Requirements document for a parking garage control system
Chapter 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 depending 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 nonreserved 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 keypad.
Chapter 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. Otherwise 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 monthly ticket in the card reader of the ticket machine.
  2. The ticket machine checks if it is a valid monthly ticket.
  3. If it is a valid monthly ticket 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 notifies the system of which ticket number has just been paid.

- Exit
  1. The driver returns to his car and drives to an exit station.
2. The driver puts the ticket or monthly ticket in the ticket reader.

3. The ticket reader looks up the ticket by ticket number and checks to see if the fee was paid within the last 15 minutes or if the item inserted is a valid monthly ticket. If not, nothing happens. The driver has to see the cashier.

4. Otherwise 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 total number of spaces and number of reserved parking spaces with the help of a device.

- Tickets

1. Monthly and daily tickets will be identified by numbers and a code which the ticket reader can interpret to determine if the ticket is a monthly or a daily ticket. This functionality is outside the system. The system will receive a message from the ticket reader notifying it whether a monthly ticket or a daily ticket was read and what the number on that ticket is.

2. Ticket numbers will only be used in the system for identification purposes, therefore it is not necessary for the ticket numbers to be sequential.

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 keypad

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 a monthly ticket. It reads the ticket number and whether or not it is a monthly ticket 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 a monthly ticket.) With the numerical keypad there is the possibility of entering the number of total and reserved parking spaces. At the cashier the fee is paid. The cashier then notifies the system of the ticket number at this time. 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 monthly ticket 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 non-reserved 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 monthly ticket 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 total spaces or reserved spaces with the help of the control unit.
- Monthly tickets 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 monthly tickets for reserving parking spaces.
6. Emergency situations (e.g. fire) will not be considered here.
7. The monthly tickets for the reserved parking spaces are available at the cashier. The cashier controls the number of monthly tickets for reserved parking spaces on his own. When a monthly ticket is sold, the cashier notifies the system of the number of newly purchased ticket. The system will reduce the number of non-reserved spots by one, and will be responsible for increasing the number of non-reserved spots when the ticket expires after 30 days.
Chapter 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: maximum 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 for non-reserved drivers.
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 1000.

**Functional Requirement 5**

- **Description**
  
  \( k \) is divided into "\( r \)" reserved parking spaces and "\( a \)" non-reserved parking spaces.

**Functional Requirement 6**

- **Description**
  
  The PGCS should support "\( n \)" entries and "\( m \)" exits. The PGCS has to handle simultaneous entries and exits.

**Functional Requirement 7**

- **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 8**

- **Description**
  
  Monthly tickets are good for 30 days.

**3.1.2 Update Requirements**

**Functional Requirement 9**

- **Description**
  
  Cashier’s notification to the system of a new monthly ticket purchase keeps track of that ticket by number and increases the value of \( r \) by one.

  - **Input**
    
    Cashier enters new monthly ticket number and presses the “monthly” button on the control panel.

  - **Processing**
    
    Check to see if \( r + 1 \leq 0.4 \times k \) and if \( a - o \geq 1 \), if so then update value of \( r \) by 1 and store new monthly ticket, otherwise produce error message.

  - **Output**
    
    New value of \( r \), \( a \), or error message

**Functional Requirement 10**

- **Description**
Cashier’s notification to the system that a ticket has been paid.

- **Input**
  Cashier enters the ticket number and presses the “enter” key.

- **Processing**
  System keeps track of ticket number and time paid, so that it can check when driver exits.

**Functional Requirement 11**

- **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**
  Set new value of ‘r’

- **Output**
  New value of ‘r’.

**Functional Requirement 12**

- **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**
  Check to make sure that k is greater than ‘r’ + ‘o’ and that ‘k’ * .4 > r. Set new value of ‘k’.

- **Output**
  New value of ‘k’, ‘a’

**Functional Requirement 13**

- **Description**
  System should deallocate reserved spots when the monthly tickets expire.

- **Processing**
  Each morning at 12:01 AM the system should check to see how many tickets were purchased 31 days ago and should reduce r by this many.

- **Output**
  New value for r, a
3.1.3 Entry Requirements

These requirements characterize the requirement for one entrance.

Functional Requirement 14

- **Description**
  The state display should represent the state of occupancy of the public parking spaces. It should display ‘vacancies’ if there is a non-reserved parking space available at that moment. It should display ‘full’, if there is no non-reserved parking space available at that moment.

Functional Requirement 15

- **Description**
  Every driver with a monthly ticket will insert the ticket into the ticket reader. If it is valid, then the gate will open. A valid reserved ticket will always permit successful entry at any entry station.

- **Input**
  Driver inserts monthly ticket.

- **Processing**
  The ticket number will be looked up in the system to determine if the purchase date of the ticket is 30 days or less prior to the current date. If so, then the gate will open.

- **Output**
  Gate is opened.

Functional Requirement 16

- **Description**
  Every driver using a non-reserved spot, 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’-‘o’ > 0.

- **Output**
  Provide a ticket to the driver if a parking space is available and open gate.

Functional Requirement 17

- **Description**
  Decrease the number of available spots if the driver enters the lot.

- **Input**
  Successful completion of requirement 16 and car passes through induction loop.
• Processing
  Increase value of ‘o’ by 1.

Functional Requirement 18

• Description
  Each driver will be given at most one ticket to enter the parking garage.

• Input
  Press the button while gate is open.

• Processing and Output
  Requests for a ticket while the gate is open will be ignored.

Functional Requirement 19

• Description
  If button is pressed and there are no non-reserved spaces, then nothing happens.

• Input
  Driver presses the button and ‘a’ – ‘o’ = 0.

• Processing
  Nothing happens

Functional Requirement 20

• Description
  If more than one car wants to enter the parking garage through different entry stations, the PGCS has to handle all events in the order they occur.

• Input
  Several drivers press the request buttons before any of them have been issued tickets to enter.

• Processing
  Events should be handled in order in which they occur, and only permit drivers in if there are open spaces.

• Output
  Enable entry to the various drivers

3.1.4 Exit Requirements

These requirements characterize one exit.

Functional Requirement 21

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

- **Input**
  Driver puts a monthly ticket in the ticket reader.

- **Processing**
  Look up the ticket in the system and determine if the purchase date was 30 days or less before the current date. If so then the ticket is valid. Open the gate.

- **Output**
  If it is valid, gate opens. If not, nothing happens.

**Functional Requirement 22**

- **Description**
  A driver with a non-reserved ticket arrives at the exit and puts the ticket into the ticket reader. If the ticket is valid, the gate opens.

- **Inputs**
  Driver puts a ticket in the ticket reader.

- **Processing**
  The ticket is looked up in the system by number. If it was paid in the last 15 minutes then it is valid. If the ticket is valid then the gate opens. Once the car has passed through the induction loop then the value of ‘o’ is decreased by one. If the ticket is invalid then nothing happens.

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

**Functional Requirement 23**

- **Description**
  If several cars leave the parking garage at the same time the PGCS has to handle the events in order.

**Control Unit Requirement**

3.2 External Interface Requirements

The PGCS has to provide an interface to get messages from the numeric keypad used by 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, the loops and the control unit. The PGCS will get signals from and will send signals to these devices.

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 most 20 sec., unless a car is in the induction loop.

Performance Requirement 4

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

Performance Requirement 5

- Description
  Purchasing a monthly ticket changes allocation of a, r within 15 sec.

Performance Requirement 6

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

Performance Requirement 8

- 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
   Not Applicable

3.4.4 Transferability / Conversions
   Not Applicable

3.4.5 Caution
   Not Applicable