# A review and comparison of service e-contract architecture metamodels

Ali Braytee[1], Asif Qumer Gill, Paul J. Kennedy[1] and Farookh Khadeer Hussain[1]

[1]School of Software, Center Quantum Computation and Intelligent Systems, University of Technology Sydney, Sydney, New South Wales 2007, Australia
{ali.braytee@uts.edu.au,asif.gill@uts.edu.au,paul.kennedy@uts.edu.au,farookh.hussain@uts.edu.au}

**Abstract.** An adaptive service e-contract is an electronic agreement which is required to enable adaptive or agile service sourcing and provisioning. There are a number of e-contract metamodels that can be used to create a context specific adaptive service e-contract. The challenge is which one to choose and adopt for adaptive services. This paper presents a review and comparison of well-known e-contract metamodels using the architecture theory. The architecture theory allows the analysis of the e-contract metamodels using a three-dimension analytical lens: structure, behavior and technology. The results of this paper highlight the metamodels structural, behavioral and technological differences and similarities. This paper will help researchers and practitioners to observe whether the existing e-contract metamodels are appropriate to the adaptive services or if there is a need to merge and integrate the concepts of these metamodels to propose a new unifying adaptive service e-contract metamodel. This paper is limited to the number of compared metamodels.

**Keywords:** Adaptive e-contract, adaptive service, contract metamodel, foundational ontology.

## 1 Introduction

In the modern service economy, focus shifts from the traditional upfront fixed service contract to the adaptive or agile service e-contract. This is because, the context of the modern enterprise encourages the use of agile services and e-contracts to deal with complex dynamic business requirements [1]. In service dominant logic (S-D logic), service is the application of competence (knowledge and skills) to benefit other parties [2]. The interaction between the parties in the service ecosystem may lead to value co-creation which means that all the engaged models earn value. The value in service science is the refinement of the system which co-creates the benefits or value to all involved parties. Traditional service provisioning is based on an upfront fixed contract, which is not appropriate and aligned to the modern business requirements where a dynamic response is required to deal with changing service demands [3]. The emerging service science body of knowledge [4] suggests shifting focus from the traditional service delivery to a service offering through voluntary and dynamic interactions between service systems. A service system could be an organisational function or capability that offers services to other service systems. The adaptive enterprise service system theory states that a modern enterprise is an ecosystem of adaptive or agile service systems that voluntarily interact with each other and exchange adaptive or agile services for value co-creation through a dynamic or adaptive contract [3]. The type of dynamic interactions between service systems could be either bilateral relations which consist of service provider and service consumer, or multilateral relations between many service systems or parties [5]. The dynamic interactions between service systems could be enabled

through a dynamic adaptive service contract (e-contract) as opposed to an upfront fixed service contract.

A contract is an agreement between two or more entities to create a business relation and to define a set of rules such as obligations, permissions and prohibitions for the business process [6]. A traditional contract is a static in nature which includes a static description of the business expectations of both parties. Furthermore, a traditional contract is not able to govern the relation between dynamic, ever-changing and adaptive services and it will not be able to monitor the clauses of such dynamic contracts. An e-contract is an electronic agreement between the service parties. It is composed of a set of entities and relationships in order to satisfy the service governance and compliance. In addition to the structural depiction, an e-contract has the ability to work with dynamic events and actions to handle the exceptions once the contract clauses are violated by either party. An e-contract can be established through a metamodel. There are a number of e-contract metamodels such as $ER^{EC}$, contract enforcement among others. Organisations are often unsure about the choice of a metamodel for establishing a context-specific e-contract. There is a need to understand the similarities and differences between the different metamodels. This study reviews the most well-known metamodels are the following: "$ER^{EC}$ Metamodel", "Three-layered e-contract enforcement metamodel" and "Secure e-contract metamodel" by using the architecture theory (ISO/IEC 42010) in order to understand the scope and usability of the available metamodels. This work is significant because it helps practitioners and researchers in finding the shortcomings of the existing e-contract metamodels and serves as a knowledge base for developing a new unifying adaptive service e-contract metamodel. Additionally, It clarifies whether the existing e-contract metamodels are appropriate for adaptive service system or there is a need to propose a new adaptive service e-contract. The contribution of this work to the existing literature body is that, it is the first work of that systematically reviews and compares the most well-known contract metamodels. This paper is organised as follows. Section 2 presents the research method. Section 3 discusses the analysis. Section 4 presents the discussion before concluding in section 5.

## 2 Research method

This review paper applies the qualitative review approach [7] which is appropriate to address the research question in hand. The research process is two-fold. Firstly, we set the review criteria based on the architecture theory (ISO/IEC 42010). The review criteria provided us with a three-dimension analytical lens: structure, behavior, and technology. Secondly, we applied the review criteria to the most well-known e-contract metamodels identified from the literature which are the following: "$ER^{EC}$ Metamodel", "Three-layered e-contract enforcement metamodel" and "Secure e-contract metamodel". The review criteria helped us to systematically analyse and compare each contract metamodel from the perspectives of the three dimensions. The structure perspective allowed us to analyse and compare the properties and relationships of the elements available in the three metamodels. The study of structure in its own right is not sufficient; we also need to analyse the behavior of the structure. The behavioral perspective allowed us to analyse and compare the semantics of the elements available in the reviewed metamodels. Finally, the technological perspective helped us to analyse and compare the supporting technology in the metamodels. This research approach provided a foundation for further research in the possible integration or merger of the well-known contract metamodels to create a more comprehensive unifying contract metamodel for agile or adaptive services. In the remainder of the paper, we will use terms electronic contract, e-contract and contract interchangeably.

## 3 Analysis: metamodels

In this section, we will analysis the reviewed contract metamodels.

### 3.1 $ER^{EC}$ metamodel

The $ER^{EC}$ metamodel has evolved over a period of time since the first publication of the foundation in 2001. It has been continuously updated, the most recent updated version of the metamodel being published in 2013 [8]. The $ER^{EC}$ model is shown in Figure 1. It is based on the entity relationship model that captures the main concepts/entities of the contract and transforms the conceptualizations into workflows. This definition was used in the early versions of this metamodel, however its architecture has evolved over a period of time. Despite changes and evolution in the architecture of the $ER^{EC}$, the fundamental structure of the e-contract model remains the same [9]. In the most recent version of the $ER^{EC}$ model [8], the authors proposed the metamodeling approach which is able to define template for contracts.

This metamodel presents the main concepts of the e-contract. The evolved $ER^{EC}$ metamodel organises the most abstract entities as core entities and defines a contract template. The template or model instantiates the $ER^{EC}$ metamodel that includes the domain constraints and relationships for the e-contract application [8]. The use of the template allows the e-contract model to evolve. The e-contract template approach will be further discussed in this paper in the next section (Three-layered e-contract enforcement metamodel section). Furthermore, the template has several features such as adding new concepts into the model and reflecting the most real entities by modeling these concepts in the domain application. In addition, the latest versions of the $ER^{EC}$ have proposed a meta-modeling approach to respond to the events that trigger run-time changes to the e-contract or exceptions due to violation in an existing contract clause [8]. Now, the $ER^{EC}$ metamodel architecture will be analysed from its structure, behavior and technology perspectives. The structure of the $ER^{EC}$ e-contract composed of the following core entities: parties, activities and clauses. The $ER^{EC}$ e-contract must have two or more parties to initiate a contract and can have nested subcontracts. A subcontract is a contract which is created by one of the involved parties and may contain different parties, activities and clauses. A contract has a set of activities which are divided into tasks. The activities control the behavior of the e-contract. Each activity is used by the parties. The last core entity is the contract clause. An e-contract has a list of clauses and every clause can refer to other clauses. A clause is fulfilled when a relevant activity is successfully executed. Furthermore, clauses are considered as constraints or rules which may or may not be linked to the activities [9]. The other entities of the e-contract are financial entities such as payment and budget. The former captures the amount of service usage and the way of paying the other party, and the latter describes the maximum amount of money that the customer is obliged to pay. A contract has a specific period of time. More entities are declared in $ER^{EC}$ such as role and exception [9]. Role depicts the position of the party which is linked to the parties via many-to-many relationships, and exception is executed once a clause is violated. Finally, an event can trigger action(s) related to an e-contract. An event can be predictable or unpredictable, and can be initiated at any time during the contract's execution. There are a number of events that may occur during contract execution, such as a contract database event, a temporal event or an external event. A contract event may refer to a situation such as add role, add party, or start payment. Furthermore, a temporal event can be specified as a time-based workflow, such as a payment frequency [10]. In summary, contract structure describes the contract elements and their relationships (see in Figure 1).

Further, we need to analyse the behavior of the $ER^{EC}$ e-contract structure, which is often linked to the contract run-time environment. The behavior of the e-contract model can be

described and analysed in terms of contract activities or workflows. The $ER^{EC}$ entities are connected to the workflow activities that execute the e-contract. The parties are related to roles and clauses to events that may trigger workflow activities or actions. The $ER^{EC}$ workflow system uses the activity commit diagram (ACD), which is a list of atomic activity transactions. These transactions are executed sequentially. Krishna et al. (2004) states that "ACD ensures the consistent, atomic and durable execution of the activities". They specify a monitoring process to check the activity status and log messages. A contract activity can be rolled back in cases of failure and a message can be logged to report failures. Roll-back is not the only way to handle contract activity failure situations. There are a number of other options which can be applied to handle a failure situation, such as failure compensation, alternative activity, time-based retry or re-execution. $ER^{EC}$ specifies the ECA (event-condition-action) concept in order to monitor the events in the e-contract model, however this procedure is based on a manual analysis process. It searches for a specific activity or behavioral-related words to create the conditions and actions, such as if-else, contract violations, among others [10]. In summary, the review of the $ER^{EC}$ metamodel indicates that the $ER^{EC}$ offers the ability to adapt to run-time changes. These changes may refer to contract updates, exceptions or failures. The following four behavioral operations related to the $ER^{EC}$ e-contract model are identified in this review:

i Adapt: This operation is used if the change is small, such as adding or modi-fying clauses in the e-contract and it doesn't modify the structure of the e-contract.
ii Migrate: This is used to address the new requirements related to the existing e-contract by instantiating a new model, such as a subcontract.
iii Merge: This merges the current model with the new instantiated model.
iv Build: This is used when there is a need to change the structure of the e-contract model by adding new concepts [8].

The analysis of the $ER^{EC}$ structure and behavior provides important insights. However, the conceptual $ER^{EC}$ structure and behavior implementation requires technological support. This analysis is further extended to include the technology perspective. The $ER^{EC}$ specifies software components that implement the e-contract metamodel. A relational database is one such component that is used to store the structure of the e-contract, rules and workflows. Enterprise JavaBeans (EJB) components are used to support the presentation and business logic layers - similar to the model-view-controller architecture design patter (a.k.a. MVC). The web service server is implemented to allow interaction among inter-organizational parties. Finally, the E-ADOME Workflow Engine is used to support workflow management [10]. It is clear from the analysis that the $ER^{EC}$ is not simply a conceptual metamodel, rather it specifies technological support, which highlights its practical applicability and usability.

### 3.2 Three-layered e-contract enforcement metamodel

E-contract enforcement architecture is composed of three layers: document layer, business layer and implementation layer. The document layer describes the clauses of the e-contract. The business layer or e-contract enforcement layer presents the business rules in the *event-condition-action* form. Finally, the implementation layer refers to the supported technologies that are used to implement the web services for e-contract enforcement. Furthermore, document and business layers are represented by the e-contract template and actual metamodel, respectively. The analysis of e-contract enforcement indicates that the dynamic business interactions are not addressed in the e-contract clauses and there is a need to implement e-contract enforcement in order to perform monitoring process for the e-contract clauses [6]. In this section we analyse the architecture structure, behavior and technology of the e-contract enforcement metamodel.

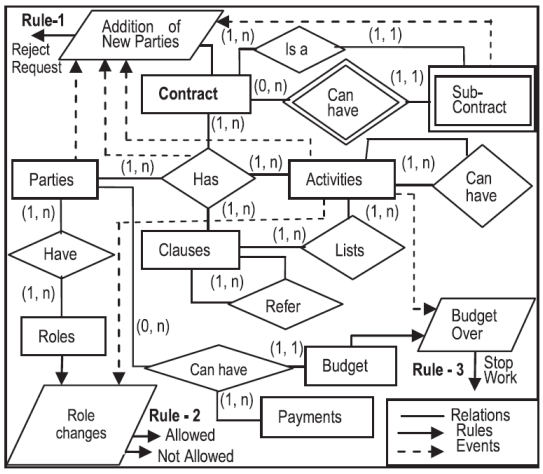


Fig. 1: $ER^{EC}$ metamodel for the e-contract (Sourced from [8])

E-contract enforcement architecture is organised into three layers. The document layer is the e-contract metamodel which comprises the e-contract template entity. The e-contract template is composed of a list of fixed entities and linked to the e-contract entity to refine the proposed template variables in the contract clauses. The e-contract entity also involves two or more parties which is related to two more entities: template variable and accepted value. A template variable is a set of variables that need to be refined during negotiation between the parties. Negotiation is an important concept of adaptive services. The result of the negotiation process will define the mutually agreed values, which can be transformed into the accepted values. The e-contract template, shown in Figure 2, is composed of a set of contract clauses. Similar to the $ER^{EC}$, contract clauses can refer to each other and are nested clauses. Every contract clause has zero to many template variables. A contract clause is divided into three different types: obligation, prohibition and permission. Obligation describes the clauses that must be fulfilled (must do clauses), and prohibition presents the action that must not be performed. Finally, permission clauses depict permissible actions or behaviors [6]. The e-contract enforcement metamodel is shown in Figure 3. It comprises the contract clause, which is related to the rules that enforce the clauses. A rule entity is related to event, condition and action. An event has three sub-types, which are business action invocation, exception and temporal event. There are also two types of events: external and internal events. An event triggers the rule. The rule evaluates the condition, which is a logical expression based on a business entity. If the condition is satisfied, then the enforcement action is executed. Typically, an action is structurally recursive; therefore, it has sub-actions and tasks. The remaining entity in a meta-model is a party which owns the business entity and communicates with the event to publish and subscribe to the events. The behavior of the contract enforcement depends on the ECA rules. The model presents the business action invocation object, which can be related to the temporal events (e.g. deadline or contract expiry date). For example, in the obligation enforcement clauses, it is obligatory to fulfil a certain clause in a specific time frame. If the condition is not satisfied by that time, it will raise an exception. The UML modeling tool can be used to design the e-contract metamodel and models. The e-contract metamodel is then implemented by using web services technology.

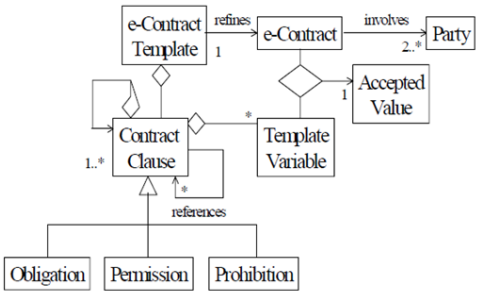


Fig. 2: A metamodel of an E-Contract Template in UML (Sourced from [6])

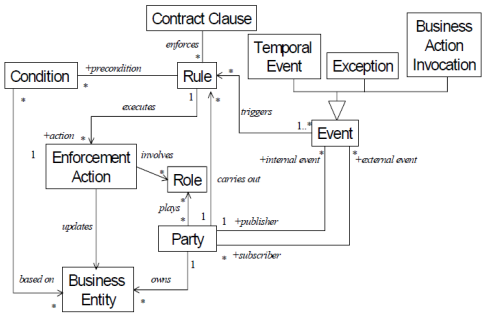


Fig. 3: A metamodel of contract enforcement in UML (Sourced from [6])

### 3.3 Secure e-contract metamodel

This section analyses the secure e-contract metamodel, which implements the security concerns of e-contract. Due to the nature of e-contract communication across organizations over the Internet, there is a need to address the contract security concerns of the involved contract parties such as authentication, integrity and non-repudiation. The appropriate handling of these concerns will help prevent the e-contract from being tampered or altered. The objective of the secure e-contract metamodel is to enhance the security and governance of the e-contract [11]. In this section, we analyse the secure e-contract metamodel from structure, behavior and technology perspectives. The secure e-contract metamodel structure is organised into seven entities. The core entity of the secure e-contract metamodel is the contract. Figure 4 shows contract attributes, such as contract duration and signature status. It also shows several relations to other entities. A secure contract has four sub-types: clause, party, exchange value and digital signature. A clause entity can refer to different clauses. A contract must have at least one clause. A party is one of the essential entities in the contract. A contract must have at least two parties. A party is linked to the digital signatures [11]. A party can have many digital signatures related to one or more contracts; however every digital signature is unique. Furthermore, a digital signature can refer to only one contract; however the contract may contain many signatures. The last sub-type of contract is the exchange value which represents the total cost and currency of the contract. The exchange value has a sub-type called service. A contract must have one and only one exchange value, and the exchange value can contain one or more services. The last entity in the secure e-

contract metamodel is the activity, which is a sub-type of clause where every clause must have one or more activities [11]. The secure e-contract metamodel authors propose the ECRC web server to manage the e-contract formation. The ECRC system acts as a third party organization to offer services such as storing e-contract templates, downloading e-contract templates, querying existing contract templates and uploading new valid contracts. The process of creating an e-contract between the parties is as follows: party A (initiator) downloads an e-contract template that suits their requirements; the party updates the content, signs and sends the draft contract to the other party B. Party B checks the content and confirms the signature, then signs the contract. The two parties send a copy of the valid contract to ECRC to save it [11]. The secure e-contract can be modeled using the UML modeling tool as shown in Figure 4 below.

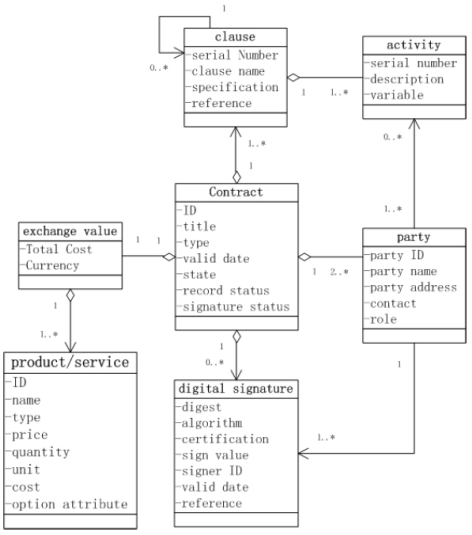


Fig. 4: UML model for secure e-contract metamodel (Sourced from [11])

## 4 Discussion

In the previous section, we analysed three well-known e-contract metamodels. In this section, we analyse, compare and discuss the results, which are summarised in Table 1. The objective of this review and comparison is to understand and observe the similarities and differences between the e-contract metamodels from structure, behavior and technology perspectives and also to identify opportunities for their merger and interoperability for an adaptive service contract.

### 4.1 Structure

The review of the metamodels structure indicates that all three metamodels have similar entities or concepts. Additionally, these entities have similar semantics. However, sometimes they have different names or spellings. We observe that the similar entities between all the metamodels are *'parties'* or *'party'* and *'clauses'* or *'clause'*. Furthermore, the *'activities'* or *'activity'* entity exist

Table 1: Review and comparison of e-contract metamodels

| | $ER^{EC}$ metamodel | Three-layered e-contract | Secure e-contract |
|---|---|---|---|
| Structure | Core entities: Party, activities and clause. Secondary entities: Financial entities, roles and subcontract. | Core entities: Clause, party and accepted value. Secondary entities: Event, rule, condition and enforcement action | Core entities: Clause, party, activity, exchange value and digital signature. |
| Behavior | Activities are mapped into contract workflows. Tasks in activity can be committed or rolled back. Monitoring through log files. | Using e-contract template to create customized contracts. Event-Condition-Action rules are used for monitoring. | ECRC is a management system that shows the steps to create a secure e-contract from templates. |
| Technology | Enterprise JavaBeans. Web services. E-ADOME for workflows. | Web services. UML. | ECRC system. UML |

in the $ER^{EC}$ metamodel and secure e-contract model, however it is not explicitly mentioned in the three-layered e-contract model. The three-layered e-contract model is divided into two metamodels which are: *'e-contract template'* and *'actual metamodel'* of contract enforcement. The event entity in the metamodel of contract enforcement has three subtypes which are: business action invocation, exception and temporal event. By applying the ontological analysis technique on these subtypes, we observed that business action invocation can be captured as an activity entity. Despite the similarities, Table 1 presents the differences between the contract metamodels. The $ER^{EC}$ metamodel proposed financial entities such as payment and budget. Payment is the price of the service and budget is the maximum amount of money reserved for the service. Financial entities considered as an important concepts in the contract. However, surprisingly, they are not presented in the other metamodels of this study. Furthermore, the $ER^{EC}$ metamodel presents the roles and subcontract entities; the former captures the roles of the parties and the latter describes the nested subcontracts as extensions of the main contract. A subcontract can be used by the service provider to delegate some tasks to the third party. On the other hand, the metamodel of contract enforcement in the three-layered e-contract presents several entities such as Event-Condition-Action, which are not modeled in the other metamodels reviewed in this study (see Table 1). Finally, the secured e-contract metamodel proposed a digital signature to ensure the integrity of the e-contract, which seems to be overlooked by the other two metamodels reviewed in this study. The structural review and comparison of the e-contract metamodels highlights the similar and different entities of the metamodels at a structural level. This review and comparison can be used as a baseline to further create a unified and richer structure of an e-contract for adaptive or agile services.

### 4.2 Behavior

This section discusses the behavioral similarities and differences among the three metamodels. The most common “behavioral" aspect of the metamodels is the template. The reviewed meta-

models agree that a new e-contract can be derived from a ready-made template. This template contains the core concepts. The new e-contract derived from the template customizes the available entities and adds new concepts related to the e-contract instance. Despite the agreement between these metamodels for using the template as a basis for generating new e-contracts, the implementation is different in each metamodel. The $ER^{EC}$ metamodel maps the contract template activities to workflows. Each activity contains at least one task and each task is able to commit or roll-back the operation. Also, it uses the log messages for monitoring purposes. However, the three-layered metamodel uses the ECA rules to monitor the actions of the e-contract. The secure e-contract also discusses the behavioral aspect through contract templates.

### 4.3 Technology

Finally, this section discusses the technological similarities and differences between the three metamodels. Table 1 shows that $ER^{EC}$ and three-layered metamodels have used web services to access the operations of the e-contract interfaces. Also, three-layered and secure e-contract metamodels have used the ECRC system and UML modeling tool to design the e-contracts. Furthermore, $ER^{EC}$ have used the enterprise JavaBeans for applications and E-ADOME for workflow management. All three metamodels discuss technological support, which is important to put the conceptual structure and behavior of the metamodels in practice.

In summary, the analysis and comparison of these metamodels indicate that they differ in defining some core concepts in contract. Further, it is noticed that surprisingly, the reviewed metamodels lack a theoretical underpinning and are not based on any theoretical or upper ontological model [12]. The similarities, differences and lack of a theoretical underpinning mark the need to create more comprehensive metamodels by incorporating elements from different metamodels and by using a solid foundation ontological theory. In order to address this issue in our current research, firstly, the core concepts of the future e-contract model will be collected from the three existing e-contract metamodels. These core concepts can be extended to contain other features of the different metamodels, such as the financial entities, security and exceptions. In the second stage, we will perform further analysis on the proposed concepts using the foundation ontology such as the Unified Foundational or upper-level ontology [12], the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [13], the Basic Formal Ontology (BFO) [14] and others. The term upper ontology is defined as "a high-level, domain-independent ontology, providing a framework by which disparate systems may utilize a common knowledge base and from which more domain-specific ontologies may be derived" [15]. Of course, deriving the specific e-contract domain ontology based on the universal concepts of the foundation ontology will provide a strong theoretical under-pinning and accuracy in the context of correct modeling. Furthermore, the e-contract concepts will inherit important features such as interoperability. Therefore, the proposed e-contract metamodel will be composed of well-analysed core concepts and templates. Also, it is important to clarify that template metamodel is an abstraction of the real concepts that specifies the abstract contract concepts. The fundamental concepts and their relationships can be derived from universal objects of the foundation ontology. Hence, based on the analysis of this study, it can be observed that the existing major e-contract concepts are slightly different from each other and they lack from a theoretical and empirical research foundation. A foundational ontology-driven approach, as proposed here, is one systematic way of researching and creating a metamodel and related template for adaptive services. It is important to mention that this review paper is limited to only three e-contract metamodels. These metamodels are considered to be the most popular e-contract models in service contract research. In future, we will include more e-contract metamodels to further extend our studies in this important area of research.

## 5 Conclusion

This paper presents a comprehensive analysis, revision and cross-comparison of three well-known e-contract metamodels: $ER^{EC}$ metamodel, three-layered meta-model and the secure e-contract metamodel. The analysis highlighted the structural, behavioral and supported technological similarities and differences of these three metamodels. Furthermore, this paper highlighted the lack of theoretical and empirical research underpinning of these metamodels, which is important for creating a comprehensive research-based metamodel. Hence, this study identifies the need for developing a unifying e-contract metamodel using the foundation ontology theory and elements from different e-contract metamodels. Also, it provides a platform for further theoretical and practical research in this important area of e-contract meta-modeling for adaptive services in the overall context of adaptive service systems. In the future, we will develop a new adaptive e-contract metamodel which contains the most appropriate entities in the reviewed metamodels in order to be considered as a standard e-contract.

## References

1. J. Pourdehnad, G. K. Bharathy, Systems thinking and its implications in organizational transformation, in: 3rd International Conference on Systems Thinking in Management, Philadelphia, PA, 2004.
2. D. Gruhl, J. Bailey, J. Spohrer, P. Maglio, Steps toward a science of service systems, Computer (1) (2007) 71–77.
3. A. Q. Gill, Applying agility and living service systems thinking to enterprise architecture, International Journal of Intelligent Information Technologies (IJIIT) 10 (1) (2014) 1–15.
4. R. L. Martin, The design of business: why design thinking is the next competitive advantage, Harvard Business Press, 2009.
5. A. Klenk, A. Beck-Greinwald, H. Angst, G. Carle, Iterative multi-party agreement negotiation for establishing collaborations, Service Oriented Computing and Applications 6 (4) (2012) 321–335.
6. D. K. Chiu, S.-C. Cheung, S. Till, A three-layer architecture for e-contract enforcement in an e-service environment, in: System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on, IEEE, 2003, pp. 10–pp.
7. J. Corbin, A. Strauss, Basics of qualitative research: Techniques and procedures for developing grounded theory, Sage publications, 2014.
8. P. R. Krishna, K. Karlapalem, A methodology for evolving e-contracts using templates, IEEE Transactions on Services Computing 6 (4) (2013) 497–510.
9. K. Karlapalem, A. R. Dani, P. R. Krishna, A frame work for modeling electronic contracts, in: Conceptual ModelingâĂŤER 2001, Springer, 2001, pp. 193–207.
10. P. R. Krishna, K. Karlapalem, D. K. Chiu, An erec framework for e-contract modeling, enactment and monitoring, Data & Knowledge Engineering 51 (1) (2004) 31–58.
11. X. Yu, Y. Chai, Y. Liu, A secure model for electronic contract enactment, monitoring and management, in: Electronic Commerce and Security, 2009. ISECS'09. Second International Symposium on, Vol. 1, IEEE, 2009, pp. 296–300.
12. G. Guizzardi, G. Wagner, Towards ontological foundations for agent modelling concepts using the unified fundational ontology (UFO), Springer, 2005.
13. A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, L. Schneider, Sweetening ontologies with dolce, in: Knowledge engineering and knowledge management: Ontologies and the semantic Web, Springer, 2002, pp. 166–181.
14. N. Guarino, Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy, Vol. 46, IOS press, 1998.
15. S. K. Semy, M. K. Pulvermacher, L. J. Obrst, Toward the use of an upper ontology for us government and us military domains: An evaluation, Tech. rep., DTIC Document (2004).