1. Introduction

1.1. Purpose

This document describes the software requirements for a parking garage control system **(PGCS).** This specification is intended for the designer, developer and maintainer of the PGCS.

1.2. Scope

The function of the PGCS is to control and supervise the entries and exits into and out of a parking garage. The system allows or rejects entries into the parking garage dependent on the number of available parking spaces.

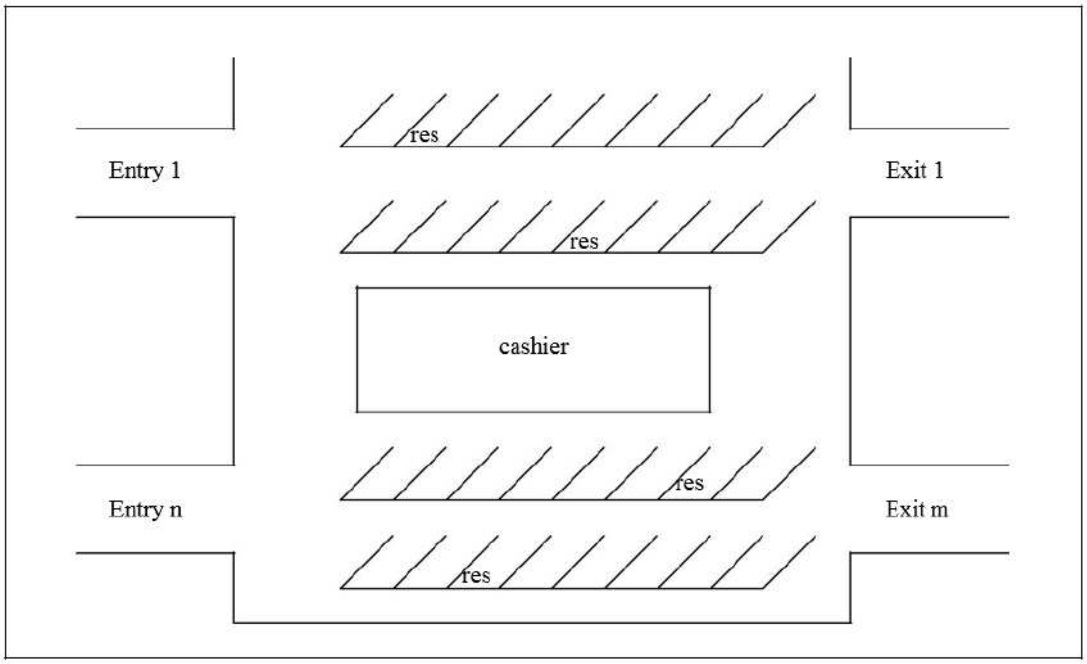
1.3. Overview

The remainder of this document is organized as follows: There will be some definitions of important terms in the next subsection. Chapter 2 contains a general description of the PGCS. Chapter 3 identifies the specific functional requirements, the external interfaces and the performance requirements of the PGCS.

1.4. Definitions

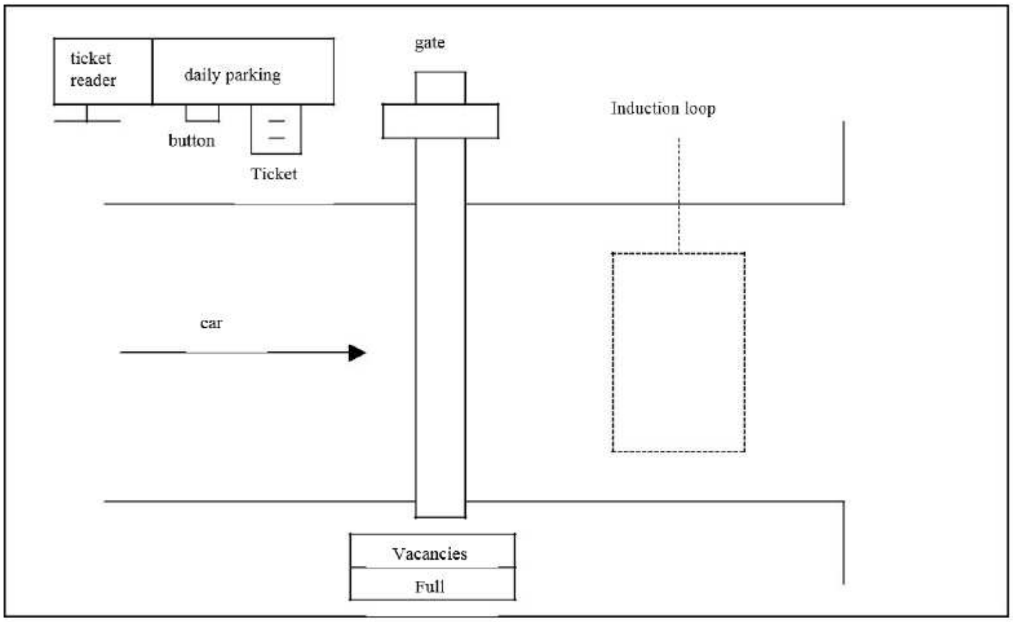
- Parking garage

  Consists of n entries and m exits. There are k parking spaces and r reserved ones. The maximum number of parking spaces is 1000.



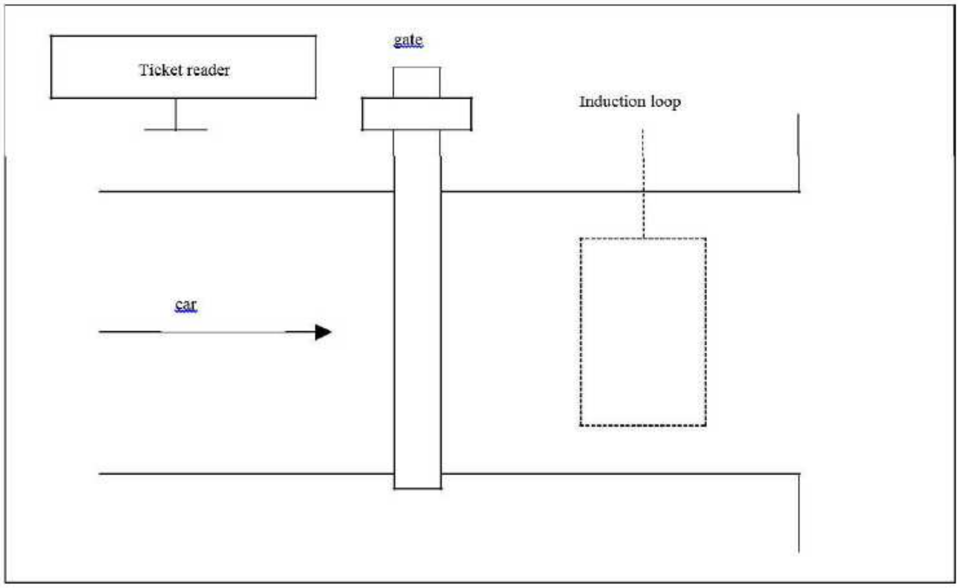
- Entrance

  An entrance consists of a gate, a state display showing whether any non-reserved parking space is available, a ticket machine with a card reader, and an induction loop. The ticket machine consists of a request button, a unit for the output of the tickets and a card reader.



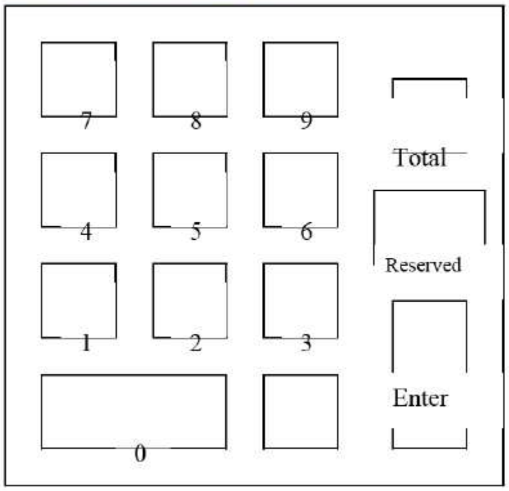
- Exit

  The exit consists of a gate, a ticket reader, and an induction loop that is behind the gate.



- Control Unit

  The control unit consists of a numerical unit.



2. General Description

2.1.Overview

To give a short overview of the functionality of the PGCS the following user scenarios are provided:

- Entry

  Driver without a reserved parking space:

  1. A driver pushes the button at the ticket machine. If the parking garage is full, nothing happens (State display is in state full)
  2. The ticket machine writes the time on the ticket and delivers the ticket to the driver. The gate will open when the driver takes the ticket.
  3. The driver enters the parking garage.
  4. After the car passes the loop, the gate is closed.
  5. The driver parks the car and leaves the parking garage.

  Driver with a reserved parking space:

  1. The driver enters his access card in the card reader of the ticket machine.
  2. The ticket machine checks if it is a valid access card.
  3. If it is a valid access card the gate opens and the driver enters the parking garage.
  4. After the car passes the loop the gate is closed.
  5. The driver parks the car in a parking space and leaves the parking garage.

Payment for drivers without a reserved parking space:

1. The driver pays the fee at the cashier
2. The cashier prints the time on the ticket after the fee is paid.

- Exit
  1. The driver returns to his car and drives to an exit station.
  2. The driver puts the ticket or access card in the ticket reader.
  3. The ticket reader checks if the fee was paid within the last 15 minutes or if the item inserted is a valid access card. If not, nothing happens. The driver has to call someone.
  4. The gate will open.
  5. After the car passes the loop, the gate is closed.

- Change occupied status
  1. In order to test and maintain the system it is possible to enter the number of occupied and reserved parking spaces with the help of a device.
  2. If the number of reservations changes (increases or decreases), the PGCS will get the new r value from the cashier.

2.2 Product Perspective

The software system is an embedded system. The characteristics of the devices will be described. The software system should control for

- each entrance:
  - a ticket machine
  - a gate
  - a card reader
  - induction loop
  - state display
- each exit:
  - a ticket reader
  - a gate
  - induction loop
- a control unit:
  - a numerical input device

In the following the abstract systems behavior will be described. The term "automatically" describes the behavior of a device that is done without control of this software system. The ticket machine will automatically print the time on the ticket if a ticket is provided. The card reader in the ticket machine automatically reads a card that is entered. The ticket reader reads a ticket or an access card. It reads the time of entering the parking garage and the time of paying the fee automatically. The induction loop is in two states: A car is present in the induction loop at that moment or not. If the state changes, a signal will be sent to the PGCS. The gates have two states: open and closed. It takes some time to change the states. The state display shows two states: full and vacancies. This is according to the number of available public parking spaces in the parking garage. (The state display is not relevant for the drivers that own an access card!) With the numerical input device there is the possibility of entering the number of occupied and reserved parking spaces. At the cashier the fee is paid. The time of payment is automatically printed on the ticket. The cashier is not controlled by the software system.

2.3 Product Functions

The software system should control the state display, gates, ticket machines and ticket readers.

- If a valid access card is entered in the card reader of a ticket machine the gate should open.
- If the request button is pressed the driver should get a ticket and the gate should be opened if there is an unreserved parking space available.
- Entering a ticket in a ticket reader should open the exit gate if the fee was paid within the last 15 minutes at the cashier.
- Entering a valid access card in the ticket reader should open the exit gate.
- The gates should be closed after the car has passed the induction loop.
- The state display should show the actual status of occupancy.
- For testing and maintenance of the system, there is the possibility of entering the number of occupied or reserved spaces with the help of the control unit.
- Monthly access cards for reservation may be purchased at the cashier.
- The number of reserved parking spaces should not be higher than 40% of k.

The cashier is not part of the software system.

2.4 User Characteristics

The system users (drivers) should not require special training.

2.5 Assumptions and Dependencies

- Assumptions about the parking garage
    1. Every parking space can be reached from any entrance.
    2. Every exit can be reached from each parking space.
    3. No entrances are convertible to exits and vice versa.
    4. A reserved parking space means that there is a free parking space available but not a specific one.
    5. There are access cards for reserving parking spaces.
    6. Emergency situations (e.g. fire) will not be considered here.
    7. The access cards for the reserved parking spaces are available at the cashier. The cashier controls the number of access cards for reserved parking spaces on its own. If there is a change (new card is sold or a card expires) the cashier will send a message of the actual number of reserved parking spaces to the PGCS.

3. Requirements

3.1 Functional Requirements

This is a list of functional requirements the system should satisfy. The functional requirements are presented in the following way:

- *Description*:
  A description of the specific requirement

- *Input*:
  A description of the inputs that the software system gets

- *Processing*:
  A description of what the software system should do with the input.

- *Output*:
  A description of the response /new state of the software system.

The input, processing and output sections are only specified when needed.

Functional Requirement 1: Data Objects

In the software the following data objects exist: k: maximal number of parking spaces in the parking garage r: number of reserved parking spaces in the parking garage a: k-r, number of parking spaces that are available. o: number of occupied non-reserved parking spaces

3.1.1 General Requirements

Functional Requirement 2

- *Description*

  The PGCS should control the entries and exits of a parking garage.

Functional Requirement 3

- *Description*

  The PGCS has to guarantee that no more than k cars are in the parking garage.

Functional Requirement 4

- *Description*

  The default value for k is 10000.

Functional Requirement 5

- *Description*

  k is divided into "r" reserved parking spaces and "a" public parking spaces.

Functional Requirement 6

- *Description*

  The PGCS should support "n" entries and "m" exits. The PGCS has to handle simultaneous entries and exits.

3.1.2 Update Requirements

Functional Requirement 7

- *Description*

  Purchasing a monthly ticket at the cashier increases value of r by 1.
- *Input*

  Purchase of a new monthly ticket.
- *Processing*

  Update value of r by 1.
- *Output*

  New value of r.

Functional Requirement 8

- *Description*

  The number of reserved parking spaces is changed with the control unit.
- *Input*

  Entry of changes then "reserved" and "enter" at the control unit.
- *Processing*

  Update the value of r.
- *Output*

  New value of r.

Functional Requirement 9

- *Description*

  The control unit can set a new value of r.
- *Input*

  Entry of number then "total" , "reserved" and "enter" at the control unit
- *Processing*

  New value of r
- *Output*

  New value of r.

Functional Requirement 10

- *Description*

  The total number of parking spaces can be written with the control unit
- *Input*

  Entry of k with "total" and "enter" at the control unit
- *Processing*

  Set new value of k.
- *Output*

  New value of k

Functional Requirement 11

- *Description*

  With the control unit it is possible to enter the total number of parking spaces currently allocated.
- *Inputs*

  The total number of parking spaces currently used in the parking garage. The number has to be confirmed with the "enter" button at the control unit.
- *Processing*

  Set the total number of parking spaces to the number that is entered in the control unit.
- *Output*

  Change of the state display depending on the number that was entered.

3.1.3 Entry Requirements

These requirements characterize the requirement for one entrance.

Functional Requirement 12

- *Description*

  The state display should represent the state of occupancy of the public parking spaces. It should display 'free'if there is a non-reserved parking space available at that moment. It should display 'occupied', if there is no non-reserved parking space available at that moment.

Functional Requirement 13

- *Description*
  Every driver should get a ticket at the entrance only if there is a parking space available.
- *Inputs*
  Driver presses the button once.
- *Processing*
  Check if there is a free parking space, i.e. $a > 0$.
- *Output*
  Provide a ticket to the driver if a parking space is available. Increase the number of occupied parking spaces.

Functional Requirement 14

- *Description*
  The gate will open if the ticket is handed out.
- *Input*
  Successful completion of requirement 13.
- *Processing*
  Ticket is handed out and gate will open.
- *Output*
  Driver can take ticket and gate is open.

Functional Requirement 15

- *Description*
  Each driver will be given at most one ticket to enter the parking garage.
- *Input*
  Press the button more than one time in a sequence.
- *Processing and Output*
  Requests after the first for a given car will be ignored.

Functional Requirement 16

- *Description*
  If a driver presses the button while any car leaves and if a > 0 the driver receives a ticket to enter.
- *Input*
  Driver presses the button within two minutes before another car leaves the garage.
- *Processing*
  If a driver is waiting for a ticket and another car leaves the parking garage, a ticket will be issued if a space is available
- *Output*
  Provide a ticket.

Functional Requirement 17

- *Description*
  If more than one car wants to enter the parking garage through different entry stations, the PGCS has to synchronize all stations.
- *Input*
  Several drivers press the request buttons before any of them have been issued tickets to enter.
- *Processing*
  Synchronize the various request.
- *Output*
  Enable entry to the various drivers

3.1.4 Exit Requirements

These requirements characterize one exit.

Functional Requirement 18

- *Description*

  At the exit a car arrives and the driver puts a reserved ticket into the ticket reader. If the ticket is valid, the gate opens.

- *Input*

  Driver puts a reserved ticket in the ticket reader.

- *Processing*

  Check if it is a valid ticket

- *Output*

  If it is valid, gate opens. If not, nothing happens.

Functional Requirement 19

- *Description*

  At the exit a car arrives and driver puts a ticket into the ticket reader. If the ticket is valid, the gate opens.

- *Inputs*

  Driver puts a ticket in the ticket reader.

- *Processing*

  Check if it is a valid ticket and when it was paid. If ticket was not paid or paid more than 15 minutes before, nothing happens. If the ticket was paid within the last 15 minutes the gate will open and the number of occupied parking spaces will be decreased by 1 and the number of available spaces increased by 1.

- *Output*

  If it is a valid ticket the gate opens. If not, nothing happens.

Functional Requirement 20

- *Description*

  If the induction loop is crossed, the gate should close.

- *Input*

  Induction loop goes from present to non-present

- *Output*

  The gate closes.

Functional Requirement 21

- *Description*

  If several cars leave the parking garage at the same time the PGCS has to synchronize all actions.

Control Unit Requirement

3.2 External Interface Requirements

The PGCS has to provide an interface to get messages from the cashier.

3.2.1 User Interfaces

Apart from the control unit there is no need for a, user interface

3.2.2 Hardware Interfaces

There has to be hardware interfaces to the ticket machines, the ticket readers, the gates and the loops. The PGCS will get signals from and will send signals to these devices. An interface to the cashier is not requested yet.

3.3 Performance Requirements

Performance Requirement 1

- *Description*

  After a car has passed the induction loop the gate has to close within 5 sec.

Performance Requirement 2

- *Description*

  If a driver requests a ticket and there are free parking spaces available, he will get the ticket within 3 sec.

Performance Requirement 3

- *Description*

  If a gate opens, it will remain open at least 5 sec.

Performance Requirement 4

- *Description*

  Only one car should pass through the gate each time it opens.

Performance Requirement 5

- *Description*

  Purchasing a reserved ticket changes allocation of a,r,o within 15 sec.

Performance Requirement 6

- *Description*

  A valid reserved ticket at an entry station will always permit successful entry.

Performance Requirement 7

- *Description*

  All changes to state variables at entry or exit station should happen within 5 sec.

Performance Requirement 8

- *Description*

  Reserved tickets are good for 30 days.

Performance Requirement 9

- *Description*

  For each car that enters the parking garage there is a, parking space available.

3.4 Attributes

3.4.1 Availability

The system has to be available 24 h/day. The parking garage won't be closed at any time.

3.4.2 Security

No tickets other than the tickets of this parking garage should be accepted by the ticket reader.

3.4.3 Maintainability

It should be easy to integrate the cashier into the software system

3.4.4Transferability / Conversions

Not Applicable

3.4.5Caution

Not Applicable