POLITECHNIKA GDAŃSKA

INTELLIGENT LOGISTICS SYSTEMS

ACTIVITY #4

Flow Control – *Wait for Event and Subflow*

dr hab. Daniel Kaszubowski, prof. PG

Department of Transport Engineering

ADTRANS
ADVANCED DATA ANALYSIS FOR TRANSPORTATION AND MOBILITY ACADEMIC TEACHERS

Co-funded by the European Union

Co-funded by the European Union. Views and opinions expressed are however those of the author or authors only and do not necessarily reflect those of the European Union or the Foundation for the Development of the Education System. Neither the European Union nor the entity providing the grant can be held responsible for them.

1. Objective and new skills

The aim of the task is to present additional possibilities of flow control using the *Wait for Event* block (*match* option*)* and *Sub Flow* repetitive processes. The basics of industrial robot control will also be introduced. Model development will be divided into two parts, taking into account the addition of new functions.

New skills
Using the *Wait for Event block – match* option
Introducing the ability to create repeatable activities using *SubFlow*
Basic functions of controlling an industrial robot

2. Assumptions and input data

The model will simulate the process of unloading pallets on a conveyor using an industrial robot:

1. Pallets for unloading appear on average every 60 seconds on the shared *Pallets (Palety)* buffer. The number of products on a pallet varies from 2 to 8 pieces.
2. The pallets are placed on a conveyor where they are transported to the unloading point by the robot – decision point *DP1* .
3. Empty pallets are released from the unloading station and removed from the system *(Puste Palety).*
4. The system also features a second flow element – empty boxes that arrive on the conveyor from a separate *Tote (Skrzynki)* source and go to the sink *Zrzut Skrzynki*. The time between their arrival is 2 minutes.

The material was based on: K. Jurczyk, FlexSim , user manual.
InterMarium Sp. z o . o. , Krakow 2022

The layout of the 3D model and the *Data* table are shown in Figure 1.

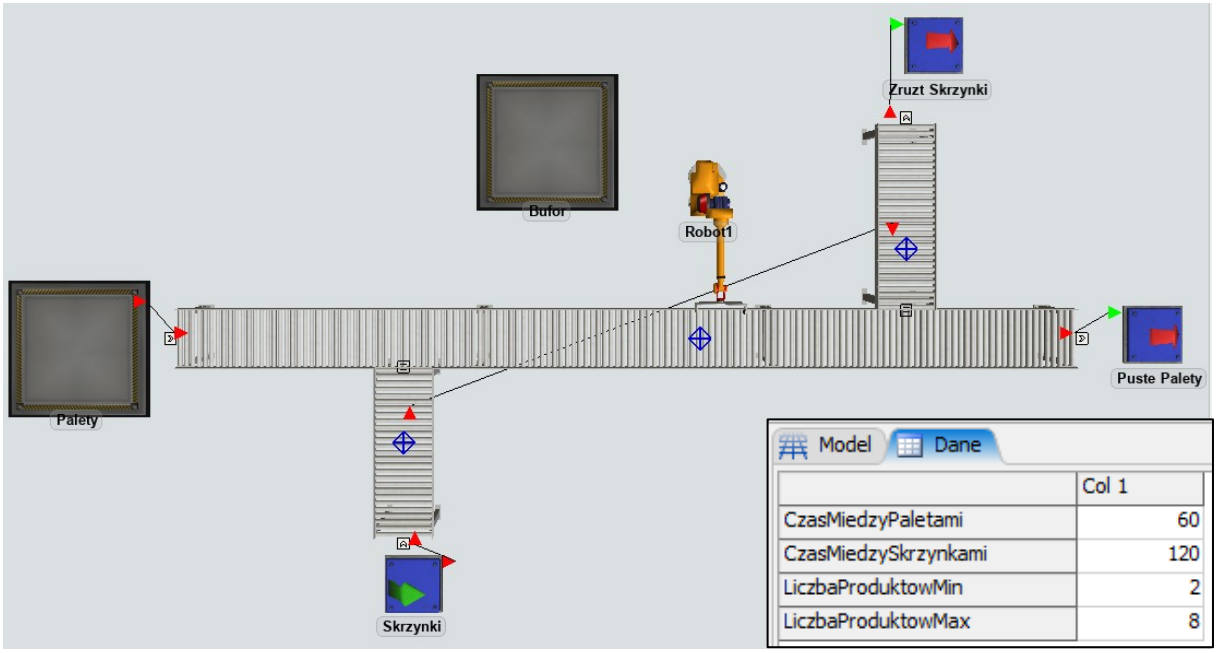


Fig. 1 3D model and *data table*

Tote (Skrzynki) source will generate flow elements *(tote)* based on the table data. The decision points on the shorter conveyors *(DP2* and *DP3)* are connected directionally. The flow elements are sent along the route defined by these points using the *On Arrival -> Send Item* trigger (Fig. 2).

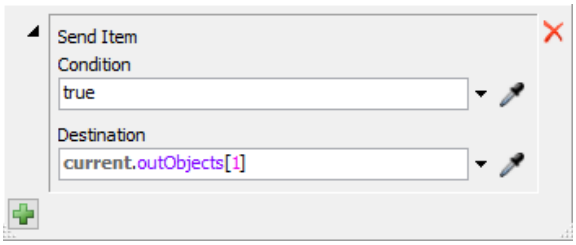


Fig. 2 *Send Item*

Figures 3 and 4 show part 1 and part 2 of the *Process Flow* creation.

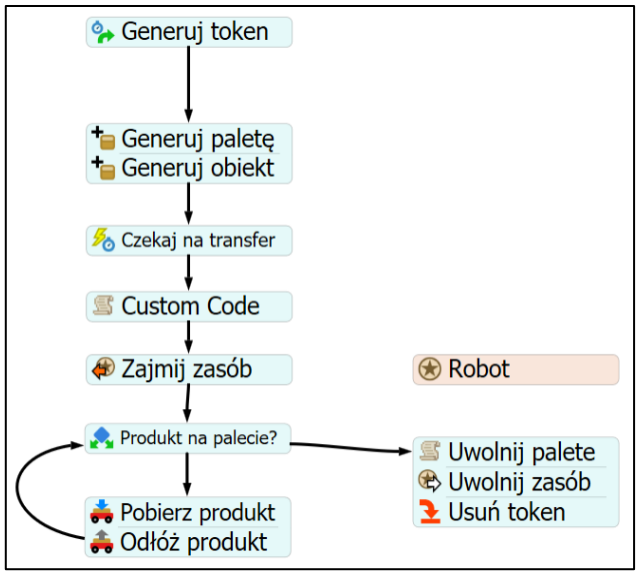


Fig. 3 *Process Flow* – Part 1

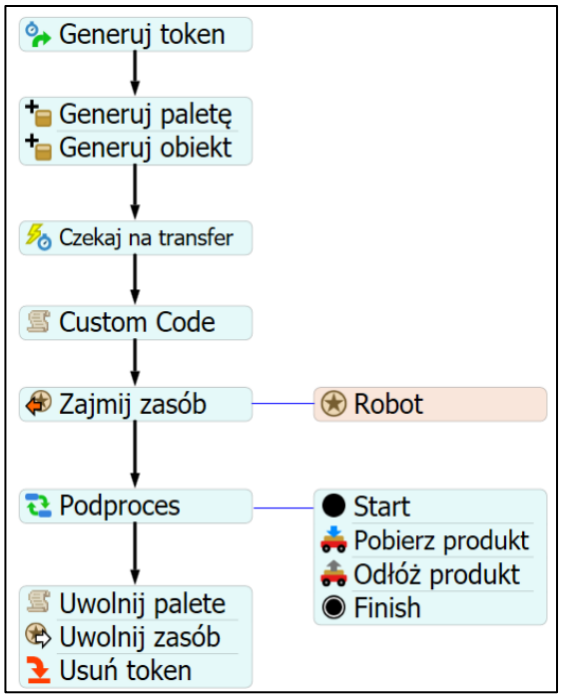


Fig. 4 *Process Flow* – Part 2

3. Model control logic

3.1. Part 1

3.1.1. Generating tokens

Tokens will be generated from the *Process Flow* level in the *Inter-Arrival Source* block on average every 60 seconds (fig. 5).

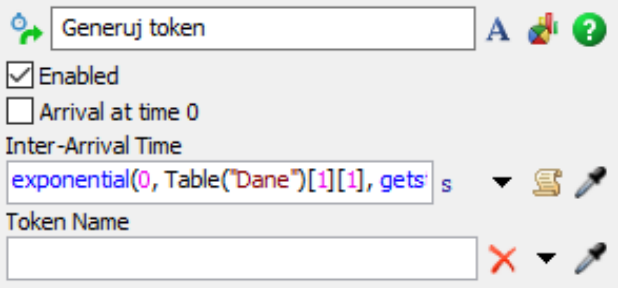


Fig. 5 *Inter-Arrival Source*

3.1.2. Generating pallets

The next task is to generate flow elements – pallets and products. Pallets are added in the *Create Object* block (Fig.6). Each token generates 1 pallet *(Quantity)* in the *Paletts (Palety)* buffer *(Create In)*. A reference to the created pallet is saved under the **pallet** label *(Assign To)*.

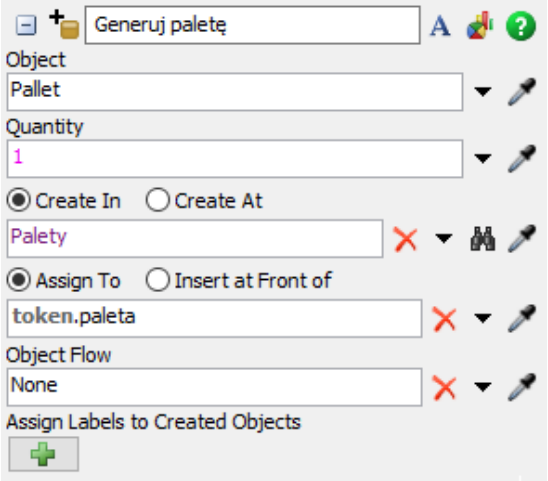


Fig. 6 *Create Object* – pallets

3.1.3. Generating products

Products that will be on the pallet are also generated in the *Create Object* block (Fig. 7). The number of products created will be determined randomly with a uniform distribution with parameters taken from the *Data* table. Generated products appear on the previously created pallet *(Generate In)*. In this case, no reference is created on the label to the created products *(Assign To – None)*.

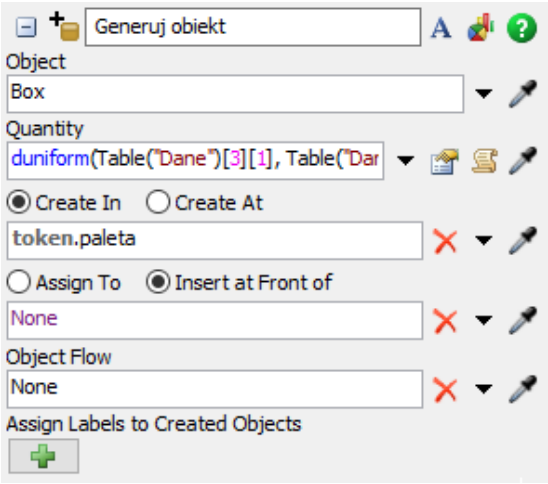


Fig. 7 *Generate Object* - products

3.1.4. Pallet stop at service station

The generated pallet is automatically transferred to the conveyor. It should be stopped at the service station *(DP1)*, where it will be unloaded by the robot and then the empty pallets will be released.

Wait for Event block (Fig. 8) will be used , which will represent the moment when the palette appears at the decision point. The block allows you to wait for a specific simulation event to occur in the indicated model object (here – *DP1)* and start a specific action at that moment. In this case, it is the appearance of a flow element *(On Arrival)* and the *Custom command Code* (Fig. 9). It allows you to select object control options *(Control)*, indicate the *Conveyor* group and specify the action *(Stop Item)*.

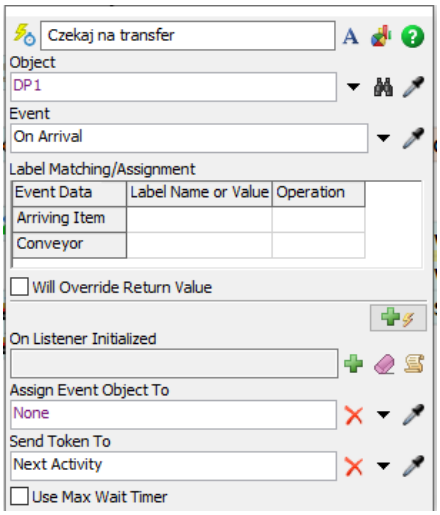


Fig. 8 *Wait for transfer*

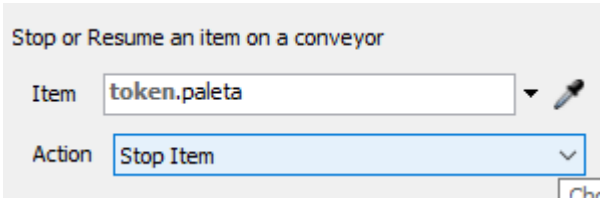


Fig. 9 *Custom Code*

After running the simulation after some time, you can see that the pallet stops in the wrong place (before *DP1*), and its stop was caused by the box that passed the decision point *DP1* (Fig. 10). This can also be caused by another pallet that, upon entering *DP1*, causes the other pallets on the conveyor to stop.

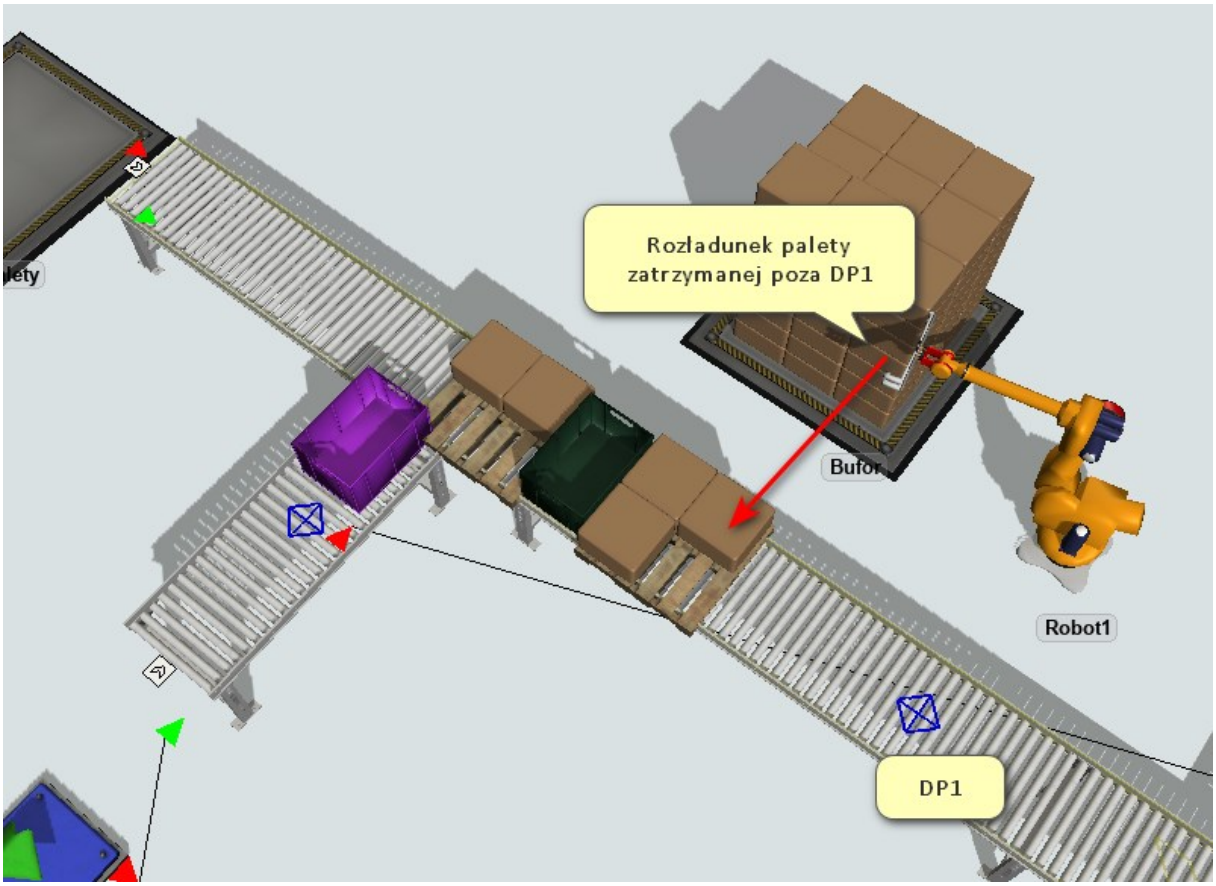


Fig. 10. Incorrect pallet stop outside DP1

This issue was introduced to discuss the operation of the *Match* function. in the *Wait for Event* block. Its initial settings in the *Label Matching* field/*Assignment* did not contain a reference to any specific flow element. As a

result, each of them, regardless of whether it was a pallet or a box, caused the pallet to stop according to the command shown in Figure 9.

In order for the model to work as intended, the changes shown in Fig. 11 must be made in *Wait for transfer*. Since each pallet has a label assigned to it, after exiting *Generate Palette,* setting the *Match* option for it will trigger the *Custom Code* specified commands only by the element with the matching label.

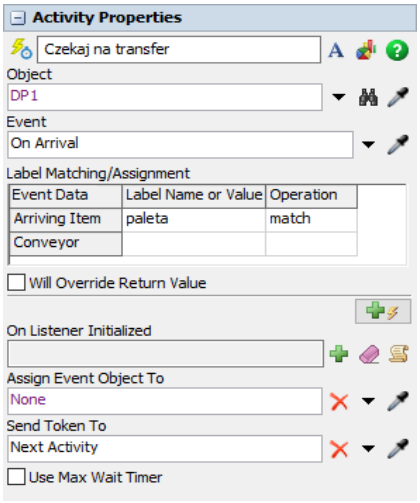


Fig. 11. Modification of the *Wait for transfer* settings

3.1.5. Robot

Robot is used to unload the pallet at point *DP1*, which is controlled by a standard pair of *Acquire Resource* and *Resource* blocks (Fig.12). The reference to the robot will be stored under the robot label in the *Assign To Label* field .

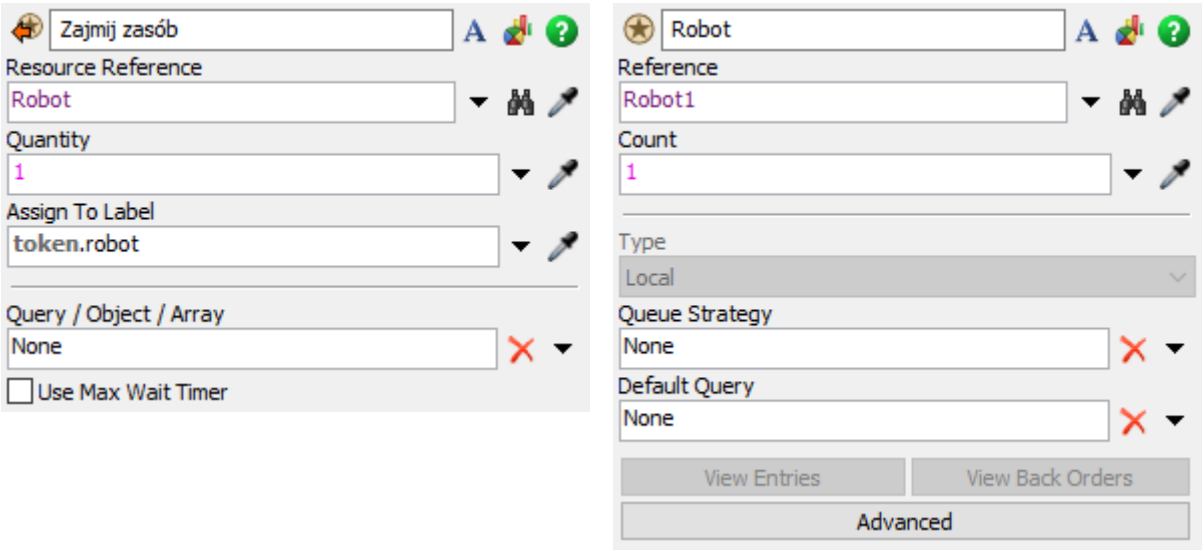
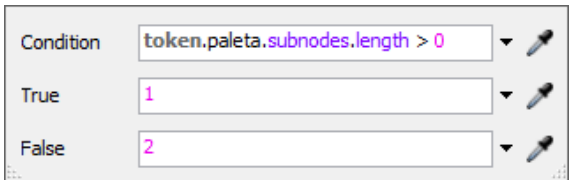


Fig. 12. Acquiring the *Robot* resource

3.1.6. Unloading the pallet

After stopping at the *DP1* service station, a repeatable sequence of picking products and putting them in the buffer is performed. The actions are repeated as long as there are products available on the pallet. Then the pallet is released and removed from the system by a conveyor *(→ Empty Pallets)*.

Decide block is responsible for checking the contents of the pallet (Fig. 13 on the right). The *Conditional Decide* option should be selected *(Send Token To)* and enter a condition that will be checked every time.



This is the expression **token.palette.subnodes.length** > 0. If the number of products > 0, the robot puts them on the buffer (connector 1), otherwise the pallet release sequence is initiated (connector 2). After being put into the buffer, the sequence is repeated (feedback to *Decide)*. The diagram of how this option works is shown in Fig.14.

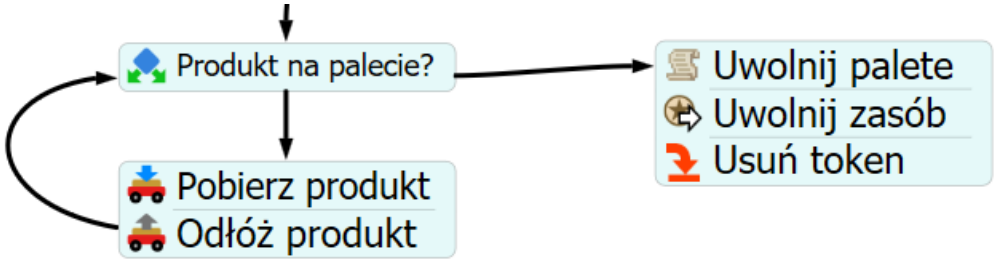


Fig. 14. *Conditional Decide* option operation

Next, you need to enter the settings for the robot control blocks. The *Load* and *Unload* blocks pair is responsible for this (Fig. 15). In *Get product (Pobierz produkt)* the command **token**.pallet.last is entered, which means that the robot takes the product that was last placed on the pallet. In *Put away product (Odłóż produkt)*the parent node for the product is the robot, therefore when indicating the item to be put away it should be included in the command **token**.robot.first (or last, because there is only one product).

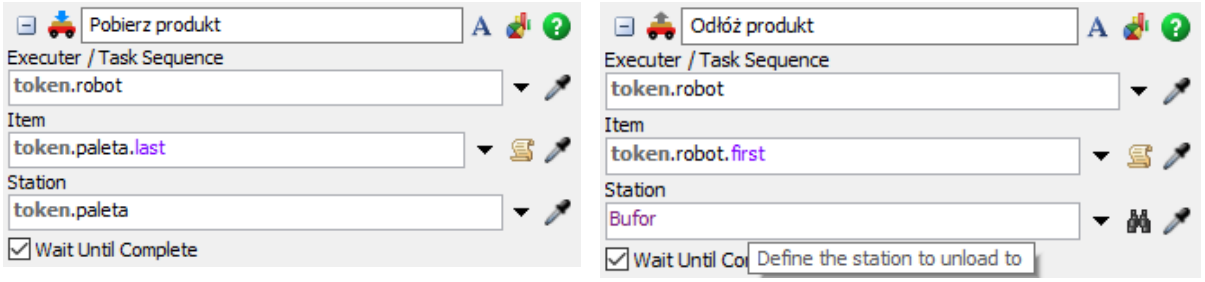


Fig. 15 Robot control

3.1.7. Pallet and robot release

Once the pallet is completely empty, it is released. As before, the *Custom Code* block will be used for this purpose. (Fig. 16), this time resuming *(Resuem Item)* moving the pallet through the conveyor.

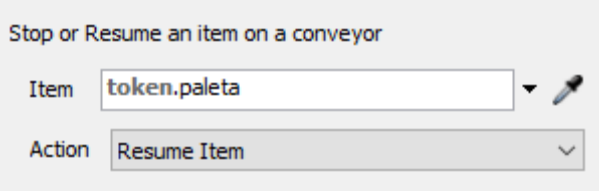


Fig. 16 Pallet release – *Cuusotm Code*

The last action is to release the robot (Fig. 17).

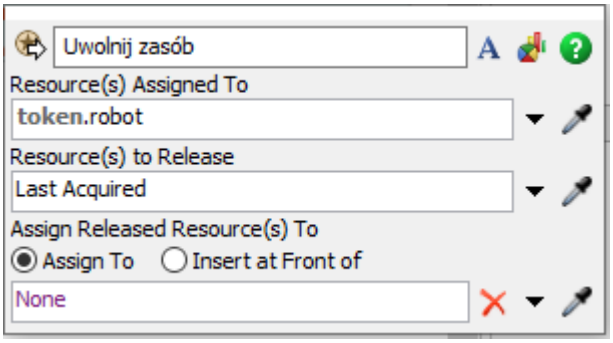
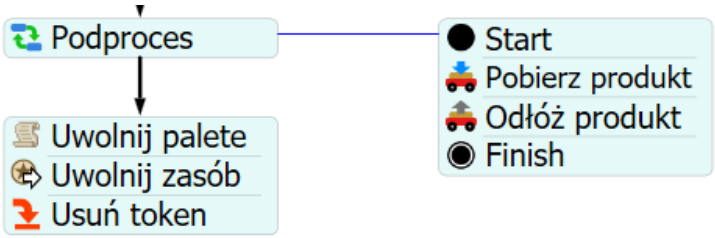


Fig. 17. Resource release

4. Part II – Model Modification

The current model will be modified by replacing the decision part with feedback (unloading the pallet) with a set of repeatable operations grouped into *Sub Flow* . The operations of picking up and putting away products repeated an appropriate number of times are grouped between the *Start* and *Finish* blocks , while the *Run (Start) Sub Flow* block is responsible for starting the subprocess and the number of its repetitions. The modified part of the model is shown in Fig. 18 (right).



In *Run Sub Flow (Podproces),* the following settings must be entered (Fig. 19):

1. *Destination* – reference to the block starting the procedure, here *Start* .
2. *Quantity* – number of repetitions of the performed procedure. In this case it will be equal to the number of products on the pallet, i.e. **token**.pallet.subnodes.length.

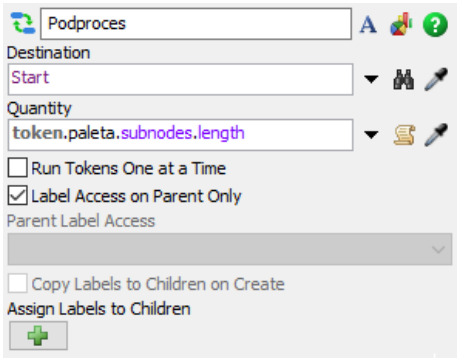
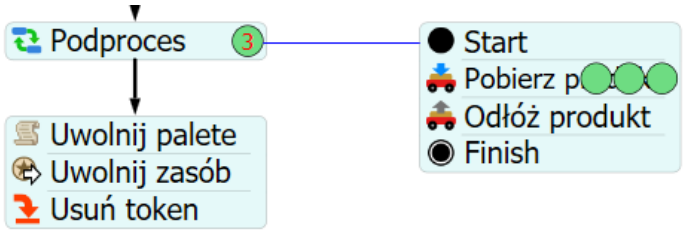


Fig. 19 *Run Sub Block Flow*

After starting the simulation, a problem occurs because the robot took only one product from the pallet and put it on the buffer, while the pallet was released without unloading the remaining products. This situation is illustrated in Fig. 20 on the right, where three child tokens can be seen *(child tokens)*, the number of which corresponds to the given number of subprocess repetitions. The resulting situation can be interpreted as an attempt to pick the same product from the pallet four times.



However, each token should be responsible for downloading one product, so they should be sent to the subprocess one by one. In this case, you should use *Run Sub Flow,* select the *Run Tokens One at a Time* option, which will result in the tokens entering the subprocess one at a time.