Virtual Infrastructure Operating System (VIOS)

EF-PI Indesit Washing Machine driver and manager

Status: Concept
Version: 1.1
Date: 10-6-2015

Contents

1 Indesit Washing Machine 3
1.1 Zigbee protocol driver 3
1.2 Indesit resource driver 4
1.2.1 Configuration 4
1.2.2 WashingMachineState 4
1.2.3 Indesit Resource Control Parameters 6
1.3 Indesit resource manager 7
1.3.1 Configuration 7
1.3.2 Registration 7
1.3.3 Control Space updates 8
1.3.4 Allocations 9
1.3.5 Widget 10

1 Indesit Washing Machine

This document describes the integration of the Indesit washing machine within EF-pi.

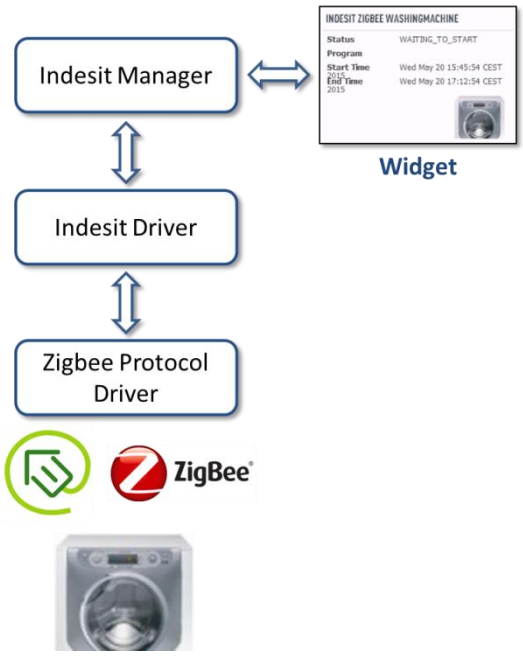

Figure 1: Overview of the components for the Indesit washing machine

Figure 1 provides an overview of all components required for the Indesit washing machine. The washing machine adheres to the Energy@home specification (http://www.energy-home.it). This specification is based upon ZigBee wireless communication.

The Zigbee Protocol Driver manages the Zigbee communication between the SmartBoxx and the washing machine. The source code of this component has not been made available by Technolution, but the binary version could be used within the context of this project.

On top of the Zigbee communication there is the functionality of the washing machine that is exposed through Energy@home. The Indesit driver makes the relevant functions available in the form of a Java interface.

The Indesit resource manager performs the final translation to the Energy Flexibility Interface (EFI). It also features a widget that provides feedback to the end user about the status of the washing machine.

1.1 Zigbee protocol driver

This protocol driver has been developed by Technolution and is proprietary. It takes care of the transport part of the communication with the washing machine.

Figure 2: Protocol driver for the Indesit washing machine

Figure 2 shows the configuration options for the zigbee protocol driver. The port and the baudrate can be specified. The baudrate needs to be 115200 in the case of the Indesit washing machine.

Of course the hardware that FPAI is running on needs to support Zigbee communication.

1.2 Indesit resource driver

1.2.1 Configuration

```
  /**
   * Configuration for Indesit Resource Driver
   */
  @OCD(description = "Indesit Resource Driver")
  public interface Config {

    // Note: applianceId needed by flexible power framework to bound
    // resource manager to resource driver
    @AD(description = "Appliance identification", deflt = "Indesit washing machine")
    String resourceId();

    @AD(description = "Address for Zigbee communication", deflt =
"30:DE:86:00:00:50:00:DE")
    String addrStr();

    @AD(description = "Stub mode Indesit Resource Driver", deflt = "false")
    boolean stubMode();
  }
```

The code snippet shows the configuration options. The most important parameter is the MAC address which is used for the Zigbee communication with the washing machine.

1.2.2 WashingMachineState

The driver communicates the state of the washing machine to the resource manager using the WashingMachineState interface, which is listed below.

```
public interface WashingMachineState extends ResourceState {

    /** Get mode of the washing machine */
```

```
    WashingMachineMode getCurrentMode();

    /** Get program name */
    String getProgram();

    /** Get earliest time to start the program */
    Date getStartTime();

    /** Get latest time to start the program */
    Date getEndTime();

    /** Get energy profile of the program */
    EnergyProfile getEnergyProfile();
}
```

This interface provides the following functionality:

- The current mode of the washing machine. These modes will be further explained in section 1.2.2.1.
- The washing program that was selected by the end user.
- The earliest start time of the washing machine.
- The latest possible start time of the washing machine.
- The energy profile that shows when and how much energy the washing machine will consume when it runs the selected program.

1.2.2.1 Washing machine modes

```
public enum WashingMachineMode {
    /** The machine is not switched on or unavailable for the Resource Manager */
    IDLE_OFF,

    /** A program is selected, but not started */
    PROGRAM_SELECTED,

    /** A complete schedule request is received */
    SCHEDULE_REQUEST,

    /** A program is started, but not running yet */
    WAITING_TO_START,

    /** The waiting program is paused */
    WAITING_TO_START_PAUSED,

    /** An program is running */
    RUNNING,

    /** The running program is paused */
    RUNNING_PAUSED
}
```

The code snippet above shows the different modes the washing machine can be in. The "WAITING_TO_START" mode is most relevant, because that will be the trigger for the resource manager to send out a control space.

1.2.3 Indesit Resource Control Parameters

```
public interface IndesitResourceControlParameters extends ResourceControlParameters
{

  /** Get start time of program */
  Date getStartTime();

  /** Get start directly */
  Boolean getStartNow();

  /** Get start directly */
  void setStartNow(Boolean startNow);
}
```

This interface provides methods to set and get the desired start time of the washing machine. The start time can be retrieved via the getStartTime Method. As an alternative there is also a method getStartNow wich returns a boolean that returns true when the washing machine should start straight away. As a counterpart there is the setStartNow method that can be set by the resource manager.

1.2.3.1 Handle Control Parameters

Within the indesit resource driver the following method is called whenever control parameters are received.

```
  protected void handleControlParameters(
      IndesitResourceControlParameters resourceControlParameters) {

    String startDelayStr = String.format("startTime=%tF %<tT",
        resourceControlParameters.getStartTime());
    log.debug(String.format("setControlParameters: %s", startDelayStr));

    if (config.stubMode()) {
      // TODO VIOS: FOR TESTING: Create timer which submits resource state
      executorService.schedule(this, STUB_MODE_SCHEDULE_TIME,
          java.util.concurrent.TimeUnit.SECONDS);
    } else {

      if (resourceControlParameters.getStartNow()) {
        deviceController.sendSchedule(0);
      } else {
        // Determinate start delay
        long startDelayMs = resourceControlParameters.getStartTime()
            .getTime() - resourceState.getStartTime().getTime();
        long startDelayMin = startDelayMs / 1000 / 60;

        deviceController.sendSchedule((int) startDelayMin);
      }
    }
  }
```

The handleControlParameters method takes an object that implements the IndesitResourceControlParameters as input.

It first checks whether the driver is in stub mode. This mode was only used for testing purpose and is not relevant here. The next step is to check whether the washing machine should start now. If so a signal to start the washing machine is sent directly. If the starting time lies in the future, the driver determines the delay until that time and sends a message to the washing machine to start after the delay period has finished.

1.3 Indesit resource manager

The Indesit resource manager translates the washing machine state information from the driver into a Timeshifter control space that is being sent to the energy app. When it receives an allocation in return it will construct the Indesit resource control parameters accordingly and sends those to the driver.

1.3.1 Configuration

The configuration for the resource manager of the washing machine is shown in the code snippet below.

```
    @OCD(description="Indesit Resource Manager")
    public interface Config {

    @AD(description = "Unique resourceID", deflt = "WashingmachineManager")
        String resourceId();

        // Note: applianceId needed by flexible power framework to bound resource
manager to resource driver
        @AD(description = "Appliance identification", deflt = "Indesit washing
machine")
        String applianceId();

        @AD(description = "Enable gui", deflt = "false") // Note: Always "false" on
SmartBoxx (Linux)
        boolean enableGui();
    }
```

The resource ID uniquely identifies this resource manager and has a default value of “Indesit Resource Manager”.

Appliance ID was a required parameter for previous versions of FPAI but has become obsolete since the 14.10 release of FPAI.

The boolean to enable the GUI was used during the development phase of this resource manager and should be set to false in the operational environment.

As becomes apparent from the explanation, all parameters can be left to their default value when instantiating the indesit resource manager.

1.3.2 Registration

```
  protected List<? extends ResourceMessage> startRegistration(
      WashingMachineState state) {
    latestState = state;
```

```
    TimeShifterRegistration registration = new
TimeShifterRegistration(config.resourceId(), timeService.getTime(),
Measure.valueOf(60, SI.SECOND), CommoditySet.onlyElectricity);

    return Arrays.asList(registration);
  }
```

The registration method is very simple; the resource manager indicates that it will only send electricity profiles to the energy app. One thing to note is the allocation delay value, which is set to 60 seconds. The washing machine will only switch on at the start of a minute, so if one wants to make sure that the washing starts at the expected time, the allocation should be sent at least 60 seconds in advance.

1.3.3 Control Space updates

The code snippet below shows the logic behind the control space updates for the Indesit resource manager.

```
  protected List<? extends ResourceMessage> updatedState(WashingMachineState state)
{
    ResourceMessage update = null;
    latestState = state;

    WashingMachineMode mode = state.getCurrentMode();

    switch (mode) {
    case IDLE_OFF:
      //sendZeroFlexibilityControlspace();
      update = new ControlSpaceRevoke(config.resourceId(), timeService.getTime());
      break;

    case SCHEDULE_REQUEST:
      sendControlParameters(new
IndesitResourceControlParametersImpl(state.getEndTime()));
      latestScheduleState = state;
      // update = getZeroFlexibilityControlSpace();
      update = new ControlSpaceRevoke(config.resourceId(), timeService.getTime());
      break;

    case WAITING_TO_START:

      if (latestScheduleState != null) {
        // Create a control space
        update = createControlSpaceFromWashingMachineState(latestScheduleState);
      } else {
        log.error("Cant build controle space, there was already a program in the
washingmachine! fixme");
      }
      break;

    default:
      // Nothing to do
      log.info("No action for: " + mode);
      break;
    }
```

```
    if (update != null && update instanceof TimeShifterUpdate) {
      log.debug(printTimeShifterUpdate((TimeShifterUpdate)update));
    }

    return Arrays.asList(update);
  }
```

The logic is highly dependent on the washing machine modes (see section 1.2.2.1). Whenever the mode changes to "WAITING_TO_START" a control space will be sent to the energy app. Some modes (like "IDLE_OFF") lead to revoking all control spaces by this resource manager. This is useful when a program is interrupted and the current control space will no longer be valid.

1.3.4 Allocations

```
  protected IndesitResourceControlParameters receivedAllocation(ResourceMessage
message) {
    IndesitResourceControlParameters controlParameters = null;

    // Is this a Timeshifter Allocation?
    if (message instanceof TimeShifterAllocation) {
      log.debug(printAllocation((TimeShifterAllocation) message));
      TimeShifterAllocation allocation = (TimeShifterAllocation) message;

      allocationStatusUpdate(new AllocationStatusUpdate(timeService.getTime(),
allocation, AllocationStatus.STARTED, "n.a."));

      List<SequentialProfileAllocation> profileAllocationList =
allocation.getSequentialProfileAllocation();
      for (int i = 0; i < profileAllocationList.size(); i++) {
        SequentialProfileAllocation profileAllocation =
profileAllocationList.get(i);
        controlParameters = new
IndesitResourceControlParametersImpl(profileAllocation.getStartTime());
      }
    }

    return controlParameters;
  }
```

After a valid allocation has been received, messages will be send to the energy app and the resource driver of the washing machine. The Energy app will receive an Allocation Status Update message to inform it that the washing machine has started (or will start very shortly…). The other message is an Indesit Resource Control Parameters message (see 1.2.3). The start time in that message is copied from the start time of the profile allocation in the TimeShifter Allocation.

1.3.5 Widget

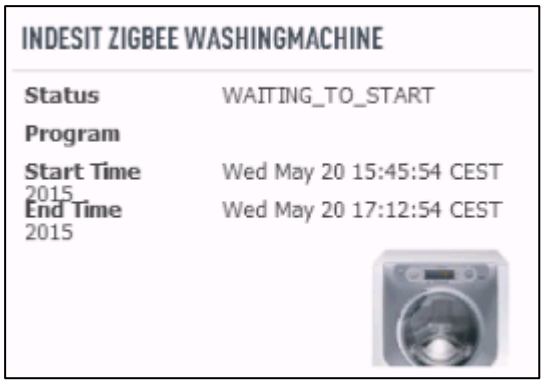

Figure 3: The widget of the Indesit resource manager

This widget shows the following information:

- The current status which corresponds directly to the washing machine mode.
- The name of the program that has been selected (if available).
- The (earliest) start time of the selected program.
- The end time, which is the latest possible start time of the selected program.