Towards an Ontological Approach to Information System Security and Safety Requirement Modeling and Reuse

O. T. Arogundade\textsuperscript{a}, A. T. Akinwale\textsuperscript{b}, Z. Jin\textsuperscript{c} & X. G. Yang\textsuperscript{a}

\textsuperscript{a} Laboratory of Management Decision and Information Systems, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
\textsuperscript{b} Department of Computer Science, Federal University of Agriculture, Abeokuta, Ogun State, Nigeria
\textsuperscript{c} School of Electronics Engineering and Computer Science, Peking University, Beijing, China

Available online: 14 May 2012


To link to this article: \url{http://dx.doi.org/10.1080/19393555.2011.652290}
Towards an Ontological Approach to Information System Security and Safety Requirement Modeling and Reuse

O. T. Arogundade¹, A. T. Akinwale², Z. Jin³, and X. G. Yang¹
¹Laboratory of Management Decision and Information Systems, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
²Department of Computer Science, Federal University of Agriculture, Abeokuta, Ogun State, Nigeria
³School of Electronics Engineering and Computer Science, Peking University, Beijing, China

ABSTRACT Misuse cases are currently used to identify safety and security threats and subsequently capture safety and security requirements. There is limited consensus to the precise meaning of the basic terminology used for use/misuse case concepts. This paper delves into the use of ontology for the formal representation of the use-misuse case domain knowledge for eliciting safety and security requirements. We classify misuse cases into different category to reflect different type of misusers. This will allow participants during the requirement engineering stage to have a common understanding of the problem domain. We enhanced the misuse case domain to include abusive misuse case and vulnerable use case in order to boost the elicitation of safety requirements. The proposed ontological approach will allow developer to share and reuse the knowledge represented in the ontology thereby avoiding ambiguity and inconsistency in capturing safety and security requirements. OWL protégé 3.3.1 editor was used for the ontology coding. An illustration of the use of the ontology is given with examples from the health care information system.

KEYWORDS reuse, requirement management, use-misuse case, ontology, safety, security, sharing

1. INTRODUCTION

The term “ontology” is derived from its usage in philosophy where it means the study of being or existence as well as the basic categories (Witmer, 2004). Ontology is often used by philosophers as a synonym for metaphysics. The term ontology, or ontologia, was conceived as far back as 1613, independently by philosophers Rudolf Gockel (Goclenius) in his lexicon philosophicum and Jacob Lorhard (Lorhardus) in his theatrum philosophicum. Ontology in the area of computer science represents the effort to formulate an exhaustive and rigorous conceptual schema within a given domain, typically a hierarchical data structure containing all the relevant elements and their relationships and rules within the domain (Gruber, 1993). Ontology, in the artificial intelligence (AI) field, is an explicit specification of a conceptualization used to help programs
and humans share knowledge (Gruber, 1995). The conceptualization is the framing of knowledge about the world in terms of entities, the relationships between them, and the constraints on them; the specification is the encoding of the conceptualization using a formal knowledge representation language. The goal of the specification is to create an agreed-upon vocabulary and semantic structure for exchanging information about that domain. In such ontology, definitions associate the names of concepts in the universe of discourse (e.g., classes, relations and functions) with a description of what the concepts mean and formal axioms that constrain the interpretation and well-formed use of these terms (Beuster, 2002).

The processing of domain knowledge for specific tasks has gained importance since a good interpretation and use of this knowledge may provide support to experts in making decisions and discovering of new knowledge from other experts. Ontology is an important part of developing a shared understanding across a project. Reuse has become an important and critical factor to be considered in system development because of the indispensable benefits attached to it. Mayank, Kositsyna, and Austin (2004) prove empirically that the level of reuse determines the effectiveness of the improvements in productivity, quality, and time-to-market, and they conclude that greater benefits are obtained when reuse is considered during the early phases of the software development lifecycle. Also, Cheng and Atlee (2007) said that requirements reuse has been pointed out as being one of the most pressing needs and grand challenges in requirement engineering (RE) research, whose solutions are likely to have the greatest impact on software engineering research and practice.

Unified Modeling Language (UML) use cases (Alexander & Maiden, 2004) are a widely used technique for elicitation of functional requirements when designing software systems. A use case illustrates what should happen in a system. Misuse case has been proposed to elicit interactions and actions that should not happen in a system (Alexander, 2003). Misuse cases have a way of bringing security, safety, and non-functional requirements as well as design tradeoffs to the surface. The originally proposed misuse case by Ian Alexander addresses outside attackers, representing them as mis-actors. Later, Rostad (2006) proposed an extension of misuse case to include vulnerabilities and inside intruders. At the initial stage of our research, we have proposed an enhanced misuse case that includes vulnerable use case and abuse case in order to be able to capture both unintended and deliberate incidents of safety and security.

The remaining sections of the work are organized as follow. In section 2, we give the motivating factors behind our work. Section 3 presents a literature review of what has been done in ontology with respect to requirement engineering. Section 4 gives a detailed description of the structure of the ontology. Finally in sections 5, 6, and 7, we specify our ontology in OWL protégé 3.3.1 editor and demonstrate it through the use of an example involving the abusive misuse case. In conclusion, we outline directions for future research.

2. MOTIVATION OF THE WORK

Use case-based requirement engineering technique is gaining popularity in the industry, but there is limited consensus to the precise meaning of the basic terminology used in use case concepts. As a result, different people use different terminology to represent actions and relationships. There is inconsistency in understanding use-misuse case concept in requirement engineering practice. It is undoubtedly clear to aim for a precise definition of the conceptual fundamental in use-misuse case, hence the need for ontology. Building a use-misuse case ontology will allow each of the concepts to be semantically represented, which will allow for knowledge sharing. In addition, the ontology presented in this paper defines safety and security concepts in a unified framework from a software engineering perspective and in a way that is comprehensible for both stakeholders and software agents.

3. RELATED WORK

The phase of requirement engineering deals with gathering the system functionality from customers. Since the involved software engineers are often not domain experts, they must learn about problem domain from the customers. A different understanding of the concepts involved may lead to an ambiguous, incomplete specification and major work after system implementation. Therefore, it is important to assure that all participants in the requirements engineering phase have a shared understanding of the problem domain. Ontology can be used to describe requirements specification documents and formally
represent requirements knowledge (Lin, Fox, & Bilgic, 1996).

Utilization of ontology in the RE process has a long history in research. The emergence of the semantic Web has increased interest in ontology for RE. Many works have been done in this area (Greenspan, 1984; Mylopoulos, Borgida, Jarve, & Koubarakis, 1990; Dardenne, vanLamsweerde, & Fickas, 1993; Dubois, 1995; Yu & Mylopoulos, 1994). Though these works were never declared or called ontology, all formalisms have much in common with ontology. As a result of the emergence of the semantic Web, the emphasis now in RE research is on sharing ontology via the Web among globally distributed development teams.

There is limited consensus to the precise meaning of the basic terminology used in use case concepts. The use-misuse case concepts have been richly interpreted and variously used, but the fact that there are no common conceptual foundations on which the various efforts can be compared seems to be a drawback to the potential benefits. In contrast to traditional knowledge-based approaches, ontology seems to be well suited for an evolutionary approach to the specification of requirements and domain knowledge. Moreover, ontology can be used to support requirements management and traceability (Mayank, Kositsyna, & Austin, 2004). Automated validation and consistency checking are considered as potential benefits compared to informal or semi-formal methods that provide no logical formalism or model theory. Also, formal specification may be a prerequisite to realize model-driven approaches in the design and implementation phase.

Joaquin et al. (2009) describe an ontology-based framework for representing and reusing security requirements based on risk analysis. The requirement ontology was based on the RE method called SIREN. The framework was for representing, storing, and reusing security requirements. The requirement ontology was built upon a reusable requirements repository organized by catalogs based on RE standards.

Vorobiev, Han, and Bekmamedova (2008) proposed a set of security ontologies that specify information security issues, allowing the sharing of a common understanding of information about attacks and defenses among humans and computers. Dobson and Sawyer (2006) identified the specific area of dependability RE as one on which little effort has been made to define ontology, even though dependability is an important area for many systems. Contrary to what occurs in many other domains, a consensus exists in the field of dependability, thanks to the works from the IEEE Computer Society Technical Committee on Fault Tolerant Computing and IFIP Working Group 10.4 on Dependable Computing and Fault Tolerance. The authors created ontology compliant with the IFIP Working Group 10.4 taxonomy.

In Karyda et al. (2006), the issue of accommodating security requirements in the development of secure applications is addressed by the use of security ontology. The ontology captures and formalizes security knowledge from security experts and aims to support and improve communication among security experts, users, and developers. Furthermore, it is intended to facilitate software developers to address security requirements at an early stage in the software development process and to support security related design choices.

The main purpose of the use-misuse case ontology is to capture and formalizes safety and security requirements to enable communication between system developers and security and safety requirements engineers. We also want a situation wherein the safety and security requirements captured for a particular application can be reused, shared, and updated. It offers a machine readable knowledge base which can be utilized in novel threat simulation. The scope of this work is thus not limited only to information security and safety, which is concerned with hardware, software, and network. It also encompasses people and facilities playing one role or the other in effective functioning of the information security and safety, which could therefore be target of attack or exploitation by an attack.

4. A DESCRIPTION OF USE-MISUSE CASE ONTOLOGY (U-MCO) CONCEPTS

In accordance to a general concept given by Protégé editor (2004), ontology involves three types of elementary concepts: class, relations, and axioms. Let us first give a detailed description of the basic concepts used in our use-misuse case ontology and explain their relations. The key ingredients of U-MCO are a vocabulary of basic use-misuse case terms and a precise specification of what those terms mean. Figure 1 shows the basic concepts that constitute our ontology.

Ontological Approach to Information System Security
The ontology depicts information about use-misuse case concepts in a simplified manner especially for nontechnical professionals. Each of these classes is well elaborated in the subsequent sections.

Based on the specific concepts discussed above, we give the concrete definition of our ontology as following:

**Definition 1**: Use-misuse case ontology is defined as

\[ U-MCO = \langle C, R, A \rangle \]

where:

- \( C = \{C_1, C_2, \ldots, C_n\} \) is a set of class;
- \( R = \{R_1, R_2, \ldots, R_m\} \) is a set of relation; and
- \( A = \{boolExp\} \) is a set of axioms.

**Being n, m, l \in N**

\( R = \text{Is-a} \cup \text{part-of} \cup \text{DR} \)

The Is-a, part-of and DR are the generalization, the aggregation and the domain specific relations, respectively.

The detailed definitions of class, relations and axioms are as follows:

Each \( C_i = (\text{name}_i, \text{superclass}_i, \{\text{relation set}_i\}, \{\text{property set}_i\}) \) which is a class representing a logical or physical entity in the use-misuse case domain. For example, \( C_0 = (\text{OutsideMisactor}, \text{Misactor}, \{\text{initiate}, \text{Isa}, \{\text{hasName}\}\}) \), OutsideMisactor is the name of this class which indicates the class is defined for outside attacker, Misactor is the superclass which include any misactor, \( \{\text{initiate, Isa}\} \) is the relation set which represents the relation of outside attacker with other classes in the use-misuse case domain.

Each relation \( R_i \) of the relation set \( R \) has some attributes that describe it and the class related by the relation. The meta attributes are name, type, characteristics, domain, range, and axioms.

The axioms are the constraints and rules that are defined on the class and relations. There are different kinds of axioms some of which are illustrated below:

- subclass relationships, for example, OutsideMisactor contained Misactor
- equivalences, for example, Vulnerability \( \equiv \) TechnicalVulnerability, AdminVulnerability, and PhysicalVulnerability
- disjointness, for example, OutsideMisactor \( C \) IntentionalinsideMisactor

Relation \( R_i = (R_{i1}, R_{i2}, \ldots, R_{im}) \) in this paper has six components: R (name, type, characteristics, domain,
4.1. Class Use Case Ontology

A use case is the specification of a set of actions performed by an actor, which yields an observable result that is of value for one or more actors or stakeholders of the system. It consists of three subclasses: HumanUsecase, ComponentUsecase, and InfrastructureUsecase. These subclasses exhibit the relation “include” with different subclasses of class Vulnerability. Some of these relations are shown with the arrows in Figure 2. Subclass HumanUsecase consists of the use cases carried out by the human actor, for example, book appointment, conduct medical test, register patient, update record, and so forth. ComponentUsecase consists of the use cases performed by the hardware connected with the network system. InfrastructureUsecase has to do with the functions of the infrastructure, for example, fire extinguishing, image capturing, and so forth.

4.2. Class Actor Ontology

An actor is a person, system, or some external entity linked to one or more system use cases. Class Actor ontology model consists of three subclasses HumanActor, ComponentActor, and InfrastructureActor. These subclasses exhibit relation “initiate” to HumanUsecase, ComponentUsecase, and InfrastructureUsecase, respectively. An instance of class HumanActor are Patient, Physician, Pharmacist, etc.; an instance of class ComponentActor and InfrastructureActor are modem and windows, respectively (See Figure 3).

4.3. Class Vulnerability Ontology

Class Vulnerability model contains main class Vulnerability and three subclasses. Vulnerability is a kind of weak defense introduced into the system by vulnerable use case. Vulnerable use case is a kind of use case whose initiation can lead to an unintentional misuse case or that may be exploited by intentional misuse cases. Therefore, they are expressed as a type of use case, with an include relation with the use case that introduced it. The three subclasses of class Vulnerability are PhysicalVulnerability, TechnicalVulnerability, and AdminVulnerability. PhysicalVulnerability is a type of PhysicalUsecase, AdminVulnerability is a type of HumanUsecase, and TechnicalVulnerability is a type of...
of ComponentUsecase. This is shown in Figure 4. Admin vulnerability is the vulnerability that led to the inside abusive misuse case initiated by the unintentional inside misactor. In some cases, he is also the actor of the use case that gives rise to the vulnerability.

### 4.4. Class Misactor Ontology

The main class of misactor model ontology is called Misactor, which consists of four subclasses: OutsideMisactor, IntentionalInsideMisactor, UnintentionalInsideMisactor, and NaturalMisactor.
The outside misactor is an outside agent that is not legally involved in the use of the information system but launches malicious attacks/goals at the information system. The OutsideMisactor can either be human or inhuman, but both initiate outside attacks. It has two subclasses: HumanOutsideMisactor, which initiates OutsidedeliberateMisusecase, and Inhuman–OutsideMisactor, which initiates EnvironmentalMisusecase.

The intentional inside misactor is one of the authorized user or participants in the use of the information system. He sabotages the information security to disrupt the proper functioning of the system and seizes privileges to perform unauthorized actions this is also with a malicious intentions. The unintentional inside misactor abuses some of the legal privileges to carry out some illegitimate actions (abusive misuse case) but not with malicious intentions. However, on some occasions the consequences may be grave. In this case the unintentional inside misactor is also the legal actor in the system. He also initiates accidental actions (mistakes) which may harm the system. The natural misactors are natural agents (e.g., rain) that cause natural disaster (natural misuse case). This model is shown in Figure 5. For clarity we only show some of the relations.

4.5. Class Misusecase Ontology Model

A misuse case describes potential behaviors that a system stakeholder deems unacceptable. Many literatures refer to this as a threat. It conveys each threat actor’s (misactor) goal in misusing/disrupting the proper functioning of the system. Class Misusecase consists of six subclasses as seen in Figure 6. They are:

- **AccidentalMisusecase**: Accidental misuse cases are mistakes committed by any of the actors involved in the system. It exhibits the relation threat to the HumanUsecase and relation affect to the HumanActor. This misuse case helps in eliciting both security and safety requirements.

- **InsideabusiveMisusecase**: InsideabusiveMisusecase exploits AdminVulnerability and in turn threatens HumanUsecase and affects HumanActor. It helps to capture safety requirements rather than security requirements. It is initiated by UnintentionalinsideMisactor who is also the actor that initiates the use case that introduces the vulnerability that is being exploited.

- **OutsidedeliberateMisusecase**: Outsidedeliberate-Misusecase is a potential attacks from crooks and
hackers to disrupt/harm the proper functioning of the system. It exhibits exploit relation to TechnicalVulnerability, AdminVulnerability, and PhysicalVulnerability. It threatens HumanUsecase, ComponentUsecase, and InfrastructureUsecase. It affects HumanActor, InfrastructureActor, and ComponentActor.

- **InsidedeliberateMisusecase:** Insidedeliberate-Misusecases are those malicious actions carried out by authorized actors in the system by exploiting some of the system vulnerabilities.

- **EnvironmentalMisusecase:** Environmental-Misusecase is an unwanted action from the environmental factors, for example, fire, flood, vandalism, power loss, and so forth. It exploits PhysicalVulnerability and TechnicalVulnerability. It threatens InfrastructureUsecase and ComponentUsecase and in turn affects InfrastructureActor and ComponentActor. The class hierarchy and some of the relations are shown in Figure 6.

- **NaturalMisusecase:** NaturalMisusecases are attacks due to natural disasters, for example, earthquakes.

## 4.6. Class Safeguard Ontology

Class Safeguard consists of two subclasses as shown in Figure 7. They are SafetySafeguard and SecuritySafeguard. The Safeguard represents constraints on the system functional requirements rather than being functional requirements themselves. As constraints, they are preventive and corrective measures against every form of misusecases, vulnerabilities, and misactors and a protective measure for the actors and the use cases. SafetySafeguard are requirements needed to mitigate AccidentalMisusecase, EnvironmentalMisusecase, and InsideabusiveMisusecase and to prevent the Misactor of these misuse cases, respectively. It also protects HumanActor. The SecuritySafeguard are requirements to mitigate all other forms of security assaults (Misusecases and Misactor) and enhance the proper functioning of the information system.

## 5. USE-MISUSE CASE ONTOLOGY CODING

For ontology to be used within an application, the ontology must be specified, that is, delivered using some concrete representation. There are a variety of languages which can be used for representation of conceptual models, with varying characteristics in terms of their expressiveness, ease of use, and computational complexity.

The most recent development in standard ontology languages is OWL from the World Wide Web Consortium (W3C; http://www.w3.org/). It has the most complete set of expressions for capturing the different concepts and relationships that occur within ontology; therefore, the use-misuse case knowledge is captured in OWL protégé 3.3.1 editor. The OWL ontology fundamentals for modeling the use-misuse
case domain include ontology class and subclasses, ontology properties (data type property and object property), association between ontology class and ontology property, property characteristics, constraints or restrictions. This is illustrated in Figure 8. The figure show the class InsideabusiveMisusecase, a brief description of the class, its superclass Misusecase, the properties exhibited by individual of the class “affect,” “exploit,” and “threaten” with individual of other classes HumanActor, AdminV ulnerability, and HumanUsecase and the restrictions on the properties. Disjoint classes to this class are also displayed.

6. EVALUATION OF U-MCO

The ontology was populated with instances to measure its compliance with the initial expectations. Usage scenarios and competency questions were also used as a reference for the evaluation. The internal consistency of the ontology has been validated using pellet reasoner (http://pellet.owldl.com/). The first assessment has been done by the authors who tested the effectiveness of the ontology in answering two competency questions:

1. Display all human actors and their corresponding use cases?

SELECT ?HumanActor ?Usecase
WHERE
   ?HumanActor:initiate ?Usecase

2. List all use cases threatened by Natural Misuse cases?

SELECT ?NaturalMisusecases ?Usecase
WHERE
   ?NaturalMisusecase:threaten ?Usecase

The result of these queries is displayed in Figure 9. The structure of the U-MCO can also be navigated. Such view allows the user to become familiar with the ontology showing only the class hierarchy. Also, all classes and properties have been clearly commented to give the intended meaning of each. Moreover, it is possible to browse the data associated with each class.

By following guidance on ontology development (Noy & McGuiness, 2001) and encoding our ontology in OWL, we have served to demonstrate that the concepts in the ontology are well-defined and that the relationships between those concepts are precise. The use of OWL also provides us with application-based evaluation of the ontology content (Protégé ontology editor, 2004).

6.1. Illustrative Adoption of U-MCO

In this section we choose an example to illustrate how use-misuse case ontology can be utilized and how
it allows for sharing of knowledge and enhances unambiguous communication among teams involved in all the stages of information system development.

In U-MCO, for the purpose of eliciting safety and security requirements of an information system, there must be an instance of any of the subclasses of the superclass Actor and use cases. We shall consider this application.

An Electronic Health Record (EHR) is a patient-oriented, aggregated, longitudinal system of systems which assembles health information about a patient over a wide area network from, potentially many geographically dispersed data sources. An EHR provides individuals with an aggregate, secure, and private lifetime record of their key health history and care within the health system and shares encounter information available electronically with authorized health care providers and the individual anywhere and anytime in support of high quality care. It may draw on health information from sources such as electronic medical records (EMRs), drug repositories, centralized lab data sources, and other point-of-service applications over many encounters to assemble a complete health record about the patient. It is a patient-centric document that may contain information from a broad range of providers other than family physicians, such as specialists, social workers, pharmacists, radiologists, dietitians, physiotherapists, and nurses (Boaden & Joyce, 2006). The goal is to identify safety and security requirements.

Considering this scenario, we need to identify instances of the classes and subclasses in our ontology. In order to achieve this, the user need to click on each of the classes starting from the Actor followed by the use cases, the misuse cases, and the misactor. Clicks on any of these classes display the subclasses and textual description of each in natural language. The description in natural language will give the user a
proper understanding of the classes and the subclasses. Bearing this in mind, the user can adequately indentify the actor in the scenario, the use cases, the misuse cases and the misactor in that order. Having identified these instances, safety and security requirements can then be captured based on the misuse cases identified. All these information are stored in the repository (the ontology). Once these information and instances are stored, it can then be shared without ambiguity and consistently. The requirements can also be reused and traced for possible corrections.

We shall be sharing the use case model, the actor model, the vulnerability model, the misuse case model, and the safeguard model of an actor (patient) from the e-health care system. These use cases are retrieved from Health Information Technology Resource Toolkit developed through the University of Kansasa HISPC project (http://ehealth.kansashealthonline.org). We shall add the following instances of class HumanUsecase for an instance of HumanActor (patient):

- View medical record; get laboratory result; schedule appointment; change appointment; receive reminder for checkups; print a copy of medical record; renew medication.

We shall add the following instances of class Misusecase:
- AccidentalMisusecase “loss of password”;
- InsideabusiveMisusecase “self treatment”;
- InsidedeliberateMisusecase “unauthorisview of patient record”;
- OutsidedeliberateMisusecase “spoofing”; NaturalMisusecase “flood.”

We shall add the following instances for class Vulnerability:
- PhysicalVulnerability “wooden window”;
- TechnicalVulnerability “lack of firewall”; 
- AdminVulnerability “view medical record.” We also add the following instances of Safeguard:
- SafetySafegurd “safety awareness,” “strict ethical training”; SecuritySafeguard “iron window release info on a need-to-know basis.”

Once these are well represented in the use-misuse case ontology, they can be explicitly shared and reuse among teams of developers for this particular application.

7. CONCLUSION AND FUTURE WORK

Information system safety and security requirements must be carefully considered not as an isolated aspect
but rather as an element considered in all stages of the system development lifecycle, from requirement analysis to implementation and maintenance. In this case, reuse is an all important and critical factor to be given attention to in nonfunctional requirement engineering, hence the need for the concepts and relations of use-misuse case for the purpose of eliciting safety and security requirement to be formally defined so that they can be shared by the community of security requirement engineers.

Existing security ontology concentrates on security mechanisms that can be applied in the development of information security, while our approach focuses on eliciting safety and security requirements of information security which will aid or force the system developer to focus on what the system must do and not how it is to be done and avoid the trap of making assumptions about how the requirements will be accomplished. Security and safety requirements are distinct from architectural mechanisms that are traditionally used to fulfill them. This ontology can be the basis of a safety and security requirements engineering process for a specified application. The U-MCO presented in this paper is still a prototype which requires refinement, elaboration, and introduction of more complex competency questions.

The benefit of this ontology cannot be overemphasized. It makes communication among system developers working in different locations straightforward. In addition, the development of further software is simplified because developers may refer to the well-formalized OWL syntax we used for defining the UMCO. On the other hand, since the formalization we propose was designed in collaboration with experts in the field, the proposed terminology and processes are trustworthy. The security knowledge of the proposed information security through the shared repository can be traced and managed. The semantics and reasoning capabilities of the UMCO allow misuse cases to be linked to the use cases. It also allows, the misuser cases to be correlated with the safety and security requirements needed to mitigate them.

Another important benefit of this ontology is that it will enable the developer to practice higher level of reuse. Reusing security and safety requirements will help to increase their quality. Inconsistency, errors, ambiguity, and other problems can be detected and corrected for an improved use in subsequent projects.

For our future work, we will elaborate and refine the ontology in order to have a full repository. We shall incorporate concepts and knowledge which can be used for risk assessment into the ontology.

REFERENCES


**BIographies**

**O. T. Arogundade** is a PhD student at Academy of Mathematics and System Sciences, Graduate University of Chinese Academy of Sciences, Beijing China. She received a B.Sc. degree in computer science from the University of Ado-Ekiti, Ekiti State, Nigeria. She had the M.Sc. degree in computer science from the Federal University of Agriculture, Abeokuta, Nigeria. Her current research interests include security requirement engineering, reuse, ontology, business/IT alignment and risk modeling. She has published many articles in journals and conference proceedings.

**A. T. Akinwale** is an associate professor and researcher at University of Agriculture, Abeokuta, Nigeria. His research interest encompasses Management Information System, Query Algorithms Optimization and cybernetics. Adio holds Master degree in Cybernetic and Computer Science and Ph.D. degree in Economic Informatics all from Oskar Langer University, Wroclaw, Poland. He has published many articles in journals and conference proceedings.

**Z. Jin** received the MS degree in computer science and the PhD degree from the Changsha Institute of Technology, China, in 1987 and 1992 respectively. She is now a professor of computer science in the Academy of Mathematics and System Science, Chinese Academy of Science. Her current research is on software requirements engineering and Knowledge Engineering. She has published more than 90 referred papers in the area of requirements engineering and knowledge-based software engineering.

**X. G. Yang** is a full professor at Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences (CAS). He is currently the director of Key Laboratory of Management, Decision and Information Systems (MADIS), CAS. Prof. Yang’s research interests include risk management, operation research, social network, decision support systems. He has published nearly 100 papers in both domestic and international journals. He has many years of working experience both as an academic and practitioner.