LIPS: A Protocol Suite For Homeostatic Sensornet Management

Jonathan Tate and Iain Bate
Department of Computer Science, University of York
Heslington, North Yorkshire, YO10 5DD, United Kingdom
Email: { jt | iain.bate }@cs.york.ac.uk

Abstract

Sensornets are often deployed into inaccessible, dangerous, or changeable physical environments. Centralised control and management is generally infeasible. Autonomous, self-configuring, and self-managing mechanisms are required to provide a suitable infrastructure which reliably supports distributed applications, hiding any underlying instability. The Lightweight Integrated Protocol Suite (LIPS) coordinates time-sensitive activity, and regulates network size and density, in self-managing cellular sensornets. Although components can be implemented in isolation, each contributes part of a larger, integrated solution.

1. Introduction

Sensornets epitomise the growing trend toward complex systems. They can be large, consisting of thousands of wirelessly connected processing units. Their behaviour can be highly stochastic, leading to problematic deployment and low dependability [1]. This work addresses these issues by creating a logical cellular architecture and providing control over when messages are sent (to reduce collisions, minimise latencies, and make latencies more predictable) and organising duty cycling (to improve energy efficiency and availability).

Motes are generally equipped with the bare minimum of resources required to support the distributed application. Lightweight protocols with probabilistic success guarantees may be better suited than heavier, but more predictable, deterministic alternatives [2].

The *Lightweight Integrated Protocol Suite* (LIPS) provides a foundation upon which to build sensornet systems for which *dependability* and *security* are significant factors. Properties of particular interest are *availability*, *reliability*, *safety*, and *maintainability*.

Reliability is fundamentally important in achieving probabilistic guarantees of real-time behaviour, as lost messages may imply delays, missed deadlines and wasted energy [3]. However, individual sensornet nodes are prone to unpredictable, and sometime correlated, failures [4]. Distributed scheduling is difficult as there is usually no global clock, and many independent local clocks steadily drift out of synchronisation [5].

Sensornets become harder to manage and measure as they grow in size, and performance may decline with increasing size [6]. Unstable deployment environments may also induce suboptimal performance. If we cannot reduce the application requirements, and optimal protocol tunings cannot deliver the required performance, we might instead reconfigure the deployment network. System size can be minimised, addressing scalability issues, and its characteristics can be controlled, addressing the unpredictability of physical environments.

We assume we cannot control the environment, or change the location or available resources of nodes following deployment. We can, nevertheless, control the size and structure of a virtual sensornet [7] by managing the duty cycle of individual nodes in the physical network. We construct virtual networks by composing a logical hierarchy of cells. Within these virtual networks there is an adequate number and density of actively participating nodes in all physical regions, to support distributed application requirements, without unnecessarily wasteful redundancy or duplication.

LIPS employs four cooperating protocols to provide the autonomous management of low-level infrastructure required to provide the virtual cellular sensornet. Cooperation minimises the management overhead. The first two protocols, LISP and DCAP, coordinate time-sensitive activity within and between cells respectively, yielding a globally synchronised sequence of periodic timing events. The third protocol, CDAP, cyclically allocates duty periods to available nodes. The fourth protocol, ADCP, maintains a correctly sized working set of nodes drawn from a larger total population.

LIPS resides between the *Data Link Layer* and the *Application* layer in a conventional protocol stack, but makes no assumptions about either. Applications can therefore remain ignorant of these cross-layer con-

cerns, with energy conservation being given the highest priority. Some LIPS component protocols have previously been published in isolation; this paper shows how they cooperate, and evaluates the resulting benefits.

2. Related work

Sensornet operation is *greatly affected by different inter-related factors such as network traffic flows, network topologies, and communication protocols*, and the interaction of these factors is often unclear [8]. Timely and effective manual network configuration may be impossible owing to problem scale, difficulty in accessing individual nodes, and the highly dynamic nature of typical deployment environments.

Lin *et al.* [9] identify that sensornet performance may vary dramatically and unpredictably. Some links are stable in the long term, whereas the quality of other links varies considerably over seconds or minutes. The *competence* link metric differentiates between stable and unstable links. Considering both short- and long-term behaviour together in routing decisions enables better performance then either in isolation.

Lee *et al.* [10] argue that sensornet management mechanisms must be *energy-efficient*, *self-forming*, *self-configuring*, *self-stabilising*, and *scalable*. Effective sensornet management must operate efficiently, continually, and autonomously, providing an application-agnostic platform [8]. The sensornet should dynamically reconfigure its behaviour, and structure, in response to changing environments and application requirements. However, many existing sensornet management approaches provide only passive monitoring [10], requiring human intervention to induce response.

MANNA [11] is the first sensornet management architecture. The sensornet management function is divided into three planes, with strictly defined interplanar coordination policies. The *functional plane* describes the centralised, distributed, or centralised framework through which management information is distributed. The *physical plane* describes the location, resources, and state, of sensornet assets. The *information plane* describes the data, and data processing, collected and distributed within the network. Although comprehensive, MANNA requires a centralised *Management Information Base* against which all decisions are based.

BOSS [12] bases a service discovery management architecture on the UPnP protocol. BOSS provides *network information*, *localisation*, *synchronisation*, and *power management* functionality. Although UPnP is widely used, it is too heavyweight to run on typical sensornet mote platforms [12]. BOSS provides a bridge between a conventional network managed by UPnP, and a clustered sensornet in which nodes are only indirectly managed by UPnP. BOSS does not support automatic sensornet management; an end-user must be present in the system control loop, monitoring network state and deciding appropriate reactions.

WinMS [10] is an adaptive, policy-based sensornet management system. The sensornet designer specifies parameter threshold values which define acceptable system behaviour. If a measured value exceeds this threshold, the mapped set of actions is triggered automatically; no human oversight or intervention is required. However, it is not clear how sensornet designers obtain appropriate threshold values, or determine appropriate reactions to threshold breaches, without detailed knowledge of a specific running network.

Many communication protocols are described in the literature, which accept general and/or protocol-specific parameters to fine-tune performance for particular use cases [13]. Although these new protocols are valuable contributions to the field, a sensornet designer must balance the demands of multiple performance objectives. It is not always appropriate to create a new protocol to address a single factor in isolation [14], but this may be necessary if the existing options are not compatible or if their union is suboptimal.

Sensornet applications may divide tasks among multiple nodes. Subtasks may be allocated such that traffic flows and processing chains are aligned along routes from raw data sources to the eventual consumers. Achieving correct behaviour requires the subtasks of distributed tasks to be executed in the right place and at the right time, such that the processing elements occur in the right order and at times when the required raw data is available. Many timing and clock synchronisation approaches for sensornets exist; a detailed survey by Sundararaman *et al.* can be found in [15].

Raghunathan *et al.* [16] observe that energy optimisation for sensornets is complex as it *involves not only reducing the energy consumption of a single sensor node but also maximising the lifetime of an entire network*, requiring dynamic tradeoffs between *energy consumption*, *system performance*, and *operational fidelity*, yielding *up to a few orders of magnitude* of improved lifetime. Motes may operate for just days at full power, but sensornets may be active for months.

Dutta and Culler [17] state that *abstractions that are consistent over the diversity of power management techniques remain elusive*. The most common techniques are *duty-cycling*, cycling subsystem power to reduce its average energy draw, *batching*, buffering multiple operations to amortize a high startup or overhead cost, *hierarchy*, ordering boolean operations by their energy consumption and invoking low-energy operations before high-energy ones when the desired re-

sult is a conjunction of the operations, and *redundancy reduction*, reducing or eliminating redundancy through compression, aggregation, or message suppression.

3. Requirements and assumptions

3.1. Time-sensitive behaviour

Sensornets must manage behaviour over a diverse range of timescales. Individual network packet transmissions may occupy periods of the order 10^{-3}s or less, and task scheduling decisions may consider much shorter periods for *MHz*- or *GHz*-clocked processors, whereas networks may be required to operate continually for periods of the order 10^{7}s if lifetimes extend into years [16]. As the duty management problem encompasses durations spanning at least 10 orders of magnitude it seems unlikely that a single mechanism could be optimal for all purposes.

We therefore give separate treatment to *short-*, *medium-* and *long-term* schedule management. We informally define *medium-term* tasks as being of the order of seconds to minutes, with shorter tasks of subsecond duration being *short-term* and larger tasks of hours or longer being *long-term*. This is based on the Burns-Hayes model of *time bands* [18]. We also use the Burns-Hayes notion of zero-length *events* as the foundation for synchronisation of *activities* and *behaviours* occurring within *time bands*.

Our *short-term* band is equivalent to Newell's *biological band*, our *medium-term* band equivalent to Newell's *cognitive band*, and our *long-term* band equivalent to Newell's *rational* and *social* bands. Our *medium-term* band supports instances of distributed sensing and processing behaviours, these being the sensornet equivalent of *cognition*. Our *long-term* band focuses on managing populations and ensuring fair distribution of work; parallels exist between these and the *social* and *rational* aspects of Newell's model, although the equivalence is necessarily imperfect as we are considering machines rather than human systems.

3.2. Cellular network structure

The physical region inhabited by the sensornet is divided into a number of cells. This division of physical space defines the physical boundaries for the region occupied by each network cell. We assume full connectivity exists between all nodes in a cell. We assume that physical cell boundaries do not overlap. Wireless packet broadcasts may cross boundaries into neighbouring cells, thus enabling intercellular traffic.

We are not concerned with the physical distribution of nodes within cells. Depending on the design of the network, and the method of node deployment, this physical distribution may be uniform or nonuniform. When the management protocol is running, each cell should contain the same number of actively participating nodes, no greater than the total cell population.

At any given time any non-degenerate cell contains one or more nodes. Each node resides in exactly one cell. Each node knows to which cell it belongs at any given time, but does not need to know the other members, if any, of this cell. Geographical location, rather than node identity, is used to label sensor data and data flow endpoints. This follows from a generally accepted assumption that individual sensornet nodes do not have globally unique identifiers [19].

We assume that all nodes within a cell are equal peers, such that all cell members can support an equal share of the processing and storage burden for the distributed application. The failure of a single node may temporarily reduce the capability of the cell, but should not cause the entire cell to fail.

Unless each cell is a *singleton cell* containing exactly one node, the logical network is smaller than the physical network. It follows that protocols which become less reliable as the network size increases will benefit from addressing the network at the level of the logical cell rather than the physical node. As fewer physical nodes are involved, it is possible to achieve corresponding decreases in energy consumption and network congestion among other factors.

3.3. Communications

Consider a large sensornet consisting of many nodes, divided into cells containing smaller numbers of nodes in close geographic proximity. Within a cell each node has a similar view of the physical environment, and similar connectivity to nearby base stations or surrounding cells [20]. It follows that all nodes within a cell are approximately equivalent with respect to extracellular entities and environmental context.

Suppose that an external entity broadcasts a message received by all members of a cell. Unless the message is intended for a specific member of that cell, it is unclear which cell member or set of cell members should respond. Data packets to be forwarded to remote destinations need only be rebroadcast once; if all cell members rebroadcast this wastes energy, increases contention for the wireless medium, and risks collisions. If a tasking message requests that a sample value be read from the physical environment then all cell members will produce equivalent readings [21]. Consequently, energy and network capacity may be wasted in delivering multiple redundant messages.

By deterministically assigning responsibility for response, we implicitly identify the nodes which will not be required to respond. These nodes can switch unused

energy-hungry subsystems into low power modes. The consequent energy saving extends the useful lifetime of sensornets composed of nodes with finite energy resources [22]. Sensornets can run indefinitely if duty cycle allocation allows nodes to scavenge energy from the environment at the rate of consumption [23].

4. The LIPS protocol suite

Owing to the unpredictable nature of the physical environment and deployment methods, and imperfections in the reliability of network hardware, it will not suffice to construct a static logical network configuration. We must arrange for the network to continually self-monitor and self-manage to adapt to changing circumstances and cope with individual node failures without compromising system success. LIPS defines a set of protocols which cooperate harmoniously to achieve these goals. However, the sensornet designer may omit some protocols if they are not required; dependencies are discussed in section 4.5.

Dutta and Culler [17] identify the four main software techniques for energy efficiency improvement in sensornets as being *duty-cycling*, *batching*, *hierarchy*, and *redundancy reduction*. LIPS provides direct support for duty-cycling and redundancy reduction, as this is possible in an application-agnostic manner. Batching and hierarchy require application-dependent behaviour and are not directly exploited. However, these are not mutually incompatible with LIPS, and in fact could usefully be integrated by the sensornet designer.

Firstly, we consider synchronisation in the time domain. Section 4.1 describes the *Lightweight Improved Synchronisation Primitive* (LISP), which produces a synchronised sequence of periodic timing events for coordination of time-sensitive distributed behaviour. LISP operates within a fully-connected set of nodes, each of which has its own unreliable timer but no global clock, such as a cluster or cell of sensornet motes. Section 4.2 describes the *Dynamic Cellular Accord Protocol* (DCAP) which synchronises multiple instances of LISP such that time-sensitive behaviour can be coordinated across hierarchical sensornets composed of multiple clusters or cells.

Secondly, we consider dynamically managed logical network configuration. A logical network of controlled size and physical density is created from a larger physical network, enabling lightweight scale-sensitive protocols to function acceptably. This depends on the timing synchronisation provided by LISP and DCAP. Section 4.3 describes the *Cyclic Duty Allocation Protocol* (CDAP) which constructs and maintains a distributed duty schedule within a closed system of active sensornet nodes, dividing responsibility between these nodes. This provides *medium-term* duty coordination.

Section 4.4 describes the *Active Duty Control Protocol* (ADCP) which maintains the set of active nodes required by DCAP. A fixed number of nodes are selected at all times from a potentially larger pool of available nodes, with wear balancing implemented such that resource consumption is fairly distributed among all nodes but without excessive churn. This provides *long-term* duty coordination.

Numerous MAC protocols suited to sensornets have been proposed [24] at the *Data Link Layer* which manage contention for a shared medium. These protocols generally define and impose order over only a short period, and over a short distance in the case of a wireless shared medium. It is assumed that a MAC protocol will be selected and implemented to provide *short-term* coordination, but this is not provided by LIPS. The *medium-* and *long-term* coordination protocols described in sections 4.3 and 4.4 place no special requirements on the MAC protocol.

There is insufficient space to include the full details of each constituent member of LIPS here. Instead, we present an informal description of the aims and methods of these protocols and interrelationships. We refer the reader to other publications which provide in-depth technical details and analysis for each protocol.

4.1. LISP

We implement a lightweight feedback-driven primitive called the *Lightweight Improved Synchronisation Primitive* [25] to build and maintain a cyclic duty schedule. It is based on a variant of the biologically-inspired *synchronisation* phenomenon, which has been extensively studied and formalised elsewhere [26].

Inter-node coordination is achieved by cells acting as closed systems of pulse-coupled oscillators. The desired emergent synchronisation property requires all interacting entities to share an appropriate set of behaviours. LISP defines behaviours appropriate to the domain of sensornets, using a variant of *synchronisation* called *desynchronisation* [27].

Each cell contains n active nodes, all acting as periodic oscillators with period of e, the *system epoch* length. Each node measures its local oscillator phase, ϕ in the interval $[0, \phi_{max}]$ within an epoch. When $\phi = \phi_{max}$, the oscillator fires and triggers a synchronisation pulse event. A small, anonymous synchronisation pulse packet is broadcast, and the oscillator resets $\phi = 0$.

Each cell member receives these packets from all cell members, ignoring all instances except the *predecessor* and *successor* packets received immediately before and after a given node's own synchronisation

event. After a node receives this successor packet, it calculates the phase midpoint of the predecessor and successor, ϕ_{mid}. The node moves its own phase, ϕ, toward the midpoint, ϕ_{mid} by some phase difference $\phi - \phi_{mid}$ scaled by the *feedback parameter*, $f_\alpha \in (0, 1]$.

The system converges on a stable steady state where the order of node oscillator firing events does not change over time, and the sequence of events is distributed with equal interpulse delay between each consecutive event pair [27]. Improved variants of this primitive employ historical data to filter out spurious signals and noise [25] so as to cope with timing error, such as *release jitter* and *clock drift*, and communications error, such as *lost pulses* and *phantom pulses*.

4.2. DCAP

The *Dynamic Cellular Accord Protocol* [28] complements LISP, as defined in section 4.1. Whereas LISP achieves *desynchronisation* [27] within a network cell, DCAP achieves *synchronisation* between network cells [26]. *Synchrony* can be considered the complement of *desynchrony*. Special attention is given to preventing the potentially opposing targets of LISP and DCAP leading to stalemate or instability.

We assume a node can hear synchronisation pulse packets originating in its own cell, and in adjacent cells, and distinguish between these cases. This can be implemented by encoding cell ID data into existing LISP packets; no additional packets are required. Typical planar tilings require just 4 unique identifiers [28], which can be implemented using 2 data bits.

A separate instance of LISP runs in each cell. Within a cell, LISP pushes synchronisation events apart. Between cells, DCAP pulls synchronisation events together. Synchronisation packets originating within the cell are channelled to LISP. The remaining events of extracellular origin are channelled to DCAP, which stores the time of each. When the LISP successor event is observed, the node calculates the phase offset between each extracellular event relative to its own. The arithmetic mean of these phase offsets is calculated, and the node moves its own phase toward this, the action being scaled by the *feedback parameter*, f_β.

The system converges on a stable steady state in which the firing time of any given node's synchronisation event is equal to that of the equivalent nodes in all adjacent cells, within the margin imposed by packet transmission times. Eventually the entire network converges, such that each cell contains an equivalent sequence of periodic synchronisation events, provided that the graph of cells is connected [26]. This hierarchical coordination provides network-wide synchrony.

DCAP requires that each node listens for extracellular synchronisation events occurring within a defined period, beginning halfway between the LISP predecessor event and the node's event, and ending halfway between the node's event and the LISP successor event. This necessary listening period is equal to the duty period assigned to each node by CDAP, and hence the two protocols are entirely compatible.

4.3. CDAP

The *Cyclic Duty Allocation Protocol* [29] is an application- and platform-agnostic lightweight protocol to cycle duty between the nodes of a network cell. System epochs are divided into portions of equal length, e, and allocated fairly among nodes, such that each node is assigned responsibility for one portion during each epoch. The specified number of nodes are deterministically assigned this responsibility at any arbitrary time, removing ambiguity as to which node must respond to stimuli. As cells approach stable LISP equilibrium states, nodes can identify periods in which energy-saving states can safely be entered.

Each node is assigned an equal duty period of length $d = {}^{e}/_{n}$. LISP provides the timing of predecessor and successor synchronisation events relative to a node's own synchronisation event. The assigned duty period for that node starts halfway between predecessor and self, and ends halfway between self and successor. This deterministically assigns equal duty periods in strict order when LISP is stable.

Equal length duty periods are assigned to nodes in strict, unchanging round-robin manner. These do not overlap, and exactly one node is assigned *on-duty* at any given time. Applications may require $k > 1$ nodes *on-duty* simultaneously at all times within each region, perhaps to provide sufficient processing capacity or sensor coverage redundancy. We simply arrange for each node to remain on duty for k such assigned periods, remaining on duty for time dk each epoch.

When a node is *on-duty*, it is assigned responsibility for executing local cell duties, such as packet forwarding or sensing, as required by the sensornet application. When a node is *off-duty* it has no such responsibility, and energy-hungry components such as wireless communications modules can be switched off.

A naïve implementation would block necessary incoming LISP event timing data, as communications would be disabled during predecessor and successor events. We therefore have nodes maintain small *synchronisation windows* during which communications modules are active, though listening only, around the predicted times of these events. Window size is progressively reduced, as a function of either consecutive successful prediction count or measured prediction error magnitude, until they are small but sufficiently

large to handle realistic timing error. A well-defined recovery process handles recovery from unexpected cell instability or overzealous window shrinking.

Figure 1 illustrates the CDAP states and state transitions in UML statechart format. There are two composite states; *stable* and *unstable*. The former is adopted by nodes residing in cells that are sufficiently stable that it is reasonable to predict the timing of predecessor and successor phase neighbour synchronisation events in subsequent epochs, whereas the latter is adopted by nodes when there is insufficient data or insufficient cell stability to make any such prediction with confidence.

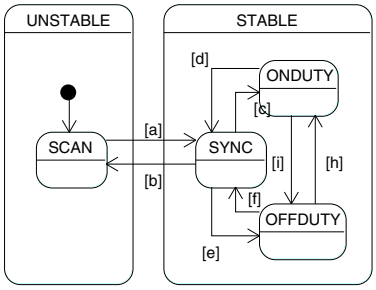

Figure 1. Finite State Machine for CDAP States

4.4. ADCP

The *Active Duty Control Protocol* [30] regulates the population of a network cell to maintain an active population of fixed size. This ensures that applications, which require a certain level of processing resource provision or sensor redundancy to acquire data of sufficient quality, are supported. Protocols assuming a given active cellular population size, such as DCAP, are able to work effectively. Nodes which are not currently required for active service can be placed into very low-power deep sleep states for *long-term* energy management and conservation.

If too many nodes are active, some are made inactive. If too few nodes are active, some of the pool of spares are made active. It is assumed that each node is aware of the target active cell population n, and the total number of nodes which are available for selection for active duty m, but does not have omniscient access to the current active cell population δ.

Figure 2 shows ADCP states in a UML statechart. There are three composite states, corresponding to three distinct cell subpopulations. Nodes in the *inactive* composite state are members of the *inactive set*, Γ. Nodes in the *reserve* composite state are members of the *reserve set*, Λ. Nodes in the *active* composite state are members of the *active set*, Δ.

We must know the current active node population before we can decide whether this must be changed. All nodes apply the same mechanism in the same environment and thus produce similar estimates, although any individual estimate may be inaccurate. Nodes which listen to the wireless medium for complete epochs count LISP transmissions, and those listening periodically measure inter-transmission delays.

Each active node in Δ estimates δ. If a node estimates $\delta > n$, it probabilistically decides whether to join Λ; the probability is proportional to the degree of cell overpopulation. If a node estimates $\delta = n$ it may volunteer, with fixed probability, to sleep and join Λ to provide *long-term* fair duty cycling within its cell.

Each reserve node in Λ usually sleeps, but occasionally wakes and estimates δ. If a node estimates $\delta < n$, it probabilistically decides whether to remain awake, becoming active and joining Δ; the probability is proportional to the degree of cell underpopulation. Inactive nodes in Γ are released into Λ to replenish the node pool at a rate equivalent to that of node failure.

4.5. LIPS interactions and dependencies

The same lightweight synchronisation pulse transmissions are reused by all LIPS protocols in a variety of ways toward the achievement of multiple goals. In addition to reducing energy consumption by minimising the energy cost of each transmission, the number of transmissions is also minimised. Furthermore, all protocols have been designed to share these transmissions, thereby enabling inter-protocol cooperation and avoiding conflict. This is in contrast to the selection of arbitrary sets of unrelated protocols which may have very different assumptions about the task, may be incompatible owing to complex and unforeseeable interactions, and may make different demands on the underlying infrastructure and high-level application.

Figure 3 illustrates protocol relationships, with dependencies shown by arrows. LISP and DCAP handle

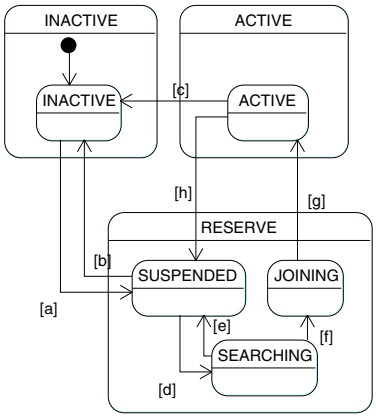

Figure 2. Finite State Machine for ADCP states

timing coordination, CDAP and ADCP handle *duty* coordination, and node state management controls *energy* consumption. The best results are obtained when all LIPS protocols are simultaneously active and thus able to cooperate toward a shared goal. If only a subset is implemented, functionality equivalent to that of the excluded protocols may be provided by other means.

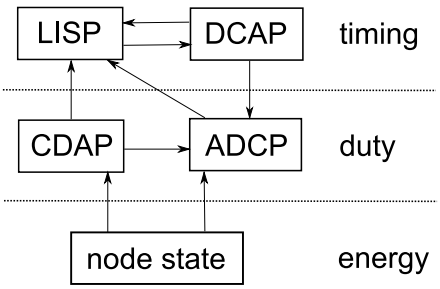

Figure 3. Dependencies between LIPS elements

LISP forms the foundation of the protocol suite. Implemented in isolation, it provides a cell-local timing primitive; although useful, its utility is limited without integration with other network management functions. The other protocols could be modified to use a different source of periodic synchronisation messages, transmitted in strict round-robin rotation with equal interpulse delay, but we do not examine this option further.

DCAP extends LISP to work across a multicellular network, by coordinating the activity of multiple cell-local instances. It is obvious that DCAP can be omitted entirely in unicellular networks, or disabled in very short-lived networks following initial synchronisation.

CDAP uses timing data extracted from observed LISP activity to allocate duty periods. CDAP also ensures that the resulting schedules do not starve LISP and DCAP of raw timing observation data. CDAP can be omitted entirely if task and sleep scheduling is not required, for example where energy sources are infinite and unmanaged multi-node responses are acceptable.

ADCP uses data derived from LISP synchronisation events to estimate node populations. This is a required input to the probabilistic decision making processes employed to manage node populations, in itself necessary for the correct functioning of DCAP. ADCP can be omitted entirely if the node population is guaranteed to be correct, for example in controlled manufacturing environments with infallible mote hardware platforms.

If both CDAP and ADCP are implemented, all CDAP activity occurs within the ADCP *active* state. However, ADCP is not a prerequisite for CDAP.

5. Evaluation

We now evaluate the protocol suite behaviour in non-ideal sensornets and deployment environments. Whereas the papers in which the constituent members of LIPS are defined [25], [28], [29], [30] evaluate protocols in isolation, in this section we evaluate the set of all protocols working together. This is significant; all LISP suite members must at least be mutually compatible, and ideally would be mutually cooperative.

With a suite of four protocols of which some are optional, and assuming numerous measurable behaviours for each, it is obvious that combinatorial explosion renders infeasible any attempt to address all combinations within a paper of reasonable length. We focus on those necessary to demonstrate correct interoperation.

Our evaluation strategy begins by considering timing properties, examining whether LIPS can coordinate timing behaviour across cellular networks, then considering if this remains successful when duty cycling and state management are introduced using CDAP and ADCP. Having confirmed that timing management behaviour is not adversely affected, we examine the extent to which CDAP and ADCP can cooperate when achieving their related but fundamentally different duty management goals.

Real-world sensornets may take any spatial configuration as required to fit the physical environment, but here we evaluate the simple planar case. A network of $c = 16$ cells was constructed with the cell boundaries taking the shape of a hexagonal planar tiling. Each cell has a target active population $n = 10$ because this is generally an energy-efficient cluster size for sensornets containing hundreds to thousands of nodes [31]. However, LIPS will function correctly with other cluster sizes and cells counts, either larger or smaller.

Scalability problems, implicitly associated with some protocols and control mechanisms, may only become apparent in large sensornets [6]. For sensornets not implementing ADCP we specify that each cell has total available population of $m_a = n = 10$, giving a total network population of $cm_a = 160$ nodes. For sensornets implementing ADCP we specify that each cell has total available population of $m_b = 50$, such that $m_b > n$. This models scenarios in which a sensornet is deployed with many spare nodes, on the assumption that many will fail during the network lifetime and must be replaced automatically. This gives a total network node population of $cm_b = 800$ nodes.

In an infinite hexagonal planar tiling each hexagonal cell is surrounded by 6 neighbouring cells. However, in a finite example, the cells around the perimeter may have 2, 3, 4, or 6, neighbours. To ensure that each cell has all 6 neighbours we join the north and south edges, folding the plane into a cylinder, then join the east and west edges, folding the cylinder into a torus. This removes nonuniformity of cell adjacency degree,

avoiding any edge effect influence in measurements.

5.1. Timing synchronisation event generation

Consider the production of periodic timing events, against which elements of distributed activity can be coordinated. Timing event production must be globally synchronised to support globally distributed behaviour. All other behaviour is dependent on the LISP synchronisation events; if the latter are produced correctly, all other LIPS functionality will work as expected. We first consider activity within a single cell, as managed by LISP, then consider the coordination of timing and state management behaviour across the network by progressively adding DCAP, CDAP and ADCP in turn. We examine whether the single cell behaviour has been changed by interaction with its peers.

E_α measures time for LISP to reach stability in a specified cell [25]. E_β measures time for DCAP to reach stability across all cells [28]. We measure each of E_α-E_β, where defined, in network configurations *A*-*D*. *A* is a unicellular network, a single cell of the 16-cell network defined above, implementing LISP. *B* extends *A*, adding DCAP for cross-network coordination within the full 16-cell network. *C* extends *B*, adding CDAP to minimise duty within stable cell populations in the medium-term. *D* extends *C*, adding ADCP to maintain stable cell populations in the long-term.

	A	B	C	D
E_α	38	47	56	108
E_β	-	67	62	117

Table 1. Epochs to reach stable steady state

Table 1 indicates the number of system epochs, each of e time units, required for LIPS timing coordination protocols to reach their steady stable states. At this point, the correct behaviour has been obtained.

Under configuration *A* there is only one cell, so the corresponding $E_\beta(A)$ value is undefined as its definition requires the existence of at least two cells. This $E_\alpha(A)$ value gives a baseline for comparison in the related multi-cellular network configurations *B*-*D*.

Examining the LISP stabilisation periods $E_\alpha(A)$ to $E_\alpha(D)$, and the DCAP stabilisation periods $E_\beta(B)$ to $E_\alpha(D)$, we observe that in all cases the combination of LIPS protocols did not prevent the timing synchronisation protocols from attaining the correct stable state in a finite number of system epochs. This is the most important result; it demonstrates the successful and simultaneous implementation of all LIPS protocols.

There is a price to be paid for the introduction of additional LIPS suite members, in that protocol stabilisation times increase with each addition. This is not unexpected, as each protocol either overlays additional factors in the timing of synchronisation events, or potentially interferes with the reception of the associated synchronisation event packets. However, the increase is a relatively small number of epochs. Provided that epochs are of reasonable length, usually orders of magnitude smaller than minutes [25], stabilisation periods remain a relatively small proportion of sensornet lifetimes which may extend into months or years [16]. Furthermore, performance during stabilisation is not worse than that observed without LIPS.

In all multi-cellular network configurations *B*-*D*, we observe that DCAP takes longer to stabilise than LISP. This is expected behaviour [28], as the influence of DCAP is designed to be smaller than that of LISP until such time as LISP stabilises, at which point the influence of DCAP becomes significant. It follows that DCAP stabilisation will generally follow LISP stabilisation, unless by chance the initial configuration of the network is unusually conducive to the former.

Introducing CDAP increases stabilisation times as it reduces the synchronisation listening periods toward a defined minimum. Whereas this is desirable, until the network has attained LISP stability it is possible that from time to time the synchronisation window is too short to enclose a required synchronisation event. Network designers implementing CDAP alongside LISP and DCAP should consider tuning the synchronisation window shrinking policy so as to avoid being overly aggressive, and hence avoid the occurrence of many such synchronisation window prediction misses.

Introducing ADCP increases stabilisation times as it changes the cell active population; every time this changes, the cell must again restabilise. This is particularly pertinent at the outset, when many such changes occur as the active population is reduced from m_b to n. It follows that network designers implementing ADCP should consider damping population changes [30] when LISP stability is attained, to balance responsiveness to node loss against stability.

5.2. Duty cycling and state management

We now consider state management and duty cycling functionality. There are two primary goals; to manage redundancy, and to minimise energy consumption.

Firstly, we consider redundancy management. The geographic distribution of nodes within a sensornet is typically such that more than one node is suitably positioned to sense a given event in the physical environment, to forward an incoming data packet, or to respond to an application request for resources. The sensornet must somehow construct a mapping between action requests and responding nodes at runtime.

In a cellular sensornet, this can be achieved by

having exactly one node responsible for any incoming request at any given time, and sharing this responsibility between cell members. In its simplest form this implies a mutual exclusion property. This can be extended for any x nodes simultaneously responsible for request handling where $1 \leq x \leq n$ [29].

y	{A, B}	C_a	C_b	D
0	0.0000	0.0014	0.0058	0.0043
1	0.0000	0.9889	0.9832	0.9611
≥ 2	1.0000	0.0096	0.0110	0.0345

Table 2. Mutual exclusion measurements

Table 2 indicates the proportion of total network runtime for which the number of nodes, y, simultaneously assigned responsibility for handling requests is 0, 1, or ≥ 2. Assuming $x = 1$, the ideal proportion for $y = 0$ would be 0, the proportion for $y = 1$ would be 1, and the proportion for $y \geq 2$ would be 0. Measurements were taken under configurations *A* to *D* within a single cell, representative of all network cells, by sampling y every 1×10^{-3} seconds in the modelled sensornet.

Under configurations *A* and *B* there is no attempt to enforce mutual exclusion, and all nodes are simultaneously assigned responsibility for handling requests. As each cell contains more than one node, the required mutual exclusion property trivially never holds, which is undesirable where excessive redundancy harms application behaviour or energy efficiency.

Configurations C_a and C_b are similar to configuration *B*, but with CDAP activated. Under C_a cells have $m_a = 10$ nodes, whereas under C_b cells have $m_b = 50$ nodes. In each case we observe that at almost all times, around 99% and 98% of total runtime respectively, the desired mutual exclusion condition of $y = 1$ holds. For around 1% of time the undesired condition $y \geq 2$ holds, and for less than 0.5% the undesired $y = 0$ condition holds. Enforcement of mutual exclusion by CDAP is not harmed by LISP and DCAP moving the relative timing of the synchronisation events.

Configuration *D* is similar to configuration C_b, but with ADCP activated. This reduces the active population from $m_b = 50$ to $m_a = 10$, creating a network of similar scale to configuration C_a. Enforcement of mutual exclusion by CDAP is not harmed by ADCP modifying the identity and size of the active population. The proportions of time for which y is 0, 1, or ≥ 2 fall between the equivalent figures for C_a and C_b, reflecting the fact that the active population undergoes a similar transition from the start of network runtime.

Secondly, we consider energy efficiency. Subtle tradeoffs may exist in balancing energy efficiency and system performance [16]. Sensornets are typically severely resource constrained, so controlling the rate of energy consumption is very important in managing network performance and lifetime. This is generally achieved by switching off nodes or node subsystems when not required, or reducing subsystem performance to the minimum required to support the application.

LIPS favours the former approach, minimising redundancy rather than reducing capacity. Chemical batteries subjected to high current drain cannot deliver their full rated energy capacity. Switching nodes into very low power states implies low discharge currents, allowing batteries to recover a significant proportion of lost capacity, due to the *relaxation effect* [16].

	{A, B}	C_a	C_b	D
proportion	1.0000	0.1018	0.0199	0.0192

Table 3. Relative time assigned to active duty

Table 3 shows the average proportion of time nodes are assigned to active duty. At other times nodes are not assigned to active duty, either in the medium-term by CDAP or the long-term by ADCP, and hence can enter low-power energy saving states. This measure is independent of any particular sensor mote platform.

Under configurations *A* and *B*, all nodes are assigned to active duty at all times. This maximal redundancy yields a suboptimal energy efficiency profile, but provides an excellent baseline for further comparison.

Configuration C_a assigns nodes to active duty for approximately $\frac{1}{m_a} = \frac{1}{10}$ of all time. Configuration C_b assigns nodes to active duty for approximately $\frac{1}{m_a} = \frac{1}{50}$ of all time. This indicates that the union of LISP and DCAP does not inhibit CDAP functionality.

Configuration *D* assigns nodes to active duty for approximately $\frac{1}{m_b} = \frac{1}{50}$ of all time. As nodes transition between ADCP *active* and *reserve* populations over time, each node is eventually allocated an equal proportion of the duty burden by CDAP. However, as there are 50 candidate nodes per cell in *D*, as compared to 10 per cell in C_a, the assigned duty time for each node is scaled by approximately $\frac{m_a}{m_b} = \frac{1}{5}$. This yields a similar duty burden per node to C_b, while retaining the desired duty period length of C_a.

6. Conclusions

The *Lightweight Integrated Protocol Suite* (LIPS) coordinates time-sensitive activity, and regulates network size and density, in self-managing sensornets. Each protocol contributes part of a larger, integrated solution to the problem of automated low-level infrastructure management for self-managing virtual cellular sensornets. Implementing the full suite offers the greatest potential for improved performance over an unmanaged, or manually managed, sensornet infrastructure.

References

[1] F. Zhao, "Challenge problems in sensornet research," Keynote at NSF NOSS PI meeting and Distinguished Lectures at Johns Hopkins and Princeton, 2005.

[2] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. 11th International Conference on Acoustics, Speech, and Signal Processing*, 7-11 May 2001, pp. 2033–2036.

[3] I. Akyildiz and I. Kasimoglu, "Wireless sensor and actor networks: research challenges," *Ad Hoc Networks*, vol. 2, no. 4, pp. 351–367, 2004.

[4] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *Proc. 1st European Workshop on Sensor Networks*, 19-21 January 2004, pp. 307–322.

[5] S. PalChaudhuri, A. Saha, and D. Johnson, "Adaptive clock synchronization in sensor networks," in *Proc. 3rd International Conference on Information Processing in Sensor Networks*, 22-23 April 2004, pp. 340–348.

[6] J. Tate and I. Bate, "Sensornet protocol tuning using principled engineering methods," *The Computer Journal*, vol. 53, no. 7, pp. 991–1019, 2010.

[7] A. Jayasumana and Q. Han, "Virtual Sensor Networks - a resource efficient approach for concurrent applications," in *Proc. 4th International Conference on Information Technology*, 2-4 April 2007, pp. 111–115.

[8] M. Yu, H. Mokhtar, and M. Merabti, "A survey of network management architecture in wireless sensor network," in *Proc. 7th PostGraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting*, 26-27 June 2006, pp. 201–206.

[9] S. Lin, G. Zhou, K. Whitehouse, Y. Wu, J. Stankovic, and T. He, "Towards stable network performance in wireless sensor networks," in *Proc. 30th Real-Time Systems Symposium*, 1-4 December 2009, pp. 1–11.

[10] W. Lee, A. Datta, and R. Cardell-Oliver, "WinMS: Wireless sensor network-management system," The University of Western Australia, Tech. Rep. UWA-CSSE-06-001, June 2006.

[11] L. Ruiz, J. Nogueria, and A. Loureiro, "MANNA: A management architecture for Wireless Sensor Networks," *IEEE Communications Magazine*, vol. 41, no. 2, pp. 116–125, February 2003.

[12] H. Song, D. Kim, K. Lee, and J. Sung, "UPnP-based sensor network management architecture," in *Proc. 2nd International Conference on Mobile Computing and Ubiquitous Networking*, 13-15 April 2005, pp. 7–12.

[13] H. Karl and A. Willig, *Protocols And Architectures For Wireless Sensor Networks*. Hoboken, NJ: Wiley, 2005.

[14] C. Ee, R. Fonseca, S. Kim, D. Moon, A. Tavakoli, D. Culler, S. Shenker, and I. Stoica, "A modular network layer for sensornets," in *Proc. 7th USENIX Symposium on Operating Systems Design and Implementation*, 6-8 November 2006, pp. 18–27.

[15] B. Sundararaman, U. Buy, and A. Kshemkalyani, "Clock synchronization for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 281–323, 2005.

[16] V. Raghunathan, C. Schurgers, S. Park, and M. Srivastava, "Energy aware wireless microsensor networks," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 40–50, March 2002.

[17] P. Dutta and D. Culler, "System software techniques for low-power operation in wireless sensor networks," in *Proc. 16th International Conference on Computer-Aided Design*, 6-10 November 2005, pp. 925–932.

[18] A. Burns and G. Baxter, *Structure for Dependability*. London: Springer, April 2006, ch. "4. Time bands in systems structure", pp. 74–88.

[19] L. Ni, Y. Zhu, J. Ma, Q. Luo, Y. Liu, S. Cheung, Q. Yang, M. Li, and M. Wu, "Semantic Sensor Net: an extensible framework," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 4, no. 3/4, pp. 157–167, 2009.

[20] V. Paruchuri, S. Basavaraju, A. Durresi, R. Kannan, and S. Iyengar, "Random asynchronous wakeup protocol for sensor networks," in *Proc. 1st International Conference on Broadband Networks*, October 2004, pp. 710–717.

[21] Y. Gao, K. Wu, and F. Li, "Analysis on the redundancy of wireless sensor networks," in *Proc. 2nd International Workshop on Wireless Sensor Networks and Applications*, 19 September 2003, pp. 108–114.

[22] C. Liu, K. Wu, and J. Pei, "An energy-efficient data collection framework for wireless sensor networks by exploiting spatiotemporal correlation," *IEEE Transactions on Parallel Distributed Systems*, vol. 18, no. 7, pp. 1010–1023, 2007.

[23] X. Jiang, J. Polastre, and D. Culler, "Perpetual environmentally powered sensor networks," in *Proc. 4th International Conference on Information Processing in Sensor Networks*, 25-27 April 2005, pp. 65–70.

[24] I. Demirkol and C. Ersoy, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 115–121, April 2006.

[25] J. Tate and I. Bate, "An improved lightweight synchronisation primitive for sensornets," in *Proc. 6th International Conference on Mobile Ad-hoc and Sensor Systems*, 12-15 October 2009, pp. 448–457.

[26] R. Mirollo and S. Strogatz, "Synchronization of pulse-coupled biological oscillators," *SIAM Journal Of Applied Mathematics*, vol. 50, no. 6, pp. 1645–1662, December 1990.

[27] A. Patel, J. Degesys, and R. Nagpal, "Desynchronization: The theory of self-organizing algorithms for round-robin scheduling," in *Proc. 1st International Conference on Self-Adaptive and Self-Organizing Systems*, 9-11 July 2007, pp. 87–96.

[28] J. Tate and I. Bate, "A feedback-driven timing synchronisation protocol for cellular sensornets," in *Proc. 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 8-12 November 2010, pp. 482–491.

[29] J. Tate and I. Bate, "Energy efficient duty allocation protocols for wireless sensor networks," in *Proc. 14th IEEE Conference on Engineering of Complex Computer Systems*, 2-4 June 2009, pp. 58–67.

[30] J. Tate and I. Bate, "Maintaining stable node populations in long-lifetime sensornets," in *Proc. 15th IEEE Conference on Engineering of Complex Computer Systems*, 22-26 March 2010, pp. 159–168.

[31] D. Wang, "An energy-efficient clusterhead assignment scheme for hierarchical wireless sensor networks," *International Journal of Wireless Information Networks*, vol. 15, no. 2, pp. 61–71, June 2008.