Schematic Generation of English-prose Semantics for a Risk Analysis Language Based on UML Interactions

Gencer Erdogan*, Atle Refsdal* and Ketil Stølen*

*Department for Networked Systems and Services, SINTEF ICT,
PO Box 124 Blindern, N-0314 Oslo, Norway
Email: {gencer.erdogan, atle.refsdal, ketil.stolen}@sintef.no
†Department of Informatics, University of Oslo,
PO Box 1080 Blindern, N-0316 Oslo, Norway

Abstract—To support risk-driven testing, we have developed CORAL, a language for risk analysis based on UML interactions. In this paper, we present its semantics as a translation of CORAL diagrams into English prose. The CORAL semantics is developed to help software testers to clearly and consistently document, communicate and analyze risks in a risk-driven testing process. We first provide an abstract syntax and a translation algorithm. Then, we evaluate the approach based on some examples. We argue that the resulting English prose is comprehensible by testers, is consistent with the semantics of UML interactions, and has a complexity that is linear to the complexity of CORAL diagrams in terms of size.

Keywords-risk analysis language; risk-driven testing; UML interaction; sequence diagram; CORAL diagram;

I. INTRODUCTION

In earlier work, we presented a systematic method for designing test cases by making use of risk analysis [1], [2]. As part of the method, we also introduced a risk analysis language based on UML interactions which we refer to as CORAL. CORAL extends UML interactions with constructs for representing risk-related information in sequence diagrams, and it is specifically developed to support software testers in a risk-driven testing process. As we explain in [1], [2], testers may use CORAL in three consecutive steps to identify, estimate, and evaluate risks. The graphical icons representing risk-related information in CORAL are based on corresponding graphical icons in CORAS [3]. This is a deliberate design decision because the graphical icons in CORAS are empirically shown to be cognitively effective [4]. However, without supporting natural-language semantics, CORAL diagrams, i.e., interactions represented by CORAL constructs, may be interpreted differently by different testers. Thus, in order to help software testers to clearly and consistently document, communicate and analyze risks, we present a structured approach to generate the semantics of CORAL diagrams in terms of English prose. We evaluate the approach based on some examples.

The remainder of this paper is organized as follows. Section II lists the success criteria our approach aims to fulfill. Section III gives a stepwise explanation of the approach, and presents the examples we base our evaluation on. Section IV elaborates on the fulfillment of the success criteria. Section V provides an overview of related work. Finally, Section VI gives some concluding remarks.

II. SUCCESS CRITERIA

There are three key design decisions that shape our success criteria.

First, the main target audience of CORAL is software testers. CORAL is supposed to be used by testers to document, communicate and analyze risks in a risk-driven testing process. Thus, our first success criterion is: The resulting English prose must be comprehensible by software testers when conducting risk analysis.

Second, CORAL is based on UML interactions and only extends UML interactions with constructs representing risk-related information. Thus, our second success criterion is: The CORAL semantics of the constructs inherited from UML interactions must be consistent with their semantics in the UML standard.

Third, the approach must ensure scalability. Thus, our third success criterion is: The complexity of the resulting English prose must scale linearly with the complexity of CORAL diagrams in terms of size.

III. APPROACH

Inspired by CORAS [3], we generate the English-prose semantics in three consecutive steps, as shown in Figure 1. In Step 1, we translate a CORAL diagram into a corresponding textual representation. This step takes a CORAL diagram as input. First, for each construct in the CORAL diagram, we identify its corresponding syntactical element in the abstract syntax of CORAL. Second, we replace the variables in the syntactical element with content from the construct in the diagram. Third, we perform a structured approach to generate the semantics of CORAL diagrams in terms of English prose. We evaluate the approach based on some examples.

The remainder of this paper is organized as follows. Section II lists the success criteria our approach aims to fulfill. Section III gives a stepwise explanation of the approach, and presents the examples we base our evaluation on. Section IV elaborates on the fulfillment of the success criteria. Section V provides an overview of related work. Finally, Section VI gives some concluding remarks.
We use the following undefined terms in the grammar: identifier, asset lifeline, frequency, conditional ratio, and consequence. The term identifier is assumed to represent any alphanumeric string. The term asset lifeline is assumed to represent any alphanumeric string describing the name of an asset lifeline. The term frequency is assumed to represent any alphanumeric string describing a likelihood value in terms of frequency. The term conditional ratio is assumed to represent an expression that evaluates to a subset of $\mathbb{R}_{\geq 0}$ (the non-negative real numbers including 0). The term consequence is assumed to represent an alphanumeric string describing the impact an unwanted incident has on an asset.

- risk interaction = message | weak sequencing | potential alternatives | referred interaction | parallel execution | frequency assignment | conditional ratio assignment | consequence assignment;

- risky message = rm(identifier, transmitter lifeline, receiver lifeline, message category);
- unwanted incident message = uim(identifier, transmitter lifeline, asset lifeline);
- transmitter lifeline = general lifeline | deliberate threat lifeline;
- receiver lifeline = general lifeline | deliberate threat lifeline;
- general lifeline = gl(identifier);
- deliberate threat lifeline = dtl(identifier);
- message category = non-manipulative | manipulative;
- non-manipulative = general;
- manipulative = new | alter;
- weak sequencing = seq({risk interaction}^{-});
- potential alternatives = alt({risk interaction}^{-});
- referred interaction = ref(identifier);
- parallel execution = par({risk interaction}^{-});
- frequency assignment = fa(kind, risky message, frequency) | fa(transmission, unwanted incident message, frequency);
- kind = transmission | reception;
- transmission = !;
- reception = ?;
- conditional ratio assignment = cra((kind, risky message), conditional ratio, (kind, risky message));
- consequence assignment = ca(unwanted incident message, consequence);
B. English-prose Semantics of CORAL

The English-prose semantics of a syntactical element is defined by the function \( [\cdot] \), which is defined below for the excerpt of the abstract syntax presented in Section III-A. Let the syntactical variables:

- \( d \) range over risk interaction
- \( id \) range over identifier
- \( t \) range over transmitter lifeline
- \( r \) range over receiver lifeline
- \( al \) range over asset lifeline
- \( f \) range over frequency
- \( cr \) range over conditional ratio
- \( c \) range over consequence

The pair of square brackets, ‘\( [\cdot] \)’, is a part of the semantics that is used to enclose an operand.

\[
\begin{align*}
\text{seq}(d_1, d_2, \ldots, d_m) &= [d_1] \text{ weakly sequenced by } [d_2] \text{ weakly sequenced by } \ldots \text{ weakly sequenced by } [d_m] \\
\text{alt}(d_1, d_2, \ldots, d_m) &= \text{either } [d_1] \text{ or } [d_2] \text{ or } \ldots \text{ or } [d_m] \\
\text{ref}(id) &= \text{refer to interaction: } id \\
\text{par}(d_1, d_2, \ldots, d_m) &= [d_1] \text{ parallelly merged with } [d_2] \text{ parallelly merged with } \ldots \text{ parallelly merged with } [d_m] \\
\text{rm}(id, t, r, \text{general}) &= \text{the message } id \text{ is transmitted from } [t] \text{ to } [r] \\
\text{gl}(id) &= id \\
\text{dtl}(id) &= \text{the deliberate threat } id \\
\text{rm}(id, t, r, \text{new}) &= \text{the new message } id \text{ is transmitted from } [t] \text{ to } [r] \\
\text{rm}(id, t, r, \text{alter}) &= \text{the altered message } id \text{ is transmitted from } [t] \text{ to } [r] \\
\text{uil}(id, t, al) &= \text{the unwanted incident } id \text{ occurring on } [t] \text{ impacts asset } al \\
\text{fa}(?, (id, t, r, \text{new}), f) &= \text{the reception of the new message } id \text{ by } [r] \text{ from } [t] \text{ occurs with frequency } f \\
\text{fa}(!, (id, t, al), f) &= \text{the unwanted incident } id \text{ occurring on } [t] \text{ impacts asset } al \text{ with frequency } f \\
\text{cra}(!, (id, t, r, \text{alter}), cr, (?, (id, t, r, \text{alter}))) &= \text{the transmission of the altered message } id \text{ from } [t] \text{ leads to its reception by } [r] \text{ with conditional ratio } cr \\
\text{ca}((id, t, al), c) &= \text{the unwanted incident } id \text{ occurring on } [t] \text{ impacts asset } al \text{ with consequence } c
\end{align*}
\]

We demonstrate the schematic translation of CORAL diagrams into English prose by first giving some examples of CORAL diagrams (see Figure 2), and then translating these diagrams into their corresponding English prose using the translation functions (see Figure 3). The CORAL diagrams in Figure 2 were obtained by applying our method [1], [2] on a guest book that is available in the Damn Vulnerable Web Application [8].

IV. DISCUSSION

In this section, we discuss the fulfillment of the three success criteria given in Section II.

A. The resulting English prose must be comprehensible by software testers when conducting risk analysis

The comprehensibility of the resulting English prose is supported both from a general viewpoint and from a software testing viewpoint.

From a general viewpoint, we observe the following two points. First, the structure of the translations in Figure 3 is similar to the structure of their corresponding CORAL diagrams in Figure 2. In particular, the ordering of the translated CORAL constructs is maintained. For example, let us consider the translation in Figure 3a. The first sentence states: “The new message forgedURLReplacingMsgWithXSScript is transmitted from the deliberate threat Hacker to C”. By comparing the translation in Figure 3a to its corresponding diagram in Figure 2a, we see that the first sentence corresponds to the first message in the diagram. Similarly, we see that the second sentence in Figure 3a corresponds to the second message in Figure 2a, and so on. Second, the user-defined text is unchanged in the translations. By user-defined text, we mean the text typed in CORAL diagrams, such as the text on messages, lifelines, frequency assignments, consequence assignments, and so on.

From a software testing viewpoint, we observe that risk-related concepts from CORAL are integrated with concepts from UML interactions, in the resulting English prose. UML interaction is among the top three modeling language within the testing community and is often used for testing purposes [9]. It is therefore reasonable to assume that testers understand the concepts from UML interactions. Moreover, we find it reasonable to assume that testers also comprehend the risk-related concepts we introduce in CORAL, such as altered messages and messages representing unwanted incidents, because these are concepts that are also known within the testing community. For example, in fuzz testing, the expected behavior of a system is altered by providing invalid, unexpected, or random data, which may lead to unwanted incidents [10].

To illustrate this, let us consider the first message in Figure 2d. This message represents an altered message. In CORAL, an altered message is a message in the system model which has been altered due to unexpected system...
behavior or unexpected input data. Figure 3d shows the corresponding translation as: “The altered message submit(name,XSSscript) is transmitted from GBForm to GBDatabase”. The translation shows that we have a message that is transmitted between two lifelines (concepts from UML interactions), and we have added that the message is altered (risk-related concept from CORAL).

B. The CORAL semantics of the constructs inherited from UML interactions must be consistent with their semantics in the UML standard

The CORAL constructs inherited from UML interactions are messages and the interaction operators: seq, ref, alt, par and loop. The interaction operator weak sequencing (seq) is defined and related to CORAL in Section III.

According to UML [5], a message defines a particular communication between lifelines of an interaction. The signature of a message is the content specification of that very message. A message also defines the lifeline from which it is sent (i.e., the transmitter lifeline) and the lifeline receiving it (i.e., the receiver lifeline). Thus, a message may be defined as the triple \((id, t, r)\), where \(id\) represents the signature, \(t\) represents the transmitter lifeline, and \(r\) represents the receiver lifeline. We define a message in a similar manner. However, we also distinguish between the category of a message, i.e., whether it is manipulative, non-manipulative, or an unwanted incident, as explained in Section III. As we can see from the translations in Figure 3, the English prose of messages are generated according to their category, and contain information about the message signature, the lifeline transmitting the message and the lifeline receiving the message.

According to UML [5], an interaction-use (ref) refers to an interaction. The interaction-use is shorthand for copying the contents of the referred interaction where the interaction-
Figure 3. Schematic translation of the corresponding CORAL diagrams in Figure 2 into English prose.

use is. Figure 2b shows an example of an interaction-use named Sign guest book. The interaction referred to by this interaction-use is shown in Figure 2e. We use the term ‘*refer to interaction*’ to denote an interaction-use, as shown in the translations in Figures 3a and 3b.

According to UML [5], the interaction operator potential alternatives (alt) designates that the operands represent a choice of behavior. UML requires that the chosen operand must have an explicit or implicit guard expression that evaluates to true. An implicit true guard is implied if the operand has no explicit guard. In CORAL, we currently allow only for the usage of implicit true guards. However, the syntax and semantics of CORAL is easily extendable to support explicit guards as well. As shown in Figure 3a, we use the term ‘*either*’ in front of the first operand of an alt operator, and then the term ‘*or*’ between each subsequent operand to reflect the disjunctive behavior of the alt operator.

According to UML [5], the interaction operator parallel execution (par) designates a parallel merge between the behaviors of the operands. A parallel merge defines a set of traces that describes all the ways that events of the operands may be interleaved without obstructing the order of the events within the operands. We use the term ‘*parallelly merged with*’ between each operand to denote a parallel merge between the behaviors of the operands.

C. The complexity of the resulting English prose must scale linearly with the complexity of CORAL diagrams in terms of size

As illustrated by Figure 2 and Figure 3, the definition of the translation function in Section III-B ensures that the structure of its output mirrors the input diagram, and that there is a linear relationship between the size of input and output. A formal argument that this would hold for
any diagram $d$ could be given based on induction over the syntactical structure of $d$.

V. RELATED WORK

To the best of our knowledge, no risk-driven testing approach provides a similar schematic generation of natural language semantics as presented in this paper. Most approaches use risk tables/matrices or risk annotated models as a means for documenting, communicating and analyzing risks. However, some approaches provide guidelines for documenting risk-related information in natural-language semantics. Redmill [11] provides a set of guide words with associated definitions, which may be used as a basis for documenting risk-related information. Gleirscher [12] provides a similar approach and makes use of a safety analysis pattern for describing informal test cases. Nazier and Bauer [13] provide a template for documenting safety-risk information, while Kumar et al. [14] provide a template for documenting risk-related information within the domain of aspect oriented programming. Souza et al. [15] use a taxonomy based questionnaire for documenting risk-related information.

VI. CONCLUSION

CORAL is a risk analysis language based on UML interactions, and it is specifically developed to support software testers in a risk-driven testing process. CORAL extends UML interactions with constructs for representing risk-related information in sequence diagrams.

In this paper, we presented a structured approach to generate the semantics of CORAL diagrams in terms of English prose. The CORAL semantics is developed to help testers to clearly and consistently document, communicate and analyze risks in a risk-driven testing process. In particular, it helps testers to: (1) obtain a correct understanding of CORAL diagrams, (2) analyze risks posed on the system under test in a clear and consistent manner, and (3) clearly communicate risks posed on the system under test.

We argue that the resulting English prose is comprehensible by testers because: (1) it preserves the structure of CORAL diagrams, (2) it keeps the user-defined text in CORAL diagrams unchanged, and (3) it uses concepts that are known to software testers. In addition, the resulting English prose of the constructs inherited from UML interactions is consistent with their semantics in the UML standard [5]. Moreover, the complexity of the resulting English prose scales linearly with the complexity of the CORAL diagrams in terms of size.

ACKNOWLEDGMENT

This work has been conducted as a part of the DIAMONDS project (201579/S10) funded by the Research Council of Norway, the NESSoS network of excellence (256980) and the RASEN project (316853) funded by the European Commission within the 7th Framework Programme, as well as the CONCERTO project funded by the ARTEMIS Joint Undertaking (333053) and the Research Council of Norway (232059).

REFERENCES