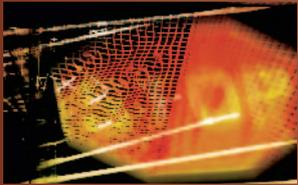


Toward Trustworthy Software Systems

Wilhelm Hasselbring, University of Oldenburg
Ralf Reussner, University of Karlsruhe



TrustSoft's holistic approach to software trustworthiness looks beyond security.

Software controls an increasing number of complex technical systems, ranging from Internet-based e-health and e-government applications to embedded control systems in factories, cars, and aircraft. Because software failures can now cause extensive, even disastrous, damage, the successful deployment of such systems depends on the extent to which we can justifiably trust them.

Academia, government, and industry are aware of the problem. In May 2005, the US Center for National Software Studies issued a report, *Software 2015: A National Software Strategy to Ensure U.S. Security and Competitiveness*, that identified software trustworthiness as the most important focus of future research (www.cnsoftware.org/nsg).

Organizations such as Microsoft's Trusted Computing Group (www.trustedcomputinggroup.org) and Sun Microsystems' Liberty Alliance (www.projectliberty.org) are currently leading the debate on "trustworthy computing." However, these and other initiatives primarily focus on security, and trustworthiness depends on many other attributes.

To address this problem, the University of Oldenburg's TrustSoft Graduate School (<http://trustsoft.uni-oldenburg.de>) aims to provide a

holistic view of trustworthiness in software—one that considers system construction, evaluation/analysis, and certification—in an interdisciplinary setting.

QUALITY ATTRIBUTES

Software trustworthiness consists of several attributes, as the following recent examples of major software failures indicate.

Correctness refers to the absence of improper system states. In 2004, the A2LL software used by Germany's social services prevented bank transfer payments of unemployment benefits to several hundred thousand recipients by incorrectly aligning their account numbers.

Safety indicates the absence of catastrophic environmental consequences. In 2000, a programming glitch in the power-train module of Ford's Explorer and Mountaineer SUVs caused the cruise control to accelerate suddenly without warning.

Quality of service includes three quantifiable attributes:

- *Availability*—probability of readiness for correct service. In 2003, the W32.Blaster worm exploited a Windows DCOM RPC interface buffer overrun vulnerability to exe-

cute a distributed denial-of-service (DDoS) attack against the Microsoft Windows Update site.

- *Reliability*—probability of correct service for a given duration of time. In 2004, software in the diesel control unit of the Mercedes-Benz Vito and Viano delivery vans occasionally deactivated the fuel supply.
- *Performance*—response time and throughput. Frequent user accesses of the central server component of Lower Saxony's Nivadis criminal tracking system, deployed in 2003, for very small interactions caused heavy overload of the system while processing business transactions, resulting in extremely long response times.

Security refers to the absence of unauthorized access to a system. In 2004, the W32.Sasser worm penetrated Windows' Local Security Authority Subsystem Service to cause abnormal system shutdowns in infected machines.

Privacy indicates the absence of unauthorized disclosure of information. In 2002, hackers broke into the Transurban CityLink electronic toll system in Melbourne, Australia, and posted data from 500,000 credit cards on the Web.

ATTRIBUTE RELATIONSHIPS

These quality attributes can have two basic types of relationships.

Intrinsic relationships exist if one attribute affects another. For example, DDoS attacks are usually considered a security threat, but essentially they attack system availability. Indeed, security can be regarded as a composite of confidentiality, integrity, and availability.

Likewise, reliability metrics often apply to execution time. Consequently, models that predict reliability depend on a system's anticipated performance. In such cases, the relation between reliability and performance is intrinsic.

Extrinsic relationships occur when attributes behave in an opposing way—for example, an increase in

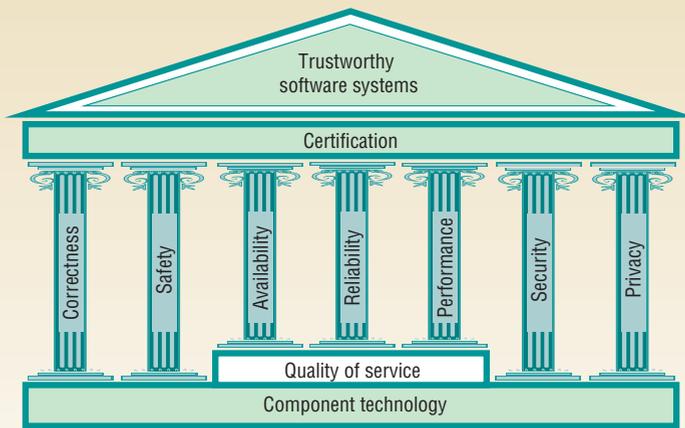


Figure 1. TrustSoft "research building." The new graduate school focuses on the construction, evaluation/analysis, and certification of trustworthy software systems.

security or reliability often decreases performance. In this case, the relation between reliability and performance is extrinsic.

However, such antagonistic relations between two attributes do not exist per se; rather, they are a property of specific software architectures. For example, performance and reliability are not always antagonistic, as the replication of resources can increase both.

A HOLISTIC APPROACH

Isolated investigations of individual attributes can't possibly capture the complexity in practical system design problems. What's required is a holistic approach to software trustworthiness that utilizes multidimensional optimization techniques.

Improved certification is one major goal of such research. Users will be increasingly less willing to accept software, particularly in newer pervasive systems that are hidden from view, without some credible validation of its trustworthiness.

The current practice of not certifying software in a way established for more traditional technical systems, and of largely relieving vendors from liability, isn't tenable in the long run. Those who can certify quality attributes of their products and contractually guarantee liability will achieve a significant competitive advantage in the marketplace.

TRUSTSOFT

To tackle these challenges, the German Research Foundation funded the TrustSoft Graduate School at the University of Oldenburg for nine years beginning in April 2005. The school includes 14 PhD supervisors and approximately 20 PhD students organized in three cohorts over the funding period.

TrustSoft aims to advance knowledge of the construction, evaluation/analysis, and certification of trustworthy software systems in an interdisciplinary setting. We are developing new methods for the rigorous design of trustworthy software systems with predictable, provable, and ultimately legally certifiable system properties.

As Figure 1 shows, component technology is the foundation of our research program. The choice of a component architecture greatly influences the resulting software systems' nonfunctional properties. Component-based design is highly successful in mechanical, electrical, and other engineering disciplines, and we aim to apply the same paradigm to software engineering, as do many other research groups.

TrustSoft researchers are investigating all quality attributes related to trustworthiness—represented as pillars in Figure 1—for both individual components and composite software systems. Scientific methods vary with the attributes under investigation. For

example, we use mathematical proofs to demonstrate correctness on certain abstraction levels.

We are well aware that it is impossible to build completely error-free complex software systems. We therefore complement fault-prevention and -removal techniques with fault-tolerance methods that introduce redundancy and diversity into software systems.

Quantifiable attributes such as availability, reliability, and performance call for analytical prediction models, which require empirical studies for calibration and validation.

To consider the legal aspects of software certification and liability, TrustSoft integrates the disciplines of computer science and computer law. Software engineers will learn which models, methods, and tools are required to certify properties, while law students will learn how to better incorporate software engineering technical standards into legislation.

Given its small complement of researchers, we recognize that TrustSoft alone will have a limited impact on trustworthy software development. However, the expressed support of Microsoft and Sun Microsystems for the new graduate school indicates industry recognition of the need for a holistic approach to this problem. We encourage other institutions to follow our example. ■

Wilhelm Hasselbring is a professor of software engineering and chair of the TrustSoft Graduate School at the University of Oldenburg, Germany. Contact him at hasselbring@informatik.uni-oldenburg.de.

Ralf Reussner, previously vice chair of TrustSoft, is a professor of software engineering at the University of Karlsruhe, Germany. Contact him at reussner@ipd.uni-karlsruhe.de.

Editor: Jack Cole, US Army Research Laboratory's Information Assurance Center, jack.cole@ieee.org; <http://msstc.org/cole>