salabim

Discrete event simulation in Python

What’s this poster?

- An introduction to Discrete Event Simulation
- An overview of available tooling
- Introduction to salabim
- Some sample models and output

Just to wet your appetite and join the poster session

What is Discrete event simulation (DES)

A **discrete-event simulation** (**DES**) models the operation of a system as a (discrete) sequence of events in time. Each event occurs at a particular instant in time and marks a change of state in the system.[1] Between consecutive events, no change in the system is assumed to occur; thus the simulation time can directly jump to the occurrence time of the next event, which is called **next-event time progression**.

What is used for?

- Materials handling logistics
- Air ports
- Sea ports
- Hospitals
- Warehousing
- Robotics
- Network analysis/design
- Crowd simulation
- …

What tools are available (1)?

High end GUI specialized packages, like AnyLogic, Simio, Arena

- high quality 3D animation
- GUI based
- mostly domain specific language or complicated interface to external languages (mainly Java and C++)
- very expensive (think: USD 10000 – 20000 per user)

What tools are available (2)?

Language based solutions

Goes back to the mother of OO languages: Simula (80s)

Several generations of similar tooling

Several support languages (R, Julia and, of course, Python)

Python packages (open source):

- SimPy
- salabim

salabim features

Process description methodology, with

- activate, passivate, hold, request, interrupt, ...
  all based on coroutines, implemented as generators
  no threads, no async
- Real time animation, including video production
- Statistical distributions
- Automatic data collection (monitoring)

Sample model: bank with 3 clerks

Clients arrive randomly, according to a given distribution

Bank employees serve one client at a time, with a uniform service time

One queue for all clerks

Very simpliefied model: bank with 3 clerks

```
import salabim as sim


class Customer(sim.Component):
    def process(self):
        self.enter(waitingline)
        for clerk in clerks:
            if clerk.ispassive():
                clerk.activate()
                break  # activate at most one clerk


class Clerk(sim.Component):
    def process(self):
        while True:
            while len(waitingline) == 0:
                yield self.passivate()
            self.customer = waitingline.pop()
            yield self.hold(sim.Uniform(20, 30))


env = sim.Environment(trace=True)
sim.ComponentGenerator(Customer, iat=sim.Uniform(5, 15), force_at=True)
clerks = [Clerk() for _ in range(3)]

waitingline = sim.Queue("waitingline")

env.run(till=50000)
waitingline.print_histograms()
waitingline.print_info()
```

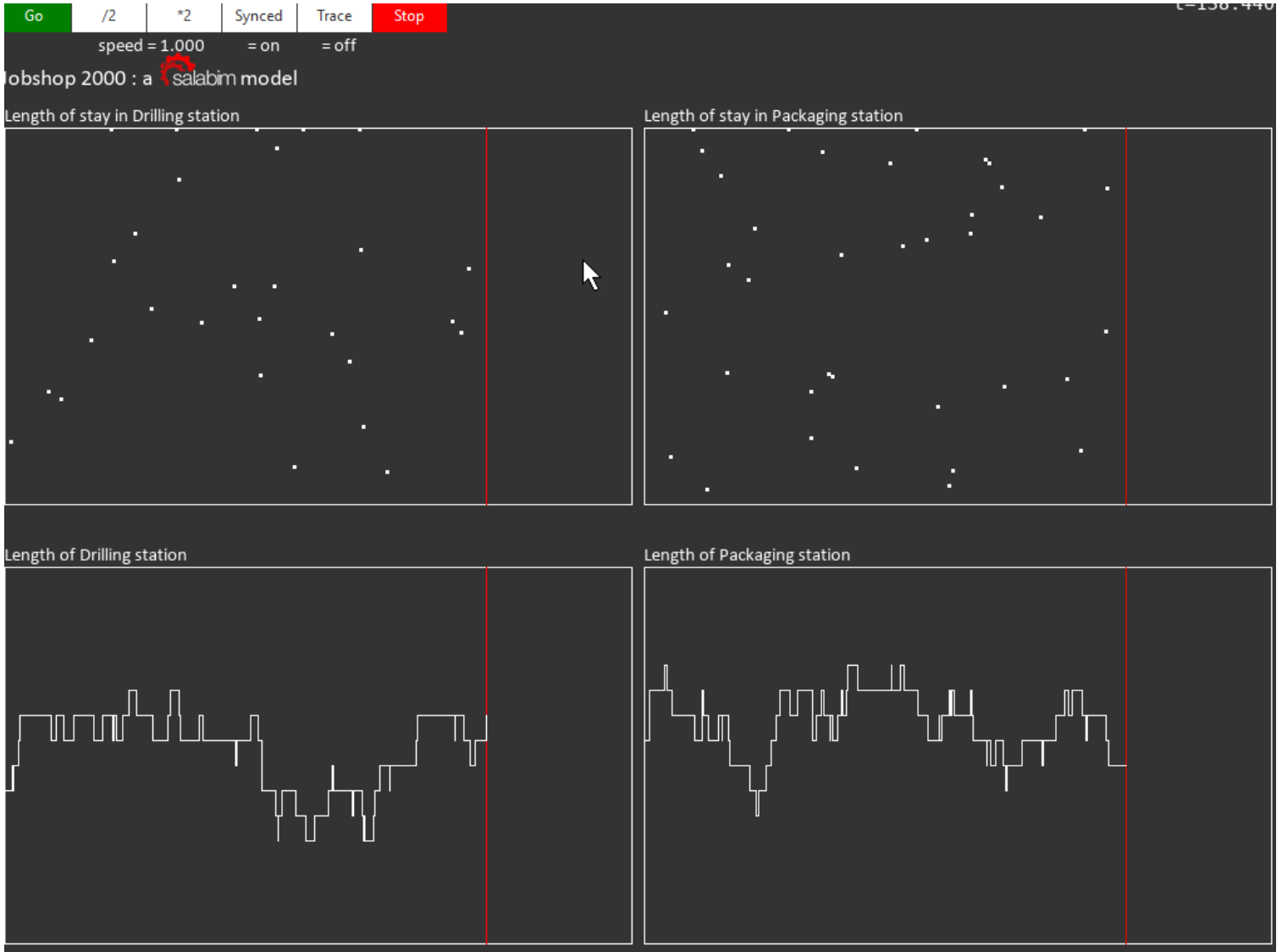
Sample model: bank with 3 clerks

```
26     736.238 Customer.generator.0 current
31                                  customer.73 create
31                                  customer.73 activate                scheduled for    736.238 @    6  process=process
31                                  Customer.generator.0 hold +11.724   scheduled for    747.963
 6     736.238 customer.73          current
 7                                  customer.73                         enter waitingline
12                                  customer.73 passivate
21+    737.616 clerk.2              current
22                                  customer.68 activate                scheduled for    737.616 @   12+
20                                  customer.71                         leave waitingline
21                                  clerk.2 hold +30.000                scheduled for    767.616 @   21+
12+    737.616 customer.68          current
31                                  customer.68 ended
21+    743.003 clerk.0              current
22                                  customer.69 activate                scheduled for    743.003 @   12+
20                                  customer.72                         leave waitingline
21                                  clerk.0 hold +30.000                scheduled for    773.003 @   21+
12+    743.003 customer.69          current
31                                  customer.69 ended
26     747.963 Customer.generator.0 current
31                                  customer.74 create
31                                  customer.74 activate                scheduled for    747.963 @    6  process=process
31                                  Customer.generator.0 hold +5.023    scheduled for    752.986
 6     747.963 customer.74          current
 7                                  customer.74                         enter waitingline
12                                  customer.74 passivate
26     752.986 Customer.generator.0 current
31                                  customer.75 create
31                                  customer.75 activate                scheduled for    752.986 @    6  process=process
31                                  Customer.generator.0 hold +8.377    scheduled for    761.363
 6     752.986 customer.75          current
 7                                  customer.75                         enter waitingline
12                                  customer.75 passivate
21+    753.935 clerk.1              current
22                                  customer.70 activate                scheduled for    753.935 @   12+
20                                  customer.73                         leave waitingline
21                                  clerk.1 hold +30.000                scheduled for    783.935 @   21+
```

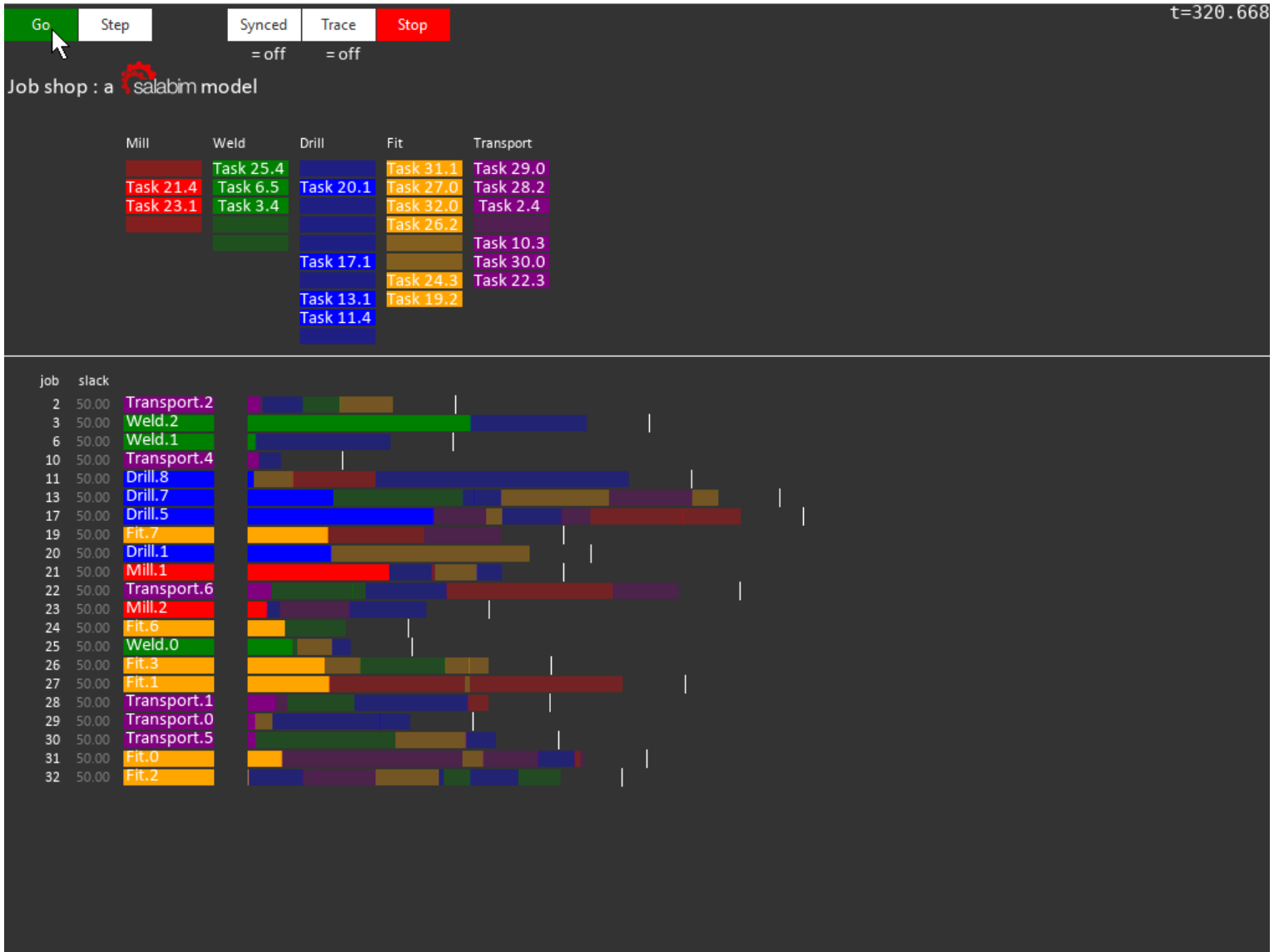
Sample model: bank with 3 clerks

```
Histogram of Length of waitingline
                        all    excl.zero         zero
-------------- ------------ ------------ ------------
duration          50000        50000            0
mean                  8.427        8.427
std.deviation         4.852        4.852

minimum               0            0
median                9            9
90% percentile       14           14
95% percentile       16           16
maximum              21           21

           <=      duration     %  cum%
            0          1500.619   3.0   3.0 **|
            1          2111.284   4.2   7.2 ***  |
            2          3528.851   7.1  14.3 *****      |
            3          4319.406   8.6  22.9 ******             |
            4          3354.732   6.7  29.6 *****                  |
            5          2445.603   4.9  34.5 ***                        |
            6          2090.759   4.2  38.7 ***                           |
            7          2046.126   4.1  42.8 ***                              |
            8          1486.956   3.0  45.8 **                                 |
            9          2328.863   4.7  50.4 ***                                   |
           10          4337.502   8.7  59.1 ******                                    |
           11          4546.145   9.1  68.2 *******                                        |
           12          4484.405   9.0  77.2 *******                                           |
           13          4134.094   8.3  85.4 ******                                              |
           14          2813.860   5.6  91.1 ****                                                  |
           15          1714.894   3.4  94.5 **                                                       |
           16           992.690   2.0  96.5 *                                                          |
           17           541.546   1.1  97.6                                                             |
           18           625.048   1.3  98.8 *                                                             |
           19           502.291   1.0  99.8                                                               |
           20            86.168   0.2 100.0                                                               |
           21             8.162   0.0 100                                                                  |
          inf             0       0   100
```

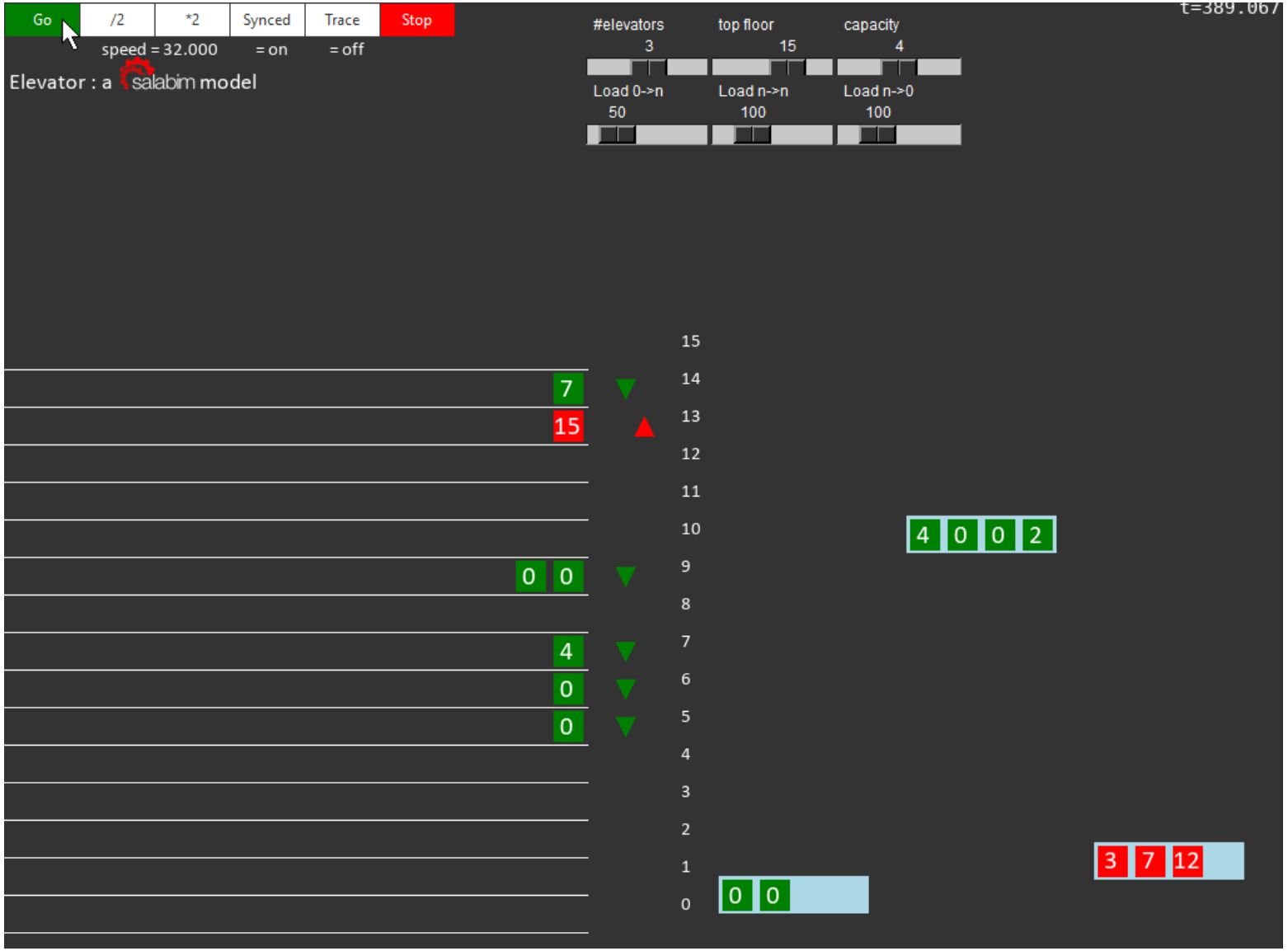
Demo real time monitoring (job shop model)



Demo real time animation (job shop model)



Demo real time animation (elevator model)



Technical details

No dependencies, apart from

- PIL for animation
- OpenCV for video production

Runs also on iPads under Pythonista

Active Google Groups user group

Very easy to install (from PyPI or just include ONE source file)

Also useful for creative animations

Interested in  ?

Join the poster session

Friday 24 July, 13:15 – 14:00 CET

Or visit www.salabim.org