Title
Mote Herding for Tiered Wireless Sensor Networks

Authors
Thanos Stathopoulos
Lewis Girod
John Heidemann
et al.

Publication Date
2005

Center for Embedded Networked Sensing

Mote Herding for Sensor Networks

Thanos Stathopoulos(*), Lewis Girod(*), John Heidemann(**), Deborah Estrin(*)
*CENS Systems Lab **USC/ISI

Mote Herding: A system architecture for Tiered Wireless Sensor Networks

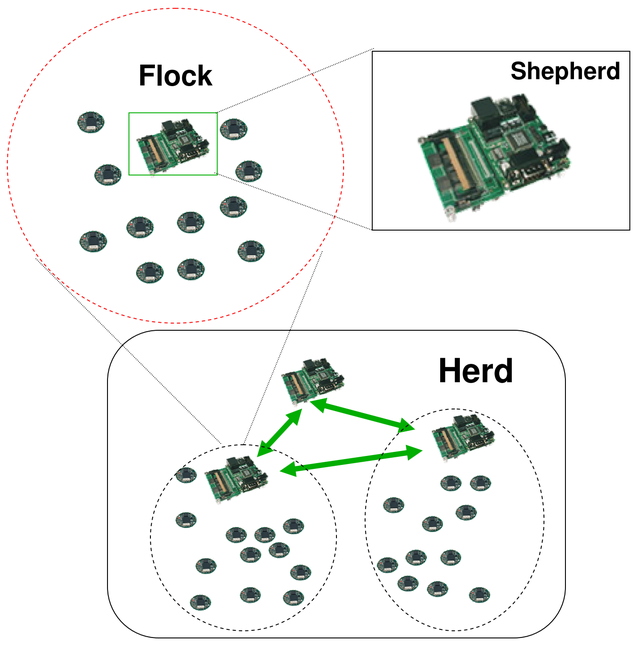
Mote Herding

Tiered systems: Collections of *motes* and *microservers* working together

Mote Herding explores guidelines to clarify how services should be developed in tiered systems

Architectural abstractions

- **Flock**: a collection of *motes* that connects to a microserver via a multihop tree
- **Shepherd**: a *microserver* responsible for a *single flock*. Shepherds connect to each other using *IP-based routing* over 802.11
- **Herd**: the collection of *flocks* and *shepherds* that forms the *entire* mote/microserver network



Design Principle

Shift system complexity from motes to microservers for those operations that require distributed decision making on the motes

Corollaries:

- **Centralize** decision making on a *microserver* to make decisions based on a *complete* set of information
- **Reduce** volatile state required for *system functionality* on motes
- **Utilize** computational and communication resources on microservers to support more *complex functionality*
- **Program** a significant part of the system in a *familiar* and resource-rich *32-bit environment*

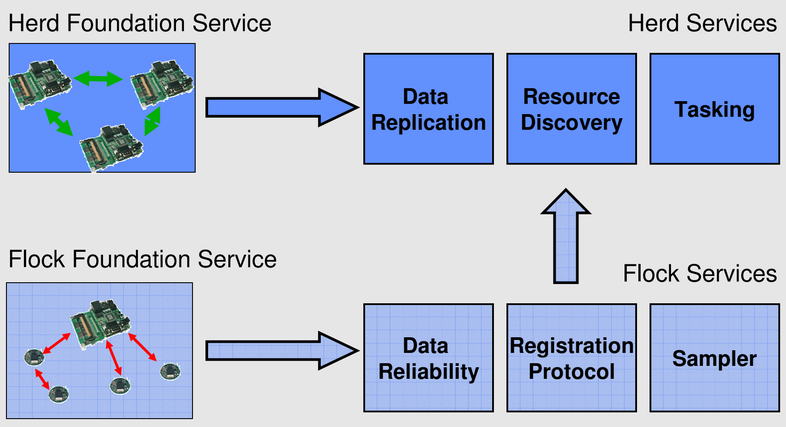
Mote Herding Services

Flock Services

- Operate on a *flock* under supervision of *a shepherd*
- Hybrid: code resides on *motes* and *microserver*
- Can use other services in the same flock but *cannot* interact with motes on *other flocks*
- Emphasize *master-slave* relationship between microservers and motes

Flock Foundation service: Mote Routing

- Provides *bi-directional* routing from motes to their shepherd
- Centralized protocol: *shepherd* makes *all* routing decisions
- Lack of distributed decision making on motes improves *stability* and *consistency*



Herd services

- Operate on the *entire* network
- Code resides only on *microservers*
- Interface to the mote network via *appropriate flock services*
- Allow for *distributed* and *collaborative* decision making between microservers

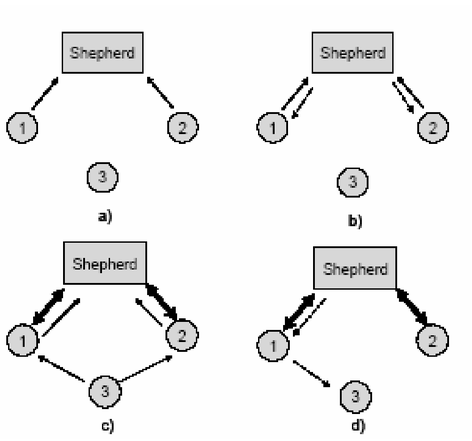
Herd Foundation service: Microserver State Replication

- StateSync: a reliable *state synchronization* protocol
- Allows *state sharing* on the microserver level for information pertaining several flocks
- Provides each microserver with a *global state view*

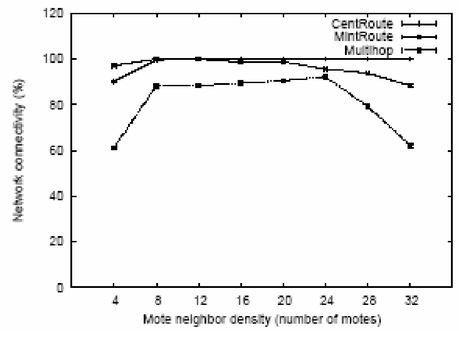
Service Details and Performance

CentRoute

- **Centralized** routing protocol based on *source routing*
- Motes broadcast beacons *only* when they wish to join a tree
- Any motes *attached* to the tree *forward* join beacons towards shepherd via unicast
- Shepherd picks *best path* using an ETX metric and sends a *unicast reply* to the mote
- Mote attaches to shepherd and uses the *last mote* on the reverse path as its *parent*
- Multi-shepherd ambiguities resolved on *microserver network*



The CentRoute *phased-join* operation



Improved connectivity at *high densities* due to lack of per-mote neighbor state

Neighbor density (motes)	CentRoute loop prob. (%)	MintRoute loop prob. (%)	Multihop loop prob. (%)
4	0	0.47	2.11
8	0	0.01	1.92
12	0	0.02	1.88
16	0	0.03	2.48
20	0	0.03	3.65
24	0	0.04	4.8
28	0	0.04	3.12
32	0	0.02	7.27

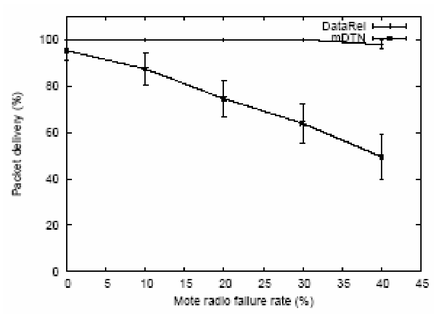
Loop avoidance improved by combination of *centralized decision making*, *source routing* and absence of *any* mote *decisions*

Usage (bytes)	RAM	ROM	RAM cost per neighbor
CentRoute	1274	17184	0
MintRoute	1689	12588	18
Multihop	1560	17292	19

Lower memory usage on the mote due to absence of O(N) state

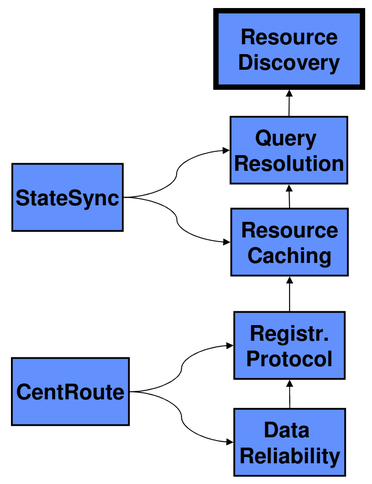
DataRel

- **End-to-end** mote data reliability protocol
- Can deliver 100% to the data at up to 30% *link failure* rates
- Low buffer usage and complexity compared to hop-by-hop protocols (mDTN)



ResDisc

- **Resource Discovery** herd service based on three subservices
- **Registration protocol** *registers* mote resources with *shepherd*
- **Resource caching** *exports* local flock resources and *imports* remote flock resources
- **Query resolution** replies to client queries based on information from resource caching



Future Work

- Implementing more services (data replication, tasking, system monitoring)
- Implement a coordinated imaging application using Cyclops and larger more powerful cameras

UCLA – UCR – Caltech – USC – CSU – JPL – UC Merced