Contribution ID: **15** Type: **not specified**

Formal verification - Robust and efficient code: Introduction to Formal Verification

Monday, 29 February 2016 16:15 (1 hour)

LECTURE 1: We will establish two general approaches to FV and where they are applicable: model checking and theorem proving. We will explore the latter in more details and have a brief look at the underlying theory, predicate logic. We will see how this family of logic systems can be used to prove abstract properties of our program and why this is useful. Practical examples will be presented and explained.

This talk aims to introduce the concepts of Formal Verification and how they can be used to the benefit of the programmer to produce robust and efficient code. We will be looking into the subject at two levels, both and overview of what FV can concretely bring programmers and going into the nitty-gritty details of theorem proving one of the methods use for FV.
In general, FV means "proving that certain properties hold for a given system using formal mathematics". This definition can certainly feel daunting, however, as we will learn, we can reap benefits from the paradigm without digging too deep into the subject.
Examples where FV can help include proving that your code cannot raise division by zero exceptions; produce optimised byte code where the optimisations are proven to be safe and help reason about concurrent systems.

Presenter: ALBERTSSON, Kim (CERN)