Causal Mechanism Graph – A new notation for capturing cause-effect knowledge in software dependability

Fuqun Huang *, Carol Smidts

Department of Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH 43210, USA

1. Introduction

Eliciting and representing cause-effect knowledge in the software dependability domain is significant for the assessment and control of dependability risks. With the increasing dependence on software and the accompanied increased software complexity, software dependability has become the major determinant of the systems’ trustworthiness for many safety-critical systems. Software dependability denotes the contribution of the software to the computer-based systems’ properties that allow us to trust the system [1]. These properties include such as reliability, safety, security, availability and maintainability [1–3] and different attributes can be emphasized in different contexts or by different stakeholders. Understanding the causal mechanisms [4], the causal factors and the interactions between the factors, underlying software dependability is therefore essential for software practitioners [5,6]. For instance, causal mechanisms can help design better assessment models and risk control approaches because causal mechanisms capture the causal entities that determine software dependability.

Domain experts are generally considered to possess the most abundant cause-effect knowledge and expert opinion elicitation is a direct way to extract such latent domain knowledge. In software dependability engineering, expert judgment is widely used to obtain information for software dependability assessment [7–11]. There are a variety of graphic methods used in the software dependability domain, such as Bayesian Networks [7,10,12,13], Fault Tree [14] and UML [15]. However, most of these methods (e.g. Fault Trees and UML) focus on representing the accurate relations between components or states of a system; they are not suitable for expressing fuzzy conceptual relations existing in one’s cognitive mental model under uncertainty. The few methods used to elicit experts mental knowledge such as Bayesian Networks and Cognitive Maps [16] do not provide a sufficient set of notations to represent comprehensive interaction mechanisms between concepts that may exist in the software dependability domain, e.g. Bayesian Networks only allow expression of conditional dependencies and Cognitive maps only allow expressing “influence” relations. A robust notation is required to elicit experts’ full range of cause-effect knowledge, accurately or fuzzily as one sees fit depending on the depth of knowledge he/she has. Hence, designing a new notation for capturing domain experts’ cause-effect knowledge is a worthy topic of exploration.

This paper provides a new notation, called the Causal Mechanism Graph (CMG), to elicit experts’ causal mechanism knowledge in software dependability. Section 2 reviews relevant existing methods. Section 3 describes the design process of the CMG.
notation. Section 4 presents the CMG notation. Section 5 presents the evaluation of the CMG notation. In Section 6 we discuss the implications of CMG to the research and industrial activities in the software dependability domain and the future work. Section 6 provides the conclusion.

2. Related work

Bayesian Networks are currently widely used to represent expert judgment on risk assessment [7,10,12,13], e.g., dependability assessment [7] and defect prediction [17]. A Bayesian Network (BN) describes the directed dependencies (directed edges) between random variables (nodes). In a Bayesian Network, strength is represented by a conditional probability of occurrence of the dependence as a function of the state of the nodes. Bayesian Networks help people make inferences on the occurrence probability of one event based on its dependence on another event. Fuzzy Bayesian inference was proposed to generalize Bayesian Statistics to fuzzy data [18]. BN is a powerful method for probabilistic inference in domains where expert knowledge is uncertain, ambiguous, or incomplete. However, probabilistic dependencies do not imply causality. Although Causal Bayesian Networks are introduced to represent causal relations by restricting additional semantics [19], the basic notations only contain nodes and edges. BN does not contain the notations required to explicitly capture the real interaction mechanisms between various entities existing in the software dependability domain.

Cognitive maps or causal maps [16,20–25] are another method used to elicit cause-effect knowledge embedded in human minds. A causal map consists of three elements: nodes (used to represent concepts), edges (the direction of edges denoted by arrows indicates beliefs) and ‘influence relationships’ (positive or negative) with strengths. Causal maps can be created either directly or indirectly. Previously existing methods generally concentrate on direct elicitation in which causal maps are produced by participants directly, e.g., through ideographic, nomothetic, and hybrid methods [26]. Direct elicitation exempts researchers from interpretation and coding, which inevitably introduces researchers’ bias. However, indirect elicitation, which extracts causal maps from documentary sources, is more meaningful in terms of allowing individuals to express thoughts using natural language [26,27] and therefore use their full range of expression. Indirect elicitation is also non-intrusive to the experts’ thinking process. These advantages allow the investigator to gain access to busy individuals who might otherwise be unwilling to participate in more intrusive, interactive procedures. In spite of its numerous applications, causal maps contain limited notations to represent relations between variables. In fact, the only relation contained in a causal map is ‘influence’. Causal maps do not contain logical combinations between various variables, time sequences, and other interaction mechanisms. Thus unfortunately, causal maps are not capable of accurately representing the complex causal mechanisms that exist in engineering domains.

Methods exist in software dependability to accurately model the relations or behaviors of technical systems. Fault Tree [14] analysis is a deductive analysis method for root cause analysis. A fault tree is developed in a top-down way to identify the ‘basic events’ that cause a system failure. Logic notations such as AND, OR, and different types of event notations are provided to construct a fault tree. In a fault tree analysis, a top event is considered as the effect of a set of basic events combined by Boolean logic. Event Tree analysis is performed in an inductive way to analyze the consequences of the primary event of concern using logic notations. Failure mode and effects analysis (FMEA) and its extension method Failure mode effect and criticality analysis (FMECA) [28] are also inductive analyses, in which the causes and effects are enumerated for each failure mode of concern. Fault tree, Event tree and FMEA have no explicit representation on the time element, which is an essential element of causality, whereas Petri nets [29] are used to describe system states and states’ transitions over time. Event Sequence Diagram (ESD) [30] was used to describe dynamic behaviors for probabilistic risk assessment. ESD includes notations to describe time delays and conditions in addition to different types of logic and event notations. These methods are sometimes combined to overcome the shortage of a method in different application contexts [31–33]. However, these methods collectively focus on the relations between components or states in a system, thus corresponding analysis requires the analyst to know a system or component thoroughly. These methods do not include fuzzy conceptual relations (e.g., correlations and contributory causal relations) existing in one’s cognitive mental model under uncertainty. For instance, an expert knows that “program complexity” can be a contributory factor to “faults”, but that does not mean that “complexity” will produce “faults”; how “complexity” interacting with other factors (e.g., the task contexts and programmer’s schema) [34] leads to “faults” is unknown to the expert. These system-oriented methods do not seem suitable for the elicitation of one’s cognitive knowledge which can be fuzzy, intuitive and/or abstract by nature.

In summary, each of the above methods has its advantages and disadvantages in different application contexts. Considering the various gaps in knowledge of previously used techniques, a set of notations that allows practitioners to straightforwardly represent experts’ knowledge of causal mechanisms in software dependability engineering is required.

3. The design process of the CMG notation

The design process used to develop the CMG notation was based on the Grounded Theory [35]. Grounded Theory was chosen to guide the CMG design process for its three essential principles [36,37] that are particularly suitable for building new theories. These principles are: using an inductive and iterative process to generate a new theory from empirical data. Deductive research seeks to verify hypotheses that are derived from an existing or pre-conceived theory. By contrast, an inductive process seeks to identify categories and relations from empirical data, and thus build a theory to account for a phenomenon. Developing a new theory is also an interactive process, in which the researchers continuously assimilate and integrate new information into the existing mental model. Thus, the theory evolves during the research process itself. The inductive and iterative features of grounded theory conform well to the cognitive mechanisms underling humans’ design activities [38,39]. Therefore, Grounded Theory is widely used to explore new knowledge or construct new theories in various domains, including software engineering [40,41].

The CMG notation is designed using an inductive and interactive process based on data provided by domain experts. We first designed semi-structured questionnaires to obtain the experts’ opinion on software dependability causal mechanisms. Then, the data were iteratively analyzed to discover the main concept categories and relation patterns contained in the answers to the questionnaires. While analyzing the data, the researchers simultaneously designed the new notations in order to represent the patterns in the data. This process proceeded iteratively until Theoretical Saturation was reached—no new notations were required to represent the information of interest in the data. Finally, using the designed notations, we converted the questionnaire responses to causal mechanism graphs and verified them with the corresponding experts.
دریافت فوری متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
بزرگ‌شانس سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات