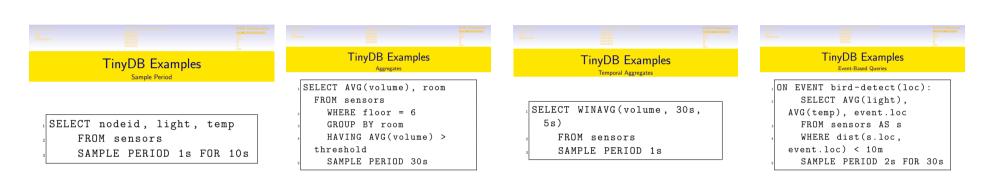
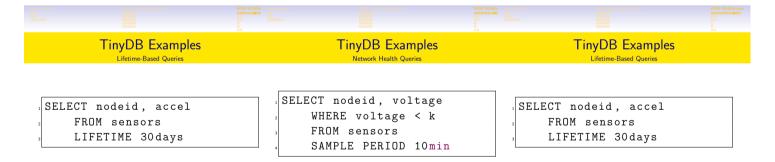
WSN Operating Systems								
	Characteristic	Architecture	Programming Model	Scheduling	Communication Protocol Support			
TinyOS	 Concurrent Programming Low mem requirements. 	 Monolithic Arch 整体架构类 Component Model Glue components together as a static image according to different requirements of application 	 Early version did not support multithreading Later, TOS event driven 	 Non-preemptive FIFO scheduling algorithm 缺点: Unfair to latter tasks especially when short tasks are waiting behind longer ones 	Provide 2 multi-hop protocols — dissemination(传播) & TYMO			
Contiki	 Lightweight C for WSN High mem requirement provide various features: preemptive multithreading, TCP/IP, GUI, etc. 	 Modular Arch 模块 化架构 Event driven— lightweight event scheduler dispatch events to running processes 	Preemptive multithreading—use protothreads, good for severely mem constraint devices	 Event driven OS所以 没有复杂的调度算法 Interrupt handler runs with priority 	Rich set of protocols reliable (IPv4, IPv6) / lightweight (uIP)			
MANTIS —Multimodal 多模式 system for NeTworks of In-situ wireless Sensors	 Multithreading OS for WSNs Lightweight & energy efficient MOS is portable across multiple platforms 	Layered Arch 分层架构 —Each layer acts as an enhanced VM to above layers	 MOS Preemptive multitasking Space allocated for RAM: a. global var space b. heap 	 Preemptive priority-based scheduling (use Round Robin in each priority class) Ready queue: Kernel, Sleep, High, Normal, Idle 	Implement the network stack in: a. user space b. MAC & PHY layers			
Nano-RK	1. Fixed, preemptive multitasking real-time (soft and hard real time) 2. High mem requirement	 Monolithic Kernel Arch Programmers can change parameters to meet the overall goal 	 Large memory consumption — saving the sate of each task Reduced performance & High energy consumption — frequent context switch 	 Priority scheduling at 2 levels: process and network Fully preemptive priority driven scheduling algorithm 	Lightweight networking protocol stack similar to sockets (communicate via socket) application bind and listen to port			

WSN Operating Systems LiteOS 1. Unix-like OS 1. Multitasking OS-Modular Arch 1. Round Robin & 1. In the form of files 2. Low mem a. LiteShell support Priority-based 2. Create file multithreading requirements b. LiteFS 2. sleep if no active corresponding to the 2. Run application as c. LiteOS Kernel task radio interface (use RR and separate threads, 3. Place data into the priority each thread has radio and transmit scheduling) memory space to later avoid errors in shared memory space.





WSN Middlewares								
	Characteristics							
TinyDB	在TinyOS基础上搭建的传 感器数据库	SRT—Semantic Routing Tree—根据语义进行 routing	table[sensors]=column[at tribute]+row[time]	当范围查询沿树传播时, 节点将仅将查询转发给查 询范围内的那些子节点, 潜在地使整个分支不必转 发或执行查询				
Maté	Event-driven On top of TinyOS	Can Programming at different levels. Adjusting parameters -> changing complete	Code can be broken into capsules of 24 byte-long instructions to fit TinyOS's packet					
Agilla	Agent-based based on Maté	WSN网络不需要预安装 (no pre-installed applications)应用,通过 代理(agent)在节点间迁移 (clone)实现 — agents migrate to nodes as needed	多个代理可以共同存在于 一个节点	Nodes are identified by their location rather than address				
Kairos	Macroprogramming middleware implemented in Python 宏编程中间件	Translates centralized program into node-specific distributed version	Execute locally at every node					
Magnet	The whole network appear as a single , unified JVM 让整个网络看起来像一个统一的JVM	VM和宏编程方法的混合	Standard Java program, then Magnet translate program into individual components					
SIXTH	Distributed system	Provides uniform access to sensing resources (sensors and data) via sensor adaptors	can be programmed directly, via SQL-like or intelligent agent	Can add and reprogram at running time				