3rd Home Assignment – Behavior Modeling

Autonomous Vehicle - Boarding Equipment

After a revision of the *Automated passenger counter (APC)* project, system architects (in accordance with the customer) have chosen to drop the current architecture in favor of a simpler solution that can be built from inexpensive components already available on the market.

The new plan is the following. The counting and identification of passengers will be realized by a *turnstile* and an *RFID card scanner*. To signal the end of the boarding phase and potential errors related to authentication and authorization (whether a passenger is allowed on board or not), an *alarm* component with light and sound signals will be installed next to the *door*.

These components will be shipped by subcontractors – the task of the team is to design the *boarding controller* unit that will be responsible for the orchestration of the components according to a process specified in Appendix 2.

The ports and interfaces of the components, the *controller* and the whole *boarding equipment* are already defined in package *IntelliBus System/4 - Functional Architecture/*: signals in *4.4 Signals/Boarding Equipment*, data types in *4.4 Signals/Enums and types*, interfaces in *4.3 Interfaces/Boarding Equipment* and ports under the blocks representing the components in *4.1 Functional decomposition/Boarding Equipment*. The arrangement of components is specified in the IBD under *Boarding Equipment* in package *4.1 Functional decomposition/Boarding Equipment*.

The behavior of subcomponents is specified in different ways.

- The RFID card scanner has a textual user manual summarized in Appendix 1.
- The Turnstile came with a State Machine Diagram that precisely specifies its behavior. You can find the diagram under the block *IntelliBus System/4 - Functional Architecture/4.1 Functional decomposition/Boarding Equipment/Turnstile* (*TurnstileBehavior*).
- The alarm is very simple, it can be turned on and off with the corresponding signals and does not give any feedback.
- The behavior of the door is specified only in (detailed) scenarios (Sequence Diagrams) found under the block *IntelliBus System/4 - Functional Architecture/4.1 Functional decomposition/Boarding Equipment/-Door* (*DoorOpens*, *DoorCloses*, *IntendedOperation* and *PossibleOutputs*).

Tasks

a. Create a State Machine model of the behavior of the door in the empty State Machine Diagram *IntelliBus System/4 - Functional Architecture/4.1 Functional decomposition/Boarding Equipment/Door/Door-Behavior/DoorBehavior*. Make sure the state machine conforms to the behavior specified on the Sequence Diagrams.

b. Based on the RFID Scanner User Manual (Appendix 1), model the behavior of the scanner in the prepared Activity Diagram *IntelliBus System/4 - Functional Architecture/4.1 Functional decomposition/Boarding Equipment/RFID Scanner/ScannerBehavior/ScannerBehavior*. Model the control and data flow (using types in *4.4 Signals/Enums and types*), signal receptions and sending (with source and target ports), as well as the timeout and forced stop behavior mentioned in the user manual (with Interruptible Activity Regions). Do not forget to mark interrupting edges and that Accept Event Actions *without input control flows* remain active until the enclosing Activity or Region is active.

c. Design and model the behavior of the *boarding controller* according to the specification in Appendix 2 and the behavior of the controlled components. Make sure not to violate critical requirements (*violating designs will be invalidated*). Do not forget to handle every possible behavior of the components. Also note that component states are not observable, so the controller may rely only on the signals received from the components. You may use English text in the guard conditions (if any). Modeling Actions with Activities is not mandatory as long as the name of the *OpaqueBehavior* explicitly specifies the type and target of the signal to send.

d. *(for extra IMSc credits) Illustrate the behavior of the* boarding equipment *with at least two scenarios modeled with Sequence Diagrams under the block* IntelliBus System/4 - Functional Architecture/4.1 Functional decomposition/Boarding Equipment/Boarding Equipment. *At least one of them should describe a scenario in which boarding completes successfully and at least two passengers board the bus, and at least one should illustrate a case when complications result in an error message to the maintainers.*

Appendix

1 RFID Scanner User Manual

Once turned on, the scanner initializes and then starts listening for *Start* signals that should be sent whenever the scanner is expected to read an RFID. Upon receiving such a signal, listening will be suspended and reading the RFID data starts. When reading is completed, the raw data is processed. The resulting data package is then sent through the scanner port in a *SuccessfulScan* signal and the scanner resumes listening to *Start* signals. If a *Stop* signal (requesting the cancellation of the scanning) is received at any time during reading, processing or sending, the scanner immediately stops the scanning subroutine and resumes listening to *Start* signals. The same thing happens if the scanning subroutine fails to complete in 5 seconds, with the addition that a *Timeout* signal is sent through the scanner port to notify the client about the failed scanning attempt.

2 Boarding Process and Requirements

The boarding process has multiple phases.

1. *Waiting for boarding*: Boarding starts when the vehicle is fully stopped, which is signaled by a *StartBoarding* signal.
2. *Opening door*: The controller should open the door of the vehicle.
3. *Authorization*: Once the door is open, the RFID scanner should be used to identify passengers. The scanner should be active until a valid scan or until the boarding process starts finishing. If the scanned data indicates that the passenger is *unauthorized* to board the vehicle, the alarm should be turned on for 500ms (no scanning should occur during this time). If the passenger is *authorized*, the *authorization phase* completes.
4. *Passenger boarding*: The controller should unlock the turnstile and wait for the authorized passenger to board. Once this happens, or the passenger fails to board in 5 seconds, the turnstile should be locked. If this is not possible, the controller should keep trying in every second. Upon successful locking, the process repeats from the *authorization phase.*
5. *Finishing*: The boarding process must start finishing when the *FinishBoarding* signal arrives. The controller has 10 seconds to finalize every pending operation and close the doors. If this is successful, it should send an *OK* signal to acknowledge the completion of the boarding process and return to the first phase (waiting for boarding). **If it fails to close the door in the 10-second window, it must not send an *OK* signal, but an *Error* signal on the appropriate port instead and shut down.** The boarding process is considered completed when *1)* the turnstile is successfully locked and *2)* the door is successfully closed. During the finishing phase, the *alarm* should be *On.*

Critical requirements toward the boarding controller[1]

- The door must not be opened before the *StartBoarding* signal arrives.
- The door must be closed when the controller sends the *OK* signal.
- Either the *OK* signal or the *Error* signal must be sent at most 10 seconds later than the arrival of the *FinishBoarding* signal.

[1] Models not satisfying these requirements are worth 0 points.