Evolving Presentity-Based Context Schemas by Estimating Context Proximity

Jamie Walters and Theo Kanter

Department of Information Technology and Media
Mid Sweden University, Sundsvall Sweden
{jamie.walters,theo.kanter}@miun.se

Abstract. The definition of what constitutes context proximity has remained largely unexplored but accepted as being a fundamental issue towards realising an architecture of connected things. Existing solutions aimed at enabling context awareness are often undermined by their dependencies on centralized architectures limited with respect to their scalability. Our previous work proposed the use of the so called Context Schema; an encapsulated representation of the information points constituting the context of a presentity. Building such a schema requires support for determining set members limited by some metric; a proximity metric. In this paper, we propose an algorithm for estimating the context proximity among presentities, enabling complete schemata of entities relevant to, and expressing the current context of a presentity. Secondly we propose an extension of a gossiping algorithm to optimize the ability create schemata as one traverses a vast and dynamic connected things infrastructure.

Keywords: Context, Context Awareness, Presentity, Self-Organization, Real Time, Context Proximity.

1 Introduction

The increasing interest in the provisioning of applications and services that deliver experiences based on context mandates the continual research into methodologies, architectures and support for delivering the context information required. Constraints on service delivery with respect to real-time availability underpins any such solution [12].

A future connected things infrastructure, with an expected device base exceeding billions [21], may be expected to support a wide range of context centric experiences ranging from personalized and seamless media access, to intelligent commuting or environmental monitoring. Such seamless connectivity will extend to and include devices such as mobile phones, personal computers or IPTV boxes. All converging towards the paradigm of *everywhere computing* [15]; the seamlessly connected *Internet of Things*.

In response to this, previous and existing works attempt to enable measures of context proximity by incorporating sensors, actuators and other intelligent artefacts into the real world. Research such as that undertaken with the AmbieSense

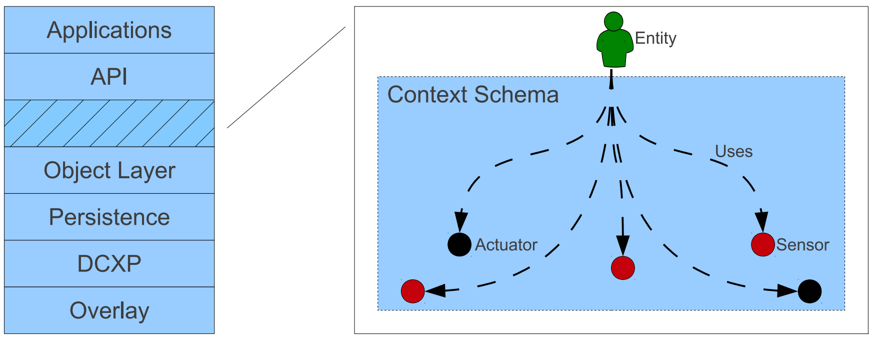

Fig. 1. Context Framework Model

project [9], sought to enable this derivation of context by means of utilizing embedded context tags. Others such as the *Smart-its Friends Project* [10], [2] or [7] uses devices attached to users as a means of establishing indicators of context. Such approaches require the adoption of specific hardware devices, subsequently creating a barrier to large scale user adoption. Alternative approaches sought to derive context from existing pervasive devices such as temperature, pressure and GPS sensors.

This is an integral part of projects such as *Senseweb* [13] and *SENSEI* [17]. The *SENSEI* project however does not provide support for dynamic changes in context as it relates to the discovery of new sensor sources, requiring the explicit modelling of the user-sensor relationships. However, the centralization of these approaches undermine their ability to scale well impacting on performance with respects to real time constraints. Citing this, we acknowledge the need to create solutions that are capable of disseminating sensor information within real-time constraints and scaling to accommodate the vast infrastructures of connected things existing in the future.

Acknowledging that sensor information required more meaning in order to realise useful services, we extended our work by creating an object-oriented ontology capable of being distributed across a peer-to-peer overlay [6]. With such a model, we enforce a *metaobject/value* split, permitting us to reason over a model without reference to its current values.

We separated the concept of a Presentity [5] from other entities such as sensors and actuators and additionally, introduced the concept of a *Context Schema*, defined as:

The collection of information points associated with and contributing to a presentity's current context

where an *Information Point* is defined as:

Any source providing information about the context of an entity or any sink capable of accepting an input effecting changes to an entity's context

Such a schema is attached to a presentity and encapsulates all its information points and relationships. An application or service with a requirement to deliver some user context-centric experience, subscribes to the current schema description; it realises a *publish/subscribe* interface to the entities described by the schema and retrieves the current context information. As a presentity traverses a connected things infrastructure it discovers new entities and consequently updates its schema to reflect this. As a result, all subscribing end points receive an updated schema and can adjust their services to accommodate this. For simplicity, we refer to information points as *points*.

While this presents a scalable distributed means of deriving continual changes to context, questions concerning the means of accomplishing such an evolving schema remains unresolved. A solution would require that we are able to derive and disseminate localized relationships from a global context infrastructure while remaining within real-time constraints. Such a solution, coupled with a context model and sensor information distribution will serve to further enable distributed context information in support of the proliferation of ubiquitous computing applications and services.

2 Motivation

The need to derive proximity metrics is critical component of any infrastructural approach to context aware computing [11]. Users navigating an interconnected *Internet of Things* require infrastructures capable of responding to queries concerning context as well as enable experiences based on the available information points within proximity.

As with any typical day within an urban environment, people are constantly on the move for business or pleasure. Within such a future cityscape, there exists multiple information points which maybe used to inform a person on the state of his surroundings. A digital ecosystem capable of providing enough information in order to derive support for services wishing to effect changes or deliver experiences to a user, based on context. Such information might include audio devices, internet connections or video devices; a range sensors including temperature, humidity, lighting and capacity sensors or even location, traffic and air quality sensors. A person with a smart device would be able to connect to and derive representations of context from these points in order to support his applications. A hearing-impaired person may simply need to find a store with a hearing-aid loop in order to be able to comfortably make a purchase.

Existing solutions such as [10], [2] and [3] could enable the discovery of such points. However, they would require the installation of additional devices. Firstly, devices attached to points around the cityscape or within buildings and secondly a device attached to each user. This solution has some advantages with respects to providing known anchor points, and partially negating the problems of GPS dead-spots. However, the user would be required to position himself within some spatial proximity of the device connected and perhaps motioning in order to initiate a proximity indicator to the supporting architecture. This would then create a connection between the user and some artefact; providing access to the information points available. Such a solution requires the explicit intervention of the user and be dependent on the user knowing the location of such points with which to synchronize. With a large population, this could become a nuisance with people queuing in order to be synced with the infrastructure, negating much of the progress made in realizing seamless ubiquitous computing. Any solution to deriving context within such a heterogeneous landscape must consider all indicators and be able to derive context information with minimal user interaction.

Other solutions enabling the ability to derive proximity through the use of spatial locating techniques such as GPS, have functional limits, such as being on a subway train. We concur with [20] and [19] and divert from the concept that physical location provides the overarching indicator of context and is required in order to enable useful context dependent services. Other work such as [17] would allow for users to be connected to the cityscape but achieves this by using largely static determinations of what constitutes the information points connected to a presentity. This gets challenged however, if the city won the Olympic Bid, and an impromptu concert was being held in the city; the user would have no access to the resources as [17] does not permit additions in such an ad hoc manner but rather an infrastructure created, designed and instantiated by administrators. This further undermines the actual dynamic behaviour of the connections and people in the real world.

Each person however, possesses context indicators hinting at his proximity with regards to other presentities within the digital ecosystem. He might have a GPS sensors, a physical proximity sensor or just a calendar event indicating his expected location.

Indexing approaches such as search engines considered this theory of connected things relative to static document content on the Internet. A document's connectivity determines its relevance with regards to the size of the entire document collection. This concept of page ranking has been explored and used both in a centralized solutions [1] as well as distributed solutions [22]. Centralized solutions such as Google index a tiny portion, less than 10 billion of the estimated 550 billion pages, on the relatively static Internet [16] [22]. Any attempt to apply such a centralized solution to locating and building context models representing such dynamic situations would be undermined by their ability to scale well. Distributed solutions based directly on the PageRank concept would not scale well to accommodate highly dynamic document sets. Current *real-time* searches are realized by targeting known content providers which could not scale to accommodate the vast and mostly ad hoc nature of a connect things infrastructure.

Previous work concerning data mining within context centric architectures [18] [4] provided a means of deriving relationships between presentities and other entities or identifying patterns in the information they contain. We however, are not in search of data mining algorithms which require a collection of data capable of supporting inference. The dynamic nature of context information undermines this and results in solutions where data-mining is carried out over historical records on the presumption that it represents an indication of current context. In contrast, consider a user entering a city for the first time, his current context would display a change in patterns rendering all services based on his previous behaviour irrelevant. Such expectations of a context support system is not unusual in a world where people are becoming increasingly mobile and dynamic.

The need therefore exists for methodologies capable of evolving and establishing localized sets of information points capable of answering a query concerning a user in real-time. Mandated are solutions that, in real-time, identifies and collates information sources considered to be within close proximity [11] to a user's context and are therefore able to provide context information supporting applications and services. The remainder of this paper is divided as follows: section 3 addresses some background work related to this paper; section 4 looks at the proposed approach to calculating context proximity; section 5 looks at how our solution would work in a distributed architecture while section 6 details our conclusions and future work.

3 Background

3.1 Distributed Context Exchange Protocol and Overlay - DCXP

DCXP [14] is an XML-based application level P2P protocol which offers reliable communication among nodes that have joined the P2P network. Any end-device on the Internet that is DCXP capable may register with the P2P network and share context information. The DCXP naming scheme uses Universal Context Identifiers (UCIs) to refer to Context Information (CI) such as sensors that are stored in the DCXP network.

3.2 The Gossiping Algorithm

Previous work [8] presented a lightweight peer-to-peer algorithm for organizing entities into small dynamic groups based on some indicator value. The main aim, was to derive the ability to maintain groups that were centred around an entity according to the preferences of the entity expressed as a known measurable value. Such organization was unstructured and occurred as values changed with respect to changes in the entity itself or changes to the entity's affinity to the value.

At its core, the algorithm is based on simple gossiping, targeted at keeping all interesting entities within a single hop from the interested entity, thereby enabling quick communication and exchange of information. At runtime, an entity, A, on initialization queries a database for another entity that is within its preferred value range, eg: *5km from its location.* It then connects to this entity, B, and queries B for any known entities within its range. Such entities are forwarded to A, evaluated and add added to its list of neighbours. It continues to probe all nodes it knows, continually updating its list of neighbours. A now has a list of all relevant entities with which it may communicate in order to fulfil some application or service dependent on proximity.

In a distributed environment, such an algorithm requires that any element of centralization be completely removed in order to achieve an ad hoc and distributed solution. In support of this, we create a resource index on a distributed overlay which provides a starting point for finding entities. A presentity issues, on joining, a search for other presentities matching its proximity criteria, retrieve an initialization list with which to start probing for entities within close proximity, recursively doing so to locate all known entities matching its criteria.

3.3 The Distributed Object Model

The CII ontology as described in [6] details an *entity-predicate-entity* triple implemented in an object-oriented framework. Such a model is similar in concepts to the semantic web approaches, however it remains advantageous with regards to the time taken to traverse and reason over an object-based model. It provides for a way to represent the relationships and interactions in an *connected things* within an *Internet of Things.* Where *things* can range from sensors and actuators to virtual information sources such as social networks, media, people and infrastructure.

The CII model can be extended with new sub-concepts of *Entity* and *Information-Source.* These concepts would be presented as classes following a standard interface. This integration would be made possible by adaptive software techniques such as a combination of computational reflection, automated compilation and dynamic class loading. Agents, applications and services reside above and use the meta-model as a source of data and deriving context information.

It was extended to include a *Schema Entity* which is attached to a presentity and describes the current model of sensors and actuators that provide context information supporting the presentity. In this way the *watchers*, regarded by [5] as *the entities interested in a presentity's prescence*, may has access to a defined real-time picture of all the information points related to a presentity. It can then choose which sensors to use in order to deliver its services.

Schema entities, however have one additional property; they expose a *publish/subscribe* interface. We take this approach in order to avoid having to synchronize large datasets distributed around the architecture. Watchers [5] can therefore subscribe to a schema and be notified as it changes. There is no need to issue queries to nodes or databases or for watchers to be concerned with checking for updated presence information.

4 Approach

A context network, within our definition, is comprised of presentities with attached information points creating a directional graph as illustrated in figure 2. A presentity connected to an information point undertakes this as a means of deriving a representation of context or as a means of effecting a change in its context. Such connections are highly volatile and dynamic but while they persist,

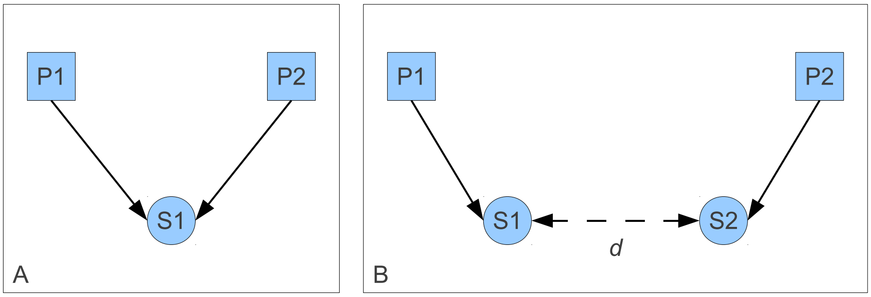

Fig. 2. Presentity & Information Point Relationships

they provide an implicit link between presentities that are attached to and deriving their context from information points within a context-based infrastructure. In figure 2, P_1 while connected to S_1, derives an implicit but existing relation to P_2 via S_1. The implication being that their connection suggests that P_1 and P_2 share, to some extent a similar context. We consider also the alternative scenario shown in figure 2 where P_1 is connected to S_1 and P_2 is connected to S_2. If S_1 and S_2 are expressing the same context indicator type, i.e. they are two information points of the same type such as a temperature sensor, then P_1 shares a context similar to that of P_2 by a function of the difference between S_1 and S_2; their point proximity.

Further, one may view the relationships that are constructed between these presentities as being states in a *Markov Chain*. The probability that P_2 possesses a close proximity to P_1 is derived solely from the current context state at P_1. Similarly to link-based ranking algorithms used in static document ranking, this probability is the degree of the relationship between P_1 and P_2. The current context state of P_1 on the other hand, has no influence on the relationships of P_2 and any other presentity within the architecture. Further, $P_2's$ relationship to P_1 is only influenced by the current context state of P_1, disregarding any past or future states.

4.1 Sensor Proximity

With this assumption, we build localised directed graphs for each presentity based on its relationships with other known entities at a given time. This presents the first problem of finding entities that lay within X_1; the context proximity limit of P_1. The architecture defined in figure 2 illustrates a multi-layered approach, comprising of an application layer residing on top of an API. We envision that applications will be able to define limits of X_1 such as: *find all people within 3km with a temperature less than* $5\,^{\circ}\mathrm{C}$ *difference* permitting us to obtain parameters needed to derive the entity sets.

Further, we assume that all sensors are able to provide context values in some discrete representation permitting comparison. We acknowledge situations where there exist representations of context that are relatively more difficult to express as discrete values. However our previous work into creating object models for context representation [6], addressed this by permitting information points to implement a comparator interface. This would allow us to compare two values; obtaining either a boolean comparison or some discrete value representing the distance between context values. Such an example would occur when comparing appointments on calender, where a simple *true/false* would suffice in comparing current meeting availability.

With parameters for what constitutes the limit of context values that the applications and services require, we proceed to query for presentities bearing a similar context or a similar context to some degree. Firstly, we query each dimension using the rules outlined below. The process is repeated for each point attached to P_1.

We create groups of points within proximity of S_1. Figure 2 illustrates points, S_1 and S_2 respectively connected to presentities P_1 and P_2. For each context dimension, we find all the points within the context proximity, X, required by the applications.

From this, we create a cluster of points with a context value similar to that of S_1 within a context proximity calculated where:

$$(|V_{S1} - V_{Sk}|) \leq X_{S1} \tag{1}$$

here, V_{S1} is the current value of S_1 and V_{Si} is the current value of S_i with X_{S1}. Therefore within a domain D, S_1 obtains a set of related sensors at time t such that:

$$G_t = \{S : S \in D_t : (|V_{S_1} - V_{S_i}| \leq X)_t\} \tag{2}$$

This is a dynamic set of information points with respect to P_1 its context dimension S_1, that continually evolves to reflect the addition or removal of sensors with respect to their current values.

We consider the fact that not all instances of S_i lie within the same proximity to S_1. This implies that S_1 shares a closer context with some members of G and subsequently those members must be given a higher preference with regards to any context dependent application or services wishing to find context information points in support of delivering some optimal user experience. Using the distance X would not be a reliable indicator of such relevance, since the scales could be different for each sensor S_i attached to P_1. We normalise these values as a function of the value with respect to the scale and the distance from S_1 as follows:

$$R_{S_i} = f(S_i) = (1 - |V_{S_i} - V_{S_1}| \cdot {X_{S_1}}^{-1}) \tag{3}$$

where:

$$0 \leq R_{S_i} \leq 1 \tag{4}$$

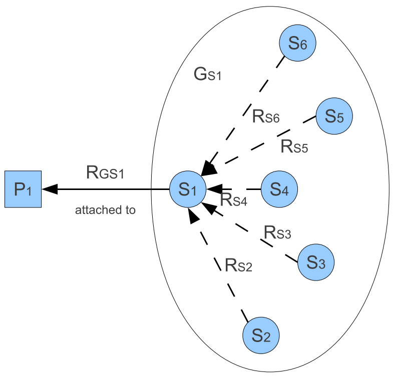

Fig. 3. Single Dimension Context Proximity Cluster

A value of 0 being at the edge and 1 being identical to X_{S_1}. This value is useful for us for two reasons, firstly it can be used to apply a weighting to the edges connecting S_1 to S_i and subsequently the edges connecting to P_1. Secondly, we use this value to calculate the average rank of the set of G_{S1}; this we express as follows:

$$R_{G_{S_1}} = \frac{\sum_{i=0}^{n}(1 - |V_{S_i} - V_O| \cdot {X_{S_1}}^{-1})}{i} \tag{5}$$

where:

$$0 \leq R_{G_{S_1}} \leq 1 \tag{6}$$

A value closer to 0 being not well connected and 1 being connected to a set of sensors with similar context values. This represents an indicator of the connectivity of S_1 with respect to its current context value, and an indicator of the probability that there exists good connections to presentities sharing a similar context to P_1. Assuming P_1 treats each sensor equally, an application interested in enabling some service based on $P_1{}'s$ context can begin by exploring the groups with the highest ranking attached to P_1. Those groups would more likely contain links to entities with a context similar to P_1. Figure 3 illustrates once such resulting cluster of information points.

4.2 Presentity Proximity

Figure 4 illustrates a set of presentities with implicit connections derived from the algorithm discussed in section 4.1. Here we derive that the two presentities posses some degree of context similarity owing to the fact that their underlying sensors supporting their context are within close proximity. By deriving the degree of this closeness, we can return to the applications and services, a list of

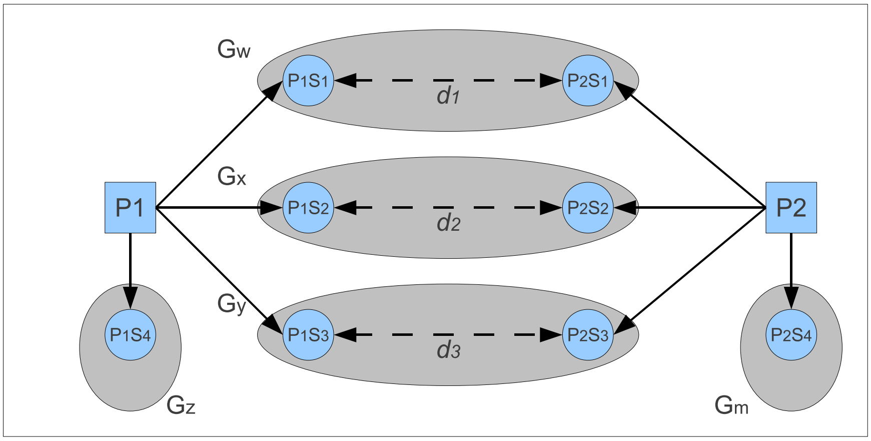

Fig. 4. Determining Presentity Proximity

presentities within proximity and a recommendation of any sensors that could be added to our schema. For example, discovering that we are in close proximity to a conference room, we could recommend that the information sources such as the whiteboard be made available.

For simplicity, we consider that P_1 is connected to four sensors, each with a cluster of sensors within its given proximity. Such that the set of *sensor clusters* of P_1 would be:

$$P_1 = \{G_w, G_x, G_y, G_z\}$$

P_2 is connected to three sensors, each with a cluster of sensors within its given proximity, such that the set of sensor clusters of P_2 would be:

$$P_2 = \{G_w, G_x, G_y, G_m\}$$

Based on this, we calculate the similarity of the set of sensors shared by P_1 and P_2. We consider each sensor cluster to which P_1 and P_2 are connected as being common members of their sets of sensors; we disregard the proximities. Using this, we calculate the set similarity as an indicator of their context similarity, PS as follows:

$$PS(P_1, P_2) = \frac{|P_1 \cap P_2|}{|P_1 \cup P_2|} = \frac{|G_w, G_x, G_y|}{|G_w, G_x, G_y, G_m, G_z|} \tag{7}$$

We do this in order to permit the comparison of values that cannot easily be measured discretely such as favourite colour, mood, etc. While we cannot easily apply metrics to such states of context, we may choose to apply non-discrete measurements obtained from learning algorithms, etc. In these expressions of context, we are unable to perform discrete distance measurements or limit proximity based on this, however we can provide mechanisms for grouping together similar values which might equate to a user saying: *I like red, but pink, and purple are also acceptable alternatives.* While it is non-trivial to calculate a measurable distance between this set of values, treating it as a set of information points supporting a presentity permits us to compare it to another non-discrete set. If a presentity was comprised entirely of non-discrete values, we could still derive a measurement of distance based on the grouping of these values and finding the degree of similarity between the presentities. Here an application could define which dimensions of context must be taken into consideration when calculating similarity, such that for an application only interested in distance and temperature, the corresponding equation could be:

$$PS(P_1, P_2) = \frac{|G_w, G_x|}{|G_w, G_x|} \tag{8}$$

or for distance, temperature and humdity, where P_2 had only two dimensions:

$$PS(P_1, P_2) = \frac{|G_w, G_x|}{|G_w, G_x, G_y|} \tag{9}$$

Further to this, we consider the equation in 3 and adjust the value derived in equation 9 to reflect the distance between the underlying expressions of context supporting the presentities. This is adjusted by a factor of the average of the rank of all the connections between P_1 and P_2. Therefore, we state the distance between two presentities, PR to be:

$$PR = \begin{cases} PS \cdot \dfrac{\sum_{n=0}^{k} R_{Sn}}{k} & \text{, i=0} \\ PS \cdot \dfrac{\sum_{n=0}^{k} R_{Sn} \cdot P_{Sn}}{\sum_{n=0}^{k} P_{Sn}} & \text{, i>0} \end{cases} \tag{10}$$

where i is the number of dimension restrictions, P, indicated by the application or service. When applying such restrictions, all dimensions must be accounted for. Each unaccounted for dimension will be ignored, effectively given a priority of 0. We provide for this as we consider that an application or service will be able to indicate context dimension priorities, eg. *find all persons within a context proximity of 0.7, prioritise by distance, then temperature* or *find all persons within 5km, prioritise by distance then temperature.* This would provide us with two dimensions $D_1 \& D_2$. The dimensions are progressively used to adjust the final ranking value of the results.

5 Presentity Proximity in a Distributed Architecture

Within a distributed architecture, the implementation of such a presentity proximity algorithm gains the best implementation with respect to performance and

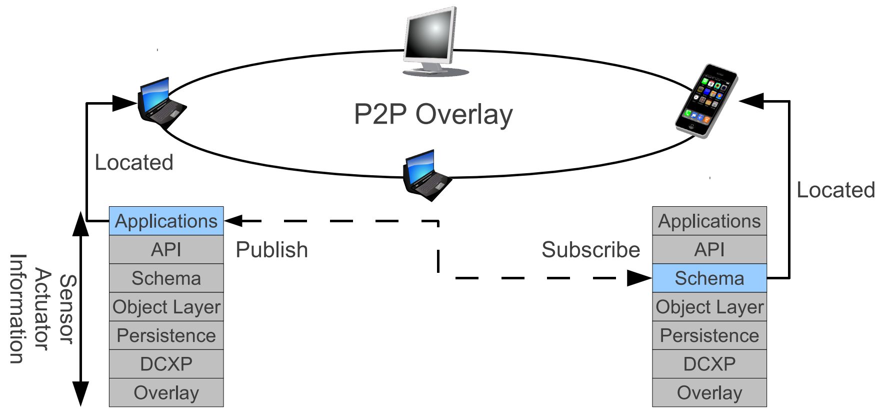

Fig. 5. Overview of a Distributed Approach

its ability to scale. As shown in figure 5, an application does not have to exist at the same node as the presentity it is trying to support. Such an application subscribes to the presentity's schema and uses this to derive the context metrics required.

Our distributed solution, implemented on DCXP (section 3.3) for messaging support, would implement the distributed gossiping algorithm described in section 3.2. An application wishing to find entities close to P_1 would accomplish this by first locating some initial nodes with a context proximity X_1 of S_1. Since our solution must remain fully distributed, we located initial nodes by issuing a search using the the range querying function of the underlying PGrid overlay. This returns a list of entities with respect to the query and constructs a running query at each peer with the following constraints:

1. The peer is responsible for a sensor fitting the criteria of the search
2. The peer is responsible for a sensor S_i with a range such that the set of sensors fulfilling the query from S_1 would be a subset of a query from S_i

Each peer is then required to:

1. Forward the sensors matching the standing query to S_1
2. Forward the query from S_1 to any node it encounters that matches 1 & 2 above

This happens only while S_1 maintains a relationship with S_i. When this is not the case, the peer responsible for S_i cancels the standing query and no longer forwards it. With this approach we minimize the number of sensors being forwarded in response to an expired query. Further, since the indicators of context such as temperature or location are likely to change gradually, applications requiring a new set of nodes in response to changed context would benefit from S_i forwarding the query to current matching nodes. This, as the query would still likely be valid for a subset of these nodes or nodes that are within their groups. We summarize this in Algorithm 1.

Algorithm 1. Finding and Ranking Relevant Information Points

```
loop
  {at the local node}

  for all information points attached to $S_1$ do
    determine maximum proximity value
    issue a range query for information points within proximity
  end for
  for all results, $S_i$ received do
    calculate proximity between $S_1$ and $S_i$
    if proximity $\leq$ max then
      add information point to list
      calculate the ranking $r_{S_i}$
      attach $S_i$ to $S_1$ with a degree of $r_{S_i}$
      calculate and update group ranking
    end if
  end for

  {at the remote nodes}

  for all range queries received do
    if there are local information points matching query then
      return information points to originator, $S_1$
      create a standing query, notifying the originator of this
    else if there are known peers with points matching this query then
      forward the query to each peer
    else
      ignore query
    end if
  end for
end loop
```

6 Conclusion

In this paper, we presented an approach to measuring context proximity among presenties in a future internet of connected things. With our approach, we are capable of building dynamic user-based context-centric clusters of information points and presentities that are capable of enabling applications and services to provide user experiences based on current context.

We proposed a simple algorithm for determining cluster members and classifying members relative to their distance from ideal value in the cluster. This provided us with a directed weighted graph between a presentity and each sensor obtaining a current context value within the range desired by a requesting application. By exploring this the paths in this graph, we derive other entities that possess a context value within our range indicating a context with some similarity to ours. Further to this, select all presentities sharing some context similarity and calculate the current schema similarity based on the groups of sensors that they possess.

This value for similarity was then adjusted using the value of the weighted edges connecting both presentities. With such connections, we can further derive new sensors to add to schemas as well as determining significant presentities within a given group of connected artefacts. This gives us a metric, we refer to as the Presentity Ranking between two presentities with attached context schemas. Within our architecture, applications are now provided with presentities with which to connect and derive new representations of context or discover new information sources such as sensors or actuators.

Our proposed solution is ad hoc permitting the addition and removal of sensors to both the infrastructure and to presentities while maintaining the ability to group and rank with respect to changes in context. We would eliminate the need to largely depend on spatial proximity and physical context tags as indicators of context, by permitting proximity derivation over all context dimensions or confined to any subset of context dimensions required for the optimum performance of an application or service. Unlike current means of finding and ranking context on the internet, all searches will be presentity-centric and in sole response to the current context.

Having considered the limitations our solution would encounter in a centralized implementation, we further discussed how it would be implemented on a distributed architecture enabling the inherent scalability required to support a future *Internet of Things*.

Future work on this includes deriving a large sample set capable of generating values for testing and bench marking. This would include creating applications on mobile phones or computers. Other work include the ability to dynamically index and rank sensors in terms of relativity to other sensors, entities and search queries across the system. By being able to usefully connect sensors and entities such as is with web content, we see the possibility of enabling a dynamic ranking and searching solution much alike modern search engines.

Aknowledgement. This research is a part of the MediaSense project, which is partially financed by the EU Regional Fund and the County Administrative Board of Västernorrland. The authors would like to thank Enrico Saviolli for his contribution to this research topic.

References

1. Google (2010), `http://www.google.com`
2. Bardram, J.E., Kjær, R.E., Pedersen, M.Ø.: Context-Aware User Authentication – Supporting Proximity-Based Login in Pervasive Computing. In: Dey, A.K., Schmidt, A., McCarthy, J.F. (eds.) UbiComp 2003. LNCS, vol. 2864, pp. 107–123. Springer, Heidelberg (2003), `http://www.springerlink.com/index/Q1MCV12D4N0B5X4L.pdf`

3. Bravo, J., Hervas, R., Chavira, G.: Modeling Contexts by RFID-Sensor Fusion. In: Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW 2006), pp. 30–34 (2006)
4. Chen, A.: Context-Aware Collaborative Filtering System: Predicting the User's Preference in the Ubiquitous Computing Environment. In: Strang, T., Linnhoff-Popien, C. (eds.) LoCA 2005. LNCS, vol. 3479, pp. 244–253. Springer, Heidelberg (2005)
5. Christein, H., Schulthess, P.: A General Purpose Model for Presence Awareness. In: Plaice, J., Kropf, P.G., Schulthess, P., Slonim, J. (eds.) DCW 2002. LNCS, vol. 2468, pp. 24–34. Springer, Heidelberg (2002)
6. Dobslaw, F., Larsson, A., Kanter, T., Walters, J.: An Object-Oriented Model in Support of Context-Aware Mobile Applications. In: Cai, Y., Magedanz, T., Li, M., Xia, J., Giannelli, C. (eds.) Mobilware 2010. LNCIST, vol. 48, pp. 205–220. Springer, Heidelberg (2010), `http://www.springerlink.com/index/UH0745GR616N7387.pdf`
7. Farringdon, J., Moore, A.J., Tilbury, N., Church, J., Biemond, P.D.: Wearable sensor badge and sensor jacket for context awareness. In: The Third International Symposium on Wearable Computers 1999. Digest of Papers, pp. 107–113. IEEE (2002)
8. Forsström, S., Kardeby, V., Walters, J., Kanter, T.: Location-Based Ubiquitous Context Exchange in Mobile Environments. Mobile Networks and Management (2010)
9. Göker, A., Watt, S., Myrhaug, H.I., Whitehead, N., Yakici, M., Bierig, R., Nuti, S.K., Cumming, H.: An ambient, personalised, and context-sensitive information system for mobile users. In: Proceedings of the 2nd European Union Symposium on Ambient Intelligence, pp. 19–24. ACM (2004)
10. Holmquist, L., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., Gellersen, H.W.: Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts. In: Abowd, G.D., Brumitt, B., Shafer, S. (eds.) UbiComp 2001. LNCS, vol. 2201, pp. 116–122. Springer, Heidelberg (2001)
11. Hong, J.I., Landay, J.: An infrastructure approach to context-aware computing. In: Human-Computer Interaction, vol. 16(2), pp. 287–303 (2001)
12. Joly, A., Maret, P., Daigremont, J.: Context-awareness, the missing block of social networking. International Journal of Computer Science and Applications 4(2), 50–65 (2009)
13. Kansal, A., Nath, S., Liu, J., Zhao, F.: Senseweb: An infrastructure for shared sensing. IEEE MultiMedia 14(4), 8–13 (2007)
14. Kanter, T., Pettersson, S., Forsstrom, S., Kardeby, V., Norling, R., Walters, J., Osterberg, P.: Distributed context support for ubiquitous mobile awareness services. In: Fourth International Conference on Communications and Networking in China, ChinaCOM 2009, pp. 1–5. IEEE (2009)
15. Lee, J., Song, J., Kim, H., Choi, J., Yun, M.: A User-Centered Approach for Ubiquitous Service Evaluation: An Evaluation Metrics Focused on Human-System Interaction Capability. In: Lee, S., Choo, H., Ha, S., Shin, I.C. (eds.) APCHI 2008. LNCS, vol. 5068, pp. 21–29. Springer, Heidelberg (2008), `http://www.springerlink.com/index/F234480028647884.pdf`, doi:10.1007/978-3-540-70585-7
16. Li, J., Loo, B.T., Hellerstein, J.M., Frans Kaashoek, M., Karger, D.R., Morris, R.: On the Feasibility of Peer-to-Peer Web Indexing and Search. In: Kaashoek, F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 207–215. Springer, Heidelberg (2003)

17. Presser, M., Barnaghi, P.M., Eurich, M., Villalonga, C.: The SENSEI project: integrating the physical world with the digital world of the network of the future. IEEE Communications Magazine 47(4), 1–4 (2009)
18. Ranganathan, A., Campbell, R.H.: A Middleware for Context-Aware Agents in Ubiquitous Computing Environments. In: Endler, M., Schmidt, D.C. (eds.) Middleware 2003. LNCS, vol. 2672, pp. 143–161. Springer, Heidelberg (2003)
19. Salber, D., Dey, A.K., Abowd, G.D.: The context toolkit: Aiding the development of context-enabled applications. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: the CHI is the Limit, pp. 434–441. ACM (1999)
20. Schmidt, A., Beigl, M., Gellersen, H.W.: There is more to context than location. Computers & Graphics 23(6), 893–901 (1999)
21. Sundmaeker, H., Guillemin, P., Friess, P., Woelfflé, S.: Vision and Challenges for Realising the Internet of Things. In: Cluster of European Research Projects on the Internet of Things, CERP-IoT (March 2010)
22. Zhu, Y., Ye, S., Li, X.: Distributed PageRank computation based on iterative aggregation-disaggregation methods. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, pp. 578–585. ACM, New York (2005)