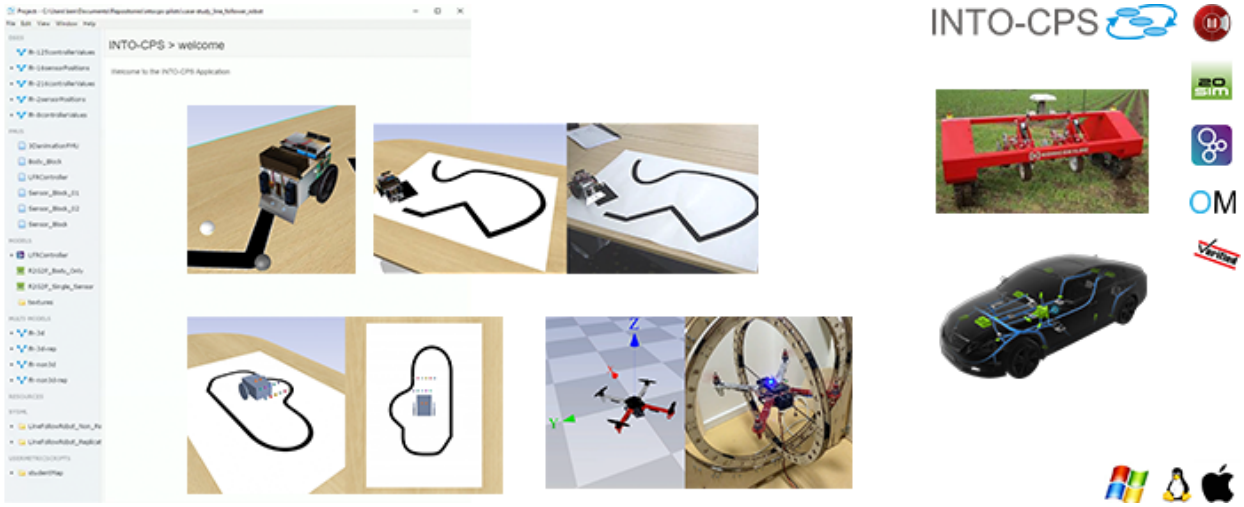
INTO-CPS Application

Release 3.4.9

Feb 24, 2020

Content

Welcome to the documentation of INTO-CPS, the Integrated Tool Chain for Model-based Design of Cyber-Physical Systems.



CHAPTER 1

First Steps

The following section explains how to download and install the software on you local machine.

1.1 Downloading the software

Releases of the INTO-CPS application are available from the INTO-CPS homepage: Releases

1.2 Installing the software

The INTO-CPS application provides a download manager which is the recommended way to obtain the essential parts of the toolchain.

CHAPTER 2

Getting started

This section describes how a project is loaded in the application, how a co-simulation can be configured and executed.

2.1 Starting the app

2.2 Installing additional dependencies

2.3 Loading a project

CHAPTER 3

Functional Mock-Up Interface (FMI)

3.1 Creating FMUs

CHAPTER 4

Tools

The INTO-CPS toolchain consists of several tools for developing Cyber-Physical Systems.

4.1 pyfmu

4.2 Indices and tables

- genindex
- modindex
- search

CHAPTER 5

Developer Documentation

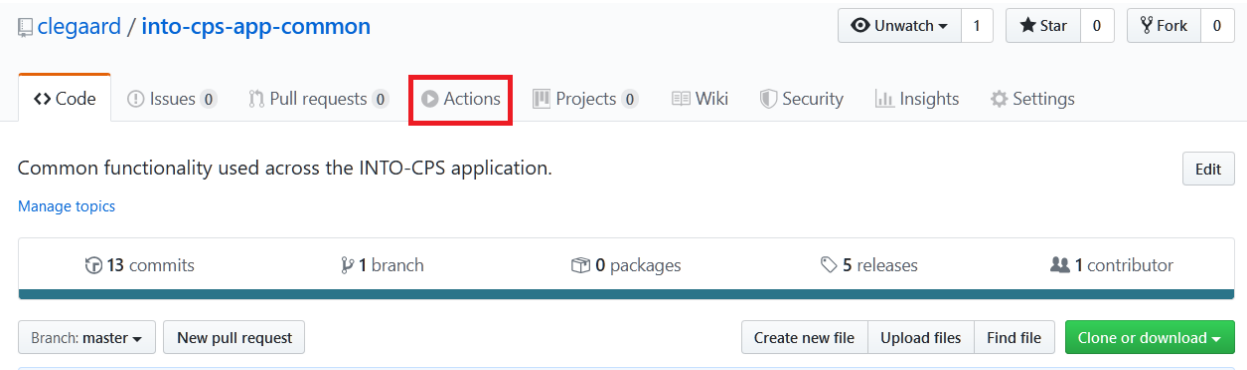
This section of the documentation provides relevant information to maintainers and contributers of the project.

5.1 Architecture

5.1.1 Overview

5.2 Project Guidelines

5.2.1 Continous Integration



5.3 Updating the documentation

5.3.1 Structure of the documentation

The documentation is generated using Sphinx and hosted at Read the Docs. Sphinx is a tool that takes as input either reStructuredText or Markdown and produces documentation in several different formats as illustrated below:

It is recommended to use reStructuredText as an input format.

An approach is used where the documentation of each individual component is stored inside the Git repository along with its source code. For example the *Maestro* co-simulation engine and the desktop client *into-cps-application* are each stored in their own repository:

```
https://github.com/INTO-CPS-Association/maestro.git
https://github.com/INTO-CPS-Association/into-cps-application.git
```

The documentation of each module is then stored inside a folder named *docs* which each contains a top level document *index.rst*. The top level documentation includes the top level document of each sub-project making it possible to compile the entire documentation together.

Each component *Git Submodules*.

The documentation is hosted on *Read the Docs* which automatically recompiles the documentation whenever a push is made to the documentation repository.

5.3.2 Adding documentation to a module

sphinx-quickstart

change name of conf

set theme

5.3.3 Building the documentation locally

The easiest way to install the tool is through PyPi as follows:

```
pip install sphinx
```

Inside the docs folder the two files *make.bat* and *Makefile* are located. On Windows the HTML documentation can be generated by invocing the make.bat script in the docs directory as follows:

```
.\make.bat html
```

Conversely, on Linux the *Makefile* is used in a similar way, by invoking main in the docs directory:

    make html

If successful the html documentation is generated to the _build directory.

Another good resource is the documentation is the Getting Started with Sphinx tutorial on ReadtheDocs website.

5.4 Components

The application consists of multiple packages, each hosted within their own repository, and published to npm under the *@into-cps* scope.

5.4.1 Installing

The packages can be installed through npm using:

```
npm install @into-cps/app-xxx
```

where *xxx* is the module name.

5.4.2 List of modules

@into-cps/application-common

Installing

The library is published to npm under the organization @into-cps. To install the library in a node project use:

The package can be installed with npm by:

```
npm i @into-cps/application-common
```

Tests

Tests are defined using the Jasmine library. This recursively searches the *test* folder for files with names matching *"*.spec.ts"*. The tests can be run using:

```
npm test
```

Continous Integration

CHAPTER 6

Welcome to INTO-CPS-Desktop-Application's documentation!

6.1 Developer Documentation

This section contains documentation meant for developers of the application.

6.1.1 Dependency Checker

This is the documentation of the dependency checker. Here we describe how to check for the presence of JRE, Python etc.

What we check for

JRE, Python.

6.2 Test

6.3 More info

CHAPTER 7

Indices and tables

- genindex
- modindex
- search