Finanziato dall'Unione europea
NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA

Università di Pisa

Project FORESEEN

Overview on Co-Simulation's State of Art

University of Pisa

April, 8 2024

FORESEEN

PROJECT FORESEEN

FORMAL METHODS FOR ATTACK DETECTION IN AUTONOMOUS DRIVING SYSTEMS

Cyber-Physical Systems

- Cyber-Physical Systems are engineered systems which are increasingly widespread thanks to their versatility.

- A CPS can be seen as a fusion of real-time systems, embedded systems, controls and distributed sensor systems.

- The main peculiarity of those type of systems is the thight coupling between the continuous physical world and the discrete software one.

Cyber-Physical Systems

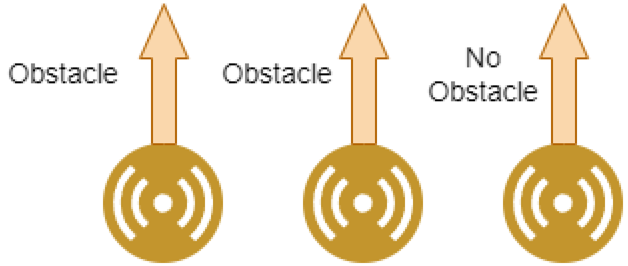
- They have many applications which span from automotive to defense and robotic systems.

- A simple example could be a modern vehicle.
  It has many sensors for different purposes, such as carriage keeping and obstacle detection.

- Sensor’s output are connected to the controller which can for example start the brakes.

Cyber-Physical Systems

- They are often safety critical, for this reason many safety measures must be taken into account.
- For example a vehicle that has a obstacle detection system needs to have more than one sensor and each one should have **replicas.**
- Replicas are used to ensure resilience in case of failure of a single component, thus providing safety over the whole system.



Cyber-Physical Systems

This type of systems are in some cases very expensive, thus making prototypes sometimes infeasible.

It is required a way in which it’s possible to develop, analyze and test such systems under various scenarios, before the actual deployment.

A widely used solution are **Digital Twins.**

Digital Twins – Overview

A Digital Twin is a software replica of a system which we want to obtain an insight of.

This approach is extremely flexible, allowing to model both software and hardware components of the real system, and sometimes even human interactions.

One of the main advantages that this solution bring is the fact that allows to test a system before its deployment.

For example if one wants to test a new algorithm for autonomous driving vehicles, it's best to try it on a Digital Twin and run various simulations rather than test it on the real system.

Digital Twins - Overview

Based on the Digital Twin's integration level, many authors divide them in 3 subclasses:

Digital Model
the data between the physical and digital object are exchanged manually, due to which any changes in the state of the physical object are not reflected in the digital one directly, and vice versa

Digital Shadow
the data from the physical object flow to the digital automatically, but the other way around is still manual. As a result, any change in the physical object can be seen in its digital copy, but not vice versa

Digital Twin
there is an automatic bidirectional flow of data between the physical and digital object. Therefore, the changes in either object, physical or digital, directly lead to changes in the other

Digital Twins - Characteristics

- **High fidelity**:

  The digital twin must be as nearly identical as possible to the real system.

  This means that it must implement the functionalities of both software and hardware components of such system.

  Having a replica with high degree of accuracy is a crucial aspect of the Digital Twin's approach.

Digital Twins - Characteristics

- **Dynamic**:

  If one runs the digital twin in parallel with the physical counterpart, since the latter evolves in a continuous manner, the software copy needs continuous data exchanges from the physical world, thus allowing the twin to evolve aswell.

  For this reason is also said to be **Self-evolving**, meaning that every change in the physical counterpart is reflected also in the digital twin.

Digital Twins - Characteristics

- **Multiscale**:

  The Digital Twin incorporates the properties of the physical system both in a macroscopic and microscopic manner, meaning that it takes into account the its size and shape but also surface roughness and others.

- **Multiphysical :**

  A well designe Digital Twin might need to also represent physical properties such as thermodynamic models and stress analysis models, but also material properties like stiffness and strengh.

Digital Twins - Characteristics

- **Hierarchical**:

  Given its multidisciplinarity it might be hard to create a well designed Digital Twin expecially if the physical counterpart is particularly complex.

  A good approach could be so see the System as a series of integrated sub-models. In this way the final Digital Twin will be the composition of many smaller replicas of much more simpler components of the overall system.

Digital Twins - Advantages

- **Cost reduction**:

  Since the replica is entirely made in software, the cost related to the prototype is highly reduced

- **Problem prediction:**

  If the Digital Twin runs in parallel with the physical one, given that it changes depending on what happens on the real world, it could be possible to detect future problems and behave accordingly.

Digital Twins - Advantages

- **Easier maintenance**:

  General maintenance approaches are based of worst case scenarios and heuristic experience. Using a Digital Twin it is possible to simulate various scenarios and find the optimal maintenance strategy.

  Also, since it is possible to run it together with the real system, it is possible to use it to validate it at all times.

Digital Twins - Applications

Since the Digital Twin approach is extremely versatile, it has many different applications, spanning from prototyping and design to online monitoring and data gathering.

Some examples of application are representede in the following slides, regarding **Automotive**, **Agriculture** and **Healthcare**.

Digital Twins - Automotive

An example of application in this scenario would be the usage of a Digital Twin to run simulations of future scenarios of the vehicle taking inputs from the real world, in orther to be able to foresee the need for maintenance, avoiding unexpected breakdowns.

Another example could be the work that will be explained in detail later, where a digital twin is used to generate data and run many simulations to test and devolop security and safety measures to implement on the vehicle, without the need to run real tests on a costy prototype.

Digital Twins - Agriculture

In agriculture Digital Twins are in the early stage of application. An example is the **Robotti** agricultural robot, which uses a digital twin for gathering usefoul data but also for categorizing actions in a workflow and for automated data analysis.

Digital Twins - Healthcare

With the global pandemic COVID-19 Digital Twins started to be utilized in healthcare aswell.

An interesting example is **Cardio Twin**. Their developers believe that through the use of simulations it is possible to foresee Ischemic Heart Desease (IHD) and strokes on a patient.

In general, DTs in healthcare aim at collecting data from real-time systems, which will later be sent to the model for simulations and evaluation.
Finally, historical data will be saved on cloud and used to provide personalized medical treatments.

Simulation to Co-simulation

An example of simulation environment is Simulink, provided by Matlab.

With Simulink its possible to define a model and run simulations changing its parameters.
During each simulation the system will change states and its outputs will be shown depending on special log functions defined by the user.

This approach becomes infeasible when the complexity of the system under analysis grows.
Assuming that a physical system needs to be replicated in software and given its complexity it has to be divided in many smaller component, each developer might want to write it using different modeling environments, such as Matlab or Open Modelica

Also if one would like to change a component in the system, with standard simulation it would be required to change the whole system.

Co-simulation - Overview

The idea behind co-simulation is to have different components, which should be autonomous, that will be handled by a different simulator depending on the modeling environment that was used to develop them.

Each component will be then interconnected through input and output ports with the others.

A co-simulation consists in a series of time steps in which every component evolves and then sends data to other components.

Taking into account the previous issue, if one decides to change an element in the whole system, the only thing that need to be done is to add the new component, which will need to have the same interface in terms of output data.

Co-simulation - Overview

So which are co-simulation's main features?

- Allows to run so called co-simulations using different **simulators**, where each one can handle one or more models.

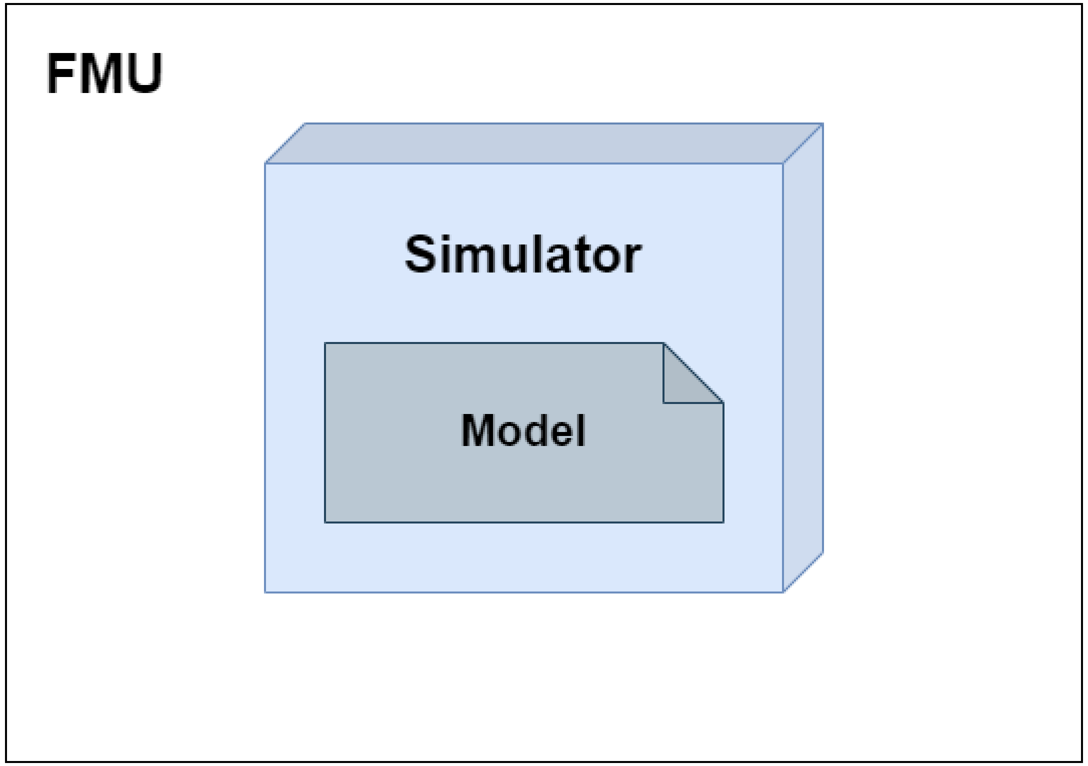
- Throughout the definition of values for parameters of each component of the **multi-model** it is possible to define scenarios and simulate them. Other hyperparameters can be set such as the time step of the co-simulation.

Co-simulation - FMI Standard

A standard widely used for cosimulation, which is the one used in the work that will be presented later, is the **Functional Muckup Iterface** (**FMI**).

Each component is developed in a possibly different environment and later exported as an FMI compliant element, which is called Functional Mockup Unit (FMU).

FMUs are the building blocks of the overall system and they will need to be interconnected between eachother to exchange data during the simulation.



Co-simulation - FMI Standard

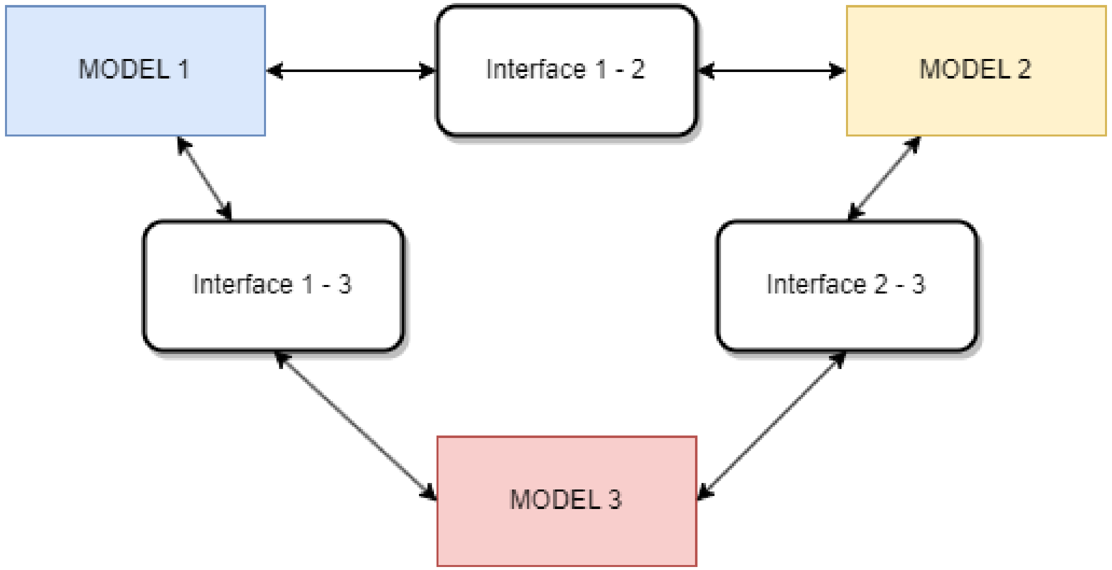
An FMU is a .zip file containing many informations, of which the most important are:

- **An XML File**:
  This contains info about the structure of the Unit, in terms of inputs, outputs and parameters. For each one it will be stored its name, default variable and other usefoul data.
- **C Functions**:
  Those are used to manage the evolution of the simulation and the data exchange between FMUs.
- **Additional Data**:
  For example models or libraries that might be needed during the simulation for that component.

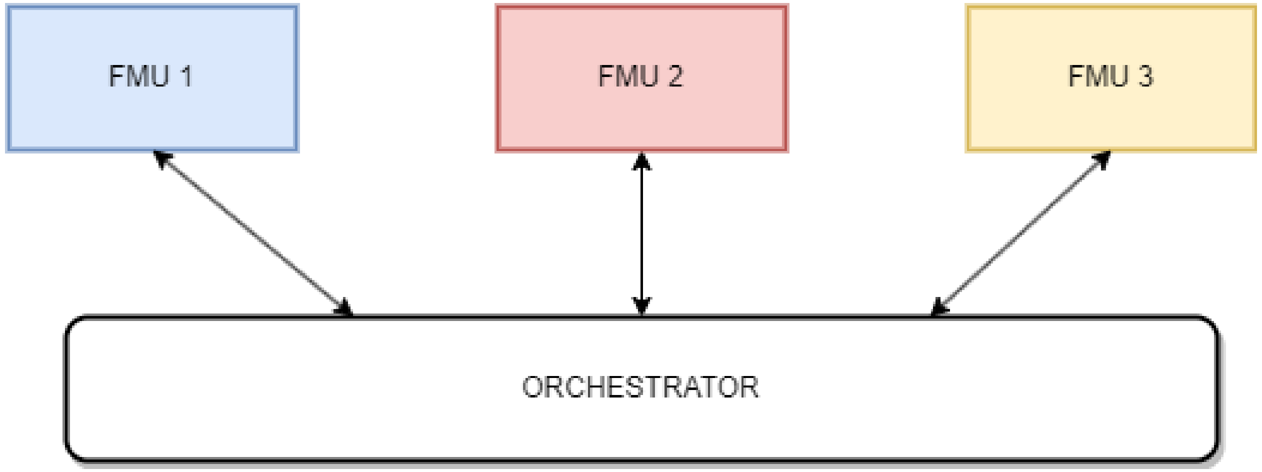
Co-simulation – Communication Frameworks

There are two types of communication frameworks for co-simulations:

Ad Hoc Co-simulation:

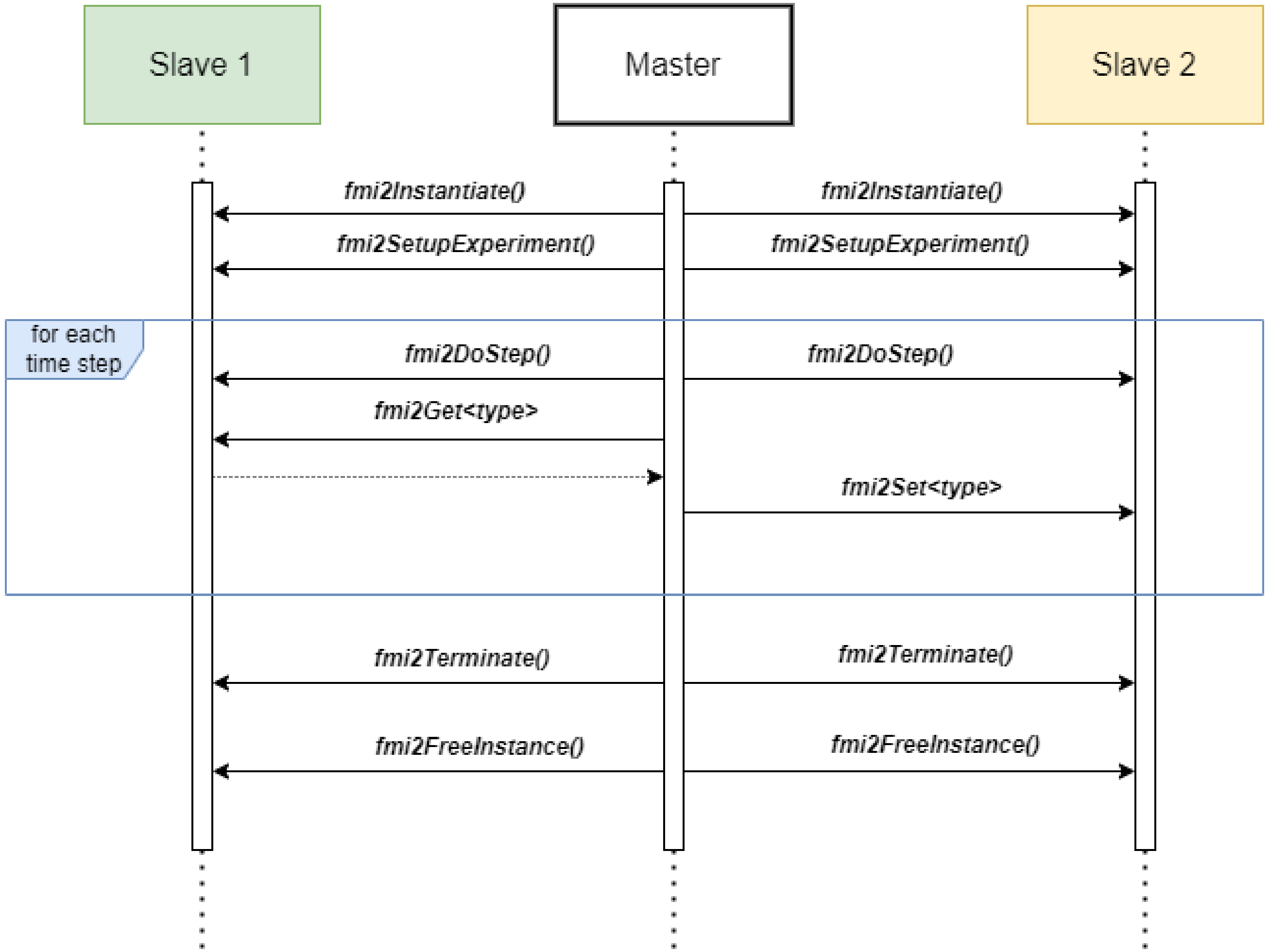


FMI Co-Simulation:



Co-simulation - Example

An example of co-simulation steps when using a General Co-simulation approach is the following:

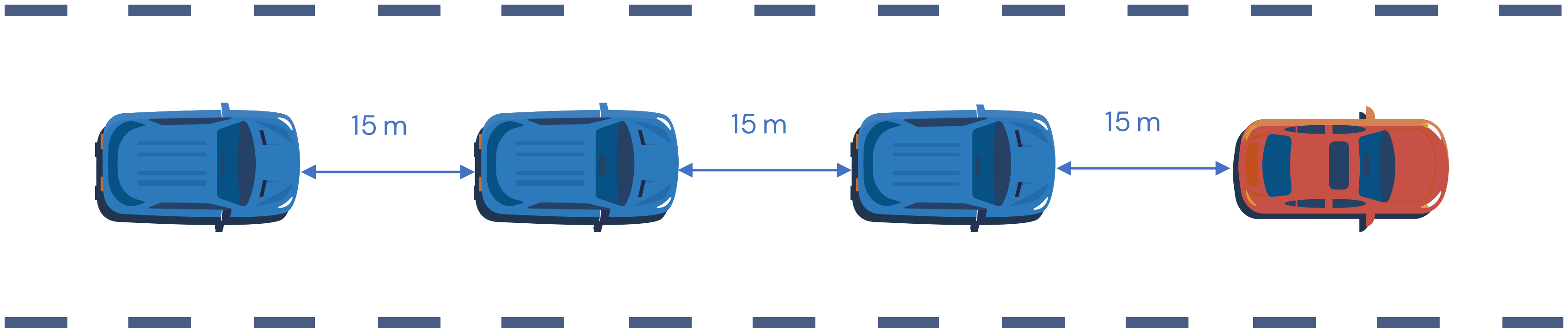


Application Example

To better understand the power of the approaches saw until now, we will present and application to a case study.

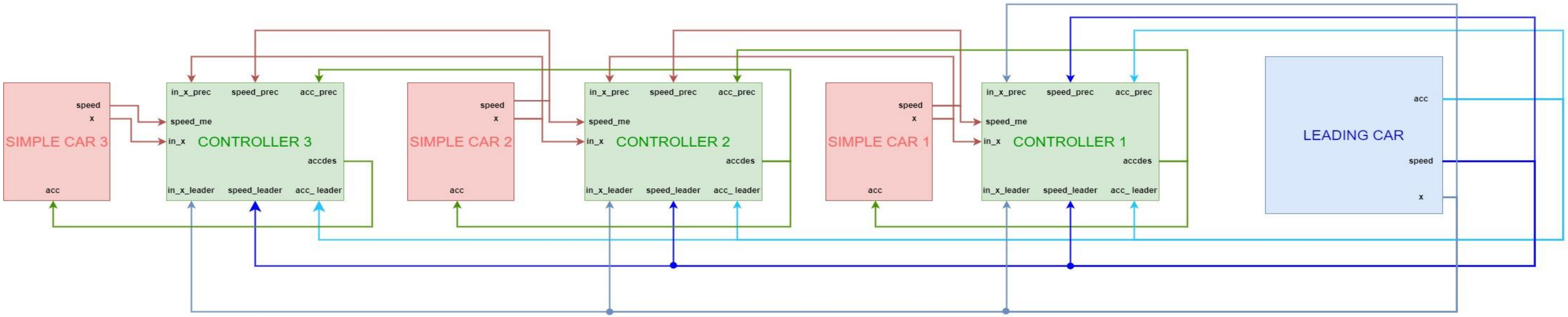
The system under analysis is a vehicle platoon composed of 1 leader and 3 followers.

For simplicity, each follower's goal is to reach a safety distance of 15 with the preceding vehicle.



Application Example

The overall system is structured like it is shown in the picture.
It should be clear the level of complexity, in this case, if we chose to use standard simulation to study it.
Instead, one could simply consider each vehicle as an FMU and just replicate the follower 3 times in a multi-model. This also simplify every possible change to the vehicle that might be apported later. If one wants to change the algorithm, it has to do it just once.



Application Example

A couple of final considerations can be made:

- The degree of accuracy of the results obtained strictly depend on the fidelity of the digital twin with respect to the physical counterpart.

- When defining a scenario, one is free to assign any value to the multi-model's parameters, but physical limitations should be taken into account, e.g. maximum deceleration for a vehicle is -9 m/s^2