软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(8):1696−1712 [doi: 10.13328/j.cnki.jos.004665]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于运行时模型的无线传感网管理方法*

陈 星[1,2], 张 伟[3,4], 黄 罡[3,4], 李隘鹏[1,2], 郭文忠[1,2], 陈国龙[1,2]

[1](福州大学 数学与计算机科学学院,福建 福州 350108)

[2](福建省网络计算与智能信息处理重点实验室,福建 福州 350108)

[3](北京大学 信息科学技术学院 软件研究所,北京 100871)

[4](高可信软件技术教育部重点实验室(北京大学),北京 100871)

通信作者: 黄罡, E-mail: huanggang@sei.pku.edu.cn

摘 要: 无线传感网是物联网的核心,主要解决物联网中的信息感知问题,通过散布在特定区域的成千上万的传感器节点,采集环境中各种信息并连接到互联网上.然而,传感设备所采集到的数据是实时的、数量庞大且无良好结构的,要将采集到的数据映射到应用系统的问题域空间,就不得不编写大量的映射代码.为了快速定制和开发物联网系统,提出一种基于运行时模型的无线传感网管理方法:首先,在传感设备管理接口基础上构造运行时模型,并维护运行时模型与采集到信息的数据同步;其次,基于运行时模型,对不同传感设备采集到的数据进行定制、抽取和合并,实现通过组合模型对场景中不同的传感设备进行统一管理;最后,通过模型转换,实现组合模型到应用场景模型的映射,从而能够面向应用场景进行物联网系统的开发.还实现了基于运行时模型的智慧社区原型系统,并对方法的可行性和有效性进行了验证.

关键词: 无线传感网;软件体系结构;运行时模型

中图法分类号: TP393

中文引用格式: 陈星,张伟,黄罡,李隘鹏,郭文忠,陈国龙.基于运行时模型的无线传感网管理方法.软件学报,2014,25(8):1696−1712. http://www.jos.org.cn/1000-9825/4665.htm

英文引用格式: Chen X, Zhang W, Huang G, Li AP, Guo WZ, Chen GL. Management approach of wireless sensor networks based on runtime model. Ruan Jian Xue Bao/Journal of Software, 2014,25(8):1696−1712 (in Chinese). http://www.jos.org.cn/1000-9825/4665.htm

Management Approach of Wireless Sensor Networks Based on Runtime Model

CHEN Xing[1,2], ZHANG Wei[3,4], HUANG Gang[3,4], LI Ai-Peng[1,2], GUO Wen-Zhong[1,2], CHEN Guo-Long[1,2]

[1](College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

[2](Fujian Provincial Key Laboratory of Networking Computing and Intelligent Information Processing, Fuzhou 350108, China)

[3](Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

[4](Key Laboratory of High Confidence Software Technologies of Ministry of Education (Peking University), Beijing 100871, China)

Corresponding author: HUANG Gang, E-mail: huanggang@sei.pku.edu.cn

Abstract: Wireless sensor network (WSN) plays an important role in the field of IOT (Internet of things), which performs the function of information perception. Thousands of devices as well as sensors are spread in specific areas to collect all kinds of physical information to pass onto the Internet. However, the data gethered from sensors' interfaces is real-time, extremely large and unstructured, hence requiring great effort in mapping to the conceptual application layer. To customize and develop IOT systems more efficiently, this paper proposes an approach based on runtime model to managing wireless sensor networks. First, manageability of wireless sensors is abstracted as runtime models which automatically and immediately propagate any observable runtime changes of target resources to corresponding

* 基金项目: 国家自然科学基金(61222203, 11271002, 61103175); 教育部科技重点项目(212086)
收稿时间: 2013-12-30; 修改时间: 2014-04-29; 定稿时间: 2014-05-29

architecture models. Second, a composite model of wireless sensors is constructed through merging their runtime models in order to manage different kinds of devices in a unified way. Third, a customized model is constructed according to the personalized management requirement and the synchronization between the customized model and the composite model is ensured through model transformation. Thus, all the management tasks can be carried through executing operating programs on the customized model. The feasibility and efficiency of the approach are validated through a real case study of smart community.

Key words: wireless sensor network; software architecture; runtime model

物联网(Internet of things)是通过各种信息传感设备,按约定的协议,把任何需要监控、连接、互动的物品与互联网连接起来,进行信息交换和通信,以实现智能化识别、定位、跟踪、监控和管理的一种网络[1].其中,无线传感网是物联网的核心,主要解决物联网中的信息感知问题,通过散布在特定区域的成千上万的传感器节点,采集环境中声、光、热、电、力学、化学、生物、位置等各种信息并连接到互联网上.然而,传感设备所采集到的数据是实时的、数量庞大且无良好结构的,要将采集到的数据映射到应用场景中的客观事物对象属性,则不得不编写大量的转换代码[2],主要面临两个方面的挑战.一方面是设备类型的多样性,存在不同类型的传感设备对声、光、位置等不同的信息进行监测,而不同类型、不同品牌,甚至不同型号的传感设备往往提供不同的数据读取方式,因此,设备类型的多样性给信息的获取带来了较大的复杂度.另一方面是管理服务的随需性,存在不同类型的应用场景[1],包括智能交通、环境保护、平安家居等管理需求,传感设备在不同场景中往往刻画不同的客观事物,需要建立应用场景中客观事物与传感设备之间的联系,并实现相应的映射关系;即使在同一应用场景中,管理需求、部署环境、传感设备等要素也可能在长时间的运行过程中发生变化.因此,管理服务的随需性给信息的组织带来了较大的难度.

物联网系统实际上是对客观世界中的各种信息进行收集、分析和决策的过程,从系统实现的角度看,应用系统需要使用传感设备提供的管理接口来获取各种信息,并针对具体的应用场景对这些信息进行分析和处理.例如:在物流领域中,应用系统通过传感设备对供应链中的采购、生产、存储、分发、销售和售后等环节进行监控,从而根据市场需求对商品提供策略进行调整;在智能家居领域中,应用系统通过传感设备对住宅的光亮、温度和人员位置等信息进行监控,从而能够自动地对电灯、空调等家用设备进行管理.然而,目前的物联网系统开发往往使用 C/C++等相对底层的语言直接调用传感设备提供的管理接口,这种编程方式具有良好的适应性,但同时会带来高昂的编程开销.编程人员必须熟悉不同传感设备的管理接口,才能实现它们的交互,并在此基础上搭建应用系统.与分析和决策等应用系统的管理逻辑相比,这些繁杂、琐碎的编程工作并不是物联网系统的核心,但是为了使应用系统能够正常、有效地运行,信息获取和数据交互等底层代码开发依然需要花费编程人员大量的时间和精力.此外,由于应用系统建立在与特定传感设备绑定的底层代码的基础上,其管理逻辑无法进行复用;即使管理机制类似,编程人员仍需要开发多个应用系统来管理不同的应用场景.

物联网系统开发面临的主要问题是:其问题域与系统实现间存在着鸿沟,而通过硬编码实现问题域到系统实现的映射则会带来巨大的编程复杂性.软件体系结构用一组可管理的单元来表示系统的整体架构,能够扮演系统需求与系统实现之间的桥梁[3],常用来解决需求到实现的映射过程中系统复杂性所带来的问题[4].从软件工程的角度看,具体软件系统的知识大多隐藏在程序和文档中,模型作为文档的主要内容和程序的高层抽象,是理想的软件与管理知识的载体[5].运行时软件体系结构模型用一组可管理的单元来表示系统的整体架构,通过将隐藏在系统内部的结构、状态、配置等运行时信息显示化地描述为标准的、面向管理者视角的结构化视图,能够有效地提高物联网系统开发的抽象层次和自动化程度[6,7].运行时软件体系结构模型已经在学术界和工业界获得了广泛的关注.大量的研究工作证明了它在不同系统与管理方式下的重要作用[8−10].

为了能够根据管理需求快速定制和开发物联网系统,本文将运行时模型引入到物联网系统开发过程中,提出一种基于运行时模型的无线传感网管理方法,并在实际场景中验证方法的可行性和有效性.首先,为单一类型的传感设备构造运行时模型,基于设备管理接口实现在模型层对单一的传感设备进行监测;其次,基于运行时模型,对不同传感设备采集到的数据进行定制、抽取和合并,实现通过组合模型对场景中的不同传感设备进行统一的管理;最后,建立应用场景中客观事物对象与传感设备之间的联系,通过模型转换实现组合模型到应用场景

模型的映射,使得传感设备采集到的各种数据能够以客观世界中各种对象属性的形式表现出来.

于是,管理员就可以在业务逻辑层进行物联网系统的开发,并能够借助一些成熟模型分析规划方法和工具[11].整个方法仅需要定义一组元模型和映射规则,极大程度地降低了编码工作量.

本文第 1 节概述方法的整体框架.第 2 节介绍传感设备运行时模型的构造方法.第 3 节介绍分布式运行时模型的合并方法.第 4 节介绍组合模型到应用场景模型的转换方法.第 5 节介绍实例研究并对方法的可行性和有效性进行评估.第 6 节与相关工作进行比较.第 7 节总结全文并展望未来的工作.

1 方法概览

图 1 是基于运行时模型的无线传感网管理方法概览.方法将运行时软件体系结构模型引入到无线传感网管理过程中,通过模型构造、模型同步和模型转换来实现传感设备采集数据到应用场景中客观事物对象属性的映射.该方法主要包含 3 个部分的工作:1) 基于已有的管理接口来构造传感设备运行时模型;2) 基于分布式的传感设备运行时模型来构造组合模型;3) 通过模型转换来实现组合模型到应用场景模型的映射.

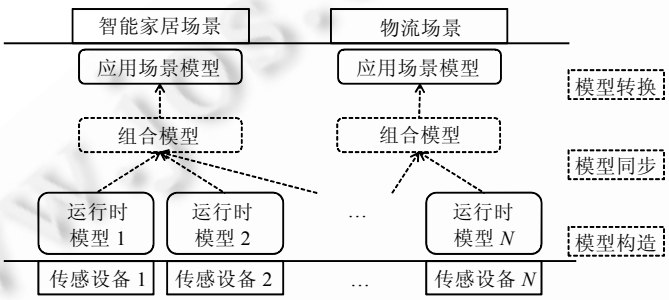


Fig.1 Overview of management approach of wireless sensor networks based on runtime model

图 1 基于运行时模型的无线传感网管理方法概览

首先,提出一种传感设备运行时模型的构造方法,以屏蔽传感设备管理接口的异构性.传感设备运行时模型是其采集到数据的一种抽象.管理员只需对采集数据的含义和数据的读取方式进行描述,构造方法就能生成相应的运行时模型,并支持运行时模型与实时数据的自动同步.于是,管理员就可以在模型层对单个传感设备进行管理.

其次,提出一种分布式运行时模型的合并方法,以屏蔽传感设备的分布性.为了实现特定的管理功能,往往需要对散布在区域内的一些特定的传感设备所采集的数据进行整合.管理员只需根据管理需求,在传感设备运行时模型的基础上,选择代表其目标数据的模型片段,该方法就能自动生成相应的组合模型,并支持组合模型与单个运行时模型的数据同步.于是,管理员就能以组合模型的形式对目标传感设备进行统一的管理.

最后,提出一种组合模型到应用场景模型的转换方法.物联网系统的开发需要适应不同的应用场景,传感设备的采集数据在不同场景中往往刻画不同客观事物的对象属性.管理员仅需要根据管理需求,定义组合模型与应用场景模型间的元素关联关系,转换方法就能自动生成相应的模型转换程序,以实现组合模型到应用场景模型的映射.于是,传感设备采集到的各种数据就能以客观世界中事物对象属性的形式表现出来.

2 传感设备运行时模型的构造方法

无线传感网是物联网的核心,主要解决物联网中的信息感知问题,而传感设备的多样性和异构性给信息的获取带来了极大的难度和复杂度.因此,本文借助 SM@RT 工具[12,13]构造传感设备运行时模型,以统一的方式对信息进行采集和处理(SM@RT 源代码可以从文献[14]下载获得).SM@RT(supporting model at run time)是由所在研究团队提出的一种模型驱动的运行时软件体系结构构造方法及工具,包括特定领域建模语言(SM@RT language)和代码生成器(SM@RT generator).SM@RT language 允许用户定义运行时软件体系结构的元模型及

访问模型.元模型定义了目标系统的结构及可管理的元素.访问模型声明了元模型中管理这些元素的方法,即调用目标系统的API.SM@RT generator在元模型和访问模型作为输入的基础上,能够自动生成维护运行时软件体系结构的基础设施,将底层系统的实时状态反映到运行时模型.如图 2 所示,id 为 158 的 RFID(radio frequency identification device)标签从 1434 房间移动到了 1621 房间;同步引擎检测到 1621 房间新增了一个 RFID 标签,并自动在 id 为 1621 的 Reader 元素(RFID 读取器)的关联元素中增加一个 Tag 元素(RFID 标签);同时,同步引擎也检测到 id 为 158 的 RFID 设备离开了 1434 房间,并自动在 id 为 1434 的 Reader 元素的关联元素中删除这个 Tag 元素.在之前的工作中,我们对运行时模型的构造方法与支撑机制进行了深入的研究[15,16],并对传感设备运行时模型的构造方法进行了初步探讨[17],本文不再赘述.

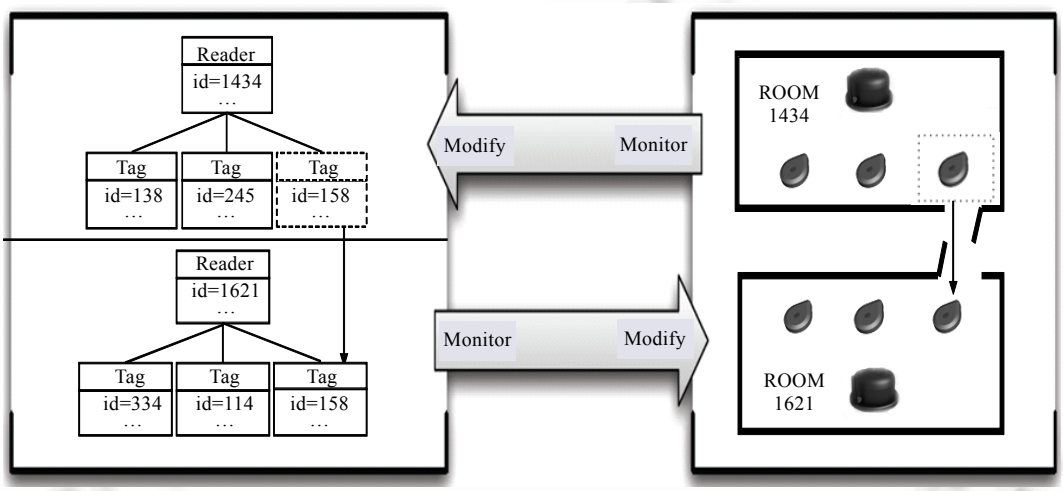


Fig.2 Synchronization between runtime models and wireless sensor devices

图 2 运行时模型与无线传感设备的同步

3 分布式运行时模型的合并方法

无线传感网往往由众多不同类型的传感设备构成,应用系统需要根据管理目标对不同传感设备所采集到的数据进行统一的分析和处理.例如,安保系统需要对区域内所有位置感知设备进行监控,而节能系统需要对区域内所有照明设备和亮度感知设备进行监控.这些传感设备需要被统一管理起来,以实现不同类型的复杂管理任务.本文提出一种分布式运行时模型的合并方法,在传感设备运行时模型的基础上,以组合模型的形式对其进行重组织,实现不同类型传感设备的统一管理,主要工作包括模型定制和数据同步.

3.1 模型定制

模型定制是指,针对特定的管理需求,在多个传感设备运行时模型中截取相应的模型片段,将它们组合成一个新的模型,即组合模型.

图 3 所示场景是对 id 为 1621 的会议室的照明设备进行管理,需要同时用到采集位置信息和亮度信息的传感设备,根据室内的人员和亮度情况来决定照明设备的开关状态.传感设备分别提供了位置信息的采集系统和环境信息的采集系统,因此,我们在以上两种设备的运行时模型基础上进行模型的定制,截取 id 为 1621 的位置信息采集系统的整个模型以及环境信息采集系统运行时模型中 id 为 1621 的 Tag 元素,将它们合并为一个新的特定于该管理需求的组合模型.如图 3 所示,组合模型中 Reader 元素及其关联的 Tag 元素反映了会议室内的人员情况,Esensor 元素的 brightness 属性反映了会议室内的亮度情况.当会议室正在使用且室内亮度不足时,则打开照明设备.

定制的模型片段需要能够从特定的传感设备运行时模型中获取目标设备所采集到的信息.传感设备运行

时模型与管理员定制的模型片段均是以XML文件的形式进行存储,对于模型中的每一个元素,都有且仅有一条从根节点出发的路径可以定位到该元素.如图4所示,管理员定制的模型片段(如图4(a)所示)和相应的运行时模型(如图4(b)所示)有着类似的组织结构.用户所定制的每一个元素都可以从运行时体系结构的根节点开始,根据定制的路径一层一层地找到相对应的元素,从而实现整个模型片段和运行时模型的数据关联.例如,id为1621的Tag元素的brightness属性在管理员定制的模型片段中,可以通过路径"⟨EnSystem⟩→⟨Tags⟩→⟨Tag id=1621⟩→brightness"找到,而在运行时模型中的查询路径也是类似的.

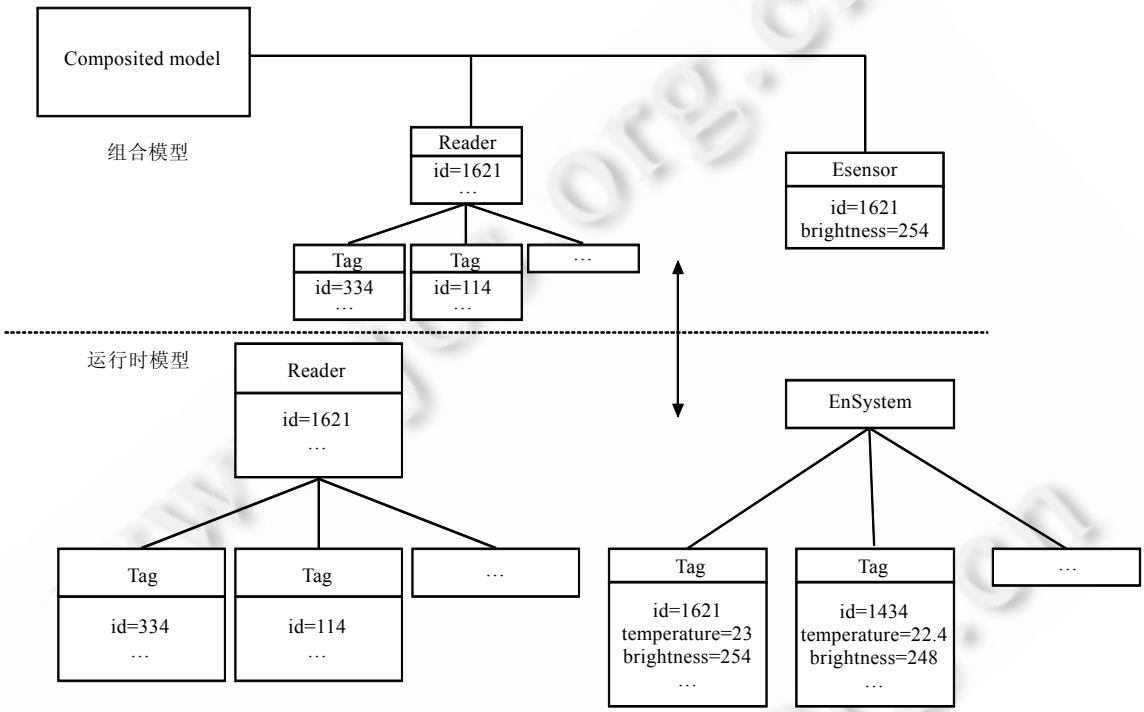


Fig.3 An example of composite model construction

图3 组合模型构造实例

```
<?xml version="1.0" encoding="ASCII"?>
<En_Sys:EnSystem xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://sei.pku.edu.cn/EnSystem/En_Sys">
    <Tags>
        <Tag id="1621" brightness="254" humidity="52" temperature="23" />
        <Tag id="1434" brightness="248" humidity="61" temperature="22.4" />
        ...
    </Tags>
</En_Sys:EnSystem>
```

(a) 系统运行时模型

```
<EnSystem>
        <Tags>
            <Tag id="1621">
                <property>brightness</property>
            </Tag>
        </Tags>
</EnSystem>
```

(b) 模型片段定制描述

Fig.4 Fragment recognition in runtime models of wireless sensor devices

图4 传感设备运行时模型中片段的识别

组合模型中的模型片段来自不同的传感设备运行时模型,它们不存在语法上的关联.因此,我们将每一个模型片段的根节点作为组合模型根节点的直接子节点进行合并.同时,为了解决各个模型片段间的元素命名冲突,我们还需要维护一个命名空间,对冲突元素的命名进行替换和记录.如图 5 所示,将 Reader 与 EnSystem 两个模型片段进行合并,而两个模型片段中均包含 Tags 和 Tag 元素,存在命名冲突.于是,将 EnSystem 模型片段中的 Tags 和 Tag 元素更改命名为 Esensors 和 Esensor 元素,再进行模型片段的合并.

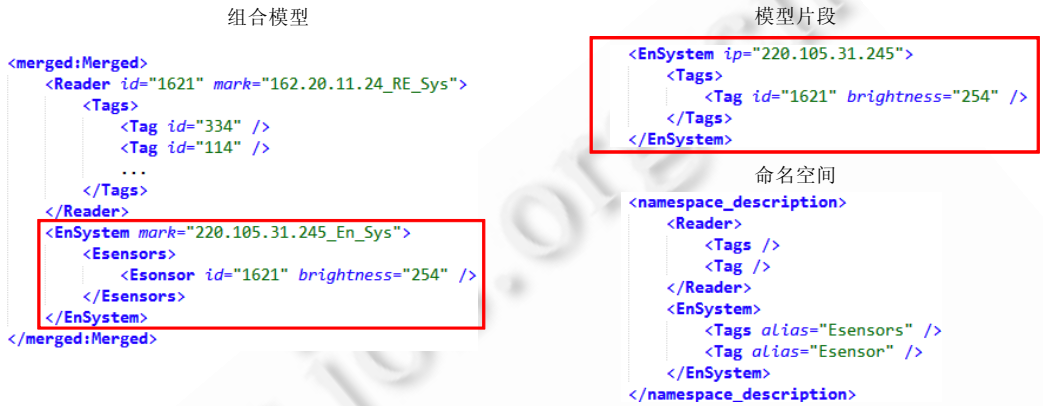


Fig.5 Solution to naming conflict in composite models

图 5 组合模型中元素命名冲突的解决方法

3.2 数据同步

在模型定制的基础上,只有维护组合模型与系统运行时模型的数据同步,才能实现组合模型对多个系统的统一管理.组合模型由不同的系统运行时模型元素构成,其数据一致性由不同的模型片段分别完成.我们在系统运行时模型处部署一个模型片段副本,通过轮询机制比较发现模型的变化,自动生成相应的模型操作,发送给组合模型并执行.所有模型操作类型中,只有 Set,Add 和 Remove 操作能够产生相应的模型变化,图 6 对其描述规则和产生效果进行了明确的定义.

Add	Description	⟨action node=“TypeS” type=“add”⟩ ⟨query node=“TypeF” condition=“Constraint”/⟩ ⟨set key=““value=””/⟩ ⟨set key=““value=””/⟩ ⟨/action⟩
	Post condition	∃TypeS *s*, ∃TypeF *f*, *s*∈*f*∧*f* incondition of Constraint∧*props*⊆*s.properties*
	Illustration	在模型中,寻找类型为“TypeF”且满足约束“Constraint”的元素, 以其为父节点,添加一个类型为“TypeS”的子节点,并为其属性赋值
Set	Description	⟨action key=““value=”” type=“set”⟩ ⟨query node=“Type” condition=“Constraint”/⟩ ⟨/action⟩
	Post condition	∃Type *n*, *n* incondition of Constraint∧*prop*∈*n.properties*
	Illustration	在模型中,寻找类型为“Type”且满足约束“Constraint”的元素, 将其属性 key 的值赋为 value
Remove	Description	⟨action node=“Type” condition=“Constraint” type=“remove”/⟩
	Post condition	∀Type *n*, *n* notincondition of Constraint
	Illustration	在模型中,寻找类型为“Type”且满足约束“Constraint”的元素,将其删除

Fig.6 Three kinds of model operations

图 6 3 种模型操作

当系统运行时模型发生变化时,为了维护组合模型与系统运行时模型的数据一致性,需要能够感知这一变化,并将其反馈给组合模型.这一过程的难点在于模型变化的发现和相应模型操作的生成,我们是通过深度优先

算法来实现的:

a) 根据模型片段的定制描述,在系统运行时模型中抽取新的模型片段;
b) 从旧模型片段的根节点开始,依次对每一个节点在新旧模型片段中进行比较:若该节点是根节点,则比较是否相同,若不同则报错,若相同则执行步骤 d);若该节点不是根节点,则执行步骤 c);
c) 检查新的模型片段中是否有此节点:若有,则执行步骤 d);若没有,则根据该子节点的信息生成相应的“Remove”操作,执行步骤 f);
d) 检查新的模型片段中此节点的属性值是否有变化:若没有,则执行步骤 e);若有,则根据该子节点信息生成相应的“Set”操作,执行步骤 e);
e) 在新旧模型片段中,对此节点的子节点进行比较,检查新的模型片段中是否有子节点的增加:若没有,则执行步骤 f);若有,则根据该子节点信息生成相应的“Add”操作,执行步骤 f);
f) 在旧模型片段中寻找下一个节点:若存在下一个节点,则重复步骤 b);若不存在,则算法结束.

4 组合模型到应用场景模型的转换方法

在不同的应用场景中,传感设备采集的数据往往用来刻画不同的客观事物,需要建立应用场景中客观事物对象属性与传感设备采集到的数据之间的关联关系.如图 1 所示:应用场景模型描述了特定管理场景中的客观事物,而组合模型则表示受管传感设备采集到的数据的集合;应用场景模型中元素及其属性信息来自组合模型,需要建立它们间的关联关系,使得能够通过应用场景模型直观的获取客观事物的状态信息.本文方法通过构造组合模型与应用场景模型间的元素映射关系来实现组合模型到应用场景模型的转换,并维护应用场景模型中元素属性与组合模型中相对应元素属性的值相等.特别地,组合模型到应用场景模型的转换需要编写模型转换程序来实现;本文完成了一种模型转换方法,能够根据组合模型、应用场景模型以及它们间的元素映射规则,自动生成相应的模型转换程序.图 7 展示了模型元素间的 3 种基本映射关系:

1. 模型元素间的“一对一”映射关系.组合模型中,一种类型的元素与应用场景模型中一种类型的元素对应.特别地,应用场景模型中元素的属性可以在组合模型中的对应元素中找到对应的属性.在组合模型向应用场景模型发生转换时,应用场景模型中元素的属性会根据其在组合模型中关联元素的对应属性进行赋值.例如,组合模型中 Power 类型的元素与应用场景模型中 Light 类型的元素均表示照明设备,且 Power 元素的属性 id 和 switch 在 Light 元素中存在对应的属性 id 和 on_off,因此,Power 类型的元素与 Light 类型的元素存在“一对一”映射关系.
2. 模型元素间的“多对一”映射关系.组合模型中两种或多种类型的元素与应用场景模型中一种类型的元素对应.特别地,应用场景模型中元素的属性在组合模型中的对应属性分布在两种或多种元素中.在组合模型向应用场景模型发生转换时,应用场景模型中的元素需要在组合模型中查询与之相对应的两种或多种元素来进行其属性的赋值.例如,组合模型中 Reader 类型的元素与应用场景模型中 Room 类型的元素均表示房间,但是应用场景模型 Room 元素的 brightness 属性无法在组合模型 Reader 元素中找到对应属性,其对应属性在组合模型 Esensor 元素中(模型转换时,可以在组合模型中查找得到 id 属性值与 Reader 元素相同的 Esensor 元素).
3. 模型元素间的“一对多”映射关系:组合模型中,一种类型的元素与应用场景模型中两种或多种类型的元素对应.当模型转换时,组合模型中的元素需要根据自身的属性信息,选择应用场景模型中一种特定类型的元素进行映射.例如:组合模型中 Tag 类型的元素表示 RFID 标签,应用场景模型中 Person, Pet,Car 类型的元素表示携带 RFID 标签的人、宠物或物品;模型转换时,组合模型中的 Tag 元素会根据其 type 属性值映射为应用场景模型中 Person,Pet 或 Car 类型的一个元素.

组合模型与应用场景模型间的元素映射关系均可以表示为以上 3 种基本映射关系的组合.本文设计了一套映射关系描述规则及其自动生成代码的方法,使得管理员能够通过定义模型间的元素映射规则来得到相应的模型转换程序(QVT 语言[18]).模型间的元素映射规则通过一个 XML 文件进行描述.关键字定义如下:

1. helper:用于描述元素之间的映射关系.helper 标签一般包含 3 个属性,分别是属性 key,value 和 type.value 表示组合模型中的元素,key 表示应用场景模型中对应的元素,type 表示元素映射关系的类型.当 type 取值为“basic”时,表示该 helper 表示的是元素间的“一对一”映射关系或“多对一”映射关系;当 type 取值为“multi”时,表示该 helper 表示的是元素间的“一对多”映射关系,此时,helper 往往成组出现,并用属性 condition 描述映射发生的条件.由于元素通常还会包含属性或其他元素,因此,helper 标签里通常会嵌套 helper 标签、mapper 标签和 query 标签,其中,mapper 标签和 query 标签均表示属性间的映射关系.
2. mapper:用于描述属性之间的映射关系.mapper 标签通常包含两个属性——key 和 value.value 表示组合模型中的属性,而 key 表示应用场景模型中对应的属性.特别地,属性所属的元素由外层的 helper 标签定义.
3. query:用于描述属性之间的映射关系.query 标签通常包含 4 个属性,分别是属性 key,value,node 和 condition.其中,属性 key 和 value 的定义与 mapper 标签类似,而应用场景模型中属性所属的元素也是由外层的 helper 标签定义.然而,组合模型中属性所属的元素是由其他两个属性 node 和 condition 定义的,它们分别表示元素的类型和需要满足的条件.特别地,query 标签常用来描述元素间的“多对一”映射关系.

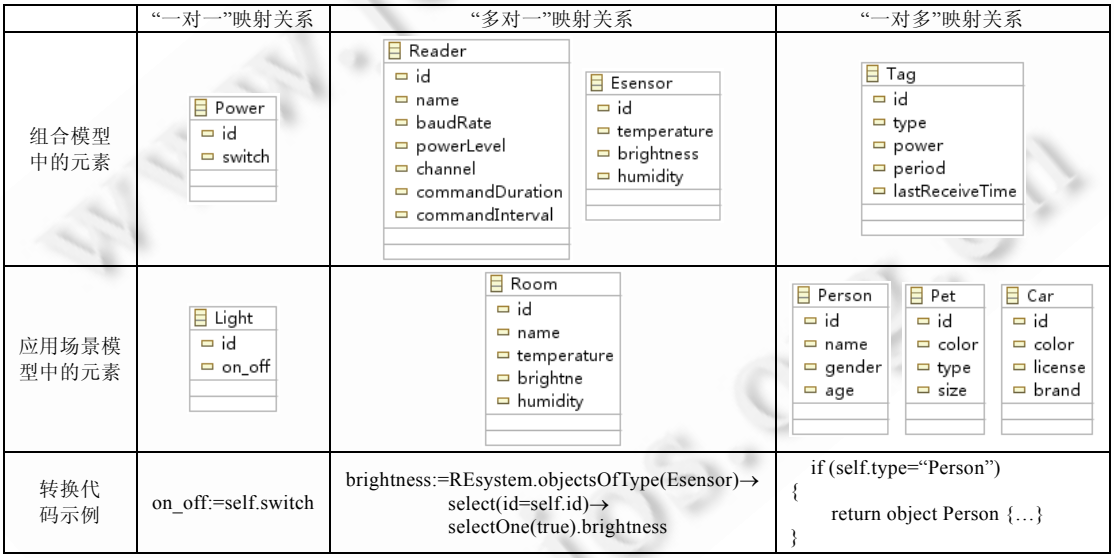


Fig.7 Three types of basic mapping rules between model elements

图 7 模型元素间的 3 种基本映射关系

如图 8 所示,在以上关键字的基础上,我们就可以根据模型映射关系,对元素间的映射规则进行描述了:

1. 模型元素间的“一对一”映射关系.第 1 个例子是描述组合模型中 Power 类型元素与应用场景模型中 Light 类型元素间的“一对一”映射关系,其中,用 helper 标签表示 Power 元素到 Light 元素的映射,用 mapper 标签表示 Power 元素中属性 id,switch 到 Light 元素中属性 id,on_off 的映射.
2. 模型元素间的“多对一”映射关系.第 2 个例子是描述组合模型中 Reader 类型元素和 Esensor 类型元素与应用场景模型中 Room 类型元素间的“多对一”映射关系.组合模型中 Reader 元素和应用场景模型中 Room 元素均表示房间,因此,用 helper 标签和 mapper 标签来描述这两种类型的元素及其属性间的映射.然而,Room 元素比 Reader 元素包含更丰富的信息,这些信息存在于 Esensor 元素中.因此,用 query 标签来描述 Esensor 元素的属性到 Room 元素对应属性的映射.query 标签中的 key 和 value 分

别表示应用场景模型和组合模型中的元素属性;node 属性值为 Esensor,表示属性所属元素的类型为 Esensor,而 condition 属性值“id=self.id”则用来描述元素需要满足的条件.

3. 模型元素间的“一对多”映射关系.第 3 个例子是描述组合模型中 Tag 类型元素与应用场景模型中 Person,Pet 和 Car 类型元素间的“一对多”映射关系.Tag 元素可能映射为 Person,Pet 或 Car 元素,用一组 type 值为“multi”的 helper 标签来表示这一映射.例如,当 Tag 元素的 type 属性值为“Person”时,组合模型中的 Tag 元素映射为应用场景模型中的 Person 元素.

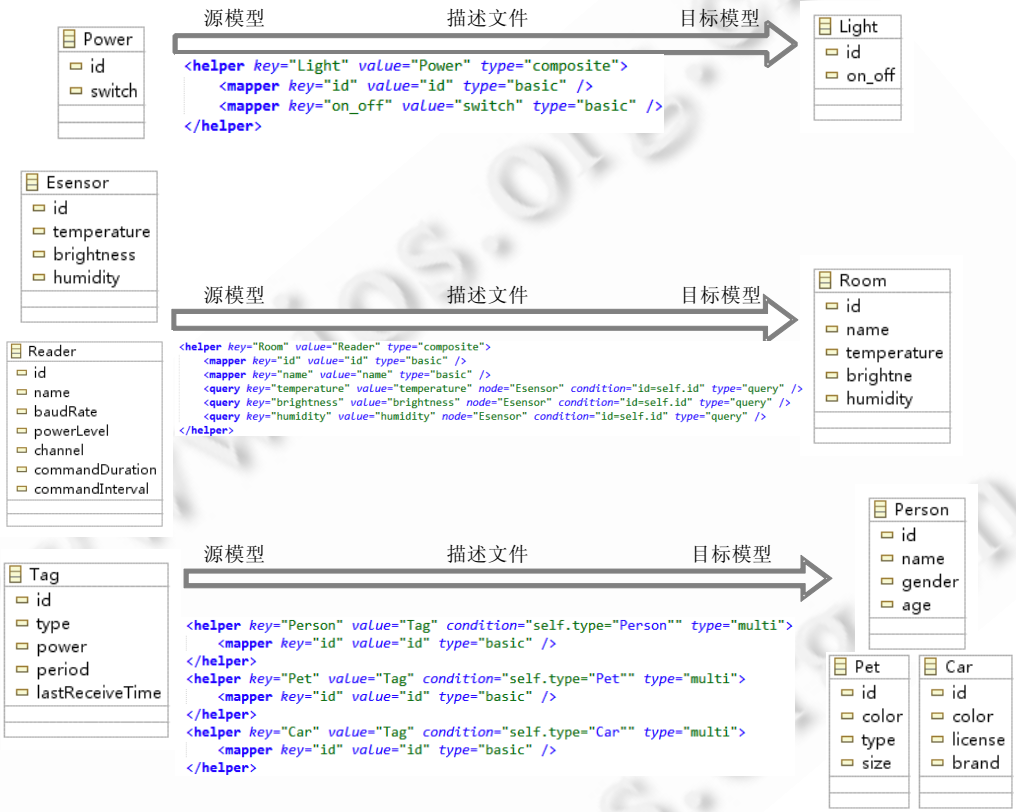


Fig.8 Descriptions of basic mapping rules between model elements

图 8 模型元素间基本映射关系的描述

映射规则与模型转换代码存在强相关性:映射规则中的每一个 helper 标签生成转换代码中的一个方法,实现元素的映射;映射规则中的每一个 mapper 标签生成转换代码中的一条简单赋值语句,实现属性的映射;映射规则中的每一个 query 标签生成转换代码中的一条查询赋值语句,实现属性的查找及映射.通过上述定义,模型转换代码可以在映射规则基础上自动生成,如图 9 所示.

5 实例研究

智慧社区是物联网重要应用领域之一,然而多样化的传感设备和不断变化的居民需求,给其应用系统的开发带来了巨大的难度和复杂度.为了验证方法的可行性和有效性,本文针对儿童、宠物和车辆的安全保障问题,在 RFID 感知设备和居民信息系统的基础上实现了基于运行时模型的险情通报系统**,通过居民、儿童和宠物

**该系统为“祥云智慧社区”项目的核心系统之一.该项目获得第 5 届全国大学生创新创业年会“创业项目奖”等奖项.本文作者作为项目负责人受到央视《经济信息联播》的采访(http://jingji.cntv.cn/2013/06/17/VIDE1371476880975235.shtml).

等位置信息的关联关系发出险情通告.例如,在宠物独自离开社区时发出警告:首先,构造了 RFID 感知设备和居民信息系统的运行时模型;其次,通过组合模型对目标信息进行了抽取和合并;最后,根据险情通报的管理需求构造应用场景模型,并实现了组合模型到应用场景模型的转换.此外,本文还将方法与传统解决方案进行了比较,对方法的可行性和有效性进行了评估.

一对一

```
helper Power :: Power2Light : Light{
    return object Light{
        id:=self.id;
        on_off:=self.switch;
    }
}
```

一对多

```
helper Tag :: Tag2Person : Person{
    if (self.type = "Person"){
        return object Person{
            id:=self.id;
        }
    }
}
...
```

多对一

```
helper Reader :: Reader2Room : Room{
    return object Room{
        id:=self.id;
        name:=self.name;
        temperature:=EnSystem.objectOfType(Esensor)->select(id=self.id)->selectOne(true).temperature;
        brightness:=EnSystem.objectOfType(Esensor)->select(id=self.id)->selectOne(true).brightness;
        humidity:=EnSystem.objectOfType(Esensor)->select(id=self.id)->selectOne(true).humidity;
    }
}
```

Fig.9 Example of the code generation based on mapping rules

图 9 基于映射规则的代码生成示例

5.1 RFID感知设备与居民信息系统的运行时模型

RFID 感知设备能够确定人员与物品的位置信息,主要包含 RFID 读写器和 RFID 标签;在社区内的关键地区布置 RFID 读写器,并为每一个目标人员或物品发放 RFID 标签;RFID 读写器可以实时采集到区域内的标签信息,以此能够判断目标人员或物品所在的位置.图 10 描述了 RFID 感知设备包含的主要元素及其属性,其中:Reader 描述了 RFID 读写器的基本配置信息,并包含一个 RFID 标签的列表;Tag 则描述了 RFID 标签的基本状态信息.Reader 元素与 Tag 元素的关联关系,反映了 Tag 元素代表的 RFID 标签的位置信息.当 RFID 标签的位置发生变化时,其 Tag 元素相关联的 Reader 元素也会改变.

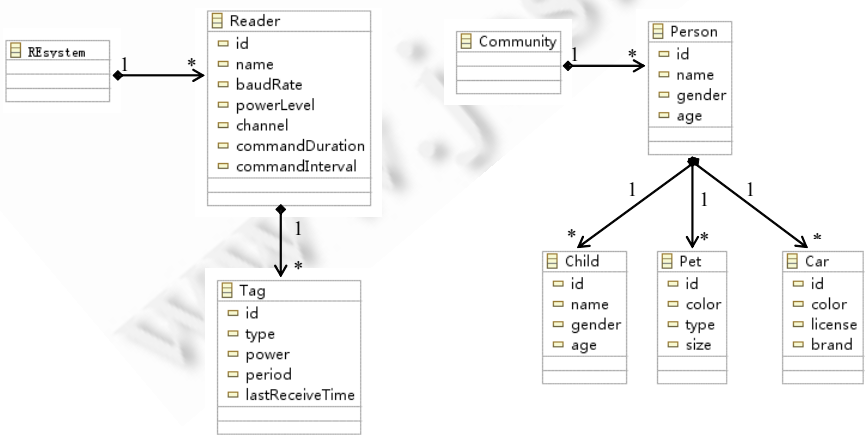


Fig.10 Architecture-Based models of RFID system and information system

图 10 RFID 系统与信息系统的体系结构模型

居民信息系统汇总了社区内居民及其所属物品的信息,主要包含居民(person)、儿童(child)、宠物(pet)和车辆(car)等类型的对象;信息系统同时记录了上述对象间的关联关系,其中,儿童、宠物、车辆与居民间存在关联关系.图 10 描述了居民信息系统包含的主要元素:Person 元素描述了居民的基本信息,并包含关联的儿童、宠物和车辆的列表;Child,Pet 和 Car 分别描述了儿童、宠物和车辆的基本信息,均包含一个关联人列表.

基于 RFID 感知设备运行时模型,可以在模型层获取 RFID 读写器与标签的基本信息和关联关系;基于居民信息系统运行时模型,则可以在模型层获取居民及其所属物品的基本信息和关联关系.

5.2 RFID感知设备与居民信息系统的组合模型

险情通报系统需要根据居民、儿童、宠物和车辆的关联关系及位置信息判断险情并发出警告,因此,本文以模型片段的形式,从 RFID 感知设备运行时模型中抽取对象位置信息,从居民信息系统运行时模型中抽取居民及其所属物品信息,将它们组合成一个新的模型.如图 11 所示,组合模型中的元素及其属性均来自运行时模型,且同时包含了上述两个方面的信息;通过组合模型,就可以确定居民及其所属物品的具体位置.例如,王虎携带的 RFID 标签的编号为 1243,而 1243 号 RFID 标签与 083 号的 RFID 读取器具有关联关系,且 083 号 RFID 读取器安装在社区南门,那么,王虎所在的位置为社区南门.

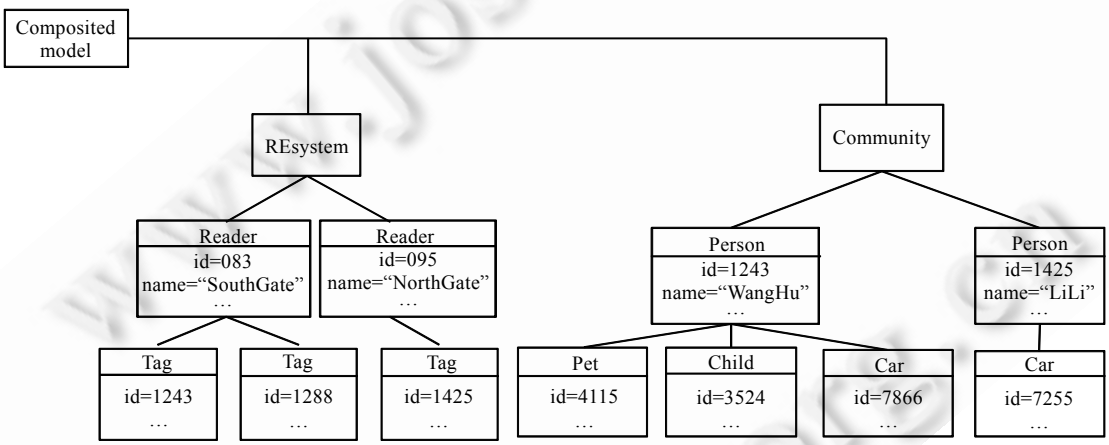


Fig.11 Composite model of alarm system

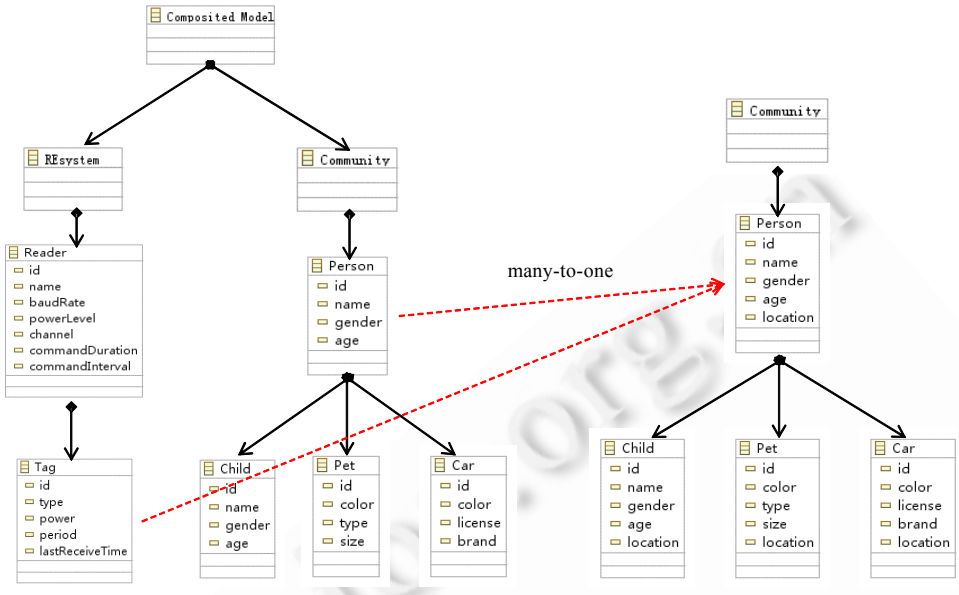
图 11 报警系统组合模型

5.3 组合模型到应用场景模型的转换

尽管组合模型中包含了居民、儿童、宠物、车辆的基本情况、关联关系及其位置信息,但是管理员无法从组合模型的对象属性中直接获取目标数据,仍然需要编写额外的代码对组合模型进行分析.险情通报系统需要直观地获取居民、儿童、宠物和车辆的位置信息及其关联关系,如图 12 所示,本文构造了险情通报系统的应用场景模型:Person 元素描述了居民的身份和位置信息,并包含关联的儿童、宠物和车辆的列表;Child,Pet 和 Car 分别描述了儿童、宠物和车辆的身份和位置信息,均包含一个关联人列表.

进一步地,需要建立组合模型与应用场景模型间的关联关系.组合模型中 Person 元素、Child 元素、Pet 元素、Car 元素和 Tag 元素到应用场景模型中 Person 元素、Child 元素、Pet 元素和 Car 元素的映射关系是整个模型转换的关键.下面以组合模型中 Person 元素和 Tag 元素到应用场景模型中 Person 元素的映射为例,介绍该映射关系的描述方法.

组合模型中的 Person 元素与应用场景模型中的 Person 元素均表示居民,然而,组合模型中居民的位置信息以 Reader 元素与 Tag 元素关联关系的形式进行表示,而应用场景模型中居民的位置信息则用 Person 元素的 location 属性来描述.因此,该映射实际上是组合模型中 Person 元素和 Tag 元素到应用场景模型中 Person 元素的"多对一"映射.



Mapping description

```
<helper key="Person" value="Person" type="composite">
        <mapper key="id" value="id" type="basic" />
        <mapper key="name" value="name" type="basic" />
        <mapper key="gender" value="gender" type="basic" />
        <mapper key="age" value="age" type="basic" />
        <query key="location" value="Reader.name" node="Tag" condition="id=self.id" type="query" />
</helper>
```

Fig.12 Descriptions of mapping rules of model transformation

图 12 模型转换的映射规则描述

图 12 展示了该映射的映射关系描述,其中,应用场景模型中 Person 元素的 location 属性由组合模型中对应 Tag 元素相关联的 Reader 元素的 name 属性决定.

通过组合模型到应用场景模型的转换,组合模型中的任何变化都能自动反映在应用场景模型中.于是,居民、儿童、宠物和车辆的位置信息及其关联关系就能以客观事物对象属性的形式表现出来.

5.4 方法评估

我们从以下 3 个方面进行评估.

5.4.1 基于运行时模型的位置感知系统开发

本文开发了基于运行时模型的位置感知系统.如图 13 所示,应用场景模型中的每一个模型元素代表一个实际存在的个体,其位置信息和从属关系以元素属性及关联关系的形式表现,并且与客观事物的实时状态保持一致.因此,管理员能够通过应用场景模型对小区内居民、儿童、宠物和车辆的位置进行监测.构造特定类型传感设备的运行时模型,管理员仅需要使用 Eclipse 模型框架(eclipse modeling framework[19])定义其元模型和访问模型,传感设备运行时模型就能够在 SM@RT 工具基础上自动生成;同时,传感设备运行时模型只需构造 1 次,就能在不同的物联网应用场景中进行复用.因此,基本不会带来额外的工作量.在运行时模型基础上,管理员只需提供模型片段的定制描述和模型转换的映射规则,就可以得到相应的物联网应用场景模型,并能够以对象化的方式访问传感设备采集到的各种数据.在这一过程中,管理员不需要熟悉各种传感设备及其监控接口,仅需要在模型层进行一些规则的定义.因此,构造物联网应用场景模型所带来的额外工作量是可以接受的.

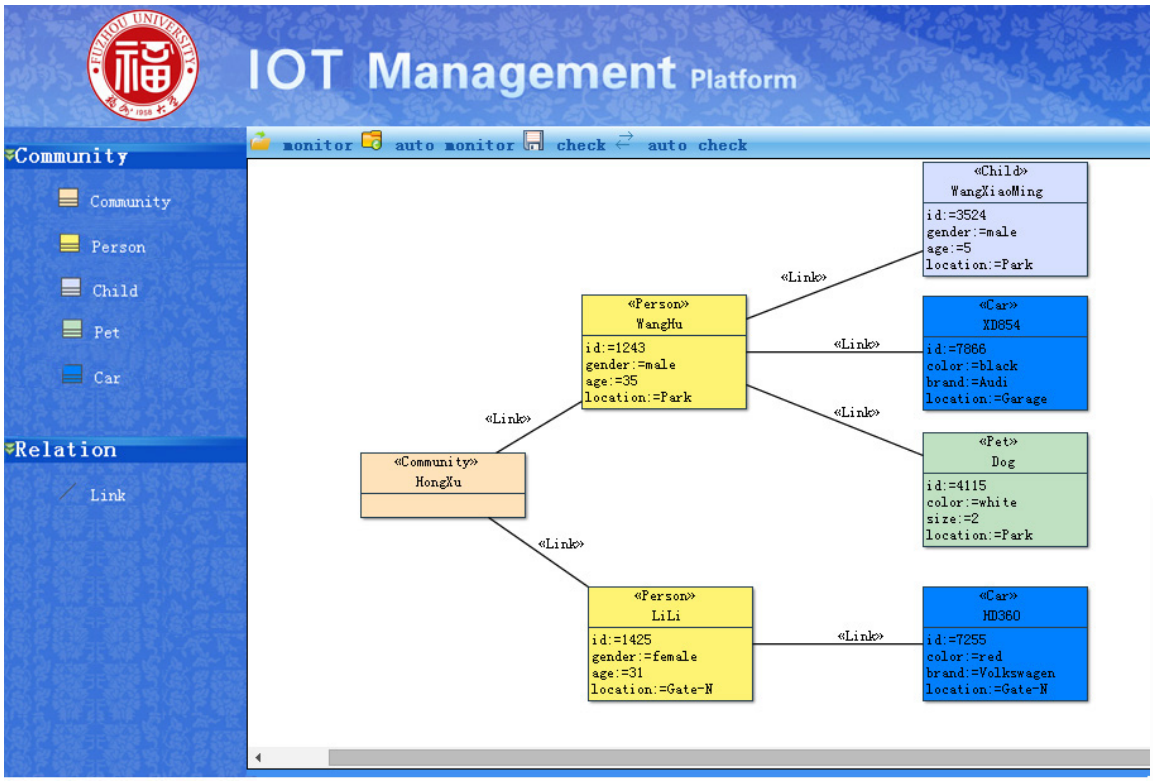


Fig.13 Location-aware system based on runtime model

图 13 基于运行时模型的位置感知系统

5.4.2 通用语言与模型语言的编程复杂度比较

基于应用场景模型,传感设备采集到的各种数据能够以客观事物对象属性的形式表现出来,极大地降低了熟悉各种不同类型传感设备的管理接口带来的难度和复杂度.为了验证方法的可行性和有效性,本文针对险情通报系统,对基于模型方法和传统方法的编程难度、复杂度以及程序执行效率进行比较.险情通报系统是通过监测居民、儿童和宠物等位置信息的关联关系来发出险情通告,本文分别用模型语言 QVT[18]和传统语言 Java 实现了这一功能,例如在宠物独自离开社区时发出警告.居民信息系统中存储了居民、宠物等的信息及所持的 RFID 标签,而根据部署在社区关键位置的 RFID 读取器可以得出不同 RFID 标签的大致范围,从而判断居民、宠物等所处的位置并分析是否需要发出警告.如图 14 所示,实现相同的管理功能,用 QVT 语言与用 Java 语言相比,其编程难度和复杂度都要小得多,QVT 程序的代码行数不到 Java 程序的 1/5.一方面,Java 程序中需要编写大量的代码来访问传感设备采集到的各种数据,虽然这些代码不是管理逻辑的核心,但其开发依然需要花费编程人员大量的时间和精力.另一方面,模型语言包含了一些模型层的特殊操作,例如“select”表示选出符合某种条件的特定类型的对象,这些操作提供了许多常见的复杂功能,简化了管理逻辑的实现.

5.4.3 管理接口与运行时模型的性能比较

进一步地,本文对基于运行时模型的 QVT 程序与基于传感设备管理接口的 Java 程序的执行效率进行了比较.如图 15 所示,与 QVT 程序相比,Java 程序的执行时间较短.其主要原因包含以下两个方面:第一,在传感设备数据读取过程中,两种语言的程序均是基于相同的访问接口实现的;而在模型方法中,需要额外的一些操作来维护运行时模型与传感设备间的数据同步;第二,在 RFID 标签配对及查询过程中,虽然模型语言提供了一些模型层的复杂操作,但是使用传统语言编程在性能上往往更加有效.然而从系统管理的角度来看,这种执行时间上的差异是可以接受的.

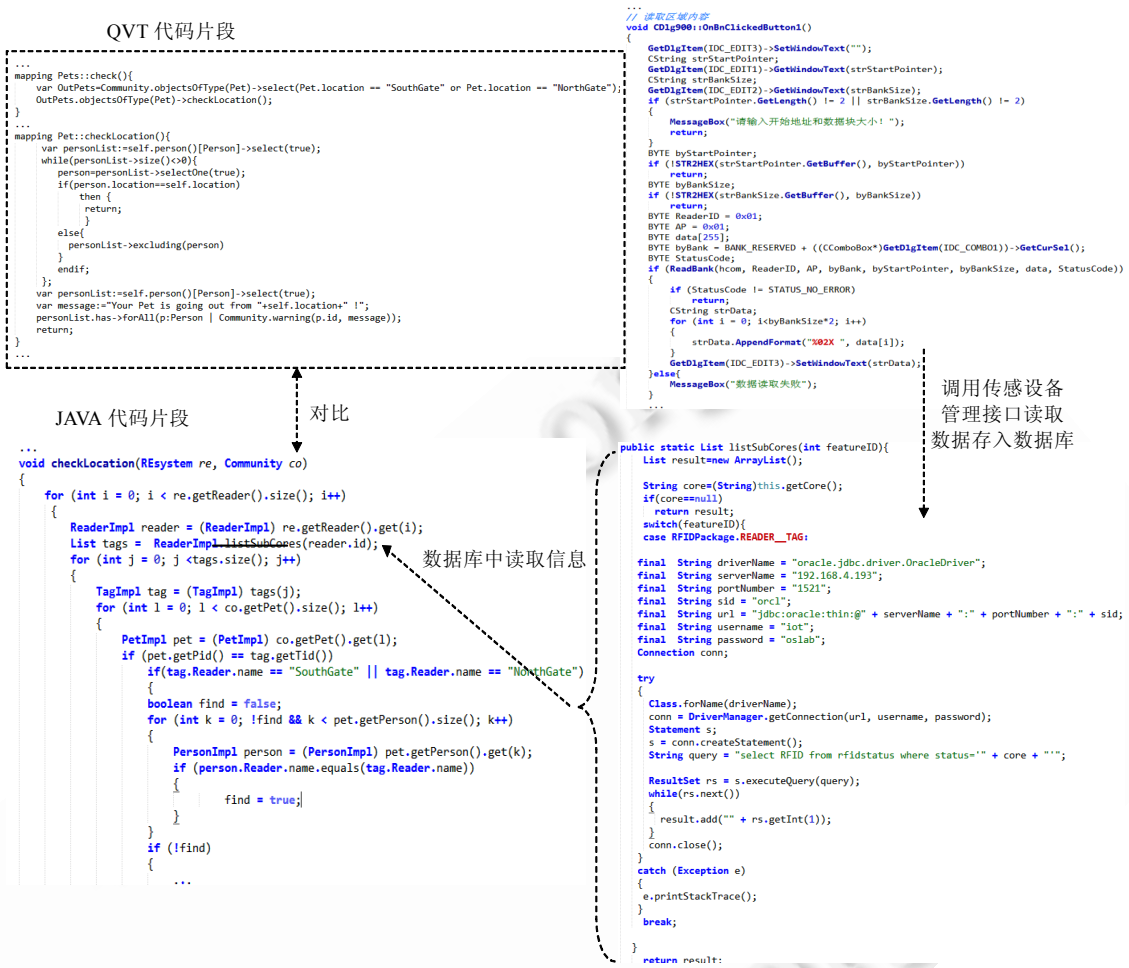


Fig.14 QVT and Java code of alarm system

图 14 报警系统的 QVT 与 Java 代码

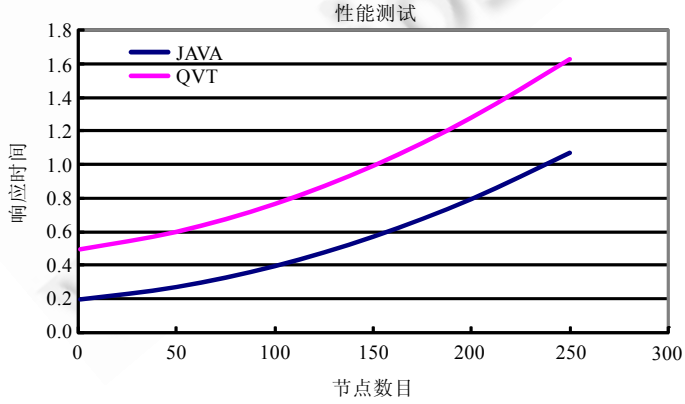


Fig.15 Performance comparison between the QVT and Java programs

图 15 QVT 程序与 Java 程序的性能比较

6 相关工作

在当前的物联网系统开发过程中,编程工作一般都接近操作系统级别,这要求程序员对底层系统的相关技术有非常深入的了解,并使程序员将注意力集中在底层系统的相关问题上,而不是应用逻辑本身[2].因此,数十种编程语言被提出,希望在保证效率的情况下,尽可能地简化物联网系统的开发过程.例如, nesC[20]对 C 语言进行了扩展,在保持 C 语言代码高效特点的同时,允许开发人员以事件驱动的方式进行编程,并为系统安全特性的代码编写提供了支持.TeenyLIME[21]在 nesC 的基础上,采用元组空间作为数据访问模型对节点间通信进行了封装.TinyDB[22]对整个无线传感网进行抽象,将其视为一个数据库系统,并提供了类 SQL 的声明式语言对节点数据进行访问.尽管这些编程语言在一定程度上降低了物联网应用的开发难度,但是节点系统异构、应用系统与底层节点紧耦合等问题依然困扰着开发人员.同时,一些研究工作[23,24]试图基于面向服务的体系结构来解决节点数据访问的问题,将传感设备采集到的数据进行封装,并提供 RESTFul 服务形式的访问接口.然而,这些工作并没有对节点系统进行抽象,开发人员还是需要面向底层数据进行编程.

模型驱动的开发方法能够对底层系统进行抽象,有助于解决应用系统与底层节点紧耦合的问题[25].已有一些工作[26−30]对无线传感网上的应用系统的建模方法进行研究.例如,文献[26]将整个模型的体系结构分成了 3 层,分别是 DSL 描述的领域相关模型、UML 描述的平台独立模型和支持代码自动生成的平台相关模型,从而使得开发人员可以在业务逻辑层对无线传感网应用系统进行描述.文献[27]则提供了一个通用元模型,允许用户在功能性需求和非功能性需求两个方面进行建模,并能够进一步拓展成为领域相关元模型和平台相关元模型.这些工作将模型方法应用于应用开发的需求阶段和设计阶段,有效提高了物联网编程的抽象层次.然而,物联网系统需要能够在长时间运行过程中适应需求、环境及其自身的变化,而传统的模型方法却无法在系统运行过程中对管理逻辑进行修改和调试.

目前,运行时模型被广泛应用在不同的系统中来支持系统自修复[31]、动态自适应[32]等管理功能.我们的研究团队在模型驱动工程方面也进行了深入的研究:给定系统元模型与一组管理接口,SM@RT 工具[12]就能自动生成代码,在保证性能的前提下实现模型到管理接口的映射;当系统元模型发生变化时,SM@RT 可以自动生成新的映射代码.前期工作[13]对以上内容进行了详细的论述.同时,为了弥补建模语言本身的非完全形式化问题,我们所在的研究团队在模型分析及模型容错方面进行了研究.前期工作[33]提出一种 MOF 元模型扩展机制以支持元模型的向上兼容,从而实现在模型集成过程中模型的自动转换.该方法在体系结构级别的系统容错实践[34]中进一步得到了验证.在之前的工作[17]中,我们所在的研究团队还尝试搭建了智慧实验室应用场景的运行时模型,并在此基础上开发了若干典型的管理应用.然而,对于每一个管理场景,需要建立全新的运行时软件体系结构模型,开发代价较高.本文方法建立在以上前期工作的基础上.

7 结束语

传感设备所采集到的数据是实时的、数量庞大并且无良好结构的,要将采集到的数据映射到物联网系统的问题域空间,就不得不编写大量的映射代码,主要面临设备类型多样性和管理服务随需性两个方面的挑战.为了能够根据管理需求快速定制和开发物联网系统,本文将运行时体系结构模型引入到无线传感网的管理过程中,提出了一种基于运行时模型的无线传感网管理方法.该方法通过构造传感设备运行时模型,实现在模型层对单一的传感设备进行管理,通过模型合并实现场景中不同的传感设备的统一管理,通过模型转换使得面向应用场景进行物联网系统的开发成为可能.

未来工作的重点主要包含两个方面:一方面,将该方法运用到实际生产环境中,运用多样化物联网监控设备实践不同的管理场景;另一方面,在该方法基础上进行管理策略的研究,基于模型分析、推理等技术,实现系统容错、可靠性保障等高级管理功能,进一步简化物联网系统的开发过程.

References:

[1] Atzori L, Iera A, Morabito G. The Internet of things: A survey. Computer Networks, 2010,54(15):2787−2805. [doi: 10.1016/j.comnet.2010.05.010]

[2] Mottola L, Picco GP. Programming wireless sensor networks: Fundamental concepts and state of the art. ACM Computing Surveys (CSUR), 2011,43(3). [doi: 10.1145/1922649.1922656]

[3] Garlan D. Software architecture: A roadmap. In: Proc. of the 22nd Int'l Conf. on Software Engineering, Future of Software Engineering Track. New York: ACM Press, 2000. 91−101.

[4] Mei H, Shen JR. Progress of research on software architecture. Ruan Jian Xue Bao/Journal of Software, 2006,17(6):1257−1275 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/1257.htm [doi: 10.1360/jos171257]

[5] France R, Rumpe B. Model-Driven development of complex software: A research roadmap. In: Proc. of the 29th Int'l Conf. on Software Engineering, Future of Software Engineering Track. Washington: IEEE Computer Society Press, 2007. 37−54. [doi: 10.1109/FOSE.2007.14]

[6] Bencomo N, Blair G, France R. Summary of the workshop Models@run.time at MoDELS 2006. In: Proc. of the LNCS Satellite Events at the MoDELS 2006 Conf. Genoa: Springer-Verlag, 2007. 227−231. [doi: 10.1007/978-3-540-69489-2_28]

[7] Blair G, Bencomo N, France R. Models@ run.time. Computer, 2009,42(10):22−27. [doi: 10.1109/MC.2009.326]

[8] Huang G, Mei H, Yang FQ. Runtime recovery and manipulation of software architecture of component-based systems. Automated Software Engineering, 2006,13(2):257−281. [doi: 10.1007/s10515-006-7738-4]

[9] Occello A, Dery-Pinna AM, Riveill M. A runtime model for monitoring software adaptation safety and its concretisation as a service. In: Proc. of the 3rd Workshop on Models@run.time. Berlin: Springer-Verlag, 2008. 67−76.

[10] Wu YH, Huang G, Song H, Zhang Y. Model driven configuration of fault tolerance solutions for component-based software system. In: Proc. of the 15th Int'l Conf. on Model Driven Engineering Languages and Systems. Innsbruck: Springer-Verlag, 2012. 514−530. [doi: 10.1007/978-3-642-33666-9_33]

[11] Rushby JM. Model checking and other ways of automating formal methods. In: Position Paper for Panel on Model Checking for Concurrent Programs, Software Quality Week. 1995.

[12] Huang G, Song H, Mei H. SM@RT: Applying architecture-based runtime management of internetware systems. Int'l Journal of Software and Informatics, 2009,3(4):439−464.

[13] Song H, Huang G, Chauvel F, Xiong YF, Hu ZJ, Sun YC, Mei H. Supporting runtime software architecture: A bidirectional-transformation-based approach. Journal of Systems and Software, 2011,84(5):711−723. [doi: 10.1016/j.jss.2010.12.009]

[14] Peking University. SM@RT: Supporting models at run-time. 2009. http://code.google.com/p/smatrt/

[15] Song H, Xiong YF, Chauvel F, Huang G, Hu ZJ, Mei H. Generating synchronization engines between running systems and their model-based views. In: Proc. of the Models in Software Engineering (the MoDELS Workshops). Denver: Springer-Verlag, 2009. 140−154. [doi: 10.1007/978-3-642-12261-3_14]

[16] Song H, Huang G, Xiong YF, Chauvel F, Sun YC, Mei H. Inferring meta-models for runtime system data from the clients of management APIs. In: Proc. of the 13rd Int'l Conf. on Model Driven Engineering Languages and Systems. Oslo: Springer-Verlag, 2010. 168−182. [doi: 10.1007/978-3-642-16129-2_13]

[17] Zhang W, Song H, Huang G. Object oriented accessing approach for wireless sensor network devices and data. Journal of Frontiers of Computer Science and Technology, 2011,5(12):1076−1084 (in Chinese with English abstract).

[18] Object Management Group. Meta object facility (MOF) 2.0 query/view/transformation (QVT). 2005. http://www.omg.org/spec/QVT

[19] Eclipse. Eclipse modeling framework. 2008. http://www.eclipse.org/modeling/emf/

[20] Gay D, Levis P, von Behren R, Welsh M, Brewer E, Culler D. The nesC language: A holistic approach to networked embedded systems. In: Proc. of the ACM SIGPLAN 2003 Conf. on Programming Language Design and Implementation. New York: ACM Press, 2003. 1−11.

[21] Costa P, Mottola L, Murphy A, Picco G. Programming wireless sensor networks with the TeenyLIME middleware. In: Proc. of the 8th Int'l Conf. on Middleware. Newport Beach: Springer-Verlag, 2007. 429−449. [doi: 10.1007/978-3-540-76778-7_22]

[22] Madden SR, Franklin MJ, Hellerstein JN, Hong W. TinyDB: An acquisitional query processing system for sensor networks. ACM Trans. on Database Systems, 2005,30(1):122−173. [doi: 10.1145/1061318.1061322]

[23] Spiess P, Karnouskos S, Guinard D, Savio D, Baecker O, Souza LMSD, Trifa V. SOA-Based integration of the Internet of things in enterprise services. In: Proc. of the IEEE Int'l Conf. on Web Services. Los Angeles: IEEE Press, 2009. 968−975. [doi: 10.1109/ICWS.2009.98]

[24] Janowicz K, Broring A, Stasch C, Schade S, Everding T, Llaves A. A restful proxy and data model for linked sensor data. Int'l Journal of Digital Earth, 2013,6(3):233−254. [doi: 10.1080/17538947.2011.614698]

[25] Beckmann K, Thoss M. A model-driven software development approach using OMG DDS for wireless sensor networks. In: Proc. of the Software Technologies for Embedded and Ubiquitous Systems. Waidhofen-Ybbs: Springer-Verlag, 2010. 95−106. [doi: 10.1007/978-3-642-16256-5_11]

[26] Losilla F, Vicente-Chicote C, Alvarez B, Iborra A, Sanchez P. Wireless sensor network application development: An architecture-centric MDE approach. In: Proc. of the Software Architecture. Aranjuez: Springer-Verlag, 2007. 179−194. [doi: 10.1007/978-3-540-75132-8_15]

[27] Akbal-Delibas B, Boonma P, Suzuki J. Extensible and precise modeling for wireless sensor networks. In: Proc. of the Information Systems: Modeling, Development, and Integration. Sydney: Springer-Verlag, 2009. 551−562.

[28] Thang NX, Geihs K. Model-Driven development with optimization of non-functional constraints in sensor network. In: Proc. of the 2010 ICSE Workshop on Software Engineering for Sensor Network Applications. New York: ACM Press, 2010. 61−65. [doi: 10.1145/1809111.1809128]

[29] Shimizu R, Tei K, Fukazawa Y, Honiden S. Model-Driven development for rapid prototyping and optimization of wireless sensor network applications. In: Proc. of the 2nd Workshop on Software Engineering for Sensor Network Applications. New York: ACM Press, 2011. 31−36. [doi: 10.1145/1988051.1988058]

[30] Rodrigues T, Dantas P, Delicato FC, Pires PF, Pirmez L, Batista T, Miceli C, Zomaya A. Model-Driven development of wireless sensor network applications. In: Proc. of IFIP 9th Int'l Conf. on Embedded and Ubiquitous Computing. Melbourne: IEEE Press, 2011. 11−18. [doi: 10.1109/EUC.2011.50]

[31] Sicard S, Boyer F, De Palma N. Using components for architecture-based management: The self-repair case. In: Proc. of the 30th Int'l Conf. on Software Engineering. New York: ACM Press, 2008. 101−110. [doi: 10.1145/1368088.1368103]

[32] Morin B, Barais O, Nain G, Jezequel JM. Taming dynamically adaptive systems using models and aspects. In: Proc. of the 31st Int'l Conf. on Software Engineering. Washington: IEEE Computer Society Press, 2009. 122−132. [doi: 10.1109/ICSE.2009.5070514]

[33] Chen XP, Huang G, Chauvel F, Sun YC, Mei H. A framework for the integration of MOF-compliant analysis methods. In: Proc. of the 2nd Asia-Pacific Symp. on Internetware. New York: ACM Press, 2010. 1−10.

[34] Li JG, Chen XP, Huang G, Mei H, Chauvel F. Selecting fault tolerant styles for third-party components with model checking support. In: Proc. of the 12th Int'l Symp. on Component-Based Software Engineering. East Stroudsburg: Springer-Verlag, 2009. 69−86. [doi: 10.1007/978-3-642-02414-6_5]

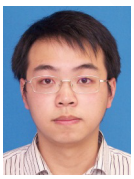
附中文参考文献:

[4] 梅宏,申峻嵘.软件体系结构研究进展.软件学报,2006,17(6):1257−1275. http://www.jos.org.cn/1000-9825/17/1257.htm [doi: 10.1360/jos171257]

[17] 张伟,宋晖,黄罡.无线传感设备及数据的对象化访问方法.计算机科学与探索,2011,5(12):1076−1084.

陈星(1985－),男,福建永春人,博士,讲师,CCF 会员,主要研究领域为分布式系统,软件中间件,软件工程.
E-mail: chenxing@fzu.edu.cn

李隘鹏(1992－),男,硕士生,主要研究领域为分布式系统,软件中间件,软件工程.
E-mail: fzuliaipeng@gmail.com



张伟(1978－),男,硕士生,主要研究领域为分布式系统,软件中间件,软件工程.
E-mail: zhangwei3549@126.com

郭文忠(1979－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为计算智能及其在计算机网络中的应用研究.
E-mail: guowenzhong@fzu.edu.cn

黄罡(1975－),男,博士,教授,博士生导师,CCF 会员,主要研究领域为分布式系统,软件中间件,软件工程.
E-mail: huanggang@sei.pku.edu.cn

陈国龙(1965－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为智能信息处理,计算机网络信息安全.
E-mail: cgl@fzu.edu.cn