



75 Fifth Street NW, Suite 312  
Atlanta, GA 30308, USA  
voice: +1-404-592-6897  
web: [www.Intercax.com](http://www.Intercax.com)  
email: [info@Intercax.com](mailto:info@Intercax.com)

Dr. Dirk Zwemer, InterCax LLC

# Technote: Smart Home – Modeling the Internet of Things with SysML

## Part 1 Requirements to Functional Design

Abstract.....	1
Introduction .....	2
The Challenge for the Systems Engineer .....	2
The SysML Requirements Model .....	4
SysML Functional Modeling of the Smart Home .....	5
The Security Functional Model .....	5
The HVAC Functional Model .....	7
The Audio Functional Model.....	8
Activity Decomposition Map.....	8
Tracking Requirement Satisfaction.....	9
Summary .....	11
About the Authors .....	11

### Abstract

This is the first in a series of technical notes describing the application of Model-Based Systems Engineering (MBSE) to the specification, design, procurement and evaluation of an Internet of Things (IoT) system. This section uses SysML to capture customer requirements, create a detailed functional design based on those requirements, and manage the relationship between them as the design evolves. The ability of IoT components to communicate across subsystem lines is utilized to realize system functions beyond commonly available technologies.

## Introduction

The Internet of Things (IoT) holds enormous promise of new capabilities for users and new opportunities for businesses. It also presents enormous challenges to systems engineers. Model-Based Systems Engineering (MBSE) potentially provides an efficient way to address those challenges, being holistic, integrated, flexible and object-oriented.

### Holistic

The OMG Systems Modeling Language (SysML) has become the consensus mechanism for implementing MBSE because it provides a common ground for describing requirements, behaviors, structure and quantitative constraints (parametrics). Both intuitive and precise in its semantics, it provides the working space shared by human and computer necessary to attack these complex designs.

### Integrated

The SysML standard is supported by a wide variety of software, both directly by the SysML modeling tools and indirectly by the integrations of those tools with PLM, CAD, simulation and requirements management software. In this technical note, we will use specific tools to illustrate our approach, including Syndeia and ParaMagic from Interccax, but there are many alternative options for all the capabilities described from a wide variety of vendors.

### Flexible

One virtue of SysML is the customization capability built into the language through its mechanism of stereotypes and profiles. We fully expect the language to be extended to address the special issues of the IoT, even to the development of a domain-specific language, though such is beyond the scope of this series of notes. Equally important is the flexibility in presentation, using the ability to present different views of the same model to manage complexity. New visualization capabilities are being introduced constantly as the federation of engineering models advances.

### Object-Oriented

As an object-oriented language, SysML relies on the building-block concept, where simple systems are used to build more complex ones. This mirrors the pattern of IoT, where smart devices with connectivity, systems in their own right, are joined in local networks. These local networks become parts of systems-of-systems, with control and data analytics in the Cloud. The object-oriented structure of these models makes it easy to build progressively larger and more powerful systems, using modules from a variety of domains and vendors.

## The Challenge for the Systems Engineer

Systems engineering a particular project might be thought of as four stages:

- Specify
- Design
- Procure
- Evaluate

The first note in this series will start with the specification of system requirements and use them to create a functional design as a reference architecture. The second will use this functional design to generate a physical architecture and build a parametric model for issues of cost and performance that must be evaluated for the final product. This physical structure also becomes the starting point for the procurement phase, which brings real product data into the model from a variety of external repositories and databases.

In the third note in this series, we will take the product data and compare it against functional requirements, create specific instantiations and use parametric models to evaluate cost and performance. The final note will consider how this realized system both fits into a larger system-of-systems and can access more detailed product information, e.g. CAD files or software code.

## The Problem

We begin with a familiar system, the “Smart Home” or “Connected Home” concept. Specifically, we consider three major subsystems: Home security, HVAC (heating, ventilation and air conditioning), and audio entertainment, as well as some individual appliances that are both significant power consumers and internal heat generators. The potential for interoperability among these systems is wide-ranging and the home owner starts with a wish list of features:

- Can control temperature on a room-by-room basis
- Can control temperature adjusting for in-room IR load
- Can control temperature adjusting for local appliance power dissipation
- Can control temperature adjusting for in-room occupancy
- Can detect motion to trigger alarm for break-ins through motion sensors
- Can detect fire through IR sensors
- Can send audio to speakers on a room-by-room basis
- Can control local audio volume on a room-by-room basis
- Can control local audio volume adjusting for in-room occupancy
- Can sound fire/burglar alarm warnings through motion sensors and speakers

## The Software Tools

In this series of notes, we will illustrate the concepts using a specific set of tools: MagicDraw (No Magic Inc.), MySQL (Oracle), Player Pro (Wolfram Research) and Syndeia and ParaMagic (Intercax). However, this approach is not restricted to this toolset. For example, this note could have used Rational Rhapsody (IBM) as the SysML modeling tool with Syndeia and Melody plug-ins from Intercax. Other repositories, ranging from Microsoft Excel to powerful PLM systems such as PTC Windchill or Siemens Teamcenter could be used as the source of product data or requirements.

We assume that the reader is familiar with the general concepts of MBSE and SysML. For those in need of more background, there are several valuable books available, as well as the user guides and tutorials with the tools used.

## The SysML Requirements Model

The customer wishlist in the introduction translates into the set of functional requirements shown in Figure 1. We will use these requirements to track their satisfaction status in the SysML model as the design progresses. If the SysML tool is linked to a requirements repository, these requirements could be imported (and later synchronized) using connection software like Syndeia or Cameo Data Hub.

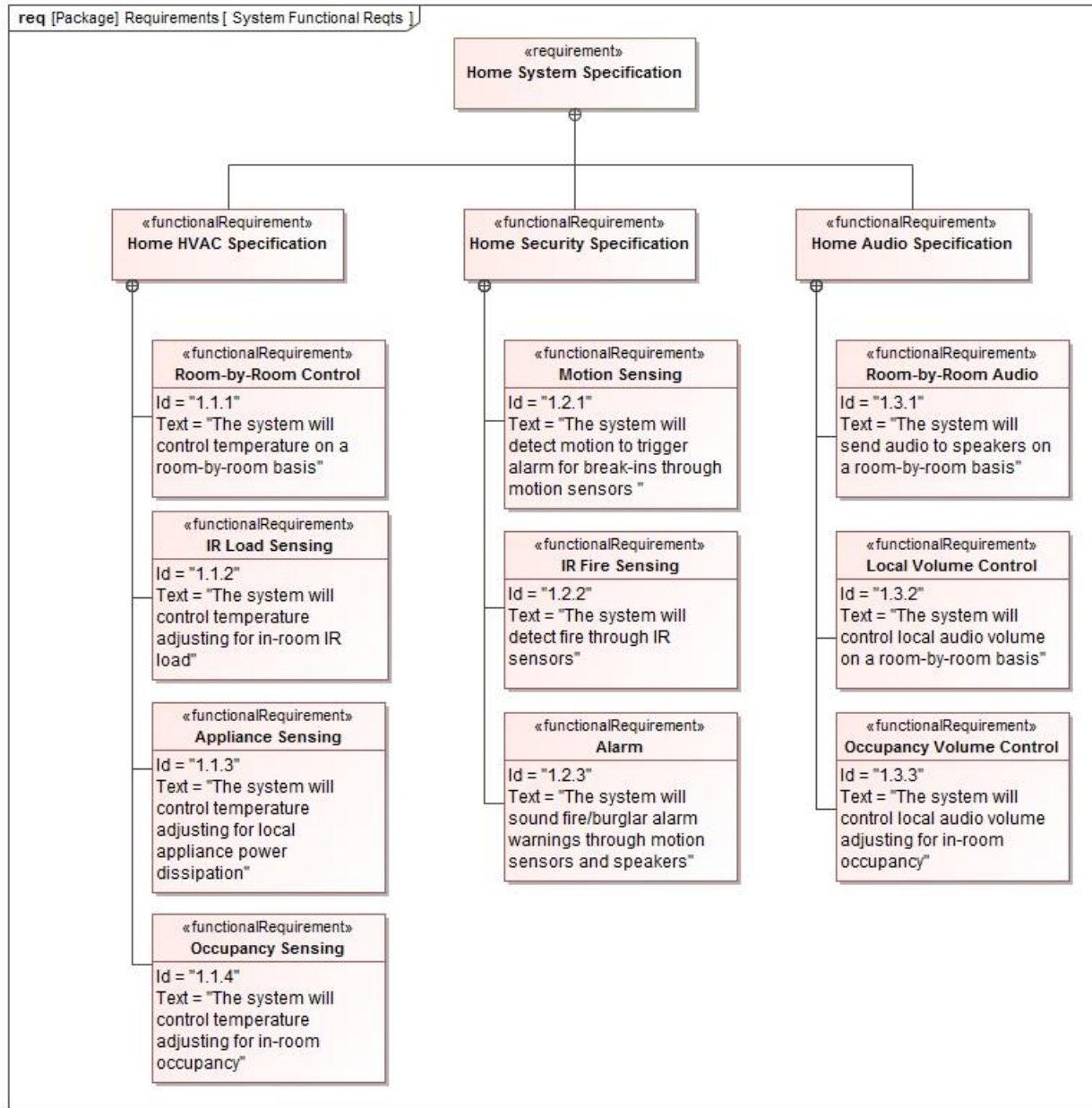


Figure 1 Functional requirements model

## SysML Functional Modeling of the Smart Home

A functional model of the system is created using SysML activity diagrams, starting with the top-level diagram shown in Figure 2. A main behavior is created for each of the three main subsystems, plus a behavior for the appliances that enables them to transmit their power dissipation values to the other components. Color coding has been added for easier interpretation: red for Security, yellow for Audio, blue for HVAC. Note that the finished functional model is shown in this section, but that the modeling process is iterative, with the information flows between functions being more fully specified as model is detailed.

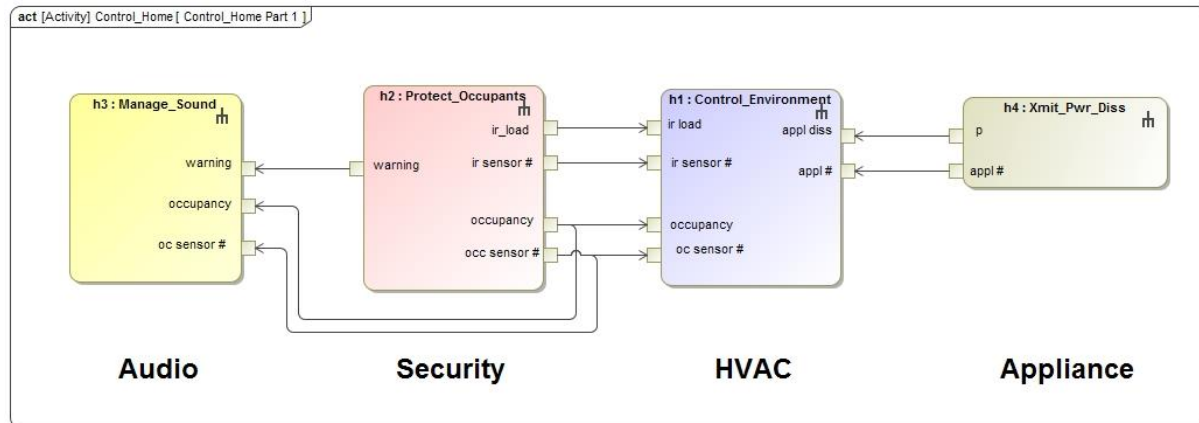


Figure 2 Top-Level activity diagram

## The Security Functional Model

The functional requirements drive the design of the Security functional model. The system must

- Detect occupancy locally
- Measure infra-red (IR) levels locally to detect heat sources, like people, as well as fires
- Deliver an audible alarm locally
- Provide central monitoring and control of these functions

The overall Protect\_Occupants function in Figure 2 can be decomposed into these four second level functions, as shown in Figure 3. This helps to identify the information that must be shared between functions. Local detection of occupancy and IR implies multiple sensors in most systems, such as a house with many rooms, so these sensors must not only supply sensor information, they must identify themselves.

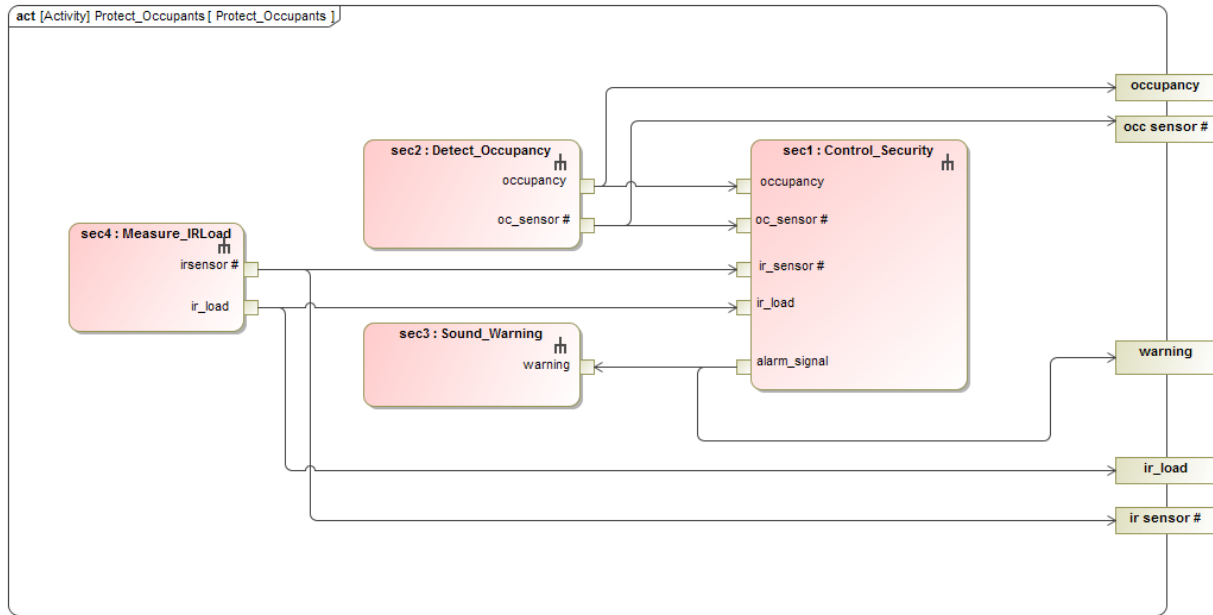


Figure 3 Second-level activity diagram for Security function

The functional model can be further decomposed to deeper levels, as illustrated in Figure 4 for the Control\_Security action in Figure 3. In this diagram, we are reading both occupancy and IR detection inputs. The IR value can identify fires and trigger a warning sequence. The occupancy value, compared against an Alarm Status value (e.g. Armed/Disarmed) stored by the control function, can trigger a break-in warning. Similar decomposition diagrams are created for the other functions in Figure 3.

We are also beginning to make design choices. Note that in Figure 3, the occupancy and IR values are made available directly to systems outside Security, for example, to local HVAC or Audio components to enhance local control of these functions, rather than passing only through a central controller. This can potentially provide faster local response and better system redundancy by increasing the number of available communication links.

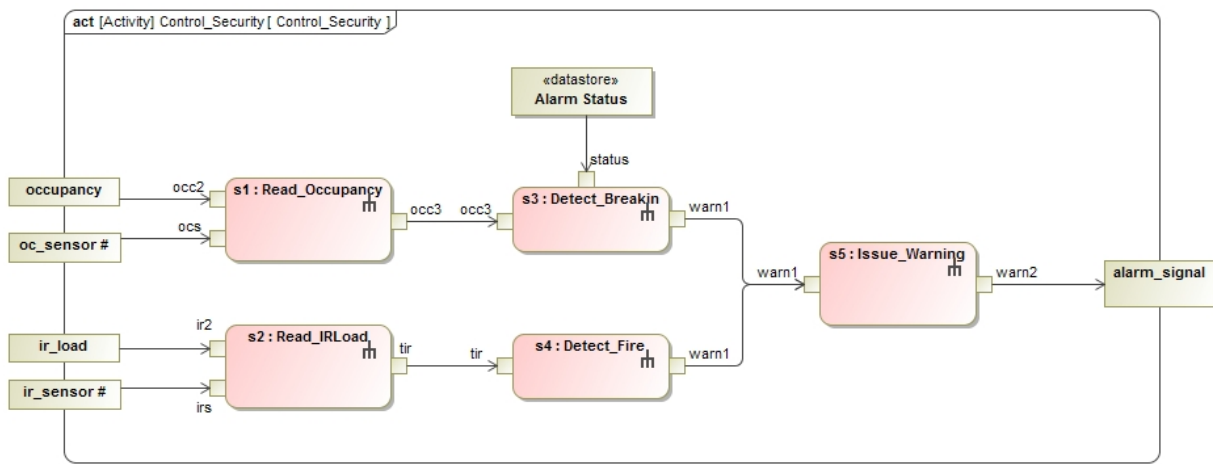


Figure 4 Third-level activity diagram for Control\_Security function

## The HVAC Functional Model

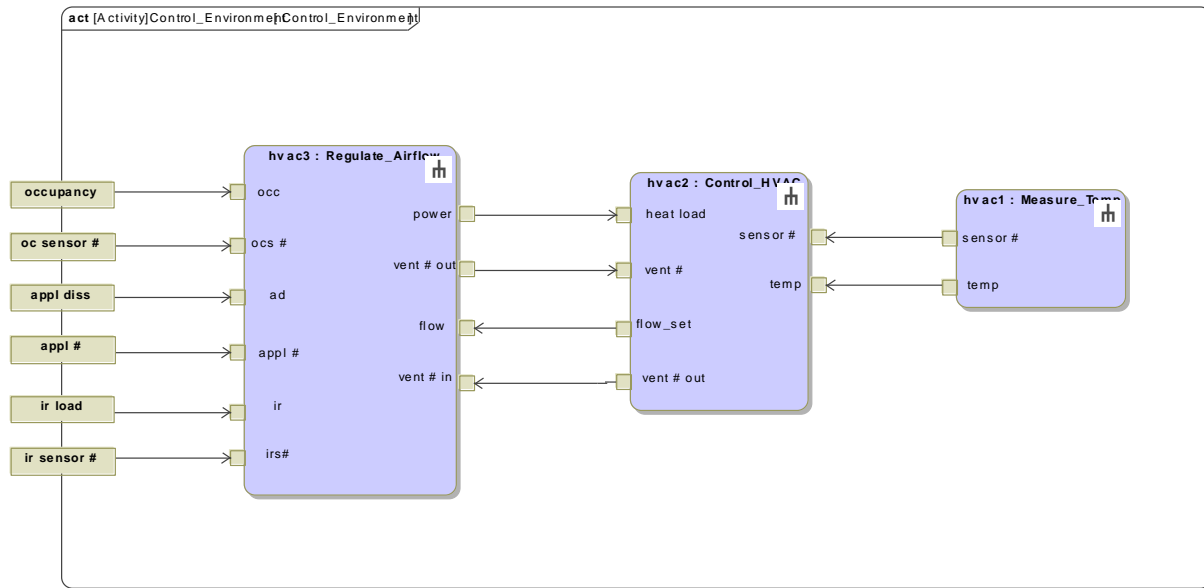


Figure 5 Second-level activity diagram for Control\_Environment (HVAC) function

The decomposition of the Control\_Environment activity is shown in Figure 5, mirroring the intent to measure temperature and control airflow locally, as well as the supporting control function. An example of the third-level decomposition is shown in Figure 6 for Regulate\_Airflow. We anticipate an airflow component with the ability to read local occupancy, IR and appliance power dissipation values, and to store programmed IDs for those local components.

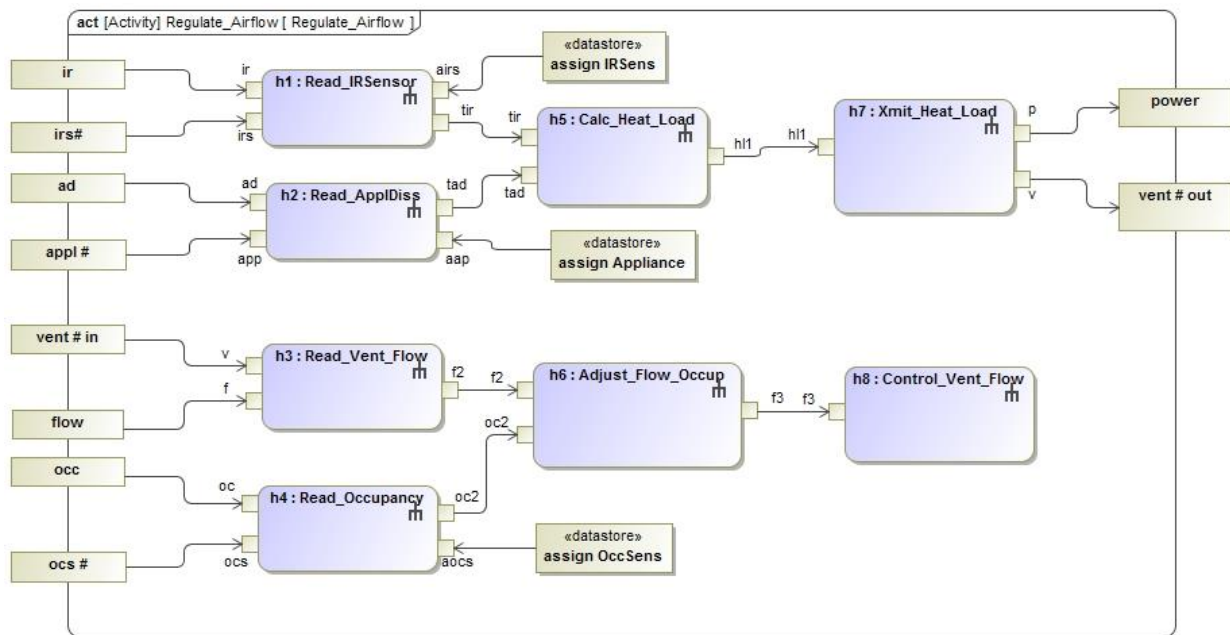


Figure 6 Third-level activity diagram for Regulate\_Airflow function

## The Audio Functional Model

Our examples from the third subsystem for Audio follow a similar pattern to HVAC, where the local actuator functionality (i.e. Control\_Speaker) uses information from external sources (occupancy) as well as audio program and local volume control from within the subsystem.

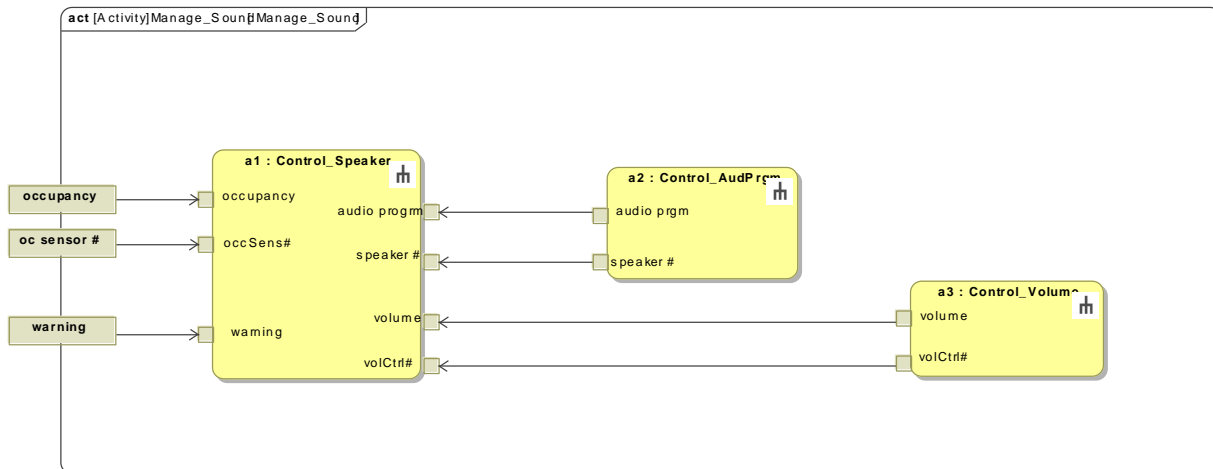


Figure 7 Second-level activity diagram for Audio function

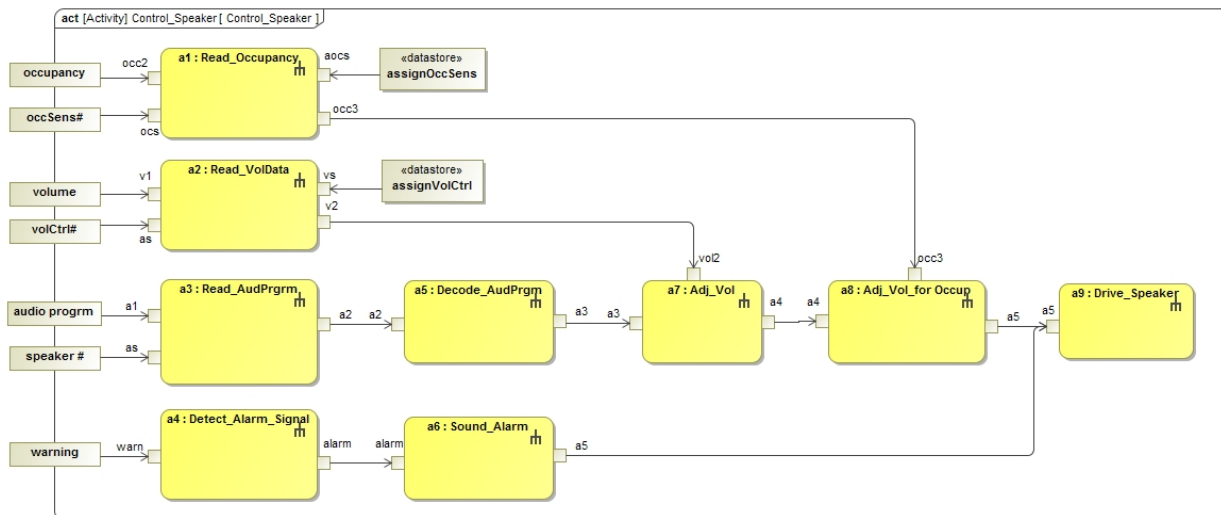


Figure 8 Third-level activity diagram for Control\_Speaker function

## Activity Decomposition Map

Only a few of the individual activity diagrams at the third and fourth levels have been shown in this note. For those interested, the complete model can be downloaded from this website (link). The full activity decomposition map is shown in Figure 9. In the next note in the series, the list of fourth-level activities in the reference model will be compared against the operations available for the physical components that will implement these functions.



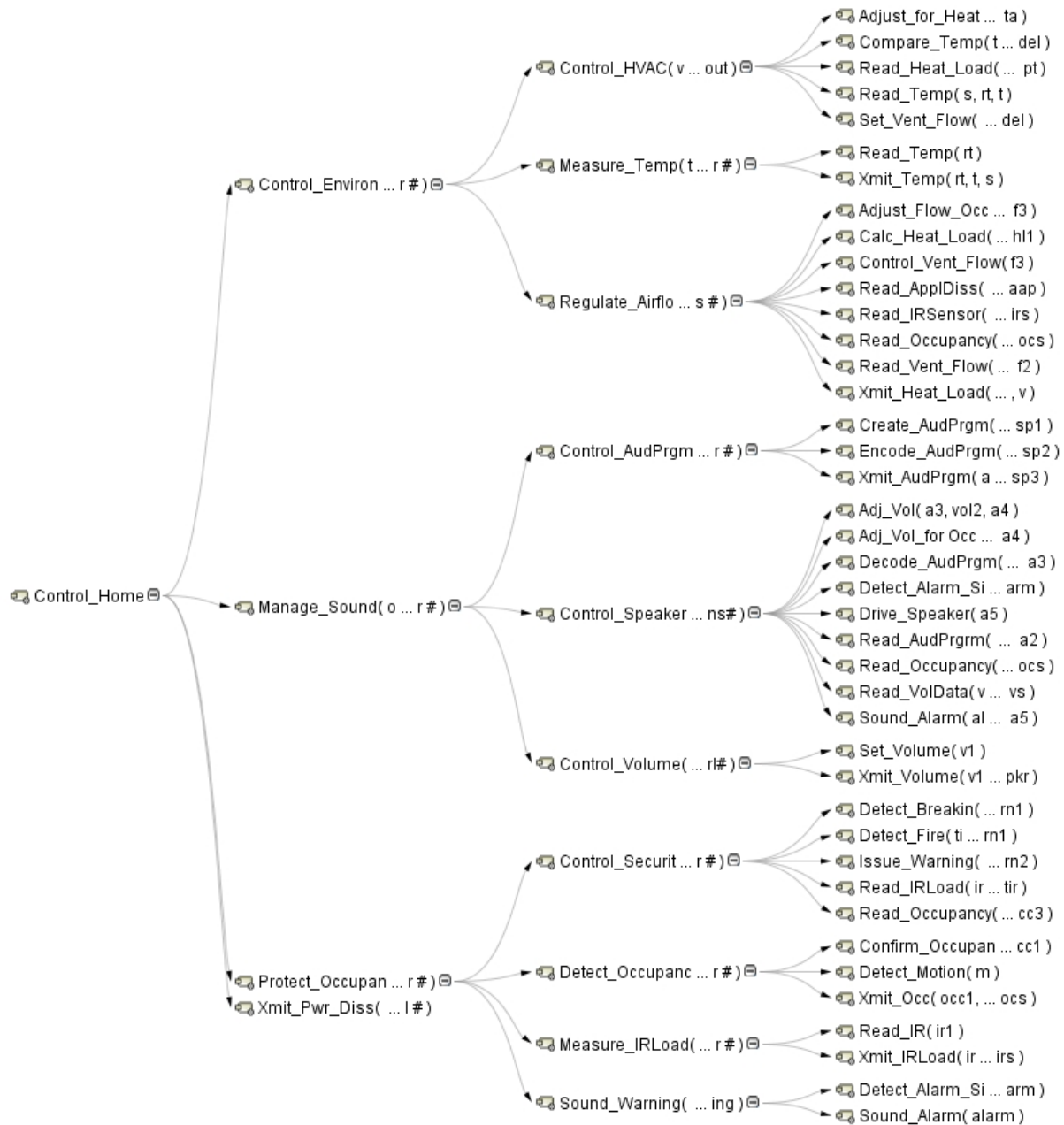


Figure 9 Activity decomposition map

## Tracking Requirement Satisfaction

SysML modeling tools make it relatively easy to track the relationship between functions and the functional requirements that they satisfy. Figure 10 shows a requirements diagram where <<satisfy>> relationships are created between the HVAC-based requirements and the fourth-level activities of the HVAC subsystem. For all requirements, a matrix representation, as in Figure 11, is more compact and makes the identification of unsatisfied requirements more obvious.

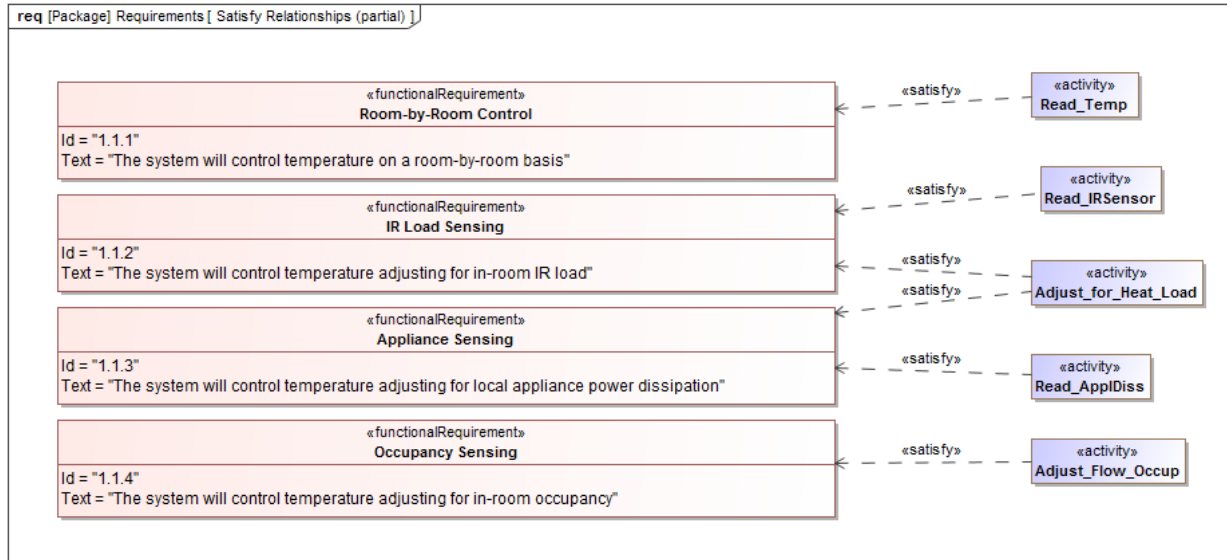


Figure 10 Requirements diagram showing <<satisfy>> relationships for fourth-level activities

	1 Home System Specification [Requirements]												
	1.1 Home				1.2 Home				1.3 Home				
	1.1.1 Room-by-Room Control	1.1.2 IR Load Sensing	1.1.3 Appliance Sensing	1.1.4 Occupancy Sensing	1.2.1 Motion Sensing	1.2.2 IR Fire Sensing	1.2.3 Alarm	1.3.1 Room-by-Room Audio	1.3.2 Local Volume Control	1.3.3 Occupancy Volume Control			
<b>Functions</b>													
<b>Audio Functions</b>													
Adj_Vol_for Occup( occ3, a5, a4 )	1							1					
Adj_Vol( a3, vol2, a4 )	1							1					
Encode_AudPrgm( ap1, sp1, ap2, sp2 )	1							1					
Read_AudPrgm( a1, as, a2 )	1							1					
Read_Occupancy( occ2, ocs, occ3, aocs )	1							1					
Xmit_Volume( v1, v2, spkr )	1							1					
<b>HVAC Functions</b>		1	2	2	1								
Adjust_Flow_Occup( f2, oc2, f3 )	1	1											
Adjust_for_Heat_Load( t, pt, ta )	2	2											
Read_ApplDiss( ad, app, tad, aap )	1	1											
Read_IRSensor( ir, irs, tir, airs )	1	1											
Read_Temp( s, rt, t )	1	1											
<b>Security Functions</b>						2	2	2					
Detect_Breakin( occ3, status, warn1 )	1				1								
Detect_Fire( tir, warn1 )	1				1								
Detect_Motion( m )	1				1								
Read_IRLoad( ir2, irs, tir )	1				1								
Sound_Alarm( alarm )	1				1								
Sound_Warning( warning )	1				1								

Figure 11 <<Satisfy>> relationship matrix

## Summary

The intent of this first note on the Internet of Things (IoT) Smart Home model has been to illustrate the first part of the conceptual design, from requirements to functional design. Two key features of this process are the straight-forward decomposition (and navigation) of the functional design from high-level concepts to low-level operations, and the ability to link (and visualize) requirements and activities to assure that all requirements are being addressed.

In the following note, we will continue the process by

- Creating a reference physical architecture to carry out the desired functionality
- Adding parametric constraints to the reference physical architecture to evaluate performance and cost
- Drawing product information from external databases for a specific home and IoT product configuration

Future notes will describe the instantiation and evaluation of the specialized configuration, the incorporation of these models into larger models, e.g. the “Smart Grid” for electrical power or the Cloud for external processing power and analytics, and connection to smaller models, e.g. the CAD files and code of the individual smart components. The ability of MBSE to connect systems of different scales potentially enables to full power of the Internet of Things to be realized.

## About the Author

Dr. Dirk Zwemer ([dirk.zwemer@intercax.com](mailto:dirk.zwemer@intercax.com)) is President of Intercax LLC, Atlanta, GA and holds OCSMP certification as Model Builder - Advanced.

For further information, visit us at [www.intercax.com](http://www.intercax.com) or contact us at [info@intercax.com](mailto:info@intercax.com).

Copyright 2015 Intercax LLC