# Requirement Classification

March 29, 2022

## 1 Turnstile

The turnstile includes a coin acceptor, doors controlled by a signal (*open*). Doors block the passage and open after payment (*paid*). The notification of the passer about the possibility of passing is carried out by the LED (*enter*). The control of the opening of the doors is carried out by the sensor (*opened*). To control and prevent unauthorized entry attempts, the turnstile is equipped with two optical sensors that monitor a passer's approach to the turnstile ($PDin$) and his passing through the turnstile ($PDout$). After receiving the payment, the coin acceptor is blocked. The coin acceptor is unblocked by the rising edge of the reset signal ($reset$).

1. If the turnstile is open and the user does not pass through it for 10 seconds, the turnstile must be closed and reset coin acceptor.

$$\begin{aligned}\forall s_1.\ (&toEnvP(s) \land substate(s_1, s) \land toEnvP(s_1) \land \\ &toEnvNum(s_1, s_3) > 10s \land getVar(s_1, open) = true \Rightarrow \\ &\exists s_3.\ toEnvP(s_3) \land substate(s_1, s_3, s) \land \\ &toEnvNum(s_1, s_3) < 10s \land getVar(s_3, open) = false \\ &\land getVar(s_3, reset) = true \lor \\ &getVar(s_3, PDout) = DETECT \land \\ &\forall s_2.\ (toEnvP(s_2) \land substate(s_1, s_2, s_3) \land s_2 \neq s_3 \Rightarrow \\ &getVar(s_2, open) = true \land \\ &getVar(s_2, PDout) = NOT\_DETECT))\end{aligned}$$

2. if the turnstile was not open and the passage is not paid for, then the turnstile will not open.

$$\begin{aligned}\forall s_1.\ (&toEnvP(s) \land substate(s_1, s_2, s) \land \\ &toEnvP(s_1) \land toEnvP(s_2) \land toEnvNum(s_1, s_2) = 1 \land \\ &getVar(s_1, open) = false \land getVar(s_2, paid) = false \\ &\Rightarrow getVar(s_2, open) = false)\end{aligned}$$

3. if the open signal has not been given, the turnstile will not be opened

$$\begin{aligned}\forall s_1.\ (&toConP(s) \land substate(s_1, s) \land \\ &toConP(s_1) \land getVar(s_1, open) = false \Rightarrow \\ &getVar(s_1, opened) = false)\end{aligned}$$

4. The turnstile opens instantly.

$$\begin{aligned}\forall s_1.\ (&toConP(s) \land substate(s_1, s) \land \\ &toConP(s_1) \land getVar(s_1, open) = true \Rightarrow \\ &getVar(s_1, opened) = true)\end{aligned}$$

## 2 Robot vacuum cleaner

The robot vacuum cleaner includes a dust container, cleaning brushes, a battery and a motor. There are on (turnOn) and off (turn off) buttons. The vacuum cleaner turns on when the on button is pressed and works for a specified time (WORK_TIME). The off button turns off the vacuum cleaner until its operation is completed. The motor is controlled by the move signal. If the robot crashes into an obstacle or gets clogged with dust, the motor turns off and a sound signal (sound) is given. Obstacles in the way of movement, the filling of the container with dust and the getting stuck of the vacuum cleaner are registered by sensors (obstacle, clogged, stuck). If the robot is stuck and cannot move for 30 seconds, the motor turns off and the sound signal is given. The battery charge is registered by the sensor (charge). The motor can only be switched on if the battery charge is at least 10%. When the battery is less than 10% charged, the robot control program should start looking for a charging base.

1. Simultaneous turn on and turn off signals are prohibited.

$$\forall s_1.\ (toConP(s) \wedge substate(s_1, s) getVar(s_1, turnOn) = false \vee getVar(s_1, turnOff) = false)$$

2. The movement of the robot vacuum cleaner is possible only if there is a battery charge of at least 10%.

$$\forall s_1.\ (toEnvP(s) \wedge substate(s_1, s) \wedge toEnvP(s_1) \wedge getVar(s_1, charge) = LOW \Rightarrow getVar(s_1, movement) = false)$$

3. The movement starts when the turn on button is pressed.

$$\forall s_1 s_2.\ (toEnvP(s) \wedge substate(s_1, s_2, s) \wedge toEnvP(s_1) \wedge toEnvP(s_2) \wedge toEnvNum(s_1, s_2) = 1 \wedge getVar(s_1, movement) = false \wedge getVar(s_2, turnOn) = true \Rightarrow getVar(s_2, movement) = true$$

4. if the robot cannot move for 30 seconds, the robot should stop and sound a signal.

$$\begin{aligned}\forall s_1 s_2.\ (&toEnvP(s) \wedge substate(s_1, s_2, s) \wedge \\ &toEnvP(s_1) \wedge toEnvP(s_2) \wedge toEnvNum(s_1, s_2) = 1 \wedge \\ &toEnvNum(s_2, s) > 30s \wedge getVar(s_1, movement) = true \\ &\wedge getVar(s_2, stuck) = true \Rightarrow \\ &\exists s_4.\ toEnvP(s_4) \wedge substate(s_2, s_4, s) \wedge \\ &toEnvNum(s_2, s_4) < 30s \\ &\wedge (getVar(s_4, movement) = false \wedge \\ &getVar(s_4, signal) = true \vee getVar(s_4, stuck) = false) \wedge \\ &\forall s_3.\ (toEnvP(s_3) \wedge substate(s_2, s_3, s_4) \wedge s_3 \neq s_4 \Rightarrow \\ &getVar(s_3, movement) = true \wedge getVar(s_3, stuck) = true))\end{aligned}$$

5. If the robot crashes into an obstacle, it should stop and sound a signal.

$$\forall s_1 s_2.\ (toEnvP(s) \wedge substate(s_1, s_2, s) \wedge toEnvP(s_1) \wedge toEnvP(s_2) \wedge toEnvNum(s_1, s_2) = 1 \wedge getVar(s_1, movement) = true \wedge getVar(s_2, obstacle) = true \Rightarrow getVar(s_2, movement) = false$$

6. If the robot is clogged with dust, then it should stop and give a sound signal.

$$\begin{aligned}\forall s_1 s_2.\ (&toEnvP(s) \wedge substate(s_1, s_2, s) \wedge \\ &toEnvP(s_1) \wedge toEnvP(s_2) \wedge toEnvNum(s_1, s_2) = 1 \wedge \\ &getVar(s_1, movement) = true \wedge getVar(s_2, clogged) = true \\ &\Rightarrow getVar(s_2, movement) = false\end{aligned}$$

7. When the battery is less than 10% charged, the robot should start looking for a charging base.

$$\begin{aligned}\forall s_1 s_2.\ (&toEnvP(s) \wedge substate(s_1, s_2, s) \wedge \\ &toEnvP(s_1) \wedge toEnvP(s_2) \wedge toEnvNum(s_1, s_2) = 1 \wedge \\ &getVar(s_1, charge) = SUFFICIENT \wedge getVar(s_2, charge) = LOW \\ &\Rightarrow getVar(s_2, search) = true\end{aligned}$$