Problem Specification of a System for the Light Control in a Floor of a Building of the University of Kaiserslautern

Martin Kronenburg
Research Group Formal Methods and Deduction
kronburg@informatik.uni-kl.de

Christian Peper
Research Group Computer Networks
peper@informatik.uni-kl.de

University of Kaiserslautern

4th June 1999

Contents

Introduction i
Aggregation- and Inheritance Hierarchy ii

Description Classes

Group Datatypes 1
Description Class LightScene 2
Description Class LightSceneOffice 3
Group BasicHumanMachineInterface 4
Description Class EnteredValue 5
Group Sensors 6
Description Class NoMalfunctionSensor 7
Description Class MalfunctionSensor 8
Description Class Sensor 10
Description Class OutdoorLightSensor 11
Description Class BinarySensor 12
Description Class LightSensor 13
Description Class DoorContact 14
Description Class MotionDetector 15
Group Actuator 16
Description Class NoMalfunctionActuator 17
Description Class MalfunctionActuator 18
Description Class Actuator 20
Description Class ToggleActuator 21
Description Class Pulse 22
Description Class RangeActuator 23
Description Class Dimmer 24
Description Class BinaryActuator 25
Description Class ControlSystemActive 26
Description Class PushButton 27
Group CombinedSensorsAndActuators 28
Description Class Door 29
Description Class ControlledLight 30
Description Class DimmableControlledLight 31
Group HumanMachineInterface 32
Description Class AreaLightCtrlPanelFM 33
Description Class RoomLightCtrlPanelFM 34
Description Class LightCtrlPanelRoom 35
Group Occupancy 36
Description Class NoMalfunctionOccupancy 37
Description Class MalfunctionOccupancy 38
Description Class Occupancy 39
Group BuildingUnits 40
Description Class AreaLight 41

Description Class HallwaySection 43
Description Class RoomLight 44
Description Class BasicRoom 47
Description Class Room 48
Description Class RoomNeighb 49
Description Class Office 50
Description Class OfficeNeighb 51
Description Class NoOffice 52
Description Class NoOfficeNeighb 53
Description Class StairCase 54
Description Class Floor 55
Index 57

Introduction

This document contains a problem specification for the light control in the fourth floor of building 32 of the University of Kaiserslautern developed according to the FOREST approach. This projection includes only the natural language specifications and no signature information.

A general introduction to problem specifications developed according to the FOREST approach can be found at the following url
`http://www-avenhaus.informatik.uni-kl.de/forest/EXPLANATIONS/PS_ExplanationEnglish.ps.gz`.

A html-version of this problem specification can be found at the following url
`http://www-avenhaus.informatik.uni-kl.de/forest/EXAMPLES/DAGSTUHL/dagstuhlNLNoSig/dagstuhlNLNoSig.html`.

Aggregation and Inheritance Hierarchy

The graphical overview of the aggregation and inheritance hierarchy among the description classes consists of six figures. Every figure contains the relations within one group of description classes. Figure 1 is the main overview. It represents the relations within the group *Building Units*.

Starting with the main description class *Floor* all relations between the classes are depicted until a class is not refined or until it belongs to another group. Classes that belong to another group and are refined are displayed in a specific color. The refined structure is depicted in a subsequent figure. The relation between groups, colors, and figures is the following one:

- Group *Sensors*; red; Figure 2
- Group *Actuators*; blue; Figure 3
- Group *CombinedSensorsandActuators*; brown; Figure 4
- Group *HumanMachineInterface*; green; Figure 5
- Group *Occupancy*; violet; Figure 6

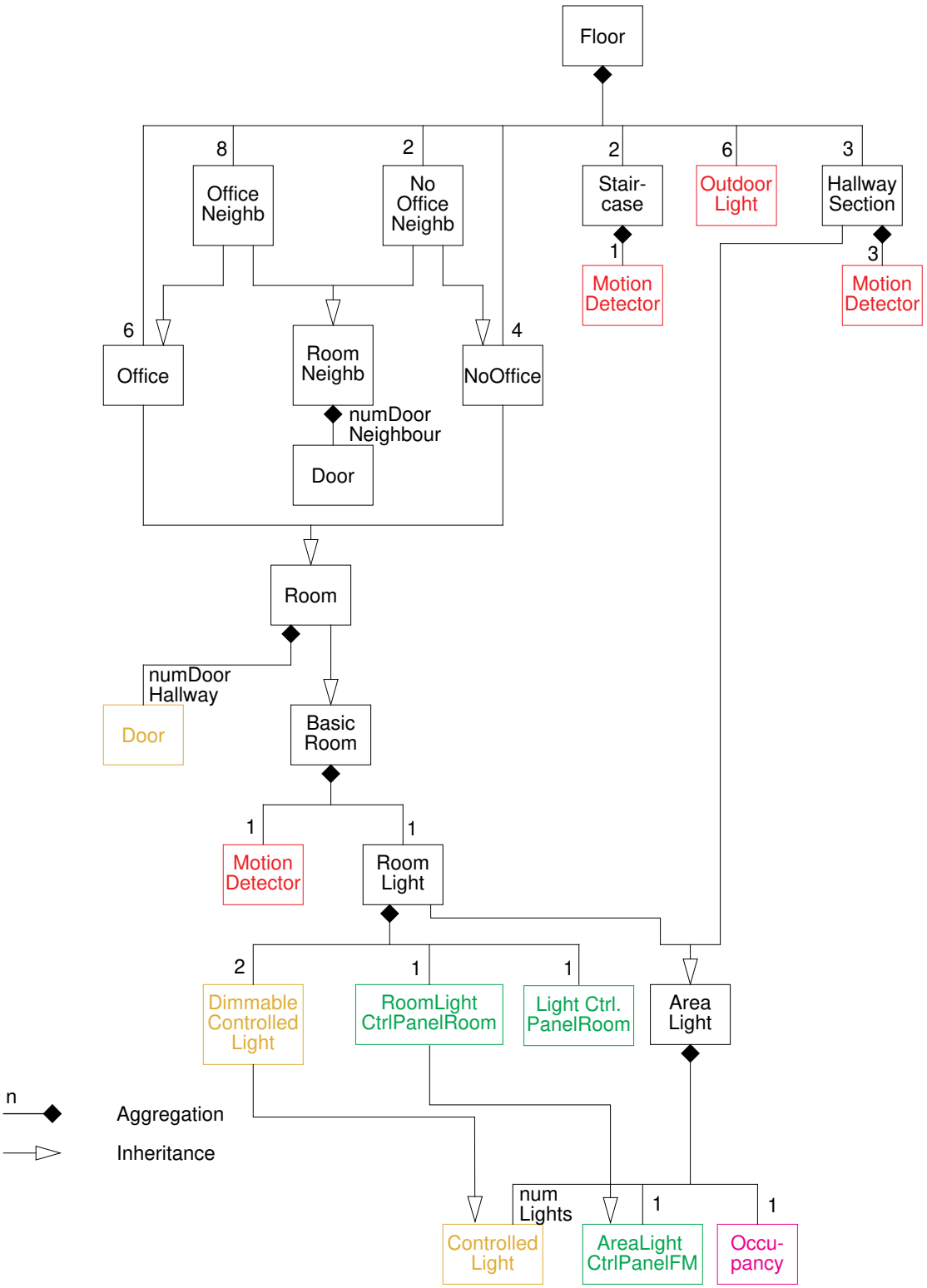

Figure 1: Class structure of group *BuildingUnits*

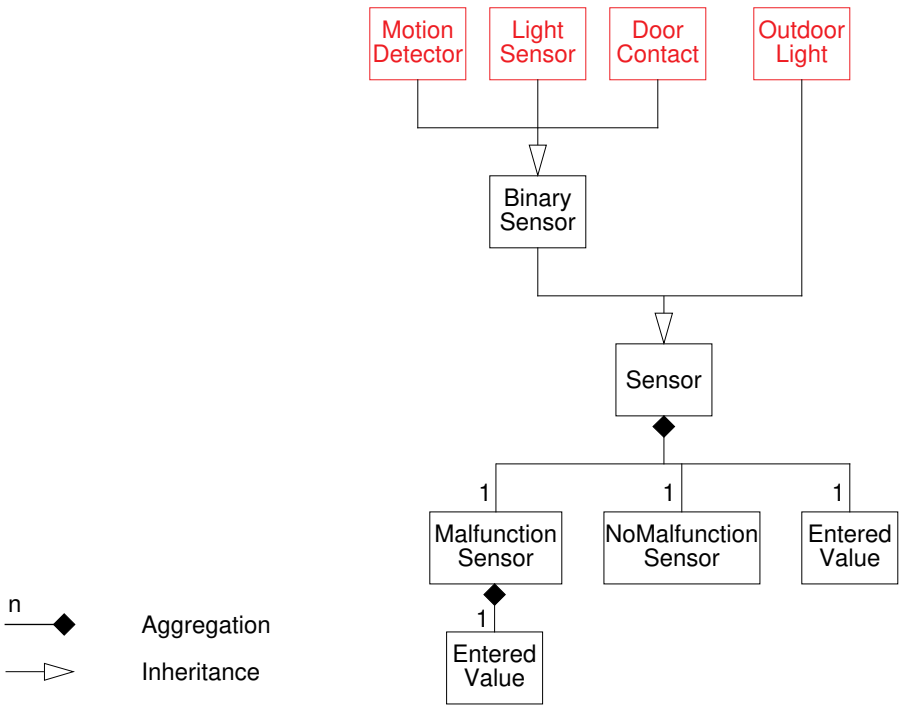

Figure 2: Class structure of group *Sensors*

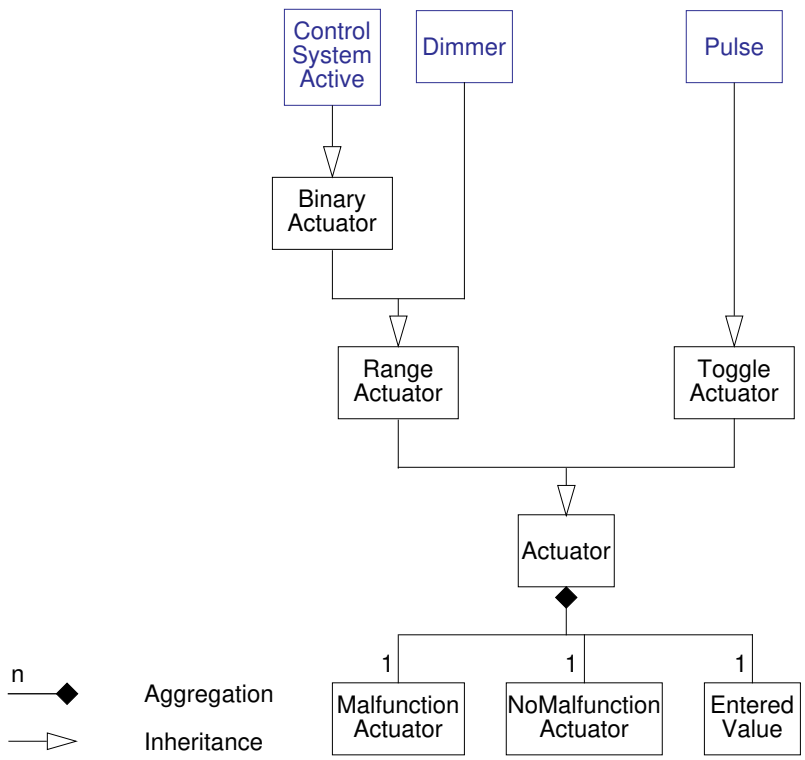

Figure 3: Class structure of group *Actuators*

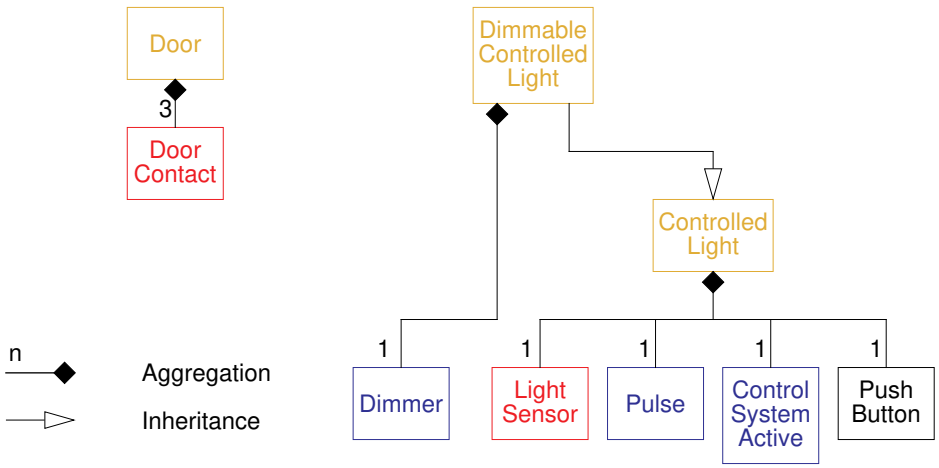

Figure 4: Class structure of group *CombinedSensorsAndActuators*

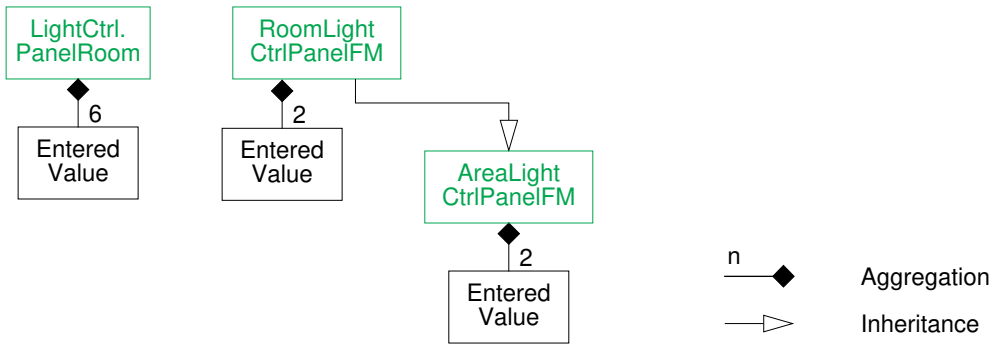

Figure 5: Class structure of group *HumanMachineInterface*

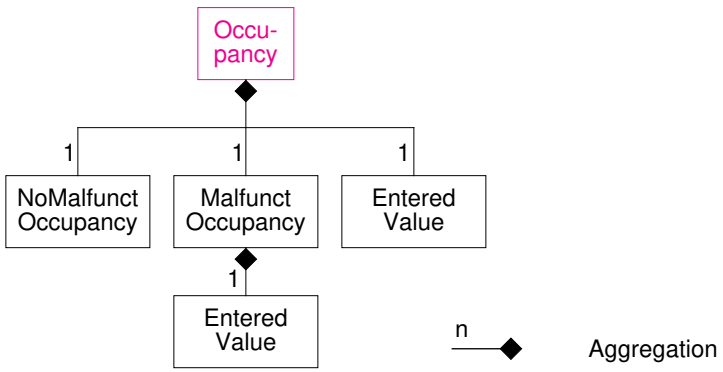

Figure 6: Class structure of group *Occupancy*

Group Datatypes

Intention

This group contains all description classes that are used as datatypes. This means that the description classes in this group are only applied to the declaration of functions or predicates. They are not used for aggregation or inheritance.

Description Class LightScene

Intention

This class provides all definitions concerning a light scene for an arbitrary room. These are w.r.t. the DIC document:

- a name,
- a value of the light intensity of the ambient light of a room,
- an ordered list containing the installed lamps. The list defines the order in which the control system has to use the lamps to reach the given ambient light value. Since this list always consists of two entries we model this list by two functions firstAmbientIlluminaire and secondAmbientIlluminaire.

Aspects concerning a light scene of an office are presented in the derived description class LightSceneOffice.

Machine Specifications

Property *M_LS1*

NL : The first lamp that should be used to reach the desired ambient light of a room and the second lamp that should be used to reach the desired ambient light of a room are always different.

Description Class LightSceneOffice

Intention

This Class contains all the entities concerning a light scene for an office. In addition to the entities belonging to the basic class *LightScene* this class also contains the following entities.

- a value of the light intensity for the desk
- An ordered list containing the installed lamps. The list defines the order in which the control system has to use the lamps to reach the given light intensity value for the desk.

Note that it is not possible for the machine to control the task light. Therefore, it is not considered here!

Machine Specifications

Property *M_LSO1*

NL : The first lamp that should be used to reach the desired desk light and the second lamp that should be used to reach the desired desk light are always different.

Group BasicHumanMachineInterface

Intention

This group contains all description classes that contain the basic definitions representing the interface between the machine and a person. This means here the possibility for a person to enter a value.

Description Class EnteredValue

Intention

This class provides all definitions to describe the entering of a value using an appropriate device (in our case this device is a control panel). For a value e that is entered (expressed by the function *entered*) a value c (expressed by the function *checked*) is determined using the function *nearestValue*. This function is intended to check the entered value e whether it is an allowed value and returns the value which is *nearest* to e.

It is assumed that the entered value can not be seen by the machine, i.e. there exists up to now no possibility to enter the value. Hence, the development of such a device is part of the overall development of the system. In our case this is the control panel, which still has to be developed. Therefore the timed function *entered* has the scope *eh*.

Requirement Specifications

Property *R_EV1*

NL : The checked value is always derived from the entered value using the function *nearestValue*. A change of the entered value is always propagated within the time *checkTime*.

Group Sensors

Intention

This group contains all description classes that contain definitions related to sensors.

Description Class NoMalfunctionSensor

Intention

This class provides all definitions to describe the behavior of a *sensor* if it has *no malfunction*.

It is distinguished between three values:

- the value of the phenomenon existing in the real world (which can never be visible for the machine) (timed function *envEntity*),
- the analogously measured value of this real world value (timed function *measuredEntity*), and
- the digitally measured value which is a conversion of the analog measured value (timed function *convertedEntity*).

Note that this class can be used to describe a sensor that already exists in the environment. This is expressed as follows: At first the value measured by the sensor, namely *convertedEntity*, has the scope *environment controlled, not visible for the machine*. Secondly, the properties D_NMS_1 and D_NMS_2 describing the behavior of the sensor are characterized as domain knowledge. Otherwise, if the sensor does not exist, they have to be requirements.

Domain Knowledge

Property *D_NMS1*

NL : The analogously measured value is always derived from the value of the phenomenon in the real world using the function *modifyReaction*. A change of the value in the real world is always propagated within the time *reactionTime*.

Property *D_NMS2*

NL : The digitally converted value is always derived from the analogously measured value using the function *modifyConversion*. A change of the value in the real world is always propagated within the time *conversionTime*.

Description Class MalfunctionSensor

Intention

This class provides all definitions to describe the behavior of a *sensor* if it has a *malfunction*.

In the case of a malfunction of a sensor, a value is determined depending on a possibly manually entered value by a user and an automatically determined value. If a value has been entered manually this value has priority over the automatically determined value.

Furthermore, the facility manager as well as the user are informed about a malfunction.

Requirement Specifications

Property *I_MS2*

NL : Initially the user is not informed about an important event concerning a sensor.

Property *I_MS3*

NL : Initially the facility manager is not informed about an important event concerning a sensor.

Property *R_MS1*

NL : Whenever a user becomes informed he is informed for at most T_{info} milliseconds.
By this requirement it is expressed, that the user is not informed permanently, but that the information is a kind of signal, e.g. an email.

Property *R_MS2*

NL : Whenever the facility manager becomes informed he is informed for at most T_{info} milliseconds.
By this requirement it is expressed, that the facility manager is not informed permanently, but that the information is a kind of signal, e.g. an email.

Property *R_MS3*

NL : Whenever the machine assumes that a malfunction of the sensor has occurred, the user is informed about this within $T_{infDelay}$ milliseconds.

Property *R_MS4*

NL : Whenever the machine assumes that a malfunction of the sensor has occurred, the facility manager is informed about this within $T_{infDelay}$ milliseconds.

Property *R_MS5*

NL : Whenever the machine assumes that the sensor is working correctly again after it has had a malfunction, the user is informed about this within $T_{infDelay}$ milliseconds.

Property *R_MS6*

NL : Whenever the machine assumes that the sensor is working correctly again after it has had a malfunction, the facility manager is informed about this within $T_{infDelay}$ milliseconds.

Machine Specifications

Property *I_MS1*

NL : Initially no value for the case of a malfunction is entered manually by a user.

Property *M_MS1*

NL : If a value for the status of the sensor in the case of a malfunction has been entered manually this is the value used by the machine in the case of a malfunction of the sensor.
Otherwise, the value used by the machine in the case of a malfunction of the sensor is the value determined automatically in the case of a malfunction of the sensor.

Description Class Sensor

Intention

This class provides all definitions to describe the behavior of a *sensor*. It is composed of the behavior of a sensor if there is no malfunction and the case that there is a malfunction.

A malfunction of a sensor can be detected automatically by the machine or it can be entered manually.

Machine Specifications

Property *I_S1*

NL : Initially it has not been entered manually whether there is a malfunction of the sensor or whether there is no malfunction of it.

Property *M_S1*

NL : The machine assumes a malfunction of the sensor iff one of the following conditions is fulfilled:

- a malfunction has been entered manually or
- it has not been entered manually whether there is a malfunction or not and a malfunction has been detected automatically.

Property *M_S2*

NL : Whenever the machine assumes that the sensor is working correctly the value used by the machine corresponds with the digitally converted version of the measured value.

Property *M_S3*

NL : Whenever the machine assumes that the sensor is not working correctly the value used by the machine corresponds with the value computed for this case.

Description Class OutdoorLightSensor

Intention

This class provides all definitions to describe the behavior of a *outdoor light sensor*. This class is derived from the description class *Sensor*. The following specific parameters of a outdoor light sensor are instantiated:

- the domains of the sensor
- the reaction and conversion time of the sensor

Both functions, *modifyReaction* and *modifyConversion*, have still to be provided by the domain expert.

Domain Knowledge

Property *D_OLS1*

NL : The conversion time of an outdoor light sensor is 1 second.

Property *D_OLS2*

NL : The reaction time of a outdoor light sensor is 10 milliseconds.

Requirement Specifications

Property *R_OLS1*

NL : The value of the sensor determined automatically in the case of a malfunction of the outdoor light sensor corresponds with the last correctly measured value.

Description Class BinarySensor

Intention

This class provides all definitions to describe the behavior of a *binary sensor*. This class is derived from the description class *Sensor*. The following specific parameters of a binary sensor are instantiated:

- the domains of the sensor: a binary sensor is a sensor that returns a 0 or a 1,
- the conversion time of the sensor,
- the transformation functions *modifyReaction* and *modifyConversion* are defined explicitely.

Domain Knowledge

Property *D_BS1*

NL : The conversion time of a binary sensor is 0 milliseconds.

Property *D_BS2*

NL : For a binary sensor the function representing the way in which a value in the real world (*envEntity*) is modified during the reaction of the sensor, resulting in an analogously measured value (*measuredEntity*), is the identity.

Property *D_BS3*

NL : For a binary sensor the function representing the way in which an analogously measured value (*measuredEntity*) is modified during the conversion into a digital value (*convertedEntity*) is the identity.

Description Class LightSensor

Intention

This class provides all definitions of a sensor determining whether a light is on or not. Note that this is not a sensor measuring the illuminance. Therefore, *on* means that the circuit of the light is closed.

Domain Knowledge

Property *D_L1*

NL : The reaction time of a light sensor is 10 ms.

Description Class DoorContact

Intention

This class provides all definitions to describe the behavior of the different kinds of door contacts. These different kinds are *door closed contact*, *door open contact*, and *door lock contact*. A door contact is derived from a binary sensor. Here additionally the reaction time of a door contact is specified.

Domain Knowledge

Property *D_DC1*

NL : The reaction time of a door contact is 10 ms.

Description Class MotionDetector

Intention

This class provides all definitions to describe the behavior of a motion detector. A motion detector is derived from a binary sensor. Here additionally the reaction time of a motion detector is specified.

A motion detector returns a 1 if a person is moving in the range of the detector.

Domain Knowledge

Property *D_MD1*

NL : The reaction time of a motion detector is 1 s.

Group Actuator

Intention

This group contains all description classes that contain definitions related to actuators.

Description Class NoMalfunctionActuator

Intention

This class provides all definitions to describe an *actuator* if it has *no malfunction*.

It is distinguished between two values:

- the value of the phenomenon as it exists in the real world (*envEntity*) (which can never be visible for the machine or controlled directly by the machine), and
- the value adjusted by the machine (*adjustedEntity*).

Note that this class can be used to describe an actuator that already exists in the environment. Therefore the value adjusted by the actuator, namely *adjustedEntity*, has the scope *machine controlled, visible for the machine*. If the actuator does not exist, i.e. the actuator itself has to be developed, the scope of *adjustedEntity* has to be *environment controlled, not visible for the machine*.

Description Class MalfunctionActuator

Intention

This class provides all definitions to describe the behavior of an *actuator* if it has a *malfunction*.

The facility manager as well as the user are informed about a malfunction.

Requirement Specifications

Property *I_MA1*

NL : Initially the user is not informed about an important event concerning this actuator.

Property *I_MA2*

NL : Initially the facility manager is not informed about an important event concerning this actuator.

Property *R_MA1*

NL : Whenever a user becomes informed he is informed for at most T_{info} milliseconds.
By this requirement it is expressed, that the user is not informed permanently, but that the information is a kind of signal, e.g. an email.

Property *R_MA2*

NL : Whenever the facility manager becomes informed he is informed for at most T_{info} milliseconds.
By this requirement it is expressed, that the facility manager is not informed permanently, but that the information is a kind of signal, e.g. an email.

Property *R_MA3*

NL : Whenever the machine assumes that a malfunction of the actuator has occurred, the user is informed about this within $T_{infDelay}$ milliseconds.

Property *R_MA4*

NL : Whenever the machine assumes that a malfunction of the actuator has occurred, the facility manager is informed about this within $T_{infDelay}$ milliseconds.

Property *R_MA5*

NL : Whenever the machine assumes that the actuator is working correctly again after it has had a malfunction, the user is informed about this within $T_{infDelay}$ milliseconds.

Property *R_MA6*

NL : Whenever the machine assumes that the actuator is working correctly again after it has had a malfunction, the facility manager is informed about this within $T_{infDelay}$ milliseconds.

Description Class Actuator

Intention

This class provides all definitions to describe an *actuator*. It is composed of the behavior of an actuator if there is no malfunction and the case that there is a malfunction.

Machine Specifications

Property *I_A1*

NL : Initially it has not been entered manually whether there is a malfunction of the actuator or whether there is no malfunction of it.

Property *M_A1*

NL : The machine assumes a malfunction of the actuator iff one of the following conditions is fulfilled:

- a malfunction has been entered manually
- it has not been entered manually that there is no malfunction and a malfunction has been detected automatically

Description Class ToggleActuator

Intention

This class provides all definitions to describe a toggle actuator. A toggle actuator is an actuator that can send a signal. This is modeled here by setting the adjusted value to 1. If there is no signal the adjusted value is 0.

Note that this class can be used to describe an actuator that already exists in the environment. Therefore, the properties D_TA_1 and D_TA_2 are characterized as domain knowledge. Otherwise, if the actuator does not exist, they have to be requirements/specifications.

Domain Knowledge

Property *D_TA1*

NL : If the value of the entity in the real world (*envEntity*) is 1 and a toggle signal arrives, i.e. the adjusted value (*adjustedEntity*) changes from 0 to 1, then within *reactionTime* milliseconds the value of the entity in the real world (*envEntity*) is 0.

Property *D_TA2*

NL : If the value of the entity in the real world (*envEntity*) is 0 and a toggle signal arrives, i.e. the adjusted value (*adjustedEntity*) changes from 0 to 1, then within *reactionTime* milliseconds the value of the entity in the real world (*envEntity*) is 1.

Description Class Pulse

Intention

This class provides all definitions to describe the *pulse line* of a light by which it can be switched on and off. This class is derived from the description class *ToggleActuator*. Here, the reaction time of a pulse is specified.

Domain Knowledge

Property *D_PUL1*

NL : The reaction time of a pulse line of a light is 10 milliseconds.

Description Class RangeActuator

Intention

This class provides all definitions to describe a *range actuator*. A range actuator is an actuator where a range of values, i.e. a set containing at least two values, can be adjusted. Consider in contrast to this the description class ToggleActuator.

Note that this class can be used to describe an actuator that already exists in the environment. Therefore, the property D_RA_1 is characterized as domain knowledge. Otherwise, if the actuator does not exist, it has to be a requirement/specification.

Domain Knowledge

Property *D_RA1*

NL : The value of the phenomenon in the real world is always derived from the value adjusted by the machine using the function *modifyReaction*. A change of the value adjusted by the machine is always propagated within the time *reactionTime*.

Description Class Dimmer

Intention

This class provides all definitions to describe the *dimmer* of a dimmable light. This class is derived from the description class *RangeActuator*. The following specific parameters of a dimmer are instantiated:

- the domains of the sensor
- the reaction time

The function *modifyReaction* has still to be provided by the domain expert.

Domain Knowledge

Property *D_DIM1*

NL : The reaction time of a *dimmer* is 10 milliseconds.

Description Class BinaryActuator

Intention

This class provides all definitions to describe a *binary actuator*. This class is derived from the description class *RangeActuator*. The following specific parameters of a binary actuator are instantiated:

- the domains of the actuator: a binary actuator is a range actuator that can adjust the values 0 and 1,
- the transformation function *modifyReaction*

Note that this class can be used to describe an actuator that already exists in the environment. Therefore, the property D_BA_1 is characterized as domain knowledge. Otherwise, if the actuator does not exist, it has to be a requirement/specification.

Domain Knowledge

Property *D_BA1*

NL : For a binary actuator the function representing the way in which a value adjusted by the machine (*adjustedEntity*) is modified during the reaction of the actuator, resulting in a value of the actuator in the real world (*envEntity*), is the identity.

Description Class ControlSystemActive

Intention

This class provides all definitions to describe the signal *control system is active* that can be sent by the machine in order to indicate that the machine is still active. In addition to the definitions of the description class *BinaryActuator*, here the reaction time of this signal is specified.

Domain Knowledge

Property *D_CSA1*

NL : The reaction time of a signal *control system is active* is 10 ms.

Description Class PushButton

Intention

This class provides all definitions concerning the control of a light using pushbuttons. The number of pushbuttons is not relevant for the behavior. Note further that the push buttons are not visible for the machine.

Domain Knowledge

Property *D_PB1*

NL : Whenever the light is off and any push button is pressed, the light changes to on within time T_PB.

Property *D_PB2*

NL : Whenever the light is on and any push button is pressed, the light changes to off within time T_PB.

Group CombinedSensorsAndActuators

Intention

This group contains all description classes in which several sensors or actuators are combined.

Description Class Door

Intention

This class provides all definitions to describe a door. A door contains three door contacts.

Description Class ControlledLight

Intention

This description class provides all definitions of sensors and actuators relevant for the control of a light that is not dimmable. This installation includes:

- a light sensor indicating whether the light is on or off (object *ls*)
- an actuator allowing the machine to switch on and off the light (object *pulse*)
- an actuator allowing the machine to signal that it is still active (object *csa*)
- a push button allowing a person to switch on and off the light (object *pb*)
  Note that a push of a button is not directly visible for the machine but only by the light sensor.

Furthermore, it is defined under which conditions a light is controlled *manually*.

Domain Knowledge

Property *D_CL1*

NL : The phenomenon in the real world which is measured by the light sensor is identical with the phenomenon adjusted by the pulse actuator.

Property *D_CL2*

NL : The phenomenon in the real world which is measured by the light sensor is identical with the phenomenon adjusted by the push buttons.

Machine Specifications

Property *I_CL1*

NL : Initially the light is not controlled manually.

Property *M_CL1*

NL : If no pulse has been sent by the machine for the last *pulse.noMalAct.reactionTime* milliseconds and the measured value of the light sensor changes the light is controlled manually.

Property *M_CL2*

NL : If a pulse has been sent by the machine within the last *pulse.noMalAct.reactionTime* milliseconds the light is not controlled manually.

Property *M_CL3*

NL : Whenever the signal *control system is active* is set to 1 it is set within one second to 0.

Property *M_CL4*

NL : Whenever the signal *control system is active* is set to 0 it is set within 59 seconds to 1.

Description Class DimmableControlledLight

Intention

This description class provides all definitions of sensors and actuators relevant for the control of a dimmable light. This installation includes in addition to a controlled light:

- a dimmer, i.e. an actuator for the control of the illuminance of the light if it is on (object *dimmer*).
  Note that this dimmer can not switch on or off the light, i.e. even if the dimmer value is zero the light can have the status on, i.e. the circuit of the light is closed. Thus, the dimmer sets the illuminance in the case that the light is on.

Group HumanMachineInterface

Intention

This group contains all description classes that contain definitions representing the interface between the machine and a person. These description classes are on a more abstract level than the description classes of group *BasicHumanMachineInterface*.

Description Class AreaLightCtrlPanelFM

Intention

This class provides all the definitions of entities concerning the control of the light in an area for which the facility manager can enter a value using the control panel. An area is here the generalization of rooms and hallways. This is

- a possibility to switch on or off all lights in a certain area.
- a possibility to determine the time a certain area has to be unoccupied continuously before the lights are switched off.

Machine Specifications

Property *I_LCPFM1*

NL : Initially an area has to be continuously empty for at least 10 minutes before the lights in an area are switched off.

Property *I_LCPFM2*

NL : Initially the lights in an area are not switched off by the facility manager.

Description Class RoomLightCtrlPanelFM

Intention

This class provides all the definitions of entities concerning the control of the light in a room for which the facility manager can enter a value using the control panel. In addition to the definitions of description class *AreaLightCtrlPanelFM* this is

- a possibility to determine the maximal delay within the reoccupation of a room leads to a reestablishing of the previous light scene.

Machine Specifications

Property *I_LCPR1*

NL : Initially the maximal delay within the reoccupation of a room leads to a reestablishing of the previous light scene is 20 minutes.

Property *M_LCPR1*

NL : The maximal delay within the reoccupation of a room leads to a reestablishing of the previous light scene is always greater than the time a room has to be continuously empty before the lights in a room are switched off.

Description Class LightCtrlPanelRoom

Intention

This class provides all the definitions of entities concerning the control of the light in a room for which a user can enter a value using the control panel of a room. These are

- a possibility to choose the current light scene
- a possibility to choose the current ambient light level
- a possibility to switch on and off the window ceiling light
- a possibility to switch on and off the wall ceiling light
- a possibility to determine whether
  - the currently set ambient light level should be applied or
  - the currently settings of the window and wall ceiling light switches of the control panel should be applied
- a possibility to determine whether
  - the light scenes are enabled by the control panel, i.e. the currently chosen light scene should be applied or
  - the currently settings of the control panel concerning the ambient switch, the ceiling light switches, and the ambient light level should be applied.

If the light is controlled using the control panel the following cases are possible:

- A chosen light scene has to be established. This is the case if the value of *lightSceneSwitch* is 1.
- A set ambient light level has to be established. This is the case if the value of *lightSceneSwitch* is 0 and the value of *ambientSwitch* is 1.
- The two lights are set directly using the control panel. This is the case if the value of *lightSceneSwitch* is 0 and the value of *ambientSwitch* is 0.

These different cases are specified in the description class *RoomLight*.

Note that such a control panel does not exist up to now and has to be developed, too.

Requirement Specifications

Property *R_LCP1*

NonFormal : The control panel should be easy and intuitive to use.

Group Occupancy

Intention

This group contains all description classes that contain definitions needed to define under which conditions an area is occupied.

Description Class NoMalfunctionOccupancy

Intention

This class provides all definitions to describe the term *occupancy* for an area in the case that there is no malfunction of the sensors used to determine occupancy in this area.

Whether the machine assumes occupancy in an area or not depends on

- three time values, dt_1, dt_2, dt_3, and
- on a predicate indicating whether a person is currently in the area or not.

Note that the status of *occupancy* in general does not correspond with the fact whether there is a person in an area or not. *Occupancy* is a derived term depending on this fact.

Machine Specifications

Property *M_NMO1*

NL : An area is defined to be occupied if and only if one of the following conditions holds:

- during the past dt_1 time units a person has continuously been in the area or
- there has already been a time span of length $dt_1 + dt_2$ such that a person has continuously been in the area, and since this has been the case for the last time, periods where the area has continuously been empty were not longer than or equal to dt_3.

Description Class MalfunctionOccupancy

Intention

This class provides all definitions to describe the behavior of the system concerning the detection of occupancy in an area in the case that some sensors used to determine occupancy have a malfunction.

In the case of a malfunction of such a sensor, a value is determined depending on a possibly manually entered value by a user and an automatically generated value. If a value has been entered manually this value has priority over the automatically determined value.

Most of the entities introduced in this description class have to be specified in the description classes where they are used.

Machine Specifications

Property *I_MO1*

NL : Initially no value for occupancy in the case of a malfunction is entered manually.

Property *M_MO1*

NL : If a value for the status of occupancy has been entered manually this is the value used for occupancy in the case of a malfunction of some sensors used to determine occupancy.
Otherwise, the value used for occupancy in the case of a malfunction of the sensors used to determine occupancy corresponds with the value determined automatically.

Description Class Occupancy

Intention

This class provides all definitions to describe the occupancy in a specific area. It is a composition of the following two cases:

- occupancy is computed under the condition that all sensors used to determine occupance have no malfunction
- occupancy is computed under the condition that some sensors used to determine occupance have a malfunction

To compose these two cases some entities are introduced that allow to determine whether some sensors used to determine occupancy have a malfunction.

Machine Specifications

Property *I_OCC1*

NL : Initially it has not been entered manually whether there is a malfunction of the sensors determining occupancy or whether there is no malfunction of them.

Property *M_OCC1*

NL : The machine assumes a malfunction of some sensors used to determine occupancy iff one of the following conditions is fulfilled:

- a malfunction has been entered manually or
- it has not been entered manually that there is no malfunction and a malfunction has been detected automatically.

Property *M_OCC2*

NL : The machine assumes occupancy if and only if one of the following condition is fulfilled:

- the machine assumes a malfunction of some sensors used to detect occupancy and the predicate representing the value concerning occupancy assumed in this case is true,
- the machine assumes no malfunction and the predicate representing the value concerning occupancy assumed in this case is true.

Group BuildingUnits

Intention

This group contains all description classes that describe units of the floor considered in this problem specification.

Description Class AreaLight

Intention

This class provides all definitions to describe the general aspects of lights in an area. An area is here the generalization of rooms and hallways. In more detail this class provides:

- A set of lights that can be conrolled by the machine. The concrete number of controlled lights can be specified by instantiating the parameter *numLights*.
- The specification of the behavior of the system concerning the aspect *safety*. Here safety means that there is sufficient light to move. This class provides the requirements concerning the case that safe light has to be established. Other cases can be considered in derived classes.
- The specification of the behavior of the system concerning the case that the facility manager controls the light using a control panel.
- The specification of the behavior of the system concerning the case that the light is controlled manually.

Domain Knowledge

Property *D_AL1*

NL : The time constant T_AL_2 is allways smaller than the time constant T_AL_1.

Property *D_AL2*

NL : The predicate *allLightsOff* is true iff all lights that are controlled by the machine are off.

Requirement Specifications

Property *R_AL1*

NL : Whenever the light is controlled by the facility manager using the control panel and the area is unoccupied for at least T_AL_1 milliseconds, the status of the machine controlled lights equals within this time the status as it is determined by the facility manager.

Property *R_AL2*

NL : Whenever the light is not controlled by the facility manager using the control panel and the area is unoccupied for at least T_AL_1 milliseconds, all machine controlled lights are off within this time and remains off at least as long as the precondition is true.

Property *R_AL3*

NL : Whenever safe light has to be established and the area is occupied for at least T_AL_1 milliseconds, the light is sufficient to move safely within this time span and remains sufficient to move safely at least as long as the precondition is true.

Machine Specifications

Property *I_AL1*

NL : Initially safe light has to be established if an area is occupied.

Property *I_AL2*

NL : Initially the light is not controlled by the facility manager using his control panel.

Property *M_AL1*

NL : The value of T_AL_1 is always 50 miliiseconds.

Property *M_AL2*

NL : The value of T_AL_2 is always 30 miliiseconds.

Property *M_AL3*

NL : The first time delay used to determine occupancy concerning the light control is always 1 second.

Property *M_AL4*

NL : The second time delay used to determine occupancy concerning the light control is always 0 second.

Property *M_AL5*

NL : The third time delay used to determine occupancy concerning the light control corresponds always with the value entered by the facility manager for the time a certain area has to be unoccupied continuously before the lights are switched off.

Property *M_AL6*

NL : The value determined automatically by the machine for occupancy concerning the light control in the case of a malfunction of some sensors used to determine occupancy concerning the light control is *occupancy*.

Property *M_AL7*

NL : Whenever safe light has to be established and the area is occupied the necessary actuators, i.e. pulse lines, are set within T_AL_2 milliseconds to establish safe light.

Property *M_AL8*

NL : The machine assumes that the light is controlled manually in an area iff at least one light in this area is controlled manually and the area is occupied.

Property *M_AL9*

NL : If the light in an area is controlled manually, the settings of all lights remain unchanged.

Description Class HallwaySection

Intention

This class provides all definitions to describe the hallway specific light requirements. Each hallway section is equipped with

- one group of not dimmable ceiling lights that are considered here as one ceiling light,
- three motion detectors: one at each end of a hallway section and one in the middle of a hallway section.

Note that there are no requirements concerning the temperature in a hallway.

Requirement Specifications

Property *R_HS1*

NL : Whenever a person is continuously in the near of a hallway door of the hallway section for at least T_HS_1 time units indicating that he wants to enter the hallway section, then the light in the hallway section will be on within this time span, and remains on at least as long as the person is near the hallway section.

Property *R_HS2*

NL : Whenever the lights are neither controllable automatically nor manually for at least T_HS_2 time units, then eventually within this time span, the light in the hallway section is on and remains so at least as long as the lights are neither controllable automatically nor manually.

Machine Specifications

Property *M_HS1*

NL : The value of T_HS_1 is always 50 milliseconds.

Property *M_HS2*

NL : Whenever the light is not controlled manually the light intensity in a hallway section that is occupied has to be greater than or equal to the light intensity needed to establish *save light*.

Property *M_HS3*

NL : The machine always assumes that there is currently no person in a hallway section iff the values of all three motion detectors are 0.

Description Class RoomLight

Intention

This class provides all definitions to describe the environment and the requirements concerning the control of lights common to any kind of room. In more detail this means that the cases are considered

- that safe light has to be established. This behavior is inherited from the description class *AreaLight*;
- that the light is controlled manually.
- that the light is controlled using the control panel. See the intention of description class *LightCtrlPanelRoom* for a more detailed explanation of this case.

In each room there are two dimmable lights that are controlled by the machine: a ceiling light at the window and a ceiling light at the wall to the hallway. Furthermore, in each room there is a control panel for entering values relevant for the control of the light. This is done by the user of a room. This control panel is not the same as the one the facility manager has.

Requirement Specifications

Property *R_RL1*

NonFormal : The standard light scene in the case that the outdoor light sensors have no malfunction is that the light intensity is sufficient to move safely.

Property *R_RL2*

NonFormal : The standard light scene in the case of a malfunction of the outdoor light sensors is that all ceiling lights are on.

Machine Specifications

Property *I_RL1*

NL : Initially the current standard light scene is the light scene that has to become standard light scene in the case that the outdoor light sensors have no malfunction.

Property *I_RL2*

NL : Initially the current light scene is the current standard light scene.

Property *I_RL3*

NL : Initially the current ambient light level is the current default ambient light level.

Property *M_RL1*

NL : The value of T_RL_1 is always 50 miliiseconds.

Property *M_RL2*

NL : The current standard light scene always corresponds with the default light scene determined by the facility manager.

Property *M_RL3*

NL : The current ambient light level always corresponds with the default ambient light level determined by the facility manager.

Property *M_RL4*

NL : The light intensity in a room that is occupied can be less than the light intensity needed to establish *save light* iff the light is controlled manually or light scenes are enabled or the ambient light level is enabled.

Property *M_RL5*

NL : Whenever the room becomes occupied now and was occupied sometimes within the last T_RL_2 milliseconds and the light scene l was last chosen, the light scene l is the current light scene.

Property *M_RL6*

NL : Whenever the room becomes occupied now and has been continuously unoccupied for the last T_RL_2 milliseconds, the current light scene is the current standard light scene.

Property *M_RL7*

NL : Whenever the room is occupied, but has not just become occupied now, and the light scene l becomes chosen, this light scene l is the current light scene within time T_RL_1.

Property *M_RL8*

NL : Whenever the room is occupied and the light is not manually controlled and light scenes are enabled by the control panel, the window and wall lights are dimmed and switched on or off as necessary to achieve the ambient light level of the current light scene under consideration of the current light scene's lighting order.
In short: Under the mentioned conditions the current light scene has to be established.

Property *M_RL9*

NL : Whenever the room is occupied and the light is not manually controlled, light scenes are disabled and ambient light settings are enabled by the control panel, the window and wall lights are dimmed and switched on or off as necessary to achieve the current ambient light level.
In short: Under the mentioned conditions the current ambient light setting is established.

Property *M_RL10*

NL : Whenever the room is occupied and the light is not manually controlled and light scenes and ambient light setting are disabled by the control panel, the window and wall lights are completely switched off or on as selected in the control panels ceiling light switches.
In short: Under the mentioned conditions the light is controlled by the ceiling light switches.

Property *M_RL11*

NL : As long as the light is controlled manually, the dimmer settings of the window and wall light remain unchanged.

Property *M_RL12*

NL : Whenever the room becomes occupied now and was occupied sometimes within the last T_RL_2 milliseconds and the ambient light level l was last chosen, the ambient light level l is the current ambient light level.

Property *M_RL13*

NL : Whenever the room becomes occupied now and has not been occupied during the last time T_RL_2, the current ambient light level is the default ambient light level.

Property *M_RL14*

NL : Whenever the room is occupied, but has not just become occupied now, and the ambient light level l becomes chosen, this ambient light level l is the current ambient light level within time T_RL_1.

Description Class BasicRoom

Intention

This class provides all definitions to describe the behavior common to any kind of room. This includes the definitions of the light control in a room as well as the aggregation of the objects that are installed in each room. In this class we do not consider such objects for which the number of installed instances can vary from room to room, e.g. the number of doors to adjacent rooms. These objects are integrated in the derived description class *Room*.

Each room is equipped with

- one motion detector

Machine Specifications

Property *M_BaR1*

NL : Concerning the light control the machine always assumes that there is currently no person in a room iff the value of the motion detector is 0.

Description Class Room

Intention

This class extends the description class *BasicRoom* with the definitions concerning those objects of which in each room at least one instantiation is installed but where the concrete number of instantions can vary from room to room.

In addition to the objects introduced in description class *BasicRoom* each room is equipped with

- *numDoorHallway* many doors to the hallway

Description Class RoomNeighb

Intention

This class provides all definitions that has to be added to the definitions of a room which is connected by a door to at least one neighbour room. Such rooms are equipped additionally with

- *numDoorNeighbour* many doors to neighbour rooms

Note that this description class is not derived from the description class *Room* in order to be able to combine this description class with each kind of room no matter whether it is for example a blinded office or a non–office without blinds.

Description Class Office

Intention

This class provides all definitions specific to all offices. In addition to the definitions of the description class *Room* only the installation of the control panel is considered here.

Requirement Specifications

Property *R_O1*

NonFormal : The control panel should be movable in the offices, like a telephone.

Description Class OfficeNeighb

Intention

This class provides all definitions specific to all offices which are connected by at least one door to a neighbour room. In addition to the definitions of the description class *Office* the definitions of the description class *RoomNeighb* are also provided here.

Description Class NoOffice

Intention

This class provides all definitions specific to rooms that are no offices. In addition to the definitions of the description class *Room* only the installation of the control panel is considered here.

Requirement Specifications

Property *R_NO1*

NonFormal : The control panel should be installed near a door to the hallway.

Description Class NoOfficeNeighb

Intention

This class provides all definitions specific to rooms that are no offices and which are connected by at least one door to a neighbour room. In addition to the definitions of the description class *NoOffice* the definitions of the description class *RoomNeighb* are also provided here.

Description Class StairCase

Intention

This class provides all definitions to describe the entities belonging to a staircase which are relevant for the considered system. This is only a motion detector.

Description Class Floor

Intention

This class is the main class. It contains and combines all the objects belonging to the considered system. These objects are

- the rooms of the floor
- the hallway sections of the floor
- the parts of the staircases belonging to the floor
- the outdoor light sensors
- the outdoor temperature sensor

Domain Knowledge

Property *D_F1*

NL : The two rooms with room numbers 435 and 433 are connected by a door.

Property *D_F2*

NL : The two rooms with room numbers 433 and 431 are connected by a door.

Property *D_F3*

NL : The two rooms with room numbers 425 and 423 are connected by a door.

Property *D_F4*

NL : The two rooms with room numbers 423 and 421 are connected by a door.

Property *D_F5*

NL : The two rooms with room numbers 418 and 416 are connected by a door.

Property *D_F6*

NL : The two rooms with room numbers 416 and 414 are connected by a door.

Property *D_F7*

NL : The eastern and middle hallway section are connected by a door.

Property *D_F8*

NL : The middle and western hallway section are connected by a door.

Requirement Specifications

Property *R_F1*

NL : Whenever some outdoor light sensors are not working correctly and the hallway is occupied for at least $T_{malReact}$ time units, then eventually within this time span, the light in the hallway is on and remains on at least as long as the precondition is true.

Property *R_F2*

NonFormal : If a malfunction occurs, the control system supports the facility manager by finding the reason.

Property *R_F3*

NonFormal : The system provides reports on current and past energy consumption.

Property *R_F4*

NonFormal : All malfunctions and unusual conditions are stored and reported on request.

Property *R_F5*

NonFormal : All hardware connections have to be made according to DIN standards.

Property *R_F6*

NonFormal : No hazardous condition for persons, inventory or building are allowed.

Property *R_F7*

NonFormal : The system issues warnings on unreasonable inputs.

Machine Specifications

Property *M_F1*

NL : The machine assumes that some outdoor light sensor has a malfunction iff one of the six outdoor light sensors has a malfunction.

Property *M_F2*

NL : Whenever the machine assumes that some outdoor light sensors are not working correctly, then the standard light scene is the standard fault light scene, i.e. all ceiling lights are on.

Index

A

Actuator 16, 20
AreaLight 41
AreaLightCtrlPanelFM 33

B

BasicHumanMachineInterface 4
BasicRoom 47
BinaryActuator 25
BinarySensor 12
BuildingUnits 40

C

CombinedSensorsAndActuators 28
ControlledLight 30
ControlSystemActive 26

D

D_AL1 41
D_AL2 41
D_BA1 25
D_BS1 12
D_BS2 12
D_BS3 12
D_CL1 30
D_CL2 30
D_CSA1 26
D_DC1 14
D_DIM1 24
D_F1 55
D_F2 55
D_F3 55
D_F4 55
D_F5 55
D_F6 55
D_F7 55
D_F8 55
D_L1 13
D_MD1 15
D_NMS1 7
D_NMS2 7
D_OLS1 11
D_OLS2 11
D_PB1 27
D_PB2 27
D_PUL1 22
D_RA1 23
D_TA1 21
D_TA2 21
Datatypes 1
DimmableControlledLight 31
Dimmer 24
Door 29
DoorContact 14

E

EnteredValue 5

F

Floor 55

H

HallwaySection 43
HumanMachineInterface 32

I

I_A1 20
I_AL1 42
I_AL2 42
I_CL1 30
I_LCPFM1 33
I_LCPFM2 33
I_LCPR1 34
I_MA1 18
I_MA2 18
I_MO1 38
I_MS1 9
I_MS2 8
I_MS3 8
I_OCC1 39
I_RL1 44
I_RL2 44
I_RL3 44
I_S1 10

L

LightCtrlPanelRoom 35

LightScene 2
LightSceneOffice 3
LightSensor 13

M
M_A1 20
M_AL1 42
M_AL2 42
M_AL3 42
M_AL4 42
M_AL5 42
M_AL6 42
M_AL7 42
M_AL8 42
M_AL9 42
M_BaR1 47
M_CL1 30
M_CL2 30
M_CL3 30
M_CL4 30
M_F1 56
M_F2 56
M_HS1 43
M_HS2 43
M_HS3 43
M_LCPR1 34
M_LS1 2
M_LSO1 3
M_MO1 38
M_MS1 9
M_NMO1 37
M_OCC1 39
M_OCC2 39
M_RL1 44
M_RL10 46
M_RL11 46
M_RL12 46
M_RL13 46
M_RL14 46
M_RL2 45
M_RL3 45
M_RL4 45
M_RL5 45
M_RL6 45
M_RL7 45
M_RL8 45
M_RL9 45
M_S1 10
M_S2 10
M_S3 10
MalfunctionActuator 18
MalfunctionOccupancy 38
MalfunctionSensor 8
MotionDetector 15

N
NoMalfunctionActuator 17
NoMalfunctionOccupancy 37
NoMalfunctionSensor 7
NoOffice 52
NoOfficeNeighb 53

O
Occupancy 36, 39
Office 50
OfficeNeighb 51
OutdoorLightSensor 11

P
Pulse 22
PushButton 27

R
R_AL1 41
R_AL2 41
R_AL3 41
R_EV1 5
R_F1 56
R_F2 56
R_F3 56
R_F4 56
R_F5 56
R_F6 56
R_F7 56
R_HS1 43
R_HS2 43
R_LCP1 35
R_MA1 18
R_MA2 18
R_MA3 18
R_MA4 18

R_MA5 18
R_MA6 19
R_MS1 8
R_MS2 8
R_MS3 8
R_MS4 8
R_MS5 8
R_MS6 9
R_NO1 52
R_O1 50
R_OLS1 11
R_RL1 44
R_RL2 44
RangeActuator 23
Room 48
RoomLight 44
RoomLightCtrlPanelFM 34
RoomNeighb 49

S

Sensor 10
Sensors 6
StairCase 54

T

ToggleActuator 21