

面向个人自组织网络的硬件资源管理^①

方亚芬^{1,3}, 梁冠宇², 贺也平¹

¹(中国科学院软件研究所 基础软件国家工程研究中心, 北京 100190)

²(中国科学院软件研究所 总体部, 北京 100190)

³(中国科学院大学, 北京 100190)

摘要: 随着各类智能终端的普及, 多屏互动等跨设备使用硬件资源的应用不断涌现, 对硬件资源管理的跨平台性和交互扩展性提出了更高的要求。然而, 当前针对硬件资源共享的应用大多是缺少交互性的跨设备投影镜像技术, 或者只是相互独立的简单应用, 面向的平台、硬件资源类型单一, 难以灵活有效地应对新增需求。针对这一问题, 研究了个人自组织网络环境下多设备的硬件资源(键盘、鼠标、摄像头等)的管理调配, 并提出一个具有良好跨平台性、可扩展性的应用框架, 使得设备能够像使用本机硬件一样使用其他设备上的硬件资源, 借此方便用户对其多个设备进行管理控制。该框架包括设备间通信和硬件资源管理两层, 前者提供信息传输服务, 保证实时通信, 后者负责管理可以访问到所有硬件资源, 并提供资源申请、使用和状态改变通知等功能, 极大地方便了跨设备应用的开发。

关键词: 个人自组织网络; 多设备; 硬件资源管理; 进程间通信; 应用框架

Hardware Resource Management for Personal Self Organizing Networks

FANG Ya-Fen^{1,3}, LIANG Guan-Yu², HE Ye-Ping¹

¹(National Engineering Research Center for Fundamental Software, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)

²(General Department, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)

³(University of Chinese Academy of Sciences, Beijing 100190, China)

Abstract: While the popularity of all kinds of intelligent terminals, cross-device applications, such as multi-screen, are emerging, hence, higher requirements are put forward to the cross-platform and interactive extendibility of the management of hardware resources. However, most of the current applications that share hardware resources are mirror projection technology across devices which lacks interactivity, or just simple applications independent of each other. Besides, they are targeted at single platform or hardware resource, which make it difficult to deal with new requirements flexibly and effectively. Focused on this issue, the management and allocation of hardware resources (keyboard, mouse, camera, etc.) in multiple devices in personal self-organizing network is studied, and an application framework with good cross-platform and extendibility is proposed, which could make one device use others' hardware resources just like using its own, also could help users manage and control their devices. The framework is consisted of inter process communication and hardware resource management. The former provides information transmission service to insure real-time communication, while the latter is responsible for the management of access to all hardware resources and provides the functions of applying and using resource, status change notification, etc. With the framework, the development of cross-device application becomes very convenient and simple.

Key words: personal self organizing network; multiple devices; hardware resource management; inter process communication; application framework

① 基金项目:中国科学院战略性先导科技专项(XDA06010600)

收稿时间:2016-04-03;收到修改稿时间:2016-04-21 [doi:10.15888/j.cnki.csa.005472]

传统定义的自组织网络(Ad hoc)是一种无中心、自组织、自愈的网络,是不需要依靠现有固定通信网络基础设施的、能够迅速展开使用的网络体系,各个网络节点相互协作、通过无线链路进行通信、交换信息,实现信息和服务的共享。而个人自组织网络,并不局限于无线网络,是以有线或无线方式组织的局域网,属于 P2P 网络,针对家庭或工作局域网中一个或多个用户拥有的设备,网络中的设备无需配置即可实现互联互通。

传统的资源管理一般针对互联网中聚集的计算资源、存储资源^[1]等,而个人自组织网络中管理的是键盘、鼠标、显示设备等可供利用的硬件资源,基于此也产生了很多应用。比如美国苹果公司推出的无线技术 Airplay,多屏交互应用 WiFi-Display,开源技术 Openflint,还有用于跨设备使用键盘、鼠标的 synergy 和微软的无界鼠标。其中, Airplay 使得同一局域网内苹果系列的设备(iPhone、iPad、iPod touch 等)可以将内容(照片、音频、视频流)传送到目标屏幕上,以实现同步播放音乐、视频等功能,目前已经有越来越多的第三方设备支持 Airplay 功能^[2]。与 Airplay 相似的,还有基于 WiFi 连接的多屏交互技术 WiFi-Display,它可以实现手机/移动 PC-电视/显示器的无线连接,在无线设备与高清显示屏之间实现可靠、对等的高清视频与音频流传输^[3]。支持在多设备间发现、启动以及通信的开源技术——Openflint,目前支持 Android、iOS、web、Firefox OS 的开发,开发人员可以基于该技术,充分利用多个设备的资源以实现很多有趣的多屏互动应用^[4]。可以在多台计算机之间共用键盘、鼠标的应用 synergy,支持 Linux、Windows 和 Mac OS X 系统。类似地,微软也推出了无界鼠标应用,但是只支持 Windows 系统。

跨设备使用硬件资源的应用和技术层出不穷,体现了对设备间硬件资源共享的需求。然而,类似上述实例,它们针对的应用场景相对局限,且在跨平台性、针对的硬件资源种类以及透明交互的支持性上都有一定的不足。其中, Airplay 和 WiFi-Display 分别针对苹果系列设备和 Android 设备,面向平台单一,二者主要针对声音、视频、图片等输出,互联设备之间缺少交互性。开源技术 Openflint 针对多屏互动推出了一些应用,这些应用主要针对声卡、显卡的使用,且应用互相独立,设备间交互与应用自身紧耦合,新应用的开发需

要涵盖底层数据通信和上层应用自身的所有功能。跨设备使用键盘、鼠标的应用 synergy 和微软的无界鼠标,都只是针对局域网中鼠标、键盘这两类硬件资源,且 synergy 使用前需要手动配置待交互的设备,用户体验较差,无界鼠标只是针对微软的产品,跨平台性不好。随着各类智能终端的普及,家庭或工作局域网中可供使用的硬件资源越来越丰富,硬件资源的共享不应再局限于单一平台、单一应用,上述技术多是聚焦于某一方面,针对新增的需求体现出开发不够灵活、不够高效等局限性。本文针对这一现象,研究个人自组织网络中各种终端节点的可用资源(键盘、鼠标、打印机等)的统一管理,旨在提供一个支持多种硬件资源、跨平台且方便交互和功能扩展的应用框架,为多设备环境中的整体硬件资源提供统一的管理调度平台,并为上层的多屏互动^[5-9]应用等提供底层支持。鉴于个人自组织网络自身的特点,该应用框架需要解决通信的实时性、信息一致性维护、跨平台性、安全性等方面的问题。接下来,本文将针对这些问题做详细介绍。

1 总体设计

个人自组织网络作为一个范围相对较小的对等网络,对硬件资源的统一管理主要面临有以下四个方面的挑战:

1) 网络拓扑结构动态变化

各类终端通过网络连接,网络的拓扑结构可能随时发生变化,这就需要及时发现网络中的上/下线的设备,并将更新的信息及时反馈到各个终端,保证通信的实时性。

2) 信息一致性维护

在个人自组织网络中,没有控制中心,网络中的各节点没有主要和次要之分。各个设备共享该网络中所有设备的资源信息,并根据自身硬件资源状态响应其他节点的硬件资源服务请求,这就涉及到信息一致性维护的问题。

3) 节点异构

个人自组织网络中终端设备多样,例如笔记本、智能手机、平板电脑、智能电视等,不同设备的系统类型和自身的处理能力也有很大区别,相应地,硬件资源管理需要具有较高的可扩展性、跨平台性。

4) 安全性

鉴于网络中终端设备多样,且移动终端电源有限,

无法实现复杂的加密算法. 如何在该对等网络下实现点对点的身份认证并保证安全通信, 也是亟待解决的问题.

针对以上几点问题, 本文研究并实现了一种面向个人自组织网络的硬件资源管理框架, 该框架位于网络中各个节点的操作系统内核之上, 属于应用框架层的一部分, 框架分为两部分: 设备间通信和硬件资源管理. 其总体架构如图1所示.

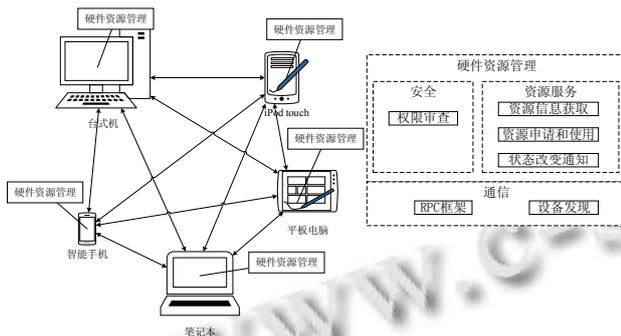


图1 系统总体框图

在该框架中, 设备间通信模块包括设备发现子模块用于发现个人自组织网络下的多个设备并提供设备IP等基本信息, 还负责不同设备组件间进行远程交互, 以及不同设备间组件的数据交换, 该模块隐藏了底层通信实现细节, 对外暴露简单的通信接口, 为硬件资源管理模块提供信息传输服务, 保证实时通信, 通过身份认证和签名的方式保证数据可信性, 通过数据加密方式保证数据机密性; 硬件资源管理模块负责管理可以访问到所有设备资源, 并提供资源申请和使用的接口, 提供设备状态改变通知等功能. 结合这两个模块, 为即时通信、多屏互动等相关应用提供底层支持. 接下来本文将对设备间通信和硬件资源管理两部分做详细介绍.

2 硬件资源管理

2.1 设备间通信

个人自组织网络中的终端设备多样, 不同设备的系统类型和自身的处理能力也有很大区别. 不同系统平台提供了不同的进程间通信方式, 如Linux下的D-Bus^[10]、Android下的Binder^[11]等, 且不同平台的数据通信在数据格式、编码方式等方面也各有特点. 智能手机、平板等移动设备多以无线的方式互相连接, 网络信号不够强且不稳定, 这就要求设备间通信能够

做到传输数据的尽量压缩, 方便解析, 以加快通信速度, 减少响应时间. 移动设备以电池这样的易耗尽能源作为电源, 其CPU性能较低, 内存较小, 存储资源有限, 难以支持复杂的加密算法, 这就要求设备间通信能够在保证通信安全的同时做到灵活且简单的数据加密.

基于上述要求, 本文将通信模块分为设备发现和通信两部分. 其中, 设备发现是设备间通信的子模块, 也是设备间通信的前提, 用于发现连入个人自组织网络中的设备, 监控主机上下线, 并提供设备的基本信息^[13,14]. 为兼顾不同平台的进程间通信, 本文设计并实现了一个统一的通信适配层, 支持跨平台访问, 隐藏通信实现细节, 为上层通信提供调用接口, 保证实时通信. 进程间通信的设计细节将在第3节详细介绍.

为压缩传输数据, 加快数据格式化和解析速度, 以减少通信占用带宽和响应时间, 本文采用google提供的ProtoBuf作为数据交换格式. ProtoBuf是一种独立于语言、平台, 可扩展的序列化结构化数据的方式, 它采用二进制数据传输格式, 压缩方便, 解析速度快, 非常适合异构环境下的数据通信^[15,16]. 当前比较流行的数据存储方式或序列化工具相比, ProtoBuf在序列化和反序列化时间以及序列化后数据压缩比率上有着明显优势, 其对比图分别见图2和图3.

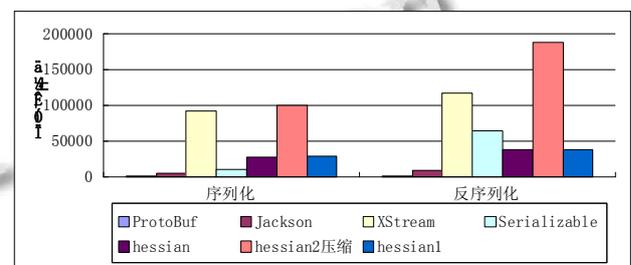


图2 序列化时间对比

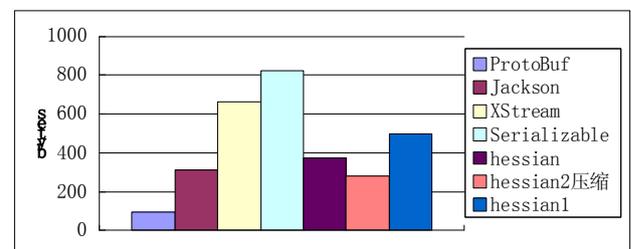


图3 序列化后数据大小对比

在通信通道建立时通过数字签名认证接口保证通信双方的可信性^[17,18]. 此外, 通信层还提供数据加密

接口,可以方便灵活修改数据加密方式以保证通信的机密性。

2.2 资源管理

硬件资源管理模块,以守护进程的形式运行在每一台设备上,提供硬件资源信息查询、硬件资源申请和使用、资源状态改变通知的功能。

2.2.1 资源描述

通过对个人自组织网络中的各类设备涵盖的硬件资源进行概括抽象,将设备的硬件资源分为输入设备、输出设备、硬盘、设备连接的打印机。图4展示了硬件资源管理模块管理的资源分类情况。

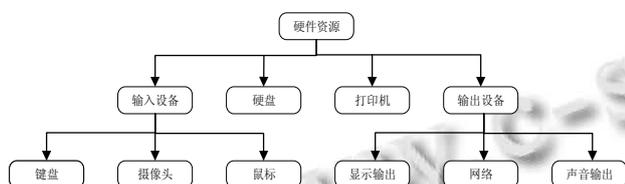


图4 系统总体框图

本文提出并制定了一种特有的硬件资源描述方法,该方法通过递归定义,统一各层硬件资源的描述格式,方便硬件资源的扩展和解析。

各层的硬件资源的描述信息都包括资源的静态描述信息和资源的详细信息 detail,图4中最下层的硬件资源(键盘、硬盘、打印机、显示输出等)的描述信息还包括标识该资源是否可用的状态信息 state。静态描述信息包括资源类型 type(该设备中资源类型的唯一标识)、资源名称 name、和资源描述附加信息 desc。硬件资源的详细信息 detail 记录该类资源的信息详情。资源状态信息 state 根据该资源是否被某一设备占用而动态变化,并以此来限制其他设备对该资源的使用。以图3为例,该设备的硬件资源(type)hardResource 的资源名称(name)为 hard Resource,资源附加描述信息(desc)为 hard resource info,资源的详细信息 detail 包含了 hardResource 类别下的硬件资源信息,即图5中打印机信息。打印机这一硬件资源的描述信息同样包括静态描述信息,即类别(type)printer、名称(name)printer info、附加信息(desc)printer of the device,还包括资源详细信息(detail)。打印机的资源状态信息(state)为0表示该打印机未被占用。其信息详情 detail 以数组的形式记录了该设备所连接的打印机详细信息,图5表明该设备仅连接了一台打印机,具体信息包括打印机的

名称 name、是否为该设备的默认打印机 isDefault、打印机是否正在使用 status、打印机的 uri 即 device-uri、打印机状态更新时间 printer-state-change-time。

硬件资源管理模块中待管理的资源以树形结构存储于配置文件中,扩展方便。各层硬件资源采用一致的描述格式,方便了对不同层级的硬件资源信息进行统一或递归式地解析。

```

1 {
2   type: 'hardResource',
3   name: 'hard Resource',
4   detail: {
5     printer: {
6       type: 'printer',
7       name: 'printer info',
8       desc: 'printer of the device',
9       detail: [[
10        name: 'Generic-PCL-6-PCL-XL-LF',
11        isDefault: true,
12        status: 'IDLE',
13        device-uri: 'smb://192.168.161.66/HP...',
14        printer-state-change-time: 'Fri Jan 30 2015 14:44:29 ...'
15      ]],
16      state: 0
17    }
18  },
19  desc: 'hard resource info'
20 }
  
```

图5 打印机的描述信息

2.2.2 资源信息获取

硬件资源管理模块启动时收集本机的硬件资源信息,并生成准确的描述信息(如屏幕尺寸、分辨率等等),初始化共享信息列表。

设备加入个人自组织网络的大致流程如图6所示,当一台台式机设备上线(加入设备集群,图6的①)时,组播自身的设备信息通知所有可达的设备,进行共享信息列表同步,集群中其他设备(智能手机、笔记本电脑等)的硬件资源信息列表也得到更新(图6的②);该设备在线期间,对应设备的硬件资源状态有所变化时,会向网络中其他设备发送该变化信息,对应设备的应用进程会收到相应的事件通知(图6的③)。该设备下线(离开集群)时,集群中的其他设备会收到其下线通知,更新自身的设备列表和硬件资源信息列表(图6的④)。在个人自组织网络中,各设备皆维护一个全局资源共享列表,这就涉及到分布式环境下的数据一致性问题。本文参考经典的 Paxos^[19]算法,针对硬件资源管理下的场景需求进行了必要的优化,最终解决数据一致性问题,这部分将在第3节详细介绍。

2.2.3 资源申请和使用

资源申请和使用的一般流程为:应用进程首先申请使用某个设备资源(非本机),通过权限审核等相关操作后,完成数据流通道的建立并返回相应的数据流对象,之后应用进程通过数据交互使用硬件设备资源。

以视频输出设备为例, 整体过程如图 7 所示, 具体步骤如下:

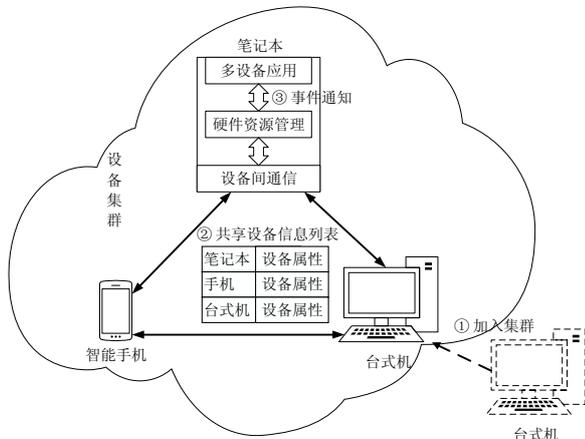


图 6 设备加入集群流程图

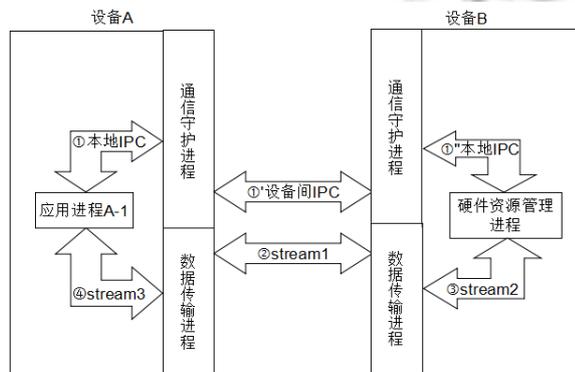


图 7 设备加入集群流程图

1) 设备 A 的应用进程 A-1 申请使用设备 B 的屏幕作为输出设备, 通过本地进程间通信(以下简称 IPC)经通信守护进程(Communication-Daemon, 以下简称为 CD)提交申请请求, 若设备 B 对应显示器资源空闲且进程 A-1 满足申请条件(权限等), 则设备 B 返回同意申请响应, 则依次完成步骤 2)3)4)5), 并基于改进的 Paxos 算法同步集群资源状态, 否则则返回拒绝申请的响应;

2) 设备 A 的硬件资源管理进程通过数据传输进程(DataTransfer, 以下简称为 DT)完成设备间的数据流(stream1)通道建立;

3) 对等节点设备 B 的硬件资源管理进程收到设备 A 应用程序的申请请求, 建立 DT 进程到视频输出进程的数据流(stream2)通道, 并连接 stream1 和 stream2;

4) 设备 A 建立应用进程 A-1 和 DT 的数据流

(stream3)通道, 并连接 stream1 和 stream3;

5) 设备 A 返回 stream3 对象给应用进程 A-1, 应用进程将产生的视频数据输入 stream3, 即可由另一台设备的视频输出进程处理, 并输出至屏幕。

3 关键实现技术

3.1 通信设计

通信模块的设计需要兼顾不同平台的适用性, 并提供一定的安全机制以确保通信的安全性。

常用的几种进程间通信方式有管道、消息队列、信号量、共享内存、套接字等, 不同平台也有不同的实现方式. Linux 下的 D-Bus 作为应用程序间通信的消息总线系统, 通过提供事务处理机制, 允许程序在其上注册并为其他程序提供服务^[10]. Android 下的 IPC 基于 Binder, Binder 是一种 C/S 的架构, 客户端可以通过同一台设备的本地进程间通信的方式, 也可以通过多个设备之间的套接字连接, 获得服务的代理接口, 直接调用服务提供的方法^[11,12]. 本文设计并实现了一个统一的通信适配层, 兼容 Linux、Android 等多个系统平台, 支持跨平台访问, 隐藏通信实现细节, 为上层通信提供调用接口, 保证实时通信。

本文设计并实现的设备间通信如图 8 所示, 主要由三部分组成: 基于代理-存根(Proxy-Stub)模式的进程通信框架, RPC 通信守护进程, 数据交换守护进程。

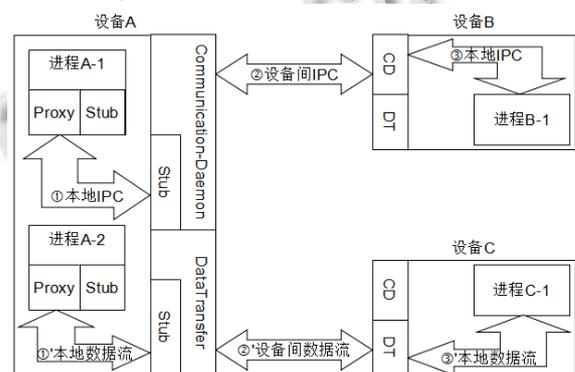


图 8 设备加入集群流程图

1) 基于代理-存根(Proxy-Stub)模式的进程通信框架, 将自己的对外接口定义在存根模块中, 同时定义对应的代理模块. 当该进程启动后, 框架机制会确保将存根的接口发布在当前运行平台的 IPC 组件中, 其他进程可以通过框架提供方法获取目标进程的代理模块, 或者使用当前平台本身提供的方法获得目标进

程的接口. 因此, 基于此框架实现的进程完全兼容其他一般进程, 框架只是隐藏了底层复杂的机制, 通过接口调用即可实现 IPC. 图 8 中进程 A-1、进程 A-2、进程 B-1、进程 C-1 等均是这种进程.

2) RPC(远程过程调用)通信守护进程, 即图 8 中的 Communication-Daemon(简称 CD), 是基于上述框架实现的进程, 在每台设备上唯一的运行着该进程, 设备间通过该守护进程实现对等通信, 需要跨设备通信的进程仅仅需要通过 CD 的代理模块调用相应的接口即可. 这样实现有以下好处:

- ① 限制唯一的 RPC 进出口, 方便实现安全验证机制;
- ② 每两台设备间仅仅维护一条通信链路, 降低开销;
- ③ 每个进程不必单独实现连接其他设备的功能, 实现了功能复用.

3) 数据交换守护进程, 即图 8 中的 DataTransfer(简称 DT), 同样基于上述框架实现, 在每台设备上唯一运行着一个进程, 通过该进程可以实现设备间点对点的数据交换, 并以提供数字签名的方式保证传输数据的可信性. 客户进程(如进程 A-2)通过 DT 的接口建立并获得一个数据流对象, 之后对数据流对象进行操作实现设备间进程的数据交换. 考虑到这两种数据连接不同的性质, 适用不同的应用场景, CD 的消息通常比较简短同时需要较快的响应速度, 而 DT 通常会长时间占用连接通道, 因而将 DT 从 CD 中分离.

3.2 资源状态一致性维护

在个人自组织网络中, 各个设备维护该网络中所有设备的资源列表, 并依据本机硬件资源状态响应资源申请请求. 本文通过对 Paxos 算法进行优化, 以保证多设备中硬件资源信息一致性, 提供资源服务.

Paxos 算法是 Lamport 于 1990 年提出的一种基于消息传递的一致性算法, 旨在解决一个分布式系统如何就某个值(决议)达成一致的问题^[19,20]. 基于 Paxos 算法, 本文对资源状态一致性维护有两大原则:

- 1) 安全原则, 即不能做错的事. 只能有一个值被批准, 不能出现第二个值把第一个覆盖的情况; 每个节点只能学习到已经被批准的值, 不能学习没有被批准的值. Paxos 算法中通过为分布式环境中的节点提交的请求赋予一个 ID, 并根据 ID 排序, 批准优先级高的

值, 拒绝排序在后的, 进而保证安全一致性.

本文基于请求到达时间确定优先级, 本着先来先服务的原则提供硬件资源服务. 应用进程申请使用某个设备(非本机)硬件资源时, 设备硬件资源管理模块首先检查该设备及应用的权限, 若满足申请要求, 则将申请请求加入硬件资源请求队列, 由硬件资源管理模块逐一处理请求, 被请求的资源空闲则返回同意申请的响应, 并通知所有在线设备该资源状态改变为被占用. 得到申请通过响应的应用会与对应设备间完成数据流通道的建立, 得到相应的数据流对象, 进行资源使用, 该过程如图 9 所示.

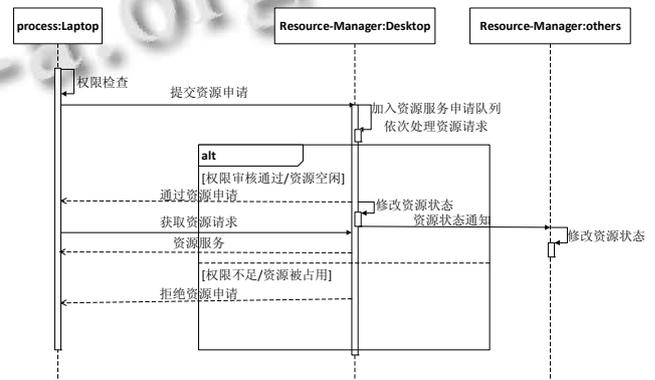


图 9 硬件资源申请和使用流程图

- 2) 存活原则, 只要有多数服务器存活并且彼此间可以通信, 终会批准某个被提议的值, 而且其他服务器最终会学习到这个值.

本文目前研究的内容中, 硬件资源服务请求者主动选择对应设备提供硬件资源服务, 而服务提供者设备作为唯一决策者决定是否提供硬件资源服务, 硬件资源状态改变后以消息通知的形式通知到个人自组织网络中互联互通的设备.

在未来研究中, 可以根据网络中各个节点的性能选举出一个或多个中心节点, 采用中心节点辅助控制与对等网络相结合的方式, 由中心节点依据应用需要以及各个节点上硬件资源的特点自主决定哪些设备更适合提供服务. 此时, 选择提供服务的设备所需要的决策者不止一个, 根据安全和存活原则, 增加服务的自主选择能力.

4 实验

本文的实验环境是 2 台台式机和 1 台智能手机, 其环境配置如下.

台式机: 4GB 内存、100GB 以上硬盘存储、4 核心处理器 (型号: Intel(R) Core(TM) i5-2400 CPU @3.10GHz), Linux Mint 17 32-bit 的操作系统, 还需安装 NodeJS(0.10.29)和 python(2.7.6)、Java 1.7.0_65、android-21 开发环境。

移动端: 型号 Nexus 6, 3GB 内存, 26GB 存储空间, 四核心处理器, Android 5.0 系统环境。

硬件资源管理系统基于 NodeJS 实现, NodeJS 是一个 Javascript 运行环境(runtime), 支持跨平台开发, 使用事件驱动、非阻塞 I/O 模型而得以轻量 and 高效, 非常适合在分布式设备上运行的数据密集型的实时应用。界面实现使用 html5、css3、JS 等技术, 具有良好的跨平台性, 支持跨设备浏览器访问。本文以鼠标和键盘为例, 采用 python 提供的第三方库 virtkey、Xlib 等实现对鼠标和键盘事件的捕获和模拟, 由 NodeJS 实现对鼠标、键盘资源请求的控制和事件的传输, 并根据解析得到的事件信息调用 python 接口模拟鼠标、键盘事件, 实现了跨设备使用鼠标、键盘。

4.1 功能

本文实现的硬件资源管理系统根据不同平台进行通信层的适配, 适用于 Linux、Andriod 等多个系统, 具有良好的跨平台型, 硬件资源类型和硬件资源服务都具有良好的可扩展性, 进程间通信模块提供数据加密接口以保证数据机密性, 硬件资源管理模块数据流传输中加入签名信息以保证数据可信性。实验中直接调用硬件资源管理模块提供的接口来实现上层应用, 极大地节省了开发时间。

在资源列表展示上, 本文实验提供一个可视界面用于显示个人自组织网络环境中各台主机设备的硬件资源, 实时监控到设备上/下线情况, 及时更新硬件资源状态信息, 详情如图 10 所示。



图 10 硬件资源信息展示

在资源使用上, 实验效果示意图如图 11 所示。



图 11 跨设备使用键盘鼠标示意图

4.1 性能

本文实现的硬件资源管理系统通过采用 ProtoBuf 的二进制数据编码机制, 数据压缩比例约为 64.37%, 减少了网络带宽的占用。本地资源管理数据的响应时间在 15ms 至 22ms 之间, 平均响应时间约 17.4ms, 个人自组织网络中跨设备硬件资源数据请求的响应时间在 28ms 至 45ms 之间, 平均响应时间约为 34.46ms, 满足实时通信对用户体验的要求(根据 2-5-10 原则, 2s 以内得到响应, 用户体验较好)。

与现在市场流行的硬件资源方面的技术或应用 (AirPlay、Openflint、Synergy、无界鼠标等) 相比, 本文研究的硬件资源管理系统 (Resource-Manager) 优劣对比见表 1 和表 2。

表 1 支持平台对比

平台 应用/技术	Windows	Linux	IOS /OSX	Android	其他平台
AirPlay	×	×	√	×	无
Openflint	×	×	√	√	Web、 FireFox OS
Synergy	√	√	×	×	无
无界鼠标	√	×	×	×	无
Resource-Manager	√	√	√	√	Web

表 2 支持硬件资源种类对比

硬件 类型 应用 /技术	显卡、 声卡	键盘、 鼠标	存储	网卡	打印机	备注
AirPlay	√	×	×	×	×	无
Openflint	√	×	×	×	×	无
Synergy/ 无界鼠标	×	√	×	×	×	无
Resource-Manager	√	√	√	√	√	范围自定义、 可扩展

由两张表可以看出, 本文研究的硬件资源管理应用框架可以支持多种硬件资源, 在跨平台性上也有较大优势。除此之外, 该框架还包括一个统一的通信适

配层,方便跨平台交互访问,隐藏通信实现细节,为上层各类应用的通信提供调用接口,保证实时交互。最终,本文提供了一个硬件视图显示层以提供局域网中的所管理的硬件信息的可视窗口,以及硬件资源使用接口以方便对多个设备的硬件资源进行统一管理调配。

鉴于研究内容只针对个人自组织网路环境,因此管理服务目前只面向局域网。通过利用该框架,开发者很方便地获得设备的硬件资源信息,实现跨设备硬件资源共享,加快了多屏互动等应用的开发速度。

5 总结

当前智能终端设备不断丰富,局域网下可供利用的硬件资源越来越多,由此也衍生出很多共享设备间硬件资源的应用。但是目前存在的大多是缺少交互性的跨设备投影镜像技术,或者只是相互独立的简单应用,且其适用的平台单一有限,难以灵活适应新增需求。本文针对这一现象,通过对个人自组织网络环境下的多设备的硬件资源管理进行研究,设计并实现了一个支持多种硬件资源、跨平台且方便交互和功能扩展的应用框架。该框架通过设备发现监控网络拓扑结构的动态变化,并结合涵盖认证和加密机制的设备间通信实现安全的实时交互、维护信息一致性。此外,框架提供一个统一的适配层,透明化实现细节,隐藏节点异构性,为上层多屏互动等相关应用提供接口,极大地简化了应用开发过程。

本文目前只针对硬件资源的统一管理使用问题,在未来的研究中,除了优化硬件资源使用效果之外,会逐步实现应用自主选择合适的运行设备以借用对应设备硬件资源,并加入对计算资源、数据资源的统一管理,实现个人自组织网络环境下各方面资源都做到物尽其用,提升用户体验。

参考文献

- 1 Vavilapalli VK, Murthy AC, Douglas C, et al. Apache hadoop yarn: Yet another resource negotiator. Proc. of the 4th Annual Symposium on Cloud Computing. ACM. 2013. 5.
- 2 邱晨.基于 Android 的 Airplay Server 端应用设计与实现[硕士学位论文].厦门:厦门大学,2014.
- 3 陈子安.基于 Android 平台的 WiFi-display sink 端的设计与实现.软件导刊,2013,(9):104-106.
- 4 颜万隆,陈月峰.基于 DLNA 协议的多屏互动系统.广东海洋大学学报,2013,33(4):83-86.
- 5 余泽喜,王香婷,马婷婷.多屏互动系统的研究与实现.智能计算机与应用,2012,2(1):69-70.
- 6 吴铁群,朱亚东,王明敏.基于 Android 平台的多屏互动系统设计.计算机应用与软件,2014,31(10):234-238.
- 7 Yang J, Wigdor D. Panelrama: Enabling easy specification of cross-device web applications. Proc. of the SIGCHI Conference on Human Factors in Computing Systems. ACM. 2014. 2783-2792.
- 8 Pouwelse J, Garbacki P, Epema D, et al. The bittorrent p2p file-sharing system: Measurements and analysis. Peer-to-Peer Systems IV. Springer Berlin Heidelberg, 2005: 205-216.
- 9 Shen Z, Tan S, Li Y. Study of multi-screen interaction on smart TV based on UPnP protocol. Journal of Convergence Information Technology, 2013, 8(10): 448.
- 10 王昭顺,李涛.基于 D-BUS 的软件组件开发技术的研究.2009 全国虚拟仪器大会论文集,2009.
- 11 段义鹏.Android 进程通信机制研究与应用[硕士学位论文].武汉:武汉理工大学,2012.
- 12 苑冰泉.Android 进程间通信机制 Binder 的分析与对比研究[硕士学位论文].南京:东南大学,2014.
- 13 冀博,李宵谋,郭猛.网络设备自动发现技术的研究与实现.西安邮电学院学报,2013,17(6):33-36.
- 14 周亦敏,沈嘉杰.Zeroconf 网络的安全性分析及改进.计算机工程与应用,2013,49(2).
- 15 李纪欣,王康,周立发,等.Google Protobuf 在 Linux Socket 通讯中的应用.电脑开发与应用,2013,26(4):1-5.
- 16 聂晓旭,于凤芹,钦道理.基于 Protobuf 的数据传输协议.计算机系统应用,2015,24(8):112-116.
- 17 王尚平,王育民,张亚玲,等.基于 DSA 及 RSA 的证实数字签名方案.软件学报,2003,14(3):588-593.
- 18 Martucci LA, Carvalho T, Ruggiero WV. A lightweight distributed group authentication mechanism. INC2004-Fourth International Network Conference. 2004. 393-400.
- 19 许子灿,吴荣泉.基于消息传递的 Paxos 算法研究.计算机工程,2011,37(21).
- 20 高石玉,艾中良,刘忠麟.应用 Paxos 算法构建自组织网络.计算机工程与应用,2014,50(6).