7 Resource Management in the Internet of Things: Clustering, Synchronisation and Software Agents

Tomás Sánchez López[1], Alexandra Brintrup[2], Marc-André Isenberg[3], Jeanette Mansfeld[3]

[1] Department of Engineering, University of Cambridge

[2] Said Business School, University of Oxford

[3] BIBA - Bremer Institut für Produktion und Logistik, University of Bremen

Abstract. The objects of the Internet of Things will be empowered by embedded devices whose constrained resources will need to be managed efficiently. It is envisioned that these devices will be able to form ad-hoc networks, and that the connection from these networks to the Internet of Things infrastructure will not always be possible. In this chapter we propose the use of clustering, software agents and synchronisation techniques in order to overcome the challenges of managing the resources of the Internet of Things objects. We argue that clustering will be beneficial to reduce the energy expenditure and improve the scalability and robustness of the object networks. Software agents will aide in the automation of task, both for the objects and the Internet of Things users. Finally, synchronisations techniques will be necessary to address the various challenges of harmonising plenty of copies of object data with potentially partially disconnected Internet of Things architecture components.

7.1 Introduction

Despite the many technical and operational questions arising from the Internet of Things concept and the various interpretations of what the Internet of Things is and what promises it will deliver, it appears there is general consensus that the Internet of Things will empower users and objects to share information in a seamless, automated manner. In this context, the Internet of Things promises a new generation of the Internet, in which global connectivity moves towards everyday objects and things, radically widening the scope of Internet-based applications. On these premises, the management of an escalating number of connected devices,

D. Uckelmann et al. (eds.), *Architecting the Internet of Things*,
DOI 10.1007/978-3-642-19157-2_7, © Springer-Verlag Berlin Heidelberg 2011

together with a movement towards their increasing autonomy and relatively limited capabilities, pose a number of challenges that are yet to be explored.

This chapter will investigate a number of techniques aimed at addressing the challenges arising from the increasing number of connected objects, such as limited computation and energy, unreliable wireless channels and the impossibility of ubiquitous network access, repetitive and mundane user interactions, given the complexity of the architecture. Within this scope, three major interconnected topics will be explored: First, the grouping of objects into *clusters* in order to overcome scalability, energy efficiency and robustness issues, secondly, the use of *software agents* to represent and manage objects and users, moving part of the complexity to the architecture and providing a bridge between the users and the things, and, thirdly, techniques for bidirectional *synchronisation* of object knowledge in order to support operations and provide resilience in situations having only intermittent or unreliable network connectivity.

Hence, this chapter attends to represent a useful contribution on the implications of the Internet of Things vision, putting emphasis in actual problems and functional needs that will arise from a future architecture that today is little more than abstract ideas. The remaining chapter is organised as follows: Section 7.2 includes a literature review about the current state of research as well as related research areas in terms of chapter scope. Section 7.3 presents general assumptions as well as a definition about the Internet-connected objects underlying this chapter. Sections 7.4, 7.5 and 7.6 refer to the three interconnected topics, namely clustering, software agents and synchronisation, and illustrate their possible adoption within an Internet of Things. Concluding the chapter section 7.7 summarises the presented concepts and gives an outlook about the expected consideration of the described concepts within the development of the Internet of Things.

7.2 Background and Related Work

7.2.1 Clustering

Clustering is a popular method of organising wireless network topologies, in which a few nodes, the cluster heads (CH), are elected as representatives to route the traffic originated in the entire network. The clustering of intelligent computing devices has been widely researched in the fields of Wireless Sensor Networks (WSN) and Mobile Ad-hoc NETworks (MANET), although aiming at different objectives. The main objective of a MANET is network reliability and the accessibility of nodes. This is realised by building meshed networks without central authorities. Each node is connected to several other nodes, which always allow alter-

native communication routes from one node to another. Due to the functionality of in-network routing, all nodes act as routers with their own routing tables; this causes high activity rates of the nodes with the corresponding energy consumption. On the other hand, the clustering approaches of WSN are more hierarchical, using CHs as decentralised authorities for realising mostly star or tree topologies. WSNs vary in their objectives: there are existing approaches aiming to fault-tolerance, load-balancing, energy consumption, increased connectivity and reduced packet delay. While MANETs are generally built to handle objects in dynamic environments, WSN are traditionally used to cluster more or less static nodes. Although the mobility rates within clusters of autonomous objects within the Internet of Things is envisioned to be higher than the traditional mobility inside WSNs and MANETs, the research on those approaches is a valuable basis for the development of energy-efficient clustering methods for autonomous objects. For a better understanding of the requirements and challenges of the clustering of objects within the Internet of Things, this section will review the literature of WSN and MANETs in this area.

We would first like to compare those ad-hoc wireless clustering protocols that consider both mobility and energy-efficiency. The properties that we would like to compare are listed below. It is important to note that this comparison does not pretend to be an exhaustive listing of properties, but just those aspects that we consider most important. The studied properties are the following:

- Type: If the protocol is specifically for WSNs or for more general MANETs.
- Controlled variable CH period: CHs may be elected for periodic or aperiodic time intervals. Aperiodic intervals provide more flexibility since they can better manage the extra resources that the CH will use. We only consider those aperiodic intervals that can be effectively controlled, listing their variable that is used to compute them.
- CH election according to node conditions: If a node is elected according to its own conditions. The type of condition is also listed. Node's condition influencing its election as CH is generally a beneficial strategy since it provides first hand decision information.
- Synchronisation: If the nodes need synchronisation for either electing the CH or operating inside the cluster. Synchronisation among network nodes is costly and must be avoided when possible.
- Global cluster information: If the cluster nodes need to store information about all the cluster members in order to perform the CH election or to operate. Global information implies poor scalability with the number of network nodes.
- Multi-hop routing: Sometimes the clustering protocol may lead to developing a routing mechanism to exchange information among network nodes. Multi-hop routing mechanisms are beneficial because they can route communication packets between two nodes that are not directly connected.
- CH election complexity: An estimation of the complexity to elect a new CH. In general, the lower the complexity, the more efficient is the proposed algorithm.

Table 7.1 shows a summary of the comparison. We found five MANET clustering protocols that explicitly consider node's energy as a factor for CH election. We also found two WSN protocols that consider not only energy but also mobility. By mobility we mean not only that nodes may move inside the network, but also that the addition of new nodes and the removal or death of nodes is also considered. The small number of related work found, suggests that it is not common for MANET clustering protocols to focus on CH energy efficiency, and is also not common for WSN clustering protocols to consider mobility. Judging by the distribution of protocol types, the latter group seems rarer than the former.
All the listed protocols consider node's residual energy in order to elect the CH, although some of them use also other factors. Controlling the period that a node will be a CH is quite uncommon. Only MoCoSo has a variable CH period that is calculated upon the residual energy of the node (Sánchez López et al. 2008). Also, only MoCoSo and Onodera and Miyazaki do not require any global information while still providing multi-hop routing (Sánchez López et al. 2008, Onodera and Miyazaki 2008). MoCoSo integrates a new hierarchical routing mechanism, called Sequence Chain, that uses the addresses of the nodes to perform nearly zero cost routing along the addressing tree. Although a similar technique is employed by Onodera and Miyazaki, their protocols do not actually implement a clustering mechanism, but rather a tree formation algorithm in which parents are chosen and reconfigured according to their residual energy.

Protocol	**Type**	**Variable CH Period**	**Conditioned election**	**Synchronisation**	**Global information**	**Multi-hop**	**Complexity**
DMAC (Basagni 1999)	MANET	No	Weight	No	Yes	No	O(n)
WCA (Chatterjee et al. 2002)	MANET	No	Weight	Yes	Yes	No	O(d+m+1) *
LIDAR (Gavalas et al. 2006)	MANET	Mobility	Energy+	No	Yes	No	O(n)
ANDA (Chiasserini et al. 2004)	MANET	No	Energy	Yes	Yes	No	O(nxc) **
Wu et al. 2001	MANET	No	Energy+	No	Yes	Yes	O(v+N[x]) ***
Liu and Lin 2005	WSN	No	Energy	No	Yes	No	O(n)
Onodera & Miyazaki 2008	WSN	No	Energy	No	No	Yes	O(n)
MoCoSo (Sanchez Lopez et al 2008)	WSN	Yes	Energy	No	No	Yes	O(y) ****

* d: number of direct neighbours; m: number of messages regarding the cluster-related status
** c: number of CHs
*** N[x]: number of neighbours of node x; v: total number of vertex in the network graph
**** y is the number of nodes that answer a CH election messages, y <= n

Table 7.1 Energy Considering MANET Clustering Protocols and Mobile WSN Protocols

We would also like to compare all the clustering protocols in WSN that, while not supporting mobility, they consider energy-efficiency in the election of the CHs. The reason for including this comparison is the novelty of these protocols in their use of residual energy as a CH election variable, which represents the current state of the art in WSN clustering. They also serve as the proof that mobility in WSN is hardly considered.

Table 7.2 shows a summary of our comparison, following a similar column distribution as Table 7.1. MoCoSo meets most of the desirable requirements for an energy efficient clustering protocol (Sánchez López et al. 2008). The most important advantage over all the other protocols is the abnegation of global cluster information. For example, LEACH, and all the protocols that derive from it, need to synchronise their communication with the CH, which can only be done by knowing all the cluster members. EDAC, being the only comparable protocol to MoCoSo in terms of variable CH period, needs to store and update in the CH the residual energy values of all the cluster members in order to choose a successor. Every node in GESC needs to store a graph of all the cluster members in order to elect the CH. Finally, in HEED, every sensor node needs also to store a list of "candidate" CHs every time that a cluster election is triggered.

Protocol	Variable CH Period	Conditioned election	Synchronisation	Global information	Multi-hop	Complexity
LEACH (Heinzelman et al. 2002)	No	None	Yes	Yes	Yes	O(n)
(Liang & Yu 2005)	No	Energy	Yes	Yes	Yes	O(n)
EECS (Ye et al. 2005)	No	Prob + Energy	Yes	Yes	Yes	O(n)
EDAC (Wang et al. 2004)	Energy	Energy	Yes	Yes	Yes	O(n)
HEED (Younis & Fahmy 2004)	No	Energy	No	Yes	No	Nit x O(n) *
GESC (Dimokas et al. 2007)	No	Significance	No	Yes	No	O(n x u) + O(n) **
MoCoSo (Sanchez Lopez et al. 2008)	Energy	Energy	No	No	Yes	O(y) ***

* Nit is the number of iterations defined beforehand
** u is the number of edges of the graph formed by the cluster nodes
*** y is the number of nodes that answer a CH election messages, y <= n

Table 7.2 Comparison of WSN Protocols

7.2.2 Software Agents

Agent Based Systems are an evolving software paradigm that strives to create software that can possess human characteristics, such as autonomy, adaptability, sociality, judiciousness, mobility and reactivity. Commonly cited definitions of computational agents found in literature are:

- Intelligent agents are software programs that continuously perform three functions: perception of dynamic conditions in the environment; reasoning to interpret perceptions, solve problems, draw inferences, and determine actions. (Hayes-Roth 1995)
- Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so, realise a set of goals or tasks for which they are designed. (Maes 1995)

From the above definitions, it can be gathered that for a software entity to be named an agent, it should maintain the following properties:

- Autonomy, which suggests that agents should operate without the direct intervention of external forces, and control over their actions and internal state
- A description of the current state of its environment; in order for it to perceive the state it is in. In the case that the environment consists of other agents, the agent needs to have "social-ability, i.e., an interaction protocol and language. An agent's social-ability might be collaborative, competitive or even antagonistic.
- Reactivity (reflex based agent) and/or proactivity (goal/utility based agent), meaning that they should respond to changes in the environment and exhibit goal-directed behaviour by taking the initiative and planning to reach its goals.
- Knowledge of how the agent's actions affect its environment, in order for reactivity and proactivity to happen.

Agents might also "learn" to improve their behaviour using feedback from its performance, evolve or self-replicate depending on the needs of a particular application. Agent design has been the focus of much debate in the study of artificial intelligence over the years. A good review of agent design is given in Russell and Norvig (2003).

Although there have been no attempts for agent integration within the Internet of Things, to date, software agents can greatly enhance the functionality of the core Internet of Things architecture in two ways: Firstly, user centric agents can enable the automation of user queries and alert users to any changes in specific items or trails. Users, if they wish so, can offload monitoring duties to a user agent and customise alerts to be sent to them. Secondly, product centric agents can enable the concept of intelligent autonomous products (i.e., things) to be integrated with the Internet of Things, and help bring the intelligent product concept alive, by enhancing services that the Internet of Things can offer to its users.

To date, we have seen many examples of Intelligent Products, which, at their highest level of intelligence, are physical objects coupled with computational software agents to pursue their goals. More formally, an Intelligent Product, defined by Wong et al. (2002) is the coupling of a product and an information based representation that (1) possesses a unique identification, (2) is capable of communicating effectively with its environment, (3) can retain or store data about itself, (4) deploys a language to display its features and requirements, and (5) is capable of participating in, or making decisions relevant to, its own destiny (Wong et al. 2002). For a recent review of intelligent product definitions please see Holmstöm et al. (2009). Although there has been many variations on this definition, and debates on what we expect from an intelligent product, the last decade saw examples of autonomous products that manufacture themselves (Bussmann and Sieverding 2001) and monitor themselves, ordering maintenance when needed (Brintrup et al. 2010). More primitive "intelligent" products have encompassed other parts of the product lifecycle, such as retail, service, and recycling, where products had no autonomy, but users gave decisions upon them using a combination of sensory data and decision support software. For a detailed review of the intelligent product research landscape, please see Brintrup et al. (2008).

We envisage that the connection to the Internet of Things will be the next step for intelligent product research, as the Internet of Things offers a powerful platform to connect products with other products and service providers. Using the Internet of Things, products can send updates on their status and service requests to their stakeholders. Intelligent products then can move into a mode where they autonomously and continuously look for ways to bring leverage to their owners and producers by maximising their lives in service. They optimise their production, configuration, search for replacement parts as well as find suppliers and negotiate with them. They can minimise their carbon footprint. They can promote themselves, advertise new services and alert users for service upgrades. When necessary, they can cooperate with other products to place batch orders, and compete with other products to acquire rare parts. They can report any faults to their producers and recycle themselves at the end of their lives. To enable this vision, there has to be a seamless, scalable, and lightweight integration of software agents within the Internet of Things.

7.2.3 Data Synchronisation

Understanding the Internet of Things as a new generation of the Internet, where more and more objects and things will be connected globally, it is to expect that the amount of data and information created and exchanged in this context will extremely increase. The data distributed in the Internet of Things can be stored in the objects themselves or in heterogeneous online repositories, and might exist in connected and/or (partially) disconnected environments. In order to maintain a coherent cross-infrastructure view of the object information, the synchronisation of data across the architecture components is necessary. Due to the complexity and pervasiveness of the Internet of Things architecture, it is envisioned that this synchronisation will be a big challenge, so services, such as data access on demand and data consistency, are provided.

Once the requirements of data synchronisation in the Internet of Things have been analysed, it is easy to find many similarities with distributed database systems. Bell and Grimson describe a distributed database as a logically integrated collection of shared data, which is physically distributed across the nodes of a computer (Bell and Grimson 1992). In the case of the Internet of Things, these data will be additionally distributed across autonomous and heterogeneous objects, adding even more complexity to the system. There has been a lot of research in distributed database systems during the last three decades. We believe that its results offer a valuable base for developing synchronisation requirements for the Internet of Things. The following requirements for distribute databases are outlined by Bell and Grimson (1992):

- Data Handling,
- Query Optimisation,
- Concurrency Control,
- Recovery,
- Integrity and Security.

Two additional requirements are added by Öszu (1999):

- Transaction Management,
- Replication Protocols.

These requirements need to be met in order to support efficient, secure and consistent data synchronisation in distributed databases, and can be set as key requirements for data synchronisation in the Internet of Things as well.

While older approaches designing distributed database systems, Bell and Grimson propose the use of a central instance (similar to a distributed database management system) to coordinate database activities, new approaches apply mobile agents without a specific master node. Such agents support distributed transactions and security tasks (Assis Silva and Krause 1997, Niemi et al. 2007, Krivokapic 1997). Assis Silva and Krause (1997) describe the agent-based concept as:

- Very suitable for supporting transactions processing in massively distributed environments,
- Very suitable for supporting activities in dynamically changing environments,
- Providing an adequate support for mobile devices,
- Fulfilling coordination requirements of different types of application.

Regarding the data itself, its synchronisation involves different types of information in order to ensure data consistency:

- Object data: information describing an object,
- Security data: information supporting access control to an objects information,
- Event data: information about an objects history.

Knowledge about the structure, syntax and semantic of object information is required to filter data that has to be synchronised. There are different approaches and standards providing such knowledge. Table 7.3 gives an overview about the related work in this area. It is not intended to be complete, but to give a summary of work that may support data synchronisation in heterogeneous, distributed environments, such as the ones found in an Internet of Things architecture.

Due to the distributed locations of object information in the Internet of Things, the network availability between all information resources is of a special interest. Suzuki and Harrison (2006) show different scenarios describing possible operations on RFID tags in connected and disconnected environments and the corresponding synchronisation operations required to update a central database managing tag data. The authors also introduce a proposal for a Data Synchronisation Protocol. Pátkai and MacFarlane (2006) also show a classification of data synchronisation scenarios.

Reference	Description	Related data
Bonuccelli et al. 2007	Clock synchronisation and global time	Event data
Cilia et al. 2004	Concept-based approach to provide content information	Semantic
Grummt 2010	Requirements for Item Information Services and in Discovery Services	All data and semantic
Canard and Coisel 2008	Scheme for key synchronisation supporting RFID authentication	Security data
Ray et al. 2000	Model of Semantic correctness	Semantic

Table 7.3 Related work in the area of synchronisation

As mentioned above, common approaches require a stable or at least partial network connection. The Internet of Things is envisaged to contain distributed heterogeneous databases, applications and services, which might be always connected, partially connected or even permanently disconnected. All of these com-

ponents will potentially receive new or updated object information, and, therefore, a new approach to secure data consistency in the Internet of Things has to be researched.

7.3 Assumptions and Definitions

The Internet of Things advocates the extension of the Internet infrastructure that we know today towards the inclusion of objects or *things* as information producers. For the sake of clarity, we could define these objects as manufactured items, whose information and state is relevant to some service or application that is connected to this Internet of Things and, therefore, to the users that make use of those services. Some manufactured objects may require power or certain computation or communication to fulfil their primary purpose. This is the case with electrical appliances or computing equipment. These objects might evolve to support the electronic hardware and software necessary to gain access to the Internet of Things infrastructure. Other objects, whose traditional purpose doesn't require power, computation or communication of any kind, will not be able to produce any information. Therefore, there is a need for devices that can be attached (and eventually embed) to them and produce information on their behalf. The capabilities of those devices will greatly influence the level of participation of those objects in the Internet of Things, the same way that the evolution of an electrical appliance towards the Internet of Things connectivity will influence its level of participation on it. For the rest of our discussion, however, we will assume that the participation level of any Internet of Things object is based at least in the following capabilities:

- A unique identity
- Ability to sense and store their condition. Condition is the status of an object obtained by interpreting the output of sensor transducers associated with it
- Ability to make their information (be it identification, condition or other attributes) available to external entities
- Ability to communicate with other objects
- Ability to take decisions about thereselves and their interactions with other objects

Since the Internet of Things enablement of objects without any previous power, computation or communication capabilities, is specially challenging, the rest of the chapter focuses on the challenges arising from them, although many of the discussion here can be applied to any Internet of Things object. We will therefore assume that the communication capabilities of the devices that represent the objects are realised over the air using radio signals. Therefore, all the protocols that will be discussed in this chapter assume wireless communications implemented by the devices that represent the objects to which they are attached. In the context of

wireless networking, an independent computing agent is generally called a "node". For this reason, we will call the devices that represent the Internet of Things objects "nodes".

The Internet of Things architecture needs to be supported by an infrastructure that connects all the architectural components. This infrastructure would have the current Internet as its core backbone, as the Internet is the most pervasive global computer networking infrastructure available today. The devices attached to the objects mentioned above would need to connect to the infrastructure in some way. Some argue that on an Internet of Things, the things themselves need to be connected directly to the Internet. However, although the IETF and other global organisations are working on embedded Internet protocol stacks, such as the 6LoWPAN or the ROLL, a generic Internet of Things architecture should not require these kinds of capabilities (Kushalnagar et al. 2007, Vasseur et al. 2010). A good reason for this is the already existing great number of legacy networking protocols that cannot be adapted to work directly on top of IP stacks (e.g. mobile phones, proprietary WSN systems). Another reason is that many of the low cost devices that could create a really pervasive Internet of Things cannot support even the lightest of the proposed embedded Internet protocols (e.g. RFID tags, low cost WSN). In this chapter, we assume that local networks of objects can communicate with the Internet of Things infrastructure transparently, either directly with the support of IPs, or via gateways that can translate legacy protocols to the ones used on the Internet. Many times we will refer to "infrastructure gateways", meaning the computing devices that serve as bridges of local networks to the infrastructure. Those bridges may or may not provide translation services.

Following the definition of object and their characteristics above, along this chapter we will also assume that objects can create networks with other objects. We will also refer to the clustering capabilities of these networks of objects, where clustering is a particular mechanism for organising the objects into networks. Generally speaking, a network may contain several clusters, and certain elected members of those clusters communicate among each other creating a certain hierarchy. It would also be possible to create further clusters with these elected members, creating a double clustering network architecture. For example, an elected member of the cluster elected members could be chosen to communicate with the infrastructure gateway, creating a single elected representative for the whole network and, therefore, for all its clusters and objects. For simplicity, in this chapter we will focus in a single clustered network, and will use the terms "cluster" and "network" interchangeably. This assumption does not limit the discussion, as the same concepts could be applied if several layers of clustering would be considered.

Finally, in an Internet of Things context, the words "objects", "things" or "products" are often used interchangeably. In this chapter, we will use any of the aforementioned words to refer to the "things" of the Internet of Things. Conversely, the words "intelligent" and "smart" are used extensively in the same context to denote the capabilities of those things to process information and to make

informed decisions that influence the objects life and that of its surroundings. We will use any combination of those words to refer to emphasise the computation and reasoning capabilities of the Internet of Things objects.

7.4 Clustering for Scalability

7.4.1 Clustering Principles in an Internet of Things Architecture

Objects such as goods, product parts, assembly machinery, logistics and transportation items (e.g., pallets, containers or vehicles), warehouses, retailer's facilities or end-user assets are eligible for condition monitoring and can provide valuable information for themselves or other objects in their vicinity. In order to monitor their condition, embedded devices with wireless communication capabilities could be attached to them, becoming a part of the object, the same way a barcode sticker is part of the vast majority of today's products.

WSN are excellent candidates for becoming the devices attached to the objects of the Internet of Things, because many of its principles of operation address the Internet of Things requirements. These requirements include the clustering needs and the assumptions presented in section 7.3. Nevertheless, there are a number of differences between the "traditional" WSN and the devices that we propose will represent the Internet of Things objects. The main differences include the lack of a standardised unique identification scheme, the assumption of static deployments, the assumption of centralised base stations and the inflexible topologies that WSN are usually constructed upon.

Common WSN features include multi-hop communication, cooperative applications and events triggered inside the network. These are characteristics of active (as opposed to passive) networking. A clustering design for the Internet of Things requires the use of active networking to create collaborative, multi-hop and always-dynamic interactions among objects, which are equipped with wireless embedded devices as mentioned in section 7.3. This strategy extends the paradigm on object information gathering, since now it is possible not only to communicate the status of more than one object at the same time, but also to trigger the reporting of information in a bottom-up approach with no need for external control (i.e., there is no need for readers to initiate the reporting process, as is the case in passive RFID). What is more the ability to forward messages inside the same networks and provides the opportunity for distant objects, which were previously unable to reach the reader in a single hop, to report their status information to the system.

Active networking also creates the possibility of extending the information of an object by using other nearby object's information to enrich its status. To maxi-

mise the potential of these attributes, it would be beneficial to design a data structure model which organises the information of all the objects. The maintenance of such structure would be rooted on events originated at the active network itself, providing a real-time repository of network and object information. This structure would also provide the basis for sharing real-time object information among several information consumers. The type of information stored in this online repositories, as well as its synchronisation with the objects' real-time data, is a topic that we address in section 7.6.

One of the most important limitations of the devices attached to objects is power: since the devices are not powered by readers but by batteries, every action in which the device is involved, such as sensing or using the wireless transceiver, consumes part of its energy. For this reason, the protocols that manage node communication and networking must be carefully considered, since these devices are expected to function for months or years with the same battery charge. Clustering can be used to manage the power of the devices that represent the objects in the Internet of Things networks. In essence, clustering extends the network lifetime by electing a representative network member, or CH, which collects all the communication within the network and forwards it to the outside (so-called data aggregation). CHs consume more energy than the rest of the network members and their role must be periodically rotated in order to avoid the premature exhaustion of their battery power. The rotation of each CH is realised through an election process which computes the "best" candidate for the next period. This election may consider the particular *static* capabilities of each node (e.g. longer radio range, computing power), as well as its current *dynamic* status. One of the most important dynamic attributes of a node is its current residual energy. The election of the best candidate is paired with the decision of how long it will remain as the new CH. In the same way, the election itself is based on dynamic and static information about a node and the time that a node will have. The role of CH can be static (i.e. always the same time) or dynamic (i.e. a different time for each CH election), depending on the attributes of this specific node.

The election mechanism should be a distributed decision via collaborative messaging among all the nodes of the cluster. Centralised solutions cannot provide the scalability features that the Internet of Things should encourage, specifically when networks and clusters may be formed of hundreds or thousands of nodes. The election mechanism should also be dynamic, in the sense that changes on the network (e.g. the election of a new CH) should not be limited to static events, such as the exhaustion of an CH's representation period, but should also be triggered by unpredictable changes, such as the addition of new objects to the group or the removal of a group of objects that were part of a particular cluster. The need for the support of dynamic operation is a result of the differences between WSN and Internet of Things architecture outlined above, especially due to the mobility of things.

In this section, so far, we have outlined how the principles of clustering can provide benefits regarding the management of resources inside Internet of Things ob-

ject networks. However, clustering is a general strategy with many dimensions, and the protocols that manage the clustering mechanisms have to be tailored to the specific challenges and needs of the Internet of Things architecture. The rest of this section is dedicated to provide a number of design guidelines based on this early evaluation.

7.4.2 The Role of Context

The clustering of autonomous objects into groups requires similarities between the participating objects, which sufficiently confine the cluster groups from each other. For the detection of such similarities the autonomous objects always require the best available up-to-date information to arrive at a substantiated and aim-oriented clustering decision. That information can arise from two different sources: out of the physical environment of the object (e.g. other objects, infrastructure gateways, environmental parameters) and/or by connecting spatially separated resources (e.g. central databases) across the Internet of Things. Due to the possibility of disconnected environments, in which object networks may temporally loose connection to the Internet of Things infrastructure, the lack of knowledge about the objects environment and its surrounding situation as well as the systemic objective of robustness, a central clustering authority is impractical; clustering decisions require the objects' direct involvement and depend on the objects' own information, especially the objects' context and their capability of context awareness. In the context of ubiquitous and pervasive computing, there are several definitions for the term of context (awareness) (Crowley et al. 2002, Dey 2000, Schilit et al. 1994, Brown et al. 1997, Ryan et al. 1998). We will use the following definitions by Dey (2000) and Schilit et al. (1994) to set the basis for the rest of our discussion about context:

> "Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." (Dey 2000)

> "A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task." (Dey 2000)

> "Such context-aware software adapts according to the location of use, the collection of nearby people, hosts, and accessible devices, as well as to changes to such things over time. A system with these capabilities can examine the computing environment and react to changes to the environment." (Schilit et al. 1994)

The usage of environmental knowledge enables contextual clustering, which does not necessarily depend on a comparison of predefined characteristics or attributes of the participating objects (e.g. shipment destination); on the contrary, it is based on the object's situational information and status. As an example, consider the clustering process of objects depending on the residual energy of the sur-

rounding objects, the duration of the proximity between them (i.e., neighbourhood) or the degree of similarity between the objects' tasks. Hence, contextual clustering offers an access to self-categorised object groups over the Internet of Things; it is driven by a contextual rather than a process perspective (e.g., process oriented package flows). The utilisation of situation-dependent attributes enables a precise and useful clustering and allows a human-like understanding of the objects' situations. However, a disadvantage of a pure application of the contextual clustering conceptualisation is the uncertainty about the rate of change of the context, namely, the time-dependent validity of the context and the differentiation between the long-term and short-term validity of the context. This time, dependency could create problems with a high rate of changes on the context surrounding a particular group of objects, triggering a high number of re-clustering processes, which will incur in the use of too many network resources. Additionally, there is the challenge of the context awareness itself, since it would be desirable for the objects to possess a generic context analysing power not limited to specific parameters (e.g. fuzzy logic, complex event processing). This would also results in high demands in terms of the computing power and, consequently, in high energy consumption. A compromise could be a hybrid clustering approach, using the objects' characteristics as well as the predefined objects' context. Hybrid clustering could reduce the re-clustering effort while partly benefiting from situation-dependent clustering advantages. In dynamic and mobile scenarios, such as those encountered in the Internet of Things, the role of context for clustering should be considered, since it would bring significant contribution for precise and efficient clustering.

7.4.3 Design Guidelines

The development of clustering algorithms for physical objects involves a number of components and protocols. Some of them are necessary for the logical infrastructure within the clusters and serve as the basis for other services and applications to build upon. These necessary components, which are fundamental for an efficient clustering process, include the CH election process, a suitable addressing scheme and an efficient routing procedure.

CH Election

This chapter proposes for the Internet of Things device networks to utilise clustering for power management and, therefore, for scalability. Although a detailed description of clustering and its benefits was presented in section 7.4.1, let's recall that the main objective of clustering is to extend the object network lifetime by electing a representative network member which collects all the communication within the network and forwards it to the outside. This section outlines the CH election mechanism that an Internet of Things device, representing an object,

would take as part of a clustered network of devices. As an important and powerful feature of this design, CHs are elected according to their residual energy.

Not every device would be eligible at any time to become a CH. It is possible for certain devices to be in range with an infrastructure gateway while some other remain "hidden" or out of range. Apart from the energy efficiency requirements, the CH election procedure should also avoid choosing a CH which is hidden while another CH from the network is in range of an infrastructure gateway. To address this issue, infrastructure gateways could send advertisement packets to announce their presence. Only CHs that receive an advertisement would participate in the CH election process.

Let T_{CH} be the duration that a node will have, once the CH role has been elected. T_{CH} could be calculated as a function of the node's residual energy:

T_{CH}= C x Residual Energy, where C is a constant

In each CH, a node would calculate its own proposed T_{CH} according to the above equation, and would send its proposal to the rest of the cluster members. A consensus decision would be made, and the selected node would become the new CH for the time T_{CH}. Although the factors that would elect a new CH could vary, a straightforward decision would select the proposal with the highest computer T_{CH}, since this would minimise the number of CH election procedures and, therefore, conserve more energy over time. A random delay, also function of the node's residual energy could be introduced to avoid collisions in the wireless channel when a big number of nodes are in the same cluster.

A CH election procedure would start when any of the following situations occur:

- T_{CH} expires
- The current CH cannot communicate with any infrastructure gateway
- A CH, whom did not participate in the previous election and has more residual energy than the current CH, receives an advertisement packet from a gateway
- A CH cannot communicate with the current CH before T_{CH} expires
- A new object is added to the CH's network.

According to this, if a network loses its CH and no CH receives an advertisement packet from a gateway, the election process would not start again to choose a new CH. This situation is undesirable, because the condition information of the associated objects may still be useful locally (e.g., for storing it in the node's memory for a later synchronisation – see section 7.6). Moreover, it is not practical to reject new associations due to temporal disconnections. To avoid this problem, the CH election procedure could be started by any node which runs for more than a certain amount of time without being able to communicate with its CH. When the connection with the information infrastructure is re-established, networks with a CH selected in this way would start a regular CH election procedure again.

Cluster Membership

The above mechanism to select a CH would take place inside a cluster of nodes. But how do the nodes decide to become part of the same cluster?

In order for objects and networks to find other objects and networks, one or several nodes could send periodic discovery broadcast packets. Nodes receiving these packets would process the packet information and decide to become part of the same network or cluster by sending back a response packet. The results would be communicated to all the network members and a new CH election process would begin. This process could involve several objects and networks at the same time. We could call the process "association" by which multiple objects and networks join together to form a unique cluster.

```
Association request reception
-------------------------------------
Input: List of association requests
Output: Association response, update to local network

if (list of requests NOT null)
      HAL = list of requests from networks with highest number of nodes;
      if (I am in HAL)
          #Nodes: #Nodes + #Nodes in the list of association requests;
          Initiate CH election
          Send association response to the list of association requests;
          Update my network;
      end
end

Association response reception
-------------------------------------
Input: Association response
Output: Update to local network

According to the association response...
      change my network ID;
      change my address;
      update #Nodes;

Update my network;
```

Fig. 7.1 Algorithm for Global Knowledge by Localised Association Procedures

As presented earlier, the objective of object clustering is not only to manage the energy resources of the network, but also to form collaborative groups of objects that share a common situation or purpose. For this reason, we could propose an as-

sociation procedure which considers aspects of the object's nature in order to filter and classify potential object interactions before they occur. This association procedure would therefore have two phases: The first phase, in which association requests would be organised and filtered, and a second phase, in which the final association procedure would take place, and which would include the update of the cluster attributes, such as the election of a new CH or the re-factoring of routing addresses.

Association requests could include information not only about static attributes of the objects (e.g. ID, address, network), but also dynamic and contextual attributes. We could divide these attributes into two groups, regarding the need for their presence in order for the analysis of the requests to proceed. *Mandatory* attributes would need to be held by both, the request sender and the receiver, while *optional* attributes would not be a pre-condition for the association procedure to continue, but rather they would add up to the decision on membership of the requesting object. At the same time, due to the heterogeneity of the Internet of Things, attributes coming from different objects might not always be totally compatible, and a certain degree of fuzziness in the extraction and comparison of association request attributes would be necessary.

While the comparison of static and dynamic attributes could be relatively easy to implement, contextual information is more difficult to compare. For objects involved in different contextual situations, the contextual attributes might mean different things. For example, for objects involved in a shipment, the shipment's destination would be the most important contextual attribute. For objects stored in a warehouse, the delivery date could be more important than the destination. This kind of contextual clustering could be realised, for example, by defining priority decision rules for different types of autonomous objects depending on possible locations. However, this methodology would just take the 'meta-context' into account; but there would be as many contexts as objects are present in the situation, deduced from the individual object perspectives. Even if the membership decisions could be limited to the object's location, the 'meta-context' of the situation, the combination of present objects, their internal status and the integration of context into membership decisions would still be a challenge for the computing capabilities of embedded devices. We could envision that this first phase of the clustering mechanism would grow in complexity as the capabilities of the object's embedded devices would increase, from just location-based prioritisation to the analysis of complex statistical data and rules. However, it would also be necessary to evaluate the complexities of such algorithms against the quality of the resulting clusters, since any complex processing would have important repercussions on the energy expenditure, the scalability and the robustness of the resulting clusters.

In wireless networks, a network identifier is usually selected to distinguish a particular network from the others. If clusters of objects are built in a meaningful and contextually rich manner, this identifier could also provide a useful hint on the 'theme' of the cluster. The objects from the Internet of Things are likely to have unique identifiers which follow a meaningful encoding. Examples of this encoding

can be found in the Electronic Product Code set of standards (Armenio et al. 2009). When a cluster is updated with the addition of new members, decisions should be made on which identifier would represent the resulting cluster. This decision could be taken in a second phase of the association procedure. Another important task of this phase would be the assignment of local addresses for communication and routing. Some guidelines on the design of an addressing scheme will be given later in this section.

Many clustering mechanisms, as well as other network-wide operations, may require the knowledge of the number of nodes of the network, or the number of nodes in particular parts of the network, such as an address branch for hierarchical addressing. Often, this knowledge is considered as a 'global' attribute, since only a 'global viewer' would have access to the information of every single node in the network. In distributed networks, such as the one that we are describing in this chapter, it is however possible to set mechanisms in place that will keep every node updated of global attributes with little extra processing. In the case of the number of nodes of the network, the association procedure could keep track of them with the assumption that every network starts up with a single node, and that successive association procedures are build up to create large clusters and networks. When a particular attribute needs to be equal network-wide (e.g., the network ID), this global knowledge could be very useful, for example, to decide which party in an association keeps its attributes and which one will have to update them. Figure 7.1 presents a simple algorithm demonstrating how this global knowledge can be obtained from localised association procedures. This algorithm can be made more complex by taking into consideration issues such as:

- Calculation of addresses
- Hierarchical node structures (e.g., which node will become the parent and which the children)
- Calculation of the number of nodes in specific hierarchy branches to decide, for example, which branch will have to re-assign its addresses
- Changes on the direction of the parent-children relationships due to network mergers.

A disassociation procedure would need to be undertaken if an object or group of objects leave the network. This process would need to update all the attributes presented before, such as the number of nodes per network or the addresses of the nodes. The algorithms would also need to include provisions for re-merging a network whose routing structure was broken, the selection of a new network identifier if the previous one is not representative enough after the disassociation, the election of a new CH if the previous CH left the network, etc.

Addressing and Routing

Dynamic address allocation is a common problem for wireless ad-hoc networks where mobile nodes constantly join and leave the network. Unlike wired networks, which lack of strong power and infrastructure constraints, mobile networks

must optimise their operation and keep the connectivity even when unexpected topology changes occur. Wireless Sensor Networks pose additional challenges due to their especially scarce resources.

Due to the complexity of fitting a full protocol and application set into the sensor nodes, WSN dynamic addressing has been somehow left aside in favour of other research areas, such as routing, MAC layer design, synchronisation protocols, etc. However, the recent interest in the community to design networks that can adapt to the environment is forcing to reconsider all aspects of WSN autoconfiguration and maintenance, including dynamic addressing.

One of the main challenges in the mobile object networks of the Internet of Things is the support for dynamic associations of objects. The purpose of considering dynamic association of objects is to provide a flexible solution in which the network members do not need to be known in advance. In this way, a great variety of applications can fit into the architecture without the obstruction of an inflexible design. The networks resulting from object interactions are likely to be relatively small: a group of boxes or pallets in a freight, robots in an assembly line, containers in a cargo bay, parts of complex assets, such as machinery or vehicles, etc. However, it is also likely that the shadowing effect provoked by objects in the environment would prevent every object to reach the network CH in a single hop. Furthermore, larger networks are also viable, such as those in big warehouses or retail shops. For these reasons, we would like to consider a multi-hop addressing and routing protocol, simple enough to adjust to the object's embedded devices constraints but fulfilling all the requirements of the Internet of Things networks. We devise the following requirements for an Internet of Things addressing scheme:

- Addresses must be unique inside a network
- Addresses must be reused when objects leave
- Addressing must be dynamic. Address assignation should be fully distributed
- Addressing must be scalable
- Support for network merge and split should be provided
- The protocol overhead must be minimised

To meet these requirements, the following list summarises the ideal properties that an addressing scheme for the Internet of Things devices should have:

- *Hierarchical*: The nodes involved in this addressing scheme are the devices that represent objects. Nodes receive addresses organised in a tree structure. When two or more objects associate, the object that computes an address becomes a parent. Hence, senders of association requests become children.
- *Distributed and unique:* Each node should be only responsible for assigning addresses to its children. The address a child receives should be derived from its parent address in a way that makes that address unique for the network.
- *Scalable:* If a node leaves a group, its address should become automatically available for any other node joining with the same parent. Moreover, the ad-

dresses should not be limited in size by the scheme, but rather increase their size as the network becomes bigger. Network merges should also be supported by reassigning the addresses of the network with the smallest number of nodes.
- *Low overhead:* A parent should only need to know its immediate children to assign addresses in a unique manner. The addressing scheme should provide routing along the tree with nearly no cost. A node could know how many hops it is away from any destination by just analysing the address, and parents could route packets following the tree by just comparing its address with the packet's destination address. Hierarchical assignation of addresses should allow parents to provide shortcuts to the destination in an equally simple way by routing packets to neighbours that are closer to the destination than following the tree.
- *Extensible:* The addressing scheme should provide mechanisms to allow an unlimited number of children per parent even if the original limits for address space assignation per parent have been reached.

An addressing scheme that fulfils all of these properties was proposed by Sánchez López et al. (2010) as part of a distributed protocol for the management of resources inside Smart Object networks (Sánchez López et al. 2008).

7.5 Software Agents for Object Representation

The Internet of Things envisages a global architecture in which objects become first class citizens of the Internet, and therefore they are not only able to report their status to human users via computing infrastructures, but are also able to communicate with each other and other Internet of Things components in order to influence their own destiny. Although the Internet of Things can potentially serve any type of objects, commercial products and assets constitute one of the main drivers for its conceptualisation. On this note, we would like to discuss the influence of products in the Internet of Things vision, and use this discussion to introduce the role of software agents for object representation. We start this discussion by listing the communication scenarios that might typically occur in a product lifecycle:

Product to product communication:

- Products that request service, asking other products for batch orders
- Problem co-diagnosis, where products of the same firm consult each other for undiagnosed failure modes

Product to supplier communication:

- Products asking service provision (including recycling, maintenance, scrappage, and logistics) from suppliers with a given service request, time, and price
- Manufacturer communicating product upgrades or recalls to products

- Products communicating performance data to manufacturer

Product to user communication:

- Product performance and actions
- Product location and state
- Upgrades, promotions and additional services

The Internet of Things may benefit from the automation of the above communications and actions between product and user (suppliers, manufacturers and owners). Computational agents provide us with a suitable abstraction as well as practical tools to carry out this task. Agents could be used to take on mundane monitoring tasks from a human user, such as the monitoring of a set of products travelling across a supply chain, the arrival of products on a specific location, the divergence of a product from a pre-specified path, the health, expiry or maintenance actions of a product and so on. The human user shall be able to preconfigure queries and subscribe to alerts on these queries. In addition to monitoring users, we might encounter the need for service providing agents in the Internet of Things. These could be organisational agents that offer maintenance or logistics to products, for example.

The Internet of Things would require the automation of data gathering and analysis on the "things" by making the things themselves responsible through the intelligent product paradigm. Hence we need to examine ways in which objects can characterise themselves, communicate with others and automate their actions. One suggestion might be a Service Oriented Architecture (SOA), which refers to a design paradigm where various services can be loosely coupled and accessed via "Web Service protocols", such as XML, SOAP, WSDL, etc. This method promotes a service view rather than a product based view. On the other hand, academic literature and industrial frontrunners in the area argue that a product centric view is an intuitive one that distributes risk and reduces bottlenecks (Brintrup et al. 2010). Given that multiple organisations and objects will use the Internet of Things architecture throughout a product's lifecycle, it is important that a scalable and interoperable architecture is proposed, which reduces reliance on centralised databases and processing. An important point here is to aim for a generic architecture that can be used in as many product lifecycle scenarios as possible. Having solely an SOA-based architecture could bias the architecture towards sensor nodes that require high processing power and memory in order to wrap messages that are compatible with web services. This would have a cost implication and, therefore, bias the use of the architecture to complex high value products, and is consequently unfavourable.

Agent based systems and associated technologies, such as object-oriented, peer-to-peer and service-oriented architectures, have matured to the point where intelligent products can leverage their potential. An agent oriented viewpoint provides an intuitive encapsulation to the intelligent product, while being practical. The approach allows complex decision making without having to go through

many architectural layers. Recent advancements in agent-based open software, such as Open Source Cougaar and JADE (COUGAAR 2010 and JADE 2010), point to synergetic environments where SOA principles work in harmony with those of agent-based systems. The Internet of Things shall therefore aim to provide an architecture that will make use of the strengths of both, SOA and agent-based systems. Intelligent reasoning can then occur at each level of the system to reduce overall system load and increase responsiveness, while SOA principles, such as modularity, reuse and abstraction, will be exploited.

The current thinking in the area points to the use of an architecture that will allow agent characterisation using the Ontology Web Language, agent definition via re-usable XML based plugins, a discovery service for finding the requested service, followed by a one-to-one interaction between the provider and client, similar to SOA. For instance, finding supplier user agents through yellow pages and then negotiating with them on a one-to-one basis may well follow this procedure. There may also be instances where this exact procedure is not required, for example, when products need to communicate to one another to arrange batch orders, negotiate scheduling, or learn from one another for co-diagnosis. There may also be complex decision making at the object level, such as deciding the next step of production or which sub-components to recycle. Having agents representing objects on the network and processing these decisions would make the architecture work faster and be more applicable to a large number of scenarios. Since we might potentially have millions of product agents on the Internet of Things, the product characterisation, data storage and communication protocols shall be as lightweight as possible. Here, clustering can play the important role of increasing scalability, and the same concepts applied to the clustering of physical devices can be extended for clustering of agents. In fact, with appropriate and accurate-enough information, many of the clustering burdens could be relegated to the agents themselves, which would reside in parts of the architecture with much more readily available computing resources than those of the embedded devices in the objects themselves. Decisions taken by the agent system would then be synchronised with the physical world, providing a balance between scalability, resource management and real-time information.

7.6 Data Synchronisation

7.6.1 Types of Network Architectures

Data synchronisation in the Internet of Things depends on the availability of connectivity within the network architecture in which the objects are moving. There are three types of architectures that have to be examined:

- Connected architectures (Internet),
- Partitioned architectures (Intranet, Extranet),
- Disconnected architectures (local resources).

Connected Architectures
The Internet is a globally distributed network of computers carrying many services and information resources. It is assumed that resources are always connected to share and update information. Objects moving in a connected architecture do not need a special synchronisation method, because object information can be updated at any place inside the network in real-time (see Figure 7.2).

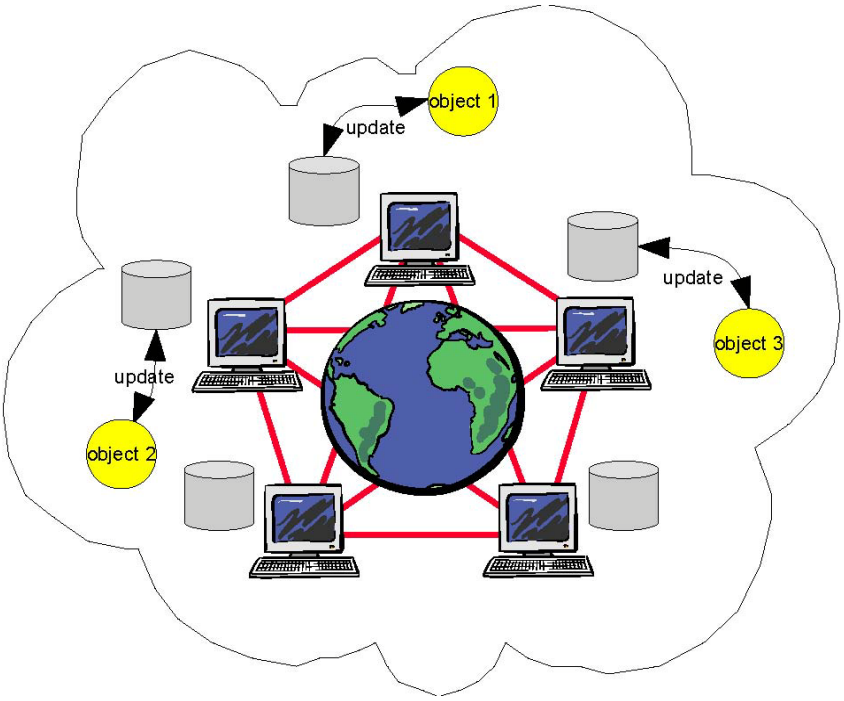

Fig. 7.2 Internet Architecture

The EPCglobal Architecture, as a proposal for implementation of the Internet of Things supporting the logistic supply chain, is based on this connected architecture (Armenio et al. 2009). Objects may be identified by the Electronic Product Code while related information is stored in distributed repositories. Capturing and accessing object information requires a stable network connection.

Partitioned Architectures

Intranet and Extranet may be seen as types of partitioned networks. They base on the same technology as the Internet, but they are only reachable inside a closed area (Intranet) or with a special authentication (Extranet). Beside partitioned networks, there are partially disconnected devices, such as mobile devices, which are disconnected while updating object information (e.g., for maintenance). Data will be updated at the mobile device first and has to be synchronised to related network information resources when the mobile device will be connected again. Synchronisation is similar to those supporting objects moving between partitioned networks. But there is a new complexity, because a mobile object may be connected to the network before data is synchronised between the mobile device and the network resources (see Figure 7.3).

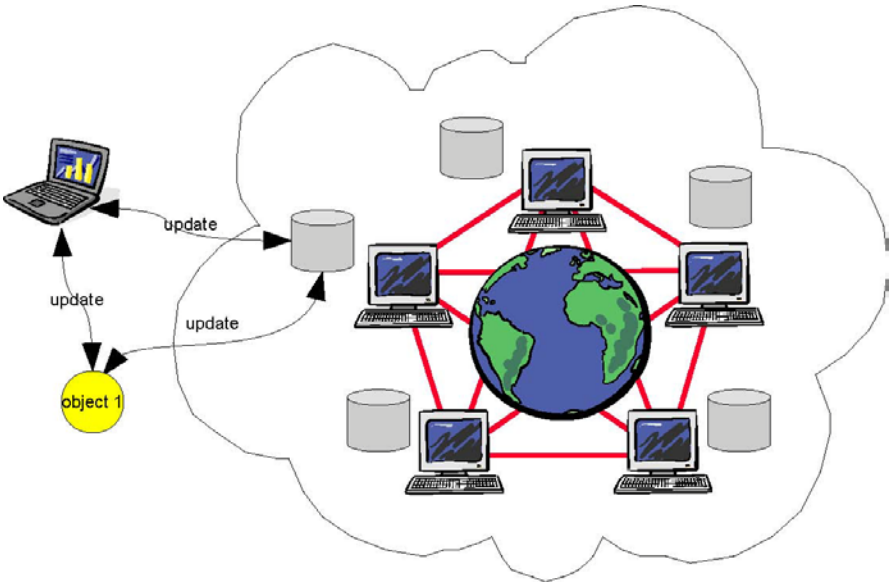

Fig. 7.3 Partitioned Architecture

Objects moving in partitioned networks may act the same way as objects inside the Internet: Objects move within the physical world while information is generated and exchanged within information networks. Information about an object might be generated while it has no reliable network connectivity (e.g. accumulation of sensor data). Therefore, it is necessary to have a robust mechanism to synchronise such information updates to the network when connectivity is available, in order that the data can be available within a single company and also shareable with other organisations. Other challenges for data synchronisation to be considered, are synchronising data (e.g., configuration instructions) from the network to an object that only has intermittent connectivity, or interpreting the current 'state' of an object correctly, when some of the event information that should contribute to the state of determination arrives late, out of sequence, or not at all.

Disconnected Architectures

Many things that interact with the Internet of Things may not have permanent reliable connectivity to communication networks. This may be caused by missing technical resources (e.g., in remote areas) as well as technical restrictions that may not allow Internet connectivity (e.g., in dangerous production areas). Nevertheless, there might be objects that have to be maintained inside such areas. Those things would need information about their life cycle without having permanent reliable Internet connectivity. By contrast to their special needs, many of today's Internet-based applications require a stable network infrastructure and routinely store object information into networked repositories.

A permanent disconnected architecture has to distinguish between objects and other resources without network availability. Objects may be disconnected, because they are unmovable or only moving in environments without network availability. Mobile devices could be used to exchange information between disconnected objects and other network resources. In the case of disconnected applications or repositories, information could be exchanged via mobile objects. Both cases are shown in Figure 7.4.

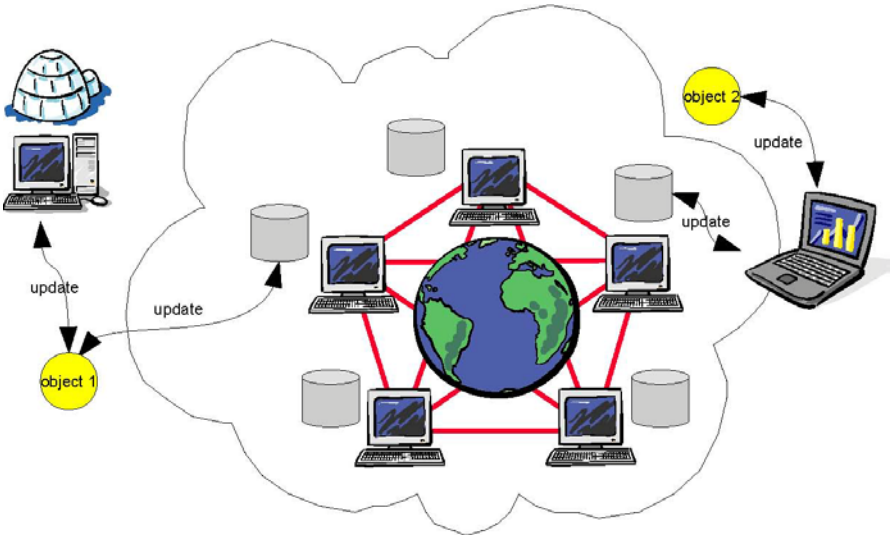

Fig. 7.4 Disconnected Architecture

Interfaces

The Internet of Things has to handle the information exchange in heterogeneous distributed networks with characteristics of all architectures described above. Therefore, it is important to identify all the interfaces for information exchange and to define synchronisation mechanisms to assure data consistency and security. Table 7.4 shows a summary of possible types of information exchange and the related time of synchronisation considering all architectures.

Architecture	Types of information exchange	Time of synchronisation
Internet	Real-time between object and connected resources	Real-time
Partitioned networks	By objects	Next connection
Partially disconnected	By objects and mobile devices	Next connection
Permanently disconnected (object)	By mobile devices	Next connection

Permanently disconnected (local resources)	By objects	Next connection

Table 7.4 Types of Information Exchange

Synchronisation in the Internet of Things has to handle these different types of information exchange. The autonomy of objects and the possibility to change data in (partially) disconnected environments causes difficulties with the serialisation of object data and the availability of consistent data at any location within the Internet of Things.

7.6.2 Requirements and Challenges

Requirements

The whole Internet can be seen as a huge distributed heterogeneous database (Niemi et al. 2007). The Internet of Things will increase that enormous database and add new possibilities of information exchange (see Table 7.4). Because of the similarities between the Internet of Things and distributed database systems it is necessary to research the technical requirements of distributed databases (see section 7.2.3, Bell and Grimson 1992, Öszu 1999, Leanvitt 2010) considering the architectural characteristics of the Internet of Things (see section 7.6.1).

Data Handling deals with the distributed allocation of data to the nodes of a computer network and the transformation of heterogeneous data. It has to solve problems resulting from the distribution of data as well as defining a common database description to make heterogeneous data understandable. Common distributed databases use a global master represented by a distributed database management system to organise data allocation. Most of them support relational databases using SQL to query data. There are different approaches to implement object-oriented databases, but they have not the acceptance of relational databases. Only during the last few years non-relational databases have reached increased popularity (Leanvitt 2010). NoSQL databases use different technologies for data handling. The most popular types are key-value stores, column-oriented databases and document-based stores. A well-known implementation is Bigtable by Google using the column-oriented approach (Chang et al. 2006). The main advantage of NoSQL databases is their better scalability, but they do not ensure consistency. *Data Handling in the Internet of Things* will add the problem of (partially) disconnected environments and autonomous objects. It remains unclear if it will be possible to keep only one global instance of object information and to manage distributed data, because the availability of data access cannot be assured permanently. Clustering will be an approach to support distributed data allocation without permanent network availability (see section 7.4). Data transformation will aim for harmonising existing databases like relational or XML databases and object

data, which is expected to be stored in a differing way. Object data will need efficient approaches for data management, because storage capacity will be limited. There is a need to define a common global language to understand all information resources within the Internet of Things.

Query Optimisation is necessary to provide efficient access to distributed databases which are affected by huge data resources and the possibility of unstable connectivity. Structural details of information access should be hidden from the user. The most common language for receiving and manipulation data is SQL.

Although SQL is very powerful and well standardised, it is focused on relational databases. There are also proposals for querying other resources, such as SPARQL, which is used for the Semantic Web, or the different implementations of OQL for querying object-oriented databases.

Query Optimisation in the Internet of Things has to support querying of data in heterogeneous structures and should provide methods to handle (partially) disconnected architectures. The structure of object data has to be examined considering query mechanisms as well as scalability.

Transaction Management has to organise the correct execution of database transactions, which are series of actions that have to be processed as single indivisible units. A transaction has four important properties:

- Atomicity (executing a transaction as a single unit),
- Consistency (transforming a database from one consistent state to another),
- Independence (providing execution independent from another transaction),
- Durability (making transaction results persistent in the database).

Those properties are known as ACID properties. There has to be a good concurrency control as well as a recovery system to provide them as a whole. While common relational databases follow the ACID approach, NoSQL databases implement a set of weaker properties named BASE (Basically Available, Soft-state, Eventual consistency). *Transaction Management in the Internet of Things* will not be able to implement all ACID properties, although they are essential to assure data consistency. It can be expected that they will not be achieved at any time and any place. Providing consistency and independence will require intelligent approaches of synchronisation, due to the data manipulation needs in (partially) disconnected environments. Software agents can be suitable to bridge this gap (see section 7.5, Assis Silva and Krause 1997). Furthermore, transaction methods of NoSQL databases have to be examined. The CAP theorem is also worth mentioning here. It states that it is impossible to provide consistency, availability and partition tolerance in a distributed system at the same time (Gilbert and Lynch 2002). Providing consistency in the Internet of Things will cause similar problems, because of the partitioned architectures.

Concurrency Control comprises different methods to ensure transaction management in distributed databases. It is concerned with scheduling and serialisation of transactions and offers different techniques of concurrency control. A transaction consists of a sequence of reads and writes. The entire sequence of reads and

writes by all concurrent transactions in a database is a local schedule, while such a sequence affecting distributed databases is a global schedule. Ordering all reads and writes of a schedule in a way that they can be processed sequentially one after the other means serialisation. To support concurrency control, Bell and Grimson (1992) distinguish three techniques:

- Locking methods,
- Timestamp methods,
- Optimistic methods.

Concurrency Control in the Internet of Things has to manage large amounts of data resources, which are distributed among common databases with permanent network availability, data resources in (partially) disconnected environments and autonomous objects. Therefore, scheduling and serialisation will find a new complexity in that context. Object information could be updated in different locations that may not be connected with each other. If those locations are partially disconnected, synchronisation would be possible during the next connection at an unknown time. In the meantime, object information could have changed again, which may cause problems for serialisation. If we assume that objects are moving, it could be suitable to support data consistency with a local object-oriented data management. As mentioned before, this solution could be supported by software agents. On the other side, there might be stationary objects which could not be supported by that solution. Instead, these objects would require a synchronisation of a partially disconnected database. The traditional mechanisms of concurrency control would also reach their limits.

Locking methods are not suitable in the context of the Internet of Things. Assuming that objects are working autonomously and data resources can exist in partially disconnected environments, the usage of locking methods could cause a large number of deadlocks that would prevent further synchronisation. Considering that updating object information could be possible at different times and in partially disconnected environments, the usage of a *timestamp method* could be suitable. Serialisation could be realised by timestamps of data updates. Nevertheless, the usage of such a method would require a global reference time to work correctly. Bonuccelli proposed the enforcement of a global clock and described it as a difficult task (Bonuccelli et al. 2007). Finally, *optimistic methods* are based on the premise that conflict is rare. Considering the disordered movement of objects in the Internet of Things, these methods could be hardly considered suitable in this context.

Recovery refers to the ability to ensure the consistency of data resources in case of unpredictable failures of hardware or software components. Like concurrency control, recovery is tightly linked to transaction management, because a transaction is the smallest recovery unit. Considering the ACID properties of a transaction, concurrency control ensures consistency and independence, while recovery provides durability and atomicity (Bell and Grimson 1992). Recovery requires a history of transactions to identify the point where to restart transactions after the

occurrence of an error. Such a history might be implemented by a log file. Corresponding to transaction management and concurrency control it has to be distinguished between local and global transactions in distributed systems. *Recovery in the Internet of Things* has to solve similar problems to the concurrency control. Especially working in (partially) disconnected environments may cause problems. The usage of an object-oriented history could be suitable to ensure the consistency of object data. Supporting recovery methods by a history would also require a global time reference.

Integrity and Security aim to avoid the corruption of data resources. *Integrity* tries to ensure the logical correctness of data. There might be local and global constraints to avoid the storage of incorrect data. *Security* mechanisms protect data resources from the access of unauthorised users. Depending on their special needs, users may have different views on the data resources. On this basis, they would need to provide identification and get authentication in order to access the associated data. Encryption of data is an additional approach to protect data if an unauthorised user gets access to a secured data resource.

Integrity and Security in the Internet of Things are of great importance, because many different users will request and manipulate many heterogeneous data resources. There is a need to guarantee that no data will be manipulated or destroyed by unauthorised users. Working in (partially) disconnected environments will be of special interest, because constraints and authorisation rules have to be known locally in cases where no network will be available.

Replication is the process of storing redundant data resources with the aim to support recovery methods and data availability in distributed environments. Redundant copies of the same data resource require methods to assure consistency among those copies. Öszu (1999) mentions the one-copy equivalence as an important consistency criterion. It requires that the value of all copies should be identical after a transaction. A typical replication control protocol is the Read-Once/ Write-All (ROWA) protocol. *Replication in the Internet of Things* is not expected to be a suitable approach. As discussed earlier, there is a potential for various non-homogeneous copies of object data across the architecture, due to the manipulation of data in disconnected environments and the usage of autonomous objects. It is expected that data might not be in the same state at every point in time. It will therefore not be possible to support one-copy equivalence. Nevertheless, experiences from replicated databases could be applied to use different copies of object data (although not always in the same state) to improve data availability in partially disconnected environments as well as recovery in the Internet of Things.

Challenges

From the perspective of data management, the Internet of Things can be considered as a heterogeneous distributed database with the following special architectural characteristics (see section 7.6.1):

- The existence of (partially) disconnected environments
- The handling of (mobile) autonomous objects

Following this idea, the Internet of Things will need a data management strategy providing the same tasks as the data management of distributed environments, but considering these special needs. Section 7.6.2 gave a summary of all those tasks and identifies the special requirements in the Internet of Things. As a result, the following topics would need further investigation before data synchronisation can be introduced in an Internet of Things architecture:

- Advantages and disadvantages of global and shared data management in (partially) disconnected environments,
- Defining a global language fitting the needs of common data structures and object-associated data structures,
- Methods for transaction management, concurrency control and recovery in environments not being able to achieve the ACID properties,
- Implementation of a global reference time in (partially) disconnected environments,
- Guarantee of data integrity and security in (partially) disconnected environments.

We can therefore conclude that finding intelligent solutions to overcome these challenges is a precondition for providing efficient and secure synchronisation in the Internet of Things.

7.7 Summary and Conclusion

Among the many challenges in the realisation of the Internet of Things vision, many times the management of the resources of the embedded devices that will power the Internet of Things objects is overlooked. In this chapter, we have discussed three techniques that will assist these constrained devices to empower the Internet of Things services for extended periods of time, while providing the objects with enhanced capabilities that positively influence the collection of object information.

The clustering of the Internet of Things objects will support the networking of autonomous intelligent objects by influencing their lifetime, scalability and robustness. By considering both, the energy of the devices as well as their context, the presented techniques can not only ensure that the objects will be able to produce information for longer periods of times, but also that only those objects involved in the same contextual situation will cooperate and share information. The use of software agents can add up to these benefits by taking representation of both, the physical objects and the Internet of Things users, in situations requiring automation and objective decision making. Examples of these situations are the monitoring of products in supply chains, the management of procurement processes for objects representing commercial products or the execution of periodic queries for object information in networked databases. At the same time, software

agents can aid the clustering processes by moving part of the decision making process to the architecture side, reducing the burden of the embedded devices attached to the Internet of Things objects. Finally, object data synchronisation will be needed in order to cope with (partially) disconnected environments where objects are not permanently connected to the Internet of Things infrastructure. While many lessons can be learned from the research in distributed databases, the unique requirements of the Internet of Things will call for a new set of solutions in areas such as data integrity, transaction management, concurrency control or a global language for object information management.

References

Armenio F, Barthel H et al. (2009) The EPCglobal Architecture Framework. http://www.epcglobalinc.org/standards/architecture/architecture_1_3-framework-20090319.pdf. Accessed 11 June 2010

Assis Silva FM, Krause S (1997) A distributed Transaction Model Based on Mobile Agents. In: Rothermel K, Popescu-Zeletin R (eds) Proceedings of the First International Workshop on Mobile Agents. Springer, Berlin-Heidelberg

Basagni S (1999) Distributed Clustering for Ad Hoc Networks. Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks, Fremantel

Bell D, Grimson J (1992) Distributed Database Systems. Addison Wesley Publishers Ltd.

Bonuccelli M, Ciuffoletti A, Clo M, Pelagatti S (2007) Scheduling and Synchronization in Distributed Systems. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.39.509. Accessed 9 June 2010

Brintrup A, Ranasinghe D, McFarlane D, Parlikad A (2008) A review of the intelligent product across the product lifecycle. Proceedings of the 5th International Conference on Product Lifecycle Management, Seoul

Brintrup A, McFarlane D, Owens K (2010) Will intelligent assets take off? Towards self-serving aircraft assets. IEEE Intell Syst. doi:10.1109/MIS.2009.89 (In Press)

Brown PJ, Bovey JD, Chen X (1997) Context-Aware Applications: From the Laboratory to the Marketplace. IEEE IEEE Pers Commun. doi:10.1109/98.626984

Bussmann S, Sieverding J (2001) Holonic control of an engine assembly plant-an industrial evaluation. Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference, Tucson

Canard S, Coisel I (2008) Data Synchronization in Privacy-Preserving RFID Authentication Schemes. Proceedings of 4th Workshop on RFID Security, Budapest

Chang F, Dean J et al. (2006) Bigtable: A Distributed Storage System for Structured Data. Proceedings of the 7th Conference on USENIX Symposium on Operating Systems Design and Implementation - Volume 7 (Seattle, WA, November 06 - 08, 2006). USENIX Association, Berkeley

Chatterjee M, Das SK, Turgut D (2002) WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks. Clust Comput. doi:10.1023/A:1013941929408

Chiasserini CF, Chlamtac I, Monti P, Nucci A (2004) An energy-efficient method for nodes assignment in cluster-based Ad Hoc networks. Wirel Netw. doi:10.1023/B:WINE.0000023857.83211.3c

Cilia M, Antollini C, Bornhövd A, Buchmann A (2004) Dealing with Heterogeneous Data in Pub/Sub Systems: The Concept-Based Approach. Third international workshop on distributed event-based systems DEBS '04, Edingburgh

COUGAAR (2010) An Open-Source Agent Architecture for Large-Scale, Distributed Multi-Agent Systems. http://www.cougaar.org/. Accessed 20 June 2010

Crowley JL, Coutaz J, Rey G, Reignier P (2002) Perceptual Components for Context Aware Computing. Proceedings of the UBICOMP 2002, Goteborg

Dey A (2000) Providing Architectural Support for Building Context-Aware Applications. Dissertation, Georgia Tech

Dimokas N, Katsaros D, Manolopoulos Y (2007) Node Clustering in Wireless Sensor Networks by Considering Structural Characteristics of the Network Graph. Proceedings of the International Conference on Information Technology 2007, Las Vegas

Gavalas D, Pantziou G, Konstantopoulos C, Mamalis B (2006) Lowest-ID with Adaptive ID Reassignment: A Novel Mobile Ad-Hoc Networks Clustering Algorithm. Proceedings of the 1st International Symposium on Wireless Pervasive Computing, Phuket

Gilbert S, Lynch N (2002), Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services. doi: 10.1145/564585.564601

Grummt EO (2010) Secure Distributed Item-Level Discovery Service Using Secret Sharing. http://www.faqs.org/patents/app/20100031369. Accessed 20 June 2010.

Hayes-Roth B (1995) An architecture for adaptive intelligent systems. ARTIF INTELL. doi:10.1016/0004-3702(94)00004-K

Heinzelman WB, Chandrakasan AP, Balakrishnan H (2002) An Application-Specific Protocol Architecture for Wireless Microsensor Networks. IEEE Trans Wireless Commun. doi: 10.1109/TWC.2002.804190

Holmstöm J, Kajosaari R, Främling K, Langius K (2009) Roadmap to tracking based business and intelligent products. Comp Ind 60: 229-233. doi:10.1016/j.compind.2008.12.006

JADE (2010) Java Agent Development Framework. http://jade.tilab.com/. Accessed 20 June 2010

Krivokapic N (1997) Synchronization in Distributed Object Systems. Proceedings of BTW'1997, pp.332-341

Kushalnagar N, Montenegro G, Schumacher C (2007) IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.73.4790. Accessed 20 June 2010

Leanvitt N (2010) Will NoSQL Databases Live Up to Their Promise?, Comp. doi: 10.1109/MC.2010.58

Liang Y, Yu H (2005) Energy Adaptive Cluster-Head Selection for Wireless Sensor Networks. Proceedings of the 6th International Conference on Parallel and Distributed Computing, Applications and Technologies, Dalian

Liu JS, Lin CHR (2005) Energy-efficiency clustering protocol in wireless sensor networks. J Adhoc Netw. doi:10.1016/j.adhoc.2003.09.012

Maes P (1995) Artificial Life Meets Entertainment: Life like Autonomous Agents. CACM. doi: 10.1145/219717.219808

Niemi T, Niinimäki M, Sivunen V (2007) Integrating Distributed Heterogeneous Databases and Distributed Grid Computing, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.2. 1963. Accessed 9 June 2010

Onodera K, Miyazaki T (2008) An Autonomous Algorithm for Construction of Energy-conscious Communication Tree in Wireless Sensor Networks. Proceedings of the 22nd International Conference on Advanced Information Networking and Applications – Workshops. IEEE Computer Society, Washington

Öszu MT (1999) Distributed Databases. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.33.2276. Accessed 9 June 2010

Pátkai B, McFarlane D (2006) RFID-based Sensor Integration in Aerospace. http://www.aero-id.org/research_reports/AEROID-CAM-009-Sensors.pdf. Accessed 9 June 2010

Ray I, Ammann P, Jajodia S (2000) Using semantic correctness in multidatabases to achieve local autonomy, distribute coordination and maintain global integrity. Inf Sci. doi:10.1016/S0020-0255(00)00062-1

Stuart Russell S, Norvig P (2003) Artificial Intelligence: A Modern Approach. 2nd Edition, Prentice Hall

Ryan NS, Pascoe J, Morse DR (1998) Enhanced Reality Fieldwork: the Contextaware Archaeological Assisstant. http://www.cs.ukc.ac.uk/projects/mobicomp/Fieldwork/Papers/CAA97/ERFldwk.html. Accessed 26 May 2010

Sánchez López T, Kim D, Canepa GH, Koumadi K (2008) Integrating Wireless Sensors and RFID Tags into Energy-Efficient and Dynamic Context Networks. Comput J. doi:10.1093/comjnl/bxn036

Sánchez López, T. Huerta Canepa, G. (2010) Distributed and Dynamic Addressing Mechanism for Wireless Sensor Networks. *Submitted to Int J Distrib Sens Netw.Will be published in November 2010.*

Schilit WN, Adams NI, Want R (1994) Context-aware Computing Applications. Proceedings of the 1st International Workshop on Mobile Computing Systems and Applications, Santa Cruz

Suzuki S, Harrison M (2006) Data Synchronization Specification. http://www.autoidlabs.org/single-view/dir/article/6/265/page.html. Accessed 9 June 2010

Vasseur JP et al. (2010) Routing Over Low power and Lossy networks (roll). http://datatracker.ietf.org/wg/roll/charter/. Accessed 20 June 2010

Wang Y, Zhao Q, Zheng D (2004) Energy-Driven Adaptive Clustering Data Collection Protocol in Wireless Sensor Networks. Proceedings of the International Conference on Intelligent Mechatronics and Automation, Chengdu

Wong CY, McFarlane D, Zaharudin A, Agarwal V (2002) The intelligent product driven supply chain. Proceedings of the 2002 IEEE International Conference on Systems, Man and Cybernetics, Hammanet

Wu J, Gaol M, Stojmenvic I (2001) On Calculating Power-Aware Connected Dominating Sets for Efficient Routing in Ad Hoc Wireless Networks. In: Ni LM, Valero M (eds) International Conference on Parallel Processing: 3-7 September 2001 Valencia, Spain. IEEE Press

Ye M, Li C, Chen G, Wu J (2005) EECS: An Energy Efficient Clustering Scheme in Wireless Sensor Networks. Proceedings of the International Profesional Communication Conference 2005, Limerick

Younis O, Fahmy S (2004) HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks. IEEE Trans. Mobile Comput. doi:10.1109/TMC.2004.41