Problem Specification of a System for the Light and Temperature 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

9th 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
Description Class DynamicInterval 6
Description Class CalendarInterval 7
Description Class ControlledInput 8
Description Class ControlledDynamicInterval 9
Group Sensors 10
Description Class NoMalfunctionSensor 11
Description Class MalfunctionSensor 12
Description Class Sensor 14
Description Class RoomTempSensor 15
Description Class OutdoorTempSensor 16
Description Class OutdoorLightSensor 17
Description Class WaterTempSensor 18
Description Class BinarySensor 19
Description Class LightSensor 20
Description Class DoorContact 21
Description Class MotionDetector 22
Group Actuator 23
Description Class NoMalfunctionActuator 24
Description Class MalfunctionActuator 25
Description Class Actuator 27
Description Class ToggleActuator 28
Description Class Pulse 29
Description Class RangeActuator 30
Description Class Dimmer 31
Description Class BinaryActuator 32
Description Class ControlSystemActive 33
Description Class ThreePointActuator 34
Description Class RadiatorValve 35
Description Class PushButton 36
Group CombinedSensorsAndActuators 37
Description Class Door 38
Description Class SunBlind 39
Description Class Radiator 41
Description Class ControlledLight 42

Description Class DimmableControlledLight 43
Group HumanMachineInterface 44
Description Class OccDelayTimes 45
Description Class AreaLightCtrlPanelFM 46
Description Class RoomLightCtrlPanelFM 47
Description Class LightCtrlPanelRoom 48
Description Class LightCtrlPanelBlindedRoom 49
Description Class TemperatureCtrlPanelFM 50
Description Class TemperatureCtrlPanelRoom 52
Description Class TemperatureCtrlPanel 53
Description Class ExpectedUsage 54
Description Class GroupOccupancy 55
Description Class AdaptiveOccupancyTimes 56
Description Class NoMalfunctionOccupancy 57
Description Class MalfunctionOccupancy 58
Description Class Occupancy 59
Group TemperatureModes 60
Description Class TempOperationMode 61
Description Class TempOperationModeSet 62
Group BuildingUnits 64
Description Class AreaLight 65
Description Class HallwaySection 67
Description Class RoomLight 68
Description Class BlindedRoomLight 71
Description Class RoomTemperature 72
Description Class BlindedRoomTemperature 76
Description Class BasicRoom 77
Description Class Room 78
Description Class BlindedRoom 80
Description Class RoomNeighb 82
Description Class Office 83
Description Class OfficeNeighb 84
Description Class BlindedOffice 85
Description Class BlindedOfficeNeighb 86
Description Class NoOffice 87
Description Class BlindedNoOffice 88
Description Class BlindedNoOfficeNeighb 89
Description Class StairCase 90
Description Class Floor 91
Index 94

Introduction

This document contains a problem specification for the light and temperature 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/LIGHT_TEMP/LightTempNLNoSig/LightTempNLNoSig.html`.

Aggregation and Inheritance Hierarchy

The graphical overview of the aggregation and inheritance hierarchy among the description classes consists of seven 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
- Group *TemperatureModes*; light-blue; Figure 7

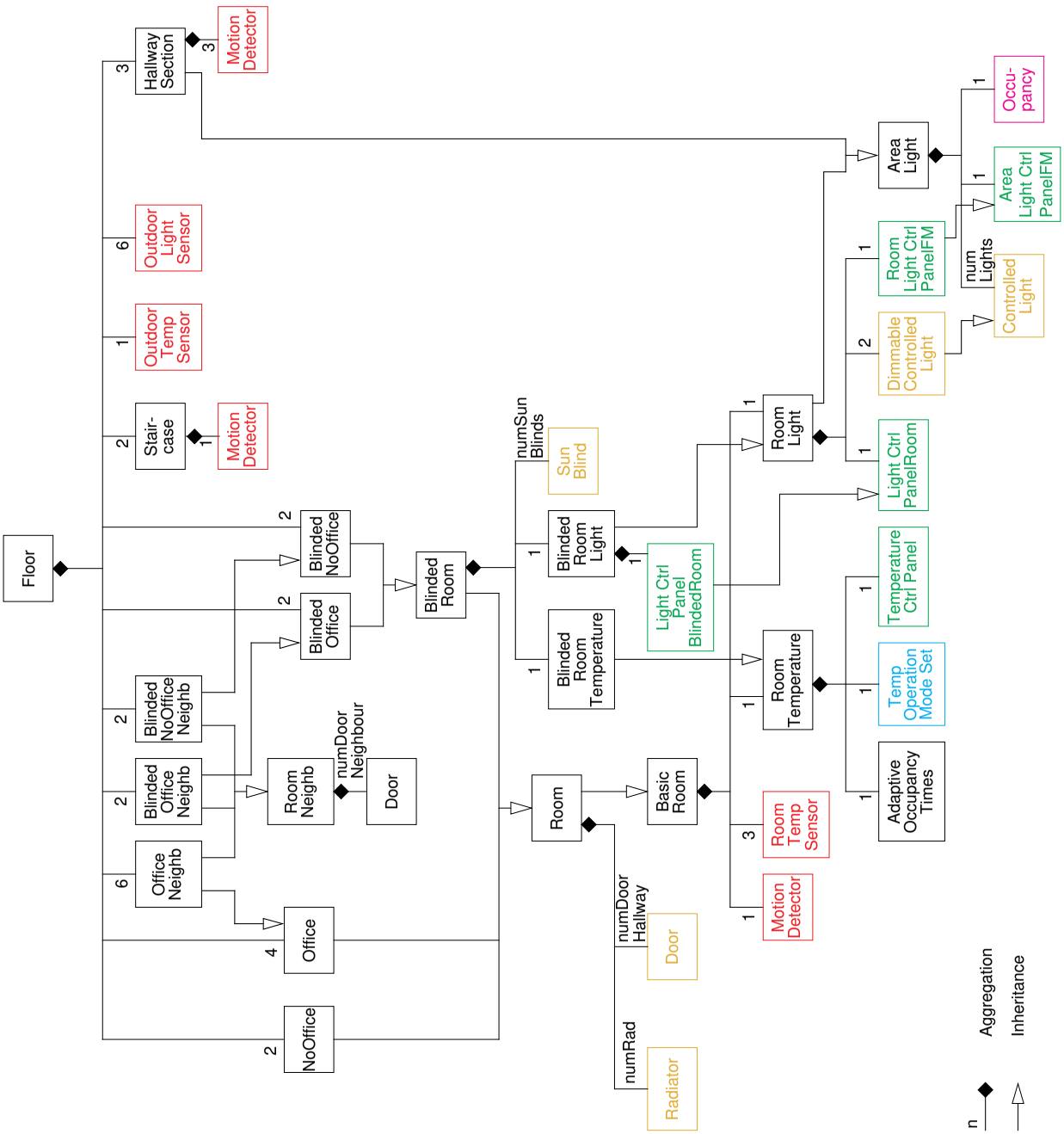

Figure 1: Class structure of group *BuildingUnits*

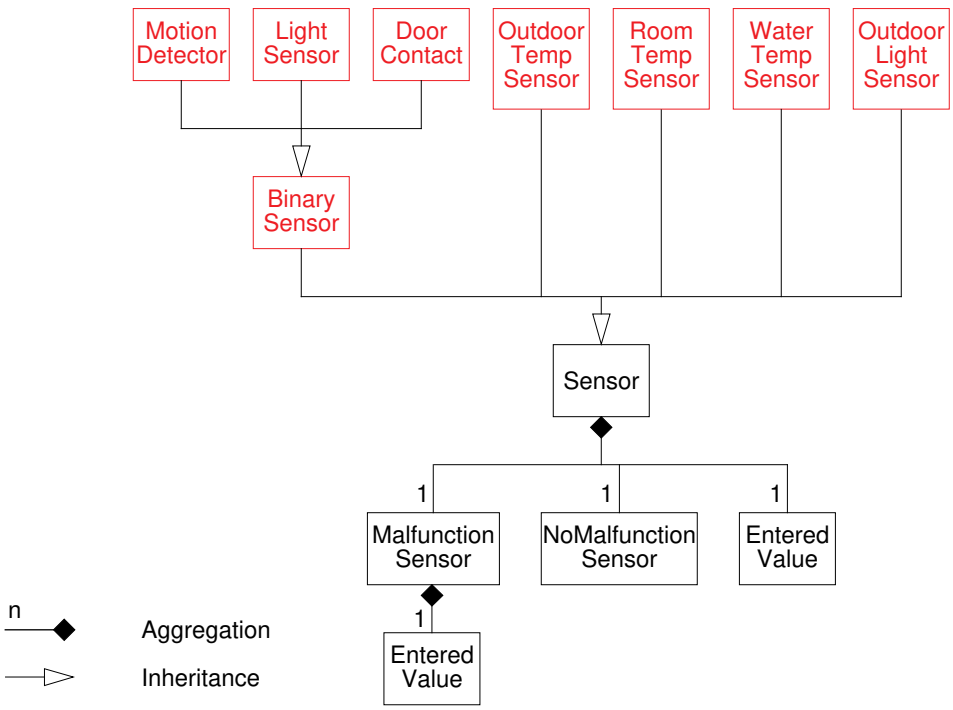

Figure 2: Class structure of group *Sensors*

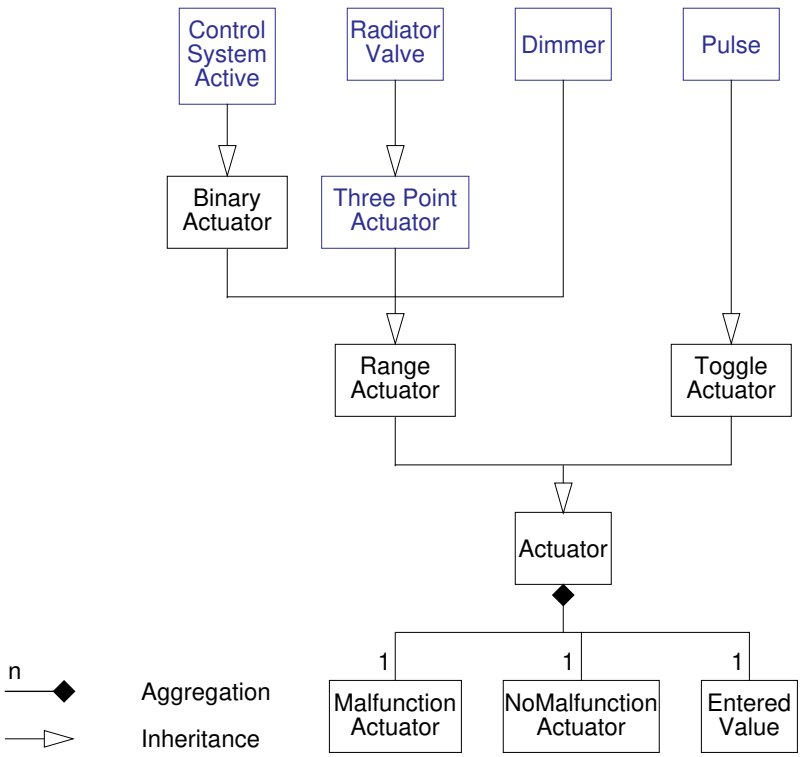

Figure 3: Class structure of group *Actuators*

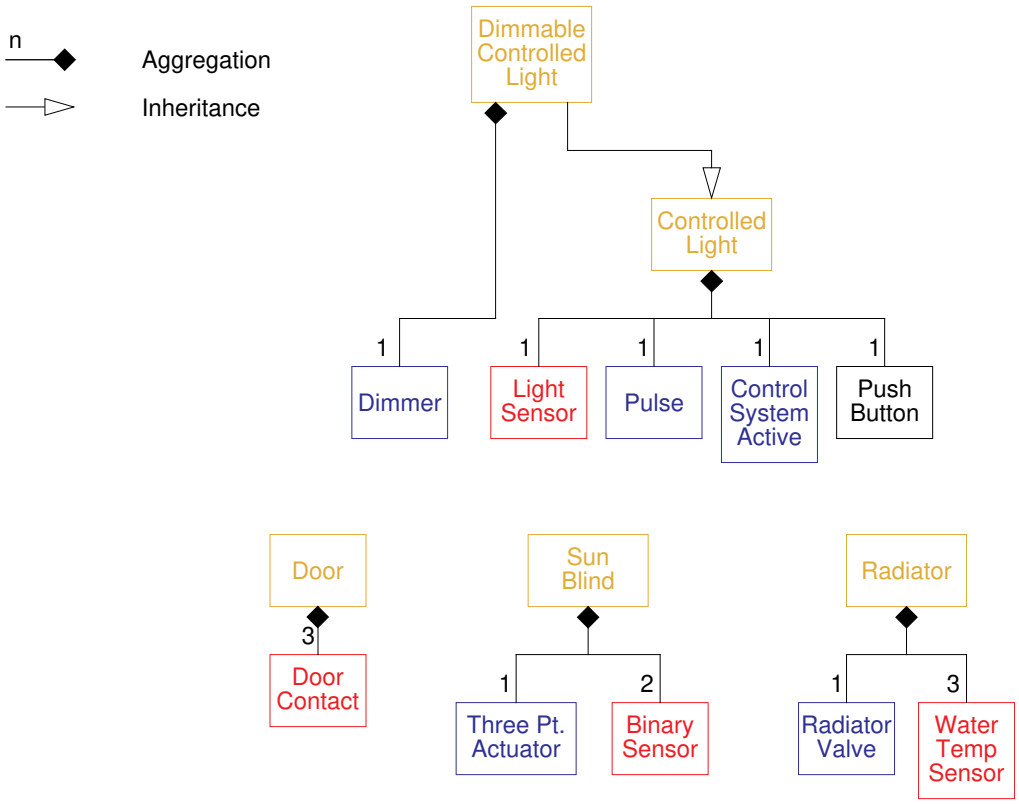

Figure 4: Class structure of group *CombinedSensorsAndActuators*

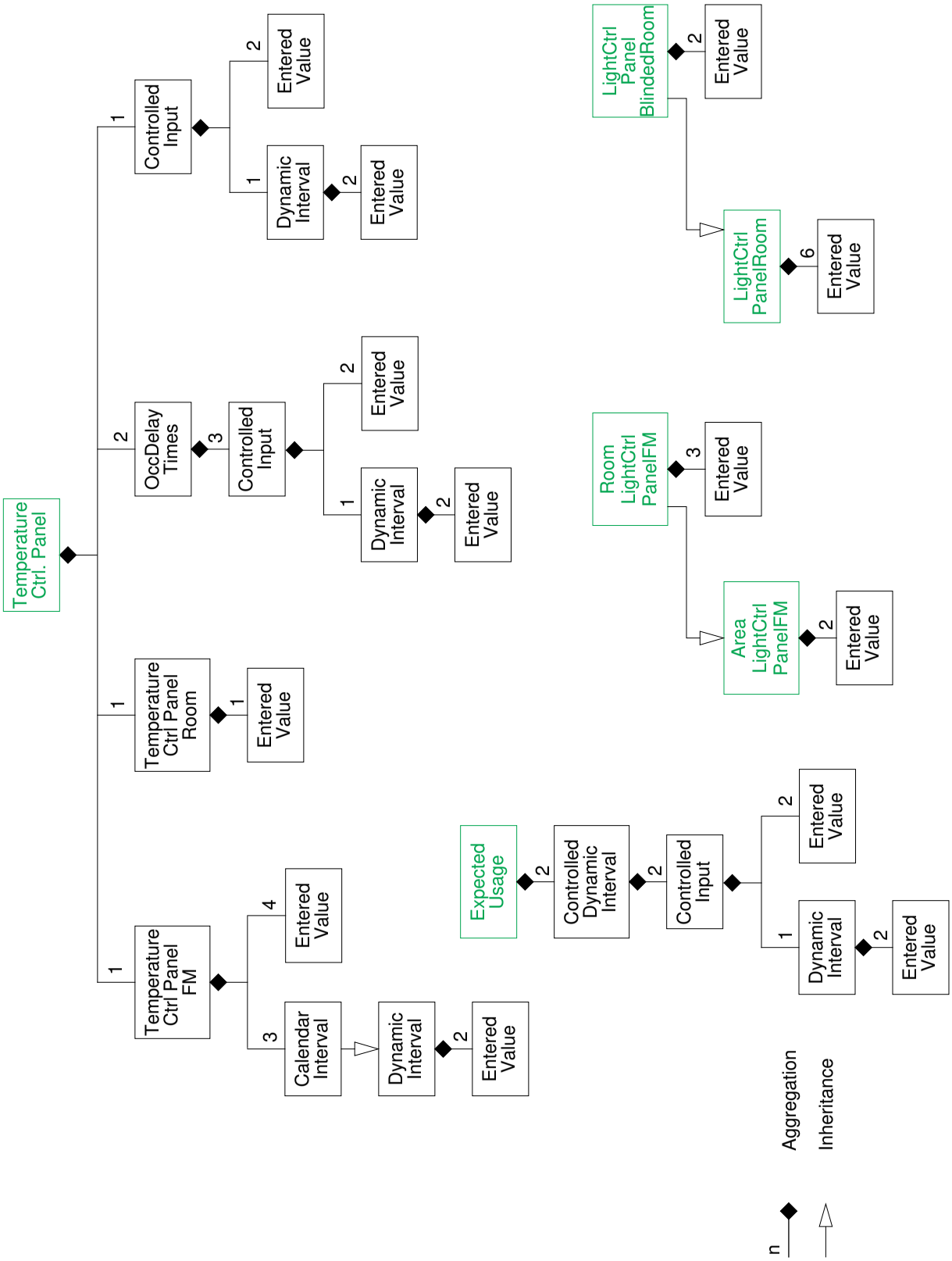

Figure 5: Class structure of group *HumanMachineInterface*

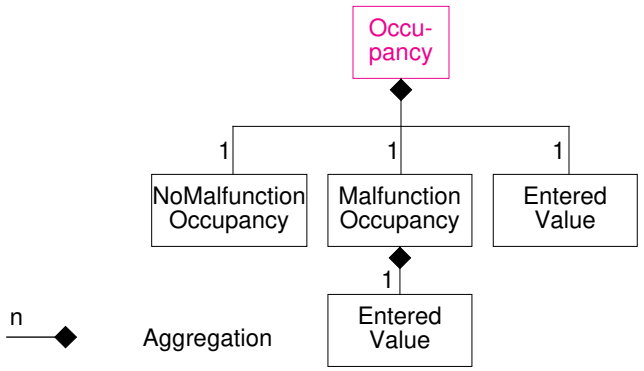

Figure 6: Class structure of group *Occupancy*

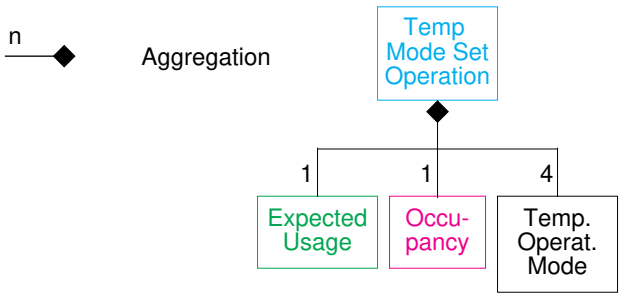

Figure 7: Class structure of group *TemperatureModes*

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*.

Description Class DynamicInterval

Intention

This class provides all definitions to describe a *dynamic interval*. An interval is a subset of a specific domain and is given by a start (object *start*) and an end point (object *end*). The domain is given by the formal parameter *INT_DOM*. Furthermore, a predicate *leq* is provided indicating whether a value of the considered domain is less or equal to another value.
Dynamic means that the values of the start and end point of the interval can change over time. Both values can be entered by a person. Therefore the start and end point are objects of the class *EnteredValue*.

Machine Specifications

Property *M_DI1*

NL : The starting point of the interval is always less than or equal to the end point of the interval.

Description Class CalendarInterval

Intention

This class provides all definitions to describe an interval of calendar times. The start and end point of the interval can change over time. Therefore, this class is derived from the description class *DynamicInterval*. Additionally, a timed predicate *actual* indicates whether the current date (*currentDate*) is in this interval or not.

Requirement Specifications

Property *R_CI1*

NL : The current time is in the calendar interval iff the interval start point is less than or equal to the current time and the current time is less than or equal to the interval end point.

Description Class ControlledInput

Intention

This class contains all the definitions that are necessary to model the *controlled* input of a value by a user. This value is represented by the timed function *value*. *Controlled* means here that the facility manager prescribes an interval in which the value entered by a user has to be. Furthermore, the facility manager prescribes a default value (object *default*) which is valid as long as a user has not entered a value. As soon as a user has entered a value this becomes the actual value unless a user changes it by entering a new value. There is no other possibility to change the value.

Machine Specifications

Property *I_CIN1*

NL : Initially a value has not been entered by a user.

Property *I_CIN2*

NL : The initially valid value is equal to the initial default value given by the facility manager.

Property *M_CIN1*

NL : If a user has entered a value once, from this time point it will be valid that a value has been entered by a user.

Property *M_CIN2*

NL : If a user has entered a value which is less than or equal to the lower bound of the allowed interval specified by the facility manager the currently valid value is equal to the lower bound of the interval specified by the facility manager.
If a user has entered a value which is greater than the upper bound of the allowed interval specified by the facility manager the currently valid value is equal to the upper bound of the interval specified by the facility manager.
Otherwise, if the entered value is within the interval specified by the facility manager, the currently valid value is equal to the value entered by a user.

Property *M_CIN3*

NL : The default value given by the facility manager has always to be within the allowed range specified by the facility manager.

Description Class ControlledDynamicInterval

Intention

This class provides all definitions to describe a *controlled dynamic interval*. An interval is a subset of a specific domain and is given by a start (object *start*) and an end point (object *end*). The domain is given by the formal parameter *INT_DOM*. Furthermore, a predicate *leq* is provided indicating whether a value of the considered domain is less or equal to another value.
Dynamic means that the values of the start and end point of the interval can change over time. For both values the facility manager can prescribe a default value which is valid as long as a user has not entered a value. Therefore the start and end point are objects of the class *ControlledInput*.

Machine Specifications

Property *M_CDI1*

NL : The starting point of the interval is always less than or equal to the end point of the interval.

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 RoomTempSensor

Intention

This class provides all definitions to describe the behavior of a *room temperature sensor*. This class is derived from the description class *Sensor*. The following specific parameters of a room temperature 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_RTS1*

NL : The conversion time of a room temperature sensor is 160 seconds.

Property *D_RTS2*

NL : The reaction time of a room temperature sensor is 10 seconds.

Description Class OutdoorTempSensor

Intention

This class provides all definitions to describe the behavior of a *outdoor temperature sensor*. This class is derived from the description class *Sensor*. The following specific parameters of a outdoor temperature 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_OTS1*

NL : The conversion time of an outdoor temperature sensor is 160 seconds.

Property *D_OTS2*

NL : The reaction time of an outdoor temperature sensor is 120 seconds.

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 WaterTempSensor

Intention

This class provides all definitions to describe the behavior of a *water temperature sensor*. This class is derived from the description class *Sensor*. The following specific parameters of a water temperature 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_WTS1*

NL : The conversion time of a water temperature sensor is 160 seconds.

Property *D_WTS2*

NL : The reaction time of a water temperature sensor is 10 seconds.

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 ThreePointActuator

Intention

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

- the domains of the actuator: a three point actuator is an actuator which can adjust three control states: *forward*, *backward*, and *stop*,
- 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_TPA_1 is characterized as domain knowledge. Otherwise, if the actuator does not exist, it has to be a requirement/specification.

Domain Knowledge

Property *D_TPA1*

NL : For a three point 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 RadiatorValve

Intention

This class provides all definitions to describe a radiator valve. In addition to the definitions of the description class *ToggleActuator*, here the reaction time of a radiator valve is specified.

Domain Knowledge

Property *D_RV1*

NL : The reaction time of a radiator valve is 60 seconds.

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 SunBlind

Intention

This class provides all definitions to describe a *sun blind*. A sun blind combines a *sun blind lifter* and a *blind rotator*. Both, lifter and rotator, are controlled using the same three point actuator. Additionally, a sun blind contains two sensors recognizing whether the blind has reached the upper or lower end.

To model the lifting of the blind and the rotating of the blind blades several states are introduced: *moveUp*, *moveDown*, *moveStop*, *rotateStatus*, *rotateIncrease*, *rotateDecrease*, and *rotateStop*.

Domain Knowledge

Property *D_SBL1*

NL : The blind lifter is always in exactly one of the three possible states *moveUp*, *moveDown*, *moveStop*.

Property *D_SBL2*

NL : The blind rotator is always in exactly one of the three possible states *rotateIncrease*, *rotateDecrease*, *rotateStop*.

Property *D_SBL3*

NL : If the blinds are moved down the rotating state, i.e. the angle of the blades is 0 degree.

Property *D_SBL4*

NL : If the blinds are moved up the rotating state, i.e. the angle of the blades is 180 degrees.

Property *D_SBL5*

NL : Whenever the status of the sun blind is set to stop, the blind is neither lifted in any direction nor are the blades rotated in any direction.

Property *D_SBL6*

NL : Whenever the status of the sun blind is set to FORWARD and the rotating state of the blades, i.e. the angle, is 180 degrees and the blind is not at the upper end, the blind is lifted up and the blades are not rotating.

Property *D_SBL7*

NL : Whenever the status of the sun blind is set to BACKWARD and the rotating state of the blades, i.e. the angle, is 0 degrees and the blind is not at the lower end, the blind is lifted down and the blades are not rotating.

Property *D_SBL8*

NL : Whenever the status of the sun blind is set to FORWARD and the rotating state of the blades, i.e. the angle, is not 180 degrees the blind is not lifted up or down in any direction but rotated in the direction of 180 degrees.

Property *D_SBL9*

NL : Whenever the status of the sun blind is set to BACKWARD and the rotating state of the blades, i.e. the angle, is not 0 degree the blind is not lifted up or down in any direction but rotated in the direction of 0 degrees.

Property *D_SBL10*

NonFormal : If the rotating angle of the blind blades is x degrees light can enter from an angle of x degrees from vertical.

Machine Specifications

Property *M_SBL1*

NL : Whenever the machine has set the actuator to FORWARD, i.e. to move up, for at least T_{mal} time units, and the *lowerEndContact* constantly signals for this time that the blind is still at the lower end, something is wrong.

Property *M_SBL2*

NL : Whenever the machine has set the actuator to BACKWARD, i.e. to move down, for at least T_{mal} time units, and the *upperEndContact* constantly signals for this time that the blind is still at the upper end, something is wrong.

Description Class Radiator

Intention

This class provides all the definitions to describe a radiator. A radiator consists of a radiator valve and of three water temperature sensors measuring the input, mean, and output water temperature, respectively.

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 OccDelayTimes

Intention

This class provides all definitions to describe the values of the three time delays dt_1, dt_2, dt_3, (see description class *NoMalfunctionOccupancy*) used to determine occupancy for the temperature control.

The values of the three time delays depend on the inputs given by a user and the facility manager. These time delays can be set for different expected *usage profiles* of an area (see description class *TemperatureCtrlPanel*).

This description class *OccDelayTimes* provides a general frame to deal with the three time delays in a common setting.

Machine Specifications

Property *M_ODT1*

NL : The initial default value of time dt_1 given by the facility manager is 1 minute.

Property *M_ODT2*

NL : The initial default value of time dt_2 given by the facility manager is 10 minutes.

Property *M_ODT3*

NL : The initial default value of time dt_3 given by the facility manager is 30 minutes.

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.

Property *R_LCP2*

NonFormal : Light scenes can be entered using the control panel which are stored and can be chosen.
Note that this is not the same as the currently chosen light scene.

Description Class LightCtrlPanelBlindedRoom

Intention

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

In addition to the definitions of description class *LightCtrlPanelRoom* this class provides the definitions concerning the settings for sun blinds, i.e. for each sun blind in the room the control panel contains

- a switch to signal that a sun blind is controlled manually and
- a switch to determine the setting of a sun blind if it is controlled manually.

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

Description Class TemperatureCtrlPanelFM

Intention

This class provides all the definitions of entities concerning the control of the temperature in a room for which the facility manager can enter a value using his control panel and which can not be changed by a user entering a value at the control panel in a room. A value entered by the facility manager can

- be valid for all rooms,
- be specific for each room type (office, meeting room, computer lab),
- be specific for each room.

This is mentioned in the intentions of the objects, too.

These objects are

- a possibility to determine the heating period. This object is the same for all rooms.
- a possibility to determine the non heating period. This object is the same for all rooms.
- a possibility to determine the non freeze period for the whole building. This object is the same for all rooms.
- a possibility to determine the non freeze period for a specific room. This object is specific for each room.
- a possibility to determine the temperature that should be applied if the operation mode *NonFreezeMode* is active. This object is the same for all rooms.
- a possibility to determine the temperature that should be applied if the operation mode *standbyMode* is active. This object is specific for each room type.
- a possibility to determine the temperature that should be applied if the operation mode *OffMode* is active. This object is specific for each room type.
- a possibility to determine whether the delay times used for determining occupancy are adaptive (i.e. the systems *learns* a typical user behavior) or not. This object is the same for all rooms.

Values for which the facility manager specifies a default value that can be changed by a user are considered in the description class *TemperatureCtrlPanel*.

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

Machine Specifications

Property *M_TCPF1*

NL : Initially it is not set that the delay times used for determining occupancy are adaptive.

Description Class TemperatureCtrlPanelRoom

Intention

This class provides all the definitions of entities concerning the control of the temperature in a room for which a user can enter a value using the control panel of a room and for which the facility manager can not enter a default value. This is

- a possibility to override the non-freeze operation mode set by the facility manager for a specific room.

Values for which the facility manager specifies a default value that can be changed by a user are considered in the description class *TemperatureCtrlPanel*.

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

Machine Specifications

Property *M_TCPR1*

NL : Initially no override for the non–freeze operation mode is set by the user.

Description Class TemperatureCtrlPanel

Intention

This class combines the two classes *TemperatureCtrlPanelFM* and *TemperatureCtrl-PanelRoom* resulting in a class that contains all entities concerning the control of the temperature in a room for which a value can be entered using a control panel. Besides the collection of the entities of the two mentioned classes in this class also those entities are introduced for which the facility manager can specify a default value that can be changed by a user.

This class does not contain the values that can be entered concerning the malfunction of a sensor or actuator. They are introduced in the description classes dealing with the malfunctions of sensors and actuators.

Requirement Specifications

Property *R_TCP1*

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

Description Class ExpectedUsage

Intention

This class provides all the definitions to describe whether a usage of a room is expected or not. This depends on several settings of the facility manager and the user.

The information whether a usage of a room is expected or not is needed to define different modes for the temperature control of a room.

Requirement Specifications

Property *D_EU1*

NL : A usage of a room is expected if and only if the following two conditions are fulfilled together:

- the current time is within the working time and
- the current time is not within the holiday time.

Description Class GroupOccupancy

Intention

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

Description Class AdaptiveOccupancyTimes

Intention

This class provides all the definitions to describe how the three time delays used to determine occupancy in an area are modified if the facility manager has chosen the delay times to be *adaptive*. In this case the machine *adapts* the delay times depending on the usage profile of a room.

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 TemperatureModes

Intention

This group contains all description classes that contain definitions needed to define temperature modes of a room.

Description Class TempOperationMode

Intention

This class provides all the definitions to describe a single *operation mode* of a room concerning temperature. A timed predicate is provided indicating whether the operation mode is active and a timed function represeenting the temperature that should be established if the operation mode is active.

Description Class TempOperationModeSet

Intention

This class provides all the definitions to describe all *operation modes* concerning the temperature control of a room. Note that for each room there is a specific set of operation modes.

Four operation modes are distinguished: *NonFreezeMode*, *ComfortMode*, *StandbyMode*, and *OffMode*. The definitions of these modes are given in this class. Which mode is currently active depends on three predicates: *expectedUsage*, *occUsed*, and *nonFreezeTime*. Based on the currently active mode the desired room temperature is determined.

Machine Specifications

Property *M_TOMS1*

NL : The operation mode *OffMode* is active in a room iff the following conditions are fulfilled together:

- currently it is not non-freeze operation time,
- no usage of the room is expected, and
- the room is not occupied.

Property *M_TOMS2*

NL : The operation mode *StandbyMode* is active in a room iff the following conditions are fulfilled together:

- currently it is not non-freeze operation time,
- usage of the room is expected, and
- the room is not occupied.

Property *M_TOMS3*

NL : The operation mode *ComfortMode* is active in a room iff the following conditions are fulfilled together:

- currently it is not non-freeze operation time and
- the room is occupied.

Property *M_TOMS4*

NL : The operation mode *NonFreezeMode* is active in a room iff currently it is non-freeze operation time.

Property *M_TOMS5*

NL : Whenever the operation mode *OffMode* is active the desired temperature of the room is the temperature that should be established in the operation mode *OffMode*.

Property *M_TOMS6*

NL : Whenever the operation mode *StandbyMode* is active the desired temperature of the room is the temperature that should be established in the operation mode *StandbyMode*.

Property *M_TOMS7*

NL : Whenever the operation mode *ComfortMode* is active the desired temperature of the room is the temperature that should be established in the operation mode *ComfortMode*.

Property *M_TOMS8*

NL : Whenever the operation mode *NonFreezeMode* is active the desired temperature of the room is the temperature that should be established in the operation mode *NonFreezeode*.

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.

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 : 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.

Property *M_AL7*

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_AL8*

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_AL9*

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

Property *M_AL10*

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 not been occupied during the last time T_RL_2, 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 BlindedRoomLight

Intention

This class provides all definitions to describe the control of lights common to any kind of room having sun blinds. It extends the definitions of the description class *RoomLight*. The special equippment of such rooms includes:

- *numSunBlinds* many blinds
- a control panel including possibilities to determine the settings of the sun blinds

The current setting of the sun blinds in a room intended by the light control depends on whether or not the machine assumes that there is a malfunction of the sensors needed to determine this setting.

Description Class RoomTemperature

Intention

This class provides all the definitions to describe the determination of the desired room temperature. For this the definitions of several objects are combined:

- an object representing the temperature operation modes, this object includes the occupancy determination for a room;
- an object representing the adaptive delay times used to determine occupancy;
- an object representing the control panel that a person can use to enter values controlling the computation of the desired room temperature.

Note that in this class it is not specified how the desired room temperature is achieved.

Requirement Specifications

Property *R_RT1*

NL : The value determined automatically by the machine in the case of a malfunction of some sensors used to determine occupancy concerning the temperature control depends on whether it is daytime or nighttime. The corresponding default values for the two time periods are used as the value determined automatically.

Property *R_RT2*

NonFormal : The use of solar radiation for heating during all times if heating is required or not disallowed, and the use of storage capacity of the building should always be preferred against the usage of the central heating unit.

Machine Specifications

Property *M_RT1*

NL : Non-freeze operation time is set if and only if the following conditions are fulfilled together:

- there is no user override for the non-freeze operation mode and
- – a room specific non-freeze time is set and currently valid or
  – a room specific non-freeze time is not set and the building wide non-freeze time is currently valid.

Property *M_RT2*

NL : The desired temperature in the operation mode *ComfortMode* is the current comfort temperature (depending on the values entered by the facility manager and the room user using the control panel).

Property *M_RT3*

NL : The desired temperature in the operation mode *StandbyMode* is the standby temperature set by the facility manager using the control panel.

Property *M_RT4*

NL : The desired temperature in the operation mode *OffMode* is the off mode temperature set by the facility manager using the control panel.

Property *M_RT5*

NL : The desired temperature in the operation mode *NonFreezeMode* is the non-freeze temperature set by the facility manager using the control panel.

Property *M_RT6*

NL : The value dt_1 used for determining occupancy is specified as follows:

- If the facility manager has chosen the adaptive mode for this room, dt_1 is the value computed by the adaptive learning function.
- If the facility manager has *not* chosen the adaptive mode for this room and usage of this room is expected, dt_1 is the value as it is computed by the entered values of the user and the facility manager for the case that usage of the room is expected.
- If the facility manager has *not* chosen the adaptive mode for this room and usage of this room is *not* expected, dt_1 is the value as it is computed by the entered values of the user and the facility manager for the case that usage of the room is *not* expected.

Property *M_RT7*

NL : The value dt_2 used for determining occupancy is specified as follows:

- If the facility manager has chosen the adaptive mode for this room, dt_2 is the value computed by the adaptive learning function.
- If the facility manager has *not* chosen the adaptive mode for this room and usage of this room is expected, dt_2 is the value as it is computed by the entered values of the user and the facility manager for the case that usage of the room is expected.
- If the facility manager has *not* chosen the adaptive mode for this room and usage of this room is *not* expected, dt_2 is the value as it is computed by the entered values of the user and the facility manager for the case that usage of the room is *not* expected.

Property *M_RT8*

NL : The value dt_3 used for determining occupancy is specified as follows:

- If the facility manager has chosen the adaptive mode for this room, dt_3 is the value computed by the adaptive learning function.
- If the facility manager has *not* chosen the adaptive mode for this room and usage of this room is expected, dt_3 is the value as it is computed by the entered values of the user and the facility manager for the case that usage of the room is expected.
- If the facility manager has *not* chosen the adaptive mode for this room and usage of this room is *not* expected, dt_3 is the value as it is computed by the entered values of the user and the facility manager for the case that usage of the room is *not* expected.

Property *M_RT9*

NL : The default value for occupancy concerning the temperature control during daytime in case of a malfunction of some sensors used to determine occupancy is *occupancy*.

Property *M_RT10*

NL : The default value for occupancy concerning the temperature control during nighttime in case of a malfunction of some sensors used to determine occupancy is *no occupancy*.

Property *M_RT11*

NL : The time delays used for determining occupancy of the temperature control in the case that a usage of a room is expected corresponds with those entered for this case.

Property *M_RT12*

NL : The time delays used for determining occupancy of the temperature control in the case that a usage of a room is not expected corresponds with those entered for this case.

Description Class BlindedRoomTemperature

Intention

This class provides all definitions to describe the control of the temperature common to any kind of room having sun blinds. It extends the definitions of the description class *RoomTemperature*. The special equippment of such rooms includes:

- numSunBlinds many blinds

The current setting of the sun blinds in a room intended by the temperature control depends on whether or not the machine assumes that there is a malfunction of the sensors needed to determine this setting.

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 and temperature 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 installed radiators. These objects are integrated in the derived description class *Room*.

Each room is equipped with

- one motion detector
- three temperature sensors

Machine Specifications

Property *M_BaR1*

NL : If the first room temperature sensor has a malfunction its value is estimated using the other two room temperature sensors.

Property *M_BaR2*

NL : If the second room temperature sensor has a malfunction its value is estimated using the other two room temperature sensors.

Property *M_BaR3*

NL : If the third room temperature sensor has a malfunction its value is estimated using the other two room temperature sensors.

Property *M_BaR4*

NL : All room temperature sensors have a malfunction iff the machine assumes for each room temperature sensor a malfunction.

Property *M_BaR5*

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.

Property *M_BaR6*

NL : Concerning the temperature 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 instantion can vary from room to room.

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

- *numRad* many radiators
- *numDoorHallway* many doors to the hallway

Requirement Specifications

Property *R_R1*

NonFormal : The desired temperature shall be reached as fast as possible during the heating up state.
This requirement is a basis for the definition of the functions *tempRadNoMalSet* and *tempRadMalSet*.

Property *R_R2*

NonFormal : The currently desired temperature is maintained during the current temperature state as good as possible.
This requirement is a basis for the definition of the functions *tempRadNoMalSet* and *tempRadMalSet*.

Machine Specifications

Property *I_R1*

NL : Initially the machine does not assume that the sensors, used to determine the settings of the actuators controlling the temperature, have such a serious malfunction that the settings of the actuators have to be determined using makeshift operations.

Property *M_R1*

NL : The machine assumes that the sensors, used to determine the settings of the actuators controlling the temperature, have such a serious malfunction that the settings of the actuators have to be determined using makeshift operations iff all temperature sensors of a room have a malfunction.

Property *M_R2*

NL : If the machine does not assume a serious malfunction of the sensors, used to determine the settings of the actuators controlling the temperature, the current setting of the radiators in a room intended by the temperature control are as determined by the function *tempRadNoMalSet*.
If the machine assumes a serious malfunction of the sensors, used to determine the settings of the actuators controlling the temperature, the current setting of the radiators in a room intended by the temperature control are as determined by the function *tempRadMalSet*.

Property *M_R3*

NL : The value adjusted by the machine for the radiator valves corresponds with the value intended by the temperature control for these actuators depending on the currently desired temperature.
Note that all radiator valves in a room are always set to the same value. A specific value for a single radiator is not determined.

Description Class BlindedRoom

Intention

This class provides all definitions specific to any kind of room with sun blinds. In addition to the definitions of the description class *Room* the control of the sun blinds is considered here. Since both, light and temperature control, can influence the setting of the sun blinds there can be conflicts. The resolution of possible conflicts is represented by the function *resolveBlindConflict*.

Domain Knowledge

Property *D_BR1*

NL : The number of sun blinds in a blinded room equals the number of radiators in a blinded room.

Machine Specifications

Property *I_BR1*

NL : Initially the sun blinds are not controlled manually.

Property *M_BR1*

NL : The machine assumes that the sun blinds are controlled manually iff a room is occupied according to the definition of this term used for temperature control and at least one sun blind is controlled manually.

Property *M_BR2*

NL : If the machine does not assume a serious malfunction of the sensors, used to determine the settings of the actuators controlling the temperature, the current settings of the sun blinds in a room intended by the temperature control are as determined by the function *tempBlindNoMalSet*.
If the machine assumes a serious malfunction of the sensors, used to determine the settings of the actuators controlling the temperature, the current settings of the sun blinds in a room intended by the temperature control are as determined by the function *tempBlindMalSet*.

Property *M_BR3*

NL : If the machine does not assume a serious malfunction of the sensors, used to determine the settings of the actuators controlling the light, the current settings of the sun blinds in a room intended by the light control are as determined by the function *lightBlindNoMalSet*.
If the machine assumes a serious malfunction of the sensors, used to determine the settings of the actuators controlling the light, the current settings of the sun blinds in a room intended by the light control are as determined by the function *lightBlindMalSet*.

Property *M_BR4*

NL : If the sun blinds are controlled manually the value adjusted by the machine for each sun blind is equal to the corresponding switch status of the control panel.

Property *M_BR5*

NL : If the sun blinds are not controlled manually the value adjusted by the machine for each sun blind is equal to the value determined for the sun blinds by the function *resolveBlindConflict*.

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 having no sun blinds. 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 installed moveably like a telephone in the offices.

Description Class OfficeNeighb

Intention

This class provides all definitions specific to all offices having no sun blinds and 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 BlindedOffice

Intention

This class provides all definitions specific to all offices with sun blinds and which are not connected by a door to a neighbour room. In addition to the definitions of the description class *Room* only the installation of the control panel is considered here.

Requirement Specifications

Property *R_BO1*

NonFormal : The control panel should be installed movable like a telephone in an office.

Description Class BlindedOfficeNeighb

Intention

This class provides all definitions specific to all offices with sun blinds and which are connected by at least one door to a neighbour room. In addition to the definitions of the description class *BlindedOffice* 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 and that have no sun blinds. 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 BlindedNoOffice

Intention

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

Requirement Specifications

Property *R_BNO1*

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

Description Class BlindedNoOfficeNeighb

Intention

This class provides all definitions specific to rooms that are no offices and that have sun blinds and which are connected by at least one door to a neighbour room. In addition to the definitions of the description class *BlindedNoOffice* 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.

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 : Energy consumption should be reduced as far as possible during maintaining of comfortable working conditions for all users.

Property *R_F3*

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

Property *R_F4*

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

Property *R_F5*

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

Property *R_F6*

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

Property *R_F7*

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

Property *R_F8*

NonFormal : The system warns about unreasonable inputs.

Property *R_F9*

NonFormal : If the whole control system has a serious malfunction, complete manual operation should be possible.

Property *R_F10*

NonFormal : The control system shall be designed such that it can be easily changed and expanded.

Property *R_F11*

NonFormal : The control system provides basic energy saving procedures automatically for control comfort. If the facility manager deviates from such settings, the system provides a report on expected results.

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.

Property *M_F3*

NL : The heating period is the same for all rooms.

Property *M_F4*

NL : The period within which the non freeze mode is set for the whole building is the same for all rooms.

Property *M_F5*

NL : The temperature that should be applied if the operation mode *Non-FreezeMode* is active is the same for all rooms.

Property *M_F6*

NL : The modus set by the facility manager indicating whether the delay times used for determining occupancy are adaptive or not is always the same for all rooms.

Property *M_F7*

NL : The temperature that should be applied in the *standbyMode* is the same for all offices.

Property *M_F8*

NL : The temperature that should be applied in the *OffMode* is the same for all offices.

Property *M_F9*

NL : The temperature that should be applied in the *standbyMode* is the same for all meeting rooms.

Property *M_F10*

NL : The temperature that should be applied in the *OffMode* is the same for all meeting rooms.

Property *M_F11*

NL : The temperature that should be applied in the *standbyMode* is the same for all computer labs.

Property *M_F12*

NL : The temperature that should be applied in the *OffMode* is the same for all computer labs.

Index

A

Actuator 23, 27
AdaptiveOccupancyTimes 56
AreaLight 65
AreaLightCtrlPanelFM 46

B

BasicHumanMachineInterface 4
BasicRoom 77
BinaryActuator 32
BinarySensor 19
BlindedNoOffice 88
BlindedNoOfficeNeighb 89
BlindedOffice 85
BlindedOfficeNeighb 86
BlindedRoom 80
BlindedRoomLight 71
BlindedRoomTemperature 76
BuildingUnits 64

C

CalendarInterval 7
CombinedSensorsAndActuators 37
ControlledDynamicInterval 9
ControlledInput 8
ControlledLight 42
ControlSystemActive 33

D

D_AL1 65
D_AL2 65
D_BA1 32
D_BR1 80
D_BS1 19
D_BS2 19
D_BS3 19
D_CL1 42
D_CL2 42
D_CSA1 33
D_DC1 21
D_DIM1 31
D_EU1 54
D_F1 91
D_F2 91
D_F3 91
D_F4 91
D_F5 91
D_F6 91
D_L1 20
D_MD1 22
D_NMS1 11
D_NMS2 11
D_OLS1 17
D_OLS2 17
D_OTS1 16
D_OTS2 16
D_PB1 36
D_PB2 36
D_PUL1 29
D_RA1 30
D_RTS1 15
D_RTS2 15
D_RV1 35
D_SBL1 39
D_SBL10 40
D_SBL2 39
D_SBL3 39
D_SBL4 39
D_SBL5 39
D_SBL6 39
D_SBL7 39
D_SBL8 40
D_SBL9 40
D_TA1 28
D_TA2 28
D_TPA1 34
D_WTS1 18
D_WTS2 18
Datatypes 1
DimmableControlledLight 43
Dimmer 31
Door 38
DoorContact 21
DynamicInterval 6

E
EnteredValue 5
ExpectedUsage 54

F
Floor 91

G
GroupOccupancy 55

H
HallwaySection 67
HumanMachineInterface 44

I
I_A1 27
I_AL1 65
I_AL2 66
I_BR1 80
I_CIN1 8
I_CIN2 8
I_CL1 42
I_LCPFM1 46
I_LCPFM2 46
I_LCPR1 47
I_MA1 25
I_MA2 25
I_MO1 58
I_MS1 13
I_MS2 12
I_MS3 12
I_OCC1 59
I_R1 78
I_RL1 68
I_RL2 68
I_RL3 68
I_S1 14

L
LightCtrlPanelBlindedRoom 49
LightCtrlPanelRoom 48
LightScene 2
LightSceneOffice 3
LightSensor 20

M
M_A1 27
M_AL1 66
M_AL10 66
M_AL2 66
M_AL3 66
M_AL4 66
M_AL5 66
M_AL6 66
M_AL7 66
M_AL8 66
M_AL9 66
M_BaR1 77
M_BaR2 77
M_BaR3 77
M_BaR4 77
M_BaR5 77
M_BaR6 77
M_BR1 80
M_BR2 80
M_BR3 80
M_BR4 81
M_BR5 81
M_CDI1 9
M_CIN1 8
M_CIN2 8
M_CIN3 8
M_CL1 42
M_CL2 42
M_CL3 42
M_CL4 42
M_DI1 6
M_F1 92
M_F10 93
M_F11 93
M_F12 93
M_F2 92
M_F3 92
M_F4 93
M_F5 93
M_F6 93
M_F7 93
M_F8 93
M_F9 93
M_HS1 67
M_HS2 67
M_HS3 67

M_LCPR1 47
M_LS1 2
M_LSO1 3
M_MO1 58
M_MS1 13
M_NMO1 57
M_OCC1 59
M_OCC2 59
M_ODT1 45
M_ODT2 45
M_ODT3 45
M_R1 78
M_R2 79
M_R3 79
M_RL1 68
M_RL10 70
M_RL11 70
M_RL12 70
M_RL13 70
M_RL14 70
M_RL2 69
M_RL3 69
M_RL4 69
M_RL5 69
M_RL6 69
M_RL7 69
M_RL8 69
M_RL9 69
M_RT1 72
M_RT10 74
M_RT11 75
M_RT12 75
M_RT2 72
M_RT3 73
M_RT4 73
M_RT5 73
M_RT6 73
M_RT7 74
M_RT8 74
M_RT9 74
M_S1 14
M_S2 14
M_S3 14
M_SBL1 40
M_SBL2 40
M_TCPF1 51
M_TCPR1 52
M_TOMS1 62
M_TOMS2 62
M_TOMS3 62
M_TOMS4 62
M_TOMS5 63
M_TOMS6 63
M_TOMS7 63
M_TOMS8 63
MalfunctionActuator 25
MalfunctionOccupancy 58
MalfunctionSensor 12
MotionDetector 22

N

NoMalfunctionActuator 24
NoMalfunctionOccupancy 57
NoMalfunctionSensor 11
NoOffice 87

O

OccDelayTimes 45
Occupancy 59
Office 83
OfficeNeighb 84
OutdoorLightSensor 17
OutdoorTempSensor 16

P

Pulse 29
PushButton 36

R

R_AL1 65
R_AL2 65
R_BNO1 88
R_BO1 85
R_CI1 7
R_EV1 5
R_F1 91
R_F10 92
R_F11 92
R_F2 91
R_F3 92
R_F4 92

R_F5 92
R_F6 92
R_F7 92
R_F8 92
R_F9 92
R_HS1 67
R_HS2 67
R_LCP1 48
R_LCP2 48
R_MA1 25
R_MA2 25
R_MA3 25
R_MA4 25
R_MA5 25
R_MA6 26
R_MS1 12
R_MS2 12
R_MS3 12
R_MS4 12
R_MS5 12
R_MS6 13
R_NO1 87
R_O1 83
R_OLS1 17
R_R1 78
R_R2 78
R_RL1 68
R_RL2 68
R_RT1 72
R_RT2 72
R_TCP1 53
Radiator 41
RadiatorValve 35
RangeActuator 30
Room 78
RoomLight 68
RoomLightCtrlPanelFM 47
RoomNeighb 82
RoomTemperature 72
RoomTempSensor 15

S

Sensor 14
Sensors 10
StairCase 90
SunBlind 39

T

TemperatureCtrlPanel 53
TemperatureCtrlPanelFM 50
TemperatureCtrlPanelRoom 52
TemperatureModes 60
TempOperationMode 61
TempOperationModeSet 62
ThreePointActuator 34
ToggleActuator 28

W

WaterTempSensor 18