从传感网到物联网：微型操作系统的现状与未来

董 玮 陈 纯 卜佳俊
浙江大学

关键词：节点操作系统

引言

自20世纪50年代起，操作系统逐渐成为计算机系统中的核心部分。早期的操作系统提供了硬件管理、批处理、多任务等基本功能，为计算机程序提供了有效的执行环境。20世纪80、90年代，操作系统的概念变得家喻户晓。分离地址空间、动态链接与加载、分时多任务调度、资源管理、图形用户界面、网络等复杂功能，成为现代计算机系统中必不可少的组成部分。

随着对未知领域与空间的拓展，人们对信息的获取方式提出了更高的要求。20世纪90年代，在美国国防部的资助下，美国加州大学洛杉矶分校、伯克利分校等最早开始了无线传感节点系统方面的研究与开发。这种智能化和网络化的节点可采集多种感知信息，并能进行智能处理，通过网络实现信息快速共享，在环境监测、目标跟踪、交通导航等领域有着广泛的应用前景。无线传感网络因此成为国内外新的研究热点。

无线传感节点是否需要操作系统？节点操作系统应该提供怎样的功能？这是无线传感网络研究的核心问题。这个问题至今都没有明确的答案。但有一点是肯定的，就是节点操作系统是极其有用的，节点操作系统提供有效的编程模型，能极大简化上层应用程序的编写。谷歌在2004年提出的MapReduce编程模型就是一例。尽管有研究者说关系数据库早在数年前就实现了类似的功能，而且处理得更加完美[1]，但MapReduce仍然在目前的数据中心、云计算环境中得到广泛使用。这是因为MapReduce方便易用，极大地提高了分布式开发的编程效率。

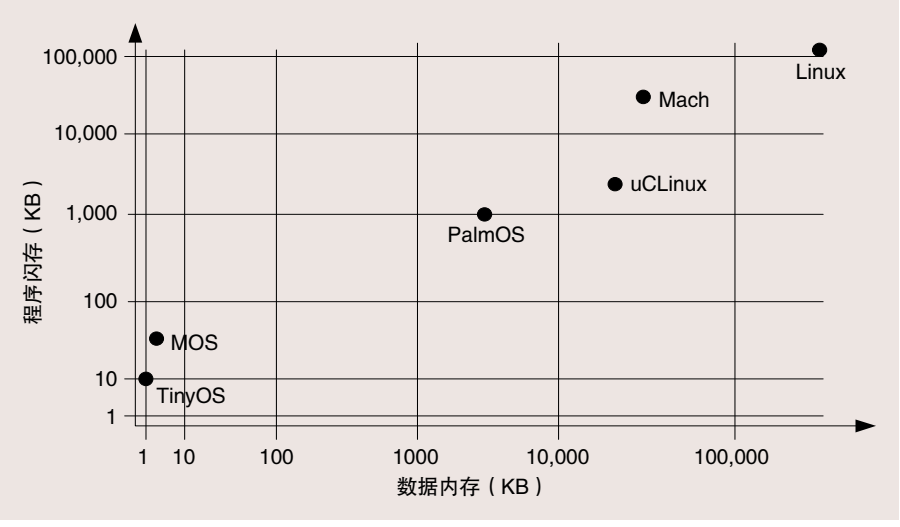

图1 节点微型操作系统和其他操作系统对比

节点操作系统能否直接借鉴传统操作系统？节点操作系统与传统操作系统存在哪些本质区别？自无线传感系

统研究之初，研究者就面临一些独特的挑战，如：节点硬件数量、种类繁多（比如传感器、微控制器、通信芯片等），节点资源受限，节点通信不可靠等等。加州伯克利分校的戴维·库勒（David Culler）团队为此开发了TinyOS，试图简化上述问题，同时使上层应用程序的编写更加方便、更具移植性。图1给出了节点操作系统TinyOS、Mantis OS与其他传统操作系统的对比。其中，横轴表示操作系统（Operating System，OS）占用的RAM（数据）的大小，纵轴表示操作系统占用的ROM（程序）大小。由此可知，节点操作系统是微型化的，既有别于如Linux、Mach那样的传统操作系统，也有别于如PalmOS、uCLinux那样的传统嵌入式操作系统。

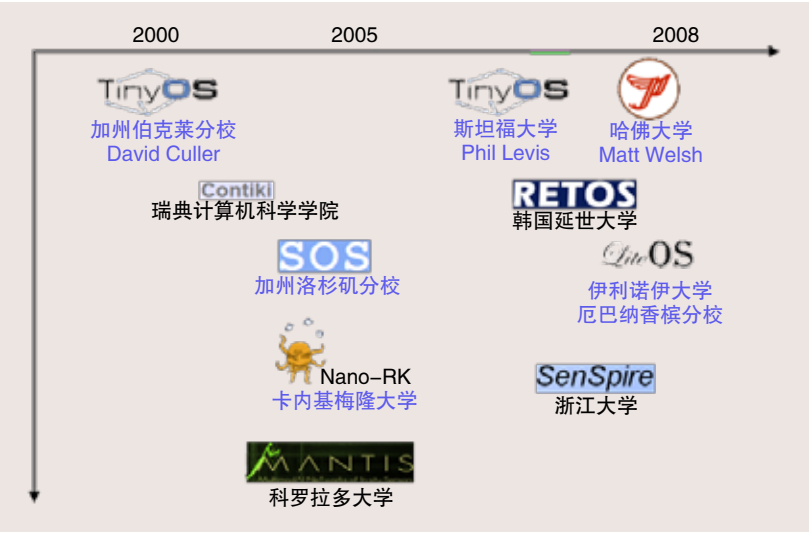

图2 节点操作系统研究团队与成果

回顾节点操作系统发展的历史，可以发现：

1. 从传感网研究角度看，节点操作系统是重要的方向之一。据Google Scholar 2010年11月23日的数据显示，对传感网络经典文献“Wireless sensor networks: a survey”[2]的引用次数达5824次，而对节点操作系统经典文献“System architecture directions for networked sensors”[3]的引用次数达3029次。可见节点操作系统在传感网研究中所占分量之重。

2. 从传感网应用角度看，节点操作系统是多个实际应用中的关键组成部分（见表1）。目前著名的SensorScope、Trio、VigilNet、FireWxNet等传感网络应用中，开发人员均使用了节点操作系统（如TinyOS、Mantis OS）来简化上层应用程序设计与开发。

表1 已有的传感网络应用及其节点微型操作系统

系统名称	研制单位	节点数	OS
SensorScope	瑞士洛桑联邦理工学院（cole Polytechnique Fédérale de Lausanne，EPFL）	97	TinyOS
Trio	美国加州大学伯克利分校	557	TinyOS
VigilNet	美国弗吉尼亚大学	200	TinyOS
FireWxNet	美国科罗拉多大学	13	Mantis OS

目前国际上已有多个团队在研究节点操作系统，并就此发表了相关论文，如图2所示。从此可知：

1. 从时间上看，节点操作系统研究大致分两个阶段。第一阶段是2005年（在TinyOS 1.x发布之后），出现了SOS、Mantis OS、Nano-RK等；第二阶段从2007到2008年（在TinyOS 2.x发布之后），出现了Pixie OS、LiteOS、RETOS、SenSpire OS等[4]。

2. 从研发团队看，包括了很多国际知名的高校，如美国的加州大学伯克利分校、斯坦福大学、哈佛大学、卡内基梅隆大学等，在系统研发方面有多年的积累。正如2009年ACM的操作系统原理会议（Symposium on Operating Systems Principles，SOSP）程序委员会主席所说的：“系统的研发是长时间的，作者介绍的项目通常需要4个人1年的工作量，代表了耗时的、高难度的工作”。

传感网及其微型操作系统

TinyOS

TinyOS是最早的节点微型操作系统，由美国加州大学伯克利分校开发。TinyOS编程基于组件（Component），组件之间可以灵活组合，与应用程序静态

编译成单个程序映像。与对象相同的是，组件可以包含代码和数据；与对象不同的是，组件只能在编译期实例化。TinyOS组件含有接口（Interface），接口内可以定义命令（Command）和事件（Event）。TinyOS核心是事件驱动的。其执行模型由中断和任务构成。中断执行的优先级高于任务，可以抢占任务的执行。任务间以先进先出的方式调度。TinyOS任务的执行方式是“Run-to-completion（执行直至完成）”，即不能自挂起，也不能被其他任务抢占。因此，I/O（读写）操作需要分阶段完成，即首先发起一个I/O请求，当I/O完成时，系统再触发相应的回调函数。TinyOS任务和中断并不维护各自的栈帧，而是共享一个系统堆栈。为了更好地支持TinyOS组件化的设计和执行模型，TinyOS使用专门为之设计的nesC（networked embedded system C）语言编程。

Contiki OS

TinyOS的组件是静态编译与链接的。这种方法虽然能优化资源的使用，但是对代码的网络重编程增加了额外的开销。这是因为，在这种模式下整个代码映像都要分发到网络中的每个节点，从而增加了传输开销和能量开销。为了解决该问题，瑞典计算机科学研究院（Swedish Institute of Computer Science）开发了Contiki OS，支持模块的动态加载。Contiki OS也借鉴了传统的协程（Coroutine）的思想，实现了另一个轻量级的线程编程模式，即Protothreads。

SOS

由美国加州大学洛杉矶分校开发的SOS也同样支持模块的动态加载。SOS的设计采用模块化的体系架构。SOS的模块和TinyOS的组件是两个不同的概念。TinyOS的组件只对编程者可见，被编译成二进制代码之后，组件就不存在了。然而，SOS模块在编译以后仍然存在，每个SOS模块都会维护额外的二进制信息（如入口函数地址、导出函数表等），便于进行动态链接与加载。为了支持这样的体系架构，SOS也支持动态内存分配。SOS的执行模型比TinyOS略微复杂。SOS的消息处理函数（类似于TinyOS任务）根据三个优先级调度，但是SOS同样不支持在消息处理函数间的抢占。

Mantis OS

为了全面支持多线程，美国科罗拉多大学开发的Mantis OS在资源受限的传感节点上实现了和传统OS类似的分时抢占式调度方式。Mantis OS支持传统的基于线程的编程模型，Mantis OS内核支持同步的I/O操作。此外，Mantis OS内核还支持常用的并发控制原语，如互斥量（Mutex）和信号量（Semaphore）。

Nano-RK

为了更好地支持时间敏感的传感网络应用程序，美国卡内基梅隆大学开发了Nano-RK。Nano-RK实现了一个基于预留（Reservation-based）的实时操作系统。Nano-RK实现了基于固定优先级的抢占多线程调度机制，以保证任务的实时性需求。同时，Nano-RK支持CPU和网络带宽预留，即任务可以显式指定它的资源需求，由操作系统提供及时、可控、有保障的访问权限。

CCF理事会换届准备工作将启动

CCF将于2012年4月进行理事会换届工作。为了做好此项工作，经CCF九届六次常务理事会议同意，CCF理事会换届准备工作即将启动。此次换届工作将由CCF理事长李国杰总体负责，由CCF秘书长杜子德任规章修订小组组长，CCF副理事长李晓明任理事会架构研究小组组长，CCF常务理事唐卫清任会员层次、种类和会费标准研究小组组长。（朱）

SenSpire OS

SenSpire OS是浙江大学承担的973基础研究计划“无线传感网络的基础理论及关键技术研究”成果，于2007年12月30日发布了1.0版（实现架构如图3）。SenSpire OS v1.0能够提供事件驱动和线程的混合编程模型、提供优先级抢占调度。

从2008年开始，SenSpire OS进行了重构和改进，并发布了SenSpire OS 1.2版。相比1.0版，SenSpire OS v1.2增加了平台支持，优化了资源开销，提供了更好的重编程机制。SenSpire OS的发布网址为：http://eagle.zju.edu.cn/home/eos/senspire。

SenSpire OS的主要特点[5]包括：采用了两阶段中断处理、基于高限协议的同步原语，以提高系统的可预测性；采用综合了事件驱动与多线程编程的混合调度模型，以提高系统的灵活性；采用了区分式内核调度、堆栈共享技术和模块化架构，以提高系统的高效性。

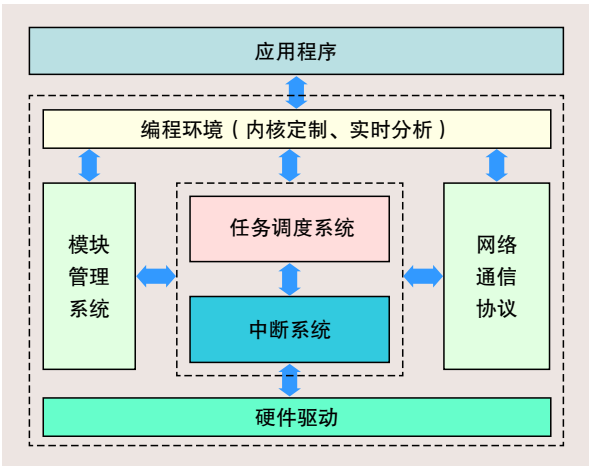

图3 SenSpire OS实现架构

物联网及其微型操作系统

近几年，随着CPS（Cyber Physical Systems）[1]、“智慧地球”等概念的推出，物联网逐渐成为新的研究热点。尽管到目前为止，物联网还没有精准的定义，但它在以下几点区别于传感网：首先，物联网实现了更透彻的感知。物联网不仅包含了普通的传感器节点，还包括了无线射频识别以及其他智能设备；其次，物联网可实现更广泛的互联，使得大量异构的感知设备都能上网，实现信息的充分共享与交换。

我们认为节点微型操作系统也是物联网体系的必要组成部分。首先、物联网中的智能设备需要统一的编程模型及系统调用来简化上层应用程序的编写。而编程模型、系统调用等正是操作系统提供的基础服务。其次，面对物联网中如此多异构的设备，迫切需要一个统一的网络层架构来实现互联。

在对国内外现状进行调研以及实践SenSpire OS的过程中，我们发现未来微型操作系统中有如下挑战问题和核心研究内容。

编程模型 一个好的编程模型需要一个方便的编程环境和一个高效的实现机制，个人计算机可以提供丰富的资源作为支持，而传感节点、智能设备目前还不能提供。但在传感网中，从节点的编程模型看，发展趋势是越来越方便，越来越统一。从TinyOS和SOS的事件驱动编程模型，到Mantis OS的多线程编程模型，再到SenSpire OS的混合编程模型，什么样的I/O编程模型才能适合物联网中设备更加多样的需求，什么样的网络化编程模型才能适合物联网中更加广泛的需求，将是值得研究的新问题。

软件维护机制 软件更新是计算机系统的必要功能之一。在操作系统层如何维护系统软件与应用软件？在传感网中，研究者提出网络重编程技术来实现软件功能的更新。然而，即使是传感网，目前也仍没有统一的重编程架构。物联网中如此多异构共存的智能设备将对操作系统的软件维护架构提出更严峻的挑战。

网络架构 目前，传感网基本采用独立的网络层协议实现互联。传感数据通过汇聚节点与互联网进行共享与交换。网络层应用程序编程接口

[1] 信息-物理融合系统，亦有译作“信息物理系统”、“网宇实体系统”、“赛博-实物系统”等。

（API）是操作系统需要提供的基础服务之一。与个人计算机不同，节点操作系统一般很少提供类似插口（Socket）的端到端通信接口。戴维·库勒指出，统一的传感网网络接口位于网络层之下、链路层之上的2.5层。自2005年提出的SP（Sensor-net Protocol，传感网协议）架构[6]以来，不少传感节点操作系统实现了类似的2.5层接口。物联网无疑将对网络层接口提出新的挑战。物联网中的智能设备需要与互联网无缝连接，因此IP（互联网协议）可能将成为标准化的网络接口。国际标准化组织也正在积极推动IPv6在节点智能设备上的标准化，IETF 6LoWPAN[2]将被集成在最新版本的TinyOS和Contiki OS中。

结语

我国一向重视计算机基础软件方面的研发，在国家973项目“无线传感网络的基础理论及关键技术研究”中，也设置了传感网节点操作系统的专门课题。

就传感网络本身而言，随着传感器网络应用的拓展，已有的微型操作系统架构是否符合真实的大规模应用的需求？在物联网中，微型操作系统又有哪些挑战性问题？物联网更透彻地感知、更广泛地互联又对微型操作系统提出怎样的设计需求？在今后很长一段时间内，这些都有待人们进一步思索和研究。■

董　玮

CCF学生会员。浙江大学计算机学院博士生。主要研究方向为传感器网络与嵌入式系统。
dongw@zju.edu.cn

陈　纯

CCF高级会员。浙江大学计算机学院教授。主要研究方向为嵌入式系统、计算机图形图像处理、计算机视觉、CAD/CAM等。chenc@zju.edu.cn

卜佳俊

CCF高级会员。浙江大学计算机学院教授。主要研究方向有嵌入式软件与系统、移动多媒体技术、信息检索与挖掘等。bjj@zju.edu.cn

参考文献

[1] Michael Stonebraker, Daniel J. Abadi, David. J. DeWitt, Samuel Madden, Erik Paulson, Andrew Pavlo, and Alexander Rasin, MapReduce and Parallel DBMSs: Friends or Foes? Communications of the ACM, 53(1), 2010

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, Wireless sensor networks: a survey, Computer Networks 38 (4) (2002) 393~422

[3] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister, “System architecture directions for networked sensors,” In Proc. of ASPLOS ,93~104, 2000

[4] Wei Dong, Chun Chen, Xue Liu, and Jiajun Bu, "Providing OS Support for Wireless Sensor Networks: Challenges and Approaches", IEEE Communications Surveys & Tutorials, vol. 12, no. 4, pp. 519~530, Fourth Quarter 2010

[5] Wei Dong, Chun Chen, Xue Liu, Yunhao Liu, Jiajun Bu, and Kougen Zheng, "SenSpire OS: A Predictable, Flexible, and Efficient Operating System for Wireless Sensor Networks", IEEE Transactions on Computers, (to appear)

[6] Joseph Polastre, Jonathan Hui, Philip Levis, Jerry Zhao, David E. Culler, Scott Shenker, Ion Stoica: A unifying link abstraction for wireless sensor networks. In Proc. of ACM SenSys 2005: 76~89

[2] Internet Engineering Task Force IPv6 over Low power Wireless Personal Area Network，互联网工程任务组基于Ipv6通信低功耗的无线个人局域网