Contribution ID: **67** Type: **not specified**

Runtime Monitoring for the Diagnosis and Recovery of Complex Physical Systems

Runtime verification focuses on techniques to check the dynamic (runtime) behaviour of a system typically with the aim of ensuring that the system is working correctly. Inlining property checks via assertions or similar techniques has been standard practice since the dawn of programmable machines. Such inlined approaches result in the interweaving of the executable system specification (the program) with the specification of properties which it should satisfy. Separation of concerns has long been identified as an important principle, and the possibility of separating these two aspects of a system is one of the objectives of most modern runtime verification approaches —allowing for (i) having different teams working on the different aspects i.e. development and quality assurance; (ii) the use of the specification across different versions, instances or even systems. Another issue with inlined assertion checking arises as the complexity of the properties increases. Although inlining a property such as 'The gas leak variable should be low when the induceSpark method is called'is straightforward, a property such as 'The openValve method should have been called before induceSpark is called'results with the developer having to introduce additional state to remember whether openValve was called in the past. More complex properties, such as 'The gas leak variable may not have been true for more than 1 minute in the last 30 minutes just before induceSpark is called', results in more complex additional state and logic to handle it, which may, in turn result in more new errors being introduced into the system.

The role of the runtime verification tool is twofold: (i) it modifies the system instrumenting code to be able to capture points of interest during its execution which are of interest with respect to the specification; and (ii) it converts the specification into a monitor, which reacts whenever a point of interest is reached, checking that the behaviour of the system does not violate the specification. Using such an approach, a specialised language can be used to write the specifications, which allows the adoption of domain specific languages which can be used to describe the behaviour more succinctly and precisely.

Runtime verification would lend itself well to complex and critical environments such as detector and accelerator control systems, where loss of detector sub-systems could hinder it from data-taking, or errors in control systems of high-energy particle accelerators with highly-destructive beams such as LHC could damage the machine, leading to months of costly downtime.

Summary

{}

Authors: Dr COLOMBO, Christian (University of Malta); DE CATALDO, Giacinto (Universita e INFN, Bari (IT)); VALENTINO, Gianluca (University of Malta (MT)); Prof. PACE, Gordon (University of Malta); VELLA, Kevin (University of Malta)

Presenters: Dr FRANCO, Antonio (INFN, Bari); Dr COLOMBO, Christian (University of Malta); VALENTINO, Gianluca (University of Malta (MT))