Table of Contents

SDT	rw	/SFB_501/teams/team3/Basis−Projekt.BaX/Entwicklung/RequirementsSp
	rw	/SFB_501/teams/team3/Basis−Projekt.BaX/Entwicklung/RequirementsSp

Used Packages

DataTypes	r−	../Datatypes/LatestRev/DataTypes.sun
Utils	r−	../Utils/LatestRev/Utils.sun
Sensors	r−	../Sensors/LatestRev/Sensors.sun
ODLSens		[unconnected]
MotDet		[unconnected]
Switch		[unconnected]
SunDet		[unconnected]
Actuators	r−	../Actuators/LatestRev/Actuators.sun
TaskLight		[unconnected]
Dimmer		[unconnected]
PulsGen		[unconnected]
CP1	r−	../Control−sub1/LatestRev/CP1.sun
Section1		[unconnected]
Section2		[unconnected]
Section3		[unconnected]
CP2	r−	../Control−sub2/LatestRev/CP2.sun
Door		[unconnected]
RoomOcc		[unconnected]
Desk		[unconnected]
Light		[unconnected]
Office2D		[unconnected]
Office1D		[unconnected]
CompLab2D		[unconnected]
CompLab1D		[unconnected]
PRoom		[unconnected]
MRoom		[unconnected]
CP3	r−	../Control−sub3/LatestRev/CP3.sun
Hallway		[unconnected]
CP4		../Control−sub4/LatestRev/CP4

CP4	r−	../Control−sub4/LatestRev/CP4.sun
Floor		[unconnected]
CP5	r−	../Control−sub5/LatestRev/CP5.sun
ControlPackage		
Control	rw	Control.sun

ecification/Control/Control.sdt

ecification/Control/

Package Datatypes


```
SIGNAL dummy;
SIGNAL ident (PID, CharString);
```

```
SYNTYPE Analog=Real  ENDSYNTYPE;
SYNTYPE Binary=Integer CONSTANTS 0,1 ENDSYNTYPE;
SYNTYPE Percent=Real CONSTANTS 0:100 ENDSYNTYPE;
SYNTYPE Day=Integer CONSTANTS 1:365 ENDSYNTYPE;
SYNTYPE WkDay=Integer CONSTANTS 1:7 ENDSYNTYPE;
SYNTYPE Celsius=Real CONSTANTS -60:100 ENDSYNTYPE;
SYNTYPE Lux=REAL CONSTANTS -1:120000 ENDSYNTYPE;
SYNTYPE HallwayIdType = Integer CONSTANTS 1:3 ENDSYNTYPE;
SYNTYPE DegF=Real CONSTANTS -60:200 ENDSYNTYPE; /*Fahrenheit*/
SYNTYPE GPM=Real CONSTANTS 0:200 ENDSYNTYPE; /*Gallon per minute*/
SYNTYPE Psi=Real CONSTANTS 0:200 ENDSYNTYPE; /*Pound per square inch*/
SYNTYPE Watt=Real CONSTANTS 0:10000 ENDSYNTYPE; /*Power unit*/
SYNTYPE ErrorType = INTEGER CONSTANTS 0:100 ENDSYNTYPE;
SYNTYPE Joule = REAL CONSTANTS 0:3000000 ENDSYNTYPE;
SYNTYPE DayTime = INTEGER ENDSYNTYPE;
SYNTYPE LightSourceType = INTEGER CONSTANTS 1:3 ENDSYNTYPE;
```

```
NEWTYPE RoomIdType
struct
rId CHARSTRING;
sId CHARSTRING;
ENDNEWTYPE;
```

```
NEWTYPE LightSourceListType
ARRAY (INTEGER, LightSourceType)
ENDNEWTYPE;
```

```
/* Fehlt: Die Prioritaetsliste fuer die Lichtquellen */
NEWTYPE LightScene
struct
name CharString;
rIll Lux;
dIll Lux;
ls LightSourceListType;
ENDNEWTYPE;
```

```
NEWTYPE CalTimeType
struct
year, month, day, hour, min, sec INTEGER;
operators
   toTime: CalTimeType -> Time;
   fromTime: Time->CalTimeType;
/*#ADT (B)*/
ENDNEWTYPE;
```

Package Datatypes

SIGNAL getIlluminance ;
SIGNAL newIlluminance (RoomIdType, Lux);

SIGNAL newHwEnergyConsumption (Joule, Duration);
SIGNAL newHwMalFunction(Errortype);
SIGNAL getHwPBDelay;
SIGNAL getHwOccDelay;

SIGNAL setHwEnergyDiffTime(Duration);
SIGNAL setHwLight(Boolean);
SIGNAL setHwPbDelay(Duration);
SIGNAL setHwOccDelay (Duration);

SIGNAL setRoomDelay (RoomIdType, Duration);
SIGNAL setRoomReoccDelay (RoomIdType, Duration);
SIGNAL setDefaultLightScene (RoomIdType, LightScene);
SIGNAL setAllDefaultLightScene (LightScene);
SIGNAL setDefaultAmbientLightLevel (RoomIdType, Lux);
SIGNAL setRoomLight (RoomIdType, Boolean);
SIGNAL setRoomEnergyDiffTime (Duration);
SIGNAL newRoomMalfunction (RoomIdType, ErrorType);
SIGNAL newRoomEnergyConsumption (RoomIdType, Joule, Duration);

SIGNAL newEntrance;

```
USE DataTypes;
```

Package Utils

1(2)

```
/*setxxValue: root sets value in remote node*/
/*pollxxValue: root polls value of remote node*/
/*putxxValue: remote node puts value into root*/

REMOTE PROCEDURE setPhysVal; FPAR IN  Analog;
REMOTE PROCEDURE putAnalogVal; FPAR IN  PID, Analog;
REMOTE PROCEDURE putBinaryVal; FPAR IN  PID, Binary;
REMOTE PROCEDURE pollPhysVal; FPAR IN  PID;
```

```
procedure toJoule;
fpar IN w Watt,
IN dimV Percent,
IN timeDiff Duration;
RETURNS Joule;
EXTERNAL;
```

```
/* Proz zum Ausdruck eines beliebigen
Binaerwerts mit der aktuellen Zeit */
procedure printBi;
fpar IN c Charstring,
IN b Binary,
IN dur Time;
EXTERNAL;
```

```
/* Proz zum Ausdruck eines beliebigen
Anaolgwerts mit der aktuellen Zeit */
procedure printAn;
fpar IN c CharString,
IN a Analog,
IN dur Time;
EXTERNAL;
```

```
/* Proz zum Umformen der Kalendereit in die aktuelle Zeit*/
/*procedure calTimeToTime;
fpar IN actTime calTimeType;
RETURNS Time;
EXTERNAL;*/
```

```
/* Proz zum Umformen der aktuellen Zeit in Kalenderzeit*/
/*procedure timeToCalTime;
fpar IN actTime Time;
RETURNS CalTimeType;
EXTERNAL;*/
```

```
/* Proz zum Ausdruck eines beliebigen
Strings mit der aktuellen Zeit */
procedure printIt;
fpar IN c CharString,
IN dur Time;
EXTERNAL;
```

USE DataTypes;

Package Utils


```
/* Proz zum Verbindungsaufbaus des Systems
mit der Umwelt (physSock)*/
procedure initSockets;
EXTERNAL;
```

```
/* Proz., die periodisch einen eventuellen Eingang an dem beiden
Socket physSock bearbeitet.*/
 procedure checkSockets;
returns INTEGER;
EXTERNAL;
```

```
/* Proz., die einen binaeren physik. Werte setzen.*/
procedure setBinOutVal;
FPAR IN actId CharString,
IN val Binary;
EXTERNAL;
```

```
/* Proz., die einen analogen physik. Werte setzen.*/
procedure setAnOutVal;
FPAR IN actId CharString,
IN val Analog;
EXTERNAL;
```

```
/* Proz., die einen physik. Werte anfordert.*/
procedure reqPhysVal;
FPAR IN actId CharString;
EXTERNAL;
```

USE Datatypes;
USE Utils;

Package Sensors

ODLSens

```
/* Signals at object input oi */
SIGNAL getOls;  /*  Get value, independent of mode  */
SIGNAL setOlsPeriod(Duration) ; /* Send value periodically with period p  */
SIGNAL setOlsEvent (Lux); /* Send value if changed  */
SIGNAL setOlsPoll ; /* Wait for getValue signal  */
SIGNALLIST  ODLSensInp=getOls, setOlsPeriod, setOlsEvent, setOlsPoll; /*    */
/* Signals at object output oo */
SIGNAL  newOls(Lux) ; /*    */
SIGNAL newODLMal;
SIGNALLIST  ODLSensOutp=ident,  newOls, newODLMal ; /*    */
```

MotDet

```
/* Signals at object input oi */
SIGNAL getMotDet ;  /*  Get value, independent of mode  */
SIGNAL setMDPeriod(Duration) ; /* Send value periodically with period p  */
SIGNAL setMDEvent ; /* Send value if changed  */
SIGNAL setMDPoll ; /* Wait for getValue signal  */
SIGNALLIST  MotDetInp=getMotDet, setMDPeriod, setMDEvent,  setMDPoll ; /*    */
/* Signals at object output oo */
SIGNAL  newMotDet(Binary) ; /*    */
SIGNAL newMotDetMal (ErrorType);
SIGNALLIST  MotDetOutp=ident, newMotDet, newMotDetMal; /*    */
```

Switch

```
/* Signals at object input oi */
SIGNAL getSwitch ;  /*  Get value, independent of mode  */
SIGNAL setSwPeriod(Duration) ; /* Send value periodically with period p  */
SIGNAL setSwEvent ; /* Send value if changed  */
SIGNAL setSwPoll ; /* Wait for getValue signal  */
SIGNALLIST  SwitchInp=getSwitch, setSwPeriod, setSwEvent,  setSwPoll ; /*    */
/* Signals at object output oo */
SIGNAL  newSwitch(Binary) ; /*    */
SIGNAL newSwitchMal (ErrorType);
SIGNALLIST  SwitchOutp=ident, newSwitch, newSwitchMal; /*    */
```

SunDet

```
SIGNAL getSun (RoomIdType);
SIGNAL newSunDetMal (ErrorType);
SIGNAL newSun (RoomIdType, Lux);
SIGNALLIST SunDetInp = getSun;
SIGNALLIST SunDetOutp = ident, newSun, newSunDetMal;
```

```
USE DataTypes;
USE Utils;
```

Package Actuators

1(1)

TaskLight

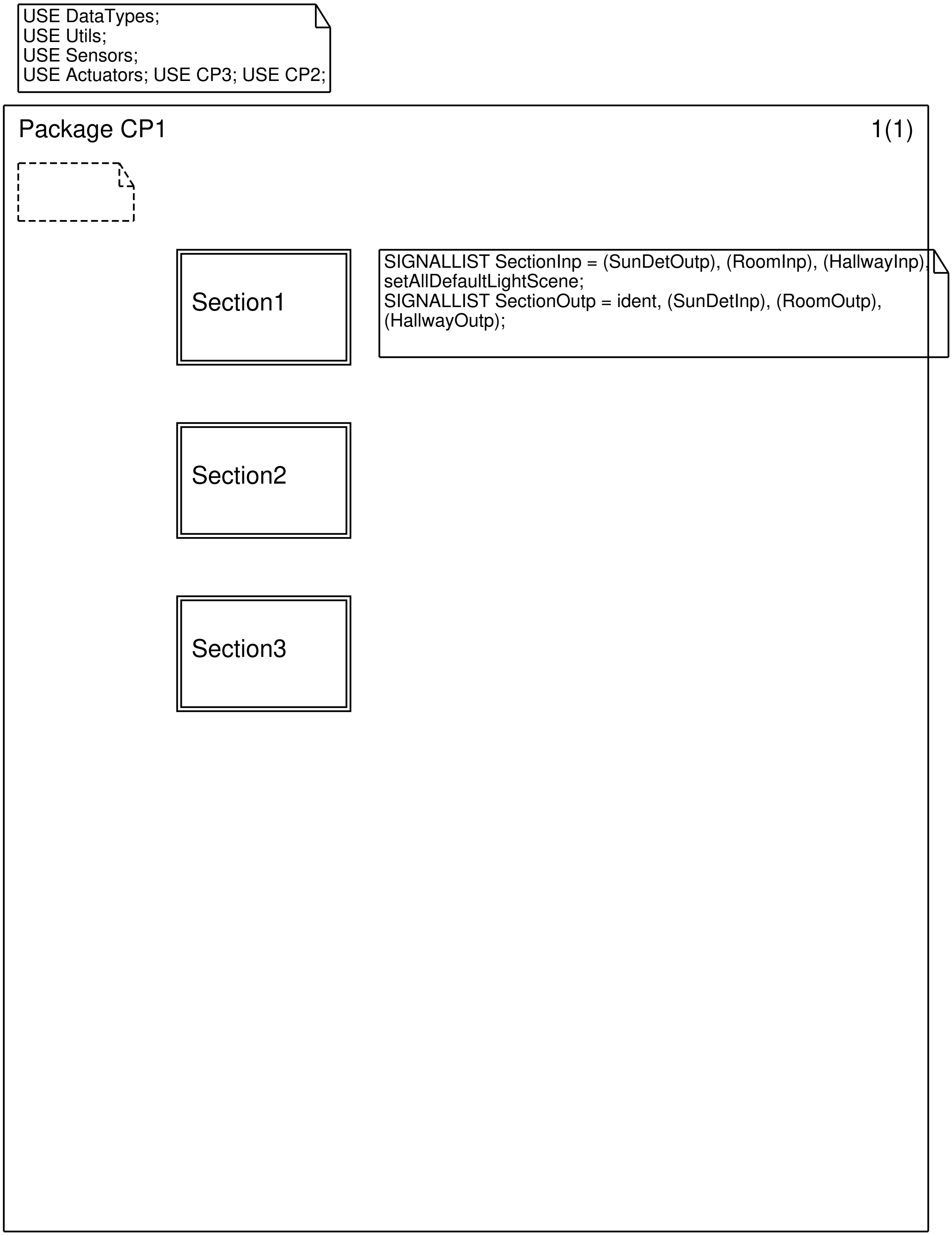
```
/* Signals at object input oi */
SIGNAL setTL(Binary);
SIGNALLIST  TaskLightInp= setTL; /*   */
SIGNALLIST TaskLightOutp=ident;
```

Dimmer

```
/* Signals at object input oi */
SIGNAL setDV(Percent);
SIGNALLIST  DimmerInp= setDV; /*   */
SIGNALLIST DimmerOutp=ident;
```

PulsGen

```
/* Signals at object input oi */
SIGNAL setPG(Binary);
SIGNALLIST  PulsGenInp= setPG; /*   */
SIGNALLIST PulsGenOutp=ident;
```


USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators; USE CP3; USE CP2;
Package CP1
1(1)
Section1
SIGNALLIST SectionInp = (SunDetOutp), (RoomInp), (HallwayInp), setAllDefaultLightScene;
SIGNALLIST SectionOutp = ident, (SunDetInp), (RoomOutp), (HallwayOutp);
Section2
Section3

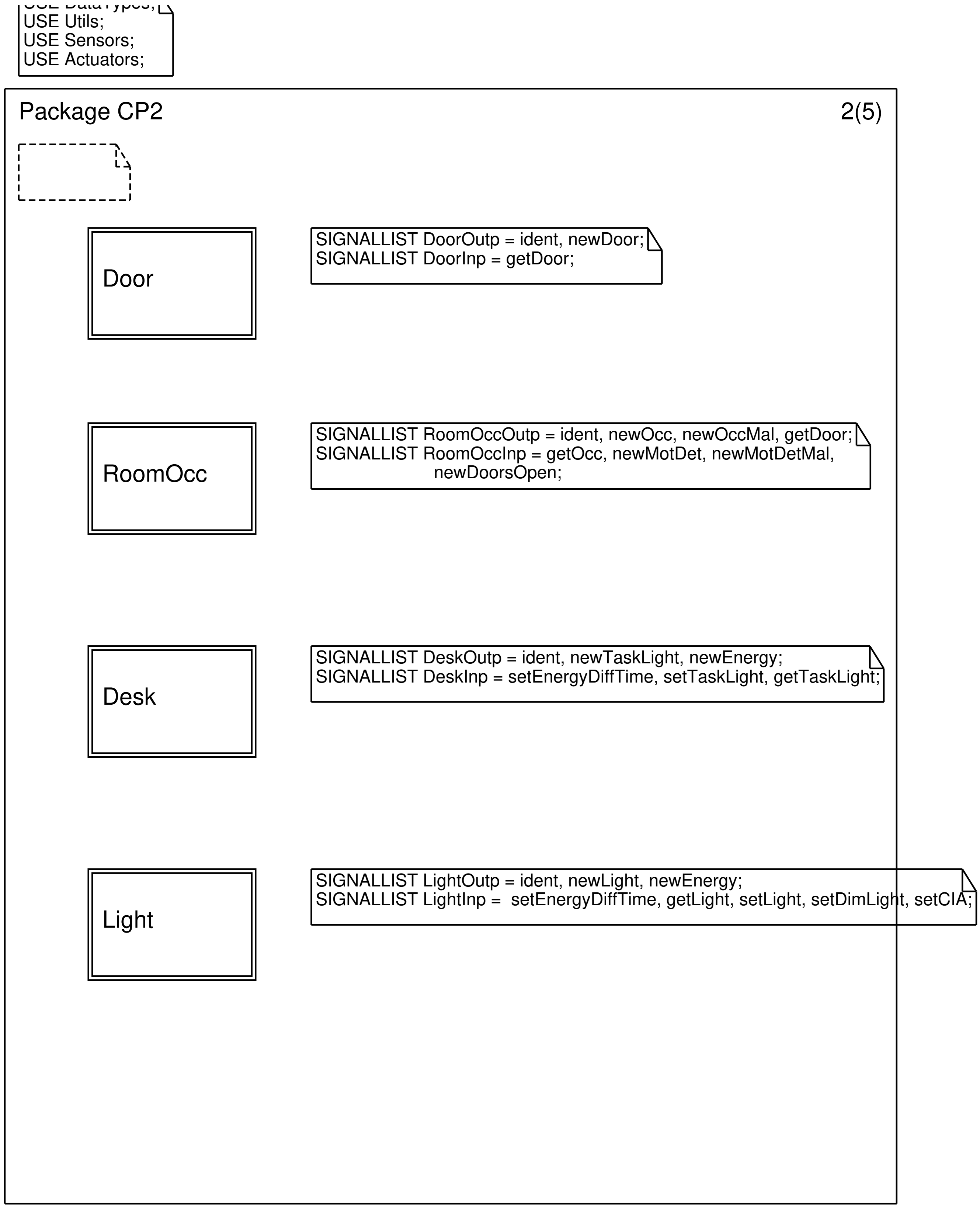
```
USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators;
```

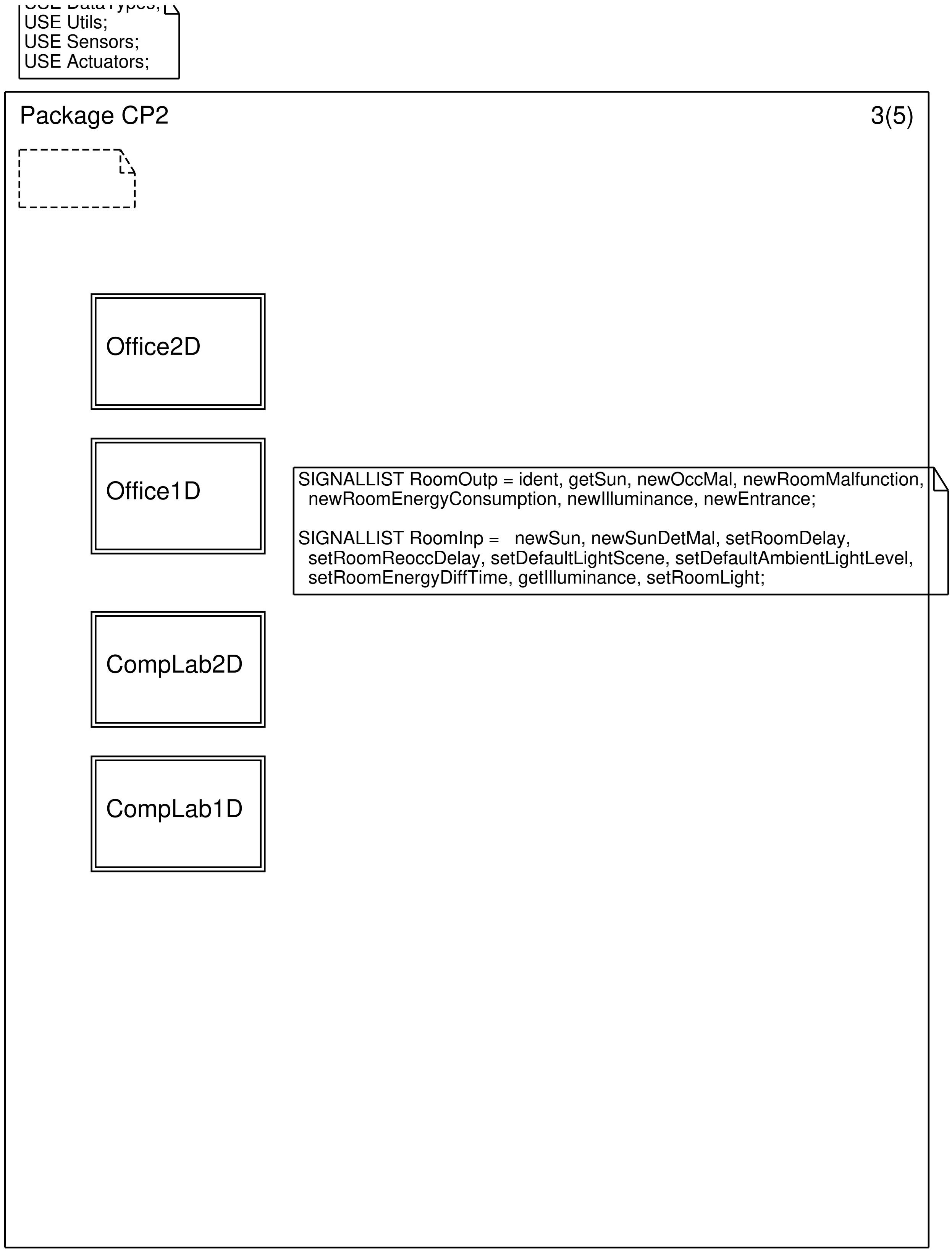
Package CP2


```
/* Signal Declarations */


/* ---- Door ----*/
SIGNAL newDoor (Boolean);
SIGNAL getDoor;
/* ---- RoomOcc ---- */
SIGNAL newOcc (Boolean);
SIGNAL newOccMal (ErrorType);
SIGNAL newDoorsOpen (Boolean);
/* ---- Desk ---- */
SIGNAL newTaskLight (Boolean);
SIGNAL setTaskLight (Boolean);
SIGNAL getTaskLight;
/* ---- Rooms ---- */
SIGNAL setDimLight (Percent);
SIGNAL setLight (Boolean);
SIGNAL getLight;
SIGNAL getDimLight;
SIGNAL setCIA;
SIGNAL getOcc;
SIGNAL newLight (Boolean);
SIGNAL newEnergy (Joule);
SIGNAL setEnergyDiffTime (Duration);

/* ---- Light ----*/
```


USE Utils;
USE Sensors;
USE Actuators;
Package CP2
2(5)
Door
SIGNALLIST DoorOutp = ident, newDoor;
SIGNALLIST DoorInp = getDoor;
RoomOcc
SIGNALLIST RoomOccOutp = ident, newOcc, newOccMal, getDoor;
SIGNALLIST RoomOccInp = getOcc, newMotDet, newMotDetMal, newDoorsOpen;
Desk
SIGNALLIST DeskOutp = ident, newTaskLight, newEnergy;
SIGNALLIST DeskInp = setEnergyDiffTime, setTaskLight, getTaskLight;
Light
SIGNALLIST LightOutp = ident, newLight, newEnergy;
SIGNALLIST LightInp = setEnergyDiffTime, getLight, setLight, setDimLight, setCIA;


USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators;
Package CP2
3(5)
Office2D
Office1D
SIGNALLIST RoomOutp = ident, getSun, newOccMal, newRoomMalfunction, newRoomEnergyConsumption, newIlluminance, newEntrance;
SIGNALLIST RoomInp = newSun, newSunDetMal, setRoomDelay, setRoomReoccDelay, setDefaultLightScene, setDefaultAmbientLightLevel, setRoomEnergyDiffTime, getIlluminance, setRoomLight;
CompLab2D
CompLab1D

```
USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators;
```

Package CP2

USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators;

Package CP2

PRoom

MRoom

```
USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators;
```

Package CP3

Hallway

```
SIGNAL setHwLight2;

SIGNALLIST HallwayOutp = ident, getIlluminance,
                    newHwEnergyConsumption,
                    newHwMalFunction, getHwPBDelay, setHwLight2;


SIGNALLIST HallwayInp = setHwEnergyDiffTime, setHwLight,
                      setHwPbDelay, newIlluminance,
                      /*newSll,*/ newEntrance, setHWOccDelay;
```

USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators; USE CP1; USE CP2; USE CP3;

Package CP4

Floor

SIGNALLIST FloorOutp = ident;
SIGNALLIST FloorInp = dummy;

USE DataTypes;
USE Utils;
USE Sensors;
USE Actuators;

Package CP5

USE DataTypes; USE Utils; USE Sensors; USE Actuators;
USE CP1; USE CP2; USE CP3; USE CP4; USE CP5;

Package Control