R6 = If the crane is positioned above the deposit belt, it may only move towards the container.

Appendix A Production Cell Case Study

A.1 System-level diagrams

A.1.1 Context diagram

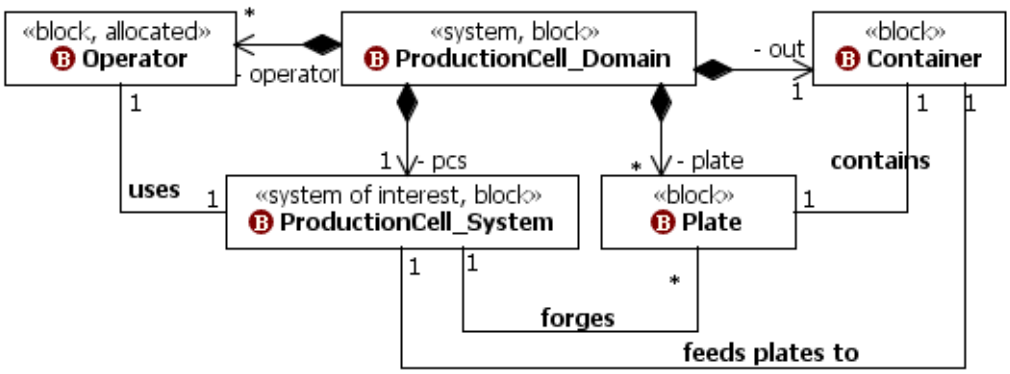


Figure 1 - System context diagram which shows the elements in the environment that may interact with the system.

A.1.2 High-level requirement Diagrams

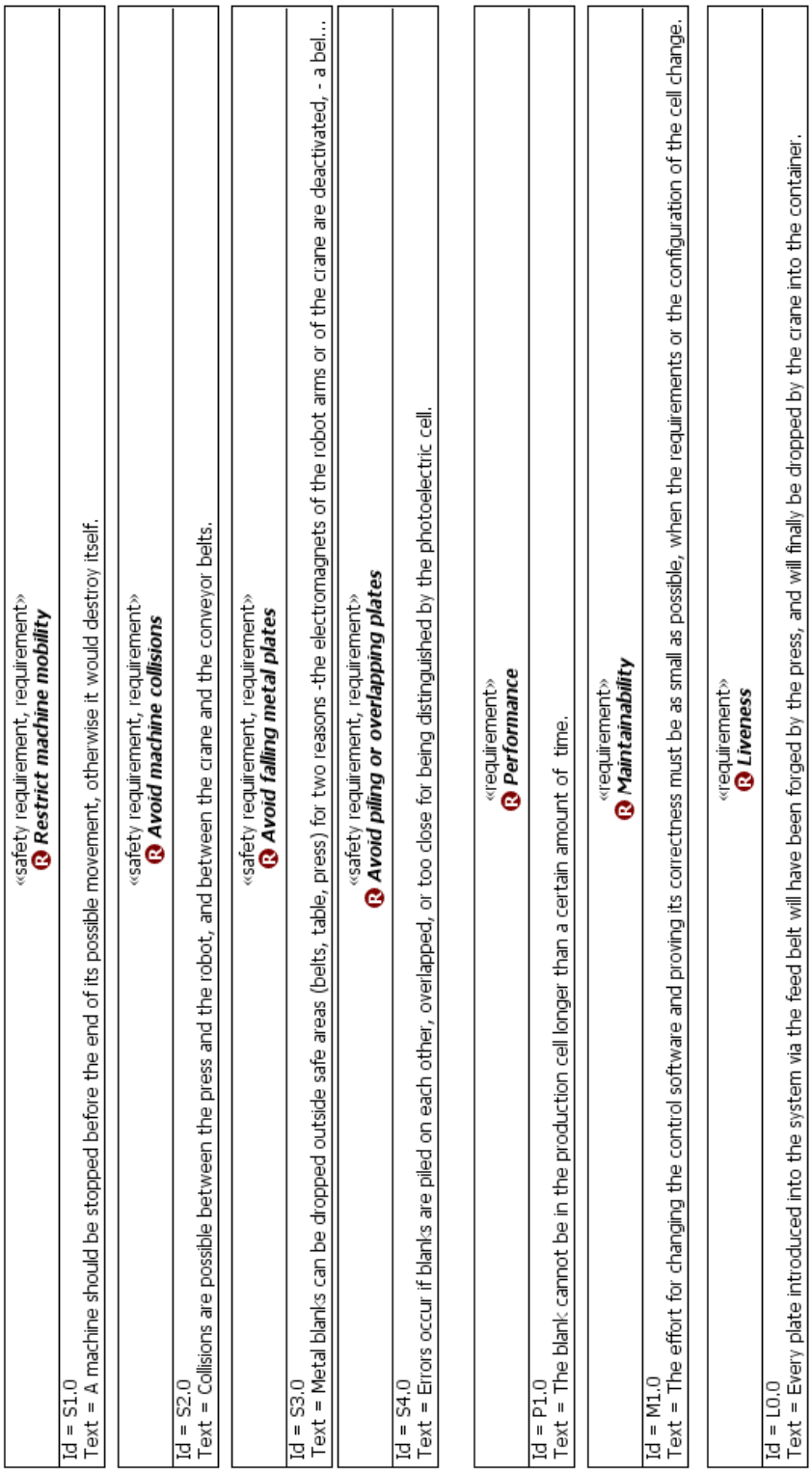


Figure 2 - System level requirements, high-level requirements which all will be further partitioned and derived.

A.1.3 Use case diagram

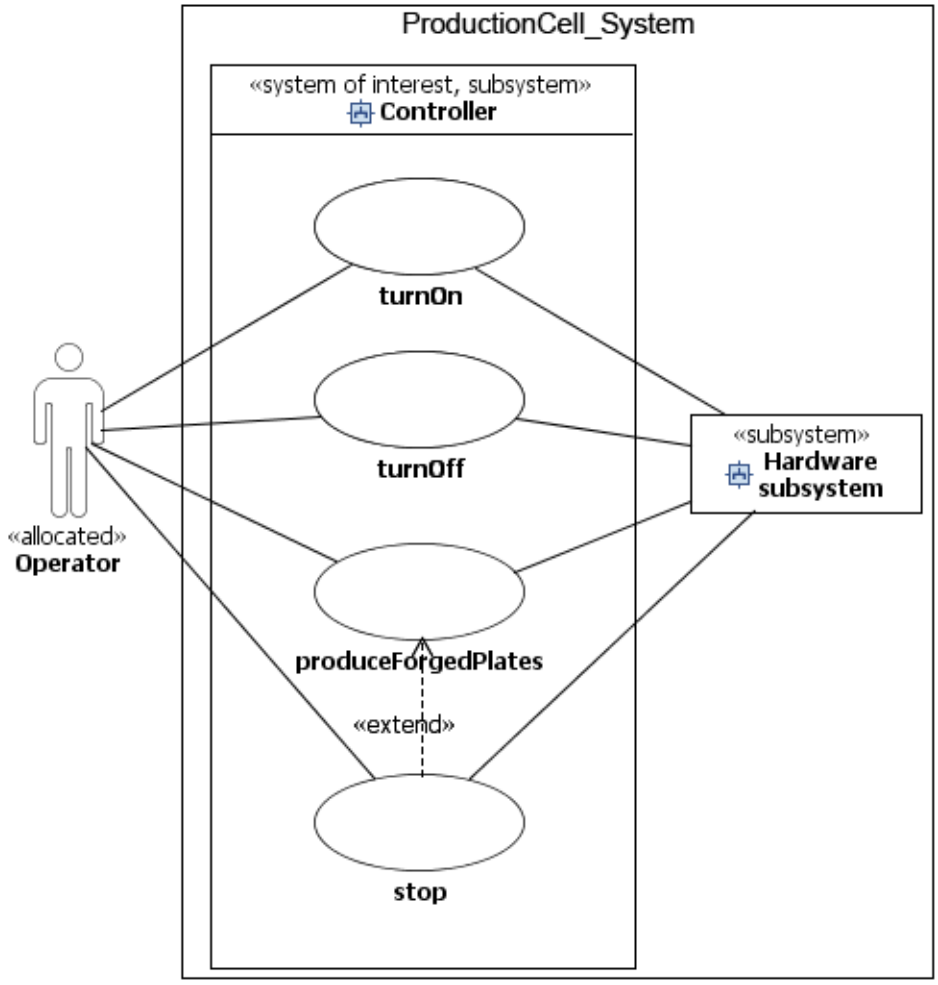


Figure 3 - Use Case Diagram for the Production Cell System, describing the system's main functions and actors

The Hardware subsystem represents the set of hardware devices which will provide input to and receive the output from the controller.

A.1.4 Use case specification for 'Turn On'

Use Case	*Turn on*
Actor	▪ Operator, FeedBelt, Table, Robot, Press, DepositBelt, Crane
Pre-condition	▪ System is turned off ▪ No plates are present in the system
Description	1. The operator turns on the system. 2. *The system* tells the machines in the system to turn on and go to their initial positions. a. *The system* commands ***the feed belt*** to turn off its motor. b. ***The feed belt*** turns off its motor. c. *The system* commands ***the table*** to move into its load position. d. *The table* lowers and rotates until it reaches its load position. e. *The system* stores a request that ***the table*** is ready for a new plate until ***the feed belt*** is ready to act on it. f. *The system* commands ***the robot*** to move to the table pick up position. g. ***The robot*** retracts both its arms and rotates until it reaches the table pick up position (position 2). h. *The system* commands ***the press*** to move to its load position. i. ***The press*** moves its movable plate upwards until in the top position, and then moves its movable plate downwards until it is in middle (load) position. j. *The system* stores a request that ***the press*** is ready to receive a plate until ***the robot*** is ready to act on it. k. *The system* stores a request that ***the deposit belt*** is ready to receive a plate until ***the robot*** is ready to act on it. l. *The system* commands ***the crane*** to move to its pickup position over the deposit belt. m. ***The crane*** *moves* towards ***the deposit belt*** and extends its arm until it is in the pickup position over the deposit belt. n. *The system* stores a request for ***the deposit belt*** to bring a plate to the end of the belt. The system stores this request until ***the deposit belt*** is ready to act on it.
Post-condition	▪ All the physical machines in the system are turned on and in their initial position and are ready to receive a plate. ▪ FeedBelt's belt is not running. ▪ Table is in load position. ▪ Robot is in position 2, arms are retracted. ▪ Press is in load position. ▪ Crane is positioned over the deposit belt, arm extended to the deposit belt extension.

	▪ No plates are in the system.

A.1.5 Use case description for 'Produce forged plates'

Use Case	***Produce forged plates***
Actor	▪ Operator ▪ FeedBelt_HW ▪ Table_HW ▪ Robot_HW ▪ Press_HW ▪ DepositBelt_HW ▪ Crane_HW
Pre-condition	▪ System and all machines are turned on ▪ FeedBelt is not running. ▪ There is no plate at the end of the feed belt.
Description	*1. The operator adds a new plate to the feed belt.* *2. The system commands the feed belt to move the plate to the end of the belt.* *3. The feed belt moves the plate to the end of the belt.* *4. When the table is in load position, the system commands the feed belt to feed the plate onto the table.* *5. The feed table feeds the plate onto the table.* *6. The system commands the table to go to its unload position.* *7. The table rotates and elevates to its unload position.* *8. When the robot's upper arm is unloaded and the robot is ready, the system commands the robot to pick up the plate from the table.* *9. The robot rotates until its upper arm points to the table, and then it extends its upper arm until it is positioned over the table, and activates its magnet to pick up the plate, before it retracts its upper arm to its retracted position.* *10. When the plate is picked from the table, the system commands the table to go to its load position.* *11. The table rotates and lowers itself to its load position.* *12. When the press is in load position and the robot is ready, the system commands the robot to load the plate to the press.* *13. The robot rotates until its upper arm points to the press, then it extends its upper arm until its magnet is positioned over the press, and deactivates its magnet to drop the plate, before it retracts its upper arm until it is in its retracted position.* *14. The system commands the press to forge the plate and to bring the movable plate to its unload (bottom) position.* *15. The press moves its movable plate upwards until it arrives to the top*

	position, where the plate is forged, then it moves to its bottom (unload) position. 16. *When the robot's lower arm is unloaded and the robot is ready, the system commands the robot to pick up the plate from the press.* 17. *The robot rotates until its lower arm points to the press, and then it extends its lower arm until its magnet is positioned over the press and activates its magnet to pick up the plate and then retracts its lower arm.* 18. *When the plate is picked up from the press, the system commands the press to go to its load position.* 19. *The press moves its movable plate to its load (middle) position.* 20. *When the deposit belt is ready to receive a new plate and the robot's lower arm is loaded and the robot is ready, the system commands the robot to drop the plate on the deposit belt.* 21. *The robot rotates so that its lower arm points to the deposit belt, then it extend its lower arm until the magnet is over the deposit belt. There it deactivates its magnet to drop the plate onto the deposit belt and retracts the lower arm to its retracted position.* 22. *When there is no plate at the end of the deposit belt, the system commands the deposit belt to move a new plate to the end of the belt.* 23. *The deposit belt brings the plate to the end of the belt.* 24. *When there is a new plate at the end, the system commands the crane to pick up the plate and bring the plate to the container.* 25. *When the crane is in its initial position over the deposit belt, it activates its magnet so that it picks up the plate and moves horizontally towards the container at the same time as it retracts its arm until it reaches its container position. It deactivates its magnet, so that the plate is dropped into the container and moves back to its initial position.*
Post-conditions	▪ The plate that was added is forged and placed into the container ▪ Other plates may be anywhere in the system

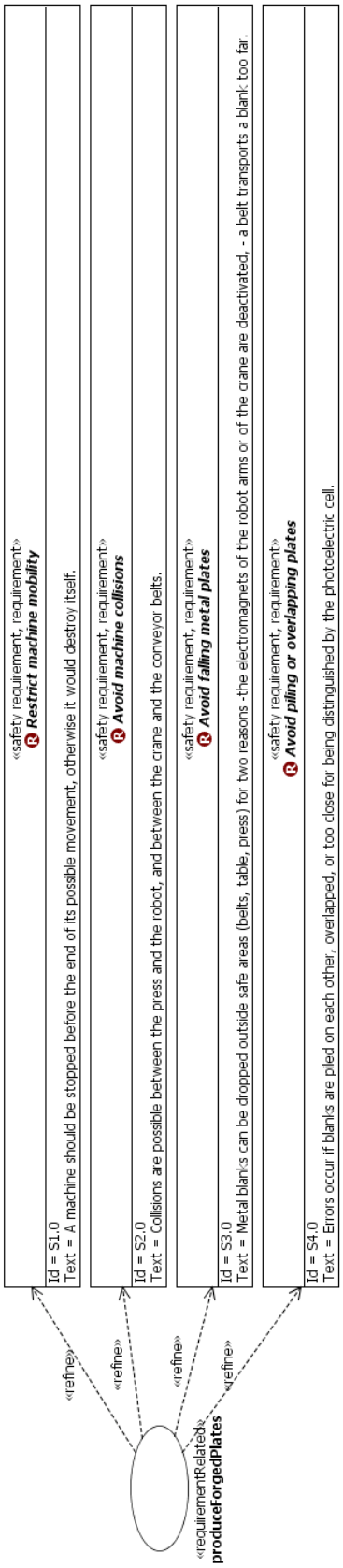


Figure 4 - System-level requirements refine the use case produceForgedPlates

A.2 Structure diagrams

A.2.1 Structure diagram with top-level blocks

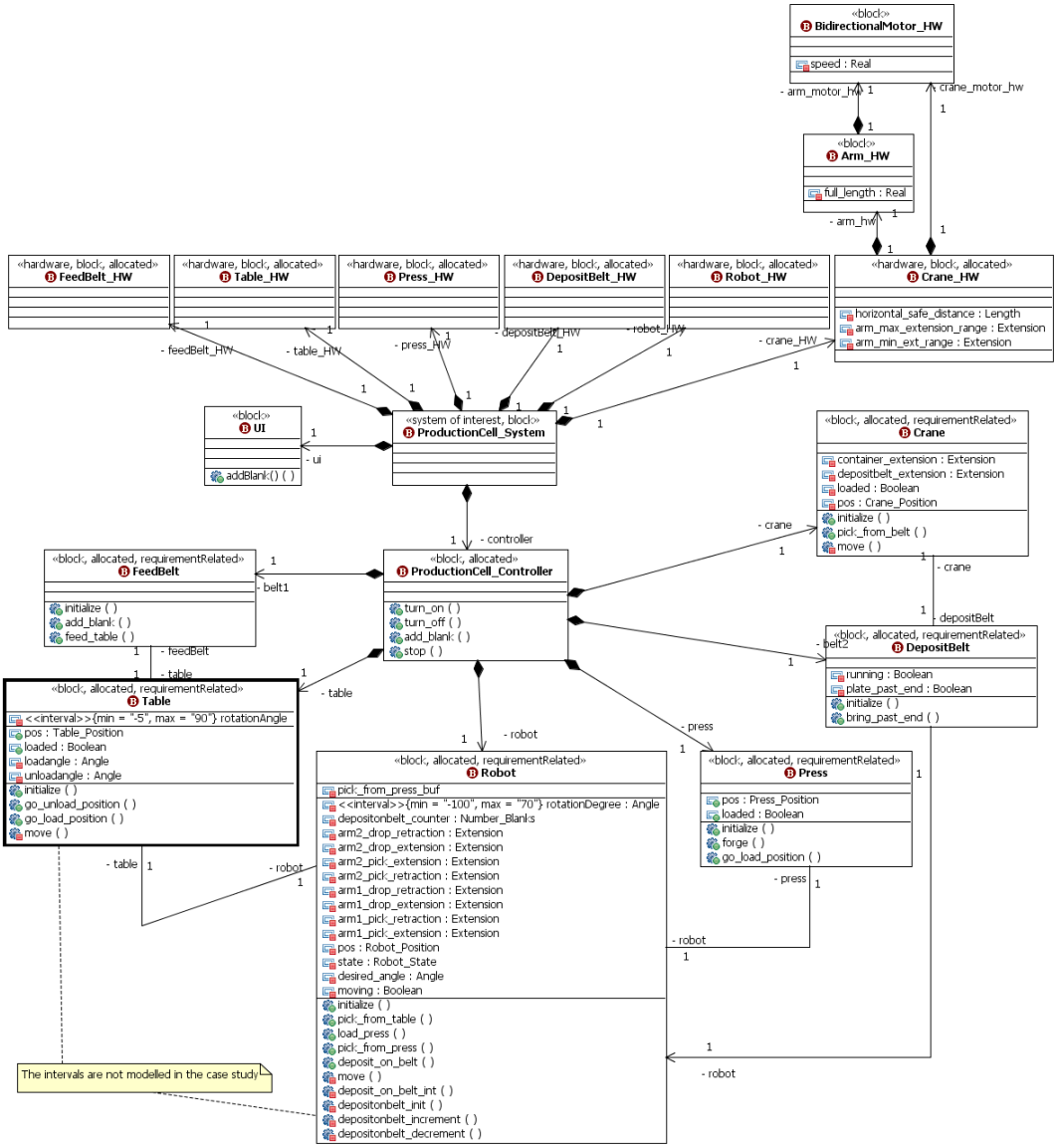


Figure 5 - bdd [Package] ProductionCell_Structure [top-level blocks]. For simplified diagram, see Figure 7.

A.2.2 Full structure diagram

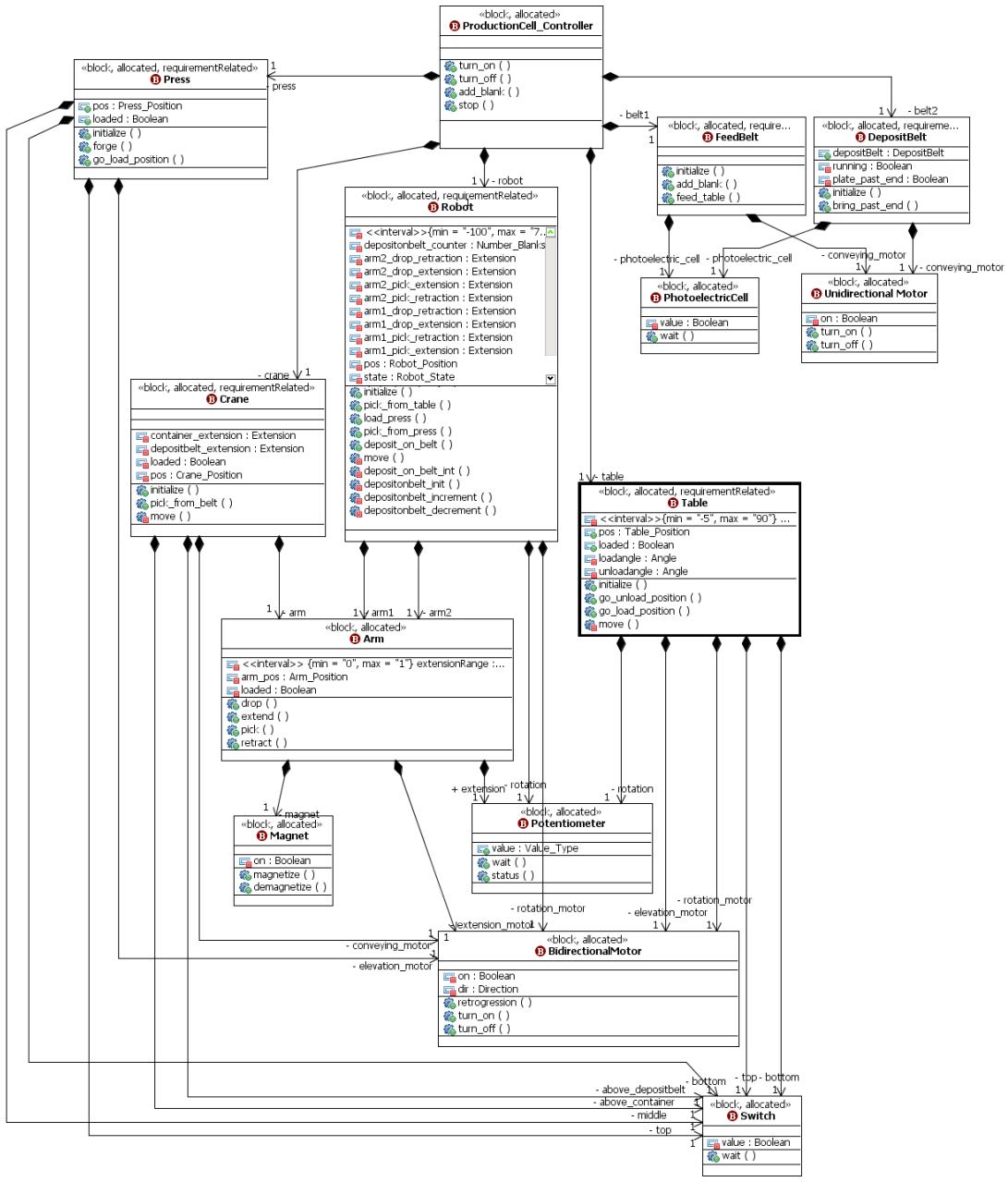


Figure 6 - bdd [Package] ProductionCell_Structure [ProductionCell_Controller and parts]. Full structure diagram of ProductionCell_Controller and its parts (without the superclasses Actuator/Sensor to avoid more cluttered diagram).

A.2.3 Simplified structure diagrams

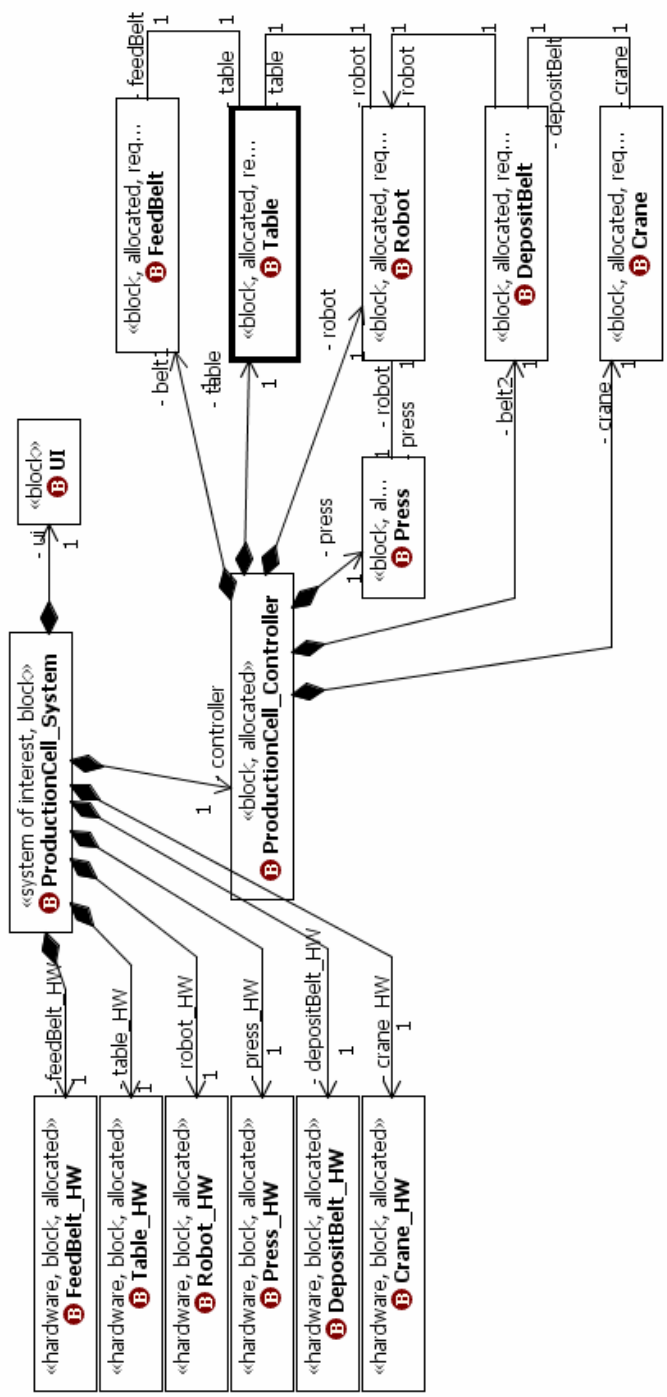


Figure 7 - Simplified structure diagram over the top-level blocks. Full diagram in Figure 5.

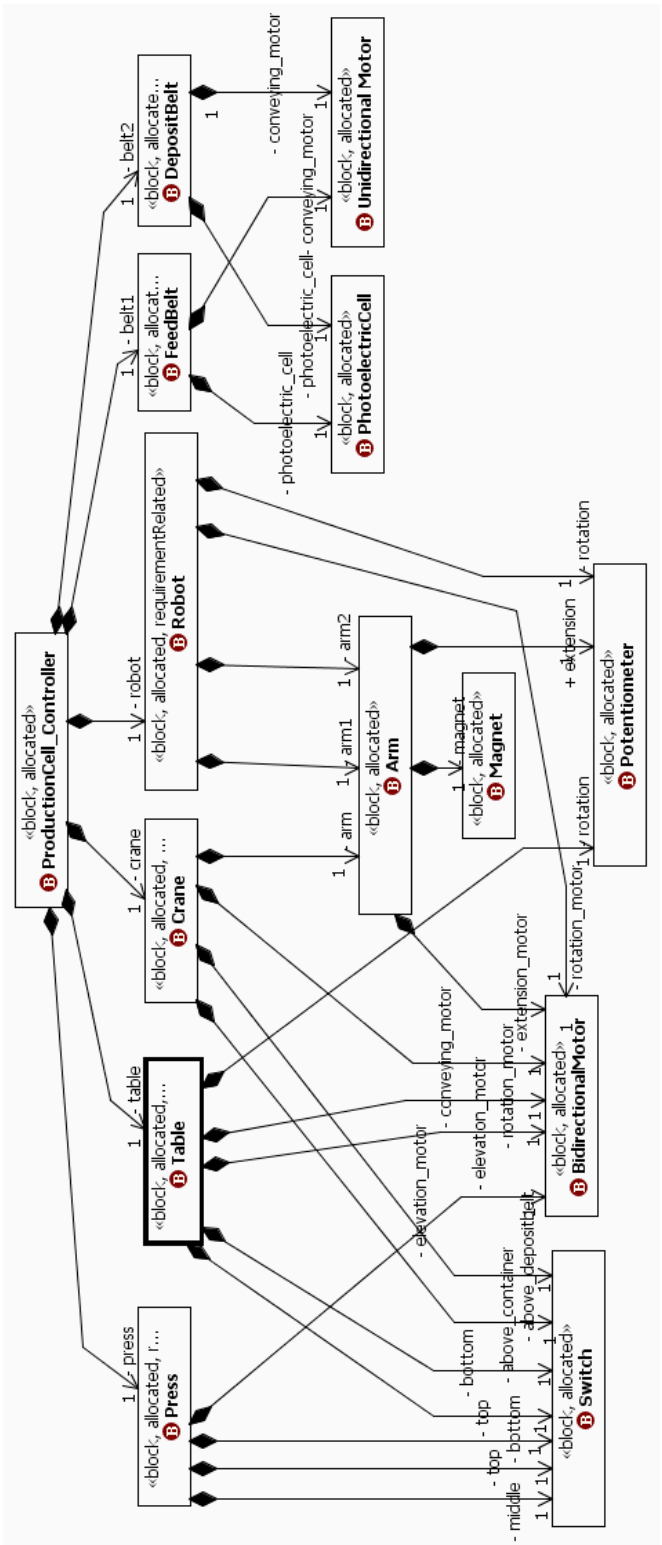


Figure 8 - Simplified stucture diagram over the system of interest and lower level blocks. Full diagram in Figure 6.

A.2.4 Internal Block diagrams

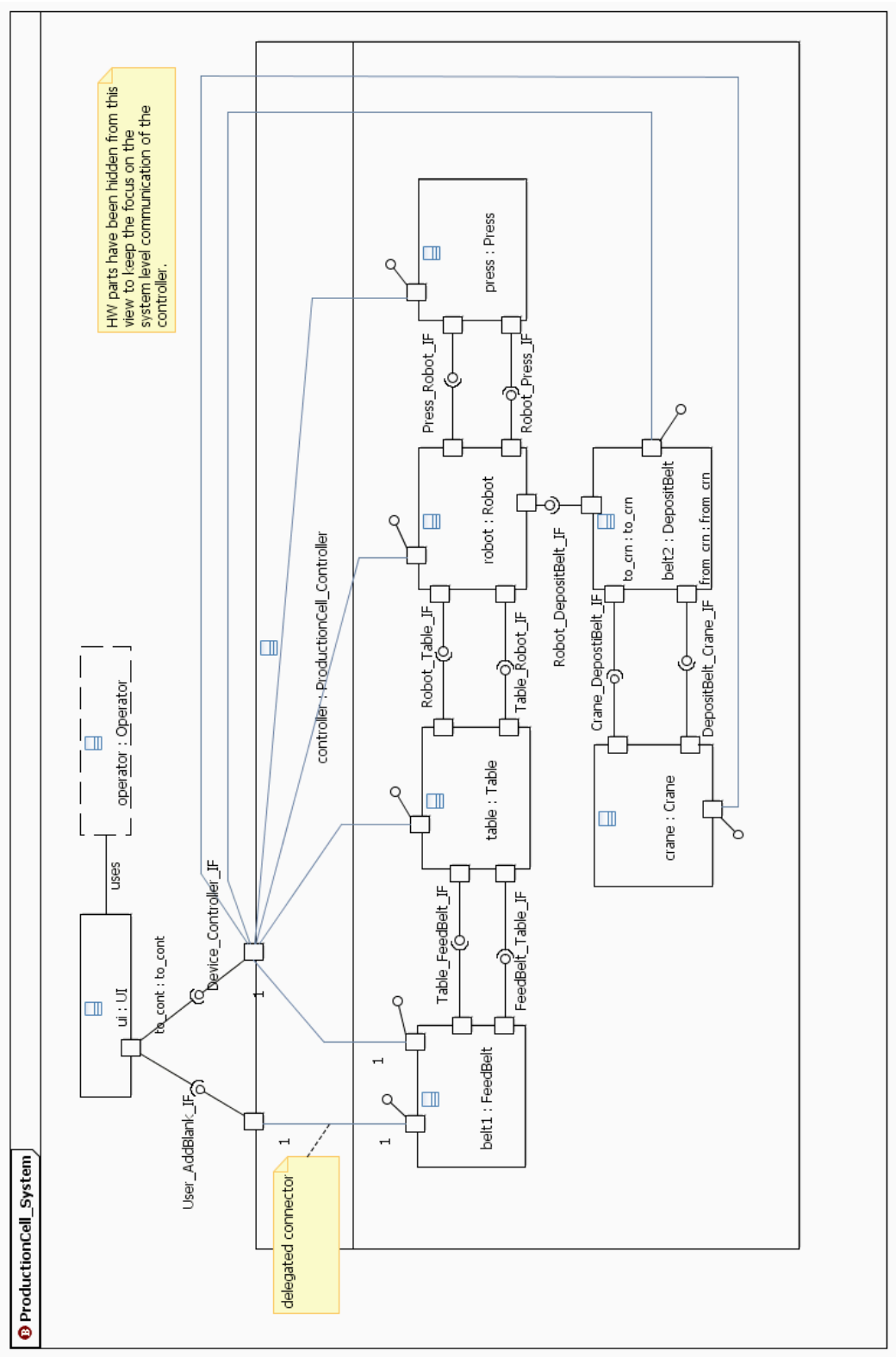


Figure 9 – ibd [Block] ProductionCell_System [Ports and interfaces between top-level blocks]. Interfaces shown in Figure 14.

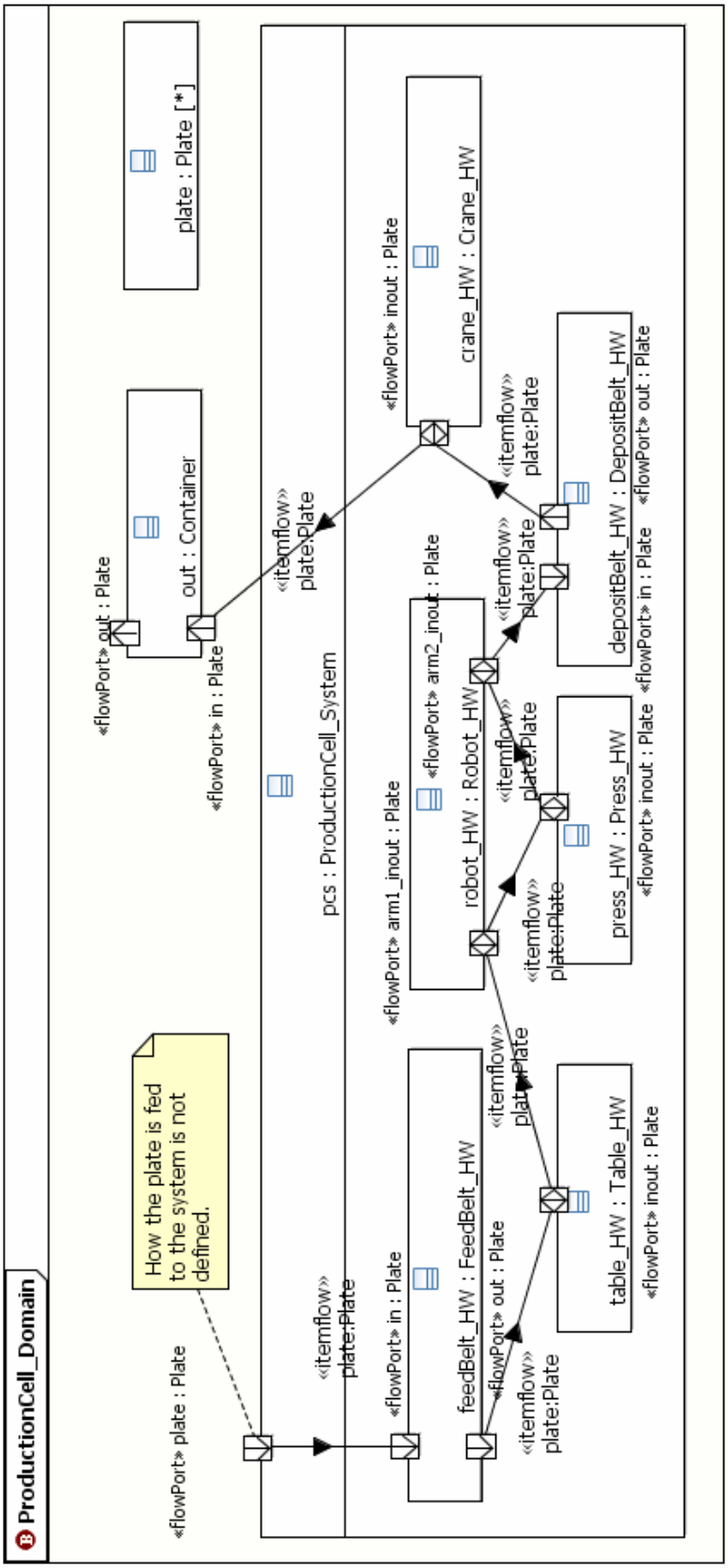


Figure 10 – ibd [Block] ProductionCell_Domain [Item Flow - Plate]

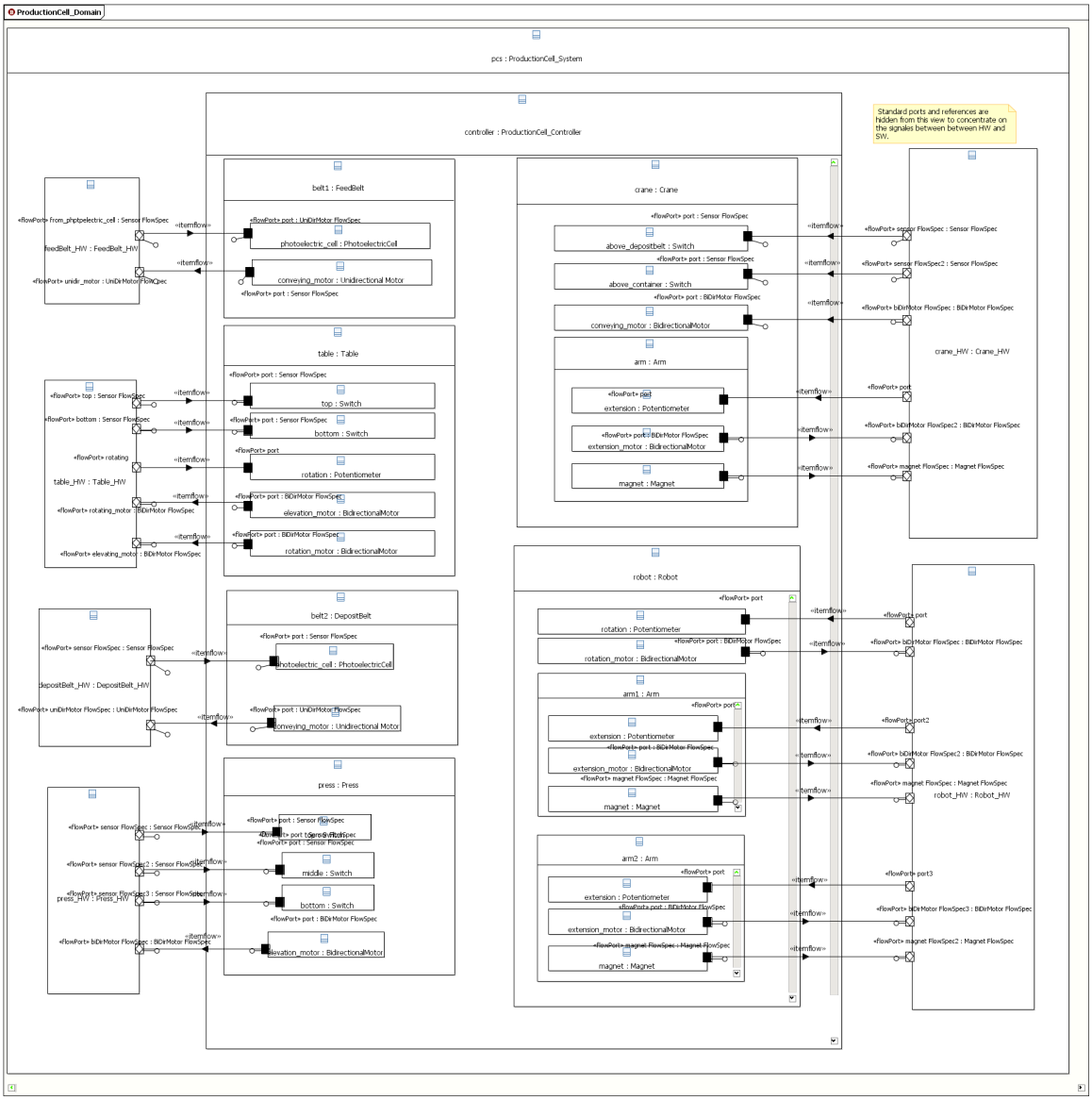


Figure 11 - ibd [Block] ProductionCell_Domain [HW/SW Communiaction], see close ups in Figure 12 and Figure 13.

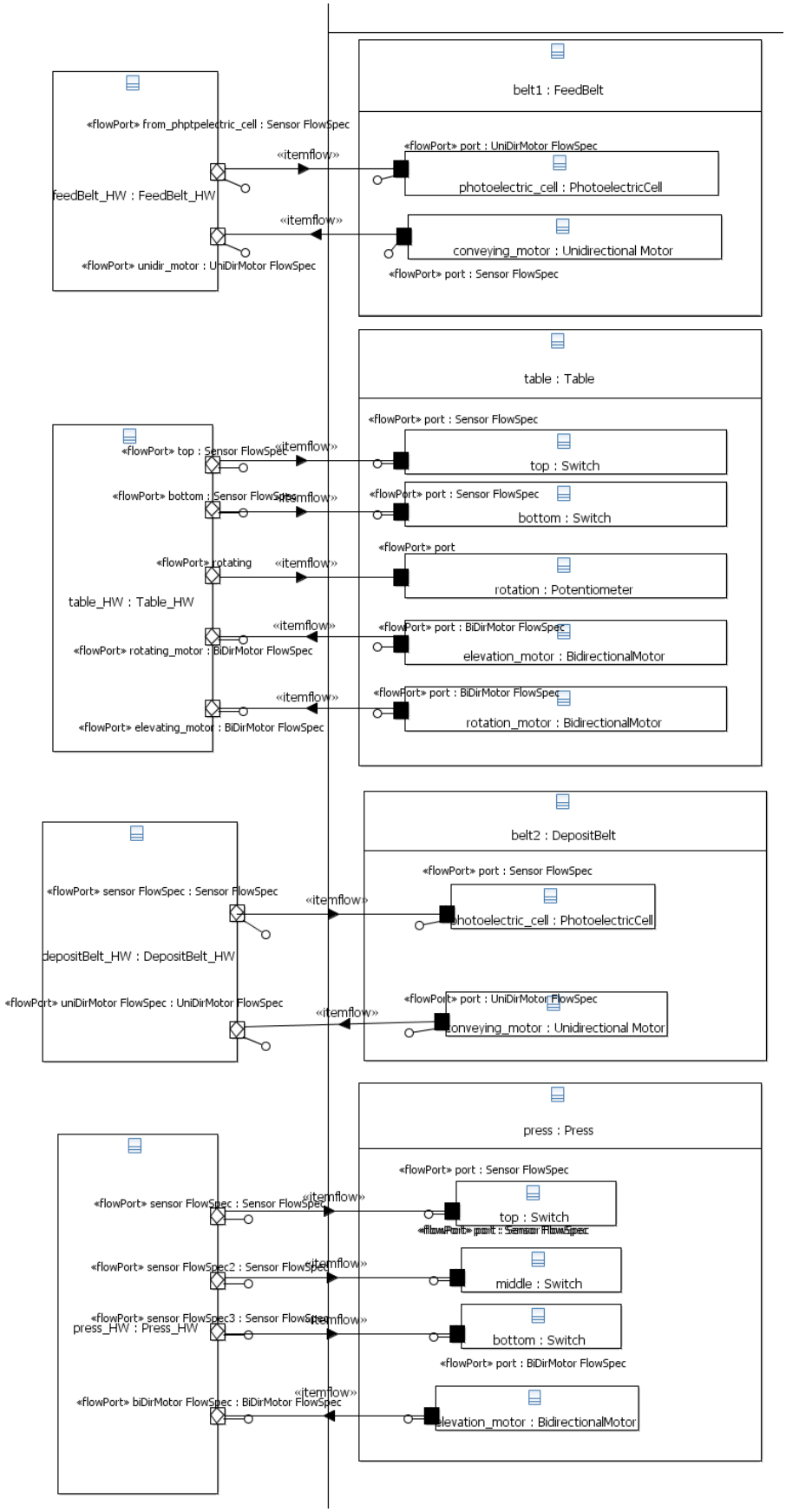


Figure 12 - Part of the ibd [Block] ProductionCell_Domain [HW/SW Communiaction]. Whole diagram in Figure 11.

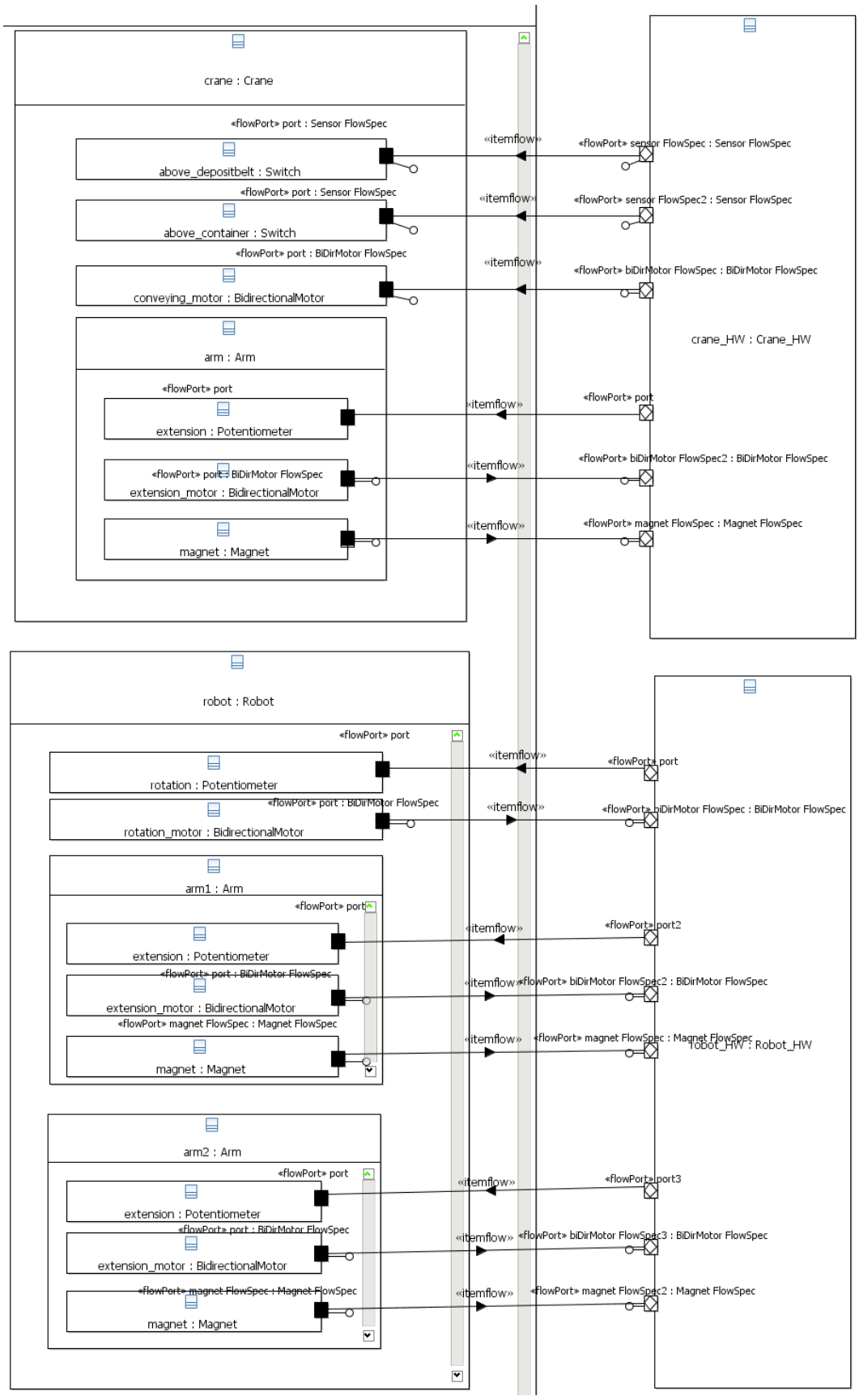


Figure 13 - Part of ibd [Block] ProductionCell_Domain [HW/SW Communiaction]. Whole diagram in Figure 11.

A.2.5 Interfaces for top-level blocks

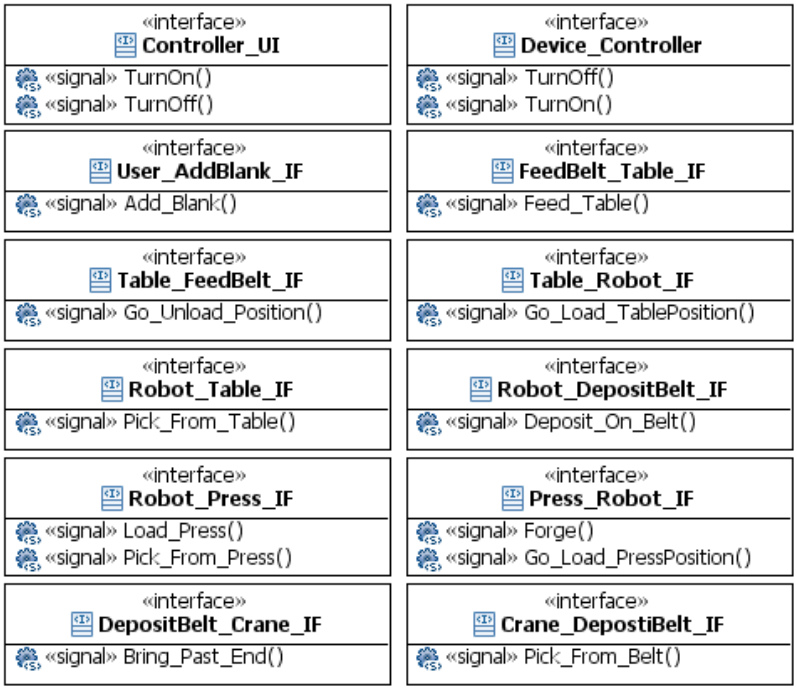


Figure 14 - Interfaces used on the standard ports of the top-level blocks. Show signal receptions of the signals in Figure 15.

A.2.6 Signals for top-level blocks

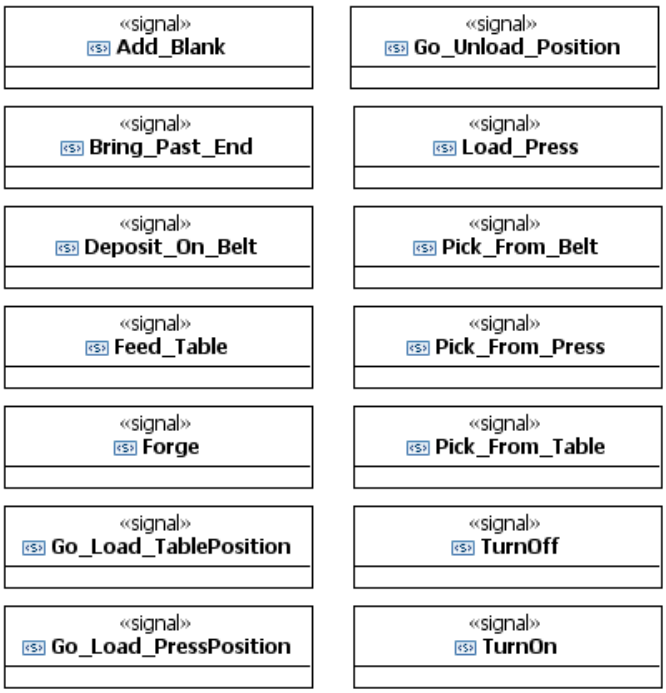


Figure 15 - High Level Signals, which are shown as receptions in the interfaces in Figure 14.

A.3 Derived block-level requirement diagrams

A.3.1 Req S1.0

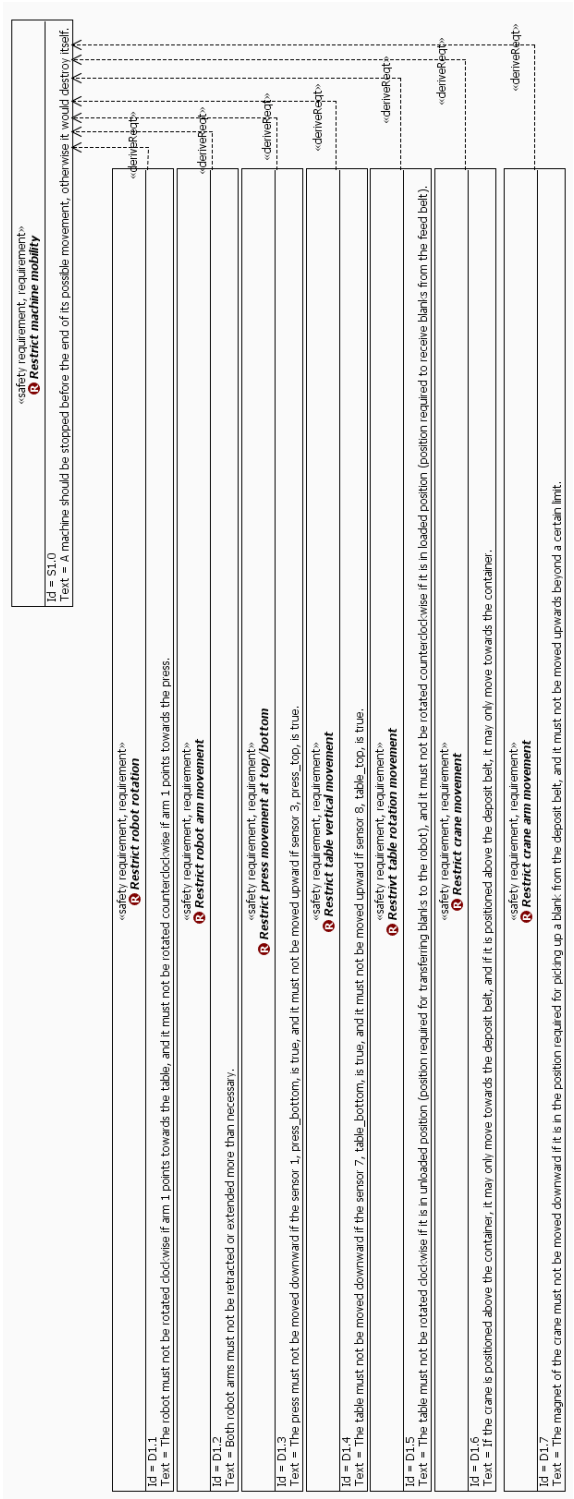
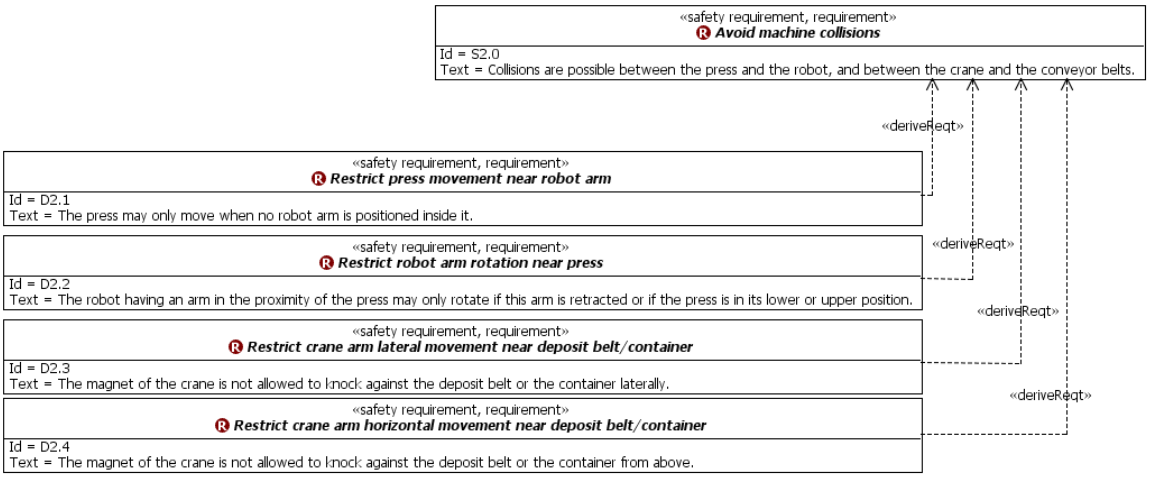


Figure 16 - req [Package] Requirements [req s1.0 derived]

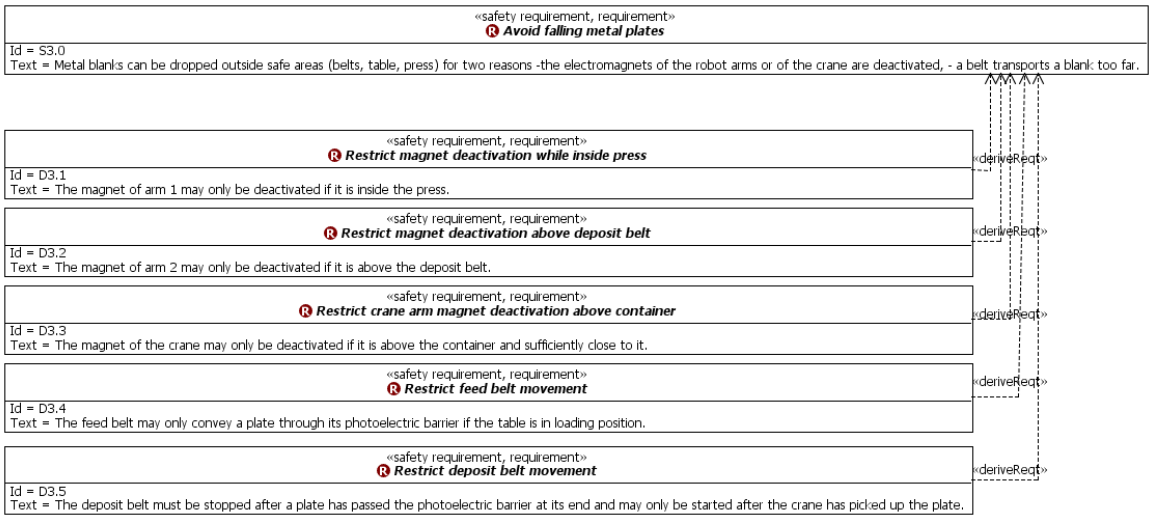
S1.0	**A machine should be stopped before the end of its possible movement, otherwise it would destroy itself.**
D1.1	The robot must not be rotated clockwise if arm 1 points towards the table, and it must not be rotated counterclockwise if arm 1 points towards the press.
D1.2	Both robot arms must not be retracted or extended more than necessary.
D1.3	The press must not be moved downward if the sensor 1, press_bottom, is true, and it must not be moved upward if sensor 3, press_top, is true.
D1.4	The table must not be moved downward if the sensor 7, table_bottom, is true, and it must not be moved upward if sensor 8, table_top, is true.
D1.5	The table must not be rotated clockwise if it is in unloaded position (position required for transferring plates to the robot), and it must not be rotated counterclockwise if it is in loaded position (position required to receive plates from the feed belt).
D1.6	If the crane is positioned above the container, it may only move towards the deposit belt, and if it is positioned above the deposit belt, it may only move towards the container.
D1.7	The magnet of the crane must not be moved downward if it is in the position required for picking up a plate from the deposit belt, and it must not be moved upwards beyond a certain limit.

A.3.2 Req S2.0



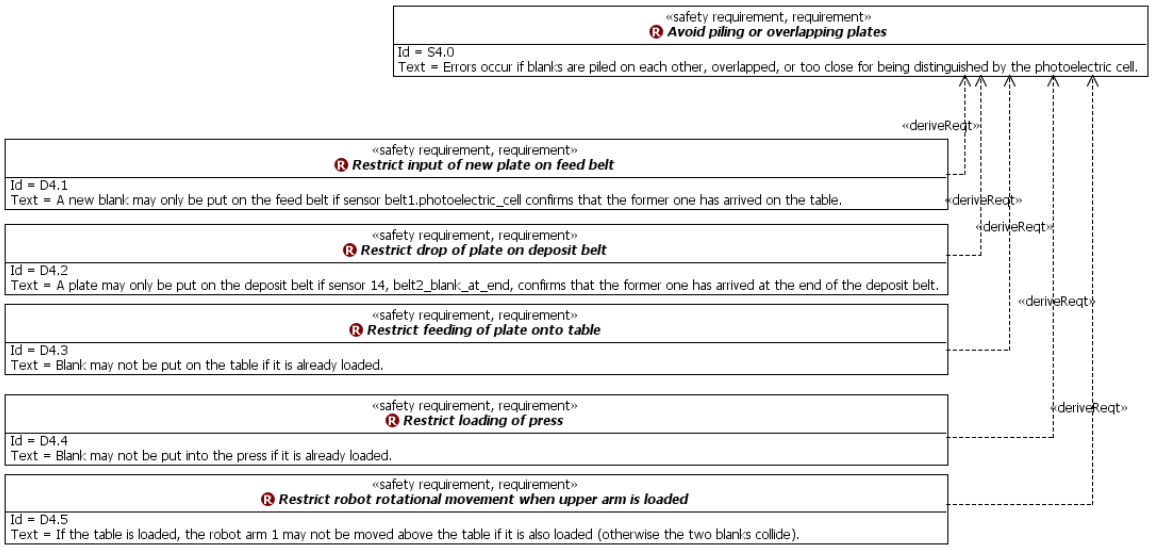
S2.0	**Collisions are possible between the press and the robot, and between the crane and the conveyor belts.**
D2.1	The press may only move when no robot arm is positioned inside it.
D2.2	The robot having an arm in the proximity of the press may only rotate if this arm is retracted or if the press is in its lower or upper position.
D2.3	The magnet of the crane is not allowed to knock against the deposit belt or the container laterally.
D2.4	The magnet of the crane is not allowed to knock against the deposit belt or the container from above.

A.3.3 Req S3.0



S3.0	**Metal blanks can be dropped outside safe areas (belts, table, press) for two reasons -the electromagnets of the robot arms or of the crane are deactivated, - a belt transports a blank too far.**
D3.1	The magnet of arm 1 may only be deactivated if it is inside the press.
D3.2	The magnet of arm 2 may only be deactivated if it is above the deposit belt.
D3.3	The magnet of the crane may only be deactivated if it is above the container and sufficiently close to it.
D3.4	The feed belt may only convey a plate through its photoelectric barrier if the table is in loading position.
D3.5	The deposit belt must be stopped after a plate has passed the photoelectric barrier at its end and may only be started after the crane has picked up the plate.

A.3.4 Req S4.0



S4.0	**Errors occur if blanks are piled on each other, overlapped, or too close for being distinguished by the photoelectric cell.**
D4.1	A new blank may only be put on the feed belt if sensor belt1.photoelectric_cell confirms that the former one has arrived on the table.
D4.2	A plate may only be put on the deposit belt if sensor 14, belt2_blank_at_end, confirms that the former one has arrived at the end of the deposit belt.
D4.3	Blank may not be put on the table if it is already loaded.
D4.4	Blank may not be put into the press if it is already loaded.
D4.5	If the table is loaded, the robot arm 1 may not be moved above the table if it is also loaded (otherwise the two blanks collide).

A.4 ValueType diagram

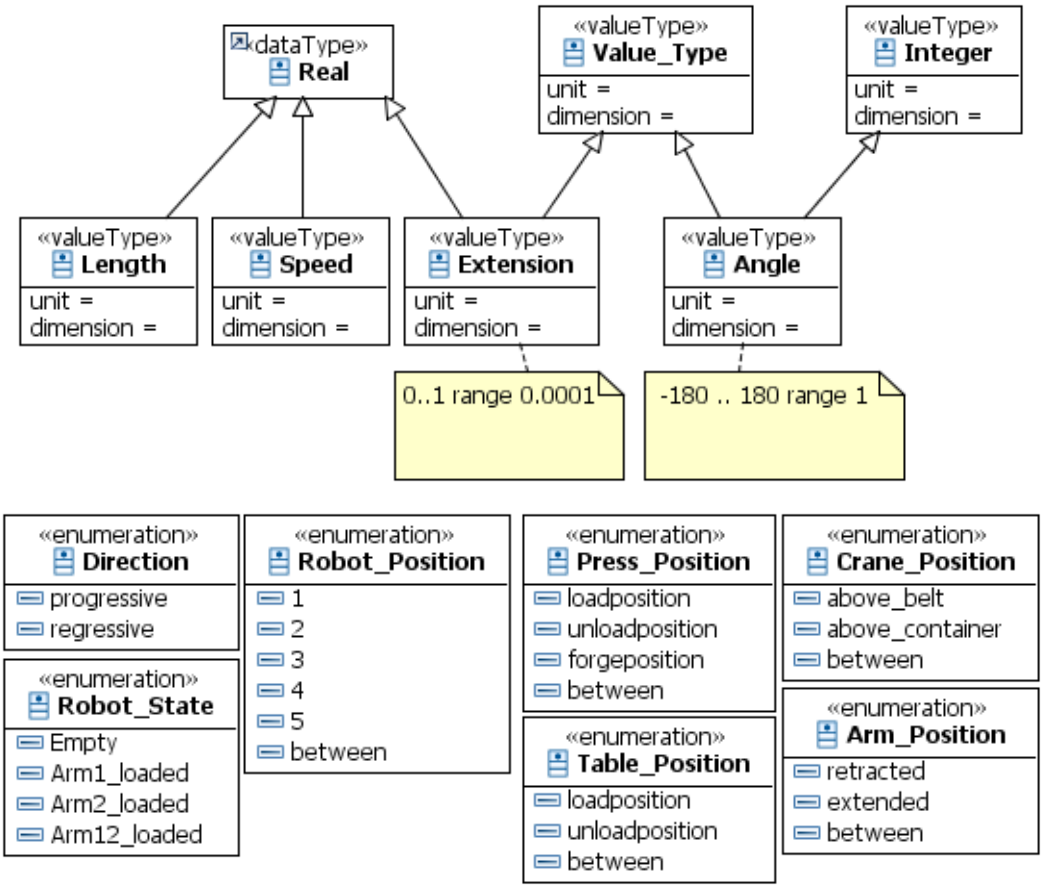


Figure 17 - bdd [Package] ValueTypes

A.5 Sequence diagrams

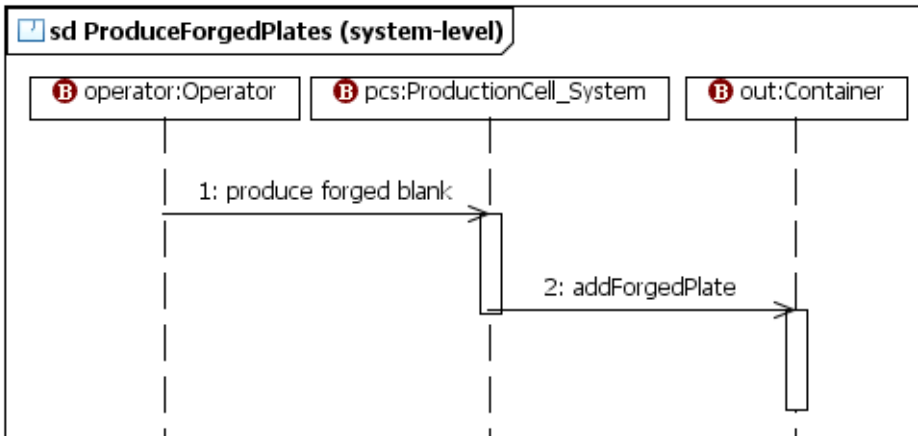


Figure 18 - sd produceForgedPlates [system-level interaction]

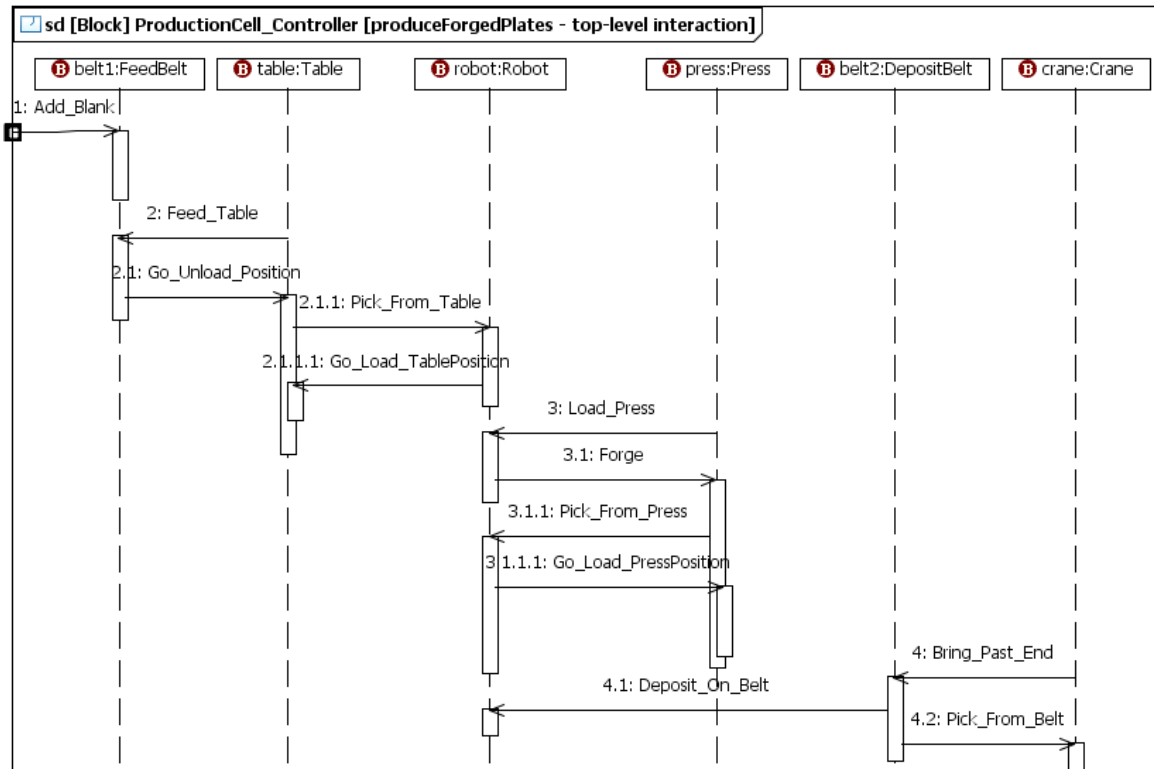


Figure 19 – sd [Block] ProductionCell_Controller [produceForgedPlates – top-level interaction]

This diagram only shows the interaction between the top-level blocks.

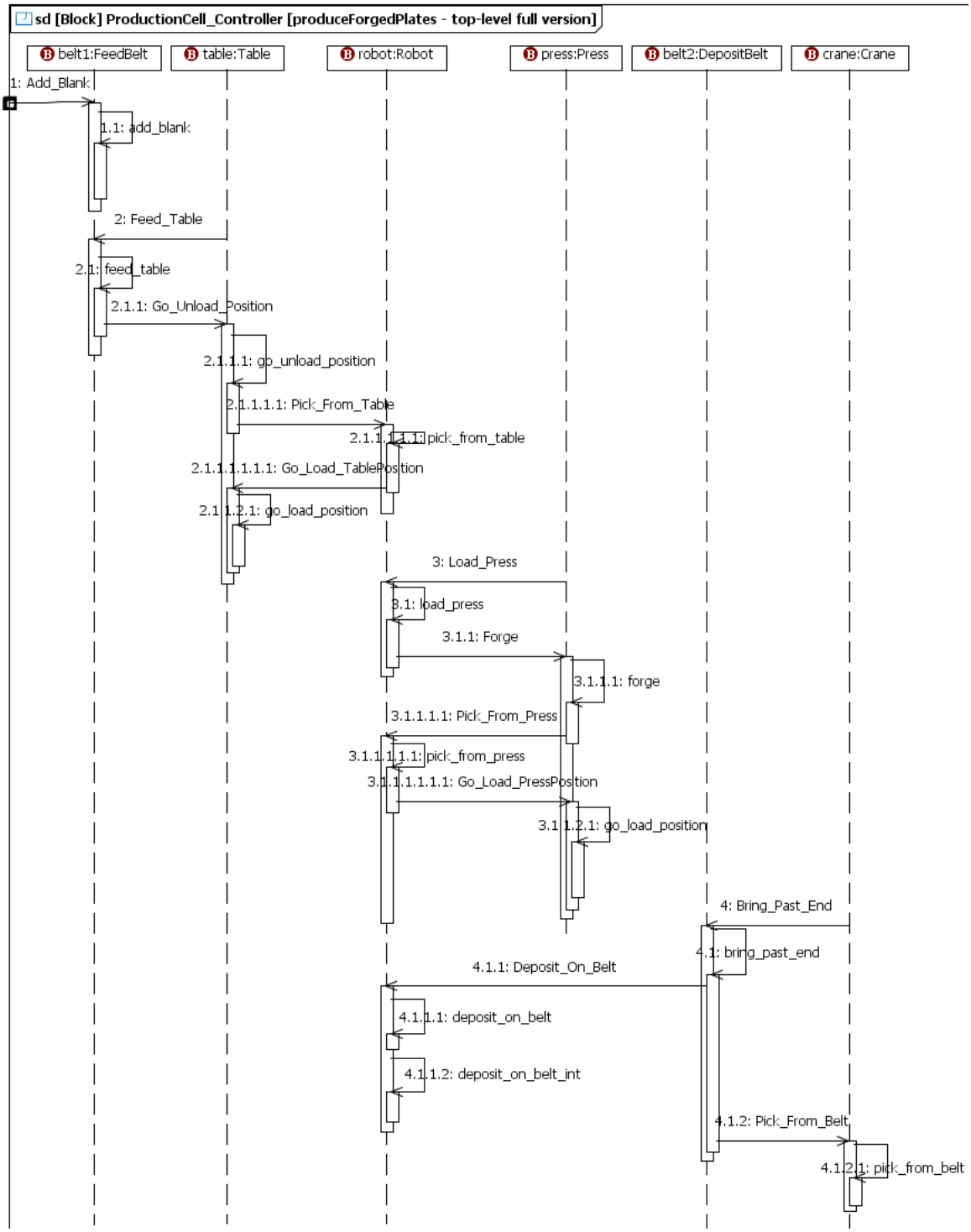


Figure 20 - sd [Block] ProductionCell_Controller [produceForgedPlates – top-level full version]

A.6 Activity diagrams

A.6.1 Block definition diagram

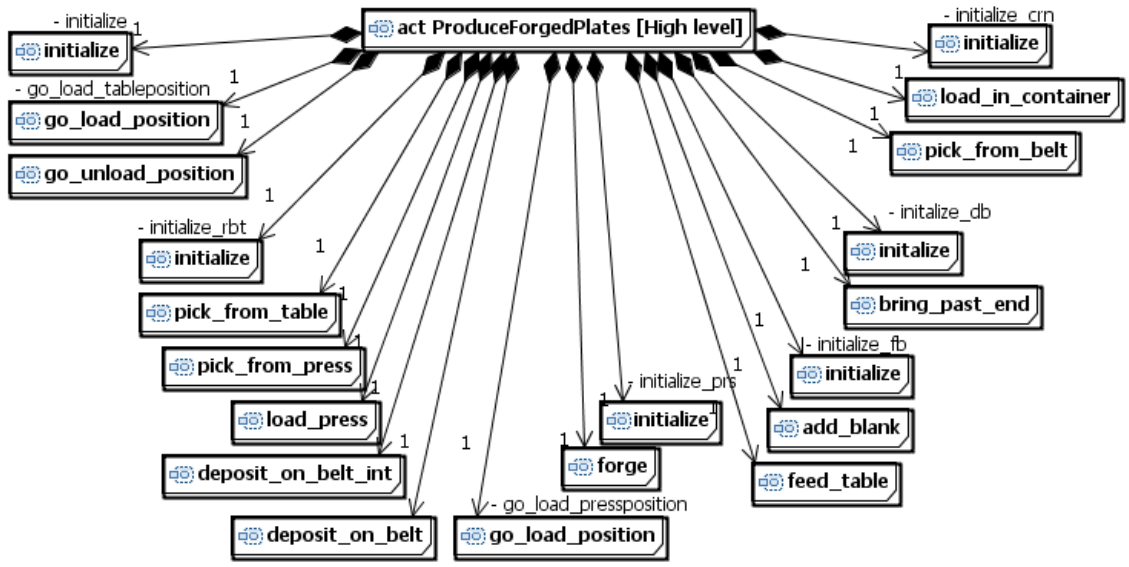


Figure 21 - bdd [Package] Activity [Composition of act produceForgedPlates]

A.6.2 High level

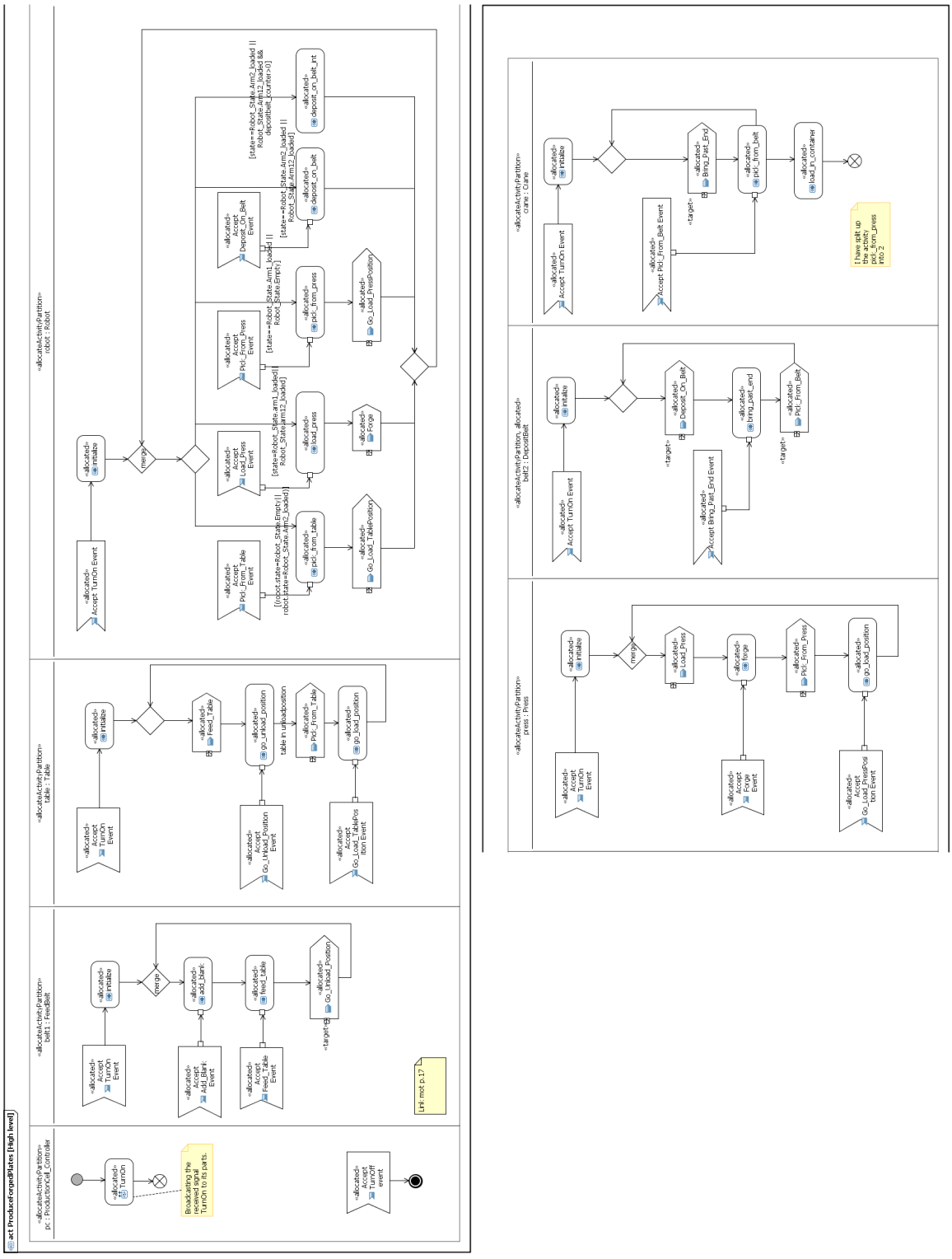


Figure 22 – The full activity diagram, act ProduceForgedPlates.
For a closer look, see Figure 23, Figure 24 and Figure 25.

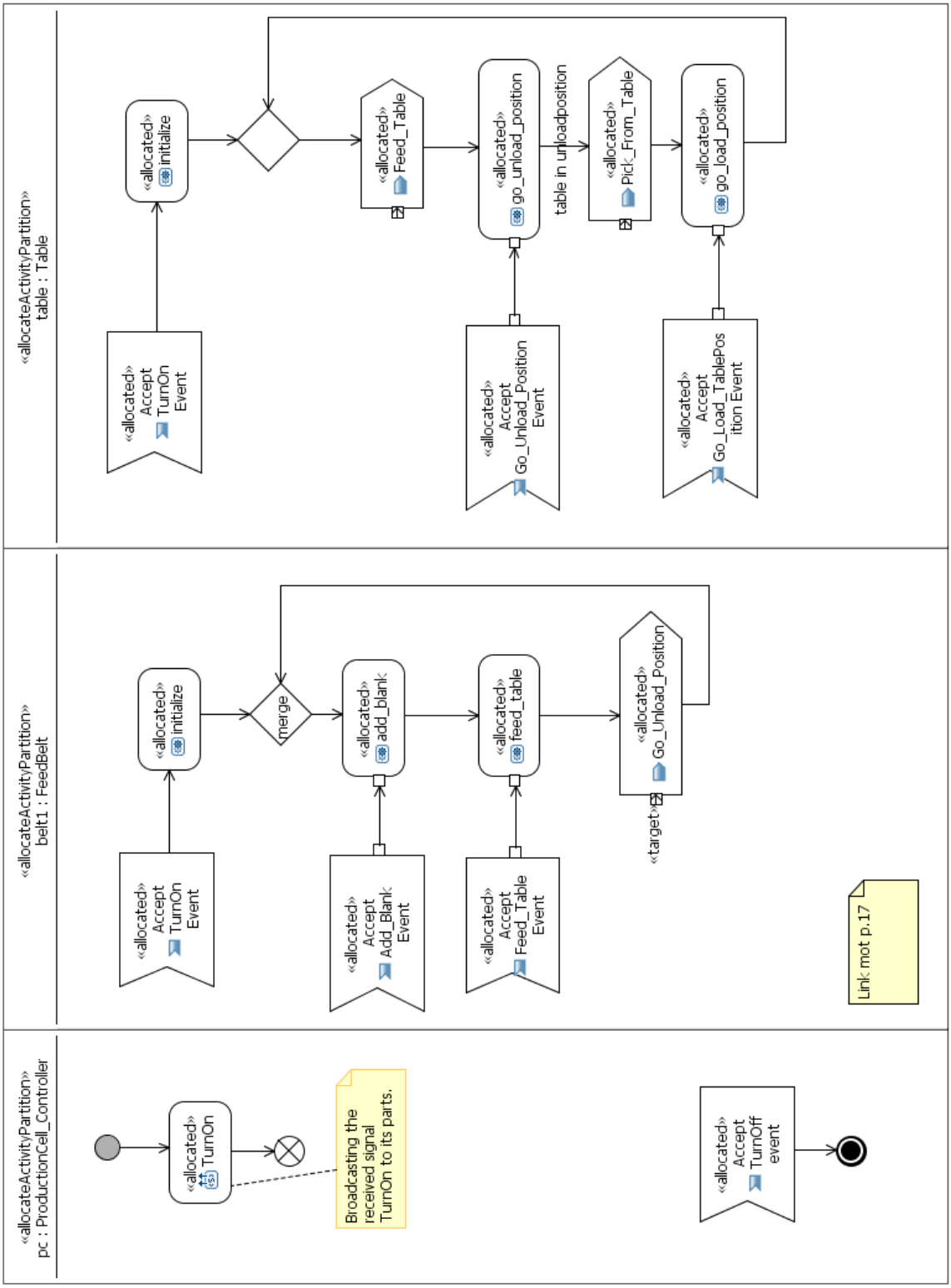


Figure 23 - Detail view of partitions for Controller, FeedBelt and Table from the diagram act ProduceForgedPlates in Figure 22

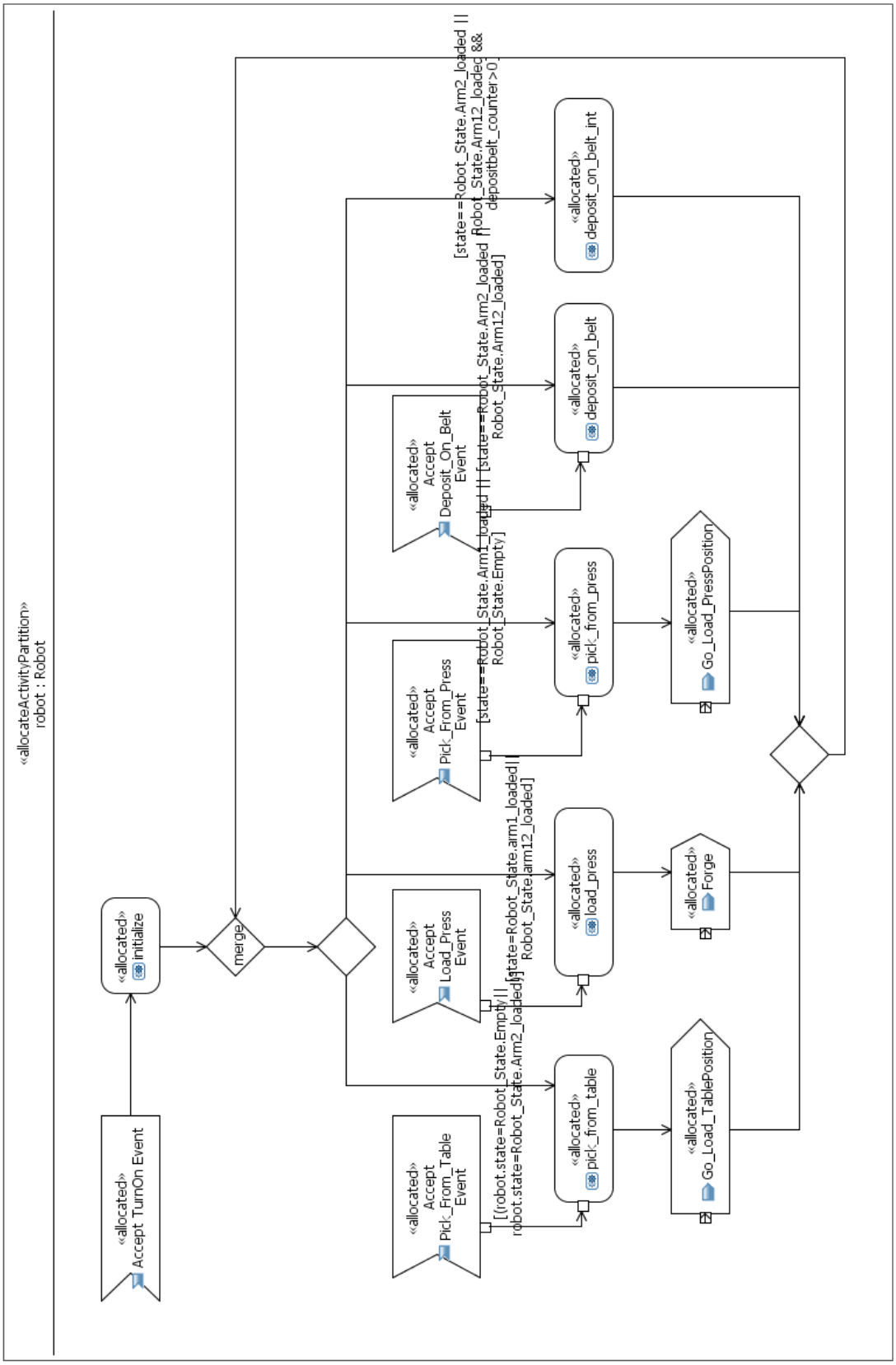


Figure 24 - Detail view of partitions for Robot from the diagram act ProduceForgedPlates in Figure 22. Note that there is an error in this diagram, see Error! Reference source not found. **for details.**

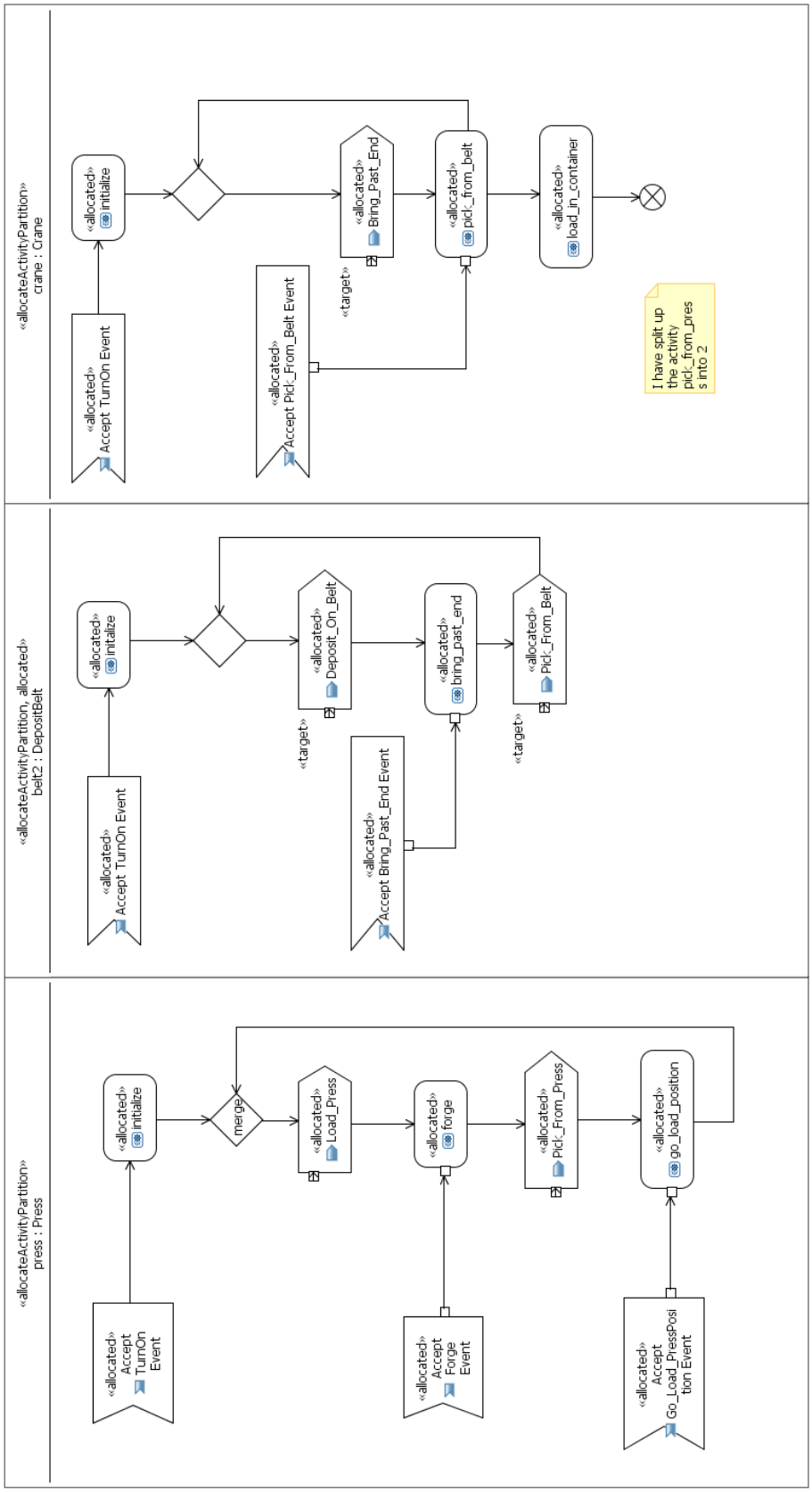


Figure 25 - Detail view of partitions for Press, DepositBelt and Crane from the diagram act ProduceForgedPlates in Figure 22.

A.6.3 Low level activity diagrams

A.6.3.1 FeedBelt

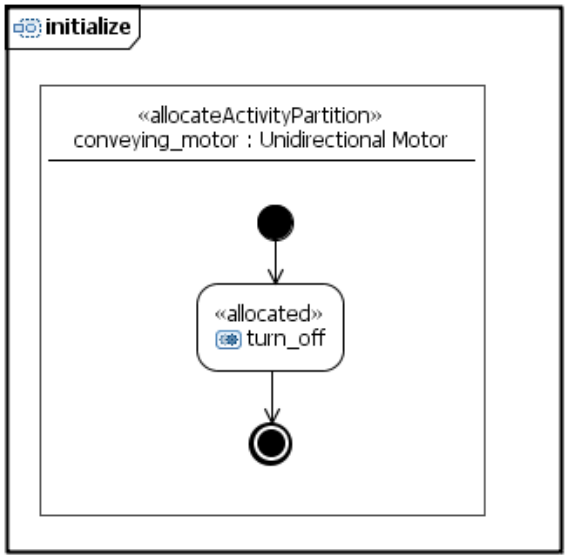


Figure 26 - Activity diagram for FeedBelt, act initialize, which is called from diagram in Figure 22.

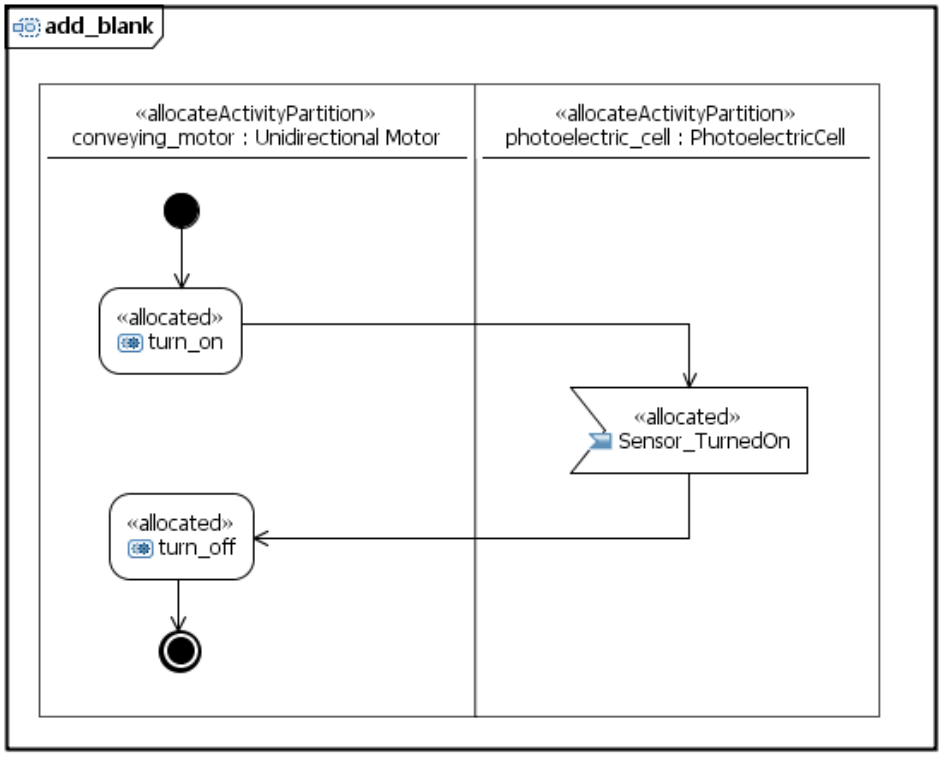


Figure 27 - Activity diagram for FeedBelt, act add_blank. which are called from diagram in Figure 22.

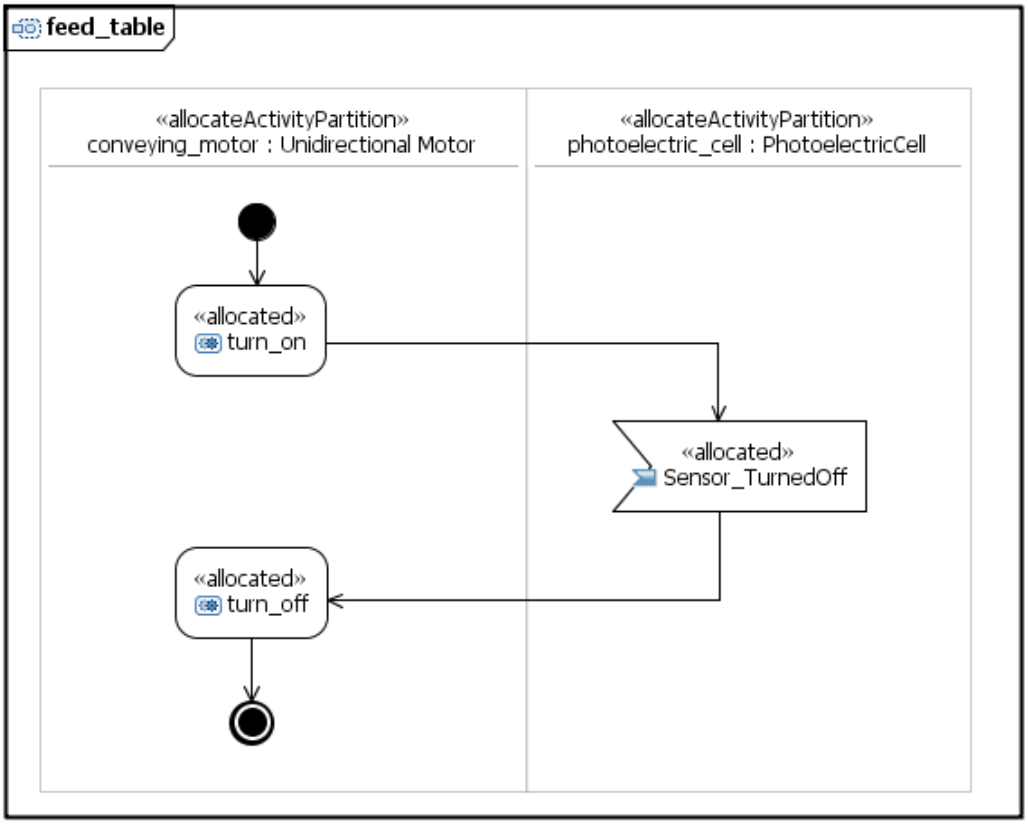


Figure 28 - Activity diagram for FeedBelt, act feed_table. which are called from diagram in Figure 22.

A.6.3.2 Table

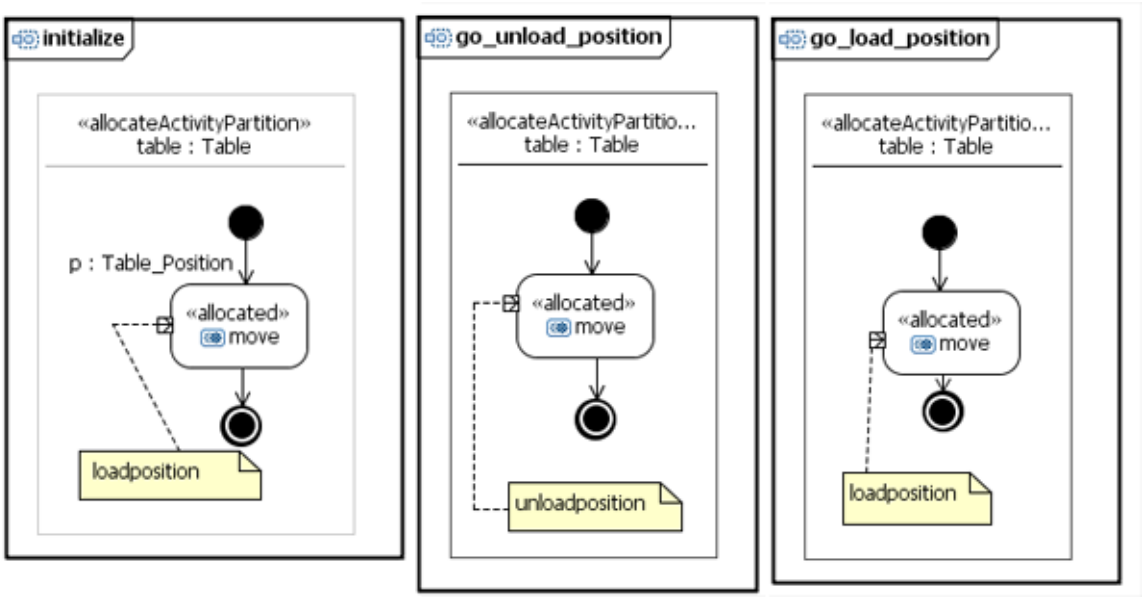


Figure 29 - Activity diagrams for Table, act initialize, act go_unload_position and act go_load_position. which are called from diagram in Figure 22.

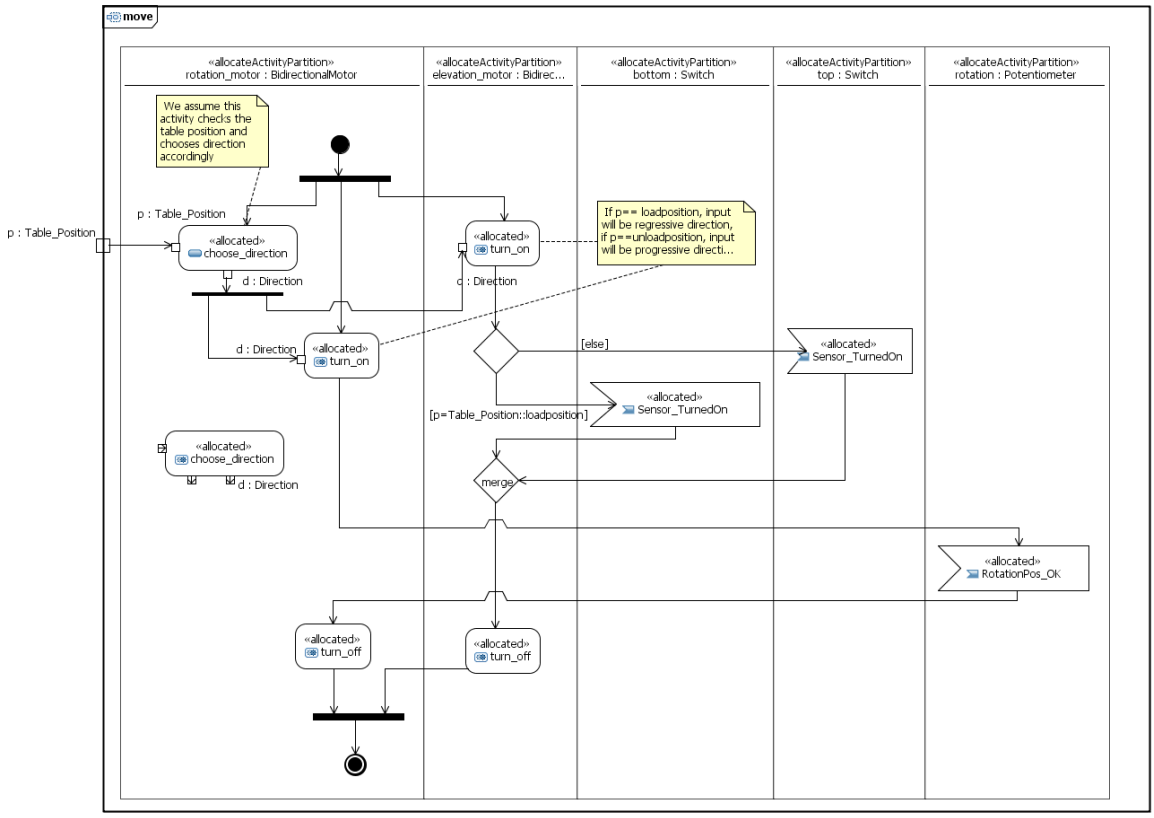


Figure 30 - Activity diagram for Table, act move, which is called from Figure 29.

A.6.3.3 Robot

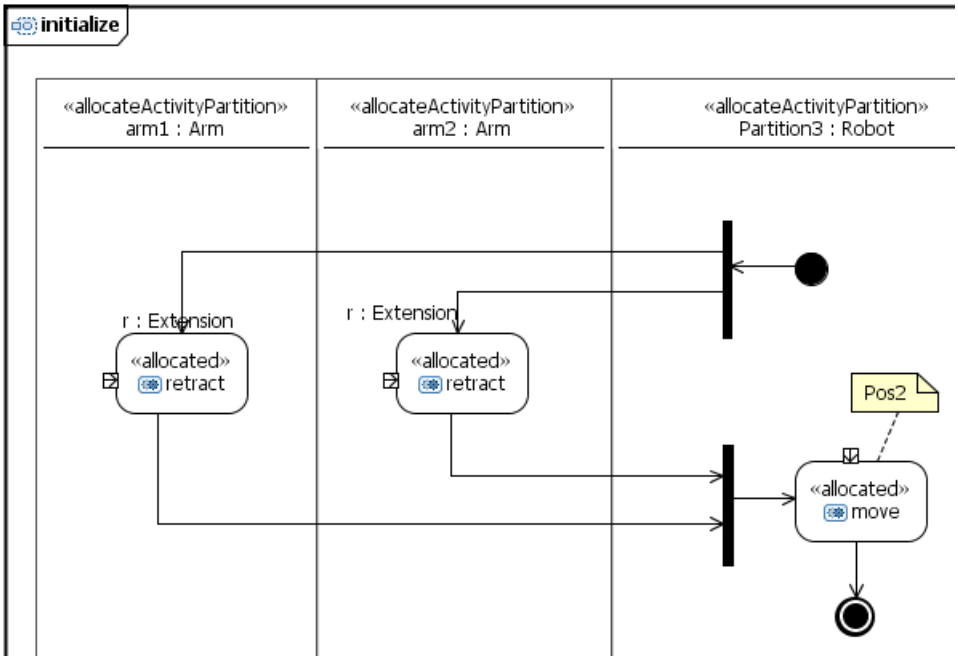


Figure 31 - Activity diagram for Robot, act initialize, which is called from Figure 22.

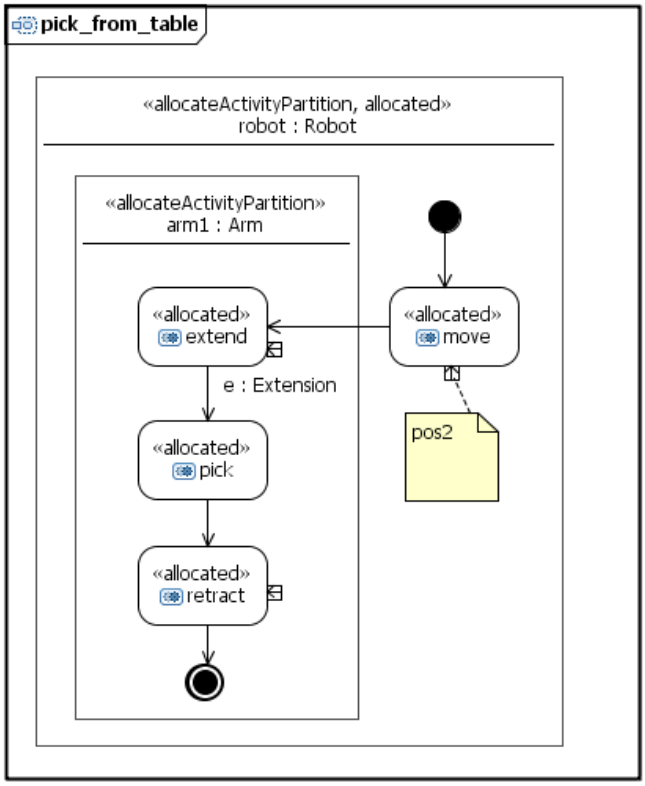


Figure 32 - Activity diagram for Robot, act pick_from_table, which is called from the diagram in Figure 22.

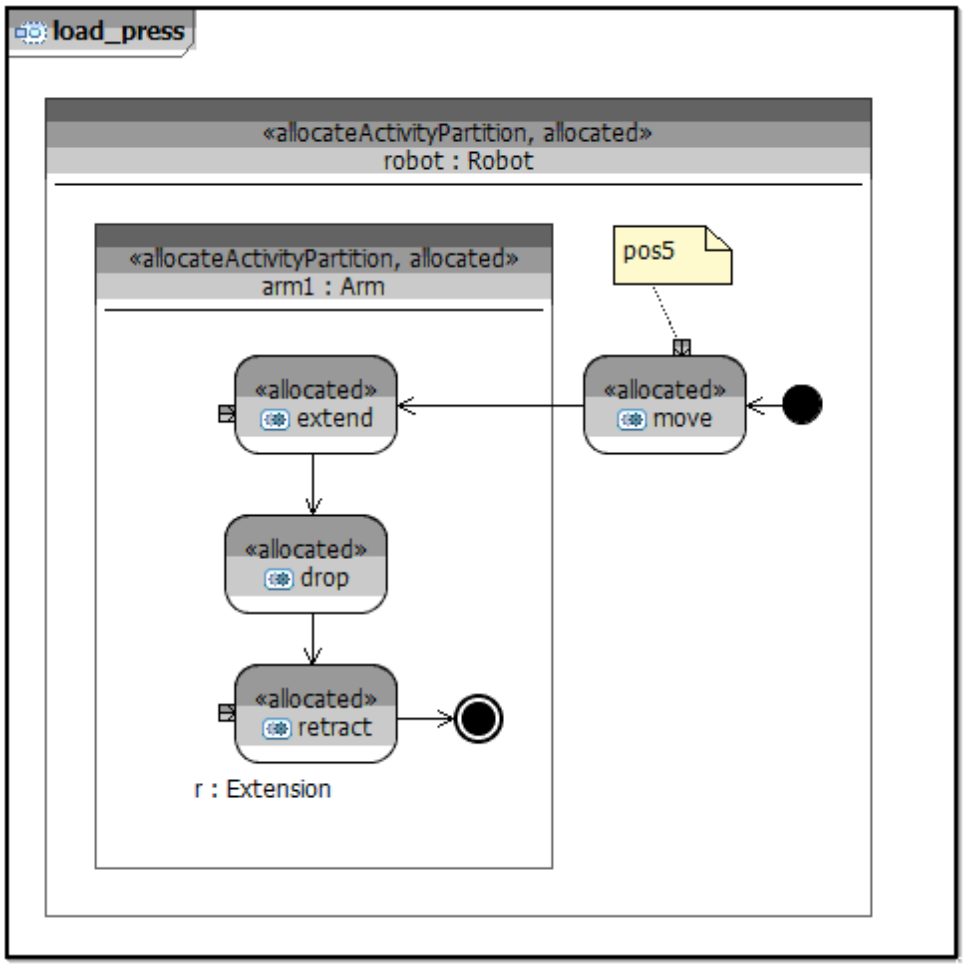


Figure 33 – Activity diagram for Robot, act load_press, which is called from the diagram in Figure 22.

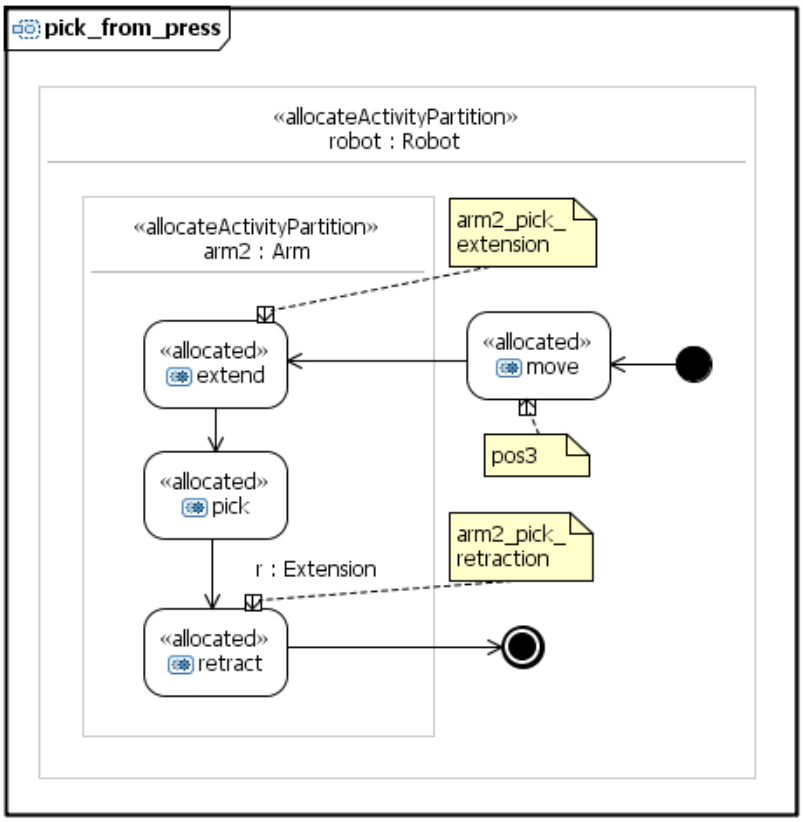


Figure 34 – Activity diagram for Robot, act pick_from_press, which is called from the diagram in Figure 22.

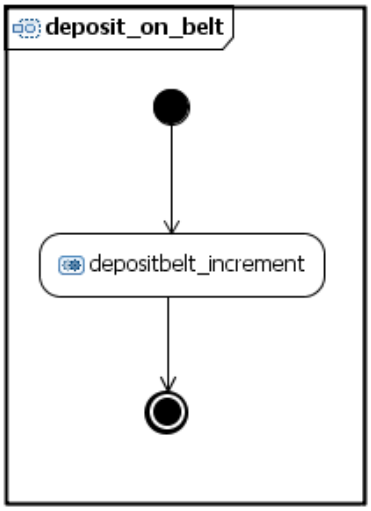


Figure 35 – Activity diagram for Robot, act deposit_on_belt, which is called from the diagram in Figure 22.

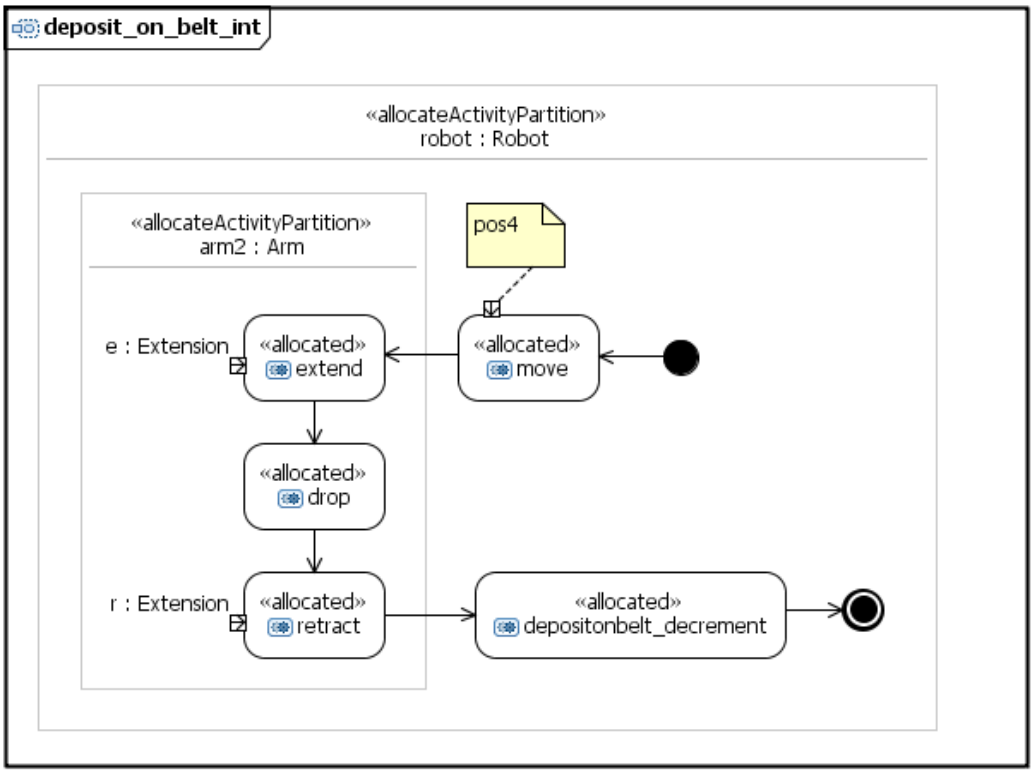


Figure 36 - Activity diagram for Robot, act deposit_on_belt_int, which is called from the diagram in Figure 22.

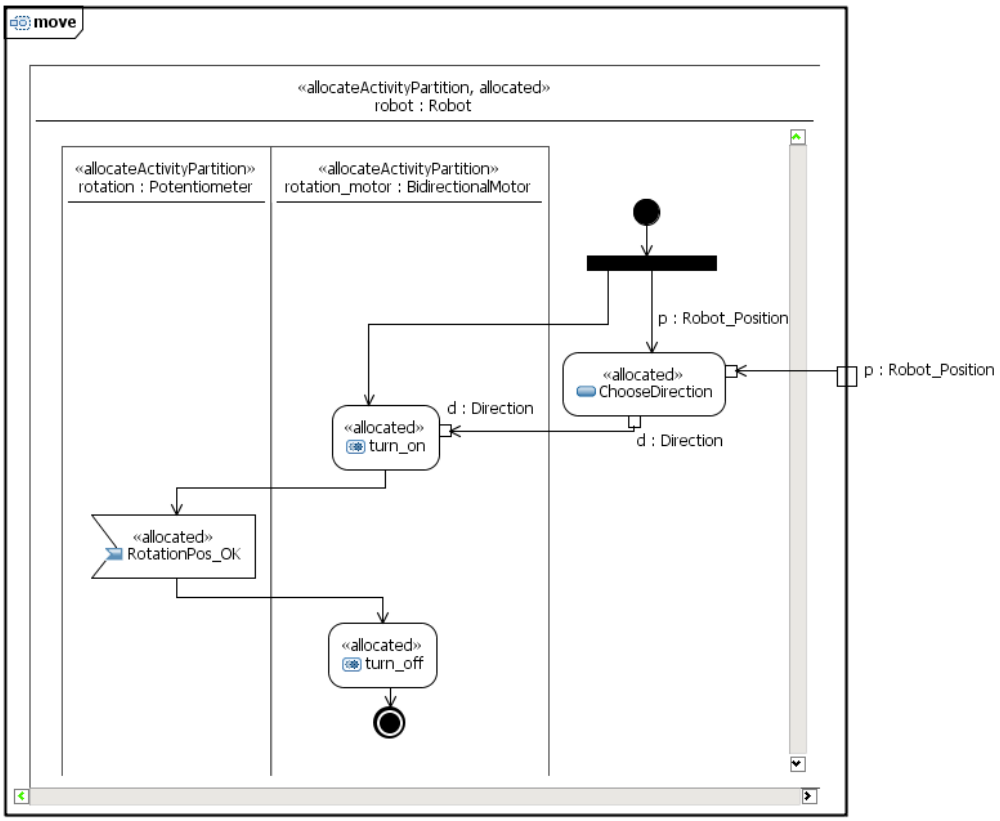


Figure 37 – Activity diagram for Robot, act move, which is called from most of the above Robot diagrams.

A.6.3.4 Press

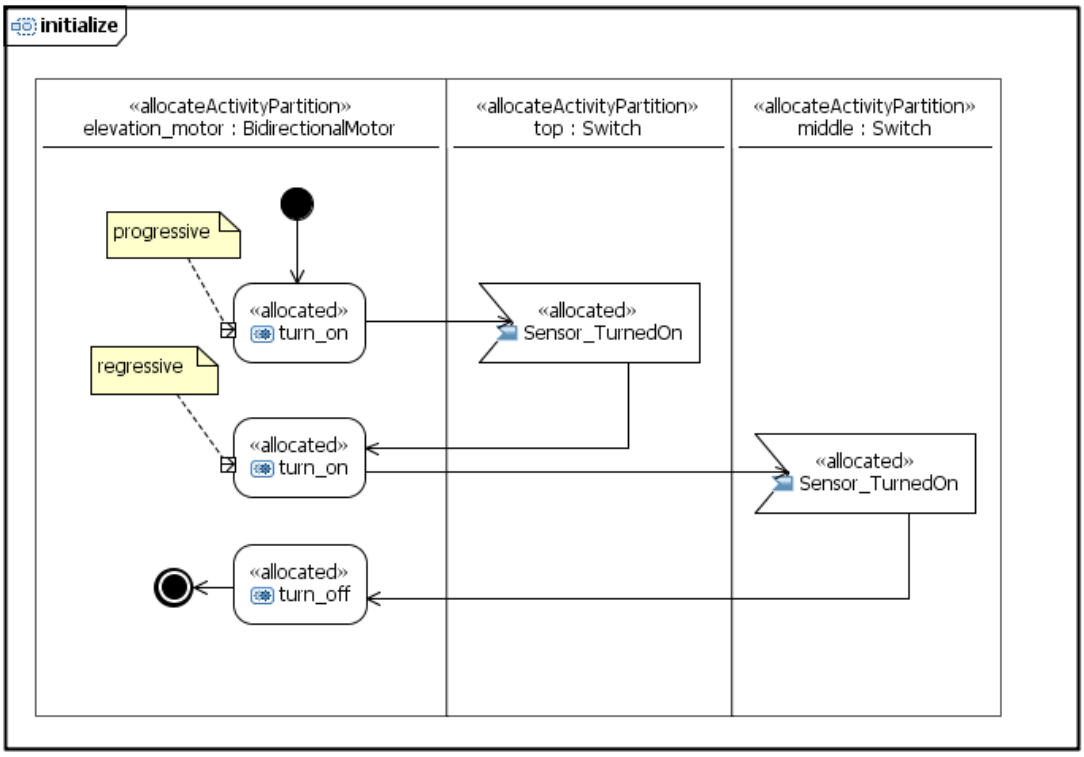


Figure 38 - Activity diagram for Press, act initialize, which is called from Figure 22.

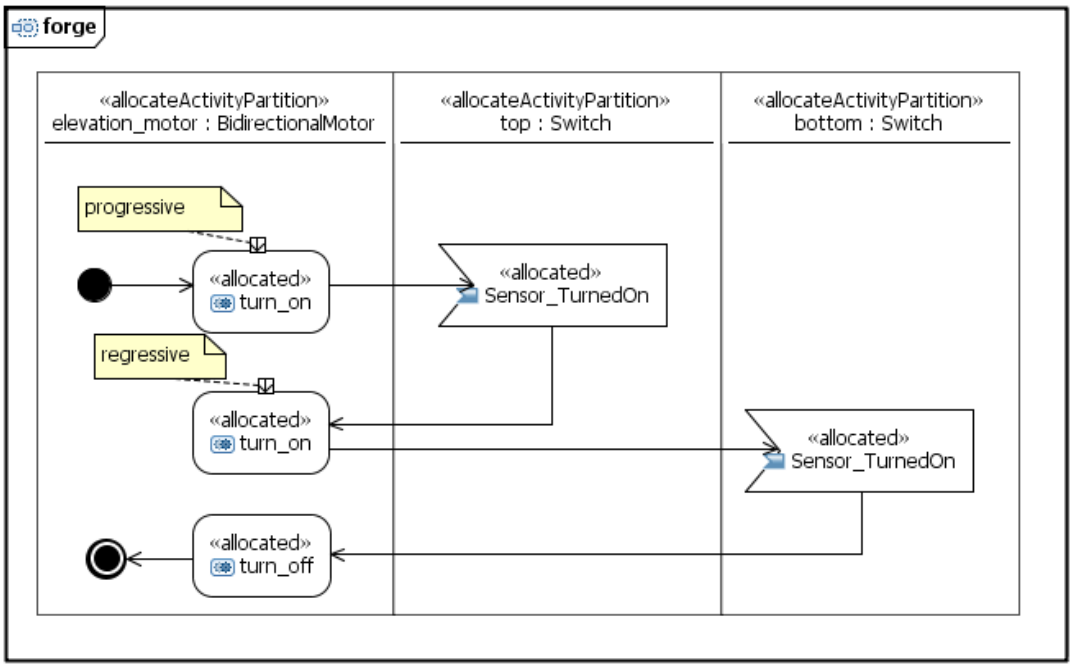


Figure 39 - Activity diagram for Press, act forge, which is called from the diagram in Figure 22.

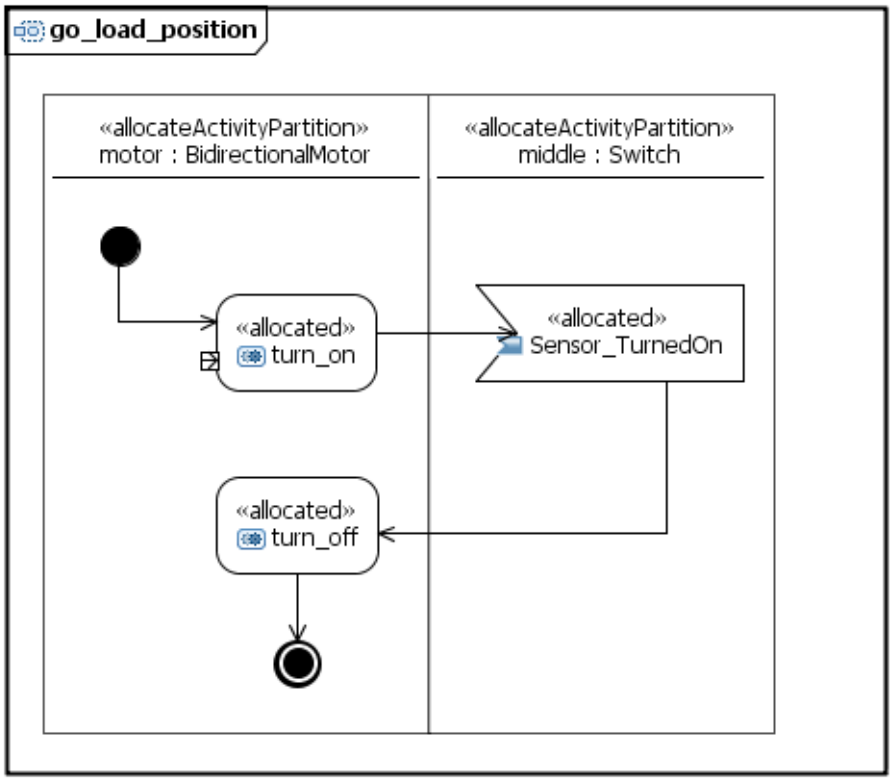


Figure 40 - Activity diagram for Press, act go_load_position, which is called from the diagram in Figure 22.

A.6.3.5 DepostiBelt

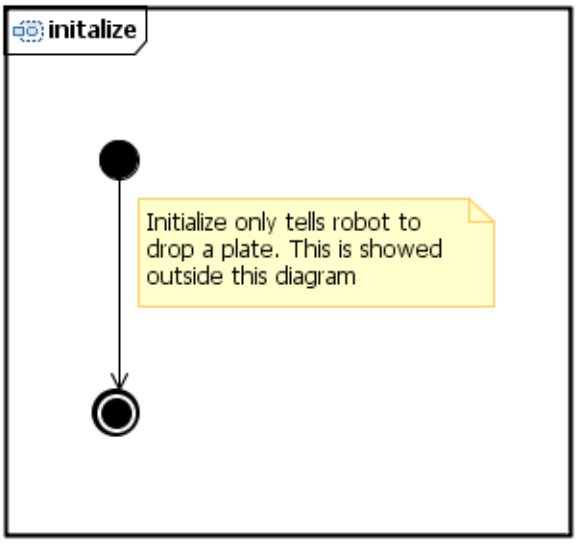


Figure 41 - Activity diagram for DepositBelt, act initialize, which is called from the diagram in Figure 22.

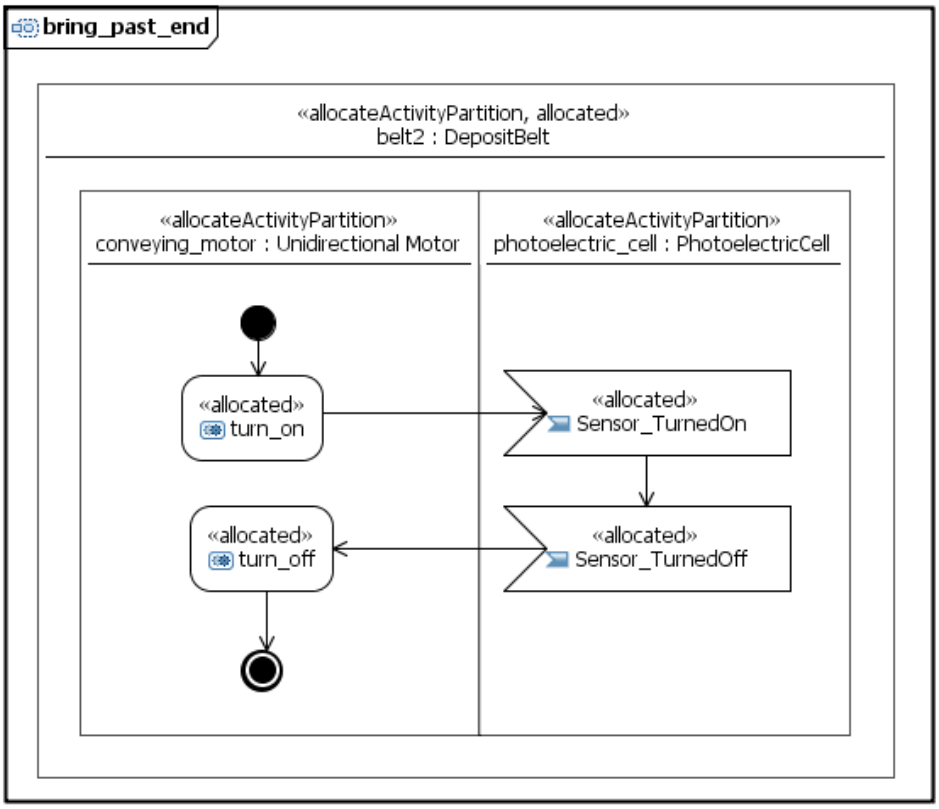


Figure 42 - Activity diagram for DepostiBelt, act bring_past_end, which is called from the diagram in Figure 22.

A.6.3.6 Crane

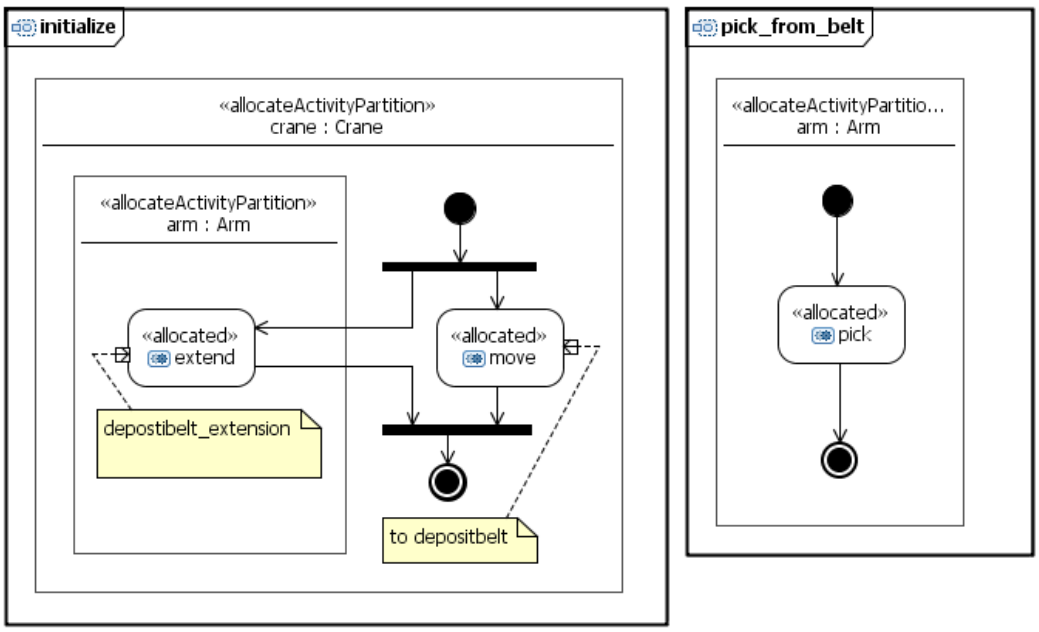


Figure 43 - Activity diagram for Crane, act initialize and act pick_from_belt, which are called from the diagram in Figure 22.

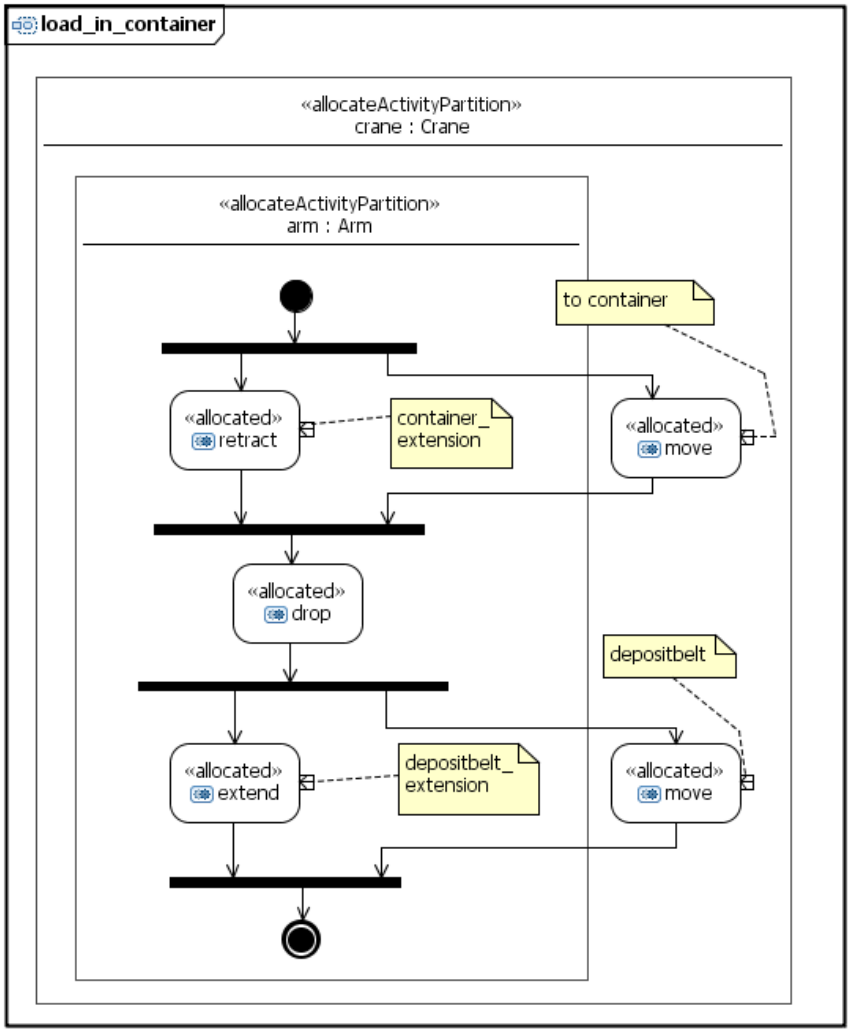


Figure 44 - Activity diagram for Crane, act load_in_container, which is called from the diagram in Figure 22. Note that this activity diagram describes the second part of the method pick_from_belt as it is described in [5], first part is described act pick_from_belt in Figure 43.

A.6.3.7 Arm

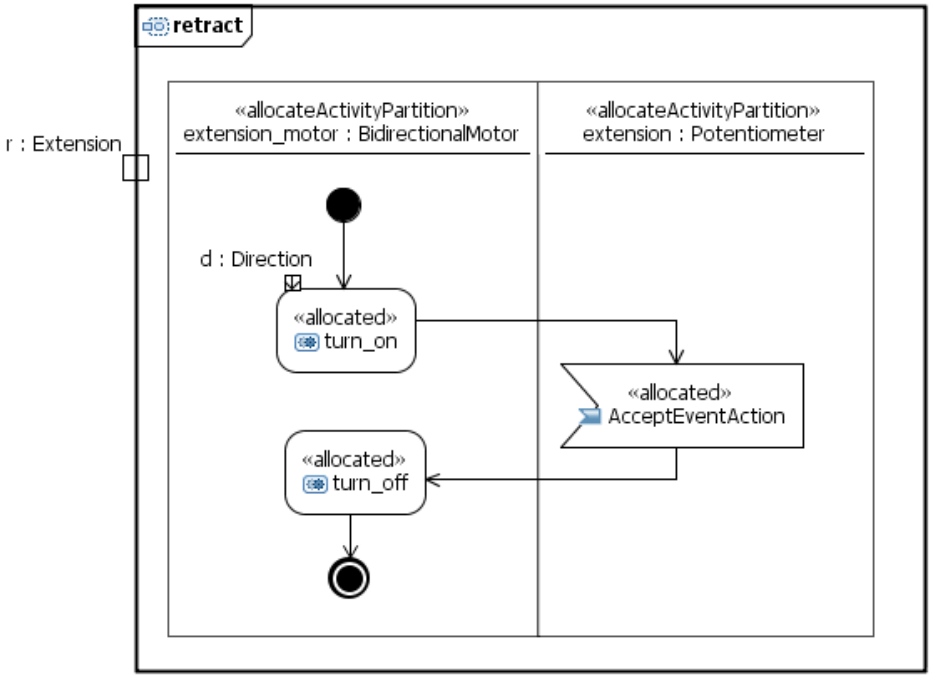


Figure 45 - Activity diagram for Arm, act retract, which is called from Robot and Crane activities.

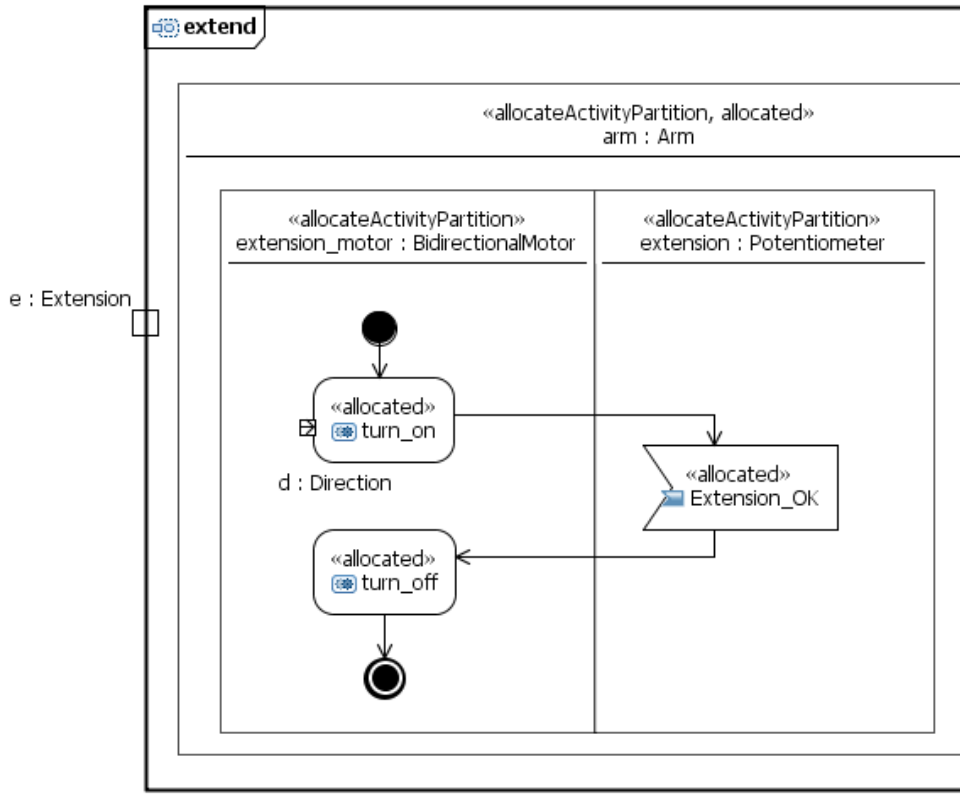


Figure 46 - Activity diagram for Arm, act extend, which is called from Robot and Crane activities.

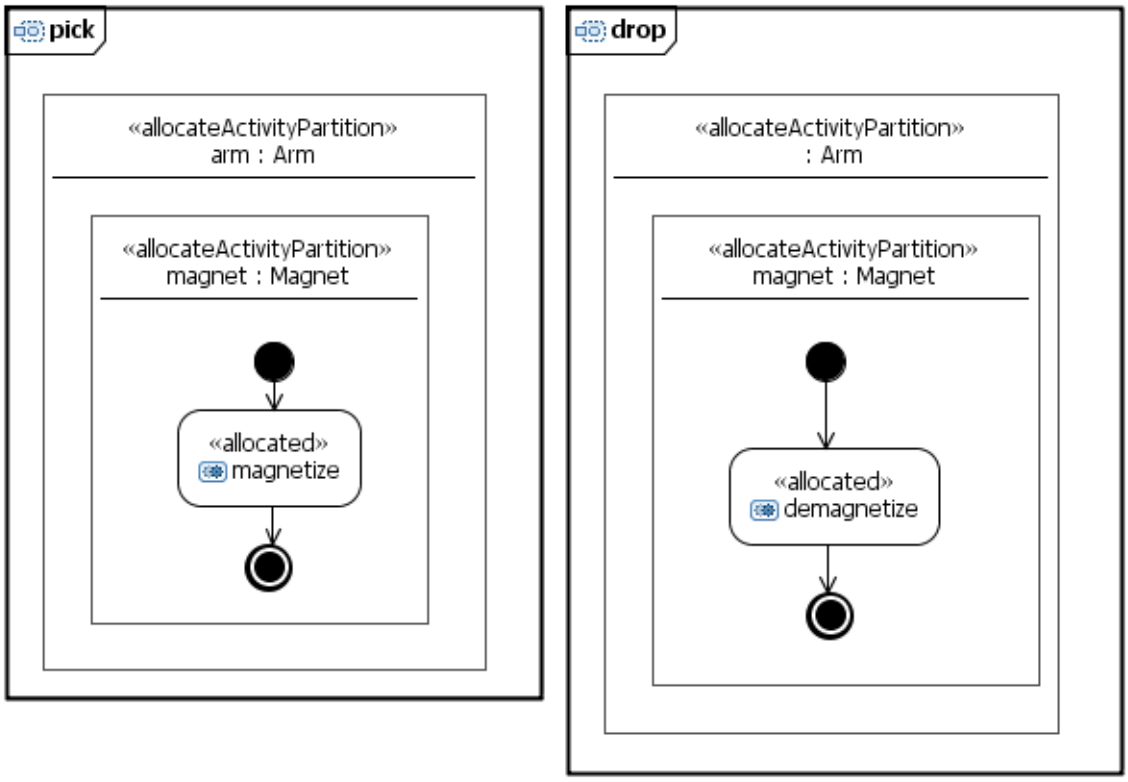


Figure 47 – Activity diagrams for Arm, act pick and act drop, which are called from Robot and Crane activities.

A.6.3.8 UniDirectionalMotor

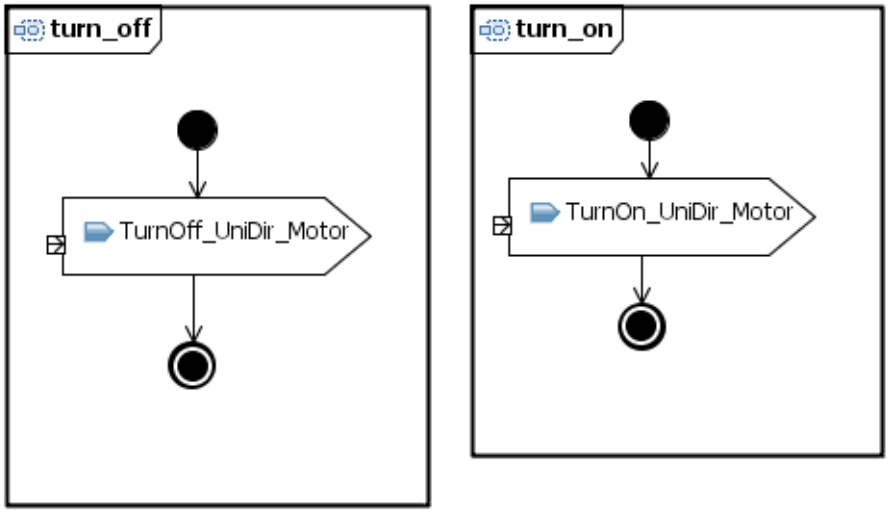


Figure 48 – Activity diagrams for UniDirectionalMotor, act turn_off and act turn_on, which are called from FeedBelt and DepositBelt activities.

A.6.3.9 BiDirectionalMotor

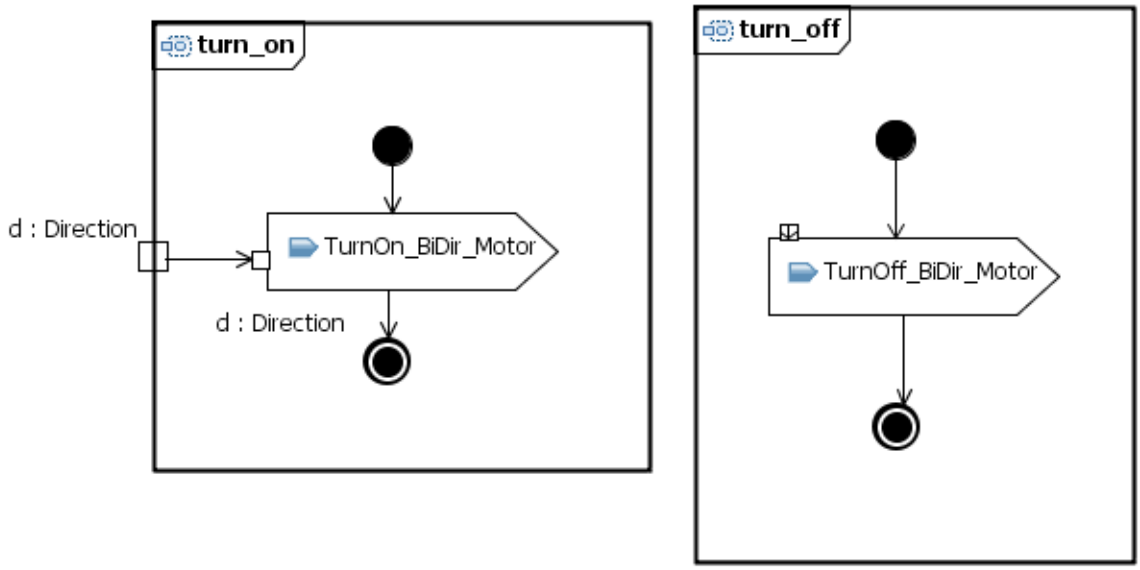


Figure 49 - Activity diagrams for BiDirectionalMotor, act turn_on and act turn_off, which are called from Table, Robot, Press and Crane activities.

A.6.3.10 Magnet

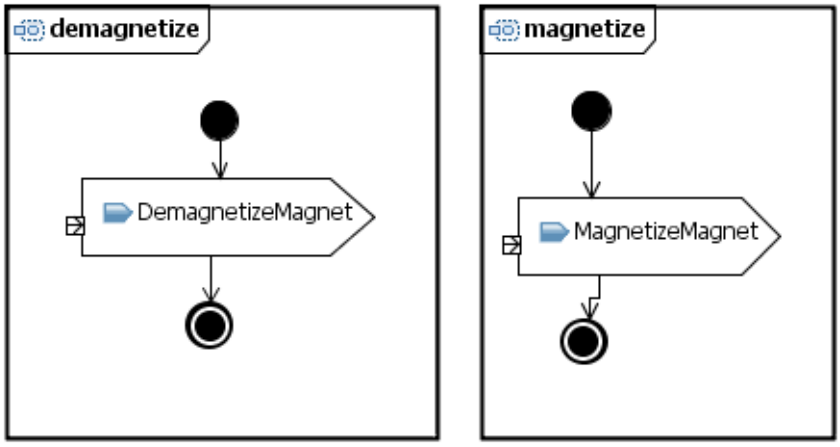


Figure 50 - Activity diagrams for Magnet, act demagnetize and act magnetize, which are called from Arm activities in Figure 47.

A.7 State machines

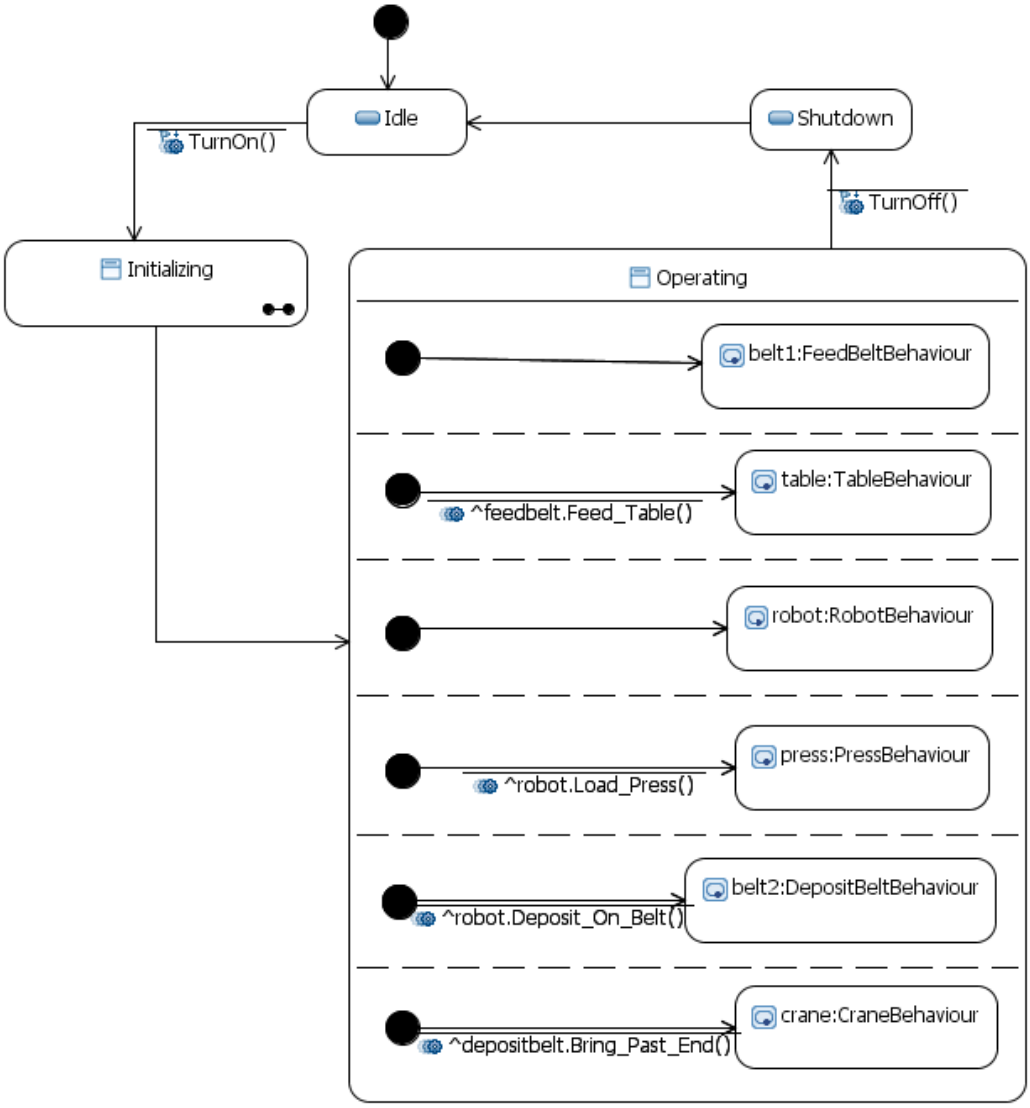


Figure 51 stm [Block] ProductionCell_Controller, with focus on the state Operating.

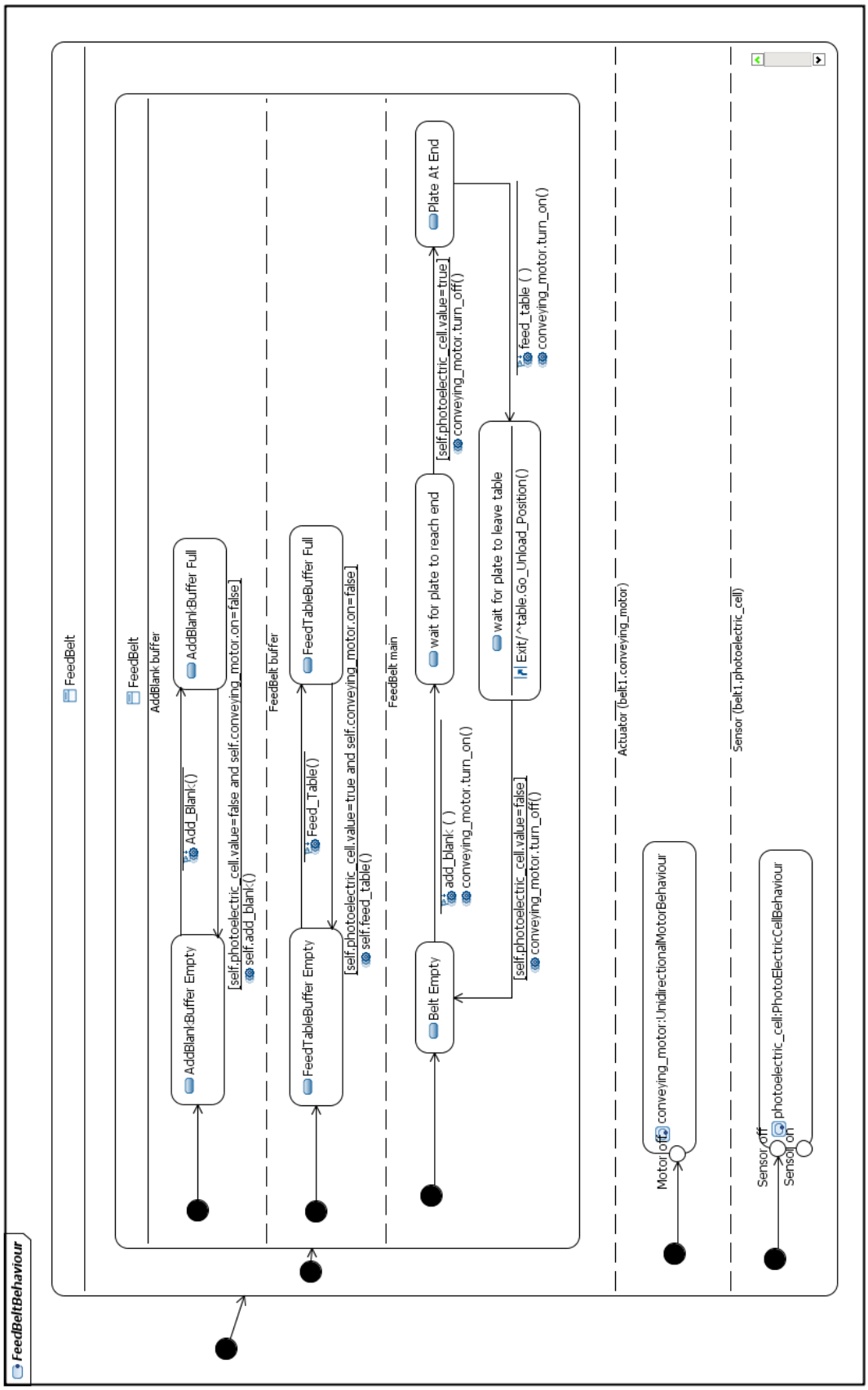


Figure 52 - stm [Block] FeedBelt [Operating].

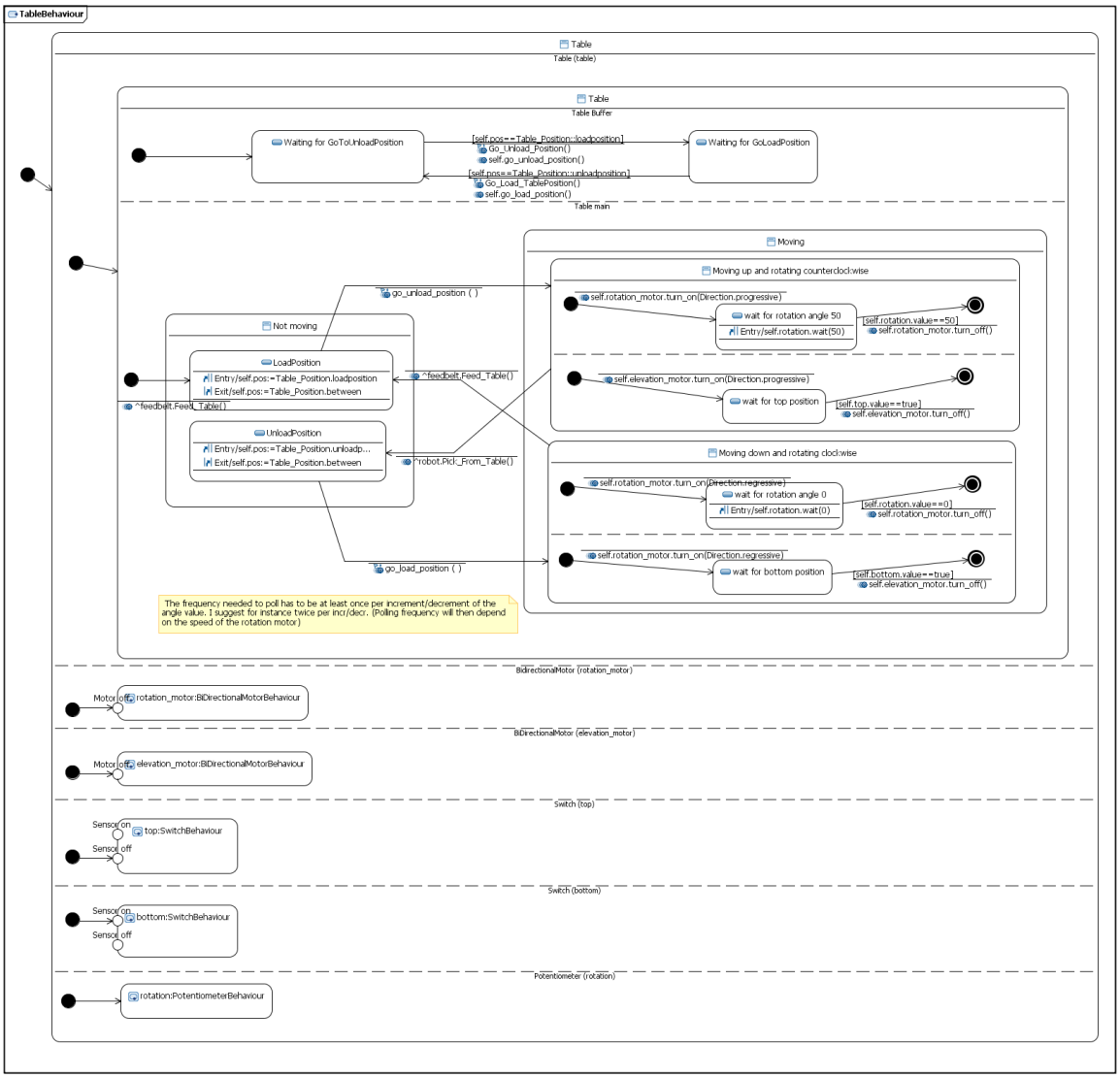


Figure 53 - stm [Block] Table [Operating]. For details, see Figure 54 and Figure 55.

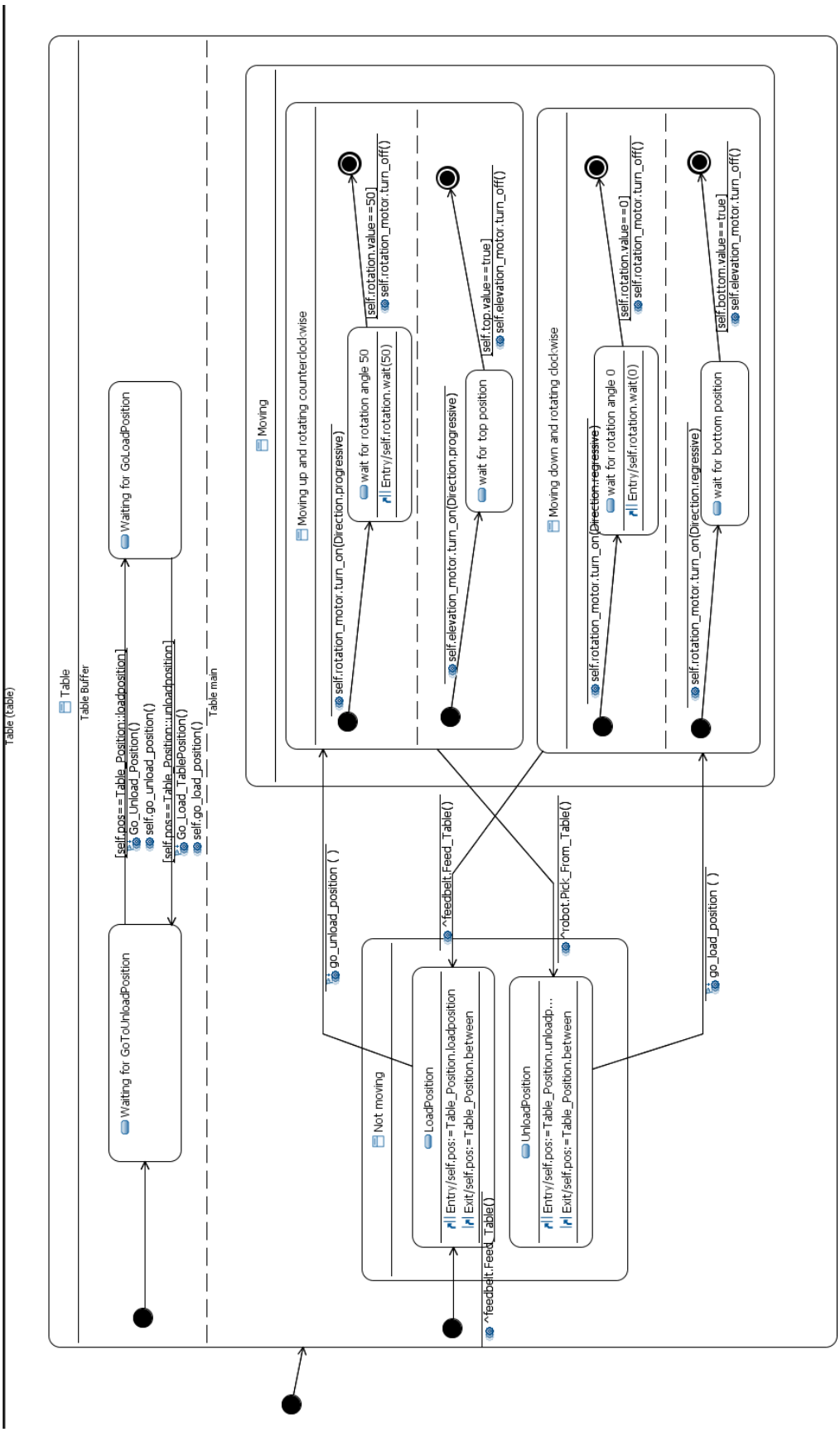


Figure 54 – Part of state machine for Table, showing only buffer and main regions. For whole diagram, see Figure 53.

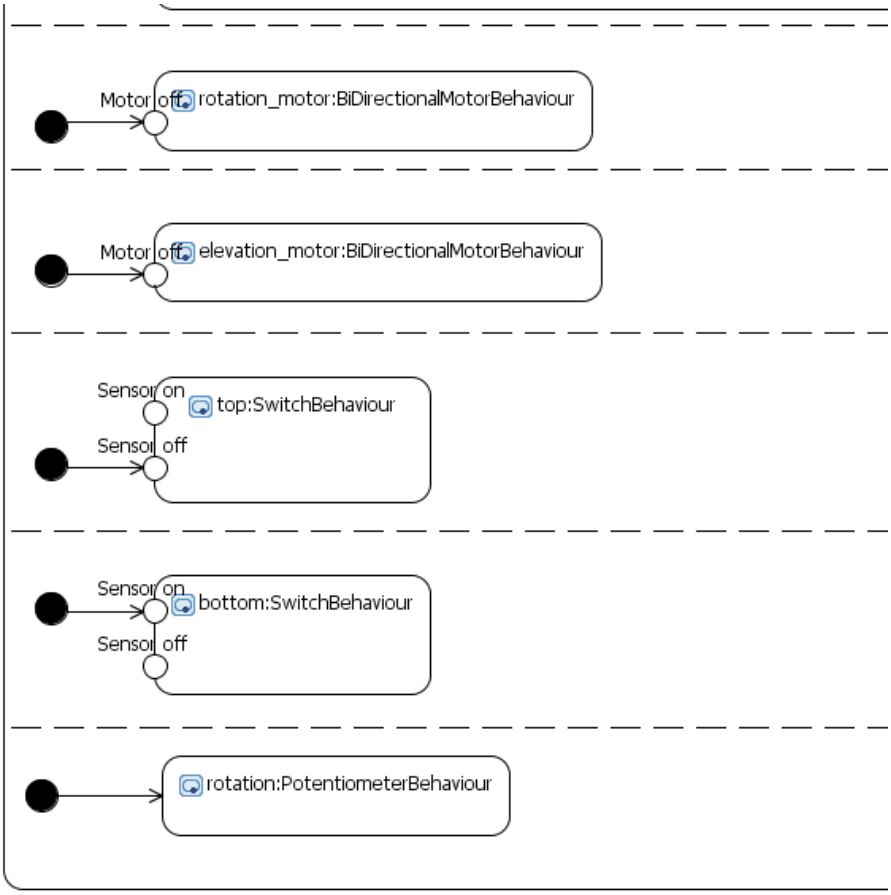


Figure 55 – Part of state machine for Table, showing only the part regions. For whole diagram see Figure 53.

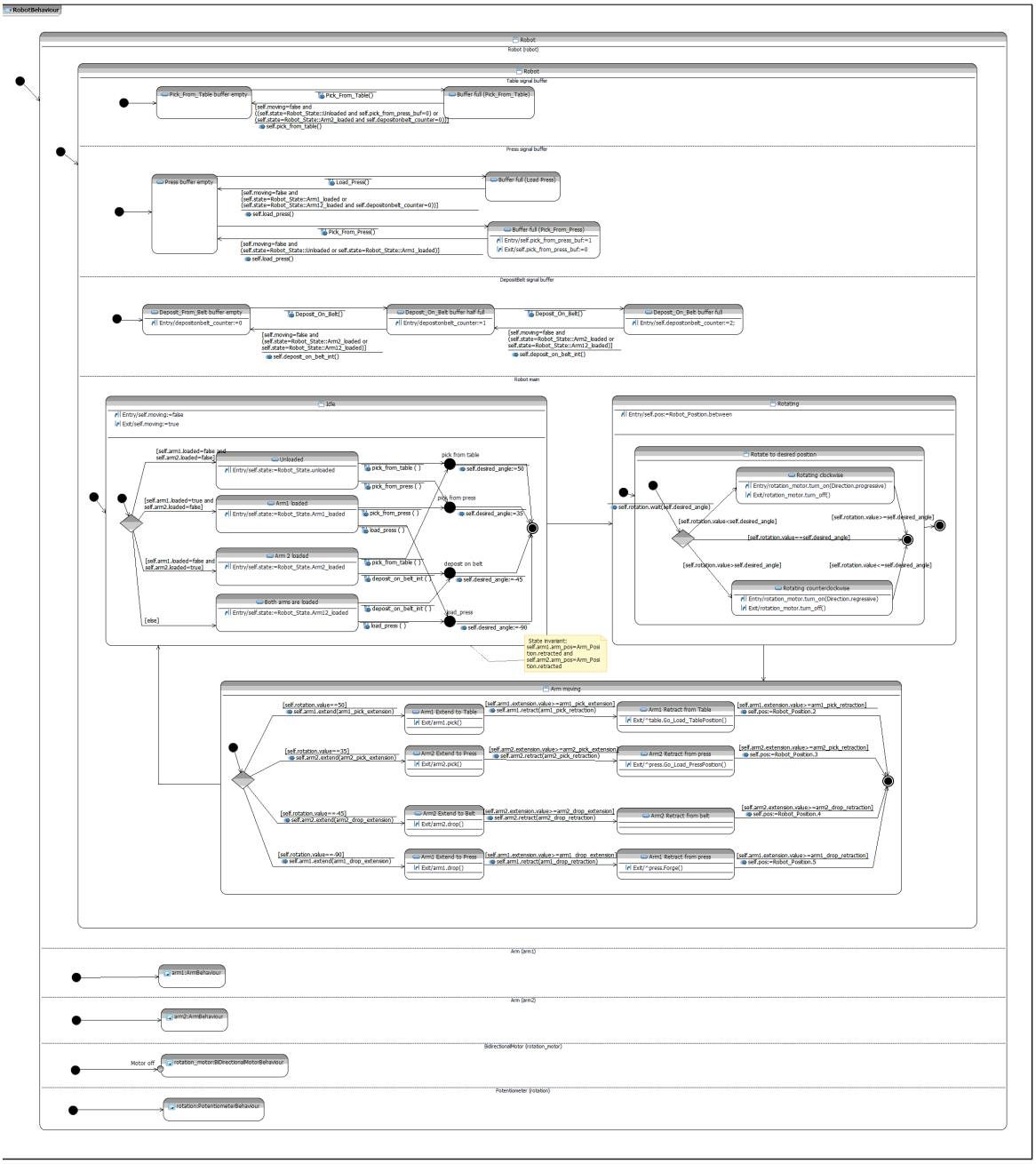


Figure 56 - stm [Block] Robot [Operating], for detail view, see Figure 57, Figure 58 and Figure 59.

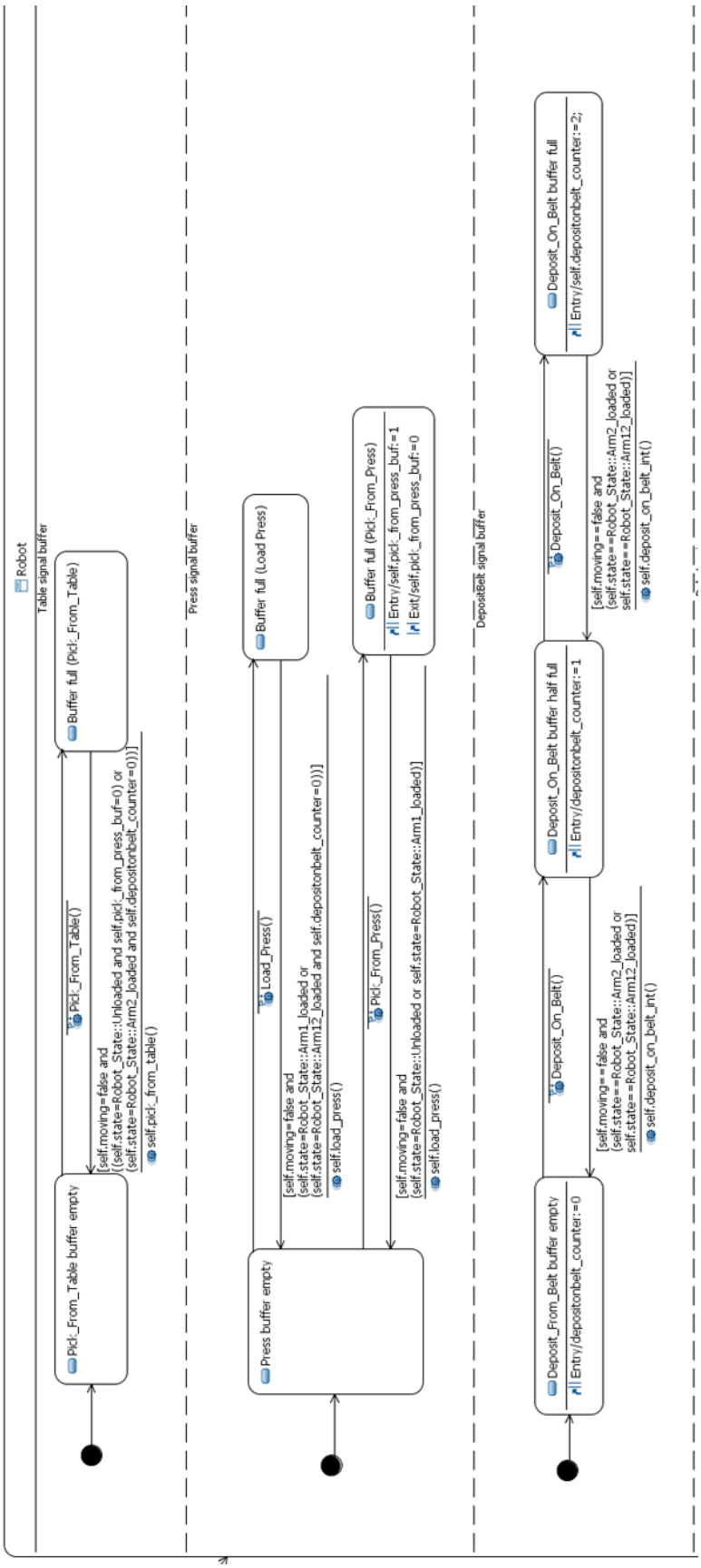


Figure 57 – Detailed view of buffer regions from stm [Block] Robot [Operating]. Whole diagram in Figure 56.

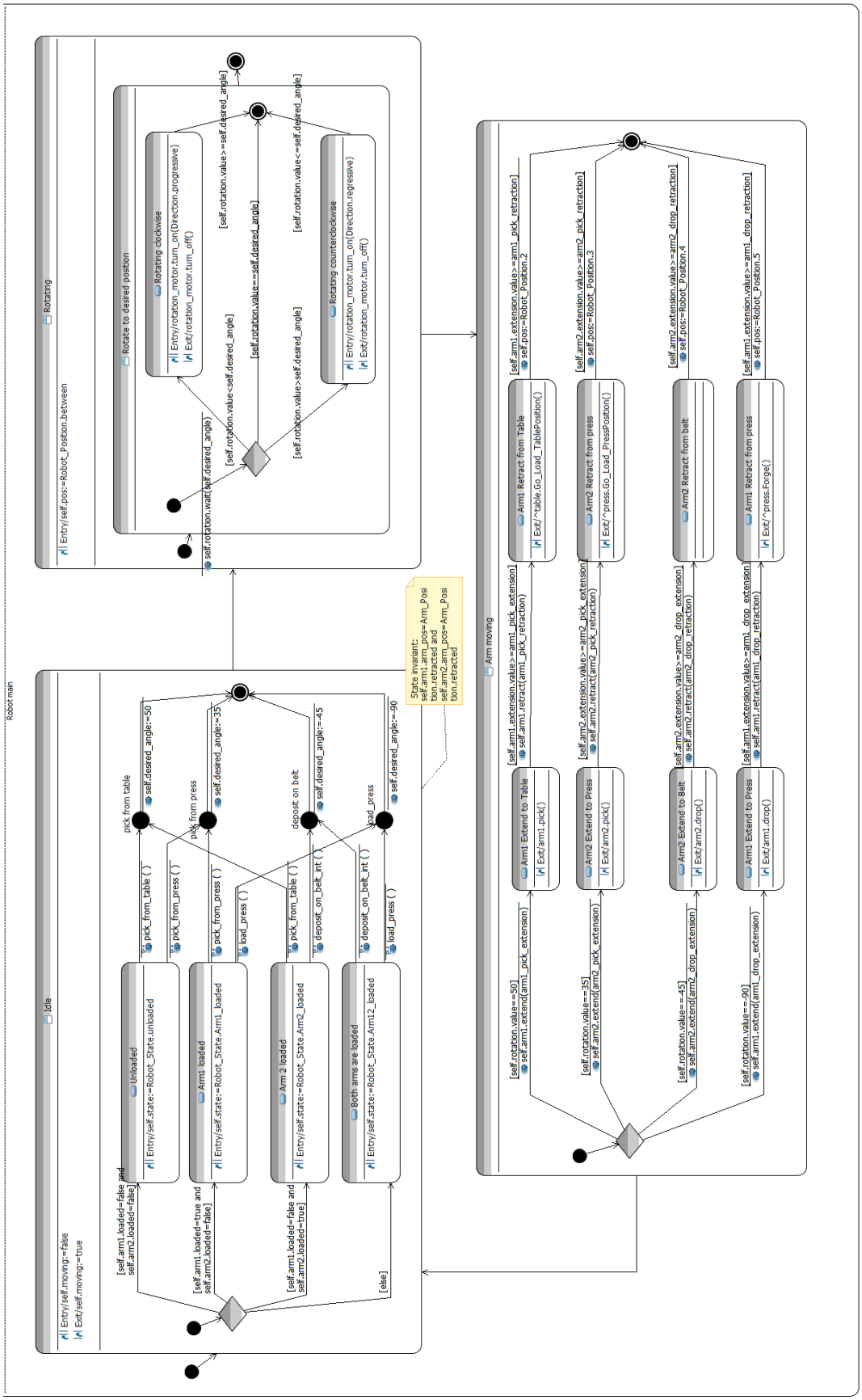


Figure 58 – Detailed view of main region from stm [Block] Robot [Operating]. Whole diagram in Figure 56.

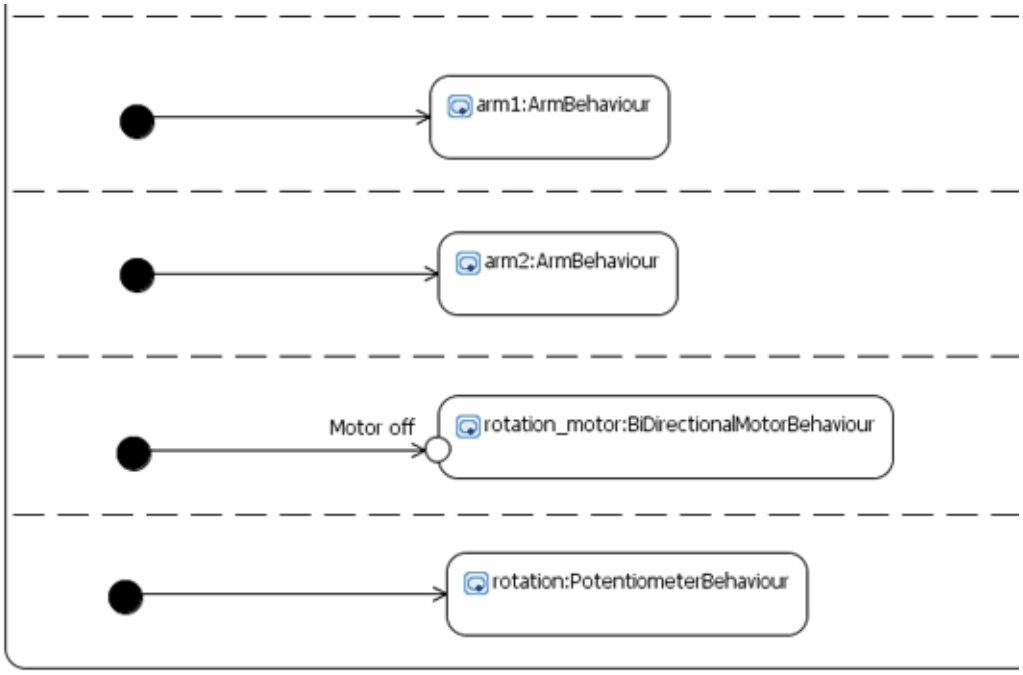


Figure 59 - Detailed view of part regions from stm [Block] Robot [Operating]. Whole diagram in Figure 53.

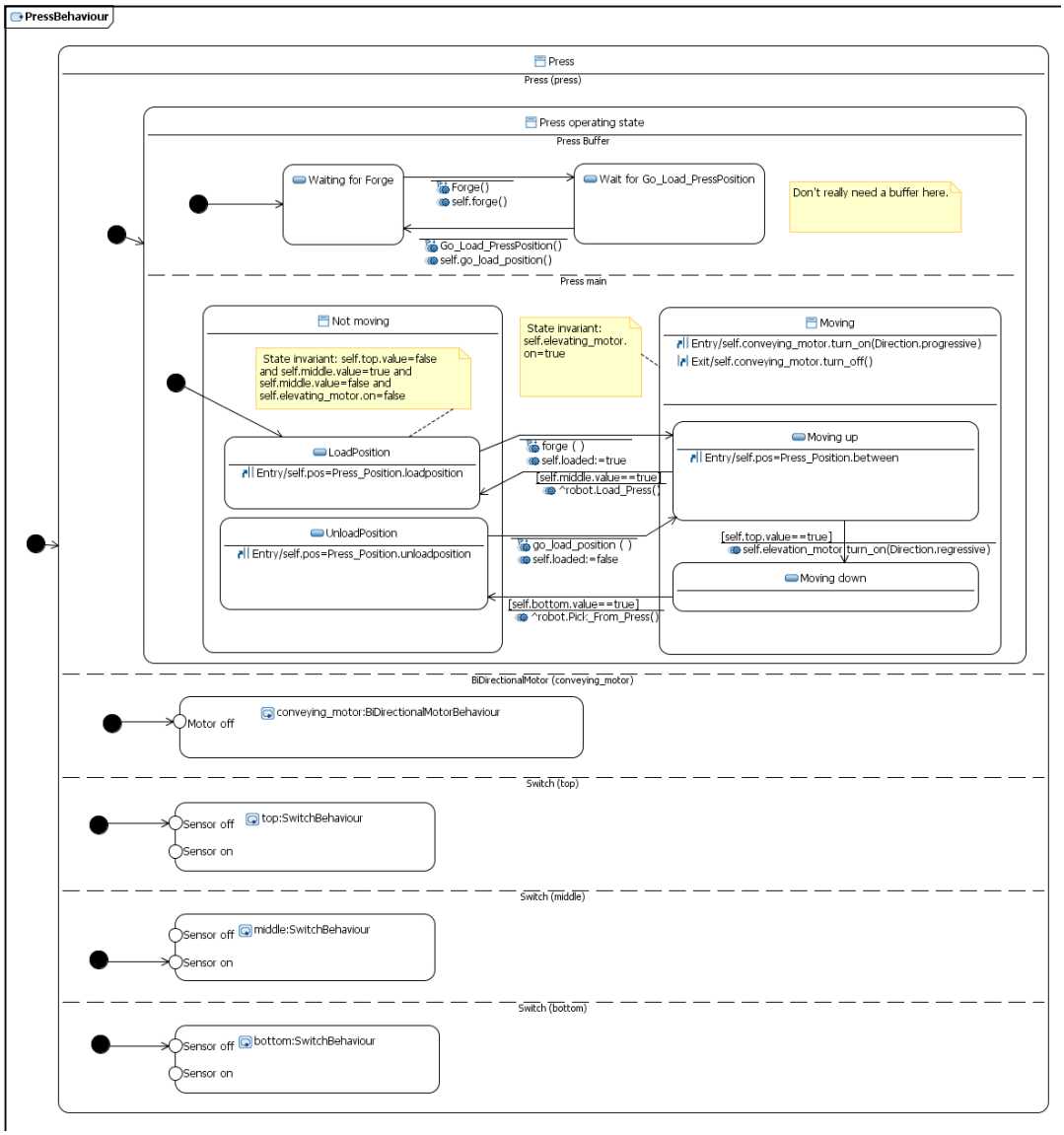


Figure 60 - stm [Block] Press [Operating]

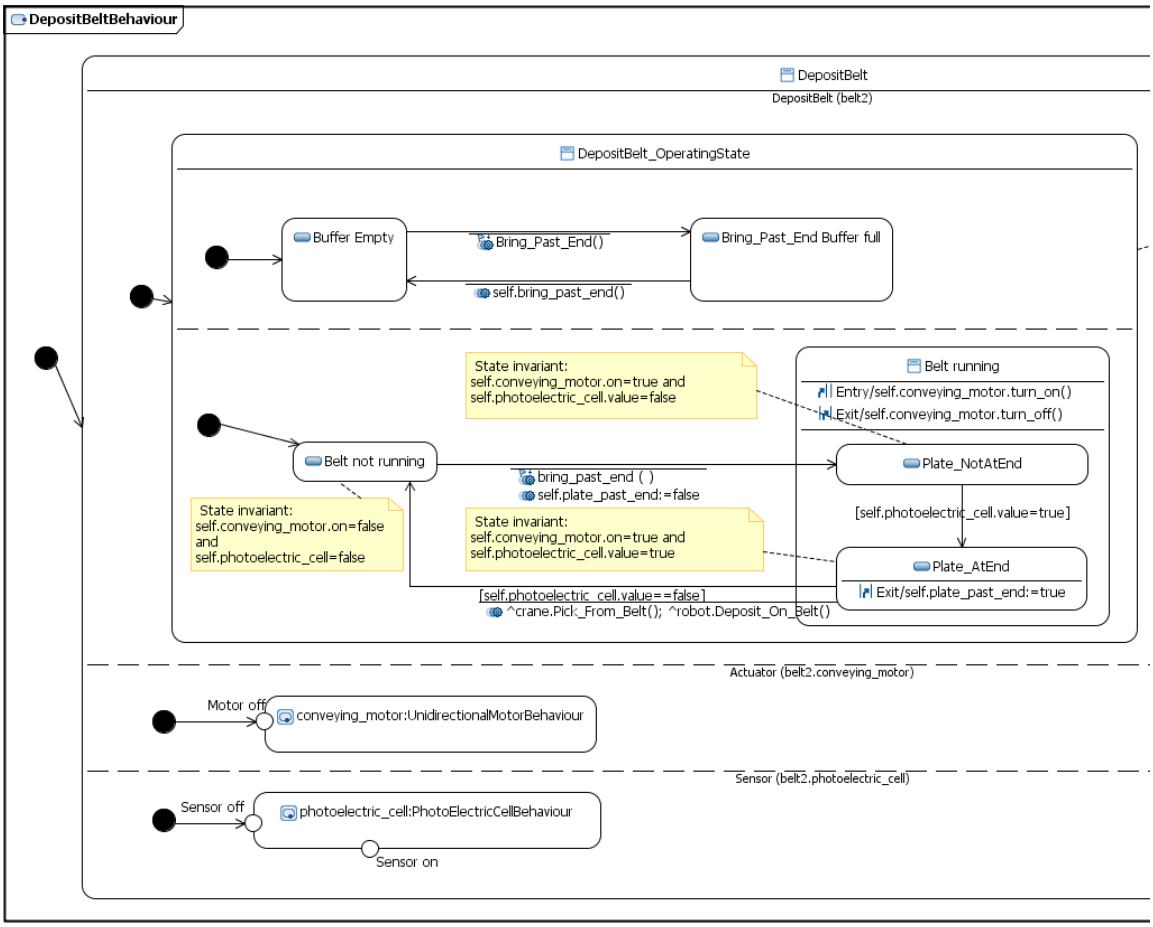


Figure 61 - stm [Block] DepositBelt [Operating]

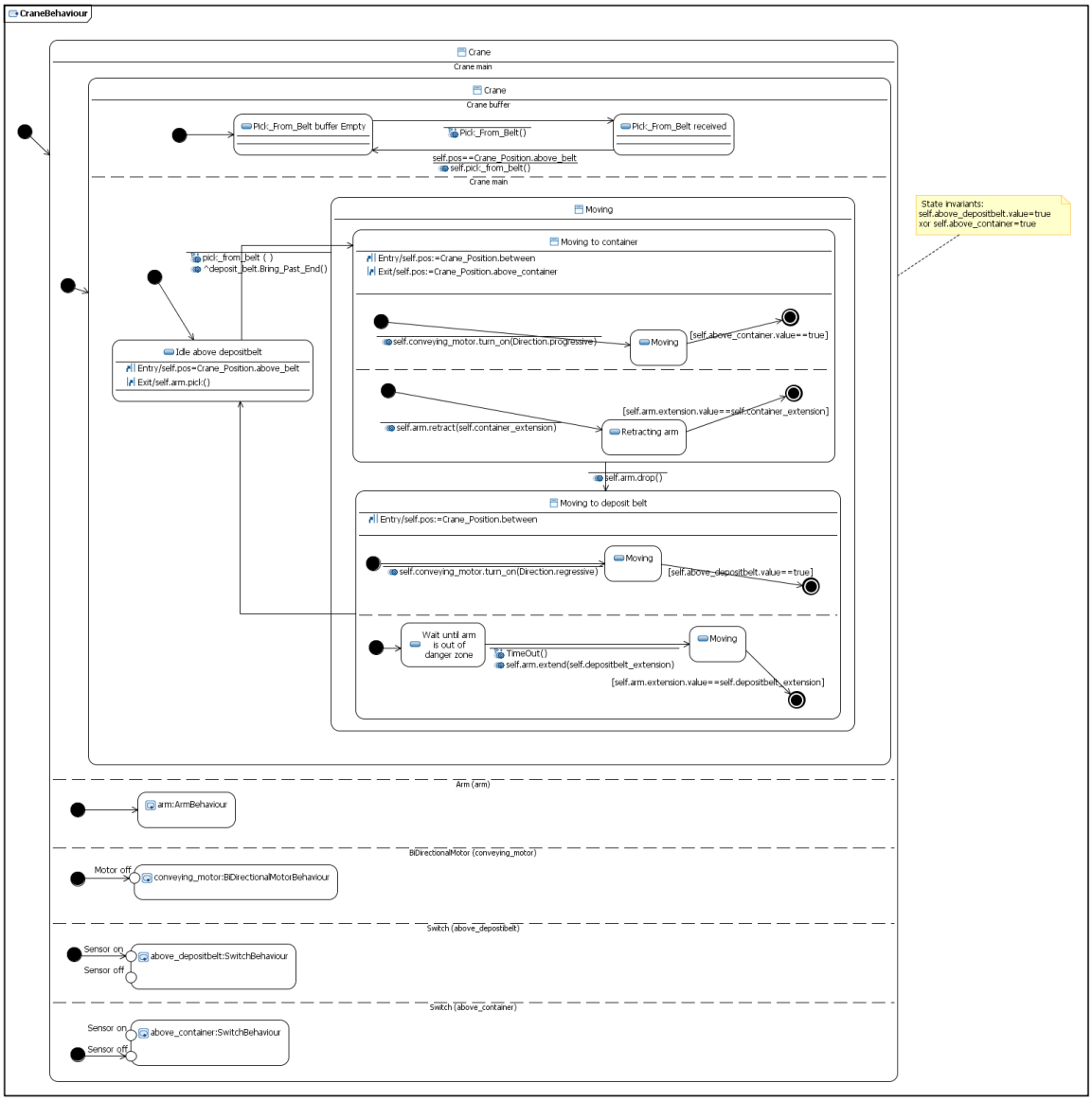


Figure 62 - stm [Block] Crane [Operating], for details, see Figure 63 and Figure 64.

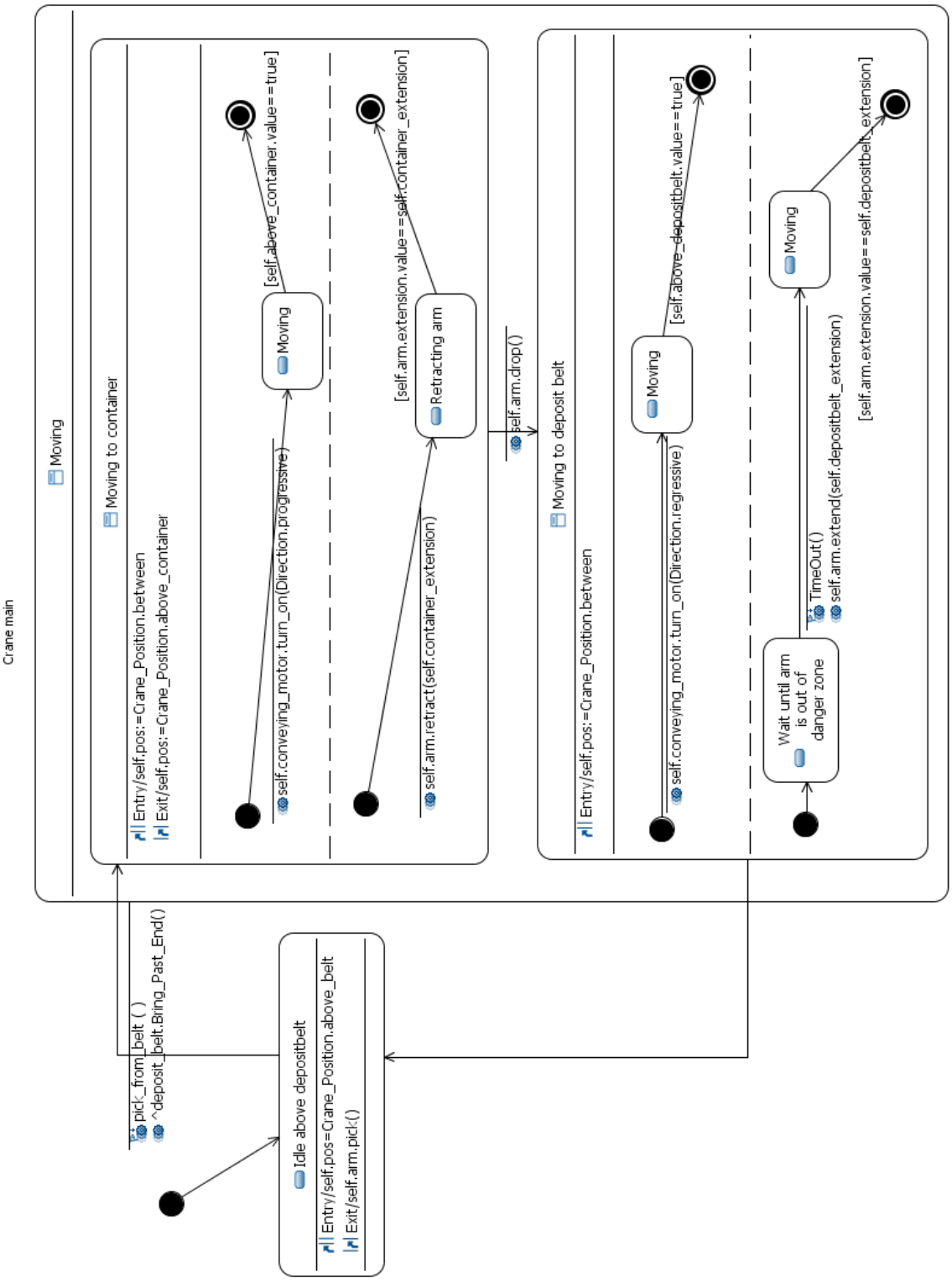


Figure 63 – Detailed view of Crane buffer and main regions from state machine diagram in Figure 62.

Figure 64 - Detailed view of Crane part regions from state machine diagram in Figure 62.

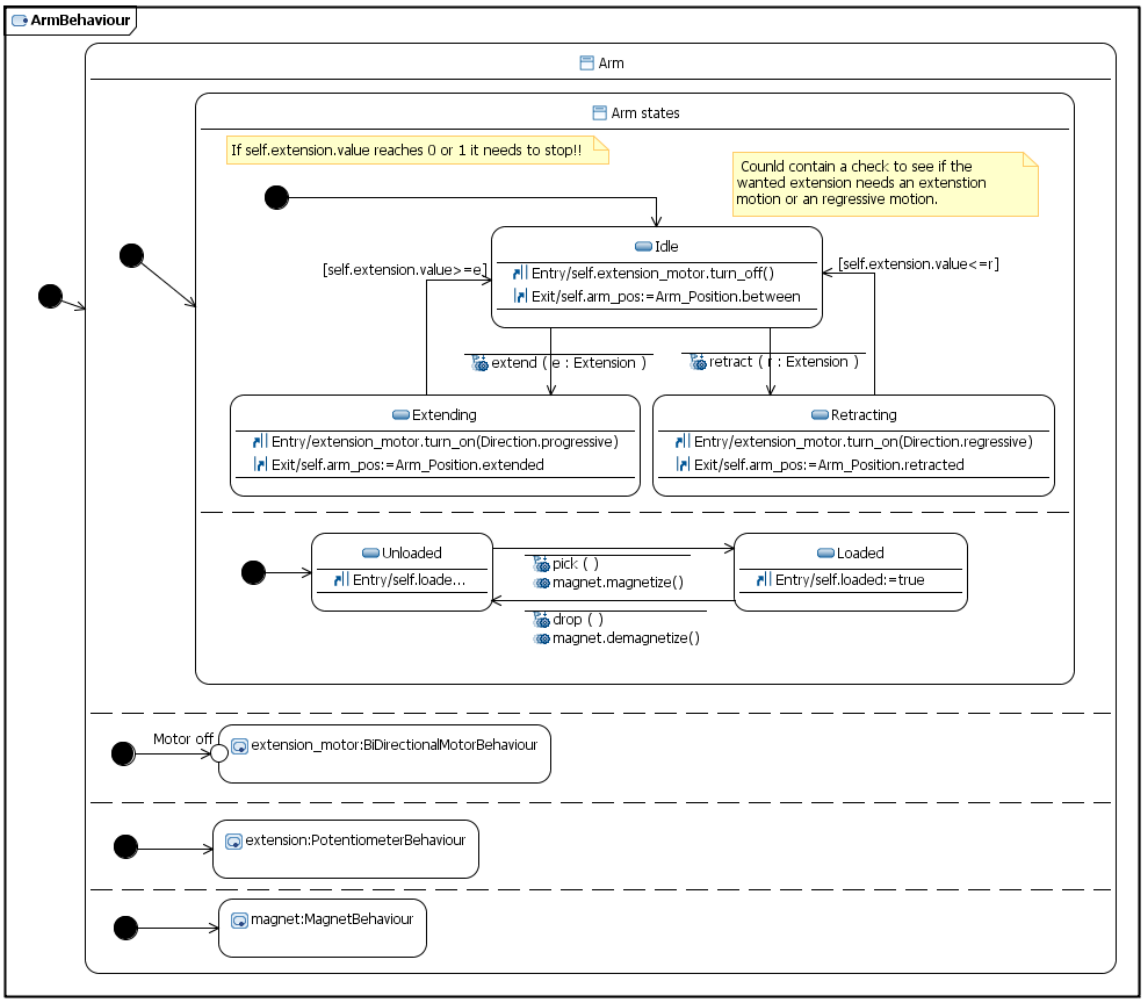


Figure 65 - stm [Block] Arm [Operating]

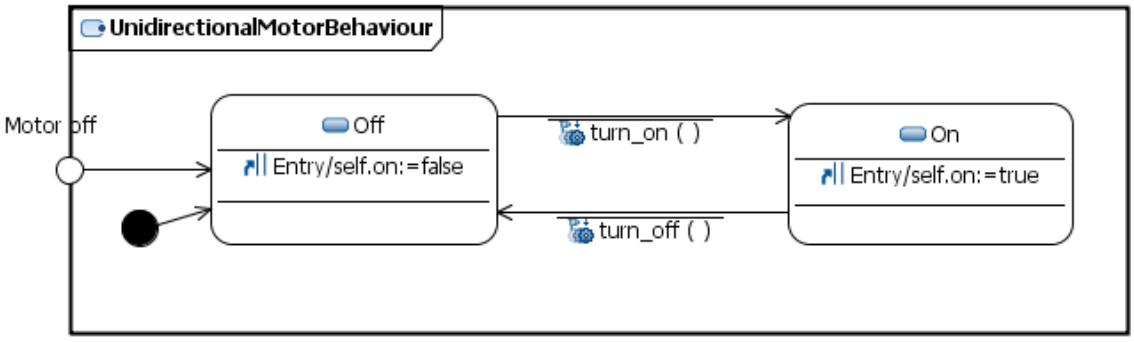


Figure 66 - stm [Block] UniDirectionalMotor [Behaviour]

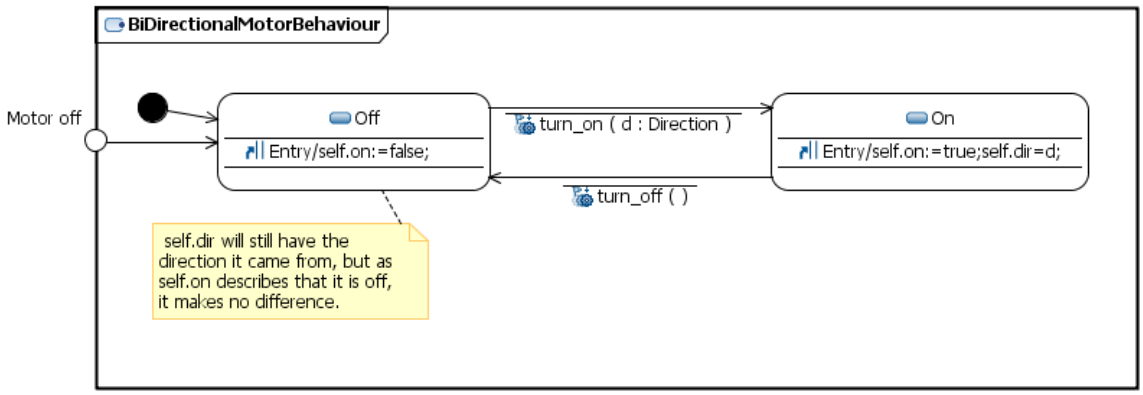


Figure 67 - stm [Block] BiDirectionalMotor [Behaviour]

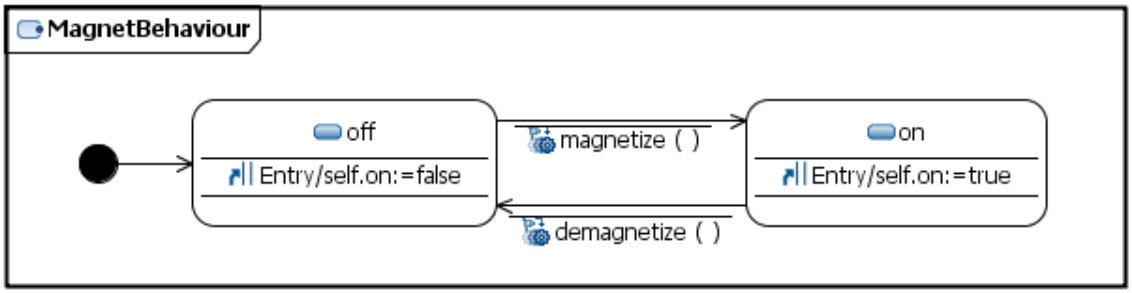


Figure 68 - stm [Block] Magnet [Behaviour]

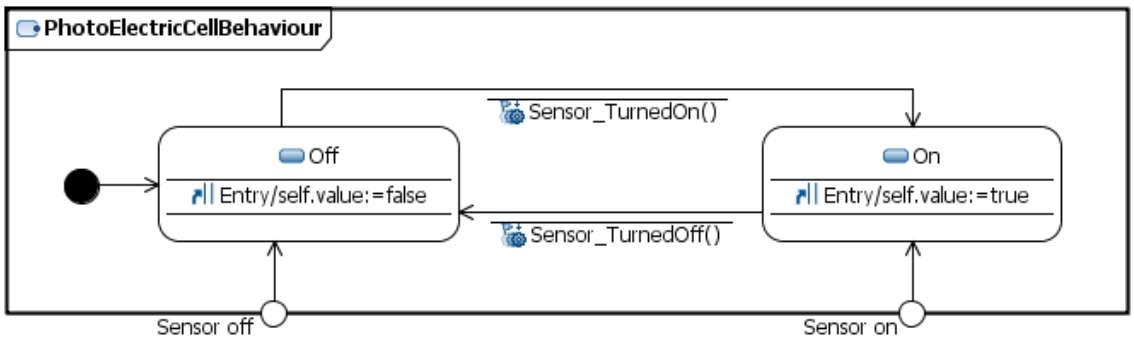


Figure 69 - stm [Block] PhotoElectricCell [Behaviour]

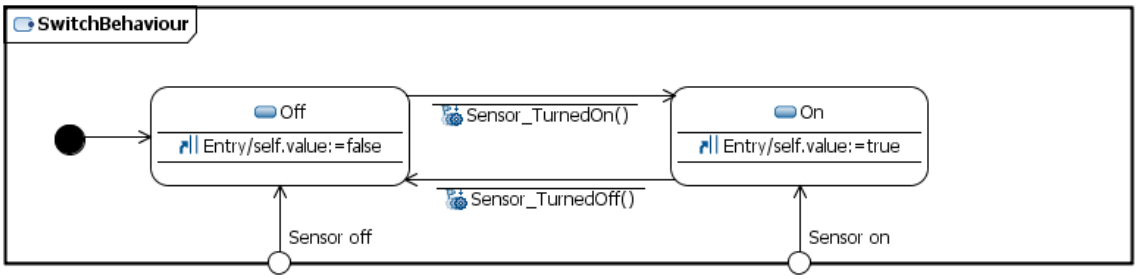


Figure 70 - stm [Block] Switch [Behaviour]

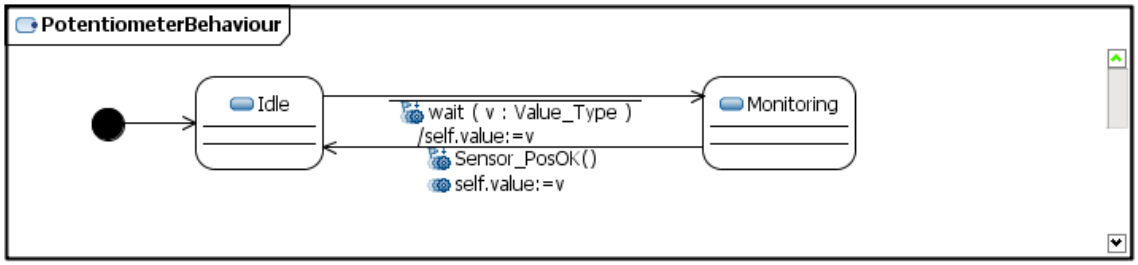


Figure 71 - stm [Block] Potentiometer [Behaviour]

A.8 Parametric diagram

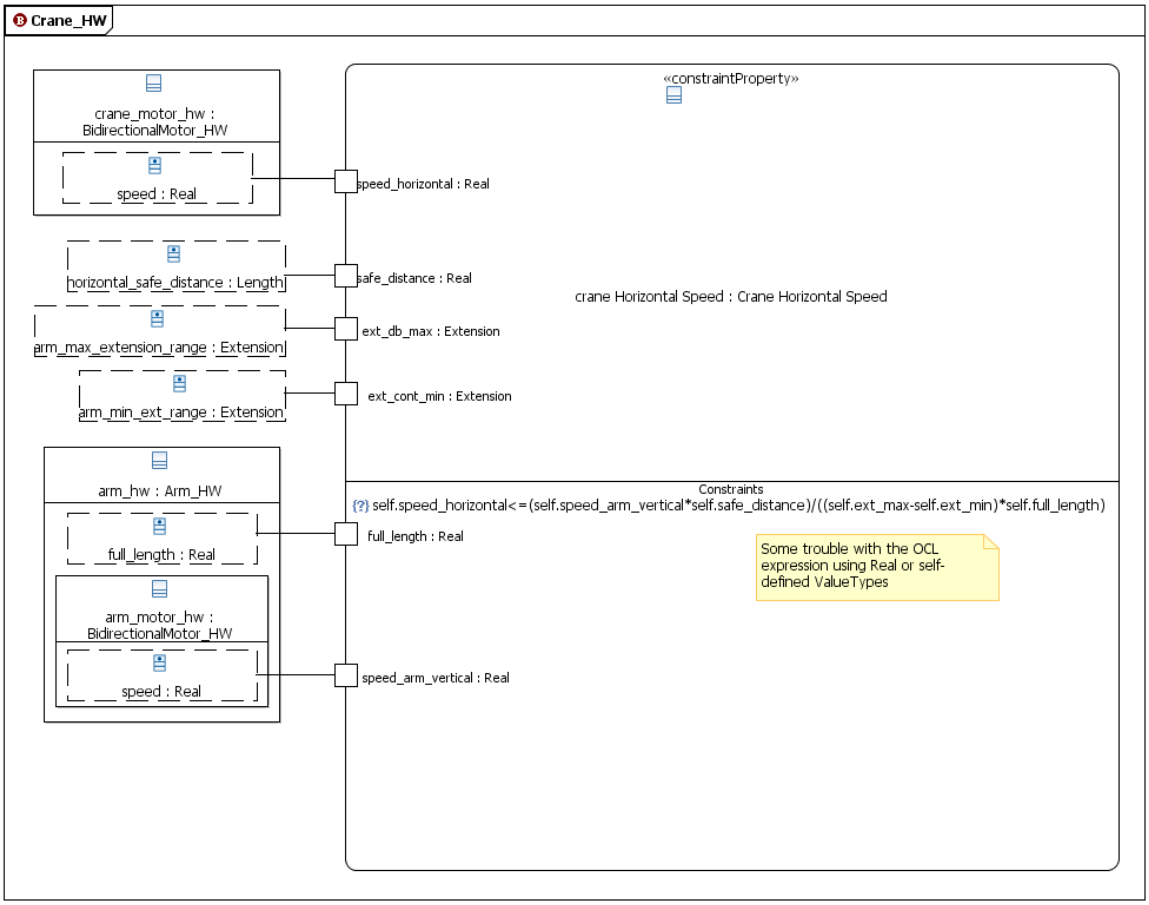


Figure 72 - par [Block] Crane_HW [Crane Horizontal Speed]

A.8.1

A.9 Cross-cutting diagrams

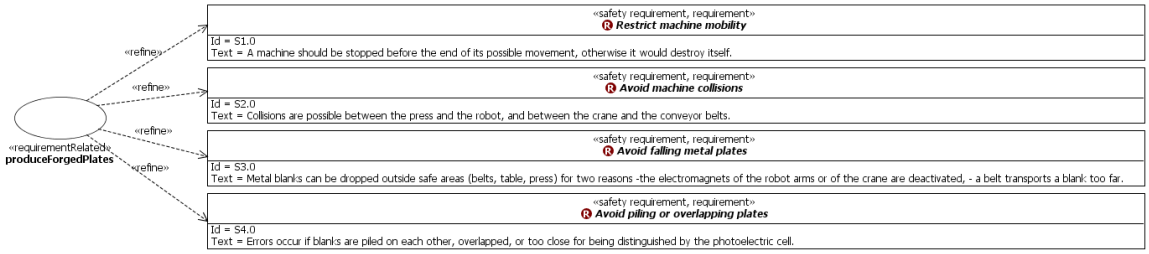


Figure 73 - High level requirements refined from use case.

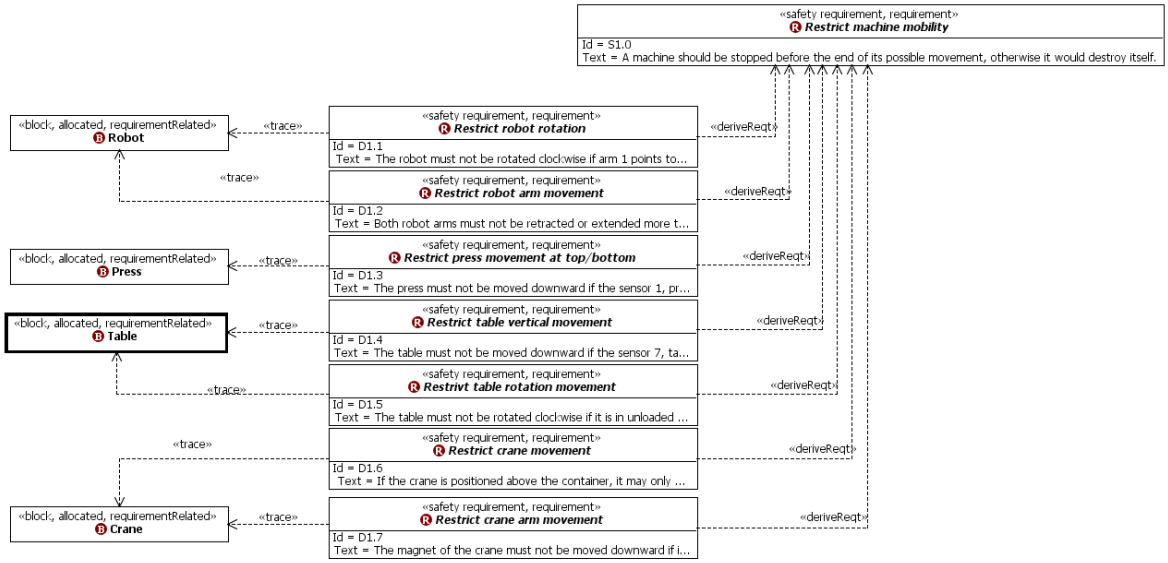


Figure 74 - Requirement S1.0 with derived requirements and traces to top-level blocks.

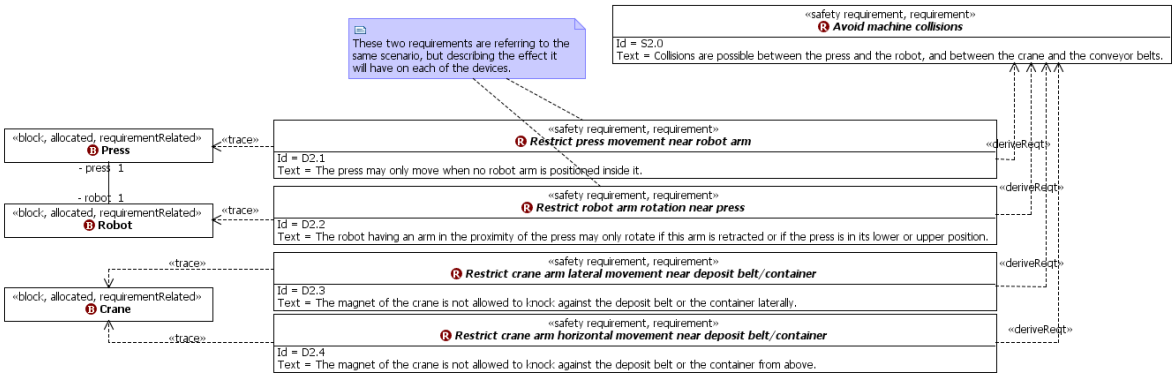


Figure 75 - Requirement S2.0 with derived requirements and traces to top-level blocks.

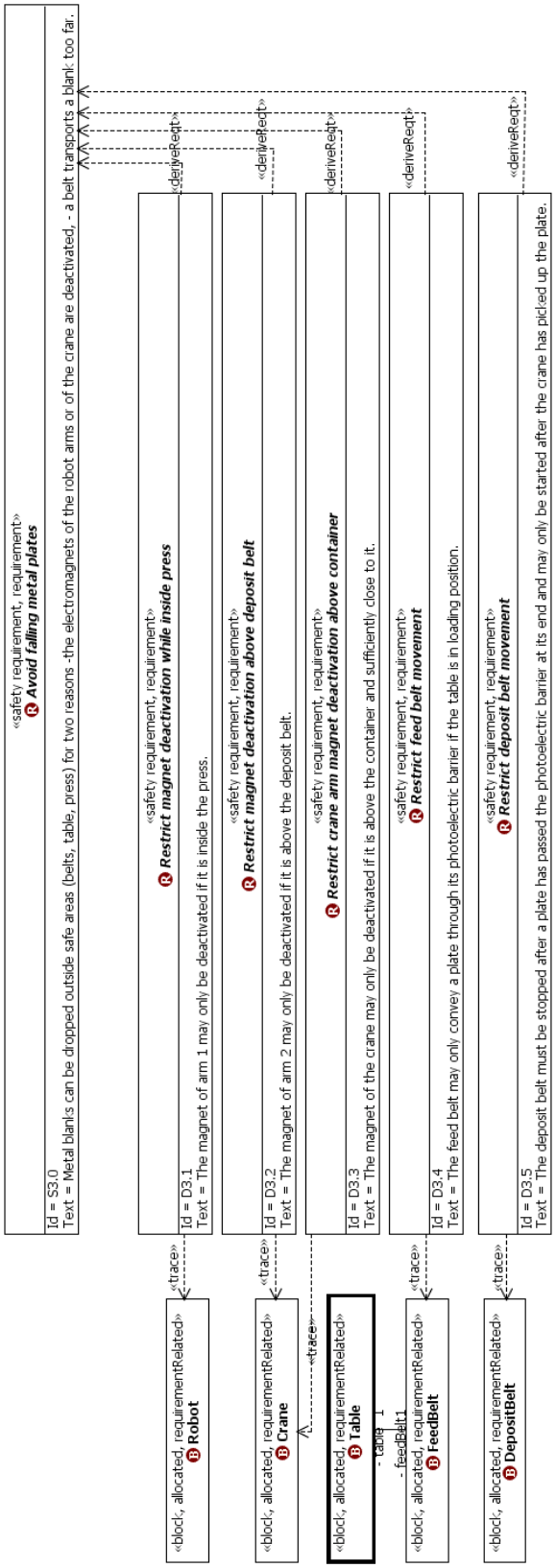


Figure 76 - Requirement S3.0 with derived requirements and traces to top-level blocks.

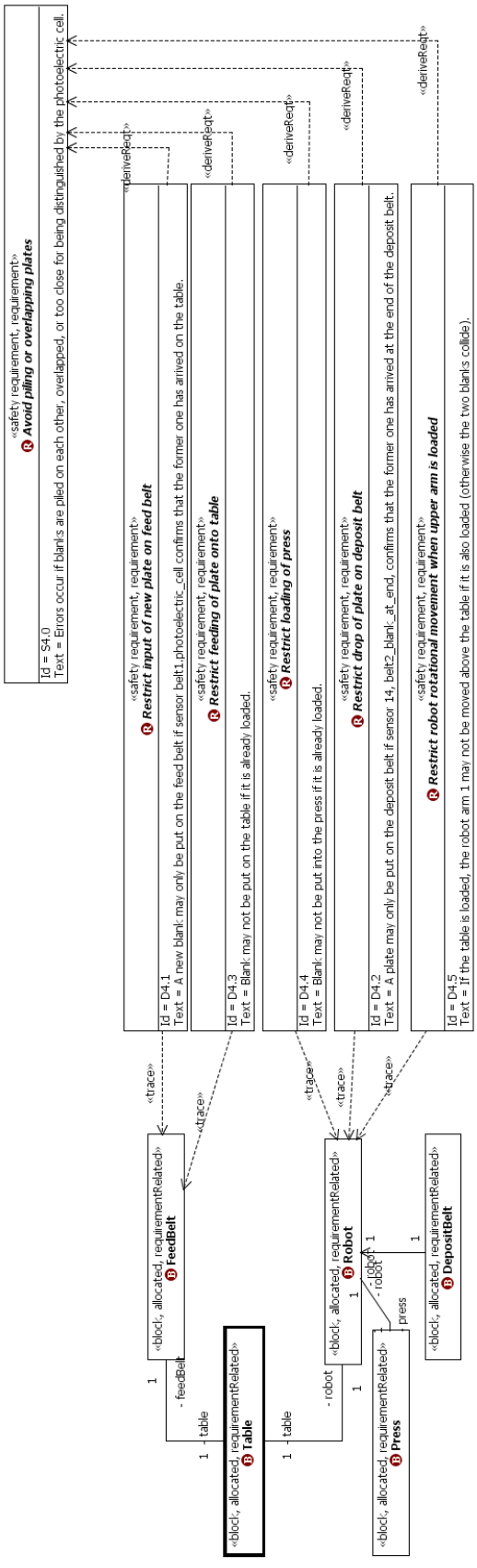


Figure 77 - Requirement S4.0 with derived requirements and traces to top-level blocks.

Appendix B Glossary

Software block	Software blocks in this model represent the physical part of the same name and contains the control logic needed to control the that part.			
Hardware block	Hardware block represent the physical part. Only the high level hardware blocks are represented in this model.			

Block	Description	Part of	Specialization of	Contains
Actuator	***Superclass*** *generalizing the software blocks which represent the hardware parts which can be activated/deactivated by the control system.*	----	----	
Arm	**Software block** in the control system which contains the logic to control the physical extendable arm of the containing machine and enables the machine to reach nearby machines at different distances to pick up/drop metal plate.	Robot, Crane	----	Potentiometer, Magnet, BidirectionalMotor
BidirectionalMotor	**Software block** in the control system which contains the logic needed to activate/deactivate the physical motor (actuator) which can move in both progressive or regressive direction.	Table, Robot, Press, Crane, Arm	ElectricMotor	
Container	**Software block** representing the final destination for the forged metal plates. No control logic.	----	----	
Crane	**Software block** in the control system which contains the logic needed to control the physical crane and communicate with the surrounding machines..	ProductionCell_Controller	----	BidirectionalMotor, 2 Switch, Arm
Crane_HW	**Hardware block** representing the physical crane.	ProductionCell_HW	----	
DepositBelt	**Software block** in the control system which contains the logic needed to control the physical deposit belt and communicate with the surrounding machines.	ProductionCell_Controller	----	UnidirectionalMotor, PhotoelectricCell
DepositBelt_HW	**Hardware block** representing the physical deposit belt.	ProductionCell_HW	----	

ElectricMotor	***Superclass*** *generalizing the software blocks which represent two types of physical motors (actuators) which can be activated/deactivated by the control system.*	----	Actuator	
FeedBelt	**Software block** in the control system which contains the logic needed to control the physical feed belt and communicate with the surrounding machines.	ProductionCell_Controller	----	UnidirectionalMotor, PhotoelectricCell
FeedBelt_HW	**Hardware block** representing the physical feed belt.	ProductionCell_HW	----	
Magnet	**Software block** in the control system which contains the logic needed to activate/deactivate the physical magnet (actuator) which is placed at the end of each arm.	Arm	Actuator	
Operator	**Block** used to describe a human in a block diagram. In this model it is included to be able to describe the human in the domain's internal block diagram. No control logic.	ProductionCell_Domain	----	
PhotoelectricCell	**Software block** in the control system which contains the logic needed to react to the input from the physical photoelectric cell (sensor).	FeedBelt, DepositBelt	Sensor	
Potentiometer	**Software block** in the control system which contains the logic needed to react to the input from the physical potentiometer (sensor) which describes either the extension range or the angle of a machine or part.	Robot, Arm	Sensor	
Press	**Software block** in the control system which contains the logic needed to control the physical press and communicate with the surrounding machines.	ProductionCell_Controller	----	
Press_HW	**Hardware block** representing the physical press.	ProductionCell_HW	----	
ProductionCell_Controller	The software top level block.	ProductionCell_Domain	----	
ProductionCell_Domain	This is the top level block in this model. It describes the whole domain of the system, which contains the system of interest (both software and hardware), the operator, the plates that will be/are forged and the container.	----	----	ProductionCell_System, Operator, Plate, Container
ProductionCell_HW	The top level hardware block this represents the hardware as a whole.	ProductionCell_Domain	----	
ProductionCell_System	Block that represents the system of interest, both hardware and softare blocks.	ProductionCell_Domain	----	ProductionCell_Controller, ProductionCell_HW

Robot	**Software block** in the control system which contains the logic needed to control the physical robot and communicate with the surrounding machines.	ProductionCell_Controller	----	
Robot_HW	**Hardware block** representing the physical robot.	ProductionCell_HW	----	
Sensor	***Superclass*** *generalizing the software blocks that represent the hardware parts which register change in the environment and provide input to the control system.*	----	----	
Switch	**Software block** in the control system which contains the logic needed to react to the input from the physical switch (sensor) which describes the position of parts of a machine.	Table, Press, Crane	Sensor	
Table	**Software block** in the control system which contains the logic needed to control the physical rotary table and communicate with the surrounding machines.	ProductionCell_Controller	----	2 BidirectionalMotor, 2 switches, Potentiometer
Table_HW	**Hardware block** representing the physical rotary table.	ProductionCell_HW	----	
UI	**Block** representing the user interface. This is not designed into further detail.	----	----	
UnidirectionalMotor	**Software block** in the control system which contains the logic needed to activate/deactivate the physical motor (actuator) which can move only in progressive direction.	FeedBelt, DepositBelt	ElectricMotor	

Appendix C Tool problems

Problem	*Description*	*Answer/(temporary) solution*
Signals (send in state machine)	**Conformance (optional):** Graphical notation for SendSignal is not available in state machines.	It is not required, but could contribute to make a simpler and more consistent diagram.
Sequence diagram	**Conformance:** Not possible to decompose lifelines, or use *extra global* c*ombined fragments* [14] .	A combined fragment can't go outside the diagram frame to include an incoming signal to a part. (e.g. include the first Add_Blank signal in a combined fragment in
Error messages	**General:** Some of the error messages during validation are very subtle and give less than little explanation for where the problematic elements are. Often occurs with association errors.	
Association problems	**General:** It may seem that some associations are left in the model after an associated block or node has been deleted from the model. Unsure when this happens.	Errors often have occurred on associations which are not connected to anything in the model.
Using existing activities in state machines	**Tool:** If we wish to use a previously modelled activity in a state machine, the whole activity is moved to/inside the state machine, without updating references in the activity's containing activity diagram where it is/may be used.	If the activity is referenced in another activity diagram, this will make the containing activity diagram incomplete.
Signals (send activity)	**Tool:** When adding a *SendSignalAction* in an activity diagram, the tool asks us to either create a Signal or select an existing element and then it automatically inserts the chosen signal in the parameters *signal* and the *type* of the automatically created *target* input pin.	Use the SendSignal action's properties tab to set all properties correct.
Messy diagrams	**Graphical:** Long qualified names and stereotypes are default in the program, and tends to clutter the diagram.	Haven't looked much into how to change the default settings, but I have tried, without finding other solutions than to select each of the elements I want to change and changing their appearance.
Control/Object flow	**Graphical:** Unable to change the default look of the	This is not a requirement, but would be a nice feature to be able to

	control and object flow to distinguish them from each other in an activity diagram.	distinguish flows in a large diagram. Solution: Select manually all connectors of one type and change colour of the lines. Not possible to make them dashed.
Activity parameter node	**Graphical:** The node is visually the same as a normal port in the diagram while the specification examples use a large rectangle node. Can make the instant understanding of the diagram more difficult.	Not an important flaw. Most for people without experience?
Activity	**Graphical:** Only way of making an activity is by making an activity diagram in the model.	The specification describes an activity as a normal block, but it is not possible to create the activity block without creating the diagram.
Item flow	**Graphical:** Deleting an item flow in an IBD diagram does not give an immediate effect visually even though they are deleted from the model.	Close and open the diagram and the item flows are gone.
Flow port	**Graphical:** Flow ports that are placed on the boundary of the containing block in an IBD will disappear when closing and opening the diagram. They are created in the model, however.	By creating an IBD for the containing block (one level above) of the block that needs the boundary port, the port will stick to the boundary.
Activity diagram and Accept Event.	**Graphical (minor detail):** When choosing a trigger event, the name isn't set.	By changing the name as in the state machine trigger events, the name would be sure to be in synch with the event. This could avoid human errors during design if just changing the name and not the triggered event. (which will make it look right, but be an error in the design of the model.)
Flow ports	**Graphical:** When typing a flow port with a flow specification, this is registered by the tool as an interface and therefore the both the port and the conjugated port is marked with a lollipop symbol (if the notation is correct, this is correct). See Figure 11 for an example.	SysML specification uses lollipop notation only for standard ports.
Activity partition	**Graphical:** The "swimlanes"	

	AllocateActivityPartitions can't be rearranged among themselves.	
IBD	**Graphical:** Troublesome to make nice graphical diagrams when opening the part compartments of parts of the main block.	When moving the bottom-most part in the ibd, this movement causes the whole containing part and its contained parts to move inside the diagram. By moving it far enough down in the diagram, the diagram keeps still until the bottom part is moved again..
Activity/Action parameter – specifying values	**User/tool:** Possible to type the parameter node, but not to specify the value.	May be user problem, not finding out how to type an input for an activity parameter node.
Activity diagram – Interruptible region	**User/Tool:** Lack of interruptible region in the activity diagram.	There are some menu choices regarding interruptible region, but we were unable to find where to find the element to draw the region.
Different configurations	**User/Tool:** [8] describes an initial values compartment to be able to define different values to different block configurations. We have not found this feature.	Tried to look for this regarding making different configurations for e.g. the different configurations for the arm.
IBD	**User/Tool:** When creating several IBDs to cover different aspects of the model, it may be easy to delete a part or port that is in use in another diagram. We get a warning, but no change to cancel. Some of the times this has happened, a delete has been impossible to undo.	