Title: AI-Generated Code for Critical Systems: Can We Trust It?

Abstract:
The need for rapid software development continues to accelerate as organizations adopt software-defined products to stay competitive. The automotive industry's software-defined vehicle (SDV) initiative is a clear example of this shift. Yet building these systems becomes significantly more complex when safety or security requirements apply, especially under strict industry standards.

To maintain development speed, teams are increasingly looking to automate and streamline both engineering and compliance activities. Advances in generative AI have made it possible for large language models (LLMs) to produce code directly from requirements. However, regulated sectors such as automotive, industrial automation, and medical technology remain cautious due to the elevated risks associated with defects in safety-critical software.

To address these risks, organizations are exploring how to combine LLM-based code generation with deterministic software-testing technologies—such as static analysis, unit testing frameworks, code-coverage measurement, and simulation environments—to strengthen confidence in generated code and reduce security vulnerabilities.

In this presentation, we will:
• Examine the risks of applying LLMs to safety-critical code generation.
• Demonstrate the role of deterministic testing tools—including static analysis and code-coverage methods—in validating AI-generated code.
• Highlight recent scientific research on LLM-based code generation for critical systems.

=====

Speaker: Marcin Zwawa

Job title: Senior Solution Architect

Bio: Marcin Zwawa is a Senior Solutions Architect and Pre-Sales Engineer with over 18 years of experience in automated software-testing technologies. He has led the technical delivery of Parasoft solutions for major clients across Europe in industries such as automotive, avionics, medical, railway, machinery, telecommunications, insurance, and banking.

His expertise includes test and process automation (CI/CD), embedded testing, integration with build systems, safety-critical standards, unit and functional testing, continuous testing, application lifecycle management, and service virtualization. Marcin also specializes in static and dynamic code analysis for C, C++, C#, and Java.