Adaptive Enterprise Architecture Modelling

Asif Qumer Gill*, Muhammad Atif Qureshi
School of Software, University of Technology, Sydney, Australia.

* Corresponding author. Tel.: +61 2 9514; email: asif.gill@uts.edu.au.
Manuscript submitted December 13, 2014; accepted March 18, 2015.
doi: 10.17706/jsw.10.5.628-638

Abstract: Agile or adaptive enterprise architecture driven software development approach requires a modelling standard to describe the existing and to-be developed artifacts both at the high enterprise level and low, detailed level. However, a single modelling standard may not be used off-the-shelf to fully support the modelling needs of an adaptive enterprise architecture driven software development needs. The modelling standards need to be systematically analyzed and integrated for a particular modelling context. This paper reviews two well-known modeling standards ArchiMate and BPMN by using the interoperability research framework. Based on the syntax, semantics and structural analysis of these two modelling standards' metamodels, it proposes a hybrid adaptive enterprise architecture modelling approach for describing and analysing the artifacts both at the high enterprise level and low, detailed level for a particular context. This paper has both theoretical and practical implications for researchers and practitioners pursuing to integrate various modelling standards.

Key words: Adaptive modelling, agile modelling, archimate, BPMN, enterprise architecture, process modelling.

1. Introduction

The modern distributed and complex enterprise agile software development requires the need for establishing an agile or adaptive holistic enterprise architecture (EA) capability [1], [2]. An adaptive EA capability is important for proactively scanning the operating environment and for seeking any changes or opportunities for *improvement* and development in the context of enterprise agile software delivery [3], [4]. The establishment of a situation-specific adaptive EA capability requires the adoption and tailoring of a standard approach for modelling the EA artifacts both at the high and low, detailed level. There are a number of modelling standards to choose from such as ArchiMate [5], Business Process Model and Notation (BPMN) [6], Unified Modelling Language (UML) [7], and Agent [8]. Each modelling standard differs in scope and notation.

A single modelling standard may not be used off-the-shelf to fully support the modelling needs of adaptive enterprise architecture driven agile software intensive organisations. This paper addresses this issue and reviews the two well-known mainstream EA modelling standards such as ArchiMate and BPMN from syntax, semantics and structural perspectives. Based on this review, it proposes the development of a situation-specific integrated hybrid modelling approach for describing and analysing the EA artifacts both at the high enterprise level and low, detailed level for a particular agile software development context.

This paper is organized as follows. Firstly, it provides an overview of the ArchiMate and BPMN metamodels. Secondly, this paper adopts the interpretability research framework [9] as an analytical approach for the review and comparison of these two standards from their syntax, semantics and structural perspectives. Thirdly, it discusses review results and the quality of our approach. Finally, it proposes and describes an integrated hybrid modelling approach for supporting adaptive EA modeling and concludes with a short discussion about future work in this important area of research.

2. Research Context: Metamodels

This section discusses the ArchiMate and BPMN metamodels in order to set the research context of this paper.

2.1. ArchiMate

ArchiMate [5] is relatively a new enterprise architecture modelling language. It can be used to model and analyse architecture within and across different enterprise architecture domains or layers such as business architecture, application architecture and technology architecture. ArchiMate business layer provides concepts such as business actor, role, collaboration, artifact, location, interface, process, service and event concepts to model the business architecture domain (see Fig. 1). ArchiMate application layer provides concepts such as application component, collaboration, interface, function, interaction, service and data object to model the application architecture. ArchiMate technology layer provides the concepts such as infrastructure node, device, system software, network, communication path, function and service concepts to support modelling the infrastructure layer.

ArchiMate allows to model relationships or interactions within the individual layer and across the layers to describe the overall EA. In addition to three main architecture layers, ArchiMate also provides two extensions: motivational concepts, and implementation and migration concepts. Motivational concepts support modelling the enterprise strategy that needs to be realised by the EA. Implementation and migration concepts support modelling the EA implementation work packages, deliverables, plateau and gap. This paper only focuses on the business layer of the ArchiMate in conjunction with BPMN.

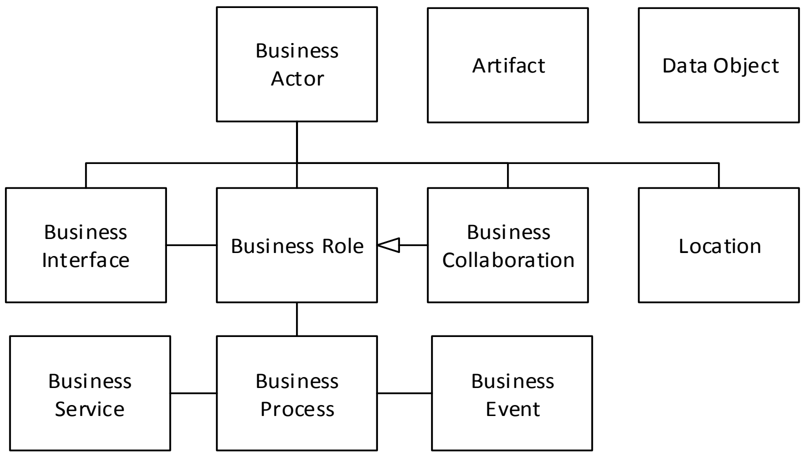

Fig. 1. Partial metamodel of archimate (business layer).

2.2. Bpmn

BPMN [4] (Business Process Modelling Notation) is the result of a standardization efforts started in 2001 by BPMI (Business Process Management Initiative) and then transferred to the OMG in 2006. BPMN is designed to support modelling of end-to-end business processes. Three basic types of models in BPMN are Processes (both private and public), Choreographies and Collaborations. These processes can be executable or non-executable. Collaborations may include Processes and/or Choreographies. Five basic categories of modelling elements used in a BPMN model are Flow Objects, Data, Connecting Objects, Swimlanes and

Artifacts.

Flow Objects are the main elements for defining the behavior of a business process. Three types of flow objects are Events, Activities and Gateways. Data are represented as Data Objects, Data Inputs, Data Outputs, Data Stores and Properties. Flow objects are connected to each other through Connecting Objects. There are four different types of connecting objects: Sequence Flow, Message Flow, Association and Data Association. Pools and Lanes are used to group modelling elements. Events and Gateways also have extended sets of elements. Fig. 2 presents the partial metamodel of BPMN that we have used in this paper for comparison with ArchiMate.

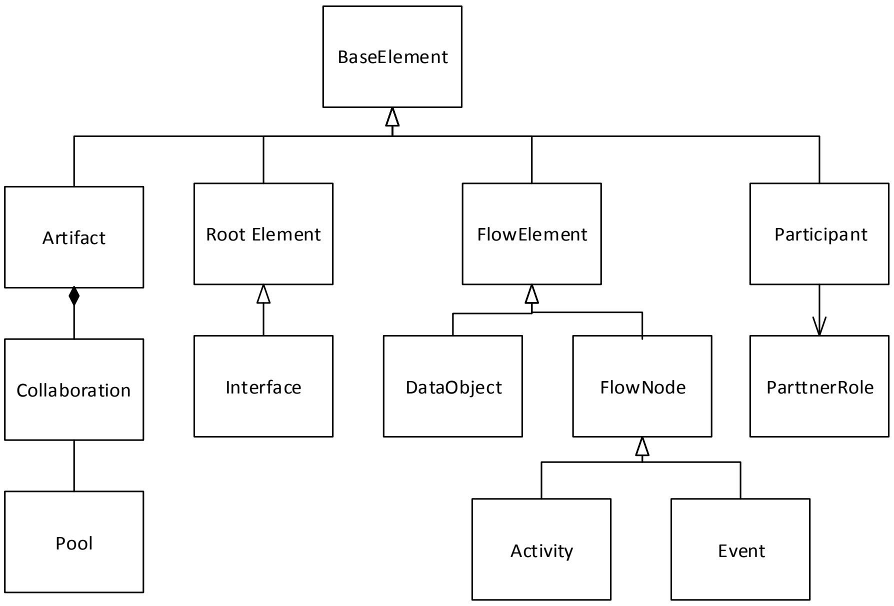

Fig. 2. Partial Metamodel of BPMN.

3. Research Approach and Analysis

There are a number of ways to analyze models and metamodels [10]-[12]. We applied the interoperability framework [9] (Fig. 3) as a research lens to review ArchiMate and BPMN metamodels. The interoperability framework (Fig. 3) seems appropriate for this research and has its roots in ontology matching and schema merging techniques [13], [14]. The framework provided a structured analysis to compare both metamodels from syntax, semantics and structural similarity perspectives. The following sections present the analysis results.

3.1. Syntactic Analysis

Syntactic similarity analysis is the comparison of metamodels' elements based on their names. We have analyzed the syntactic similarity, between a pair of conceptual elements, using the so-called Syntactic Similarity Measure (SSM) proposed by [15] (equation (1)). SSM is calculated using the edit distance formula (ED) proposed by [16]. ED counts the number of token insertions, deletions and substitutions needed to transform one lexical string Si to another *Sj*, viewing each string as a list of tokens.

The value of SSM will always be between 0 (zero) and 1 (one), where 1 represents exactly the same, 0 represents a bad match and rest of all values mean that elements have some similarity. Small values of ED indicate greater similarity between a pair of concepts. Please see below the equation (1) for details.

$$SSM(S_i, S_j) = \max\left(\frac{\min(S_i,S_j) - ED(S_i,S_j)}{\min\,(S_i,S_j)}\right) \tag{1}$$

After calculating SSM for each concept in both metamodels, the concepts that are the same or have a strong similarity are listed for further evaluation.

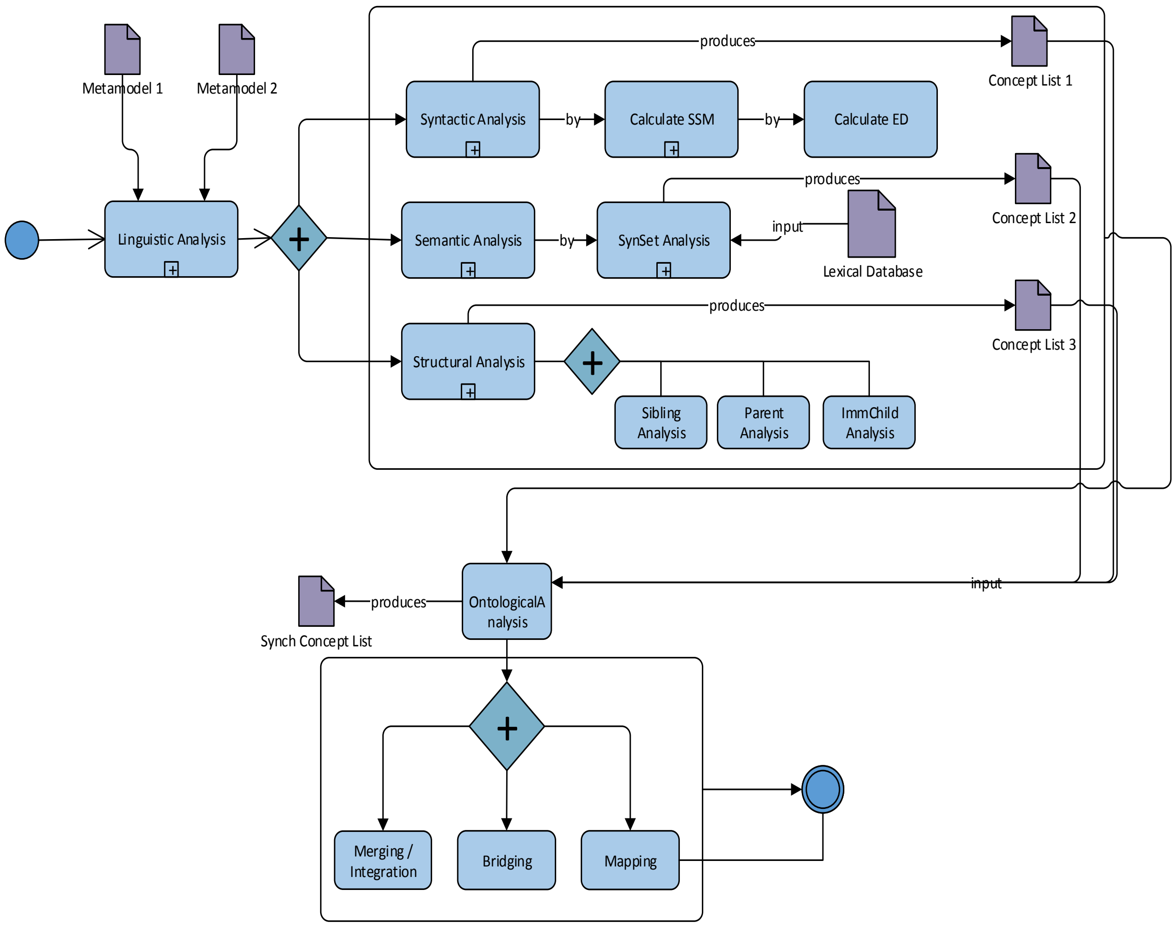

Fig. 3. The interoperability framework [9].

3.2. Semantic Analysis

The purpose of semantic analysis is to identify the concepts (conceptual elements) in both metamodels which were either not identified as similar or resulted in low syntactic similarity because of their names. It is possible that two different concepts having the same semantics may be named differently in two metamodels. Hence, identifying only syntactic similarity of metamodels' elements, isolated from their semantics, is not sufficient. For example, conceptual elements like Confirmation and Notification has approximately 60% syntactic similarity and Confirmation and Verification has 50% syntactic similarity. While considering the semantics of both of them make it is easy to determine that Confirmation has greater degree of similarity with Verification than to Notification.

Semantic analysis is done by identifying different synonyms and word senses for each concept in both metamodels. These synonyms can be found using any lexical database like WordNet [17]. Set of synonyms, called synSet, should be formed for each concept of both metamodels. Each concept in one metamodel and all of its synonyms is then compared with each concept and all of its synonyms of the other metamodel using the following equation (2). M is the matrix containing the SSM values of all synonyms for both concepts C_1 and C_2.

$$semSim(C_1, C_2) = \frac{\sum_{i=1}^{|synSet(C_1)|}\left(\sum_{j=1}^{|synSet(C_2)|} M[i][j]\right)}{|synSet(C_1)| \times |synSet(C_2)|} \tag{2}$$

3.3. Structural Analysis

In order to find the similar elements, we believe that the structure of metamodels' elements should also be compared along with their syntax and semantics. Structural similarity can be measured using different approaches proposed in the literature. Structural analysis is based on the techniques used by [13], [14]. Structural similarity of two concepts C_1 and C_2 is calculated based on the similarity of their structural neighbors. Structural neighbors of a concept C_1 are parent(C_1), siblings(C_1) and immediate Child(C_1). The structural similarity is obtained as a function of partial similarities by using the following equation (3).

$$structSim(C_1, C_2) = parentSim(C_1, C_2) \times parentCoef + siblingSim(C_1, C_2) \times siblingCoef + immChildSim(C_1, C_2) \times immChildCoef \tag{3}$$

Each of the coefficients (parentCoef, sibCoef and immChildCoef) must have value between 0 (zero) and 1 (one). Consequently the parentCoef + sibCoef + immChildCoef = 1. These partial similarities are then calculated by mean values of all the neighboring concepts. For example, the similarity between the ancestors of two concepts C_1 and C_2 can be calculated using the following equation (4).

$$parentSim(C_1, C_2) = \frac{\sum_{i=1}^{|parents(C_1)|}\left(\sum_{j=1}^{|parents(C_2)|} M[i][j]\right)}{|parents(C_1)| \times |parents(C_2)|} \tag{4}$$

M is the matrix containing SSM values for all concepts in parent set of both C_1 and C_2. The next step is to synthesize all these similarities and filter out the candidate concepts that can be merged. The overall similarity is then calculated for all pair of concepts by using the following equation (5).

$$Sim(C_1, C_2) = synSim(C_1, C_2) \times synCoef + semSim(C_1, C_2) \times semCoef + structSim(C_1, C_2) \times structCoef \tag{5}$$

Table 1: SSM and Overall Similarity of ArchiMate and BPMN

ArchiMate	**BPMN**	**SSM**	**Overall**
Data Object	Data Object	1	0.6
Artifact	Artifact	1	0.63
Business Actor	Business Rule Task	0.35	0.21
Business Role	Partner Role	0.5	0.3
Business Collaboration	Collaboration	0.30	0.18
Location	Association	0.5	0.31
Business Event	Boundary Event	0.5	0.3
Business Service	Business Rule Task	0.5	0.3
Business Interaction	Business Rule Task	0.44	0.26
Business Process	Sub Process	0.36	0.21
Business Interface	Interface	0	0

The synthesized list of concepts, that have values of overall similarity greater than the threshold value (set by the analyst), represents the correspondences of mapping points in both metamodels. Values of coefficients (e.g. synCoef) in equation (5) have the same conditions as the coefficients in equation (3) and are to be decided by the analyst and depends upon the importance (for the analyst) of some similarity measure. Table 1 presents the SSM and overall similarity between conceptual elements of ArchiMate and BPMN.

3.4. Ontological Analysis

The purpose of the ontological analysis is to critically evaluate the concept list (Table 1) produced at the end of the above analysis. This evaluation is based on the definitions of those concepts as specified in their metamodels. We name this evaluation as checking ontological semantics which we believe is different from the linguistic semantics. Different types of ontological semantics needs to be evaluated between a pair of conceptual elements. Here, the objective was to check the concepts that are not the same but do have some degree of similarity in these metamodels. For instance, Business Actor concept from ArchiMate was matched to the Business Rule Task of BPMN, the results showed that these concepts having 50% SSM and 21 % overall similarity (Table 1).

While the ontological analysis of the above mentioned two concepts reveals that the Business Actor concept from ArchiMate is similar to what is defined as Participant in BPMN. Further, here the intention was to investigate and identify those concepts that seem similar but were not identified in the syntactic and semantic checking because of the way they are named. The results of ontological analysis are presented in Table 2. Table 2 represents the concepts from both metamodels that were identified as mapping points between them for interoperability after ontological analysis. These concepts can be used to either merge or map the both metamodel. An example of such merger has been shown in Fig. 4. Fig. 4 presents these concepts as merged and mapped into a single or hybrid metamodel. This draws our attention to the opportunity of merging various metamodels into a consolidated metamodel.

Fig. 4 presents the merged metamodel for ArchiMate and BPMN as an example. The elements presented as filled boxes represent the concepts of ArchiMate and BPMN that were merged as a single concept while the unfilled boxes represent the concepts of BPMN that are mapped to other concepts to maintain the structure of original metamodels.

Table 2. Candidate Concept between ArchiMate and BPMN

ArchiMate	**BPMN**
Business Actor	Participant
Business Role	Partner Role
Business Collaboration	Collaboration
Business Interface	Interface
Location	Pool
Business Event	Event
Data Object	Data Object
Business Service	Activity
Business Process	Activity
Artifact	Artifact

For example, Business Actor of ArchiMate was matched with the Participant of BPMN, so is represented in unified metamodel as a single concept Business Actor. It is to be noted that the Participant in BPMN was a sub-class of Base Element, so Business Actor in the unified metamodel is shown as a sub-class of Base Element. Similarly, Collaboration of BPMN was merged with the Business Collaboration of ArchiMate.

Collaboration in BPMN has composition relationship with Artifact, so Business Collaboration in the unified metamodel preserves the same relationship (composition) with Artifact.

The merged metamodel (Fig. 4) has 14 conceptual elements, that is apparently more than both partial metamodels of ArchiMate and BPMN, presented as Fig. 1 and Fig. 2 respectively, but this increase in the number of elements in the unified merged metamodel actually decreases the size and complexity of both metamodels when used together as a hybrid standard. In the next section, we will discuss analysis results and quality aspects of this approach including the size and complexity measures.

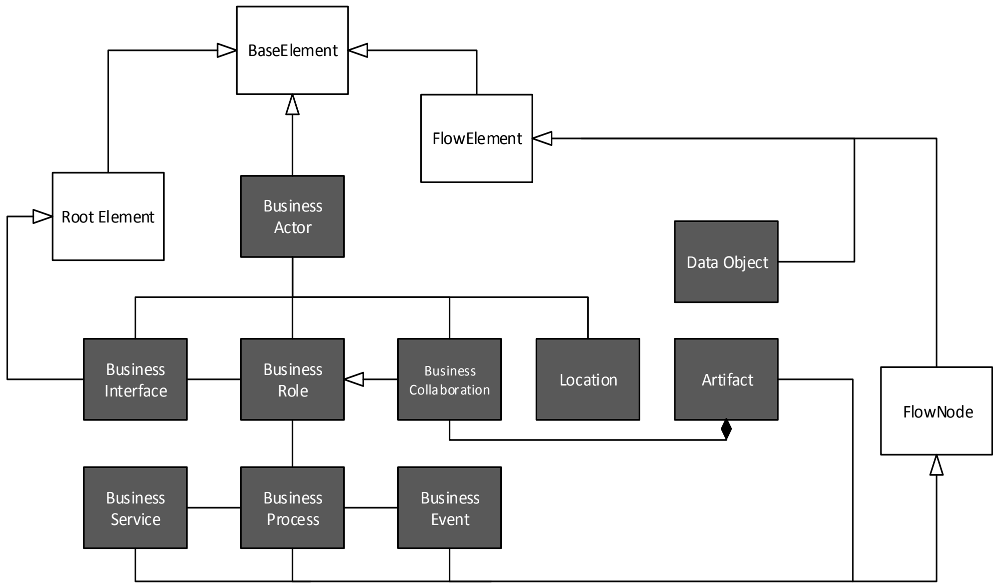

Fig. 4. The merging of archimate (business layer) and BPMN into a single metamodel.

4. Discussion

This section discusses the results and quality assessment of our approach. Our major focus regarding the quality is to analyze the quality of the review and actual matching results. We have used the standard measures of precision and recall, stemming from the field of information retrieval. These measures are based on the notion of true positives (*tp*), false positives (*fp*) and false negatives (*fn*). True positives (*tp*) are values which were correctly matched while false positives (*fp*) are those values which were incorrectly identified. These measures are defined in [15] as.

$$precision = \frac{|tp|}{|tp|+|fp|} \tag{6}$$

$$recall = \frac{|tp|}{|tp|+|fn|} \tag{7}$$

$$recall = \frac{|tp|}{|tp|+|fn|} \tag{8}$$

Calculating these values based on the results in Table 1 and Table 2 reveals that *P*= 0.54, Recall = 0.6 and FMeasure = 0.56.

Another measure that shows the effectiveness of this interoperability is calculated using size and complexity of both metamodels and the resulting metamodel with merged concepts. These results were calculated using the size and complexity measures proposed by [16]. The rational for using these measures was that these have already been used to measure the size and complexity of some process models [17] and [18]. We have used these metrics to calculate the size and complexity of both metamodels before and after merging as a single metamodel. Rossi and Brinkkemper [16] have presented a set of 17 metrics, based on the metamodels vocabulary (i.e. concepts and properties). Specifically, these are the aggregate measurement of object types, their attributes/properties, their relationships with each other and roles. Formally, a model M of a technique T is defined as M = O, P, R, X.

O is a finite set of object types. An object is defined as a thing that exists independently. For example, Class, Association and Classifier are different object types. P is a property (attribute) of an object. R is a finite set of relationship types. A relationship is an association between two or more object types. Different relationship cardinalities are treated differently. For example Composition 1 to 1 and Composition 1 to * are treated as two relationship types, as suggested by Rossi and Brinkkemper [16]. X is a finite set of role types, a role being the name of the connection between an object type and its association.

The variables R and P represent mappings from role types to relationship types and objects types; and from non-property types to property types, respectively although neither is used for the metrics calculations. The complexity of a metamodel is calculated using the following equation 9.

$$C(M)\sqrt{n(O_T)^2 +} n(P_T)^2 + n(R_T)^2 \tag{9}$$

Table 3 represents the values calculated using these metrics for both BPMN and ArchiMate and then the values of the merged metamodels.

Table 3. Complexity Comparison of ArchiMate, BPMN and Merged Metamodel

	n(OT)	n(PT)	n(RT)	C(M)
BPMN	60	109	19	125.86
ArchiMate	34	0	12	36.04
Combined	94	109	31	161.9
Merged	84	109	23	139.5

Table 4. Complexity Comparison of ArchiMate and BPMN Partial Metamodels (Used in this paper, Fig. 1 and Fig. 2) with Merged Metamodel (Fig. 3)

	n(OT)	n(PT)	n(RT)	C(M)
BPMN	13	0	3	10.20
ArchiMate	10	0	4	13.34
Combined	23	0	7	23.54
Merged	14	0	3	14.3

We can see that making these metamodels interoperable reduces their joint size and complexity. The third row shows the size and complexity of these metamodels when both have to be used in the course of any EA driven system development. As users have to cater with the size and complexity of both metamodels separately that results in overall more aggregate complexity (161.9). In contrast, the last row represents the values when these metamodels were merged, that reduces number of objects, properties and relationship types, hence resulting in overall low complexity (139.5).

This helps the users of metamodels ease of use, especially for novice users. Similarly, comparing the partial metamodels, presented in this paper as Fig. 1 and Fig. 2, depicts the same difference of size and complexity of these metamodels used as a combination (Row 3, Table 4) and the size and complexity of merged metamodel (row 4, Table 4). As, Fig. 1 and Fig. 2 are partial representation of these metamodels and does not entails the properties or attributes of the conceptual elements in those metamodels, so n(OT) in Table 4 is treated as 0 (zero) in both metamodels.

It has been further observed (from modelling standard application perspectives) that ArchiMate provides an overarching comprehensive set of modelling elements that support all three layers of the EA: business, application and technology. BPMN only supports some elements of the business layer in comparison to ArchiMate. BPMN does not support the application and technology layers of the EA. It is also noted in ArchiMate that it is appropriate for the high-level modelling of the business processes, at the enterprise level, within the overall business architecture context. However, detailed business process modelling requires much richer elements. Therefore, BPMN can be used for detailed business process modelling. The scope of BPMN is strictly limited to the modelling of business processes of the business layer. This means that other aspects of the business, which are not in the scope of BPMN, can be modeled by using ArchiMate. This draws our attention to the need for creating a unified or merged hybrid adaptive EA modelling approach by integrating or merging ArchiMate and BPMN for a particular organizational context.

5. Adaptive Enterprise Architecture Modelling

The proposed adaptive EA modelling approach is discussed here with the help of a “University Book Bank” example case study adapted from [19]. The book bank purchases old books from different sellers and then re-sells them to their customers via their online web portal. In order to demonstrate the integrated hybrid ArchiMate and BPMN modelling approach, we have analysed the business architecture of the “University Book Bank” case study and identified the following four business processes: Receive Books, Make Payment, Re-sell Books and Receive Payment.

- Book bank “Receive Books” from a supplier so that they can re-sell the books to their customers.
- Book bank “Make Payment” to their suppliers once the book is sold to the customer and customer has paid the price.
- Book bank “Re-sell Books” to their customers via bank web portal.
- Book bank “Receive Payment” online from the customer via credit card once the sale is confirmed.

ArchiMate metamodel can be used to model the business processes and their interactions (business collaboration) at the high level in the overall business architecture. Each business process (e.g. “Receive Books”, “Make Payment”, “Re-sell Books”, and “Receive Payment”) identified and modelled at the high level, can be further modelled and detailed by using the BPMN metamodel elements. Additionally, one may transform the BPMN process model to some other executable and non-executable formats [20]. This indicates the appropriate merger or integration of ArchiMate and BPMN for business modelling. Similarly, other relevant modelling approaches or metamodels can be reviewed with a view to support hybrid adaptive EA modelling [21], [22]. This marks the need for further investigation in this important area of research.

6. Conclusion

This paper presented the review and comparison of the ArchiMate and BPMN modelling standards from their syntax, semantics and structural perspectives; and proposed the development of a unified hybrid adaptive EA modelling approach. The analysis presented in this paper will help enterprise architect researchers and practitioners to understand the scope and applicability of ArchiMate and BPMN modelling standards; and make an informed decision about the adoption and tailoring of these modelling standards for their local context. In our further research, we are aiming to develop a new unified modelling language that can be used for modelling EA artifacts both at the high enterprise level and fully detailed level for a particular context.

References

[1] Purchase, V., Parry, G., Valerdi, R., Nightingale, D., & Mills, J. (2012). Enterprise transformation: Why are

we interested, what is IT, and what are the challenges. *Journal of Enterprise Transformation*.

[2] Gill, A. Q. (2014). Hybrid adaptive software development capability: An empirical study. *Journal of Software*, *9(10)*, 2614-262.

[3] D. Kutnick. Aligning and Measuring Agility. Retrieved, 2006, from http://www.gartner.com/it/products/podcasting/asset_143240_2575.jsp.

[4] Sellers, B. H. & Qumer, A. (2007). Using method engineering to make a traditional environment agile. *Cutter IT Journal*, *20(5)*.

[5] The Open Group. Retrieved, 2012, from http://theopengroup.org/archimate/downloads.htm.

[6] Documents Associated with Business Process Model and Notation (BPMN) Version 2.0. Retrieved, 2011, from http://www.omg.org/spec/BPMN/2.0/

[7] OMG, Documents associated with UML Version 2.0. Retrieved, 2005, from http://www.omg.org/spec/UML/2.0/, 2005.

[8] Carrera, Á., Iglesias, C. A., & Garijo, M. (2014). Beast methodology: An agile testing methodology for multi-agent systems based on behaviour driven development. *Information Systems Frontiers*, *16(2)*, 169-182.

[9] Qureshi, M. A. (2012). Interoperability of software engineering metamodels.

[10] Beydoun, G., Low, G., Sánchez, F. G., García, R. V., & Béjar, R. M. (2014). Identification of ontologies to support information systems development. *Information Systems*, *46 (12)*, 45-60.

[11] Beydoun, G., Sánchez, F. G., Torres, C. M. V., Lorca, A. A. L., & Béjar, R. M. (2013). Providing metrics and automatic enhancement for hierarchical taxonomies. *Information Processing and Management*, *49(1)*, 67-82.

[12] ernandes, F. F., & Song, M. (2014). UML-checker: An approach for verifying UML behavioral diagrams. *Journal of Software*, *9 (5)*, 1229-1236.

[13] Chukmol, U., Rifaieh, R., & Benharkat, N. A. (2005). EXSMAL: EDI/XML semi-automatic schema matching algorithm. *Proceedings of the Seventh IEEE International Conference on E-Commerce Technology (CEC'05)*.

[14] Sousa, J., Lopes, D., Claro, D. B. *et al*., A step forward in semi-automatic metamodel matching: algorithms and tool. *Enterprise Information Systems, Lecture Notes in Business Information Processing*.

[15] Maedche, S. (2001). Comparing ontologies-Similarity measures and a comparison study, institute AIFB, University of Karlsruhe, Internal Report, 2001.

[16] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, *10(8)*, 707-710.

[17] Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, *38(11)*, 39-41.

[18] Salton, G. (1987). Expert systems and information retrieval. *SIGIR Forum*, *21(3-4)*, 3-9.

[19] Rossi, M., & Brinkkemper, S. (1996). Complexity metrics for systems development methods and techniques. *Information Systems*, *21(2)*, 209-227.

[20] Siau, K. & Cao, Q. (2001). Unified modelling language: A complexity analysis. *Journal of Database Management*, *12(1)*, 26-34.

[21] Recker, J. C., Muehlen, M., Siau, Z. K. *et al*. *Measuring method complexity: UML versus BPMN*. *Proceedings of the* 15*th Americas Conference on Information Systems*, San Francisco.

[22] Gill, A. Q., & Bunker, D. (2012). SaaS requirements engineering for agile development. Agile and Lean Service-Oriented Development: Foundations, Theory and Practice, IGI, 2012.

[23] Jian, H. Y., & Wen, S. (2010). Transformation of BPMN diagrams to yawl nets. *Journal of Software*, *5(4)*, 396-404.

[24] Li, M., Zhou, J., & Liang, X. (2014). Modeling and description of organization-oriented architecture.

Journal of Software, *9(4)*, 867-872.

[25] Elasri, H., & Sekkaki, A. (2013). Semantic integration process of business components through Ontology alignment. *Journal of Software*, *8(12)*, 3016-3025.

Asif Q. Gill is a TOGAF 9 Certified Enterprise Architect, lecturer and researcher at the School of Software at the University of Technology, Sydney. He specializes in software-intensive agile enterprise architecture and engineering practices, tools and techniques. He is an experienced author, coach, consultant, educator, researcher, speaker, trainer and thought leader. He has extensive experience in both agile, non-agile, cloud and non-cloud complex software development environments, displaying a deep appreciation of their different perspectives in a number of IT-enabled business process improvement and transformation industry projects of varying sizes.

Muhammad A. Qureshi is a researcher at the School of Software at the University of Technology, Sydney. He specializes is metamodels in software engineering and their interoperability. His area of research includes tools and techniques in the domain of modelling.