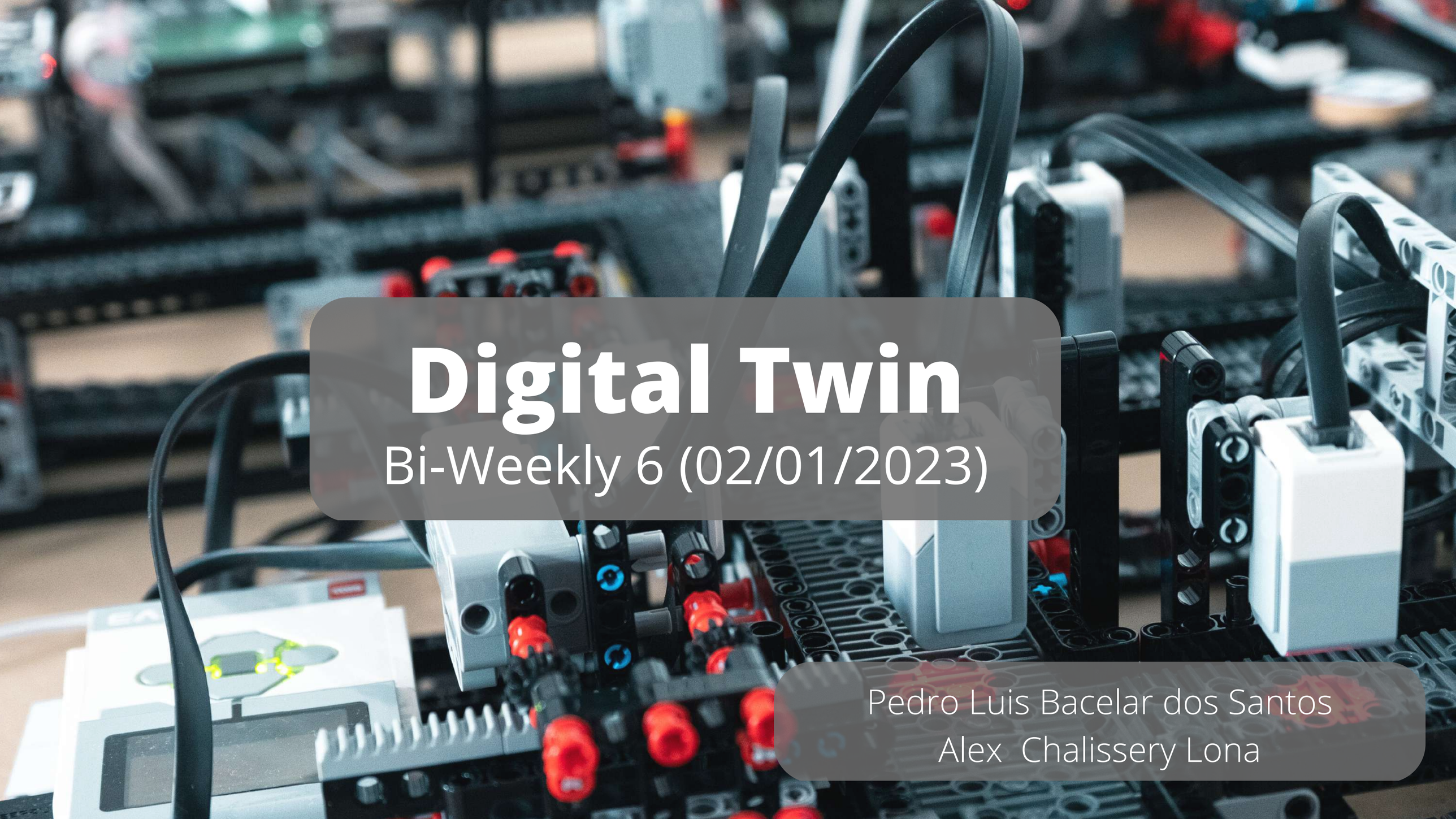


Digital Twin

Bi-Weekly 6 (02/01/2023)

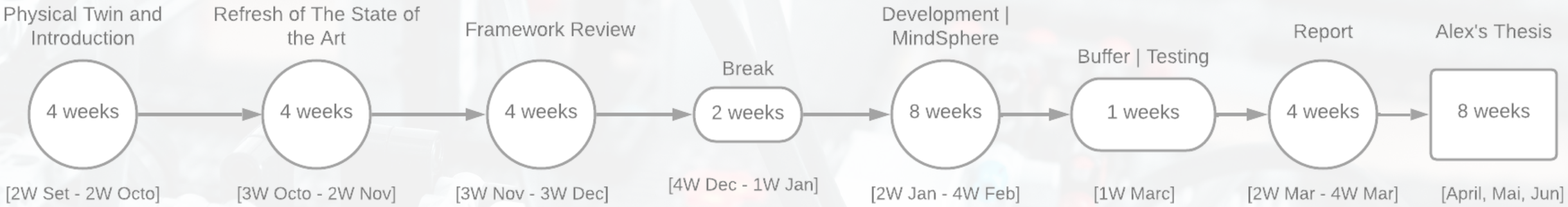
Pedro Luis Bacelar dos Santos
Alex Chalissery Lona

Agenda

- Quick Project Overview
- Report Status
- Previous Work Review
- Current Working in Progress
- Next Steps


Quick Project
Overview

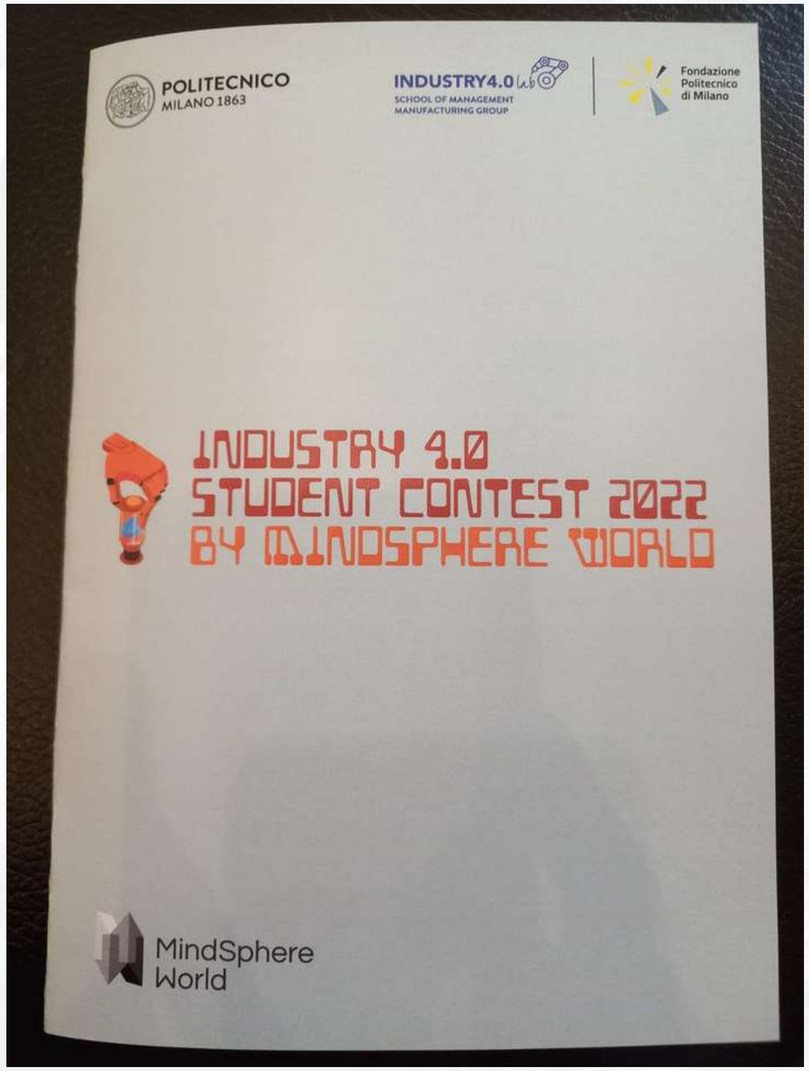
Timeline



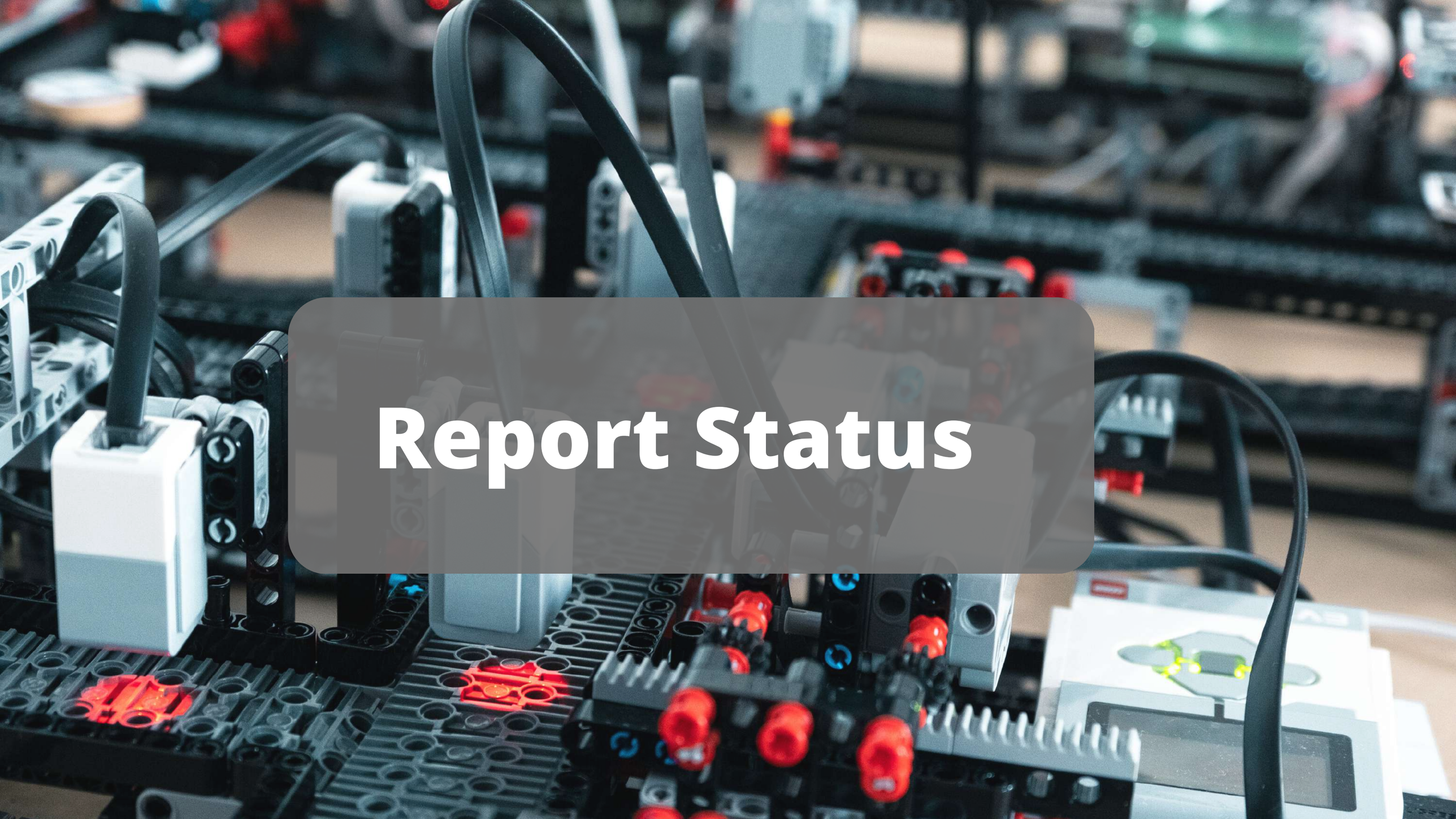
- **Previous Project Review (code review, interviews, library, etc)**
- **Digital Model (generic translator & Simulator)**
- Synchronization
- Validation
- Digital Twin's Services
- Alignment with MindSphere

MindSphere

MindSphere



- Public Awarding ceremony for Phase-I
- Selected for the next development phase
- Opportunity to test the solution at Industry 4.0 Lab
- Opportunity to test the solution with other industry partners
- Possibility to use partner software (Plant Simulation)
- MindSphere Solution: Machine Learning for predicting Remaining Cycle Time **(input to compare with simulation approach)**


Report Status

Report Status

POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Generic Digital Twin for predicting remaining Cycle Time of systems

Tesi di Laurea Magistrale in
Xxxxxxx Engineering - Ingegneria Xxxxxxx

Author: **Alex & Pedro**

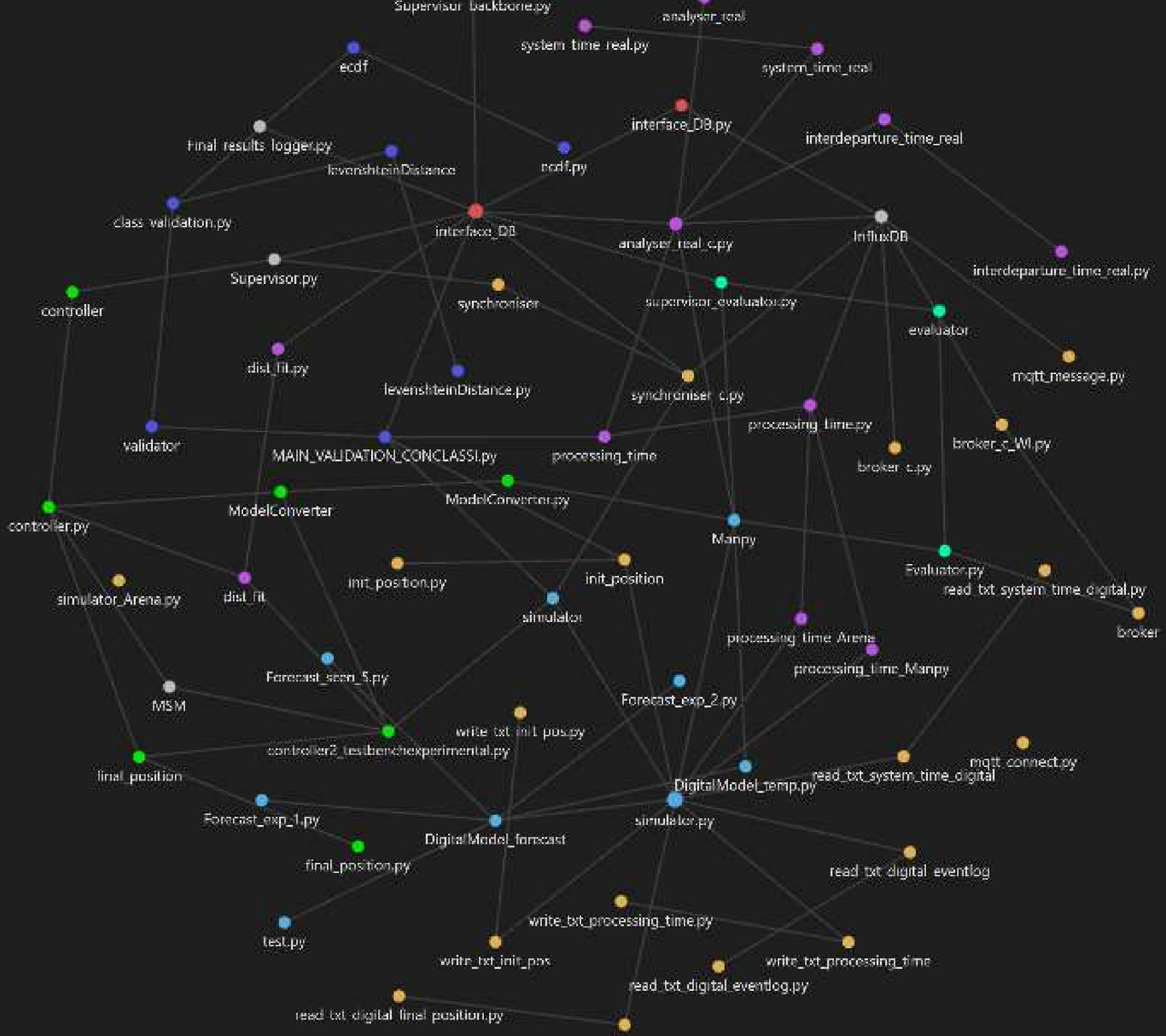

Previous Work
Review

Understanding the Existing Architecture

- supervisor class
- analytics class
- controller class
- validation class
- synchroniser class
- MSM
- simulator class
- database class

Mapping Code Functions

Obsidian



Work Review

Interview with previous researchers

- Model generation - Giovanni Lugarasi
- Validation of DIgital twins - Sofia Ganzemi
- Syncronisation of Digital twins - Edoardo Passarin

Main Aim of the Interview:

- Discussion on their works
- Clarification of doubts on the respective topics
- Know about their research experience

Available Choices

Simulator:

- ManPy / DREAM
  - Specific for Manufacturing
  - Dedicated functions
  - Written on Python2
  - Lack of updates and appropriate documentations
  - Built on SimPy

Available Choices

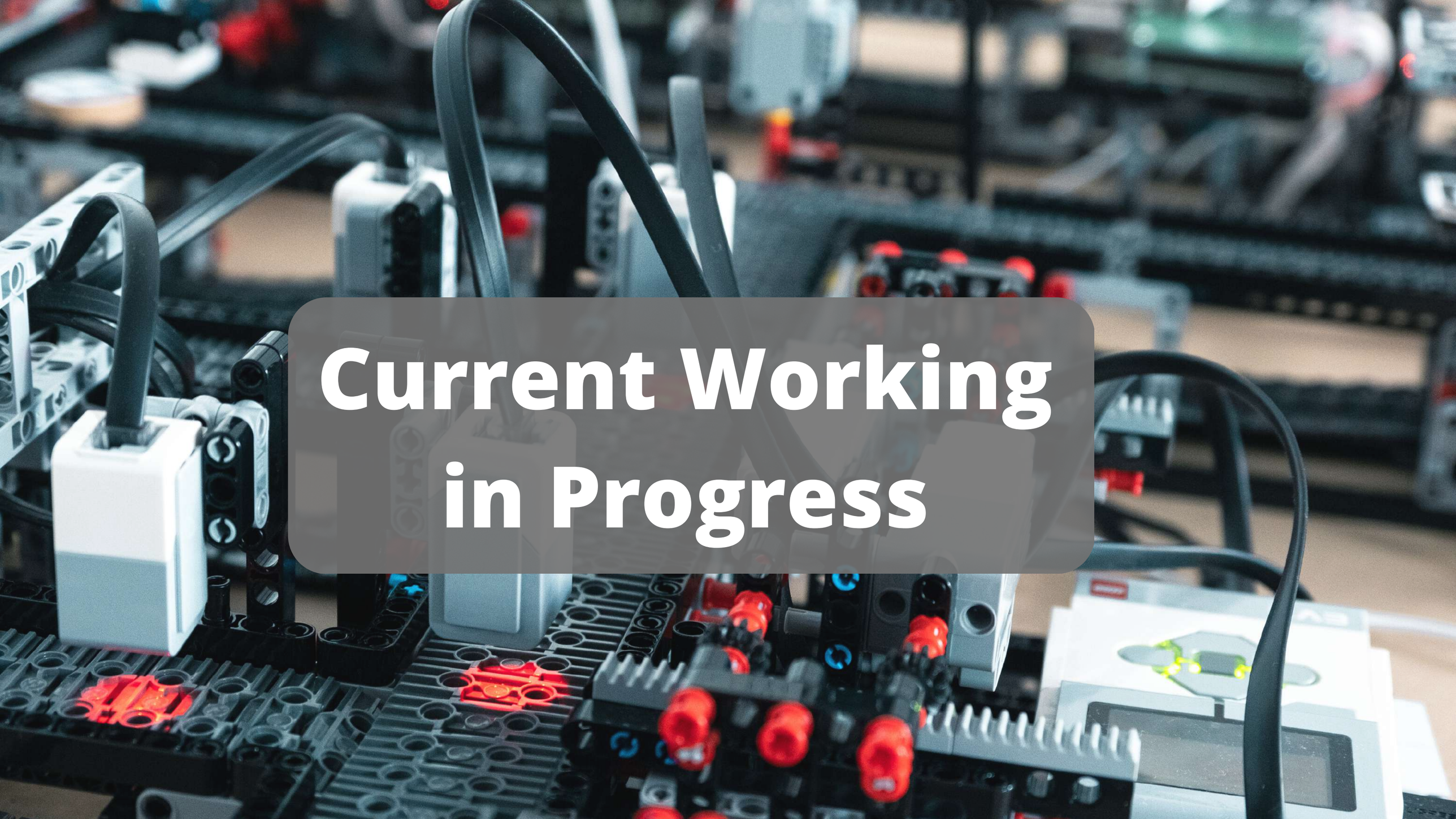
Simulator:

- SimPy
  - Generic package
  - Applicable to a wide variety of scenarios
  - continous updates and documentations

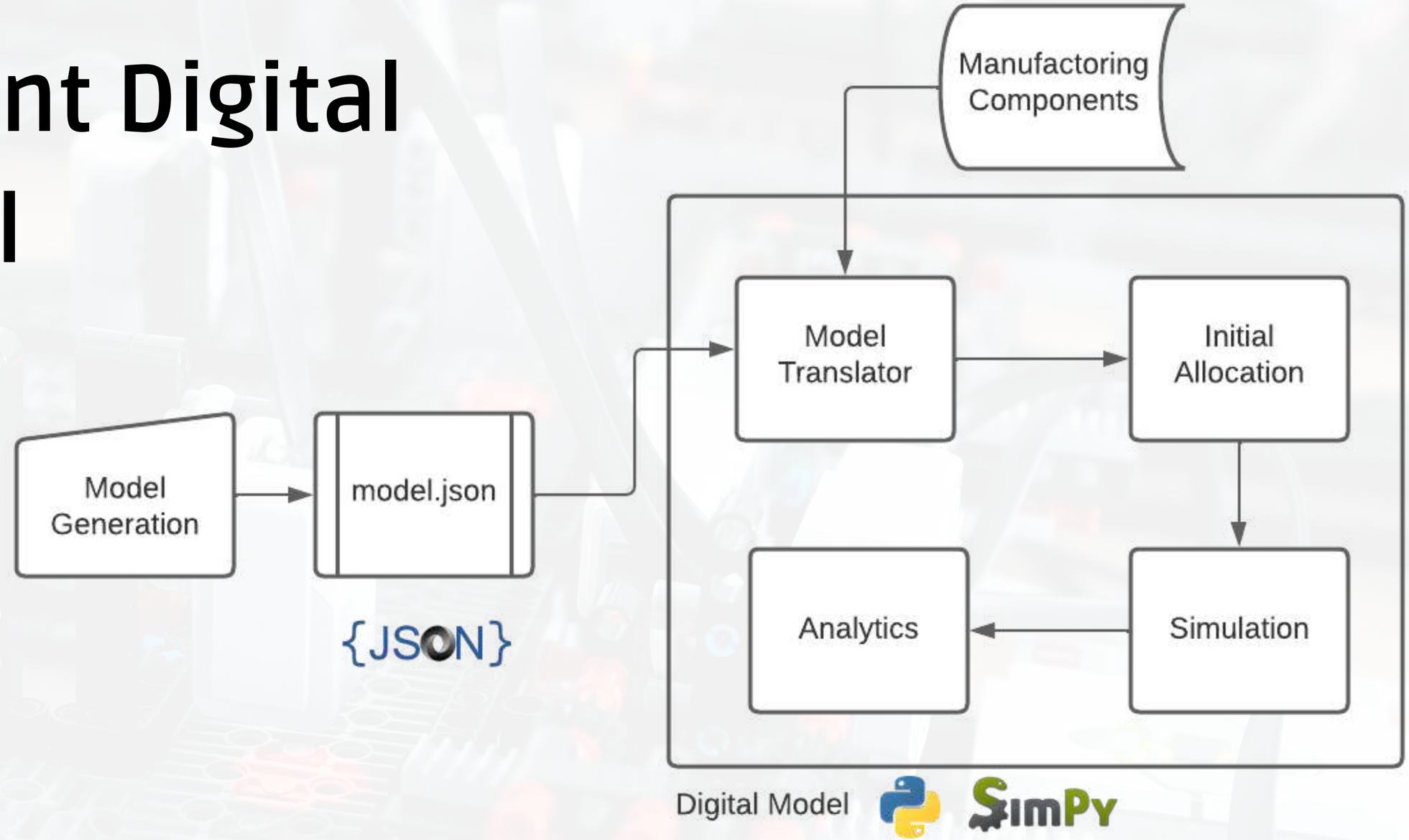
Available Choices

Simulator:

- Commercial software
  - Arena simulation
  - Technomatix plant simulation
  - Good GUI
  - complex logics for automating between Python and the respective softwares


Current Working
in Progress

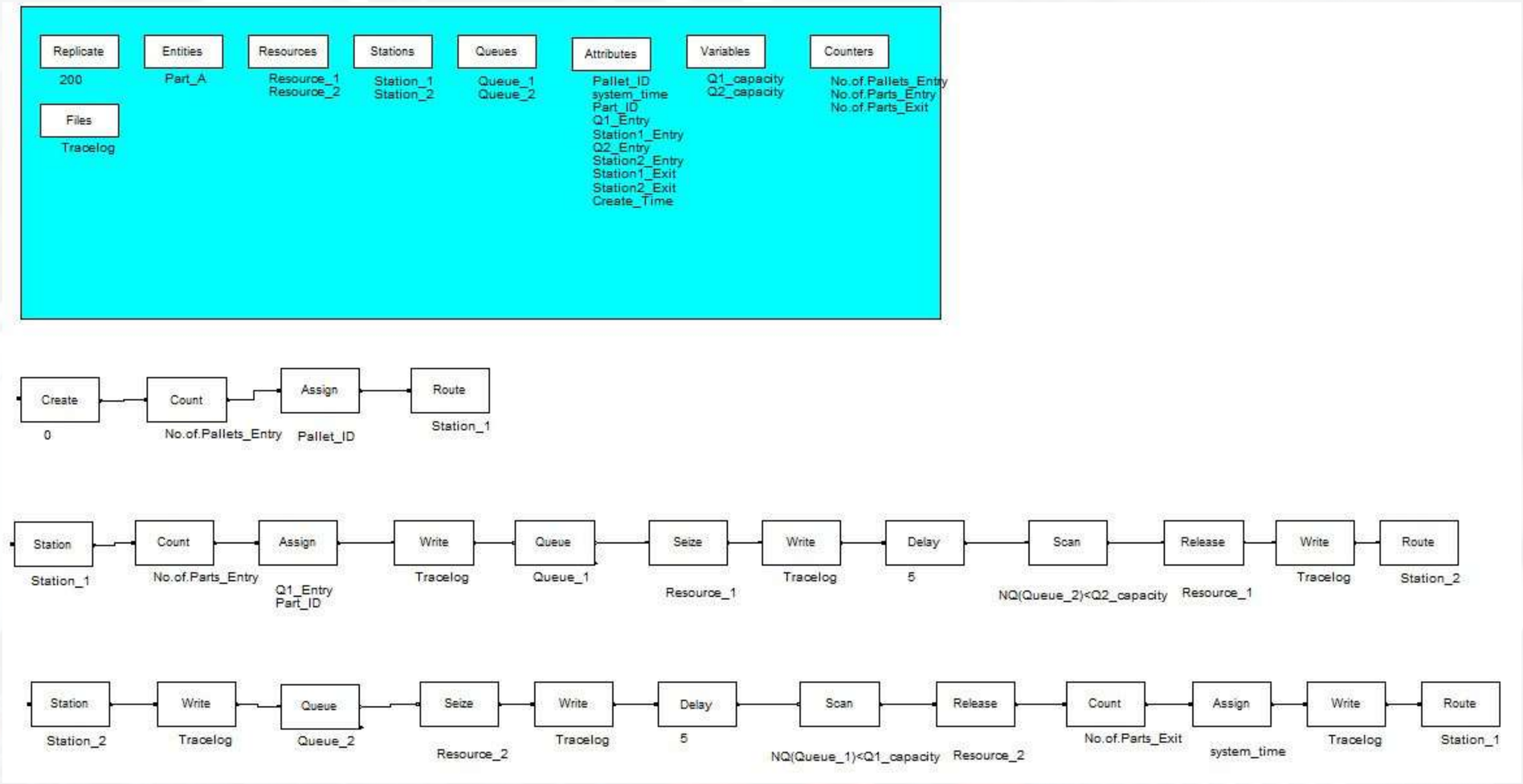
Current Digital Model



How to use it? (7 lines)

```
environment = simpy.Environment()
model_2stations_closed_path = "models\model_2stations_closed.json"
model_2stations_closed = Model(name= "model_2stations_closed",model
model_2stations_closed.model_translator()
model_2stations_closed.verbose()
model_2stations_closed.run()
model_2stations_closed.analyze_results()
```

Validation: 2s Model (Arena)



Validation: 2s Model (Arena)

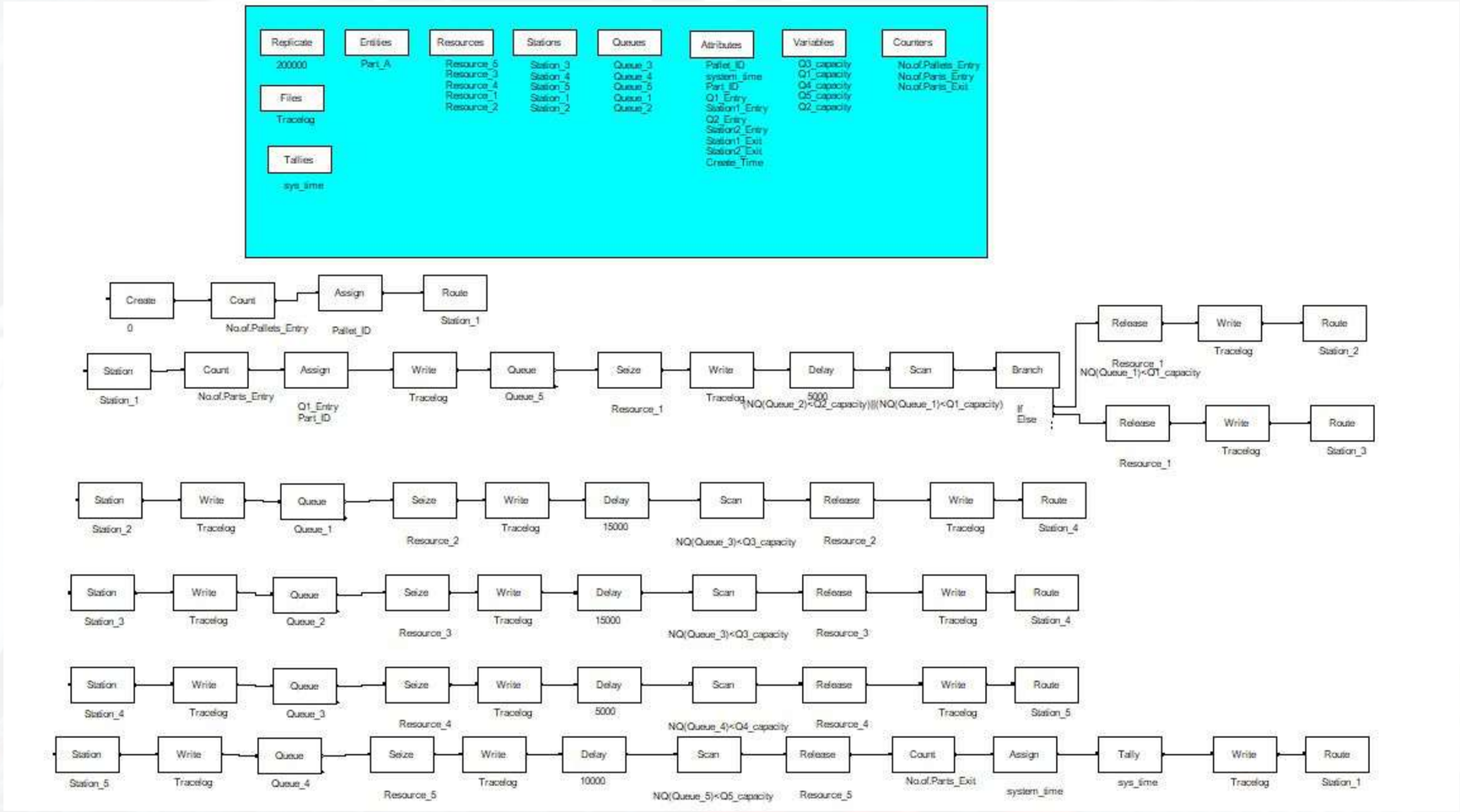
```
######### Running Analysis #########
Number of Parts finished: 39
Total time of Simulation: 201
```

```
>>> *** SYSTEM THROUGHPUT: 0.19402985074626866
>>> Cycle Time of each part:
[10, 15, 20, 25, 20, 20, 20, 20, 20, 20, 20, 20
20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20]
- Maximum Cycle Time: 25
- Minimum Cycle Time: 10
*** AVERAGE CYCLE TIME OF THE SYSTEM: 19.743589
```

Identifier	Count	Limit
No.of.Pallets_Entry	4	Infinite
No.of.Parts_Entry	43	Infinite
No.of.Parts_Exit	39	Infinite

```
:d Q2 at: 35.000000
:d Station2 at: 35.000000
:d Station1 at: 35.000000
| Station2 at: 40.000000 Parts_Exited: 7.000000 System Time: 20.0000
'ed Q1 at: 40.000000
| Station1 at: 40.000000
```

Validation: 5s Model (Arena)



Validation: 5s Model (Arena)

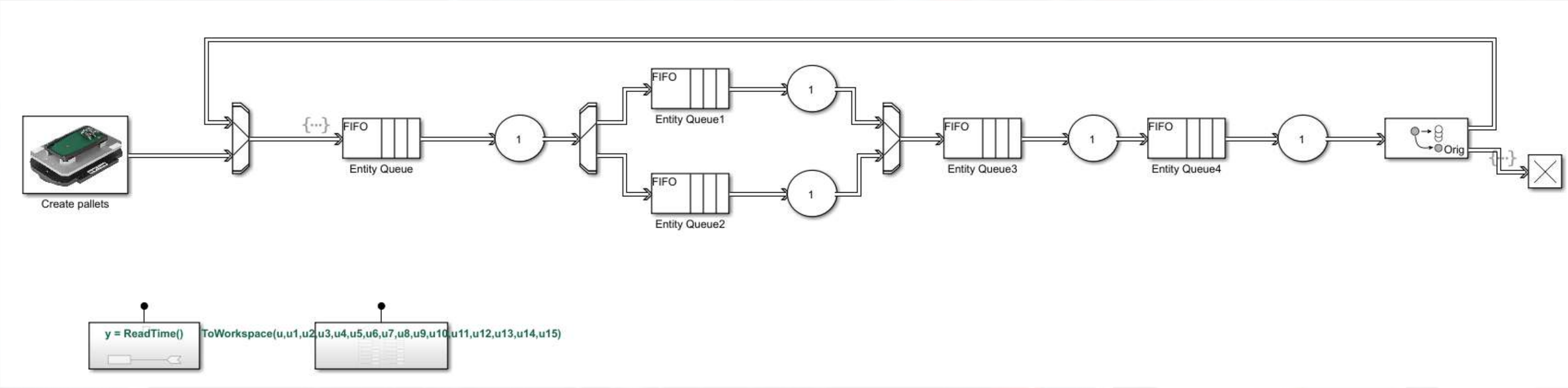
```
######### Running Analysis #########
Number of Parts finished: 17
Total time of Simulation: 200100
```

```
>>> Cycle Time of each part:
[35000, 50000, 65000, 75000, 85000, 95000, 105000, 115000,
185000, 195000]
- Maximum Cycle Time: 195000
- Minimum Cycle Time: 35000
*** AVERAGE CYCLE TIME OF THE SYSTEM: 119062.5 [time unit]
```

Identifier	Count	Limit
No.of.Pallets_Entry	100	Infinite
No.of.Parts_Entry	117	Infinite
No.of.Parts_Exit	17	Infinite

Identifier	Average	Half Width	Minimum	Maximum	Observations
sys_time	1.1971E+5	(Insuf)	35000.	2.0000E+5	17
Part_A.VATime	--	--	--	--	0
Part_A.NVATime	--	--	--	--	0
Part_A.WaitTime	--	--	--	--	0
Part_A.TranTime	--	--	--	--	0
Part_A.OtherTime	--	--	--	--	0
Part_A.TotalTime	--	--	--	--	0
Queue 2.WaitingTime	00000	(Insuf)	00000	00000	24

Validation: 5s Model (Simulink)



Validation: 5s Model (Simulink)

```
######### Running Analysis #########
Number of Parts finished: 17
Total time of Simulation: 200100
```

```
>>> Cycle Time of each part:
[35000, 50000, 65000, 75000, 85000, 95000, 105000, 115000,
185000, 195000]
- Maximum Cycle Time: 195000
- Minimum Cycle Time: 35000
*** AVERAGE CYCLE TIME OF THE SYSTEM: 119062.5 [time unit]
```

```
number_of_parts_finished = length(out.Time_Out_System)
```

```
number_of_parts_finished = 17
```

```
cycle_time = out.Time_Out_System - out.Time_In_System
```

```
cycle_time = 17×1
          35000
          50000
          60000
          70000
          80000
          90000
         100000
         110000
         120000
         130000
           ⋮
```

```
avg_cycle_time = mean(cycle_time)
```

```
avg_cycle_time = 1.1971e+05
```

Strengths & Weakness

- Flexibility
- Control Over Simulation
- Easy to integrate with database
- Components generations automated
- matching with other softwares
- Library

- Hard coding
- Visualization
- Possible bugs

Next Steps

Timeline and Next Steps

- Validate Digital Model for more complex system and different cases
- Look for more python libraries (?)
- Integrate with the Database
- Validation Implementation
- Synchronization
- Predict Remaining Cycle Time


Thank You