A Self-Governing, De-Centralized, Extensible Internet of Things within Resource-Constrained Domain

Horia A. Maior[1] and Shrisha Rao[2]

Abstract— **The Internet of Things is a term to describe networked objects not traditionally thought of as computers (e.g., cars, household appliances) which may nonetheless be connected using Internet protocols and technologies (TCP/IP, etc.). Such "things" may also be connected, for control or communication, to the traditional Internet, and may themselves also be equipped with sensors or actuators to interact with their environments. It is envisioned that an Internet of Things will be useful in particular in resource-constrained systems (e.g., with smart grids). The naive approach to an Internet of Things would use a central controller or master node of some sort to oversee the activities of all "things" in the Internet of Things. However, this has obvious drawbacks, not the least being scalability. It is therefore desirable that a "thing" govern its own actions while achieving global "self" properties (such as self-adaptivity, self-stabilization) in an Internet of Things that has to work with resource constraints (e.g., limited allowable peak electricity usage for a domestic or industrial system of many "things"). Such an Internet of Things with self-governing "things" would not require a central controller, and addition or removal of "things" would be far easier. This paper provides a theoretical model of such Self-Internet of Things, giving a set of principles and properties to the system in respect to resource-constrain and energy efficiency systems. This would involve a proposed type of behaviour for a single "thing" in such an Internet of Things, as well as the principles or protocols by which such "things" are connected with one another.**

Keywords: Internet of Things, Distributed Systems, Resource Allocation and Management.

I. Introduction and Related Work

The Internet of Things (IoT) has been a highly discussed topic in he past 10 years within many areas of research, in all government, academic and industrial contexts because of its great prospect. It dates back to 1990's, precisely with the start of Industrial automation systems [1]. Called "... the third wave of the world information industry after the computer and the Internet ..." [2], IoT is a term used to describe a network of "things"; smart objects (devices) located in the physical world, not traditionally thought of as computers (e.g., cars, household appliances). They are equipped with some sort of sensors, and are normally connected with the world surrounding them (e.g. human beings, other objects etc.), being able to exchange information with other objects using Internet protocols and technologies (TCP/IP, etc.).

[1]Horia A. Maior, Mixed Reality Lab, Faculty of Computer Science, University of Nottingham, United Kingdom. psxhama@nottingham.ac.uk

[2]Shrisha Rao, International Institute of Information Technology - Bangalore, India. srao@iiitb.ac.in

A. Self-governing, decentralized, extensible IoT, connected to a shared, variable power supply. Each thing is an object.

hehe ...

B. Sharing needs and deadlines between objects, provide them with a general view of the system. Higher priority objects with small power consumption are more likely to get powered, achieving a happy IoT

One interesting aspect of the IoT is it's future impact on our daily life. Zorzi et al [3] points out there is a first time opportunity to interact with the surrounding environments and to exchange information that previously was not available by simply looking at objects (devices). Apart from a direct interaction individual-machine (object), it will also have a indirect effect through object-to-object interaction. Objects were not used to communicate and be influenced by other objects. Prasad and Kumar [1] describes this particular aspect of IoT as the advance version of Machine to Machine (M2M) communication, where objects are exchanging information between them without a human intervention.

The application of IoT is vast, covering many areas of research. One of it's major application area is environment monitoring within Smart Cities (with Traffic Management research, Smart Parking etc.), Smart Environment (with Air Pollution research, Fire Detection research etc.) and so on. Liu and Zhou [2] finds the features of automatic and intelligent objects of IoT suitable for monitoring environment information.

IoT will be useful in particular in resource-constrained systems (e.g. with smart grids). The naive approach to an Internet of Things would use a central controller or master node of some sort to oversee the activities of all "things" in the Internet of Things. However, considering a network that includes billions of objects interconnected, this has obvious drawbacks, not the least being scalability. Moreover, being composed of a single controlling unit, such system can carry at most one activity at any single moment.

It is desirable that a "thing" govern its own actions while achieving global "self" properties (such as self-adaptivity, self-stabilization) in an Internet of Things that has to work with resource constraints (e.g., limited allowable peak electricity usage for a domestic or industrial system of many "things"). Liu and Zhou [2] describes such property as "autonomy Feature" of objects, where objects have the ability to reason, negotiate, understand, adapt and learn from other objects or environments. Such an Internet of Things with self-governing "things" would not require a central controller, and addition or removal of "things" would be far easier.

The specifications of our IoT system described above can be represented as a distributed system, where a central component is nonexisting. Instead, all objects communicate with every other object in the system in order to coordinate their actions. They also have a processing capability and self properties being responsible to control own actions.

RW:

- Maximum Demand
- Total Demand
- Demand Responsiveness
- Demand Balance

II. Problem Statement and The Model

In Section I we discussed and presented our interest and in the same time the focus of this paper. We describes the existing work within the area of IoT from various points of view, and mentioned the applications such IoT within many areas of research. In this section we will describe an abstract model of IoT from a resource-constrained point of view, formally stating the problem we address. At the end of the section, we present and prove our result. The next section will present ...

We are particularly interested creating a framework for an extensible, decentralized, self-governing, further proving the benefits of such a system in achieving energy efficient systems. We assume that the objects of the IoT system we are trying to represent are power resource consumers, and they are connected to a shared power supply, providing them with a limited, variable in time power resource.

A. Definition and Notations

Let there be numerous objects o_i that demand and may consume power resource. The set of all objects, denoted as O, together with some sort of power supply constitute the whole system.

$$O = \{o_0, o_1, ..., o_{n-1}\}, \text{ the set of all } n \text{ objects.}$$

Each object consumes a non-negative amount of power resource. To simplify the system, let us consider this non-negative amount of power constant for each object in the system (an object cannot change its power demand). Let $\mathbb{R}^+$ be the set of positive real numbers, then let the demand function, $f : O \to \mathbb{R}^+ | f(o_i) = r$, give the power demand of each object o_i. This means object o_i demands r amount of power from the power supply.

The Total Demand of system at any one time, as described by Rao [4], is the sum of all power demands for all objects in the system:

$$\sum_{i=0}^{n-1} f(o_i)$$

The power supply can be described as the total amount of power resource available to share between all the objects of the system at any time. The main property of the power supply is that it is variable in time. If we think of solar power example, during different daytime periods, available power resource may vary. Let the set T denote the set of all time instances, and $\mathbb{R}^+ \cup \{0\}$ be the set of positive real numbers including 0. Then, the function $\gamma : T \to \mathbb{R}^+ \cup \{0\}$ denotes the power supply function. With $\gamma(t) = w$, w the [4] the maximum resource limit, in our case power resource, available in the system at a given time t.

We described objects having various power demands and we described a power supply providing the system with power resource. The question here is, what if the power supply cannot satisfy with power resource all objects of in the system? Rao [4] describes that in practice, distributed processes (objects in our case) drawing some sort of resource (power resource in our case), are distinguished from one another in terms of some sort of priorities. This is because some objects are more important than others in terms of their functions and utilities, and if the power supply cannot satisfy all the objects, priority will distinguish between the the high priority to low priority objects.

Let $\delta : O \times T \to \mathbb{R}^+ \cup \{0\}$ be the priority function, with $\delta(o_i, t) = p$, where p is the priority of object o_i at time t. As described, priority function is time dependent, that means objects can change priority in time.

B. IoT as a Distributed System

In a distributed IoT, every object is networked in some way so it is able to exchange information with all other objects in the system (for simplicity, we make abstraction of how they are physically networked). It is also assumed that every object has some sort of computing capability (we assume that computing capability has no cost what so ever), and it is in some way 'self-aware' of its current 'needs' (for example each object knows or is able to find out about its power settings). Because we are building a de-centralized system, the decision-making comes to the object itself (this means that each object in the system decides on its own when is a suitable time to consume power resource), therefore, every object is also interested in an overview of the whole system (for example each object needs to know the maximum power limit $\gamma(t) = w$ but also each object has to know about what is its priority relative to other's in the network).

When creating such a system, each object in the system is "exploring" the whole system in some way. During the exploration, objects need to find out (and memorize in some way) other objects with the same priority, and in the same time find out other priorities in the system. This way, every object will have an general overview of what is their own priority relative to other object in the system. The exploration is possible trough some sort of communication and exchange of information between objects. Making abstraction of how the communication is made, let v and u be two objects; $v, u \in \{O\}$. Let $msg(v, u, m)$ be a message, where v is the sender of the message, u is the delivery object of the message and m is the message (the message can contain any kind of information coming from the sender). Following Peleg's approach [5], a message can be also broadcasted or "flooded" to/over a network (in our case system). Algorithm

1 below, is an adapted version from [5], for broadcasting a message across all n objects of a system, from a root object o_j:

Algorithm 1: Algorithm Flood

```
1 Let a source $o_j$
2 for $i \leftarrow 0$ to $n-1$ do
3     if $i! = j$ then
4         Send $msg(o_j, o_i, m)$
5     end
6 end
7 for $i \leftarrow 0$ to $n-1$ do
8     Upon receiving a message $o_i$
          • Store the message
          • Compute the message
          • Send acknowledgement
9 end
```

The number of messages sent by each object can be used to evaluate the complexity of the Algorithm 1 and all further algorithms used in this paper. Because each object is sending a message to all other objects, having n objects, in total there will be $n \times n$ messages across the whole system; therefore the Big O of Algorithm is $O(n^2)$.

Using Algorithm 1, we design an exploration algorithm that is run by all objects in the system, all trying to communicate to other objects information about their priority and their power demand. In the same time, objects receive, compute and store information about other objects. Therefore, every object will end up having an overview of their position in the system.

Let o_i be an object; $o_i \in \{O\}$. As described above, each object stores information about other priorities in the system and other objects of the same priority in the system. Let each every object o_i have to arranged lists:

- let $P_i =$ be an arranged list where each object o_i can store priorities of other objects (decreasing order); such that $P_i(0)$ is the highest priority in the set.
- let Q_i be an arranged list of objects of the same priority as o_i (increasing order); such that $Q_i(0)$ is the object with smallest demand.

Before joining the system and running the exploration algorithm, every object o_i has: $P_i = \{\emptyset\}$ and $Q_i = \{\emptyset\} \cup \{\delta(o_i, t)\}$. Please consider Exploration Algorithm (2)below.

Algorithm 2: Exploration Algorithm run by all object joining the system

```
1  $t_0 = currenttime$
2  $o_i \leftarrow$ Flood Algorithm over $G$
   sending: $msg(o_i, join, \delta(o_i, t_0), f(o_i))$
3  for $j \leftarrow 0$ to $n-1$ do
4      Let $o_j$, $(j! = i)$ receive the message
5      Upon receiving message:
       if $\delta(o_i, t_0) = \delta(o_j, t_0)$ then
6          Insert $o_i$ in $Q_j$ in order of $f(o_i)$
7          Send acknowledgement
           $(\delta(o_i, t_0) = \delta(o_j, t_0))$
8      end
9      else
10         if $\delta(o_i, t_0) \notin P_j$ then
11             Insert $\delta(o_i, t_0)$ in $P_j$ in order
12             Send acknowledgement
               $\delta(o_i, t_0) < \delta(o_j, t_0)$ (if so)
               Send acknowledgement
               $\delta(o_i, t_0) > \delta(o_j, t_0)$ (if so)
13         end
14     end
15 end
```

The Algorithm 2 has complexity Big $O(n^2)$, and this is mostly because it is using Algorithm 1 (every object sends a message to all other objects).

Being an extensible IoT system, new objects can join at any time, and objects may no longer want to be part of the system and leave at any time. An object joins the system when it demands power resource. When joining, the new object shall run the exploration algorithm. This way, other objects will make note of the new object's details (like priority and power setting), but in the same time, the new object will create it's overview of the system trough message acknowledgements from the existing objects. An object is leaving the system when it is no-longer interested in power resource at that particular time. On leaving, the object informs all the other objects about it's intentions.

As mentioned in the previous section, we are describing a system with non-preemptive power policy. Meaning that there is no partial fulfillment; when an object was allocated power resource, it is not interrupted until it has finished it's demands, or there is a change in available power resources. In other words, if a new object with high priority joins the system at some point and a low priority object has been allocated with power, the new object has to wait until the low priority object is leaving the system, or more power is provided by the power supply. Algorithm 3 describes exactly what happens when an object is leaving the system.

In Algorithm 3, the leaving object messages all other objects three important pieces of information:

- it's priority level $\delta(o_i, t)$ over (together with the list with other objects of it's priority Q_i)
- it's power demand $f(o_i)$.
- a boolean $powered$ representing whether or not the leaving object was powered or not.

In the case that the leaving object has the same priority with the one receiving the message, the last one mentioned has to remove the the leaving object from the same priority objects database Q_j (see line $5, 6$ in Algorithm 3). In the case that the leaving object is the last of its priority, the whole priority level is removed from the other priorities database P_j (see line $9, 10$ in Algorithm 3). Finally, if the leaving

Algorithm 3: Leave Algorithm run by every object leaving the system

```
1  $t = leavingtime$
2  $o_i \leftarrow$ leave the system
   Flood Algorithm over $G$
   send:
   $msg(o_i, leave, \delta(o_i,t), Q_i, f(o_i), powered = T/F)$
3  for $j \leftarrow 0$ to $n-1$ do
4      Let $o_j$, $(j! = i)$ receive the message
5      Upon receiving:
       if $\delta(o_i,t) = \delta(o_j,t)$ then
6          Remove $o_i$ from $Q_j$
7      end
8      else
9          if $Q_i - \{o_i\} = \emptyset$ then
10             Remove $\delta(o_i,t)$ from $P_j$
11         end
12     end
13     if $powered = T$ then
14         $w \leftarrow w + f(o_i)$
15     end
16 end
```

object was powered, it's power demand becomes available for other objects (see line $13-15$ in Algorithm 3).

C. Prioritized objects and a happy IoT

In he subsections above, it has been described a distributed, extensible system, with objects that are able to communicate with other objects of the system, with new objects joining the system and objects maybe leaving the system. However, we did not yet mentioned how the system is powered from the power supply, that is how each individual object receives, if so, power resource.

We discussed that objects can have different priorities, and that is because some objects are more important in a way than others. In the case of not being able to satisfy all objects with power resource, higher priority objects are more likely to get power resource than lower-priority ones. We also discussed that objects can have different power demands. That is, one object might need more power resource than other ones. In the case of objects having the same priority level, the system satisfies shall provide power resource to low consumer objects, that is to satisfy with power as many high priority objects as possible.

Let there be n objects $o_i \in O$, with $0 <= i <= n-1$. Each object has a power demand $f(o_i)$, a priority function $\delta(o_i,t)$, and each object has created its own overview of other objects priorities in the system (in decreasing order the list P_i; such that $P_i(0)$ the highest priority), and objects with the same priority (in increasing order of power demands the list Q_i; such that $Q_i(0)$ is the smallest demand object of $o_i's$ priority). The power supplies is denoted by the function $\gamma(t) = w$, providing the power resource available at every time $t \in T$. The following algorithm (Algorithm 4) describes how the power supplier, supplies the highest priority object with the smallest power demand first, and then how each object informs other objects about their actions and remaining power resource.

Algorithm 4: Power Algorithm

```
1  $t = currenttime$
2  Let $o_i$ the object running this code
3  if $\delta(o_i,t) > P_i(0)$ and $f(o_i) = Q_i(0)$ then
4      $w \leftarrow \gamma(t)$
5      if $f(o_i) <= w$ then
6          Power object $o_i$
7          $w \leftarrow w - f(o_i)$
8      end
9      if $Q_i - \{o_i\} = \{\emptyset\}$ then
10         $nextPriority \leftarrow P_i(0)$
11         Broadcast $msg(power, nextPriority, w)$
12     end
13     else
14         $nextObj \leftarrow Q_i(1)$
15         Send $msg(power, nextObj, w)$
16     end
17 end
18 Upon receiving one of the msg:
   Receiving $msg(nextObj, w)$ or
   Receiving $msg(nextPriority, w)$
   if $(o_i = nextObj)$ or
   $(\delta(o_i,t) = nextPriority$ and $f(o_i) = Q_i(0))$ then
19     if $f(o_i) <= w$ then
20         Power object $o_i$
21         $w \leftarrow w - f(o_i)$
22     end
23     if $Q_i - \{o_i\} = \{\emptyset\}$ then
24         $nextPriority \leftarrow P_i(j) <=>$
           $(P_i(j-1) > \delta(o_i,t) > P_i(j))$
25         Broadcast $msg(nextPriority, w)$
26     end
27     else
28         $nextObj \leftarrow Q_i(j) <=> (o_i = Q_i(j-1))$
29         Send $msg(nextObj, w)$
30     end
31 end
```

Algorithm 4 is run by all objects in the system individually. The first part of the algorithm, lines $3-17$, will addresses to the object with the highest priority and the lowest power demand of the system (line 3), this object having the first chance to get powered (in the case that the available power resource can satisfy its power demands - line $4-6$). From this point, the current object informs the next object with the same priority (if there is one - lines $13-16$), otherwise it broadcasts an information containing the new priority that shall receive power ($lines 9-12$). The second part of Algorithm **??**, lines $18-31$ address all the other objects of the system, which are waiting for a message addressing them or their priority level. The smallest consumer of a priority category will always be powered first. Regardless if an object gets powered, it will always send a message to the next object or priority group. This enables small demand objects with small priority to get powered in the case of a high priority object cannot fulfil its demands with available power resource.

The complexity of Algorithm 4 in Big O notation is $O(n^2)$. The worse case scenario is when all objects have different priorities, and every object is broadcasting a message once ($n \times n$).

D. Proofs of Correctness

In the previous subsections we developed a framework and created algorithms for describing our IoT model. In this subsection, we are identifying a set of properties of the system, and we are trying to prove the correctness of our proposed design by validating these properties.

Property 1: *No Starvation*. This property guaranties that all the available power will be used by the objects of the system, ant there will be not object entering a starvation situation while there is available resource in the system meeting that objects demands.

Proof 1.1 (The obvious case). When the available power resource exceeds the Total Demand of the system, all objects of the system shall be powered. Let $B = \{T, F\}$ the set of boolean values true (T) and false (F), and function $p : O \times T \to B$:

$$p(o_i, t) = \begin{cases} \text{T} & \text{if } o_i \text{ is powered} \\ \text{F} & \text{if } o_i \text{ is not powered} \end{cases}$$

$$\text{If } \gamma(t) \geq \sum_{i=0}^{n-1} f(o_i) \implies \forall o_i \in O, p(o_i = T)$$

Proof 1.2. If there is not enough power resource to meet the Total Demand, the available power will be used to the maximum possible (after running Algorithm 4).

$$\nexists o_i \in O, (p(o_i) = F) \textbf{ and } (f(o_i) \leq w);$$

Property 2: *Valid Priority*. This property guaranties that high priority objects get powered first if possible.

$$\forall u, v \in O \text{ with } (\delta(u,t) \leq \delta(v,t)) \textbf{ and } (w \geq f(u) \text{ and } w \geq f(v)), \nexists\ (p(u) = F \textbf{ and } p(v) = T).$$

and Conclusion

III. Results, Discussion and Further Work

A. Set of principles for a self- Internet of Things

- principle one - principle two - principle three etc and Conclusion

References

[1] S. S. Prasad and C. Kumar, "An energy efficient and reliable internet of things," in *Communication, Information & Computing Technology (ICCICT), 2012 International Conference on*. IEEE, 2012, pp. 1–4.

[2] Y. Liu and G. Zhou, "Key technologies and applications of internet of things," in *Intelligent Computation Technology and Automation (ICICTA), 2012 Fifth International Conference on*, 2012, pp. 197–200.

[3] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, "From today's intranet of things to a future internet of things: a wireless-and mobility-related view," *Wireless Communications, IEEE*, vol. 17, no. 6, pp. 44–51, 2010.

[4] S. Rao, "A foundation of demand-side resource management in distributed systems," in *Transactions on computational science VIII*. Springer, 2011, pp. 114–126.

[5] D. Peleg, *Distributed computing: a locality-sensitive approach*. SIAM, 2000, vol. 5.