703088 (2025WS) - PS Distributed Systems
Distributed and Parallel Systems group, University of Innsbruck
Instructors: Dr. Juan Aznar Poveda, Dr. Zahra Najafabadi Samani

Final Project - Distributed Systems

During the execution of this project, you will apply the knowledge acquired during the previous weekly exercises as well as learn to design, develop, and deploy scalable distributed applications using modern programming languages, frameworks, and tools. The outcome of this project should be:

- The implementation of the project according to the specifications presented in this description.
- Submission of intermediate weekly milestones to track progress.
- A final presentation of the project, including its implementation and evaluation.
- A final technical report that explains your solution.

Work in groups: The project should be done in groups of 2-4 students.

> **IMPORTANT:** All the members should actively contribute to the project. Inform the instructor by agreement as soon as any member of the group is not successfully carrying out their tasks without any justification.

Event-driven Geo-Distributed Application

In this project, you will design, develop, and deploy a distributed Cloud-Edge-IoT system that solves a realistic distributed-systems problem. Your system **must** involve IoT, Edge and Cloud, coordination, and handling of real data or workload.

Computing continuum layers

IoT Layer

- Generates or captures raw input data (e.g., images, sensor readings, video frames, events, gameplay actions).
- Produces data at a configurable interval (the lower the interval, the higher the system load).
- Acts as the entry point to your distributed workflow.

Edge Layer

- Performs lightweight pre-processing to reduce unnecessary load on the Cloud layer.
- Examples:
  - Filtering images for relevant content (e.g., people, vehicles, motion)
  - Detecting significant changes in sensor readings
  - Aggregating or sampling a high-frequency data stream.
- Forwards only relevant or processed data to the Cloud layer.

Cloud Layer

- Performs more computationally intensive analysis on the forwarded relevant data
- Examples:
  - Deep analysis (e.g., anomaly detection, recognition, aggregation, classification, pattern detection)
- Triggers alarms in the IoT layer based on the analysis results.

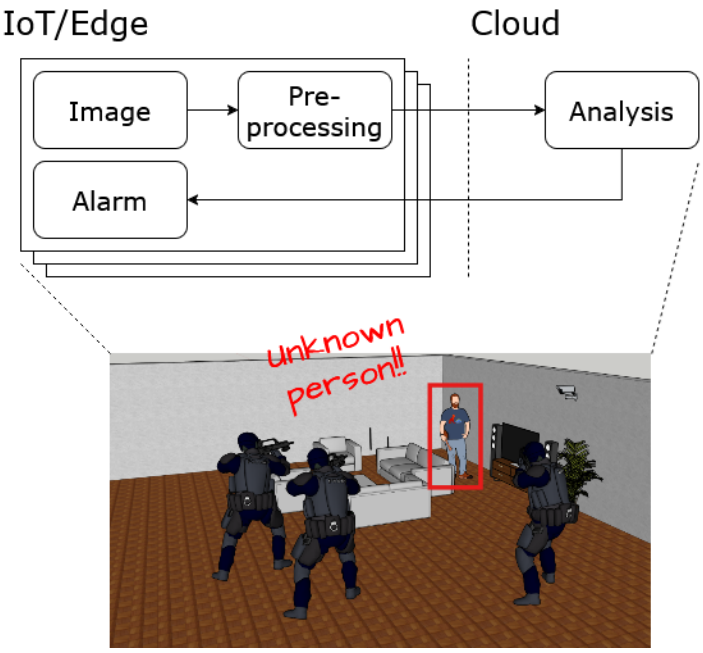
Example project topics include (but are not limited to):

- Geo-distributed Surveillance System (the system should detect unknown people and keep track of known people)
- Geo-distributed Traffic Sign Analysis System (the system should keep a record of traffic signs on the road)
- Geo-distributed Smart Building Management System (the system should record occupancy and people in th building and improve energy efficiency)
- ...

Detailed description of the surveillance system example

In this subsection, we elaborate the description of the geo-distributed surveillance system example.

- Implement a service using the WiseNET videos and can be invoked (e.g., via HTTP) to get the current image, include multiple videos to emulate different cameras, and time is sufficiently simulated (i.e., suppose 1 minute passes between subsequent invocations, then the video frames returned by the service should be 1 minute in the video apart). For the purpose of reading the video file, you use an existing library/framework such as OpenCV;
- Lightweight person detection occurs using a YOLO object detector trained to detect persons at the Edge.
- If persons are detected, Cloud analysis is initiated.
- Akka is used to coordinate the detection of known and unknown people based on events;
- In case unknown person is detected, alarm devices near the involved cameras are triggered;
- Alarms are later disarmed.



Simple example diagram of a surveillance system.

Requirements

The implementation should be as realistic as possible in terms of computing resources, distribution, heterogeneous networks, communications, data, and feasibility of the frameworks and tools employed.

1. Quality of datasets

Employ high quality, realistic, and well justified datasets for your application (e.g., WiseNET for video, OpenSky for aircraft data, traffic datasets, sensor datasets, large text collections, or your own synthetic generator).

2. Proper distribution

The application should be highly and appropriately distributed using **at least**:

- 5 IoT devices (can be simulated),
- 2 Edge devices and
- 2 Cloud resources.

Implementation requirements

- **Mandatory technology requirement.** At least one component of the cloud layer must be implemented using Akka.
- **Optional extension.** As an additional task, teams may employ Akka Cluster to enable distributed coordination and state sharing across multiple cloud-deployed nodes.

> **IMPORTANT:** The design and implementation of the remaining components of your system are entirely left to the teams. The selection of tools and frameworks will be part of the project evaluation; therefore,

> teams should carefully consider which technologies best align with the needs of their application. It may be beneficial to reflect on the technologies explored during the exercises solved in the first half of the semester.

Code quality

Code quality will be evaluated as part of the assessment, with particular attention to clarity, structure, and maintainability.

Milestones and deadlines

The project will be tracked through weekly milestones in which you should submit your current progress of the project (report and code). You should use the final report template and incrementally improve and complete it based on the following weekly milestones:

- MS1: Architectural diagram
  - Deadline: 27.11.2025, 12:00h
- MS2: IoT implementation
  - Deadline: 04.12.2025, 12:00h
- MS3: Edge implementation
  - Deadline: 11.12.2025, 12:00h
- MS4: Cloud implementation
  - Deadline: 08.01.2026, 12:00h
- MS5: System deployment and evaluation
  - Deadline: 15.01.2026, 12:00h

Final delivery

- **Technical report**. the technical report (5 pages) should be self-contained and justify your implementation rigorously, as well as provide the obtained evaluation results. **Use the template provided in OLAT**. Outline for your report:
  - Short introduction to the problem
  - System architecture (detailed diagram and explanation)
  - Implementation details (e.g., frameworks, resources, etc.)
  - Evaluation of the response time and scalability (e.g., number of devices and events processed) to prove the correctness of your implementation.
- **Presentation**. The presentation (10 + 5 minutes QA) should clearly summarize your contributions and use the following structure:
  - Cover slide (Title and the name of the members of the group)
  - System architecture
  - Slide summarizing the datasets, tools, frameworks and resources used (justify why)
  - Live demo
  - Evaluation results

Final submission

- **Final submission: including code, technical report, and presentation**
  - **Deadline: 22.01.2026, 12:00h**

  The final project will be presented in the last two lab sessions.

Grading

The project will be graded in three parts:

- Advance in weekly milestones (30%)
- Mandatory presentation (30%), graded based on:
  - Content
  - Live demo
  - Explanation of your solution
  - Clarity and correctness
- Technical report (40%)

Recall that the project represents the 50% of the total grade of the PS course. At least 30% of each individual part (i.e., exercises and project) is required to average the grade of both parts and be able to pass. Note that the content of the rest of exercises is needed for the project itself, so we highly recommend you to equally effort in all tasks and do not put some aside.