

Theory in Dependable Ubiquitous Computing

Achievements and Challenges

Marta Kwiatkowska
School of Computer Science, University of Birmingham

Ubiquitous computing entails highly distributed, large-scale networks of small, embedded, portable or wearable devices, communicating peer-to-peer over wireless medium, and operating in a timely fashion and at low power. This places heavy demands on engineering of hardware and software solutions, particularly in embedded devices where the cost of recall in case of faults can be prohibitive. Well-publicised examples of embedded software failure include the loss of NASA Mars Polar Lander in 1999¹ and BMW safety recall of over 15,000 cars due to programming errors in its iDrive unit powered by Windows CE².

A characteristic of ubiquitous computing systems is a substantial increase in complexity, both in the number of components, as well as their interactions and individual behaviour. The combination of distributed environment and dynamic topology calls for innovative solutions to management and resource control, featuring randomisation, game theory, statistical techniques and nature inspired algorithms. Mobility in 3D space requires geometric models and techniques capable of predicting the trajectories and their effect on response time. High failure rate of transmission in the wireless medium and the resulting delays necessitate probabilistic modelling, while the requirement to operate at low power demands dynamic power management policies. The complexity of the scenario makes reliability guarantees difficult to achieve, and additional assurances of *Quality of Service* – performance, availability, resource usage, timeliness of response – will be demanded by users.

The central aim of the Grand Challenge is to develop theories – *calculi*, *models* and *tools* – to understand and support the development of ubiquitous computing systems, guided by the emerging concepts and design principles arising as part of the sister, Ubiquitous Computing Systems, challenge. These concepts include *discrete* behavioural aspects, such as nondeterminism, concurrency, mobility, trust and security, as well the modelling of *real-valued* aspects, for example sensor inputs such as time or geographical positioning, and *probabilistic* behaviours. The theories must also encompass *data*, its *provenance* and *availability* in applications such as distributed databases in bioinformatics and medical healthcare.

While process calculi for mobility are generally well understood and have influenced the design of real programming languages, the modelling formalisms for real-time and probability are much less well developed. Calculi for *hybrid systems*, which combine discrete behaviour (such as gate opening or closing) with continuous dynamics (e.g. movement in space, temperature or water level) and are therefore particularly relevant for ubiquity because of their ability to

¹<http://mars.jpl.nasa.gov/msp98/>

²http://www.baselinemag.com/print_article/0,3668,a=35839,00.asp

represent motion in space and time, are in their infancy. *Probabilistic* calculi such as PEPA are more advanced, but issues such as compositionality and abstraction still cause difficulties.

This presentation will focus on *dependability* and *Quality of Service assurance*, conventionally achieved via testing and simulation, but here we particularly advocate an alternative, *model-based* verification approach called *model checking*. Model checking is an *automatic* method which can establish – with the help of a software tool – the correctness of the *model* of a given software or hardware system with respect to specification, and otherwise produce error diagnostics. When applied as part of a design process, it can detect errors in the designs before manufacture, thus improving reliability and reducing production costs. Following major successes in detecting genuine errors in standardised protocols, such as Futurebus (SMV tool), Needham-Schroeder authentication (FDR tool) and AODV (SPIN tool), model checking has become a leading research focus as well as standard industrial practice (e.g. Intel, IBM). Founded on theoretical results, it is arguably the greatest success of “*theory to practice*” *technology transfer* in computer science research.

Although less well developed, model checkers exist for timed and hybrid automata (tools UPPAAL, HyTech), and for probabilistic models (tools PRISM, ETMCC), but are limited and hampered by undecidability results or high complexity, preventing application to realistically sized problems. Despite this, there have been successes: an error has been found in a Philips audio protocol (UPPAAL), unusual behaviour found in the Crowds anonymity protocol for the Internet (PRISM) and performance and timing aspects analysed for protocols such as IPv4 ZeroConf dynamic configuration (UPPAAL, PRISM). Nevertheless, significant challenges remain, and these form the basis of this international effort with strong UK presence. We expect advances in calculi for modelling spatial mobility, resources and probabilistic behaviour in a dynamic context. Model checkers will increasingly be able to tackle more and more complex systems, thanks to advances in abstraction, infinite-state systems, hybrid systems and compositionality (via games semantics).