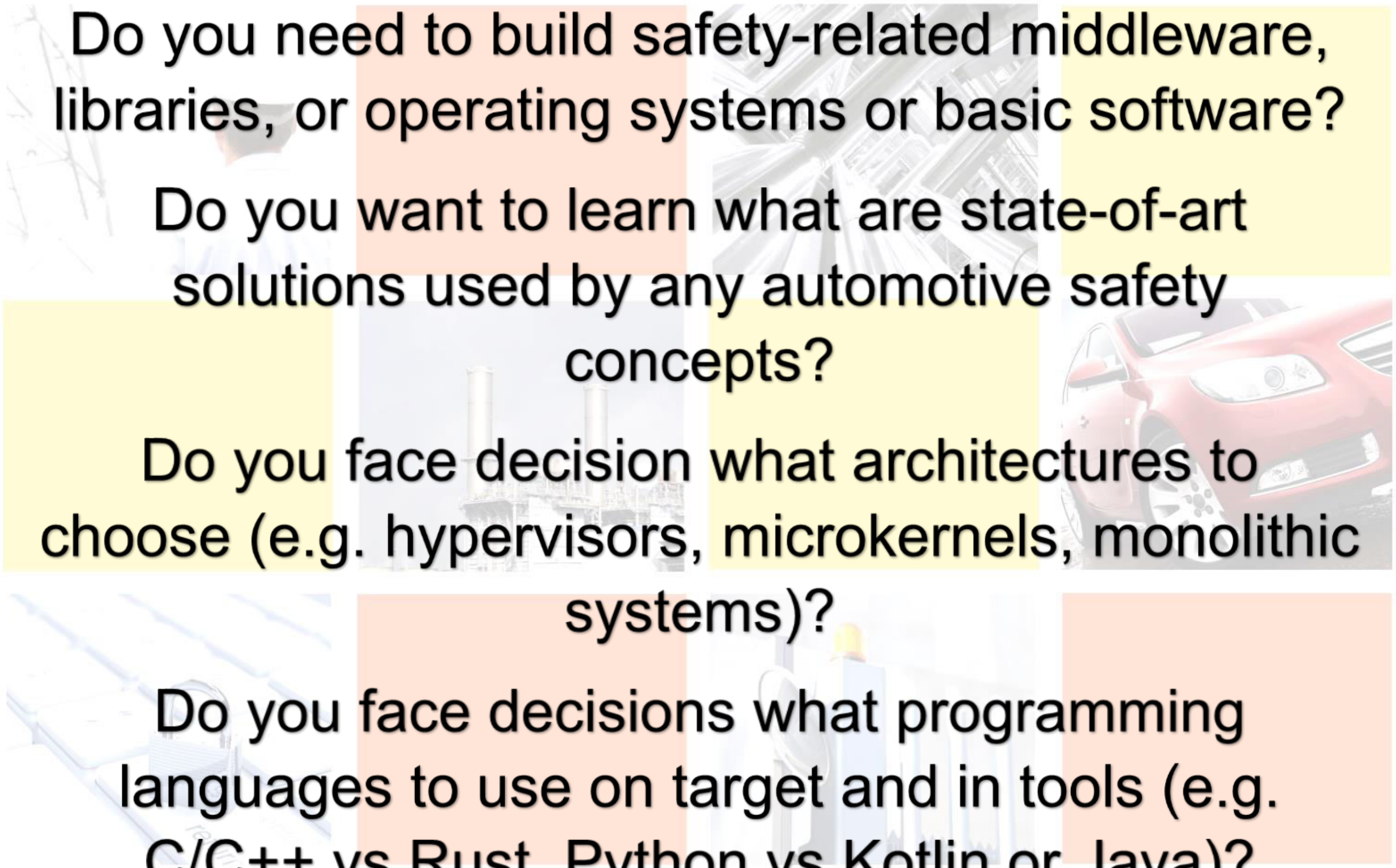
Do you want to apply a lightweight, efficient and scalable safety concepts, architectures and solutions?

Do you need to build safety-related applications?



Do you need to build safety-related middleware, libraries, or operating systems or basic software?

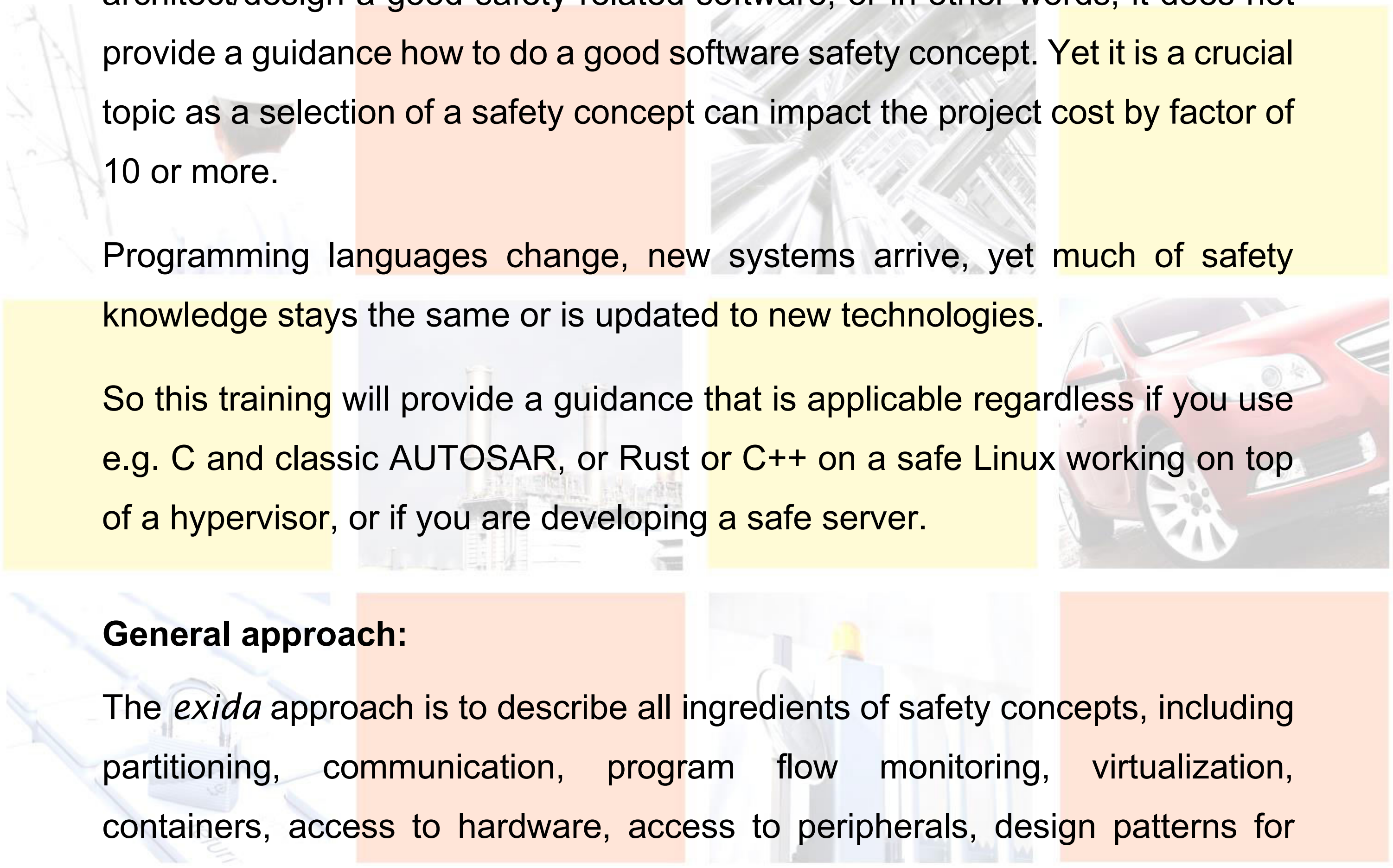
Do you want to learn what are state-of-art solutions used by any automotive safety concepts?

Do you face decision what architectures to choose (e.g. hypervisors, microkernels, monolithic systems)?

Do you face decisions what programming languages to use on target and in tools (e.g. C/C++ vs Rust, Python vs Kotlin or Java)?

Do you have existing products and need to extend it by performing re-engineering, refactoring and introduction of safety mechanisms?

DE0807 Designing good software concepts and software architecture



Standards like ISO 26262 provide not so much of technical solutions how to architect/design a good safety-related software, or in other words, it does not provide a guidance how to do a good software safety concept. Yet it is a crucial topic as a selection of a safety concept can impact the project cost by factor of 10 or more.

Programming languages change, new systems arrive, yet much of safety knowledge stays the same or is updated to new technologies.

So this training will provide a guidance that is applicable regardless if you use e.g. C and classic AUTOSAR, or Rust or C++ on a safe Linux working on top of a hypervisor, or if you are developing a safe server.

General approach:

The *exida* approach is to describe all ingredients of safety concepts, including partitioning, communication, program flow monitoring, virtualization, containers, access to hardware, access to peripherals, design patterns for monitoring and redundancy, concepts for safety-related availability, concepts for tools like generators or ML frameworks.

DE0807 Designing good software concepts and software architecture

Who should attend?

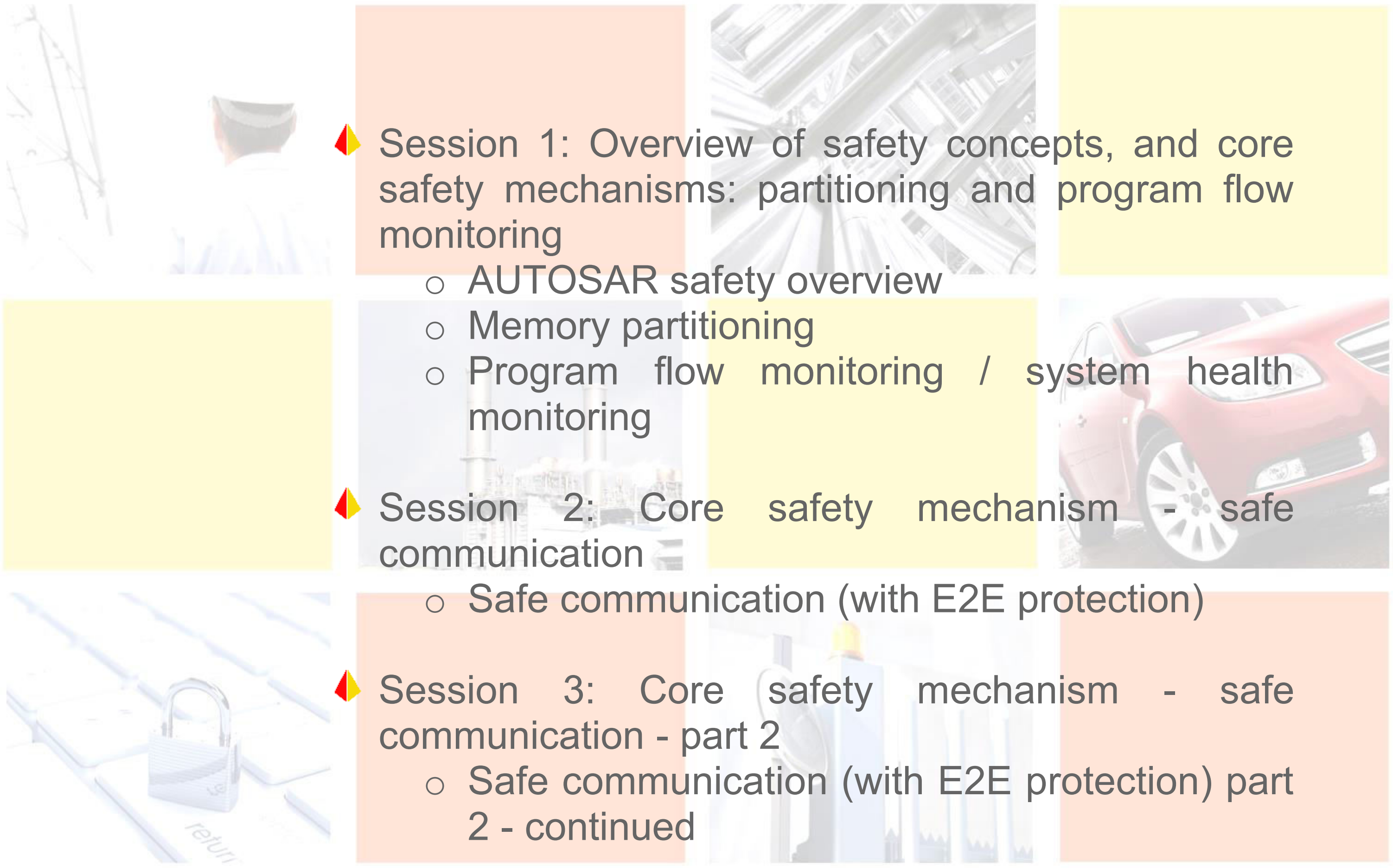
- Software Safety Engineers
- Software Safety Architects
- Product Owners
- Software Developers
- System Architects

Duration:	16 hours split as eight 2-hour sessions or four 4-hour sessions
Language:	English
Location:	online
Certificate:	Each participant gets a letter of attendance.

For more information, please contact:
Kerstin Tietel ✆ +49 89 44118232
✉ kerstin.tietel@exida.com

DE0807 Designing good software concepts and software architecture

Agenda and Content



- Session 1: Overview of safety concepts, and core safety mechanisms: partitioning and program flow monitoring
    - AUTOSAR safety overview
    - Memory partitioning
    - Program flow monitoring / system health monitoring

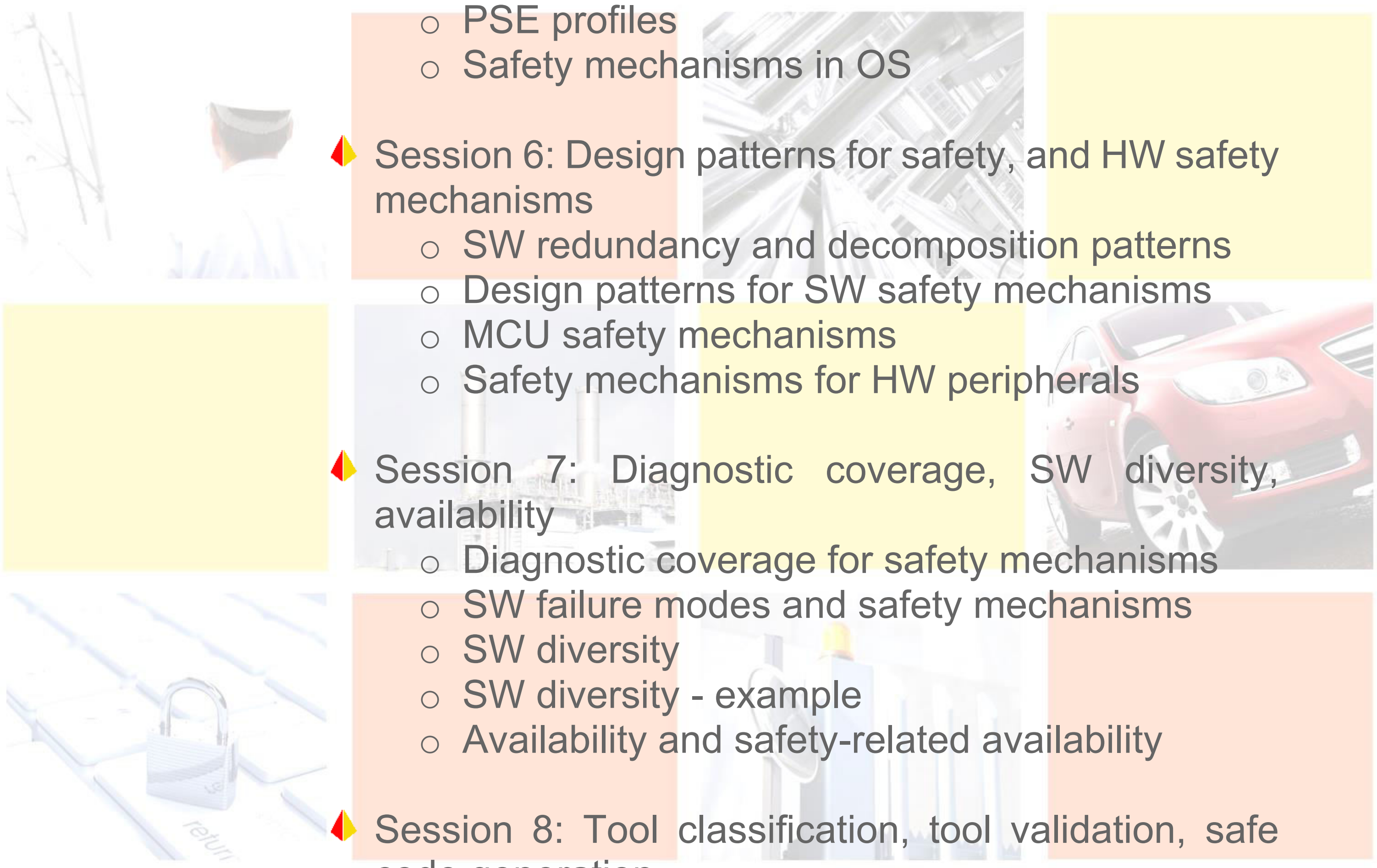
- Session 2: Core safety mechanism - safe communication
    - Safe communication (with E2E protection)

- Session 3: Core safety mechanism - safe communication - part 2
    - Safe communication (with E2E protection) part 2 - continued

- Session 4: Safety mechanisms provided by OS, decomposition and coexistence
    - Memory protection by OS
    - Timing protection by OS
    - Communication by OS
    - Mixed ASIL – decomposition and independence
    - Mixed ASIL – coexistence and freedom-from-interference

www.exida.eu

DE0807 Designing good software concepts and software architecture



- Session 5: Partitioning strategies
  - Partitioning strategies
  - PSE profiles
  - Safety mechanisms in OS

- Session 6: Design patterns for safety, and HW safety mechanisms
  - SW redundancy and decomposition patterns
  - Design patterns for SW safety mechanisms
  - MCU safety mechanisms
  - Safety mechanisms for HW peripherals

- Session 7: Diagnostic coverage, SW diversity, availability
  - Diagnostic coverage for safety mechanisms
  - SW failure modes and safety mechanisms
  - SW diversity
  - SW diversity - example
  - Availability and safety-related availability

- Session 8: Tool classification, tool validation, safe code generation
  - Tool classification
  - Tool validation
  - Safe code generation – best design patterns
  - Safe tool development