Achieving Determinism in Adaptive AUTOSAR

Christian Menard*, Andrés Goens*, Marten Lohstroh† and Jeronimo Castrillon*
* Center for Advancing Electronics Dresden (cfaed), TU Dresden, Dresden, Germany
{christian.menard, andres.goens, jeronimo.castrillon}@tu-dresden.de
† Department of EECS, UC Berkeley, USA
marten@berkeley.edu

***Abstract*—AUTOSAR Adaptive Platform (AP) is an emerging industry standard that tackles the challenges of modern automotive software design, but does not provide adequate mechanisms to enforce deterministic execution. This poses profound challenges to testing and maintenance of the application software, which is particularly problematic for safety-critical applications. In this paper, we analyze the problem of nondeterminism in AP and propose a framework for the design of deterministic automotive software that transparently integrates with the AP communication mechanisms. We illustrate our approach in a case study based on the brake assistant demonstrator application that is provided by the AUTOSAR consortium. We show that the original implementation is nondeterministic and discuss a deterministic solution based on our framework.**

***Index Terms*—automotive engineering, reliability and testing, software and system safety, software engineering**

I. Introduction

Designing and developing software for automotive applications is challenging due to stringent safety and real-time requirements. New use cases like the self-driving car have caused a dramatic increase in complexity and computational demands

```
int main() {
  s = ServiceProxy();

  s.set_value(1);
  s.add(2);
  result = s.get_value();

  std::cout << result.get();
  return 0;
}
```

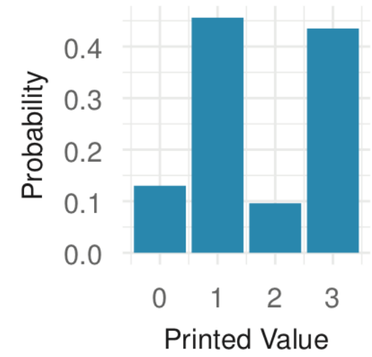

Figure 1. A nondeterministic AUTOSAR Adaptive Platform (AP) client/server application. The client manipulates the server's state variable in a series of (non-blocking) procedure calls. The client prints out one of four different results, distributed as shown in the graph on the right.

into physical damage, injury, or even loss of life. For this reason, the nuclear, aeronautics, and railways industries often rely on synchronous languages like LUSTRE [2], Esterel [3], and SCADE [4] to rule out nondeterminism in their designs of safety-critical software [5].

In the latest iteration of its well established CP standard, the AUTOSAR consortium introduced support for the logical execution time (LET) paradigm [6], [7]. This can be used to