Journal of Systems Engineering and Electronics
Vol. 23, No. 1, February 2012, pp.78–87

Capability requirements modeling and verification based on fuzzy ontology

Qingchao Dong*, Zhixue Wang, Weixing Zhu, and Hongyue He

Institute of Command Automation, PLA University of Science and Technology, Nanjing 210007, P. R. China

Abstract: The capability requirements of the command, control, communication, computing, intelligence, surveillance, reconnaissance (C4ISR) systems are full of uncertain and vague information, which makes it difficult to model the C4ISR architecture. The paper presents an approach to modeling the capability requirements with the fuzzy unified modeling language (UML) and building domain ontologies with fuzzy description logic (DL). The UML modeling constructs are extended according to the meta model of Department of Defense Architecture Framework to improve their domain applicability, the fuzzy modeling mechanism is introduced to model the fuzzy efficiency features of capabilities, and the capability requirement models are converted into ontologies formalized in fuzzy DL so that the model consistency and reasonability can be checked with a DL reasoning system. Finally, a case study of C4ISR capability requirements model checking is provided to demonstrate the availability and applicability of the method.

Keywords: fuzzy ontology, fuzzy unified modeling language (UML), fuzzy description logic (DL), model checking.

DOI: 10.1109/JSEE.2012.00011

1. Introduction

A capability, as defined by Department of Defense Architecture Framework 2.0 (DoDAF 2.0), is "the ability to achieve a desired effect under specified standards and conditions through combinations of means and ways to perform a set of tasks" [1]. The capability requirements are described, from the user view and the processes to be supported by information systems, and they should be qualified with values of capacity, speed, and accuracy. In another word, they address both functional and non-functional requirements of the to-be systems. Currently, many architecture frameworks, like DoDAF, recommend using the unified modeling language (UML) [2] to model the complicated architectural concepts and command, control, communication, computing, intelligence, surveillance, reconnaissance (C4ISR) requirements.

However, the capability requirements of systems would be initially articulated in terms of operational capability concepts. Such concepts might be full of uncertain and vague information which cannot be modeled with the existing UML method. The problem would be encountered when describing the efficiency aspect of the capability requirements, for example, a statement "we need a fast vehicle" expresses a vague requirement for the maneuver capability. Here, "fast" is an uncertain and imprecise concept of efficiency. We may simply define an attribute "speed" for the class "vehicle", but soon find it difficult to determine the quantity.

On the other hand, as a semi-formal modeling language, the UML does not support the formal specification and verification. While the DoDAF provides meta models for architectural modeling constructs and rules, there are no rigid definitions and formal specifications for them. This leads to difficulty in checking the models of different viewpoints.

To solve the problem and enable automatic model checking with the reasoning technique, the paper presents a technique of capability requirements modeling and verification. It extends the UML modeling constructs with the meta model of DoDAF so that requirements models can be built within the standard framework. A fuzzy UML is introduced so as to model the fuzzy efficiency features of capabilities. To enable automatic model checking, we present a three-layer framework for modeling C4ISR capability requirements and provide an algorithm to convert the fuzzy UML models into the ontologies specified in fuzzy description logic (DL), so that the built model can be checked for consistency and reasonability through the logical inference.

The rest of the paper is organized as follows. Section 2 points out the lack of current approaches compared with our approach. Section 3 introduces the three-layer modeling framework and discusses the fuzzy UML and the tech-

Manuscript received July 21, 2010.
*Corresponding author.

nique for the UML extension. Section 4 explains the procedure of converting the capability requirements models into the ontologies in the fuzzy DL system f-*SHOIN*(D). Section 5 gives an example of efficiency analysis of the air warning system to demonstrate the applicability of the method. Section 6 brings a conclusion.

2. Related research

2.1 Architecture framework based requirements description

The defense industry widely adopts architecture frameworks, such as Zachman [3], Ministry of Defence Architecture Framework [4] and DoDAF [1], to model the C4ISR capability requirements. Those framework approaches, however, focus more on modeling the function features while leaving a lack in modeling non-function features. The DoDAF, for example, provides the capability meta model (CMM) defining the basic concepts and relationships mostly for the function features of C4ISR capability. However, the C4ISR capability modeling is closely related to the efficiency feature description and analysis. For example, the maneuver capability is usually featured by two efficiency concepts "distance" and "time". The military forces equipped with different transportation vehicles, like cars, trains and airplanes, may have different maneuver capabilities which are evaluated by the two concepts in how far and how fast the forces can move across the battlefields.

2.2 Ontology based requirements elicitation

The Canadian research agency believed that C4ISR capability requirements analysis required the expertise and technologies based on the ontology paradigm [5]. Jin proposed an approach to the ontology-based requirements elicitation process [6]. He defined the enterprise ontology as the meta-model to help domain users capture the domain knowledge. By reusing the domain ontologies, the software models can be built automatically. Chen et al. proposed a meta-model for the Zachman framework which integrated the Bunge-Wand-Weber ontology and the Enterprise ontology [7]. Lichtblau et al. explored the new way to allow multiple ontologies with cross-domain resolution to coexist in a net-centric environment [8].

However, those approaches discussed above are only applicable for the certain and precise ontology, but can not model the uncertain and imprecise information of the C4ISR capability requirements. As a matter of fact, most current general-purpose modeling paradigms, like the UML and integrated definition (IDEF) methods, are inapplicable unless they are extended and improved.

2.3 Our approach

Based on the ontology approaches and the architecture framework techniques, the paper suggests a three-layer framework, as shown in Fig. 1, for capturing the domain knowledge and building the C4ISR capability requirements models.

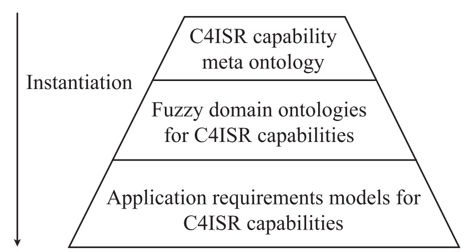

Fig. 1 A three-layer framework of capability requirements modeling

The first layer is the C4ISR capability meta ontology (CCMO) which is defined according to the CMM of DoDAF2.0. The CCMO, as shown in Fig. 2, defines the

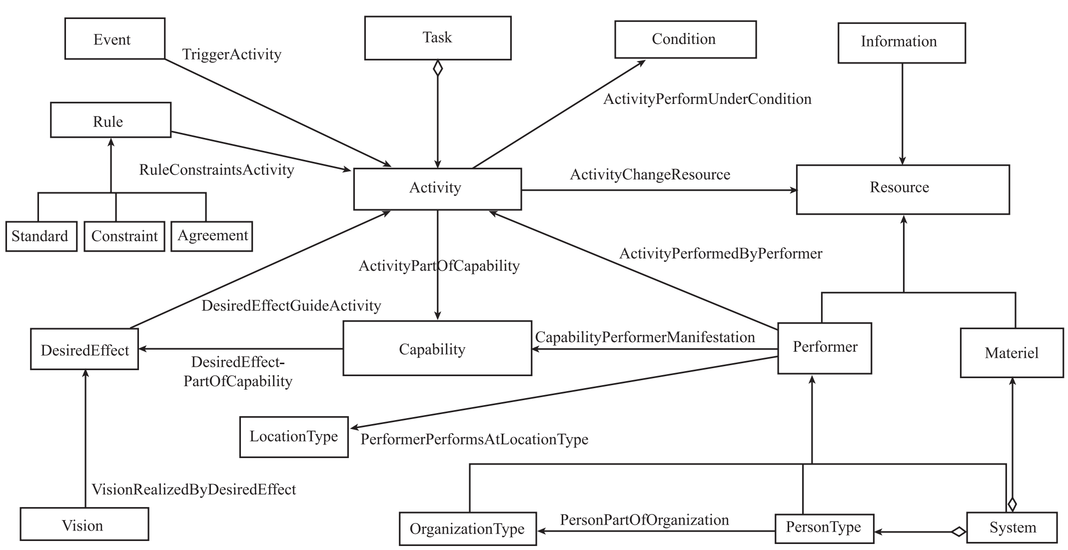

Fig. 2 The CCMO

meta-concepts and relations for modeling the C4ISR capabilities. It can guide the domain experts to capture the domain knowledge by specifying stereotypes for the domain-specific concepts and relations, and provide architectural rules and constraints for checking the capability models.

The second layer is made up of fuzzy domain ontologies for C4ISR capabilities, which are built by the domain experts and which functions are: (i) capturing the domain concepts to provide modeling templates for the application requirements models; (ii) explaining the model semantics at domain level for both function features and efficiency features of capability requirements respectively in terms of certain concepts and uncertain concepts; (iii) prescribing the domain-specific rules for the model verification.

The third layer produces the application requirements models for C4ISR capabilities, which are the final products of capability requirements analysis. To enable semantic checking of the built models, we formalize the application requirements models using f-*SHOIN*(D), the subsystem of fuzzy DL, which has satisfied ability both in expressing and reasoning. The model checking can guarantee the application requirements models consistent with both CCMO meta-concepts and C4ISR domain concepts. Furthermore, the model checking can find out the unsatisfied efficiency features in a capability requirements model. Compared with the current model checking approaches [9–11], our approach is more powerful in reasoning through the uncertain and imprecise information, and with both architectural rules and domain-specific rules.

3. Describing the C4ISR capability requirements in the three-layer framework

3.1 C4ISR capability meta ontology

Definition 1 The C4ISR capability meta ontology is built of the tuple: <Meta-concept, Meta-association, Meta-rule>.

Meta-concept is a concept set. It contains all capability related meta-concepts from the CMM of DoDAF and keeps the same semantics.

Meta-association is a relation set. It contains all capability related meta-relations from the CMM of DoDAF and keeps the same semantics.

Meta-rule is a rule set. It provides the architectural constraints which need be held by all modelers in modeling the C4ISR capability requirements [1]. The rules in Meta-rule come from the CMM, for example, “every performer must have one or more capabilities and must complete one or more activities”.

3.2 Fuzzy domain ontology for the C4ISR capability domain

3.2.1 Definition for the fuzzy domain ontology

The CCMO is usually found too abstract for modeling capability requirements, because many engineers may lack of domain knowledge which is necessary for understanding the user requirements and modeling the capabilities. Moreover, compared with the function feature of C4ISR capability requirements, the efficiency features contain vague information and are harder to model. Therefore, we suggest defining, based on the CCMO, a number of fuzzy domain ontologies for a C4ISR capability domain before modeling the capability requirements. Those ontologies, containing domain-specific concepts and relations, offer requirements description templates, define the domain semantics and can be reused for many applications from the same domain.

Definition 2 The fuzzy domain ontology for a C4ISR capability domain is built of the tuple: <Dom-concept, Dom-association, Dom-function, Dom-constraint>.

Dom-concept is a set of domain concepts. Each element is defined by a unique meta-concept of CCMO. Dom-concept should satisfy following conditions: Dom-concept $=$ DomPreciseConcept $\cup$ DomFuzzyConcept; e: concept $\rightarrow [0, 1]$ where each e is an instance defined by one of Dom-concepts; DomPreciseConcept $\cap$ DomFuzzyConcept $= \varnothing$ where the DomPreciseConcept is a set of certain domain concepts describing the function features of the domain, e.g., AdvancedMobility. DomFuzzyConcept is a set of the uncertain concepts, e.g., a wide operational scope and a fast moving speed.

Dom-association is a set of relations, associated with domain concepts. Each element is defined by a unique meta-relation of the CCMO. These relations can be classified into three types: aggregation, association and generalization. Dom-association should satisfy such a condition: r: Association $\rightarrow [0, 1]$, where each r is an instance defined by an Association which is a member of DomAssociation.

Dom-function is a complete function, which is used to trace the stereotypes of domain concepts or relations in the CCMO.

Dom-constraint is a set of domain model constraints defined according to the domain requirements.

3.2.2 Enhanced OO modeling technology

To model the C4ISR capability domain and build the fuzzy domain ontology, we adopt the extended fuzzy UML to describe the uncertain domain-specific concepts. Here we first introduce some basic concepts of the fuzzy UML, then discuss the extension of the UML.

Ma et al. presented a fuzzy extension of UML [12]. A fuzzy class defines a set of fuzzy objects with similar structure, relationships and behavior. There are three levels of fuzziness: (i) a class is fuzzy, because there are one or more uncertain attributes which are followed by a pair of words WITH *mem* DEGREE, where $0 \leqslant mem \leqslant 1$, to define the degree of membership; (ii) a class is fuzzy, because whether the objects belong to the class is uncertain, and an additional attribute (μ) with a domain of [0, 1] is specified in the class to represent the objects membership degree; (iii) a class is fuzzy, because it has a special type of attributes with fuzzy values, where a keyword FUZZY is placed in front of the attribute. Other details of the fuzzy UML can be further referred to [12,13].

According to [14], we can specify a domain-specific modeling language by extending the meta object facility (MOF) of the UML with the CCMO. Accordingly, the new stereotypes are defined by the concepts such as performer, capability and system. The new association stereotypes are defined by the relations such as ActivityPartOfCapability (APOC) and VisionRealizedByDesiredEffect (VRBDE).

To model the function and efficiency features of a specific C4ISR domain, the domain experts should firstly import the extension UML profile, then model the function features as a fuzzy class with the new stereotypes, and model the corresponding efficiency features as fuzzy attributes.

To value the fuzzy efficiency attribute, we introduce an important concept named fuzzy efficiency evaluation function (FEEF). FEEF has a value between 0 and 1, and is used to describe the possibility and quality of a capability to support a special efficiency. Taking the air warning system for example, the Air-targetInterceptionCapability (AIC) is featured by a fuzzy efficiency attribute EarlyAlarmGrade (EAG), one of the indicators for measuring whether the activity can be executed satisfying the mission effectiveness specified in the warning ahead requirement. For the case, FEEF can be defined as

$$Early(t) = \begin{cases} 1 - 0.01t, & 0 \leqslant t \leqslant 100 \text{ min} \\ 0, & t > 100 \text{ min} \end{cases}$$

where t is a variable representing the time consumed in identifying the flying object and judging its danger, the constant 100 is the maximum minutes for an air defense force to commit a mission, and the coefficient 0.01 is given by the domain experts. If $0 \leqslant t \leqslant 100\,\text{min}$, FEEF will generate an efficiency value to describe the degree of early warning.

Accordingly, we can further extend the fuzzy UML by specifying that every fuzzy efficiency attribute of a fuzzy class is defined by a fuzzy efficiency evaluation function rather than by a type. Fig. 3 gives a partial example of the

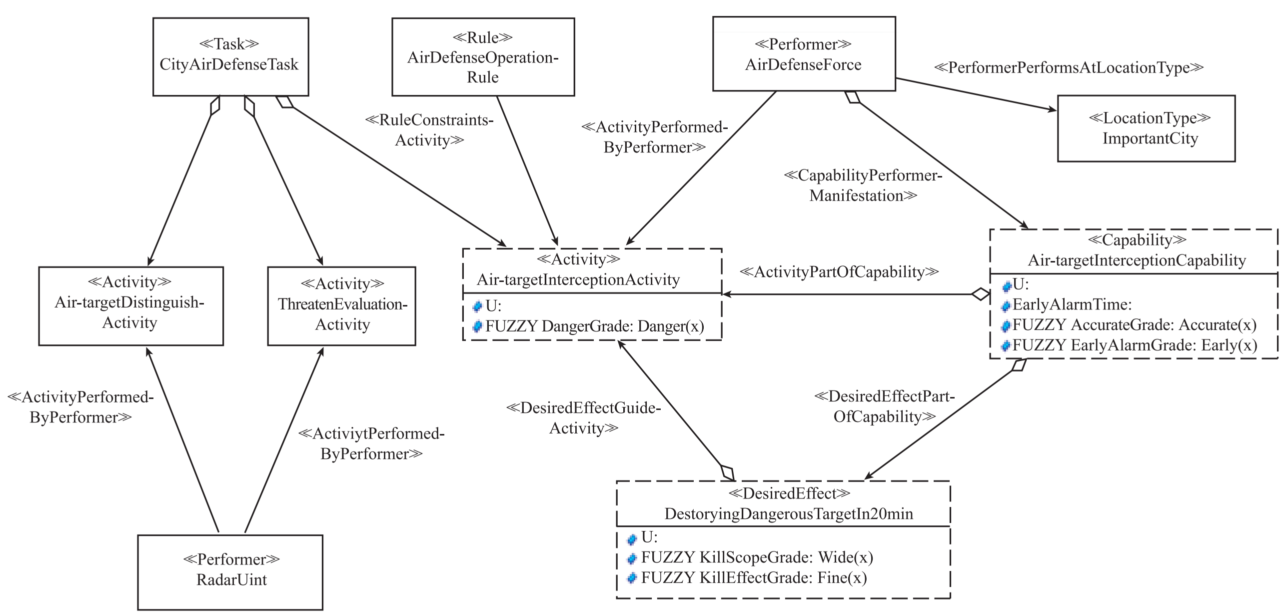

Fig. 3 The C4ISR integrated air warning fuzzy ontology

integrated air warning fuzzy ontology (IAWFO), which reflects architecture-level requirements and defines a fuzzy domain ontology for the air defense domain. These function and efficiency feature of the domain are modeled by fuzzy class and fuzzy attributes.

The method of domain knowledge acquisition is also discussed in our previous work referenced to [15]. It suggests that the C4ISR domain ontology construction is started with mission analysis by modeling the goals. These domain concepts and relations will be captured through the goal refinement with UML class diagram, usecase analysis with usecases diagram and capability analysis with activity diagram.

3.3 Application capability requirements models

Definition 3 The capability requirements model for a C4ISR application system is built of the tuple: <App-object, App-association, App-function>.

App-object is a set of application objects, every of which is defined by a unique domain concept of the fuzzy domain ontology.

App-association is a set of application system relations associated with a pair of objects, every of which is defined by a unique domain relation of the fuzzy domain ontology.

App-function is a complete function, specifying the mappings of App-object into Dom-concept or App-association into Dom-association, which is used to trace the type of domain concepts or relations in the fuzzy domain ontology.

After the domain experts have constructed a fuzzy domain ontology for the C4ISR capability domain, the engineers can reuse the fuzzy domain ontology as class templates to model the capability requirements for the C4ISR applications. By reusing the domain knowledge, the things of the problem world are identified as objects and modeled by the domain classes, and the efficiency features are specified as the fuzzy attributes defined by the fuzzy domain ontology.

By reusing the domain ontologies [16], the engineers are able to model both functional and efficiency features of the application requirements and analyze whether the built models satisfy the domain constraints or whether the designed systems can achieve the desired effect with the model checking technique as discussed in next section and needlessly worrying about lack of domain knowledge.

Based on the example of the integrated air warning system, Fig. 4 shows the capability requirements model for the application.

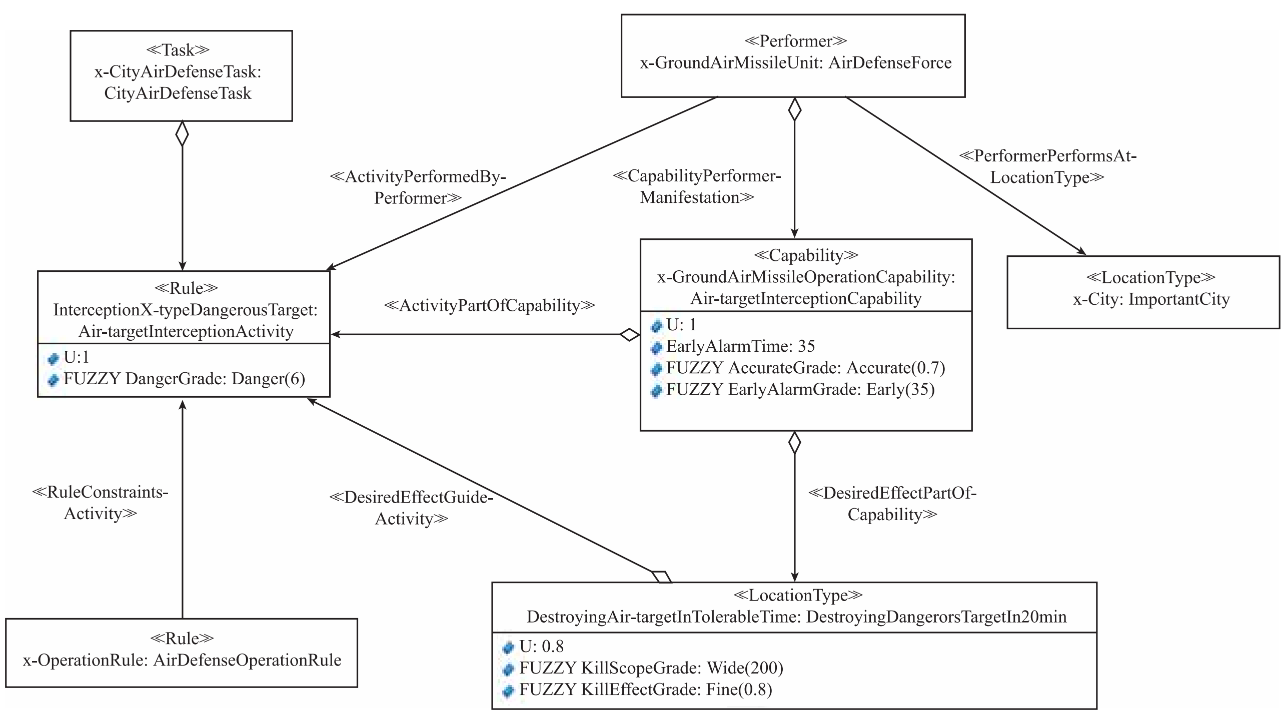

Fig. 4 Air-target interception capability of the x-air defense force

4. Capability requirements model conversion and verification

As a semi-formal modeling language, the UML does not support the model checking through the logic inference. The popular way to solve the problem is to use object constraint language (OCL) to define additional well-formedness rules that apply to the concepts in the model [2]. This is obviously difficult for the domain experts who may lack of formal language background. Here we suggest to convert the C4ISR capability requirements model into the ontology formally defined using f-*SHOIN*(D), the subsystem of the fuzzy DL which supports automatic verification with some popularly applied reasoning engines.

The principle of verification is to construct the knowledge base from the CCMO, the fuzzy domain ontology and the capability requirements models, and then verify the consistency and the reasonability of the built models with the reasoning system of the fuzzy DL. The following algorithm describes the step-by-step process of the knowledge base construction.

Algorithm Constructing the knowledge base for the capability requirements verification.

Input C4ISR capability meta ontology; the fuzzy domain ontology for a C4ISR capability domain; the capability requirements model for an application.

Output The knowledge base of C4ISR capability requirements.

Step 1 Extend the fuzzy domain ontology.

As discussed in Section 3, the capability requirements are modeled in three-tier framework. While the fuzzy DL knowledge base is built in two-tier structure (concepts-instances), the concepts at CCMO level and domain level need to be flattened out onto one level to define the application objects. In other words, the CCMO meta-concepts need to be added to the domain ontology, forming class hierarchies.

(i) Combining the concepts of fuzzy domain ontology with those of the CCMO.

NewDomainConcept = Meta-concept ∪ Dom-concept, NewDomainAssociation = Meta-association ∪ Dom-association, NewDomainConstraint = Meta-rule ∪ Dom-constraint.

(ii) Creating generalization relationships among them.

For every domain-specific concept $C_{\mathrm{dom}} \in$ Dom-concept, and *stereotype* $(C_{\mathrm{dom}}) = C_{\mathrm{meta}}$, create a generalization relationship between C_{dom} and C_{meta}, and add the new generalization relation into the NewDomainAssociation.

Step 2 Create the concepts and their instances for the knowledge base.

Create concept For every certain concept $C \in$ NewDomainConcept, create a same name concept C in TBox T. If concept C has an attribute *Att* and the range of *Att* is P, create a same name concept P and a relation *Att*, and add an axiom $C \subseteq \forall Att.P$ into the TBox T. If the attribute has a multiplicity *multiplicity*(*Att*) $= [i \ldots j]$, add an axiom $C \subseteq \geqslant iAtt \cap \leqslant jAtt$.

For every fuzzy concept *FC* ∈ NewDomainConcept, create a same name concept *FC* in TBox T and specify its fuzziness in the following way.

(i) WITH *deg_A* DEGREE at the first level of fuzziness is defined by an axiom $FC \subseteq \forall T.\{deg\_A\}$, where T is a special fuzzy concrete relation. It means that the instance membership degree to concept *FC* is less than or equal to *deg_A*.

(ii) The μ at the second level of fuzziness is implicitly defined in the fuzzy concept.

(iii) At the third level of fuzziness, the fuzzy attribute named *Att*, owned by a fuzzy concept FC and of type C, is defined by an axiom $FC \subseteq \forall Att.C$. If the attribute has a *multiplicity*(*Att*) = [*i*. . .*j*], add an axiom $FC \subseteq \geqslant iAtt \cap \leqslant jAtt$.

Create instances For every object $o \in$ App-object, which is mapped by *App-function* $(o) = C$ to a certain domain concept C with an attribute *Att* valued a, create the same name instances o and a in the ABox A, specify the class-of relationship between C and o, and add the assertions $< o : C = 1>$ and $< (o, a)$: $Att = 1>$.

For every object $o \in$ App-object which is mapped by *App-function* (o)=*FC* to a fuzzy domain concept *FC* with an attribute *Att* valued a where the instance membership degree is $\rhd\lhd n$ and the attribute membership degree is $\rhd\lhd m$(n and m are between 0 and 1), create the same name instances o and a in the ABox A, specify the class-of relationship between C and o, and add the assertions $< o$: *FC* $\rhd\lhd n >$ and $< (o, a)$: *Att* $\rhd\lhd m >$.

Step 3 Establish the relations for the knowledge base.

There are three types relations in the requirements model, association, generalization and aggregation. The conversion will be done in the following steps.

Association For every domain relation $R(A, B) \in$ NewDomainAssociation typed by the function *Type*(R) = association (i.e., R is an association relationship), if A and B are fuzzy classes with the first level of fuzziness WITH *deg_A* DEGREE and WITH *deg_B* DEGREE and the association $R(A, B)$ with WITH *deg_ass* DEGREE, create in TBox T the same name relations R with the reversed relation R^- where $R = R^{--}$ and add in TBox T such axioms: $A \subseteq \forall R.B$; $B \subseteq \forall R^-.A$; $\forall R.\mathrm{T} \subseteq \forall T.\{deg\_A\}$;

$\forall R.\mathrm{T} \subseteq \forall T.\{deg\_B\}$ and $\forall R.\mathrm{T} \subseteq \forall T.\{deg\_ass\}$.

For every instance relation $r(a, b) \in$ App-association typed by the function *App-function*$(r) = R$ and *Type* (R) = association where the possibility of the relation $r(a,b)$ is $\triangleright \triangleleft n$, add such assertion in the ABox A as $< (a,b)$: $R \triangleright \triangleleft n >$ and $< (b, a)$: $R^- \triangleright \triangleleft n >$.

Generalization For every domain relation $R(A, B) \in$ NewDomainAssociation typed by the function *Type*(R) = generalization (i.e., B is a subclass of A), if A and B are fuzzy classes with the first level of fuzziness WITH *deg_A* DEGREE and WITH *deg_B* DEGREE, create in the TBox T the relation *SubConcept* and add in TBox T such axioms: $B \subseteq A; A \subseteq \forall T.\{deg\_A\}; B \subseteq \forall T.\{deg\_B\}$ and $B \subseteq \forall SubConcept.A$.

For every instance relation $r(a, b) \in$ App-association typed by the function *App-function*$(r) = R$ and *Type* (R) = generalization where the possibility of the relation $r(a,b)$ is $\triangleright \triangleleft n$, add such assertion in the ABox A as $< (b,$ $a)$: *SubConcept* $\triangleright \triangleleft n >$.

Aggregation For every domain relation $Agg(A, B) \in$ NewDomainAssociation typed by the function *Type*(*Agg*) = aggregation (i.e., A is the aggregate part, B is the constituent part), if A and B are fuzzy classes with the first level of fuzziness WITH *deg_A* DEGREE and WITH *deg_B* DEGREE, create in TBox T the same name relations *Agg* with the reversed relation Agg^- where $Agg = Agg^{--}$ and add in TBox T such axioms as: $A \subseteq \forall Agg.B$; $B \subseteq \forall Agg^-.A$; $A \subseteq \forall T.\{deg\_A\}$; $B \subseteq \forall T.\{deg\_B\}$; $\forall T.\{deg\_A\} \subseteq \forall T.\{deg\_B\}$; $\forall Agg.\mathrm{T} \subseteq \forall T.\{deg\_B\}$; $\forall Agg.\mathrm{T} \subseteq \forall T.\{deg\_A\}$; $\forall Agg^-.\mathrm{T} \subseteq \forall T.\{deg\_B\}$ and $\forall Agg^-.\mathrm{T} \subseteq \forall T.\{deg\_A\}$.

For every application relation *agg* $(a, b) \in$ App-association typed by the function *App-function*(*agg*) = *Agg* and *Type*(*Agg*) = aggregation where the possibility of the relation $r(a,b)$ is $\triangleright \triangleleft n$, add such assertion in the ABox A as $< (a, b)$: *Agg* $\triangleright \triangleleft n >$; $< (b,a)$:$Agg^- \triangleright \triangleleft n >$.

Step 4 Build the domain rule-base.

The domain checking rules can be formally specified as the cardinality restrictions imposed on the instance associations of meta-concepts. To explain the rule-base construction algorithm, we classify those rules into two categories so as to handle them respectively.

Qualified cardinality restriction (QCR) rules The QCR rules are expressed as: $\forall x.C_1(x) \rightarrow (\# \{y|C_2(y) \wedge R(x,y)\} \geqslant 1)$. The examples of such rules are like: each Performer must have one or more Capabilities; each Performer must complete one or more Activities.

Constant cardinality restriction (CCR) rules The CCR rules are expressed as: $\forall x.C_1(x) \rightarrow (\# \{y|C_2(y) \wedge R(x,y)\} = n)$. The examples provided by DoDAF are like: each Person must be the member of only one Organization at a certain time.

Based on the above classification, the domain rules are added in f-*SHOIN*(D) system in following way, forming the domain rule-base.

Rule-base construction For every *Type* (*Rule* (R (C_1,C_2))) = QCR, add in TBox T such axioms $C_1 \subseteq \forall R.C_2$ and $C_1 \subseteq \geqslant 1R$.

For every *Type* (*Rule* (R (C_1,C_2))) = CCR, add in TBox T such axioms $C_1 \subseteq \forall R.C_2$, $C_1 \subseteq \geqslant nR$ and $C_1 \subseteq \leqslant nR$.

Having acquired the knowledge base of capability requirements, the model verification is transformed into knowledge base checking for consistency and reasonability, and is easily automated by using DL-supporting inference engines, such as Pellet [17] and FaCT++ [18], which are currently mature and popularly applied to Semantics Web. Usually, such inference engine is based on the inference algorithm Tableau, details of which can be further referred to [19].

Above algorithm has been realized and integrated into a requirements analysis tool so-called ontology-based capability requirements elicitation and analysis tool (OBCREAT) under the support of the National High-Tech Research and Development Program of China.

5. Case study on model checking

Continued on the example in Section 3, we can now examine the application capability requirements model consistency and reasonability in the way of Section 4. Fig. 4 shows a fragment of x-city air defense system model. The capability requirements models is converted into the requirements knowledge base, which is composed of the concepts, relations, instances, axioms and assertions.

CFuzzy concepts include: Task, Rule, Performer, Activity, Capability, LocationType, CityAirDefenseTask (CADT), AirDefenseOperationRule (ADOR), AirDefenseForce (ADF), Air-targetDestroyingActivity (ADA), ThreatenEvaluationActivity (TEA), Air-targetInterceptionActivity (AIA), AIC, ImportantCity (IC), RadarUnit (RU), DestroyingDangerousTargetIn20min (DDT20), Danger, Accurate, Early, Wide, Fine.

Fuzzy abstract relations include: ActivityPerformedByPerformer (APBP), RuleConstraintsActivity (RCA), APOC, DesiredEffectGuideActivity (DEGA), DesiredEffectPartOfCapability, CapabilityPerformerManifestation, PerformerPerformsAtLocation, EarlyAlarmGrade.

Fuzzy concrete relation includes: T.

Concrete fuzzy domain includes: [0,1].

Abstract instances include: x-CityAirDefenseTask (x-CADT), x-GroundAirMissileUnit (x-GAMU), x-GroundAirMissileOperationCapability (x-GAMOC), InterceptingX-typeDangerousTarget (IXDT), x-OperationRule (x-OR), DestroyingAir- targetInTolerableTime (DATT), 35, 0.7, 200, 0.8, 6.

Case 1 Consistency checking

The consistency checking is to examine whether there is a logical contradiction to the domain modeling semantics, i.e., whether the instances and relations in the application models, which are typed by the concepts of the fuzzy domain ontology, break the domain constraints declared by the ontology.

Suppose that the modeler wrongly adds in Fig. 4 a relation APBP between the application task object x-ADT and the application performer object x-GAMU. But according to the integrated air warning fuzzy ontology, such relation should only appear between Activity and Performer, and thus the consistency constraint is broken. The clash detection is processed in the following way.

When building the domain rule-base with the algorithm, there are some clash axioms automatically added in the TBox T:

A1 Performer $\subseteq \forall$ APBP.Activity

A2 ADF $\subseteq \forall$ APBP.AIA

A3 ADF $\subseteq$ Performer

A4 AIA $\subseteq$ Activity

A5 CADT $\cap$ AIA $\subseteq \perp$

and also some clash assertions in the ABox A:

a1 x-GAMU: ADF $= 1$

a2 x-CADT: CADT $= 1$

a3 (x-GAMU, x-CADT): APBP $= 1$.

The clash detection is then progressed through such reasoning:

(i) A new assertion a4 can be drawn from assertion a1 and axiom A2:

a4 x-GAMU:$\forall$ APBP.AIA $\geqslant 1$.

(ii) A new assertion a5 can be drawn from assertion a3, a 4 and $\forall \geqslant$ extended rule [19]:

a5 x-CADT: AIA $\geqslant 1$.

(iii) A new assertion a6 can be drawn from assertion a2, a5, axiom A5 and $\cap \leqslant$ extended rule [19]:

a6 x-CADT: AIA $\leqslant 0$.

Here a5 and a6 are a conjugate pair, so that there is a clash in the knowledge base. In another word, we must be careful in building the relations between the application objects by strictly sticking to their domain-specific relation restrictions.

Case 2 Reasonability checking

The reasonability checking is to examine whether there are some unachievable efficiency features in the capability model or whether the efficiency features meet the needs of the mission requirements. It could be a useful tool to make a decision on operation resources deployment.

Suppose that when integrating the x-city's air defense C4ISR system, the program leader wants to make use of the existed resources x-GAMU to achieve the desired effect DATT. In the application model shown in Fig. 4, the vague efficiency feature of DATT is described by the domain class DDT20 with a membership degree of 0.8 given by a domain expert, meaning that system should be designed to achieve the desired effect DDT20 with the possibility 80%.

Besides, the model tells us that x-GAMU has a capability AIC which can provide early alarm in 35 min to meet the need of DATT. However, checking the fuzzy efficiency evaluation function *Early*(35), we can find that fuzzy efficiency value of the early alarm efficiency of the AIC is only 0.65, too low to commit the mission (the desired effect DATT requires no less than 0.8). Such a conclusion is deduced through following reasoning.

Some clash axioms are automatically added in the TBox T when making the model conversion:

A1 ADF $\subseteq \forall$ CPM.AIC

A2 AIC $\subseteq \forall$ CPM^{-}.ADF

A3 AIC $\subseteq \forall$ APOC.AIA

A4 AIA $\subseteq \forall$ APOC^{-}.AIC

A5 DDT20 $\subseteq \forall$ DEGA.AIA

A6 AIA $\subseteq \forall$ DEGA^{-}.DDT20

A7 AIC $\subseteq \forall$ EAG.Early

A8 DDT20 $\cap$ AIC $\cap$ AIA $\cap$ Early $\subseteq \perp$

and also some clash assertions added in the ABox A:

a1 DATT: DDT20 $= 0.8$

a2 IXDT: AIA $= 1$

a3 x-GAMOC: AIC $= 1$

a4 (DATT, IXDT): DEGA $= 1$

a5 (IXDT, x-GAMOC): APOC$^{-} = 1$

a6 (x-GAMOC, x-GAMU): CPM$^{-} = 1$

a7 (x-GAMU, x-GAMOC): CPM $= 1$

a8 (x-GAMOC, 35): EAG $= 1$

a9 35: Early $= 0.65$.

The clash detection is then progressed through such reasoning:

(i) A new assertion a10 can be drawn from the assertion a1 and axiom A5:

a10 DATT:$\forall$ DEGA.AIA $\geqslant 0.8$.

(ii) A new assertion a11 can be drawn from the assertion a4 , a10 and $\forall \geqslant$ extended rule:

a11 IXDT: AIA $\geqslant 0.8$.

(iii) A new assertion a12 can be drawn from the assertion a11 and axiom A4:

a12 IXDT:$\forall$ APOC^{-}.AIC $\geqslant 0.8$.

(iv) A new assertion a13 can be drawn from the assertion a5, a12 and $\forall \geqslant$ extended rule:

a13 x-GAMOC: AIC $\geqslant 0.8$.

(v) A new assertion a14 can be drawn from the assertion a9 and axiom A7:

a14 x-GAMOC:$\forall$ EAG.Early $\geqslant 0.8$.

(vi) a new assertion a15 can be drawn from the assertion a8 , a14 and $\forall \geqslant$ extended rule:

a15 35: Early $\geqslant 0.8$.

Here a15 and a9 are a conjugate pair, so that there is a clash in the knowledge base, i.e., the mission requires that the efficiency value of early alarm is not less than 0.8, while the modeled system can only offer the capability of 0.65. As a result, the engineers may suggest applying more advanced radar system to raise the EarlyAlarmGrade of the AIC.

6. Conclusion

The paper proposes a new method of domain-specific modeling and verification for the C4ISR capability analysis. It suggests the extending UML modeling paradigm by incorporating with fuzzy modeling concepts and the meta-concepts of CMM/DoDAF to develop a domain-specific modeling language. With this modeling language, the domain experts can capture the domain knowledge within the CCMO. And then, the system engineers can continue on requirements modeling work by reusing the domain ontology to model the application requirements at the program level within a unified specification framework defined by the ontology. Then the built models can be converted into knowledge base specified in f-*SHOIN*(D) and verified for both consistency and reasonability through the logical reasoning.

The further research will be on developing a domain-specific rules description language with the fuzzy knowledge query language. Such language may completely relieve the domain experts from the formal description and allow them to define more domain rules with their familiar terms.

References

[1] US Department of Defense. DoD architecture framework version 2.0 (Volume I-II-III). http://www.us.army.mi/suite/page/454707

[2] Object Management Group. UML 2.0 Superstructure Final Adopted Specification. http://www.omg.org/docs/ptc/03-08-02.pdf.

[3] A. Fatolahi, F. Shams. An investigation into applying UML to the Zachman framework. *Journal of Information Systems Frontiers*, 2006, 8(2): 133–143.

[4] UK Ministry of Defence. MOD architecture framework overview version 1.0 (MODAF-M09-002). http://www.modaf. org.uk/.

[5] A. Auger, D. Gouin, J. Roy. Decision support and knowledge exploitation technologies for C4ISR. http://pubs.drdc.gc.ca/PDFS/ unc52/p525876.pdf.

[6] Z. Jin. Ontology-based requirements elicitation. *Chinese Journal of Computers*, 2000, 23(5): 486–492. (in Chinese)

[7] Z. Chen, R. Pooley. Rediscovering Zachman framework using ontology from a requirement engineering perspective. *Proc. of the 33rd Annual IEEE International Computer Software and Applications Conference,* 2009: 3–8.

[8] D. Lichtblau, S. Wartik. Cross-domain ontology resolution in net-centric command and control. http://www.dodccrp.org/events/12th_ICCRTS/CD/ html/paper/136.pdf.

[9] R. Van. Inconsistency management in model-driven engineering. Brussels: Vrije University, 2005.

[10] B. Daniela. Reasoning on UML class diagrams. *Journal of Artificial Intelligence*, 2005, 168 (1): 70–118.

[11] Q. C. Dong, Z. X. Wang, J. Chen, et al. A method of capability model checking based on description logic. *System Engineering and Electronics*, 2010, 32 (3): 533–539. (in Chinese)

[12] Z. M. Ma, W. J. Zhang, W. Y. Ma. Extending object-oriented databases for fuzzy information modeling. *Journal of Information Systems*, 2004, 29(5): 421–435.

[13] Z. M. Ma. *Fuzzy databases modeling with XML*. Kluwer Academic Publishing, 2005.

[14] Z. X. Wang, Q. C. Dong, H. Y. He, et al. Imprecise domain-specific modeling for C4ISR capability requirements analysis. *Proc. of the IEEE International Conference on Information Science and Techonology*, 2011: 33–38.

[15] Z. X. Wang, Q. C. Dong, B. Chen, et al. Towards C4ISR system capability requirements analysis and validation based on UML model. *System Engineering and Electronics*, 2009, 31(9): 2167–2171. (in Chinese)

[16] Z. Y. Li, Z. X. Wang. Towards a multiple ontology framework for requirements elicitation and reuse. *Proc. of the 31st IEEE Computers, Software, and Applications Conference*, 2007: 189–195.

[17] E. Sirin, B. Parsia, B. C. Grau, et al. Pellet: a practical OWL-DL reasoner. *Journal of Web Semantics*, 2007, 5(2): 51–53.

[18] D. Tsarkov, I. Horrocks. FaCT++ description logic reasoner: system description. *Proc. of the 3rd International Joint Conference on Automated Reasoning*, 2006: 292–297.

[19] G. Stoilos, G. Stamou, J. Pan, et al. Reasoning with very expressive fuzzy description logics. *Journal of Artificial Intelligence Research*, 2007, 30(5): 273–320.

Biographies

Qingchao Dong was born in 1982. He is a Ph.D. student of Institute of Command and Automation, PLA University of Science and Technology, where he received his B.S. and M.S. degrees. His research interests are requirements engineering, focusing on specification and formal verification.
E-mail: dongqingchao001@sohu.com

Zhixue Wang was born in 1961. He is a professor of Institute of Command and Automation, PLA University of Science and Technology. He received his B.S. degree from Hefei Polytechnic University, the M.S. degree from National University of Defense and Technology, and used to be a visiting researcher in Faculty of Information Technology, University of Brighton, England. His research interests are software engineering, requirements engineering, theory and technology of command automation, currently focusing on domain-specific modeling and formal verification.
E-mail: wzxcx@163.com

Weixing Zhu was born in 1978. He is a Ph.D. student of Institute of Command and Automation, PLA University of Science and Technology, where he received his M.S. degree. His research interests include theory and technology of command automation, focusing on UML&SysML modeling and simulation.
E-mail: zhuwx999@126.com

Hongyue He was born in 1985. He is a Ph.D. student of Institute of Command and Automation, PLA University of Science and Technology, where he received his B.S. and M.S. degrees. His research interests are requirements engineering, focusing on specification and formal verification.
E-mail: hehy2008@sina.com