2017-04-03

Delivery 2 of AADL Model of The Smart Home System

Milestone of our project

Delivery Date	Milestones
Mar 13, 2017	Project proposal
Mar 20, 2017	Accomplishing the subcomponents and connections of implementation of the smart home system
Apr 03, 2017	Accomplishing the definition and implementation of the main control router model
Apr 10, 2017	Accomplishing the definition and implementation of the remote server and clients model
Apr 17, 2017	Accomplishing the definition and implementation of the zigbee controller and devices module
Apr 24, 2017	Identifying and Adding related modes and flows, which include nominal and error flows
May 01, 2017	Adding error and nominal behavior for error model

The overview of our project

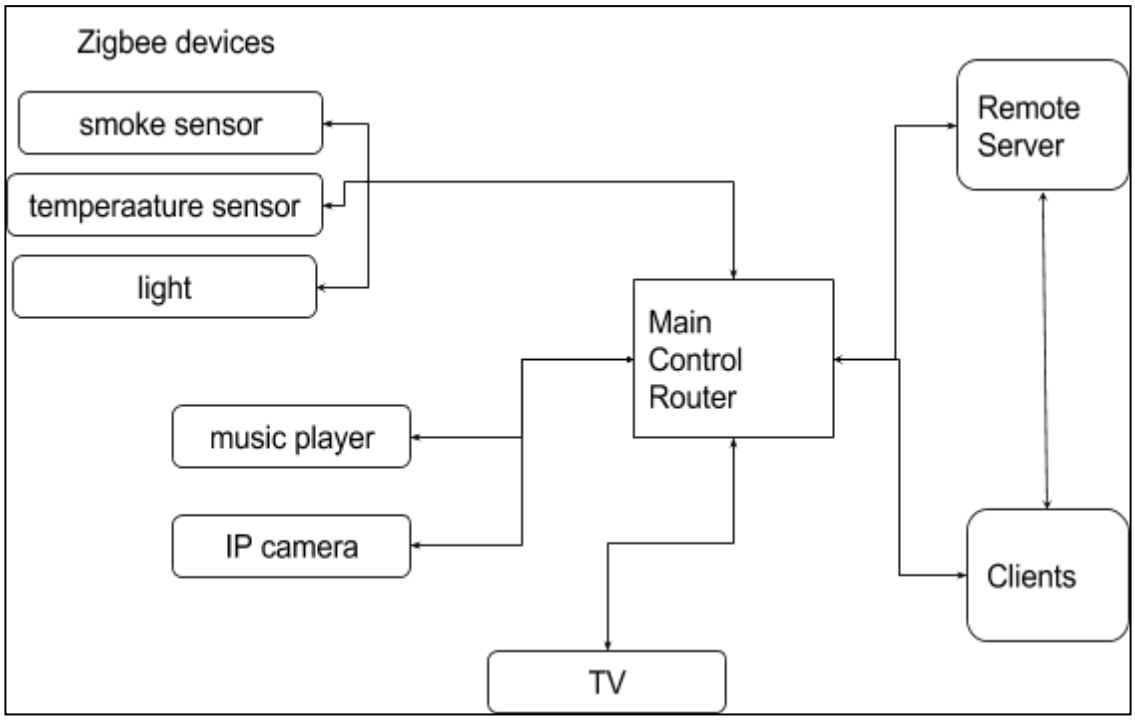

Figure 1. The Overview of Smart Home System

2017-04-03

Overview

In this delivery, according to our milestone, we accomplished the definition and implementation of the model main control router, which includes four processes: zigbee device controller, IP device controller, remote msg control, and main controller. The following gives the implementation and the code of each process:

The Main Control Router Model

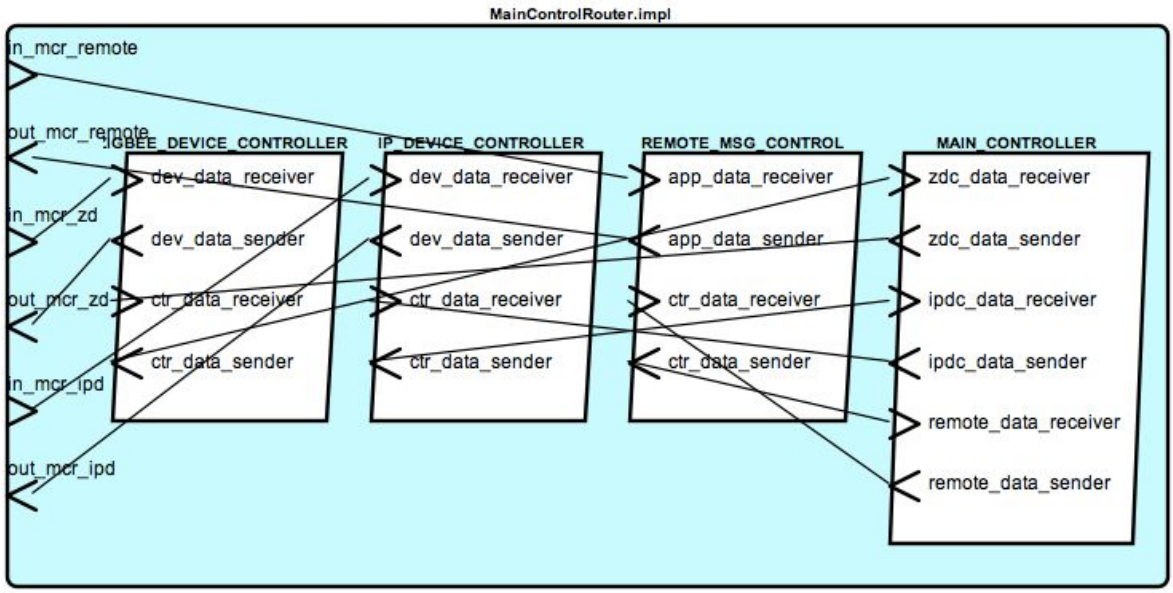

Figure 2. Main control router implementation

```
system MainControlRouter
        features
                in_mcr_remote:   in event port;
                out_mcr_remote:  out event port;
                in_mcr_zd:       in event port;
                out_mcr_zd:      out event port;
                in_mcr_ipd:      in event port;
                out_mcr_ipd:     out event port;
end MainControlRouter;

system implementation MainControlRouter.impl
        subcomponents
                ZIGBEE_DEVICE_CONTROLLER: process Zigbee_Device_Controller;
                IP_DEVICE_CONTROLLER: process IP_Device_Controller;
                REMOTE_MSG_CONTROL: process Remote_Msg_Control;
                MAIN_CONTROLLER: process Main_Controller;
        connections
                c1: port in_mcr_zd -> ZIGBEE_DEVICE_CONTROLLER.dev_data_receiver;
                c2: port ZIGBEE_DEVICE_CONTROLLER.dev_data_sender -> out_mcr_zd;
                c3: port in_mcr_remote -> REMOTE_MSG_CONTROL.app_data_receiver;
                c4: port REMOTE_MSG_CONTROL.app_data_sender -> out_mcr_remote;
                c5: port in_mcr_ipd -> IP_DEVICE_CONTROLLER.dev_data_receiver;
                c6: port IP_DEVICE_CONTROLLER.dev_data_sender -> out_mcr_ipd;
                c7: port Main_Controller.zdc_data_sender ->
Zigbee_Device_Controller.ctr_data_receiver;
                c8: port Main_Controller.ipdc_data_sender ->
IP_Device_Controller.ctr_data_receiver;
                c9: port Main_Controller.remote_data_sender ->
Remote_Msg_Control.ctr_data_receiver;
                c10: port Zigbee_Device_Controller.ctr_data_sender ->
Main_Controller.zdc_data_receiver;
                c11: port IP_Device_Controller.ctr_data_sender ->
Main_Controller.ipdc_data_receiver;
                c12: port Remote_Msg_Control.ctr_data_sender ->
Main_Controller.remote_data_receiver;
```

```
end MainControlRouter.impl;
```

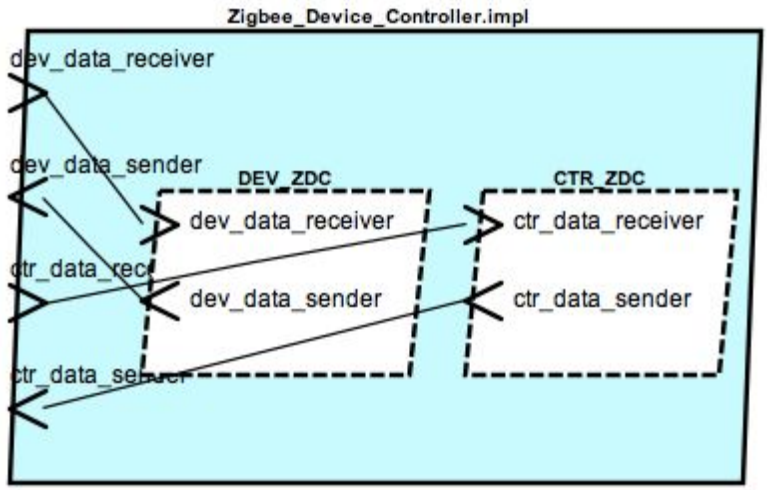

Figure 3. Zigbee Device controller implementation

```
process Zigbee_Device_Controller
	features
		dev_data_receiver:  in event port;
		dev_data_sender:    out event port;
		ctr_data_receiver:  in event port;
		ctr_data_sender:    out event port;
end Zigbee_Device_Controller;

process implementation Zigbee_Device_Controller.impl
	subcomponents
		DEV_ZDC : thread Dev_ZDC;
		CTR_ZDC   : thread Ctr_ZDC;
	connections
		c1: port dev_data_receiver -> DEV_ZDC.dev_data_receiver;
		c2: port ctr_data_receiver -> CTR_ZDC.ctr_data_receiver;
		c3: port DEV_ZDC.dev_data_sender -> dev_data_sender;
		c4: port CTR_ZDC.ctr_data_sender -> ctr_data_sender;
end Zigbee_Device_Controller.impl;

thread Dev_ZDC
	features
		dev_data_receiver:  in event port;
		dev_data_sender:    out event port;

end Dev_ZDC;

thread Ctr_ZDC
	features
		ctr_data_receiver:  in event port;
		ctr_data_sender:    out event port;
end Ctr_ZDC;
```

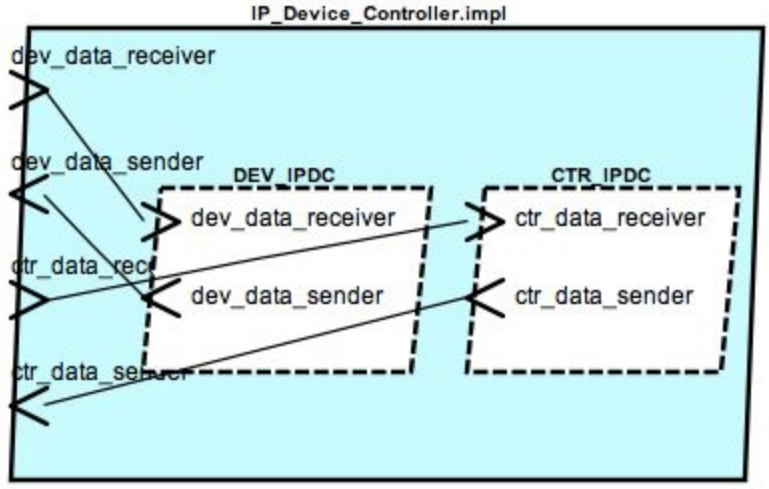

Figure 4. IP device controller implementation

```
process IP_Device_Controller
        features
                dev_data_receiver:  in event port;
                dev_data_sender:    out event port;
                ctr_data_receiver:  in event port;
                ctr_data_sender:    out event port;
end IP_Device_Controller;

process implementation IP_Device_Controller.impl
                subcomponents
                DEV_IPDC : thread Dev_IPDC;
                CTR_IPDC   : thread Ctr_IPDC;
        connections
                c1: port dev_data_receiver -> DEV_IPDC.dev_data_receiver;
                c2: port ctr_data_receiver -> CTR_IPDC.ctr_data_receiver;
                c3: port DEV_IPDC.dev_data_sender -> dev_data_sender;
                c4: port CTR_IPDC.ctr_data_sender -> ctr_data_sender;
end IP_Device_Controller.impl;

thread Dev_IPDC
        features
                dev_data_receiver:  in event port;
                dev_data_sender:    out event port;

end Dev_IPDC;

thread Ctr_IPDC
        features
                ctr_data_receiver:  in event port;
                ctr_data_sender:    out event port;
end Ctr_IPDC;
```

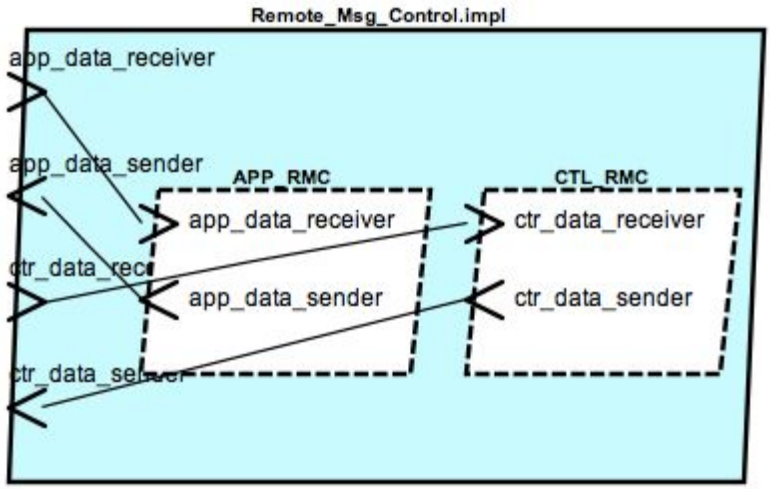

Figure 5. Remote Msg control implementation

```
process Remote_Msg_Control
        features
                app_data_receiver:  in event port;
                app_data_sender:    out event port;
                ctr_data_receiver:  in event port;
                ctr_data_sender:    out event port;
end Remote_Msg_Control;

process implementation Remote_Msg_Control.impl
        subcomponents
                APP_RMC   : thread App_RMC;
                CTL_RMC   : thread Ctr_RMC;
        connections
                c1: port app_data_receiver -> APP_RMC.app_data_receiver;
                c2: port ctr_data_receiver -> CTL_RMC.ctr_data_receiver;
                c3: port APP_RMC.app_data_sender -> app_data_sender;
                c4: port CTL_RMC.ctr_data_sender -> ctr_data_sender;
end Remote_Msg_Control.impl;

thread App_RMC
        features
                app_data_receiver:  in event port;
                app_data_sender:    out event port;

end App_RMC;

thread Ctr_RMC
        features
                ctr_data_receiver:  in event port;
                ctr_data_sender:    out event port;
end Ctr_RMC;
```

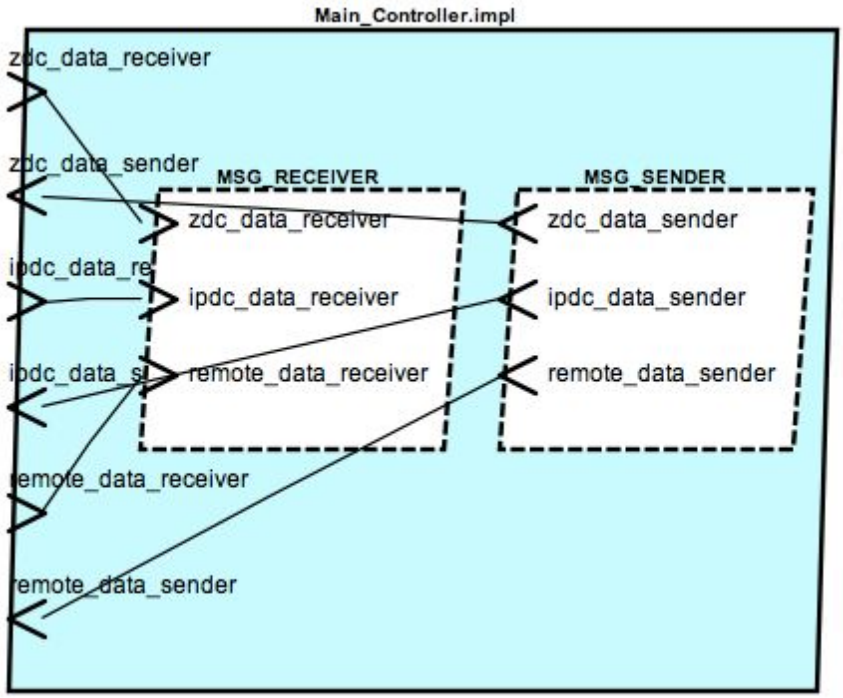

Figure 6. Main controller implementation

```
process Main_Controller
	features
		zdc_data_receiver:  in event port;
		zdc_data_sender:    out event port;
		ipdc_data_receiver:  in event port;
		ipdc_data_sender:    out event port;
		remote_data_receiver:  in event port;
		remote_data_sender:    out event port;
end Main_Controller;

process implementation Main_Controller.impl
	subcomponents
		MSG_RECEIVER :thread Msg_receiver;
		MSG_SENDER :thread Msg_Sender;
	connections
		c1: port zdc_data_receiver -> MSG_RECEIVER.zdc_data_receiver;
		c2: port ipdc_data_receiver -> MSG_RECEIVER.ipdc_data_receiver;
		c3: port remote_data_receiver -> MSG_RECEIVER.remote_data_receiver;
		c4: port Msg_Sender.zdc_data_sender -> zdc_data_sender;
		c5: port Msg_Sender.ipdc_data_sender -> ipdc_data_sender;
		c6: port Msg_Sender.remote_data_sender -> remote_data_sender;
end Main_Controller.impl;

Thread Msg_Receiver
	features
		zdc_data_receiver:  in event port;
		ipdc_data_receiver:  in event port;
		remote_data_receiver:  in event port;
end Msg_Receiver;

Thread Msg_Sender
	features
		zdc_data_sender:    out event port;
		ipdc_data_sender:    out event port;
		remote_data_sender:    out event port;
end Msg_Sender;
```

2017-04-03