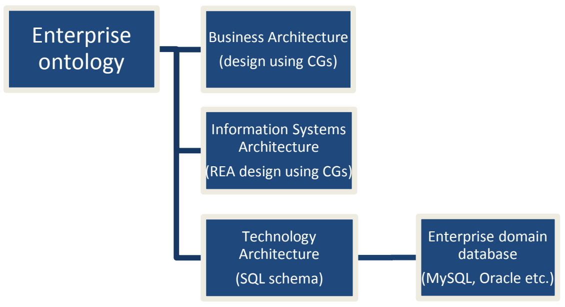
1 Introduction

The development of an enterprise ontology is one of the first steps in designing a knowledge base, a database or an object oriented system [1]. In this paper we respond to the call from Gailly, Laurier, & Poels (2008) for the development of a proof concept for the viability of having a machine-readable REA ontology specification available for different REA ontology application domains. For an REA-ontology like other business domain ontologies there is perceived to be a lack of formal representation which would be useful for the representation of the application in practice[1].

2 System requirements

The basis of the system requirements for AREA come from the Open Groups definition of the TOGAF Architecture Development Method (ADM) [2] which defines the Architecture Development Lifecycle. In addition to the ADM above we will add four further requirements; (i) to enable Model Automation (MA), the solution must be not only Human readable but also Machine readable, (ii) enable Model Visualization (MVi) and thus provide a Graphical view which can be used by domain modelling professionals, (iii) allow for Model Verification (MVe) so that domain modelling professionals can carry out tests against the proposed Enterprise ontology, (iv) provide a working solution via Model Implementation (MI) the point at which the model is converted into a usable Enterprise domain database. The diagram below in Fig. 1 identifies the components required for producing an Enterprise Architecture using AREA.

Fig. 1 - ADM Components of an Enterprise Architecture using AREA


Enterprise ontology
Business Architecture (design using CGs)
Information Systems Architecture (REA design using CGs)
Technology Architecture (SQL schema)
Enterprise domain database (MySQL, Oracle etc.)

2.1 Model Automation (MA)

Common Logic (CL).

The International Organisation for Standardisation (ISO) have defined Common Logic (CL) [3] which is a first order logic intended for information exchange and transmission. The core of the CL framework is the definition of an abstract syntax and abstract semantics for Common Logic, providing the structure for several concrete syntactic forms or dialects. One of the CL dialects is the Conceptual Graph Interchange Format (CGIF) and it is this dialect which we use for information exchange and the Automation of REA (AREA) using Conceptual Graphs (CGs).

Conceptual Graphs (CGs).
CGs have been shown to be useful in capturing and modelling business transactions and their semantics within an enterprise architecture [4]. Thus our solution uses CGs as the driving force behind MA providing a tool to define the Enterprise ontology. CGs provide both accurate semantic and syntactic refinement, aiding the designer in capturing semantics. Whilst CG's in CGIF format are machine readable they can also be interpreted and edited manually (human readable) and are useful for documentation and maintenance.

REA.
McCarthy (1982) proposed his REA theory as the solution to a generalised accounting framework in which both accountants and non-accounts could share the same enterprise data. McCarthy's [5] enterprise ontology determines that all the Economic transactions within an enterprise can be aligned to three separate entities or concepts; (i) Economic Resources, (ii) Economic Events and (iii) Economic Agents. Moreover, REA has also been demonstrated to be useful for modelling non-economic business transactions [6, 7].

2.2 Model Visualization (MVi)

Protégé.
Protégé is a free open-source contemporary ontology editor and framework and was developed at the Stanford Centre for Biomedical Informatics Research (BMIR) at the Stanford University School of Medicine for building intelligent systems. Protégé is a mature solution already used in commercial products [8] and hence has numerous add-on tools available. There are essentially two versions of Protégé – OWL and Frames, due to the more restrictive nature of Frames this version was chosen as the ontology repository. For a detailed description of the difference between Frames and OWL, refer to [9].

CG Import.
The Protégé plugin CGImport [10] was developed by the authors as a tool to allow for the import of CGs defined using the CGIF Common Logic dialect directly into the Protégé ontology repository. Thus allowing the user to define both the Business Architecture and the Information System Architecture using CGs from within a text editor or to import the CGs designed using other available CG tools such as CharGer [11].

CG Export.
The Protégé plugin CGexport [12] was developed by the authors as a tool to allow for the export of CGs stored within the Protégé ontology repository into a file using the CGIF Common Logic dialect. Thus allowing for interoperability with other available CG tools such as CharGer [11].

2.3 Model Verification (MVe)

JESS.
JESS [13] is a tool available as a Protégé plugin, used to build an expert system which can then be used to build a set of rules which can be repeatedly executed against a collection of facts such as an ontology describing an enterprise system. Thus JESS is used to validate the Enterprise Model once it is stored within Protégé.

FCA.
Several authors have shown the benefits of converting CGs into Formal Concepts [14] to allow for Formal Concept Analysis (FCA), to find out the hidden information and transactions within complex graphs. The tool FCAView[15] was further developed by the authors to allow for FCA on the enterprise model stored directly within Protégé [15-17].

2.4 Model Implementation (MI)

To complete the process and carry out MI, the final product or artefact of the solution is the enterprise domain database in the form of an SQL schema, for installation using one of the current database technologies (MySql, Microsoft SQL server, Oracle etc.).

REAtoSQL.
The Protégé plugin REAtoSQL has been developed by the authors as a tool to allow for the export of the Technology Architecture in the form of an SQL schema from the REA ontology design stored within the Protégé repository. The SQL schema can then be used to initialize the enterprise domain database.

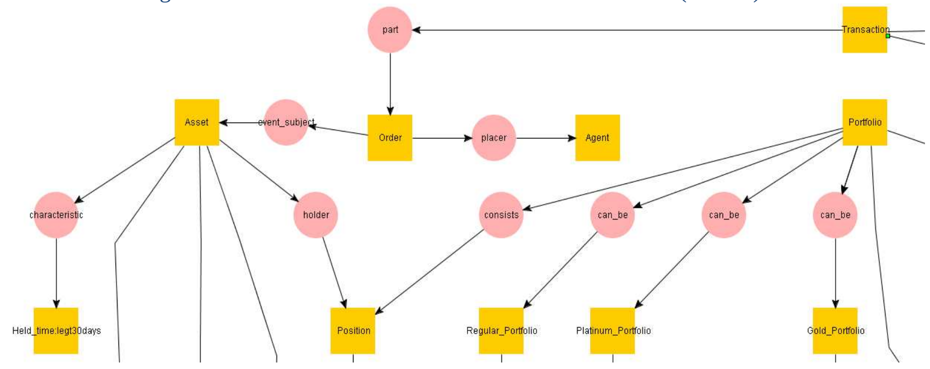
3 Solution demonstration

To demonstrate the solution, we use the example of a Financial Trading (FT) case study for the purpose of simplicity and to allow for comparison with other solutions using this same case study.

3.1 Business Architecture

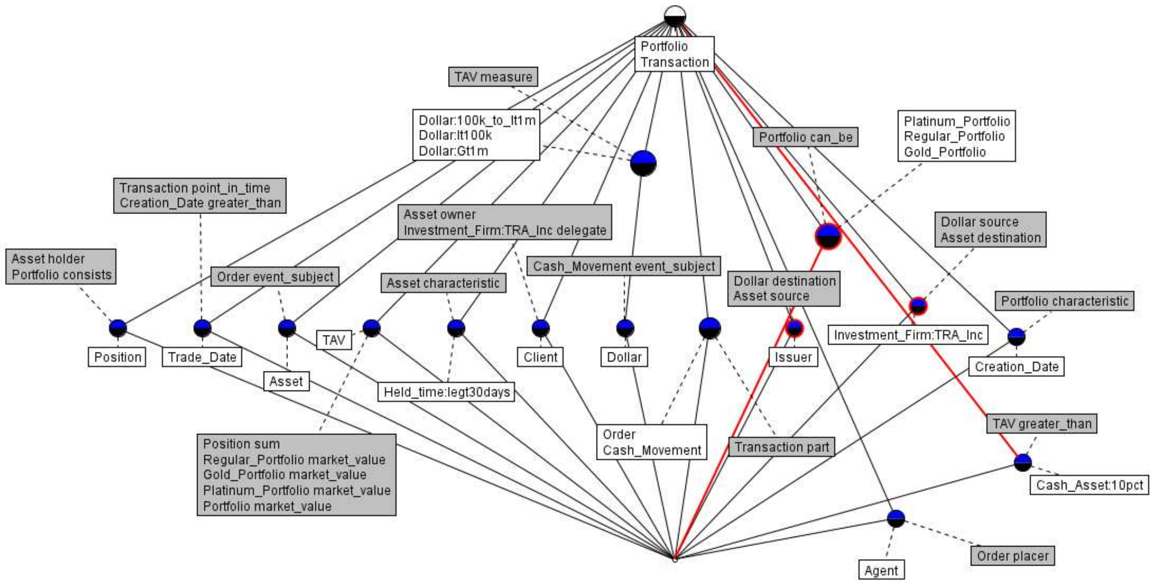
Using MA, we define the Business Architecture using CGs by importing (using CGimport) models from other tools (CharGer), or models defined manually in a text editor into Protégé for MVi. The complete Business Architecture defined using CGs can then be reviewed (MVi) in Protégé, further CGs can be added to define further business entities and expand the type hierarchy. Thus Protégé allows for MVi of the Business Architecture and to carry out further design work (Fig. 2).

Fig. 2 - Business Architecture Model Visualisation MVi (extract)



To validate the enterprise model there are then two methods for model Verification (MVe); (i) using either JESS to ‘test’ business rules or (ii) using FCAView [15] to view the Business Architecture and complete FCA directly in Protégé (Fig. 3).

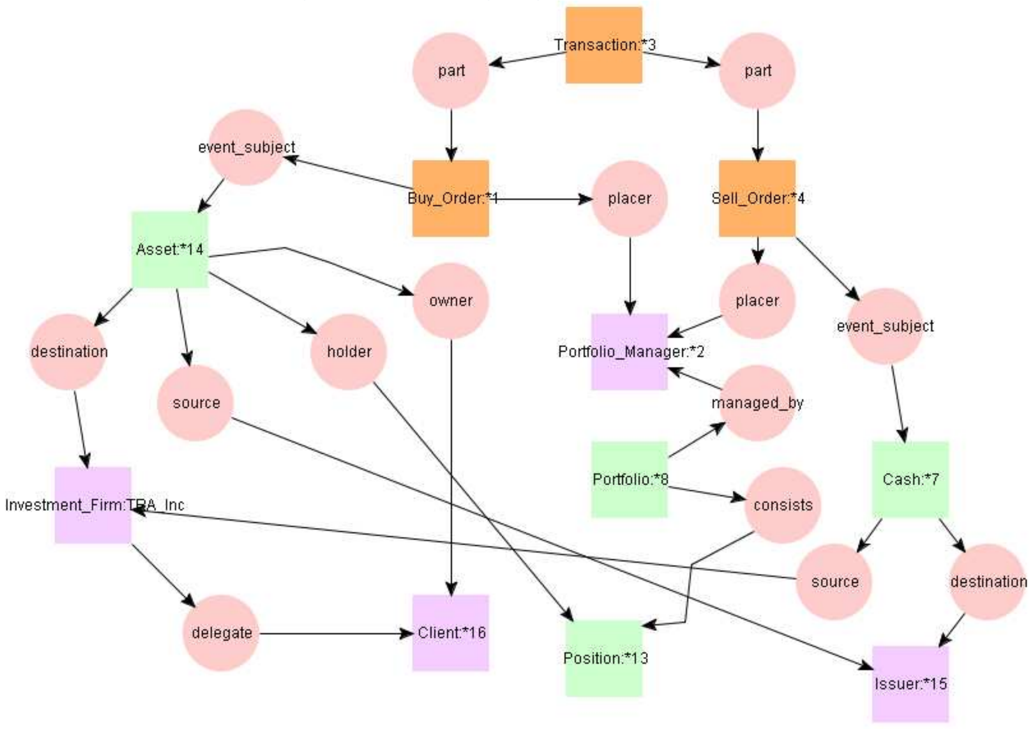
Fig. 3 - Business Architecture Model Verification FCA (MVe)



3.2 Information System Architecture

Using the CGs developed for the Business Architecture (above) we expand the design and the type hierarchy to include the Resource Event Agent (REA) design pattern. Using REA to define the Information System Architecture we focus on real things not artificial artefacts (Dunn, Cherrington and Hollander 2005) by adding detail to each of the concepts (Fig. 4).

Fig. 4 - Information System Architecture - Model Visualisation – REA entities, key: Resources-green, Events-orange, Agents-lilac



The resulting design can then be used later for the basis of the Technology Architecture in the form of an SQL schema.

3.3 Technology Architecture

Using Protégé, the Technology Architecture can be developed using the REA ontology design from above by using the developed tool **REAtoSQL**. Thus allowing Model Implementation (MI) by exporting from Protégé the SQL schema, which can then be used to load the enterprise domain database into the chosen database software (MySQL, Oracle etc.). The resulting database should then mimic the domain model as closely as possible [18].

4 Conclusions

In this paper we have responded to the call [19] for the viability of having a machine-readable REA ontology specification. A correct formal representation of the REA-ontology offers great opportunities and will facilitate the operationalization of the REA-ontology [20]. UML modelling has also been proposed as the solution to this problem, however, the solution presented in this paper uses CG's as an alternative (instead of UML). Thus the AREA toolset allows the enterprise expert to follow TOGAF [2] and define; Business Architecture, Information System Architecture and the Technology Architecture using CGs, REA and the SQL Enterprise Domain Database, all from within a unified tool - Protégé.

References

1. Geerts GL, McCarthy WE (2002) An ontological analysis of the economic primitives of the extended-REA enterprise information architecture 3:1-16
2. TheOpenGroup (2016) TOGAF® 9.1 > Part II: Architecture Development Method (ADM), Introduction to the ADM. http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap05.html
3. ISO (2016) Information technology -- Common Logic (CL): a framework for a family of logic-based languages ISO/IEC 24707:2007. http://www.iso.org/iso/catalogue_detail.htm?csnumber=39175
4. Launders I (2012) The transaction graph: requirements in semantic enterprise architectures. LAP LAMBERT Academic, Saarbrücken.
5. McCarthy WE (1982) The REA accounting model: A generalized framework for accounting systems in a shared data environment 57:554-578
6. O'Leary DE (2004) On the relationship between REA and SAP 5:65-81
7. Fallon RL, Polovina S (2013) REA analysis of SAP HCM; some initial findings
8. Enterprise Architecture Solutions (EAS) (2016) The Essential Project. http://www.enterprise-architecture.org/
9. Wang H (2015) Frames and OWL side by side. http://protege.stanford.edu/conference/2006/submissions/slides/7.2wang_protege2006.pdf
10. Fallon RL (2015) CGImport. https://github.com/RichardFallon/CGImport
11. Delugach H (2016) CharGer Conceptual Graph and knowledge modeling tool. http://charger.sourceforge.net/
12. Fallon RL (2015) CGExport. https://github.com/RichardFallon/CGExport
13. Friedman-Hill E (2016) JESS the rule engine for the JAVA platform. http://www.jessrules.com/jess/index.shtml
14. Andrews S, Polovina S (2011) A mapping from conceptual graphs to formal concept analysis. In: Anonymous Conceptual Structures for Discovering Knowledge, Springer, pp 63-76.

15. Jiang G, Fallon RL (2016) FCAView. https://github.com/FCAView/FCAView_V2.0
16. Andrews S (2016) CGFCA - Convertor of Conceptual Graph files to Formal Context files. http://sourceforge.net/projects/cgfca/
17. Fallon RL (2016) 3to2 - Convertor of Conceptual Graph files to Formal Context files. https://github.com/RichardFallon/3to2
18. Pavel. Hrubý, Kiehn J, Scheller CV (2006) Model-driven design using business patterns. Springer, Berlin.
19. Gailly F, Laurier W, Poels G (2008) Positioning and Formalizing the REA Enterprise Ontology. J Inf Syst 22:219-248
20. Gailly F, Poels G (2005) Development of a formal REA-ontology representation 160