Intro to dynamic simulations software architecture

(Turtle Sim developer edition)

Erik Palenčík

Table of Contents

1. Content ..... *Error! Bookmark not defined.*

2. Intro *4*

1. Author introduction ..... Error! Bookmark not defined.
2. Motivation to write the book 4
3. Motivation to create Turtle Engine 4

3. Simulation theory *6*

4. Monte Carlo method 6
5. Probability distribution 6
6. Statistics 6

4. Heuristics *7*

7. Linear programming 7
8. Genetics algorithm 7

5. Data structures and algorithms *8*

9. Queue 8
10. Buffer 8
11. Maps 8
12. Djikstra 8

6. Entities and definitions *9*

13. Basic entities 9
14. Smart entities 9

7. Platform architecture overview *10*

15. Entities and actors explanation 10
16. Simulation initialization 10

8. Statistics and analysis *12*

9. Working with turtle *13*

10. Simulation software problems and challenges *14*

17. Dependency tree 14

1. Intro

In modern companies the word simulation is necessity:

What can simulation software help us:

- Verify our idea
- Show weak spots
- E-documentation

1. About the author

Hello, my name is Erik Palenčík, while writing this book I work as Senior Software developer and starting my PhD study on the faculty of the mechanical engineering in Žilina – Industry Engineering field.

During study of IT (which I failed) I continued in this field by work, but I finished bachelor's in management and master's degree in Industry Engineering. I have now 9 years of experience with creation of 3D applications on web or desktop, and multiple experiences with creation of VR simulation of company processes for training purposes. Also last few years I actively develop digital twin platform for monitoring, analysis and optimalizations.

2. Motivation to write the book

As student I went to multiple books – maybe great in simulations community, but for me as free time simulation worker, they were useless, after reading four hundred pages, I read no information. So, this was the reason why to create my own publication to help students and bring fun to the simulations world.

3. Motivation to create Turtle Engine

During my work I started to experiment with Three JS and web and backend technologies, as my skillset grown and my life passed through, I decided to create my own digitalization platform – for uploading 3D models, point clouds, 360 panoramas and another interactive content.

By the time I undergone Modeling and Simulations and learning about manual simulation in this course I decided to create my own simulation engine with strong 3D visualization and knowledge of web technologies. Also, terrible UI and performance of another simulation tools like Anylogic (where UI performs terrible even on high end PC) and strong licensing like in Tecnomatix Plant simulation I decided to create my own platform, which should be modern, open source, and deployable everywhere.

Another crucial motivation was to create my own, where I will be author, and I will be free to share source code to every company or person I will be ever in contact with. There was also motivation in moving forward my software architecture skills – you can read a million books, but you will **NEVER** move forward until you sit down and try it yourself.

Turtle Engine is part of my free time activities and my university research. Platform also contains modules to boost learning and visualization of students, not only to make practical examples.

2. Simulation theory

4. Monte Carlo method

Monte Carlo is the most popular simulation method

5. Probability distribution

6. Statistics

3. Heuristics

7. Linear programming
8. Genetics algorithm

4. Data structures and algorithms

9. Queue
10. Buffer
11. Maps
12. Djikstra

5. Entities and definitions

13. Basic entities

The purpose of basic entities is to create simplified simulation with basic set of basic behavior.

Spawn

Spawn is behavior responsible for creation of entities. Spawn can work in the following modes:

1. Constant spawn time – spawn every 5 seconds
2. Normal distribution spawning – spawn in NormDist(10, 30)

Process

Delay

Buffer

Queue

Merge

Split

Switch

Sink

An opposite to spawn, sink destroys entities, sink is responsible for removing entity from the world.

14. Smart entities

The purpose of smart entities is to add additional value to

6. Platform architecture overview

Many computer languages support different architecture allowing different patterns and code architecture. Because Turtle Sim was developed with usage of Golang (as backend) – functional static typed language and JavaScript (frontend – React and Three JS) - duck typing OOP language, there are special kinds of architectures used in this cases.

15. Entities and actors explanation

In Turtle we differentiate **Entities** and **Actors** – in IT theory they can be the same thing, but in this cases I decided to use this two names separately.

Entities are objects in world which has some role and behavior, e.g. Process, Spawn, Sink, ...

Actors are objects traveling between entities being able to change it's state, wear states.

16. Simulation initialization

As every system also Turtle models have to somehow prepare load data, prepare them and then simulate.

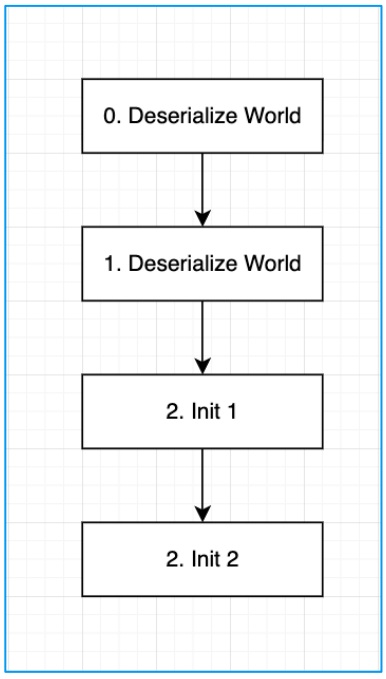


Figure 1 Simulation Initialization

The following loading stages are:

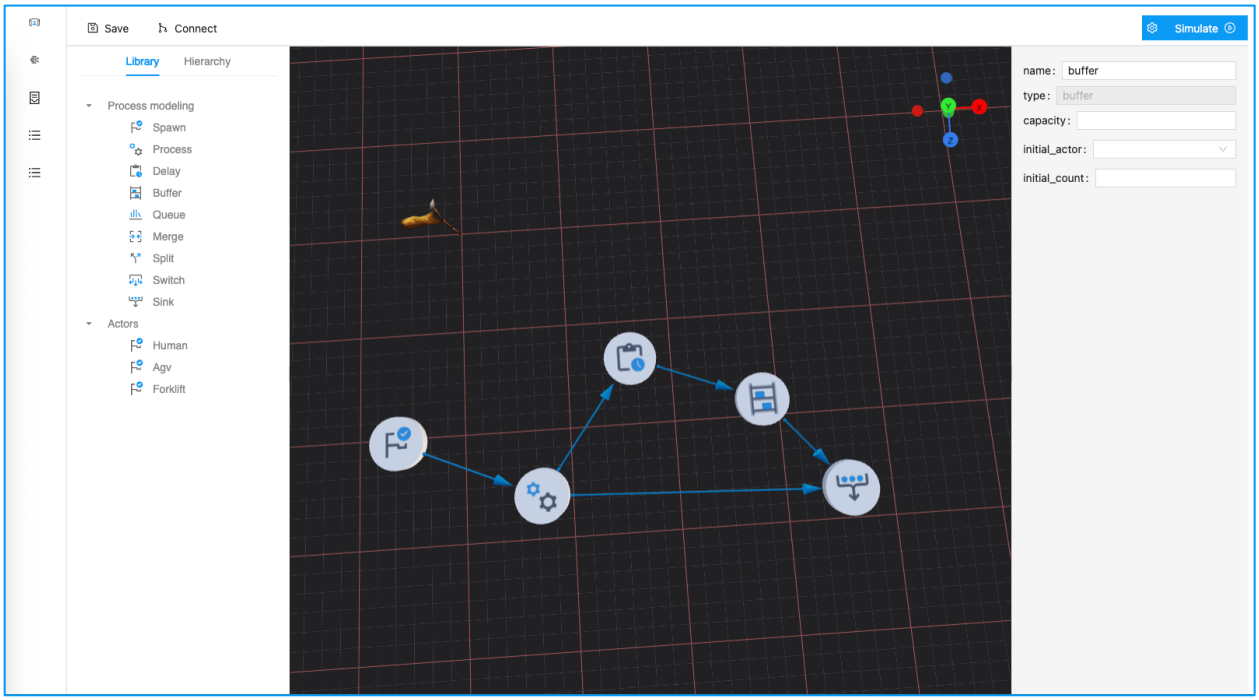
0. **World deserialization** – in this phase world is loaded from database.

1. **Init 1** – in this phase there is called function **Init1** in which you entity should prepare data
2. _____

7. Statistics and analysis

8. Working with turtle

After opening simulation platform, we can find similar interface being seen in environments like Anylogic or Tecnomatix Plant simulation.



9. Simulation software problems and challenges

This chapter is mainly sum up problems and drawbacks I had to challenge during the creation of software.

17. Dependency tree

As the platform simulation time is made in seconds (even milliseconds would have this problem), there comes one problem:

“When all process entities finish in the same step seconds how they will move actor next?”

This problem is not visible (or it’s error is so low on long run that it is irrelevant).

Solution – **Dependency tree**

XXX – popisat hierarchiu dependency tree

As solution looks promising it’s not helping. In following problem we will find why this problem is not an solution to our problem.

18. Root problem of dependency tree

As you can see on the image bellow. We came to the root of dependency tree, but we see that flow can start from two places. Red spawn point spawns entities in 00:03 second interval and the blue one in 00:05. Station afterwards manufactures in 00:10 seconds process time. Problem here is that because of **red** spawner is first in step tree, the will be always first so blue one, will be always blocked.

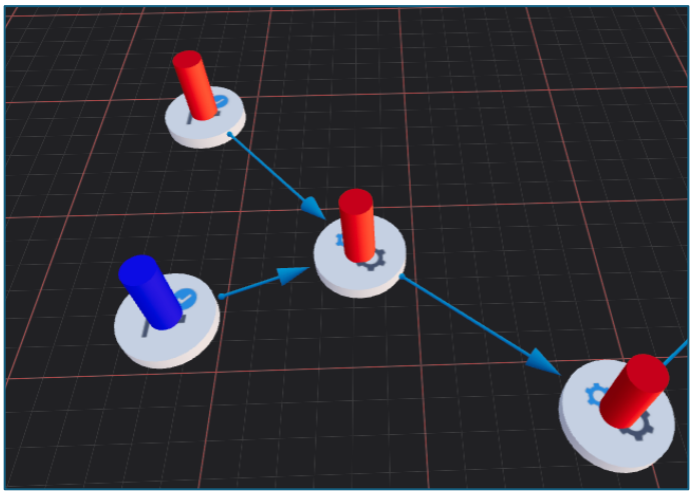


Figure 2 Root problem of dependency tree

The idea I came with is creation of lock mechanism, where blocked entities write their request to order unblocking status and this unblocking status should be primary order rule for entities sorting.

10. Used literature

11. Licenses and project dependencies

All resources used to create this software are free and under MIT license. I would especially thanks to authors of:

19. Frontend

- ThreeJS - incredible 3D rendering platform which enabled creation of this project.
- @react – fiber – thanks gentlemen for binding three JS to react, you created incredible architecture and developer friendly experience. Without it this project would be pain in ass and probably would fail.
- @react-drei – guys, big love to you, without you and your ideas my software would not be so nice and beautiful
- Antd design – thanks for providing this beautiful UI library for free.

20. Backend

- Golang – best thing which was created by google, I fell in love with this language after a few lines of code
- Gin Gonic – amazing web server framework thanks for it creation.