http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (11): 2111-2120, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.2111.2120

An Architecture Modeling Method for Supporting Reliability-based Risk Analysis of Ship Command and Control Systems

Zhiqiang Fan and Li Zhang
Software Engineering Institute, Beihang University, Beijing, China

Abstract: Architectural level reliability-based risk analysis is important for finding risky components during the early stage of developing Ship Command and Control Systems (SCCSs). Therefore, an Architecture Modeling Method (AMM) for supporting reliability-based risk analysis of SCCSs is needed. In previous studies, a generic framework was proposed to derive AMMs for large-scale software-intensive systems. The framework has been applied to derive an AMM (named as SAMM) for supporting architecture description during the development process of SCCSs. In this study, based on the framework, an AMM is defined to support reliability-based risk analysis of SCCSs by extending SAMM, named as SAMM4RRA. SAMM4RRA contains one architecture viewpoint and 11 models and an UML/SysML-based architecture description language. An industrial application of SAMM4RRA, along with the subsequent application of the derived SAMM4RRA architecture model (i.e., a deployed SCCS prototype) was conducted to evaluate SAMM4RRA. Results show that SAMM4RRA meets all the requirements for describing the architecture of SCCSs with the purpose of supporting reliability-based risk analysis.

Key words: Architecture modeling, reliability-based risk, ship command and control systems

INTRODUCTION

Warship Combat Systems (WCSs) are systems which concern personnel, weapons and devices, execute tracking, communications, navigation, target identification, data processing, threaten assessment and control of various weapons against enemies. As the key subsystems of WCSs, Ship Command and Control Systems (SCCS) are composed of hardware and software subsystems for gathering information from various sensors, calculating and displaying information and commanding and controlling various weapons to attack threatening targets. SCCSs are often large-scale, complex, real-time and software-intensive systems.

Reliability-based risk refers to the failure probability of a software product in its operational environment and the severity of that failure (Goseva-Popstojanova *et al*., 2003). Architectural-level reliability-based risk analysis of SCCSs aims to find the risky components at the stage of designing SCCSs architecture. Developers and/or testers can therefore pay more attentions on these risky components during the following stages of developing SCCSs, to reduce the whole risk of SCCSs and improve quality of SCCSs.

To analyze reliability-based risk of SCCSs on architecture level, there is a gap between architect modelers and risk analysts. Architect modelers are familiar with the SCCSs architecture, but unfamiliar with the analysis methods that are usually based on a formal or mathematical model (e.g., Cheung's Markov reliability model (Cheung, 1980). On the contrary, risk analysts are familiar with analysis methods, but unfamiliar with SCCSs architecture. Usually risk analysts describe SCCSs architecture via discussions with architect modelers. However, oral communications between them usually lead to ambiguities and inconsistencies and are time consuming. Moreover, when the analysis method is changed, the described models of SCCSs architecture cannot be reused so that the oral communications need to be repeated. Therefore, as required by industrial partner, an AMM, which is independent on any analysis method and easy to learn and understand, to support reliability-based risk analysis of SCCSs is expected.

In previous studies, a generic framework was proposed, named as GCVL (modeling Goal, domain-specific Conceptual model, architecture Viewpoint, architecture description Language), to derive AMMs for large-scale software-intensive systems (LsSiSs) (Fan *et al*., 2013). GCVL has been applied to derive an AMM (named as SAMM) to support architecture description during the development process of SCCSs (Fan *et al*., 2012). In this study, based on GCVL, an AMM is defined for supporting reliability-based risk analysis of SCCSs by extending SAMM, named as SAMM4RRA.

Corresponding Author: Zhiqiang Fan, Software Engineering Institute, Beihang University, Beijing, China Tel+008682339161

BACKGROUND

GCVL framework: In previous work (Fan *et al*., 2013), a generic framework (GCVL) for deriving AMMs for various domains was presented. It includes a model describing concepts related to LsSiSs architecture modeling and their relationships and a process of deriving a complete and consistent domain-specific AMM. We however, in this study, only briefly summarize it.

Key concepts: Modeling Goals define the objectives of architecture description of LsSiSs and are used to derive criteria for justifying whether an AMM is suitable and reasonable. Domain-Specific Conceptual Model presents essential concepts, their attributes and relationships and constraints for describing the architecture of a LsSiS in a particular domain. The conceptual models are derived from stakeholders' concerns, that is, what need to be described depend on what the stakeholders concern with. Architecture Viewpoint, Architecture View and Architecture Model are related to stakeholders and their concerns. An architecture view expresses the system architecture from the perspective of some concerns (Hilliard, 2000, ISO/IEC/IEEE, 2011). An architecture viewpoint governs an architecture view (i.e., establishes the conventions for constructing, interpreting and analyzing the view) (ISO/IEC/IEEE, 2011). An architecture view consists of one or more architecture models. An architecture model can be shared by multiple architecture views. ISO/IEC/IEEE (2011). Different architecture viewpoints/models usually capture different concepts and relationships in the domain-specific conceptual model based on different stakeholders' concerns. Architecture Description Language (ADL) is used to specify the architecture of a LsSiS by specifying model elements to realize the concepts and relationships defined in the domain-specific conceptual model. It also provides diagrams, tables and/or matrices to organize different model elements to support the corresponding architecture viewpoints and models.

Process of deriving a domain-specific AMM: To derive an AMM for a LsSiS in a specific domain, four sequential steps should be followed:

- **Step 1:** Set the modeling goals. As the first step of the process, setting modeling goals should be considered as criteria for selecting suitable modeling methods. Derived modeling goals in this step drive all the other three steps of the process, so that the modeling goals can be derived from three aspect: domain-specific conceptual model, architecture viewpoints and models and architecture description language(s)
- **Step 2:** Derive a domain-specific conceptual model. The conceptual model is derived from stakeholders' concerns. For instance, business stakeholders of LsSiSs usually have concerns such as: 1) what kinds of data that software subsystems produce and 2) which software subsystems consume these data. From these two concerns, concepts Software Subsystem and Data and their relationships Production and Consumption should be derived for the conceptual model. Stakeholders' concerns can be refined iteratively to identify more concepts and their relationships until modeling goals are achieved
- **Step 3:** Specify architecture viewpoints and models. An architecture viewpoint needs to be defined for a group of stakeholders with same or similar concerns. For each architecture viewpoint, more architecture models are needed to cover different concepts and their relationships of the domain-specific conceptual model based on different concerns of stakeholders. It is proposed using an architecture model to frame a reasonably minimum set of concepts and relationships that imply a complete structure or behavior of the concerned system. Rules for checking the consistency of multiple architecture models that frame same concepts are needed because this is important as for example an element appearing in one architecture model must be defined in another architecture model. Besides, guidelines of applying the specified architecture viewpoints and models also need to be provided for stakeholders in this step
- **Step 4:** Define or select ADL(s). According to the derived domain-specific conceptual model and the specified architecture models, to define or select ADL(s), first, it is needed to provide model elements to realize the concepts and relationships in the domain-specific conceptual model (Step 2). Model elements are selected or defined based on the principles of reusing existing language definitions as much as possible and reflecting only necessary domain-specific concepts and relationships (Karsai *et al*., 2009) which can be represented by stereotypes to extend the general modeling language and its profiles (e.g., UML (OMG, 2007) and SysML (OMG, 2008)). Second, it is needed to provide diagrams, tables or matrices to visualize, group and connect the model elements to support the architecture models specified in Step 3

Overview of SCCSs architecture: The architecture of SCCSs is complex in the sense that SCCSs are large-scale, distributed and software-intensive systems, containing many hardware and software subsystems with multiple stakeholders involved. Moreover, SCCSs are typically real-time systems and are required to have high fault-tolerance and high change-tolerance. These properties have an important impact on the design of the SCCS architecture. Fault-tolerance enables a software-intensive system to continue operating properly when one or more its components are failed (Chandhrasekaran and Choi, 2009). Redundancy is one of the mechanisms to increase the fault-tolerance of a system. Inspired by the aspects of fault-tolerance, change-tolerance denotes software's ability to evolve within the bounds of its original design-the degree to which software change is intentional (Bohner, 2007). In this regard, it requires SCCSs to effectively adapt fast-speed functional and technical changes. Therefore, to measure up to this demand, open architecture was introduced to WCSs (including their key subsystems: SCCSs) and later on it was refined into Functional Architecture (FA) and Technical Architecture (TA) (Strei, 2004).

For FA, business stakeholders focus on analyzing communication among subsystems, identifying and extracting functions of SCCSs. For TA, developers concern with implementation of SCCSs. Such an implementation is typically based on component technologies, with which components are designed with high reusability and extensibility to support the quick adaption of functional and technical changes of SCCSs. To achieve this, SCCSs are sliced into numerous functions by business stakeholders and numerous components by developers. Each subsystem of SCCSs is a composition of several functions. Actually, it is an assembly of components via establishing mapping between functions and components. Hence, the functional and technical changes can be quickly adapted by altering corresponding functions and/or components.

SAMM: An AMM to support development process of SCCSs: Along with this idea of FA and TA, the SCCSs development contains the twin-process of function and component fabrication and function and component assembly. The function and component fabrication is the process of functionalizing a series of SCCSs from business stakeholders' perspective, componentizing them from developers' view and establishing mapping of the functions and components. The assembly process however integrates a set of functions and components to build a complete SCCS. There is a practical need to model the architecture of SCCSs during the twin-process to facilitate effective communication among stakeholders. Thereby a complete AMM for SCCSs was defined by applying the GCVL framework (Section 2.1). The characteristics of SAMM are summarized in Table 1.

In step 1, three modeling goals were derived via discussions with stakeholders: (1) Support the architecture description of SCCSs in the context of the twin-process and capture all the architectural concepts and their relationships that stakeholders concern with, (2) Ensure that each stakeholder can concentrate on his/her own concern(s) and (3) Architecture descriptions of SCCSs should be easy to understand since they are mainly used for the communication purpose among stakeholders. In step 2, to satisfy the first modeling goal, stakeholders' development activities during the twin-process were analyzed to identify their 30 concerns, meanwhile, a domain-specific conceptual model, containing 22 concepts and 20 relationships, was derived from these concerns. In step 3, an architecture viewpoint was defined for each kind of stakeholder and in total 22 architecture models were proposed to frame different stakeholder concerns, to satisfy the second modeling goal. Besides, seven consistency rules and a suggested process of applying the architecture viewpoints and models were also proposed. In the guidelines, formalized as an UML activity diagram, 14, six and five modeling

Table 1: Characteristics of SAMM

Steps	SAMM constructs	Data
Step 1	# Modeling goals	3
Step 2	# Stakeholders, their concerns	3, 30
	Domain-specific conceptual model (# Concepts, # Relationships)	(22, 20)
Step 3	# Architecture Viewpoints,	3
	# Architecture Models	22
	# Consistency Rules	7
	Guidelines (# modeling activities of business stakeholders, # modeling activities of developers, # modeling activities of system integrator,)	(14, 4, 5)
Step 4	Model Elements (# Reused UML or SysML Elements, # Stereotypes)	(8, 27)
	Diagrams or Matrices (# UML/SysML Diagrams, # SysML Allocation Matrix)	(8, 1)

activities were defined for business stakeholders, developers and system integrators, respectively, based on the development activities in the twin-process. In step 4, a visual ADL was defined by extending both UML and SysML to satisfy the last modeling goal. The defined ADL includes 35 model elements (eight UML/SysML elements and 27 stereotypes) to realize the domain-specific conceptual model derived in step 2 and eight UML/SysML diagrams and one SysML allocation matrix to support the three architecture viewpoints and 22 models specified in step 3.

The defined SAMM was used by stakeholders to describe architecture of a SCCSs prototype system with modeling tool Magic Draw (No Magic, 2013). The prototype system was developed by Software Engineering Institute of Beihang University (SEI-BUAA), Beijing, China, to demonstrate the model-based integration of SCCSs. The project had around 54 man-months spent and the prototype software has more than 117,000 lines of code. SAMM and the developed architecture description of the prototype system were reviewed by a group of experts of industrial partner, including five business stakeholders, three developers and two system integrators. The experts approved that SAMM satisfies its modeling goals and their expectations.

SAMM4RRA: AN AMM FOR SUPPORTING RELIABILITY-BASED RISK ANALYSIS OF SCCSs

Here, how to apply GCVL framework and extend SAMM to define SAMM4RRA is explained.

Modeling goals: The overall objective is to provide an AMM for risk analyst to describe the architecture of SCCSs for the purpose of supporting reliability-based risk analysis. More specifically, this overall objective is refined into the following three modeling goals:

Goal 1: Support to describe architectural-level concepts that are related to reliability-based risk and their relationships and reuse the domain-specific conceptual model of SAMM as much as possible

Goal 2: Reuse the architecture viewpoints and models of SAMM, that are related to reliability-based risk, so that risk analysts do not need to concern with all architecture viewpoints and models of SAMM

Goal 3: The ADL of SAMM4RRA should be defined based on the languages that are widely used and easy to learn and understand. SAMM4RRA is required to be independent to any reliability-based risk analysis method so that the described models using SAMM4RRA can be reused when the analysis method is changed. When an analysis method is chosen, part of or the whole described models can be transformed into the models that can be inputted to the analysis method

Domain-specific conceptual model: According to GCVL, to derive a domain-specific conceptual model, it is needed to conduct an analysis on risk analysts' concerns on SCCSs architecture and extract concepts and their relationships according to the first modeling goal .

First, concerns of the risk analysts are identified via around eight-hour discussion with them, as listed in Table 2. Reliability-based risk analysis of SCCSs aims to find the risky components that have sensitive influences on the reliability-based risk of a SCCS. Therefore, analysts have the concerns C-A1 and C-A2, as listed in Table 2. In SCCSs, a concrete SCCS is composed of several software subsystems, a software subsystem consists of some functions, and a function is implemented by one or more components. Since the reliability-based risk of an element (e.g., a SCCS, a software subsystem or a function) depends on the executed probability and reliability-based risk of each of its sub-elements (e.g., software subsystems, functions and components) (Goseva-Popstojanova *et al*., 2003), concern C-A1 can be refined into concerns C-A1-1 to C-A1-6, C-S1, C-S2, C-S4 and C-D2 (Table 2). Note that C-S1, C-S2 and C-S4 are system integrators' concerns and C-D2 is a concern of developers, in the SAMM method (Fan *et al*., 2012). This means that system integrators, developers and analysts have some common concerns.

The reliability-based risk of a component depends on itself as it is the atomic element in SCCSs. From the definition of reliability-based risk (i.e., reliability-based risk refers to the probability that the software product will fail in the operational environment and the adversity of that failure (Goseva-Popstojanova *et al*., 2003), one can see that it contains two factors: Failure Probability and Failure Severity. The failure probability of a component depends on its dynamic complexity and couplings (Goseva-Popstojanova *et al*., 2003). Dynamic complexity of a component depends on the states of this component and transitions among them and couplings of a component depend on its interactions with other components (Goseva-Popstojanova *et al*., 2003; Fan *et al*., 2008). In SCCSs, some components are new designed and others are reused from legacy systems. Therefore, the Technology Readiness Level (TRL) (Department of Defense, 2009) of a component is also an

Table 2: Concerns of Analysts

No.	Concerns
C-A1	What's the reliability-based risk of a SCCS?
C-S1 and C-S2	Which software subsystems are deployed for this SCCS?
C-A1-1	What's the executed probability of each software subsystem?
C-A1-2	What's the reliability-based risk of each software subsystem?
C-S4	For a software subsystem, what functions are assigned to it?
C-A1-3	What's the executed probability of each function?
C-A1-4	What's the reliability-based risk of each function?
C-D2	For a function, which components implement it?
C-A1-5	What's the executed probability of each component?
C-A1-6	What's the reliability-based risk of each component?
C-A1-7	For a component, what's the failure probability of it?
C-A1-8	What's dynamic complexity of this component? (depends on its states and transitions)
C-A1-9	What's the coupling of this component? (depends on its interactions with other components)
C-A1-10	What's the TRL of this component?
C-A1-11	For a component, what's the failure severity of it?
C-A1-12	For a function, what's the failure severity of it?
C-A1-13	For a software subsystem, what's the failure severity of it?
C-A2	What's the reliability-based risk sensibility of each component?

SCCS and TRL, respectively standard for ship command and control system and technology readiness level

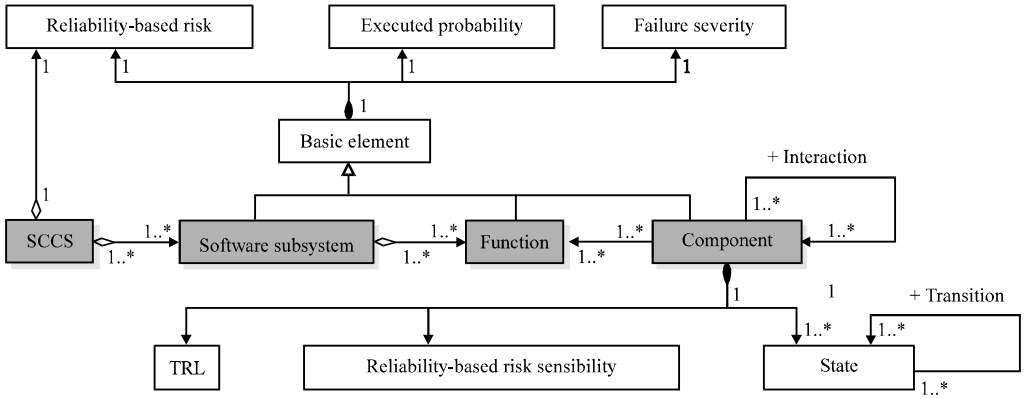

Fig. 1: Domain-specific conceptual model of SAMM4RRA

important factor to its reliability-based risk. Hence, concerns C-A1-7 to C-A1-10 are derived. In SCCSs, The failure severity of a component is derived from its implemented functions and a failure severity of a function is derived from its related software subsystems. Hence, concerns C-A1-11 to C-A1-13 are derived.

Second, based on the identified concerns in Table 2, architectural-level concepts related to reliability-based risk of SCCSs and their relationships can be derived to achieve the first modeling goal, including SCCS, Software Subsystem, Function, Component, Reliability-based Risk, Executed Probability, Failure Severity, States of components and Transitions among them, TRL and Reliability-based Risk Sensibility of components and Interactions among components, as shown in Fig. 1, where, the concepts in gray boxes are reused from SAMM.

Architecture viewpoints and models: According to the third step of deriving a domain-specific AMM of GCVL, first, Reliability-based Risk Analysis Viewpoint is defined for risk analysts. To define architecture models for this viewpoint, based on the suggestion in the third step of GCVL (i.e., using an architecture model to frame a reasonably minimum set of concepts and relationships that imply a complete structure or behavior of the concerned system), following three steps are proposed:

- **Step 1:** List all relationships among the concepts in the domain-specific conceptual model, as listed in the first row of Table 3
- **Step 2:** Group the listed relationships that imply a complete structure or behavior, along with their associated concepts. For example, relationships between concepts Failure Severity and Software Subsystem, between concepts Failure Severity and Function, between concepts Failure Severity and Component are grouped because in SCCSs: 1) each software subsystem is a combination of several

Table 3: Concepts and relationships framed by reliability-based risk analysis viewpoint

Categories	Relationships among Concepts
All relationships	SCCS and SS, SS and F, F and C, SCCS and RR, SS and RR, F and RR, C and RR, SS and EP, F and EP, C and EP, SS and FS, F and FS, C and FS, C and S, S and S, C and C, C and TRL, C and RRS
Grouped relationships	SCCS and RR, SS and RR, F and RR, C and RR, SS and EP, F and EP, C and EP, SS and FS, F and FS, C and FS, C and S, S and S
Ungrouped relationships	SCCS and SS, SS and F, F and C, C and C, C and TRL, C and RRS

SCCS, SS, F, C, RR, EP, FS, S, TRL and RRS, respectively standard for ship command and control system, software subsystem, function, component, reliability-based risk, executed probability, failure severity, state, technology readiness level and reliability-based risk sensibility

Table 4: Architecture model for reliability-based risk analysis viewpoint

Architecture models		Framed relationships	Concerns in Table 2
RRAV-1	Reliability-based Risk Model	SCCS and RR, SS and RR, F and RR and RR,	C-A1-2,4,6
RRAV-2	Executed Probability Model	SS and EP, F and EP, C and EP	C-A1-1,3,5
RRAV-3	Failure Severity Model	SS and FS, F and FS, C and FS,	C-A1-11, 12, 13
RRAV-4	Component State Model	C and S, S and S	C-A1-8
RRAV-5	Component TRL Model	C and TRL	C-A1-10
RRAV-6	Component Risk Sensibility Model	C and RRS	C-A2
IV-5	Function Allocation Model	SS and F	C-S4
TV-3	Function Realization Model	F and C	C-D2
TV-4	Component Interaction Model	C and C	C-A1-9
IV-1	Hardware Subsystem Selection Model	SCCS and SS	C-S1
IV-2	Software Subsystem Deployment Model		C-S2

RRAV, IV, TV, SCCS, SS, F, C, RR, EP, FS, S, TRL and RRS, respectively standard for reliability-based risk analysis viewpoint, integration viewpoint, technical viewpoint, ship command and control system, software subsystem, function, component, reliability-based risk, executed probability, failure severity, state, technology readiness level and reliability-based risk sensibility

Table 5: Consistency Rules among the 11 architecture model

Rules	Related Models in Table 4	Framed concepts
A SCCS in RRAV-1 must be described in IV-1	RRAV-1, IV-1	SCCS
A software subsystem in RRAV-1 to RRAV-3 must be described in IV-2	RRAV-1 to RRAV-3, IV-2	Software Subsystem
A function in RRAV-1 to RRAV-3 must be described in IV-5	RRAV-1 to RRAV-3, IV-5	Function
A component in RRAV-1 to RRAV-6 must be described in TV-3	RRAV-1 to RRAV-6, TV-3	Component

RRAV, IV, TV and SCCS, respectively standard for reliability-based risk analysis viewpoint, integration viewpoint, technical viewpoint, ship command and control system

functions and each function is implemented by one or more components, which forms a tree structure among software subsystems, functions and components; 2) the failure severity of a function or a component depends on failure severities of its related software subsystem or functions

- **Step 3:** Define an architecture model for each group of relationships or each ungrouped and individual relationship

By following the above three steps, totally 11 models are defined for Reliability-based Risk Analysis Viewpoint (RRAV), as listed in Table 4. According to the second modeling goal in Section 3.1, five architecture model of SAMM (Fan *et al.*, 2012) are reused, i.e., IV-5, TV-3, TV-4, IV-1 and IV-2.

To ensure the consistency among the architecture models that frame same concepts, four consistency rules are provided in Table 5. For example, RRAV-1 to RRAV-3 and IV-5 frame same concept Function and all the functions of a SCCS described in IV-5, therefore, we can derive a rule "A function in RRAV-1 to RRAV-3 must be described in IV-5".

When using the 11 models to describe architecture of a SCCS, first, if the SCCS has been described using SAMM, directly reuse the SAMM models in IV-1, IV-2, IV-5, TV-3 and TV-4, otherwise, use IV-1, IV-2, IV-5, TV-3 and TV-4 to describe the SCCS, software subsystem, functions, components and their relationships; second, use RRAV-2 and RRAV-3 to, respectively describe executed probabilities and failure severities of software subsystems, functions and components; third, use RRAV-4 and RRAV-5 to describe states and TRLs of components; finally, use RRAV-1 and RRAV-6 to describe the analysis results when a analysis method is selected to analyze the reliability-based risk of the SCCS. The suggested process is formalized as an UML activity diagram as shown in Fig. 2.

SARRS: An ADL supporting analysis of reliability-based risk of SCCSs: According to the third modeling goal in the ADL of SAMM4RRA (called SARRS) needs to be defined based on the languages that are widely used and easy to learn and understand. As the ADL of SAMM (Called CPaS) is defined based on UML and SysML, SARRS is defined by extending CPaS. As there is only one kind of stakeholder, i.e., risk analyst, only one

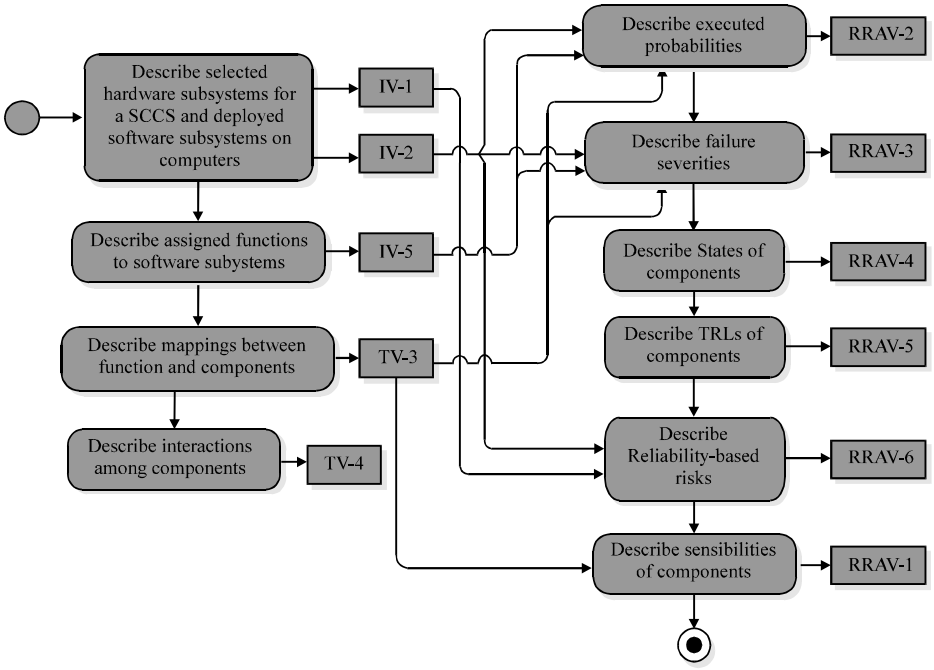

Fig. 2: Suggested process of applying the 11 models

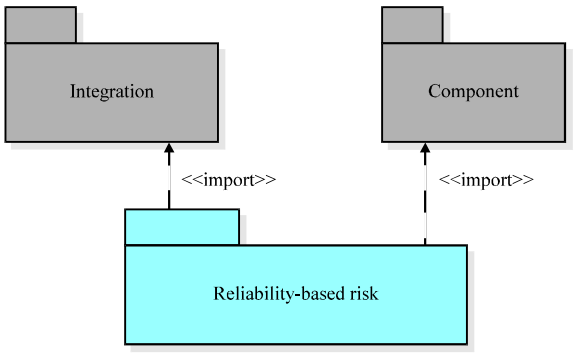

Fig. 3: Packages of SARRS

package is defined (i.e., Reliability-based Risk package) for SARRS. Since SAMM4RRA reused concepts SCCS, Software Subsystem, Function, Component and their relationships from SAMM, package Reliability-based Risk imports packages Integration and Component of CPaS, as shown in Fig. 3.

First, model elements of SARRS need to be defined to realize the domain-specific conceptual model. According to steps P1a (i.e., if an UML/SysML element has the same meaning as a concept in the domain-specific conceptual model, reuse this element to realize this concept) and P2b (i.e., if an UML/SysML element has the same meaning as a relationships in the domain-specific conceptual model, reuse this element to realize this relationships) of SAMM (Fan *et al*., 2012), UML/SysML elements "State" and "Transition" are reused to realize the concept State and relationship Transition in the conceptual model (Fig. 1). For concepts Reliability-based Risk, Executed Probability, TRL or Reliability-based Risk Sensibility, there is not an element of UML or SysML, which has same or similar meaning, structure, behaviors or relationships to one of them. These concepts are more suitable to be considered as attributes of concepts Software Subsystem, Function and Component. Therefore, stereotypes are defined to realize these concepts, as listed in Table 6.

Second, based on the principles P2 (i.e., select an UML/SysML diagram for an architecture model framing the concepts and/or relationships that correspond to elements usually presented in this diagram) and P3 (i.e., if the architecture model frames concepts and/or relationships that correspond to elements often presented in different UML/SysML diagrams, select the diagram that describe the main concepts of this architecture model) in of SAMM (Fan *et al*., 2012), three UML or SysML diagrams are selected for the six new defined architecture models, as listed in Table 7. For some architecture models, for examples, the Executed Probability Model (RRAV-2) and Failure Severity Model (RRAV-3), tables are also suggested for them because tables are clearer than UML or SysML diagrams to describe them. However, the architecture described using UML or SysML diagrams could be directly inputted to some reliability-based risk analysis methods. Therefore, both UML/SysML diagrams and tables are provided to support these models.

Table 6: Model elements of SARRS

Concepts and relationships	Stereotypes	Metaclasses	Attributes	Package
SCCS	RRA_SCCS	Class	RR: String	Reliability-based Risk
Software Subsystem	RRA_SS	Class	RR: String EP: String FS: String	
Function	RRA_F	UseCase		
Component	RRA_C	Component (UML)	RR: String; EP: String FS: String; TRL: String RRS: String	
State (P1a)		State		
Transition (P2b)		Transition		

RRA, SCCS, RR, EP, FS, TRL and RRS, respectively standard for reliability-based risk analysis, ship command and control system reliability-based risk, executed probability, failure severity, technology readiness level and reliability-based risk sensibility

Table 7: Diagrams and Tables of SARRS

Architecture	Models	UML/SysML diagrams	Table
RRAV-1	Reliability-based Risk Model (P2)	UML/SysML Usecase Diagram, UML Component Diagram	√
RRAV-2	Executed Probability Model (P3)	UML/SysML Usecase Diagram, UML Component Diagram	√
RRAV-3	Failure Severity Model (P2)	UML/SysML Usecase Diagram, UML Component Diagram	√
RRAV-4	Component State Model (P3)	UML/SysML State Diagram	
RRAV-5	Component TRL Model (P3)	UML Component Diagram	√
RRAV-6	Component Risk Sensibility Model (P2)	UML Component Diagram	√

RRAV and TRL, respectively standard for reliability-based risk analysis viewpoint and technology readiness level

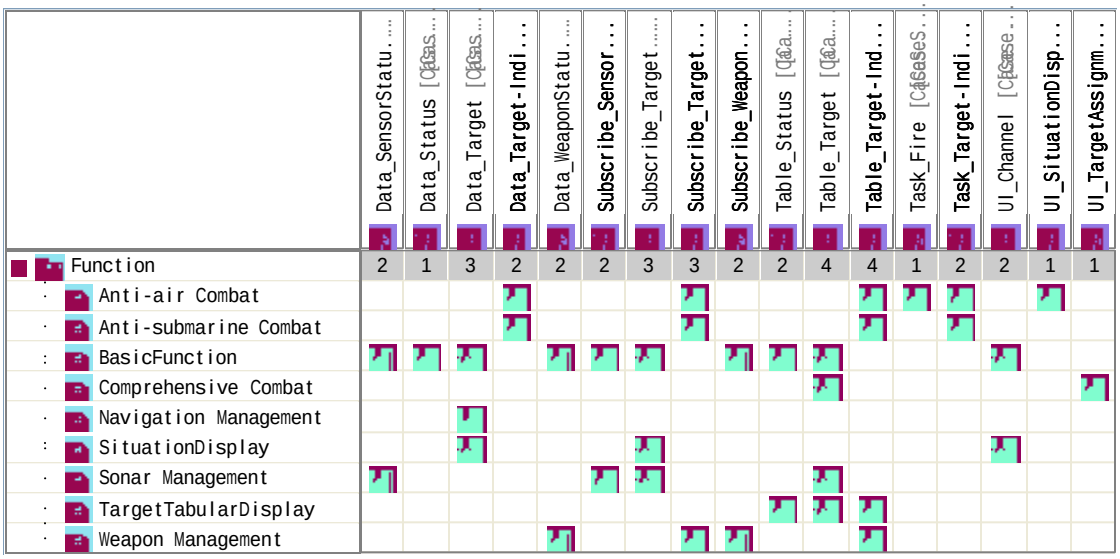

Fig. 4: Implementation of Functions by Components (TV-3)

EVALUATION

Here, a case study is presented to evaluate SAMM4RRA and the SAMM4RRA's satisfaction to its modeling goals is discussed.

Case study: To evaluate the applicability of SAMM4RRA in modeling the SCCS architecture, SAMM4RRA is used to describe the SCCSs prototype systems. Since two of the defined architecture models (i.e., RRAV-1 and RRAV-6) are only used for describing the results of reliability-based risk analysis of a SCCS (e.g., risk values of software subsystems, functions or components) and SAMM4RRA is independent to any analysis method, only another nine architecture models of SAMM4RRA are used to describe the architecture of the prototype system. As architecture of the prototype system has already been described using SAMM (Fan *et al*., 2012), the architecture description in IV-1, IV-2, IV-5, TV-3 and TV-4 are reused directly. For example, relationships between 17 components and nine functions are described in TV-3 (Fig. 4).

Since all the architecture elements of this prototype system are very large. Only part of the architecture elements of the prototype system are captured in each of

Table 8: An example of executed probabilities

Software subsystems, functions and components	Executed probabilities (%)
Information Software Subsystem	20
Situation Display	16
Data_Target	3.2
UI_SituationDisplay	8
Subscribe_Target	4.8
Navigation Management	4
Data_Target	4

Table 9: Part of characteristics of the prototype system architecture

Architecture Models		#Captured architecture elements of the prototype system
RRAV-2	Executed Probability Model	Executed Probabilities of one software subsystem, two functions and four components
RRAV-3	Failure Severity Model	Failure Severities of one software subsystem, three functions and eight components
RRAV-4	Component State Model	Five states of a component and 10 transitions among them
RRAV-5	Component TRL Model	TRLs of eight components

RRAV and TRL, respectively standard for reliability-based risk analysis viewpoint and technology readiness level

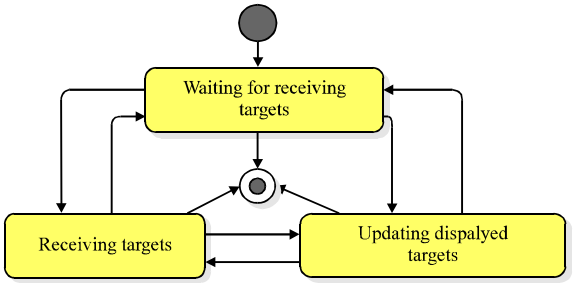

Fig. 5: States of a component (RRAV-4)

the nine architecture models. For examples, five states of a component and 10 transitions among them were described using the Component State Model (RRAV-4) as shown in Fig. 5 and executed probabilities of one software subsystem, two functions and four components were described using Executed Probability Model (RRAV-2) as listed in Table 8. The described architecture elements of the prototype system are summarized in Table 9 for reference. Finally, the developed architecture description is discussed with industrial partner. They approved that SAMM4RRA is applicable to describe SCCSs architecture for the purpose of supporting reliability-based risk analysis of SCCSs and satisfies its modeling goals and their expectations.

Discussion: Three modeling goals of SAMM4RRA were set via discussion with risk analysts. To achieve the first modeling goal (i.e., support to describe architectural-level concepts related to reliability-based risk and their relationships, and reuse the conceptual model of SAMM as much as possible), the characteristics of SCCSs architecture were analyzed. Based on the analysis, 17 concerns of risk analysts are derived and 10 concepts related to reliability-based risk and their 18 relationships were extracted. Four concepts and their four relationships are reused from SAMM. To satisfy the second modeling goal (i.e., reuse the architecture viewpoints and models of SAMM, that are related to reliability-based risk, so that risk analysts do not need to concern with all architecture viewpoints and models of SAMM), the Reliability-based Risk Analysis Viewpoint is defined for risk analysts and in total 11 architecture models (five of them are reused from SAMM) were proposed. To meet the third modeling goal (i.e., the ADL of SAMM4RRA should be defined based on the languages that are widely used and easy to learn and understand), a visual ADL was defined by extending UML and SysML. Both UML and SysML are standard visual language and used widely in industry, thus making SAMM4RRA easy to learn, use and understand. The domain experts of the industrial partner also subjectively approved that SAMM4RRA satisfies its modeling goals and their expectations.

CONCLUSION AND FUTURE WORK

Architectural-level reliability-based risk analysis is important for finding risky components during the early stage of developing SCCSs. To analyze reliability-based risk of SCCSs on architecture level, an AMM is required for describing SCCSs architecture. By applying the GCVL framework, a complete AMM (i.e., SAMM4RRA) is derived for supporting reliability-based risk analysis of SCCSs by reusing SAMM, including a domain-specific conceptual model, a viewpoint and 11 architecture models, and an UML/SysML-based ADL. SAMM4RRA can be used to describe architectural-level concepts related to reliability-based risk and their relationships and the described models are easy to understand by risk analysts and can be reused when the analysis method is changed.

In future, this study can be extended by apply SAMM4RRA to analyze reliability-based risk of a SCCS by selecting one or more reliability-based risk analysis methods.

ACKNOWLEDGMENTS

This study is supported by the Project (SKLSDE-2012ZX-13) of the State Key Laboratory of Software Development Environment, China. We would like to thank all members of our team. Also, we thank the reviewers for their valuable suggestions.

REFERENCES

Bohner, S., 2007. An era of change-tolerant systems. Computer, 40: 100-102.

Chandhrasekaran, K.V. and E. Choi, 2009. Fault tolerance for embedded control system. Proceedings of the 9th International Symposium on Communications and Information Technology, September 28-30 2009, Icheon, pp: 1316-1320.

Cheung, R.C., 1980. A user-oriented software reliability model. IEEE Trans. Software Eng., 6: 118-125.

Department of Defense, 2009. Technology Readiness Assessment (TRA) Deskbook. http://www.skatelescope.org/public/2011-11-18_WBS-SOW_Development_Reference_Documents/DoD_TRA_July_2009_Read_Version.pdf

Fan, Z., T. Yue and L. Zhang, 2012. An architecture modeling methodology for ship command and control systems. SEI-BUAA Technical Report.

Fan, Z., T. Yue and L. Zhang, 2013. A generic framework for deriving architecture modeling methods for large-scale software-intensive systems. Proceedings of the 28th Annual ACM Symposium on Applied Computing, March 18-22, 2013, Coimbra, Portugal, pp: 1750-1757.

Fan, Z., X. Zhou, J. Chen and Y. Dong, 2008. A novel model for component-based software reliability analysis. Proceeding of 11th High Assurance Systems Engineering Symposium, HASE 2008, December 3-5, 2008, IEEE., pp: 303-309.

Goseva-Popstojanova, K., A.E. Hassan, A. Guedem, W. Abdelmoez and D.E.M. Nassar *et al*., 2003. Architectural-level risk analysis using UML. Trans. Softw. Eng., 29: 946-960.

Hilliard, R., 2000. IEEE, Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE-Std http://www.win.tue.nl/~wsinmak/Education/2II45/software-architecture-std1471-2000.pdf

ISO/IEC/IEEE, 2011. 42010-2011-Systems and software engineering-architecture description. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6129467

Karsai, G., H. Krahn, C. Pinkernell, B. Rumpe and M. Schindler *et al*., 2009. Design guidelines for domain specific languages. Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling. October 25-26, 2009, New York, USA.

OMG, 2007. OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2. http://webcourse.cs.technion.ac.il/234321/Spring2009/ho/WCFiles/07-11-02-UML%202.1.2-Superstructure.pdf

OMG, 2008. OMG Systems Modeling Language (OMG SysML). Version 1.1. http://www.sysml.org/docs/specs/OMGSysML-v1.1-08-11-01.pdf

Strei, T.J., 2004. Open architecture - overview of naval open systems. The Plans, The Approach, The Promise. http://sstc-online.org/2004/PDFFiles/TJS6252.pdf