Progress in whole-lifecycle software architecture modeling

June 25 2014

The gradually increasing complexity of user requirements and runtime environments of software demands software to be of more capabilities and thus become more complex than ever. In the past several decades, there was a trend that the scale of software has been increasing continuously. Nowadays, there are tens or even hundreds of million lines of code in a large scale software system. For example, the Windows operating system scales from 15 million lines of code in 1995 to 60 million lines of code in 2007; in 2011, the scale of software in BMW 7 Series reaches 200 million lines of code; the scale of software in Airbus 380 even reaches 1 billion lines of code. When the scale and complexity of software is beyond what developers and techniques can control, the quality and efficiency of software would fail to satisfy user requirements, leading to the phenomena of software crisis. Software engineering is the main effort to deal with software crisis, whose central goal is to capture and control software complexity.

A research team in Peking University extends software architecture that originally aims to control the high-level structural complexity in the design phase of software, to the whole-lifecycle of software, and proposes an architecture-centric software development method, named ABC. The ABC method unifies the core artifacts in software development into various software architecture models, and the core activities in software development into continuous and iterative refinement, mapping, and transformation of the software architecture models, respectively. As a result, ABC realizes consistent, flexible, and systematic modeling and control of high-level software structural complexity. A comprehensive article reporting this research (with the title ABC: a method of software architecture modeling in the whole lifecycle) is published in 2014 (5) issue of *Science China: Information Science*. The main investigators of this research include Hong Mei, Gang Huang, Lu Zhang, and Wei Zhang.

The research on the ABC method starts from 1999 for large scale enterprise software and from 2006 for Internet-based software. The central idea of ABC is to introduce software architecture into all the phases of software development, utilize various automated transformation mechanisms provided by software tools to bridge the gap between the high-level design and low-level implementation, and generate the code framework and glue code based on the support of execution platforms. A set of tools and platforms (including a feature modeling tool, a software architecture modeling tool, a meta-modeling tool, a source code generation tool, a service composition tool, a component management platform, and a componentized software execution support platform) are implemented to support the ABC method. The ABC method and its supporting tool have been used in the modeling of the information system for Beijing 2008 Olympic Games, the design and modeling for a military information system, and the development of a credit risk management system for a commercial bank.

As a central artifact in the design phase, software architecture has long been a hot research area in software engineering. Compared with existing research on software architecture, the major distinctive features of the ABC method are fourfold. First, ABC defines the basic building blocks of software architecture as "components", instead of "objects" which are of finer granularity and used for complexity control at the source code level, and treats connectors between components as firs-order entities as well. Second, ABC abstracts the functions or services provided by the execution platforms into constraints and connectors, so as to naturally utilize them to realize the requirements on performance, reliability, and security. Third, ABC extends software architecture to the requirements analysis phase and also to the maintenance and evolution phase, so as to control software complexity in the whole lifecycle, based on the intrinsic connections between software complexities in different phases. Finally, several important properties of software architecture in different phases have been found and investigated, including the relationships between different requirement dependencies, the forward traceability between the feature model and software architecture, the mechanism and properties of runtime software architecture, and the backward traceability between system implementation and system design.

With the rise of the next generation of information technology (such as cloud computing, mobile Internet, and Internet of things), there has been a significant change in the paradigm of software; as a result, the complexity of software shows some new trends. Under these new application modes and runtime environments, ABC still possesses its essential value both in theory and practice, but also faces new challenges and opportunities, such as the fusion of requirements engineering and knowledge engineering, the interaction between software reuse and artificial intelligence, the cloud-based software engineering, and the Internet operating systems. These issues will be the possible future work of ABC.

More information: Mei H, Huang G, Zhang L, Zhang W, ABC: a method of software architecture modeling in the whole lifecycle. SCI CHINA Info. Sci, 2014 Vol. 44 (5): 564-587. info.scichina.com:8084/sciF/CN … abstract514469.shtml

Provided by Science China Press