Location Tracking / System Architecture / Sensor System

Technology Reports

Object Tracking System for Mobile Terminals: Architecture, Protocol and Its Evaluation

As a technology to contribute to NTT DoCoMo's future services, we have developed a mobile object tracking system that is based on a distributed architecture comprising two layers. It balances network load and is highly scalable.

Wolfgang Kellerer, Norihiro Ishikawa, Jörg Widmer and Zoran Despotovic

1. Introduction

The tracking of mobile objects such as persons is an important application for NTT DoCoMo to realize safety services, user guidance, or logistics service solutions. We have developed a system architecture for mobile object tracking based on distributed overlay network[*1] technologies. Our solution, which has been developed in a joint collaboration between NTT DoCoMo and DoCoMo Communications Laboratories Europe GmbH, covers two aspects of tracking in a hierarchical architecture: local and global tracking, leading to high scalability and cost efficiency compared to conventional, centralized solutions.

For tracking services in local areas, e.g., tracking children in a school, we investigate the feasibility of Peer-to-Peer (P2P) search strategies in an ad hoc network[*2] environment based on Institute of Electrical and Electronics Engineers 802.11 (IEEE 802.11)[*3] capable distributed nodes. For global tracking, a second tier structured P2P system interconnects all local tracking systems and allows a global query resolution. P2P is a networking technology, which maintains an overlay network among distributed nodes in a self-organizing manner. Our two-layered system consists of a local location tracking systems and a global location tracking system. These systems use P2P technology to look up and deliver shared resources.

In comparison to conventional data gathering [1], large scale tracking of mobile objects is challenging with respect to data management and localization. Data management for mobile objects is characterized by a high number of tracked objects and a high frequency of location changes. Conventional solutions as shown in **Figure 1** do not scale well due to heavy network load arising from updates sent to one central server, or they suffer from high inaccuracy if the update interval is too long (Update type in Fig.1(a)). These weaknesses can be avoided by flooding object queries to all nodes (Flooding type in Fig.1(b)). However, depending on the query frequency, this may cause unacceptable network load, leading to network instability and unavailability of data.

We describe a decentralized data management solution based on P2P concepts using a Distributed Hash Table (DHT) that combines the characteristics of both of the above extremes (updating and flooding). A DHT is a data structure which provides the following functionality: given a key it returns a data item (e.g. location of target object) associated with the key. DHT refers to a hash table which is distributed among a set of computers and does not need any central servers. The network nodes run a DHT system, into which tracked objects or sensors insert their data. The DHT enables large scale data distribution and retrieval without

*1 **Overlay network**: Logical network built on top of a physical network.

*2 **Ad hoc network**: A network in which several mobile terminals connect to each other without any dedicated infrastructure such as base stations or access points.

*3 **IEEE 802.11**: An international standard for wireless LAN defined by IEEE.

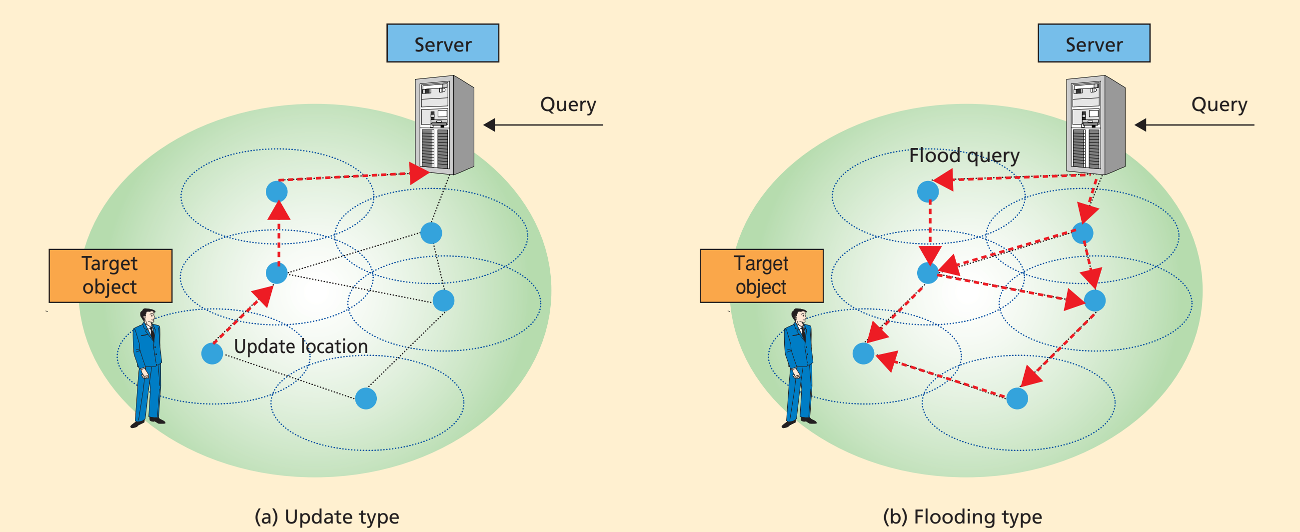

Figure 1 Conventional tracking systems

creating a bottleneck at one single node. For scalability, our concept comprises a two tier architecture which separates local tracking systems, that deal with frequent local updates, and global tracking which interconnects all the local P2P systems. Localization can be performed using various indoor (e.g., based on measurement of Received Signal Strength Indicator (RSSI)[*4] in IEEE 802.11) and outdoor (e.g., Global Positioning System (GPS)) technologies, thus building a distributed real time location system.

In this article, we describe our architecture and protocols for the local system, give an outlook on the global system, and present simulation scenarios and evaluate system performance.

We have implemented a prototype system of our architecture with NTT DoCoMo. For more information, please refer to the accompanying article titled "Object Tracking System for Mobile Terminals: Prototype System Using Cameras and Position Measurement Sensors", published also in this journal.

2. System Architecture

From an architectural point of view, we assume that our location framework is composed of a large number of local location systems corresponding to local areas in which the tracked objects usually move. To keep the data that is handled by the location tracking system providers (be they mobile operators or third parties) at a minimum, this data should not be propagated unnecessarily to other local systems. Therefore, the tracking system architecture comprises two hierarchically layered parts: the local tracking systems and a global tracking system that ties together these local systems but allows them to be kept logically separated. This architecture is shown in **Figure 2**.

While it is theoretically possible to use data collection and management algorithms [2] developed for sensor networks also for object tracking, the size of our system makes such an approach impracticable on a global level. However, it may be well suited on a local level, depending on the size of such local systems. Another advantage of a global system is that it can act as a gateway between such architecturally and technologically heterogeneous local subsystems.

As can be seen in Fig. 2, a mobile operator may offer the service to customers via a mobile terminal which is used to display location information of tracked objects and also additional information such as video or surrounding persons. It is also possible to query for the location of the objects directly

*4 **RSSI**: Level indicator of the signal power of the received signal at a receiver terminal.

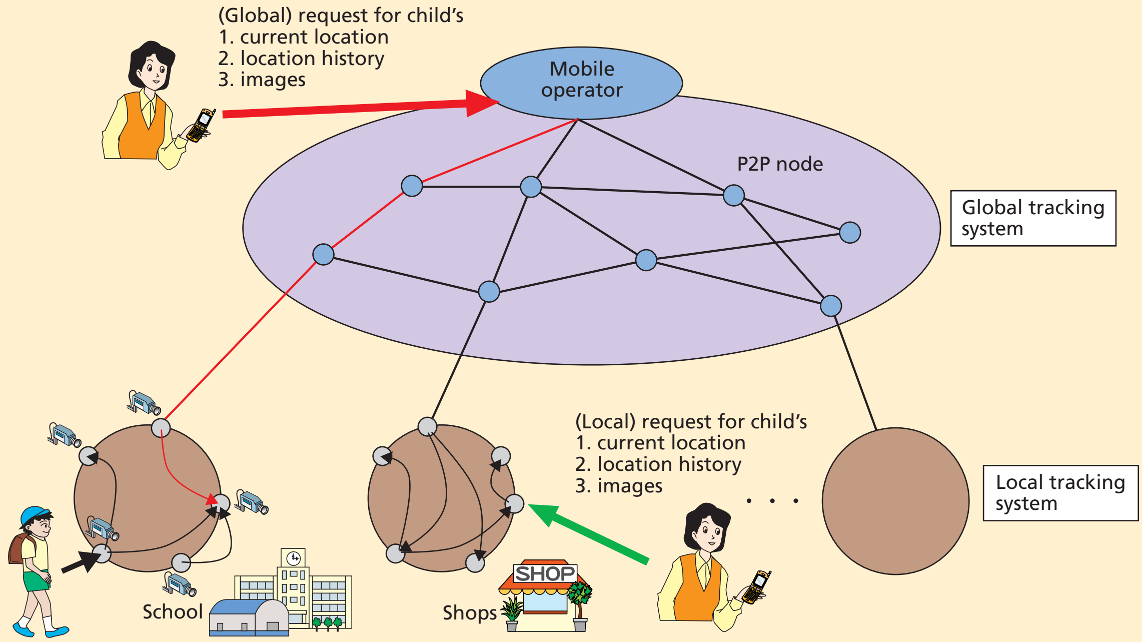

Figure 2 Two layer architecture of the location tracking system

within a local P2P system, using any short range communication technology (e.g. IEEE 802.11 or Bluetooth®*5) available in it.

2.1 Global Tracking System

The global tracking system consists of a P2P overlay network that can be provided by the mobile operator or by the providers of the local tracking systems collaboratively. For example, each location system provider might provide a node with global connectivity participating at the same time in the global system and the corresponding local system. However, also a centralized configuration of the global system is possible, in case the number of local networks that are connected together is not too large.

While a local system maintains detailed information about the tracked object at any time instant, the global network only registers the current local network of a tracked object. This means no update in the global network is necessary when a tracked object changes its location within a local area. The local network only notifies the global system in case a new object arrives in the local system or an object leaves. Thus the main purpose of the global system is to connect the local systems and to make data generated in them globally available.

2.2 Local Tracking System

Central to the architecture of the local tracking systems is a P2P system

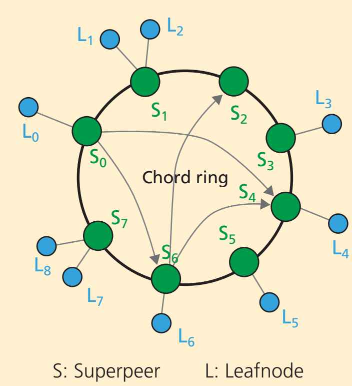

Figure 3 Hierarchical structured P2P architecture of the local tracking system

that itself is hierarchical, as shown in **Figure 3** [3]. It defines two different classes of peers: superpeers (S) and leafnodes (L). The superpeers establish a DHT-based P2P overlay in form of a Chord*6 ring [4].

Leafnodes maintain overlay con-

*5 **Bluetooth®**: Registered trademark of Bluetooth SIG Inc. in the United States.

*6 **Chord**: An algorithm to construct an overlay, which organizes nodes in a virtual ring (one dimensional logical space).

nections to their superpeer and communicate only with them. In contrast, superpeers perform multiple other tasks, e.g., they insert data objects (e.g. an observed object ID and its location) generated either by themselves or by the attached leafnodes into the overlay network and act as their owners. When performing a lookup, a superpeer resolves the object's key with the search functionality of the Chord overlay, determines the responsible superpeer by using DHT unique hash function[*7], and forwards the resulut to the lookup originator.

We stress that this is a logical view on the system configuration. However, there are several design choices to be made, when mapping this logical configuration to the physical network. An example for this is the question: how many superpeers to use and where to place them, i.e., which nodes in a given sensor network should be superpeers and which ones should be leafnodes? We plan to investigate such questions in the future.

Our goal in this configuration of the local P2P network is to achieve a good load balancing across the nodes. In this way we hope to enlarge the range of parameter values in which the system can still operate successfully (e.g., query rate or update rate). Thus, we select a number of sensor nodes close to the center of the sensor field to be superpeers, while all other sensor nodes become leafnodes and attach to their closest superpeer (Note that these connections are logical, not physical). This way we avoid a communication bottleneck in comparison to using only a single server to store the potentially large amount of data generated by the tracked objects.

3. Location Management in Local Tracking Systems

The above setup assumes that sensor nodes are active (i.e., they determine the location of an object) while tracked objects have a passive role. In this case, only sensor nodes become superpeers or leafnodes. However, it is also suitable for settings in which the tracked objects take a more active role. Our location management system [5] implies that tracked objects are active in the sense that they determine their position through RSSI measurements themselves. After the tracked object has determined its own position, it inserts it into the system through a nearby superpeer node (e.g., access point). With a centralized solution, a common design choice is to have base stations do the RSSI measurements and send their data to a central server, which finally determines the position of the tracked object and stores it. However, our solution with client-side measurements is considerably cheaper than this centralized one. We slightly raise the cost of the tracked objects because our solution uses active objects which have to be capable of running the described software to determine their positions, while the existing centralized solutions use passive objects. On the other hand, our solution can be realized by cheap low end access points running the DHT software, while the centralized solutions require expensive access points and a central server in order to determine the position through RSSI measurements.

An important design decision concerns the format of the data maintained in the network, as well as the indices that need to be built. The decision has been made based on how we can efficiently answer the two main query types (the other query types can be easily derived from these two):

- Where is the tracked object at the moment?
- Which other objects are in the proximity of the tracked object at a given moment?

To this end, we use [object ID, superpeer address and time stamp] as the format of the data maintained in the DHT, with one index built on the object ID attribute. Here, the superpeer address denotes the address of the superpeer that inserted the tuple, i.e. the superpeer through which the original data from a sensor or a tracked object was routed. This superpeer is responsible for storing other relevant information such as location and other nearby objects, etc.

The two queries can be now per-

*7 **Hash function**: A reproducible method of mapping some kind of data to a (small) value of fixed length. It has the property that it is difficult to infer the original data from the hash value.

formed as follows. Given an ID, a superpeer holding the most recent data tuple with the ID is retrieved first. Then, this superpeer is contacted to retrieve the needed information such as location or IDs of other objects. The IDs of other objects are needed for the second type of query. We believe that this type of indexing is the most suitable one. The reason is that the amount of traffic generated in the P2P network is low because only objects moving across superpeers need to be registered in the P2P network.

4. Simulations

For the performance analysis, we use an ad hoc network topology with IEEE 802.11 transceivers as a worst case scenario. Given the tradeoff between maintenance cost and management advantages of a distributed P2P system, we are mainly interested if P2P works even in a worst case setting and has similar performance as a centralized approach. Settings with mixed wired and wireless devices are expected to give better performance and are a focus of future work.

The simulations are done using the ns-2 network simulator[*8]. We use a static grid topology with 100 sensor nodes (or peers) and configure it such that each node can communicate with its four direct neighbors. Furthermore, there are 100 mobile objects that move randomly at pedestrian speed. Mobile objects themselves do not communicate, but sensor nodes can detect the presence of such objects if they are nearby.

Our hierarchical DHT protocol is executed in the environment discussed above. We chose a superpeer/leafnode ratio of 1:9 (i.e., there are 10 superpeers and 90 leafnodes in our topology), with the superpeers placed in the middle of the simulated area, and leafnodes connected to the nearest superpeer. Upon object detection, the time of sighting along with the IDentifier (ID) of the object is inserted into the DHT system at the node whose ID matches the hash of the object's ID. Queries for an object are then routed to this responsible node, which replies with the current location.

We compare our DHT system against a simple centralized query approach in which a single server queries all nodes about the current location of an object (using flooding). We refer to this as the simple query algorithm. Since location information is kept at the node at which it originates, no update traffic is necessary.

The simulation results are shown in **Figures 4** and **5**. DHT overhead is proportional to the number of update data and queries. We measure the overall traffic in terms of Media Access Control (MAC) layer[*9] transmissions per node as well as the packet delivery ratio under varying query rates and update rates. In our scenario, the overall query rate depends on how often an object location is queried, whereas the update rate directly depends on the mobility of the objects. An update occurs, each time an object has moved for a certain distance. We assume that updates happen more frequently than queries. Since there is no update traffic with the centralized simple query algorithm, we only vary the update rate for the DHT algorithm.

We see from Fig. 4 that the main source of traffic in the simulated network is the update traffic, not the query traffic. This is immediately apparent, as the number of queries only ranges from 0.1 to 10 queries per second for the whole network, while the number of updates ranges from 10 to 250 updates per second. As the query rate increases (along the x-axis) there is only a marginal increase in traffic, but there are significant differences in overhead for the DHT algorithm with different update rates.

Looking at the packet delivery ratio for the simple query algorithm in Fig. 5, we can see that it drops significantly at a query rate of 3 queries per second. Even though the overall load in the network is lower than for the DHT algorithm, traffic concentrates around the central server and leads to overload and high loss rates in this area. At the same time, the DHT maintains more than 80% packet delivery ratio for update rates below 100 updates per second.

Overall, the simple query algorithm performs better at very low query rates, where the DHT algorithm has a higher

*8 **ns-2 network simulator**: A program that enables simulating functioning of an arbitrary computer network.

*9 **MAC layer**: A layer that has a control function for preventing packet collisions when sharing communication lines among multiple nodes. This layer is a sublayer of the data link layer in the OSI 7-layer model.

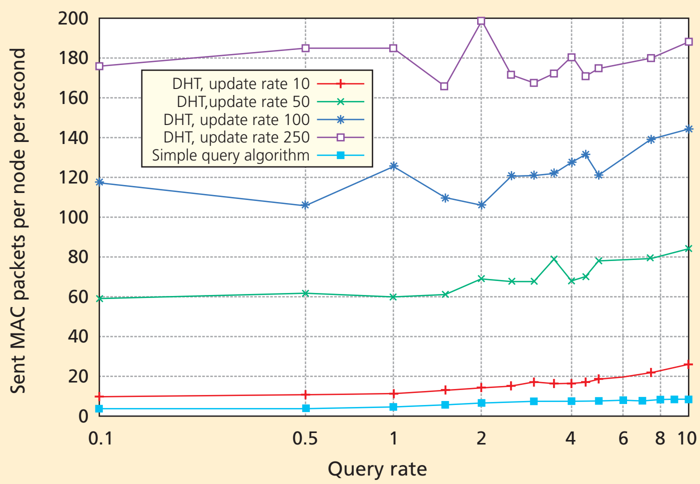

Figure 4 Overhead in terms of transmissions per node

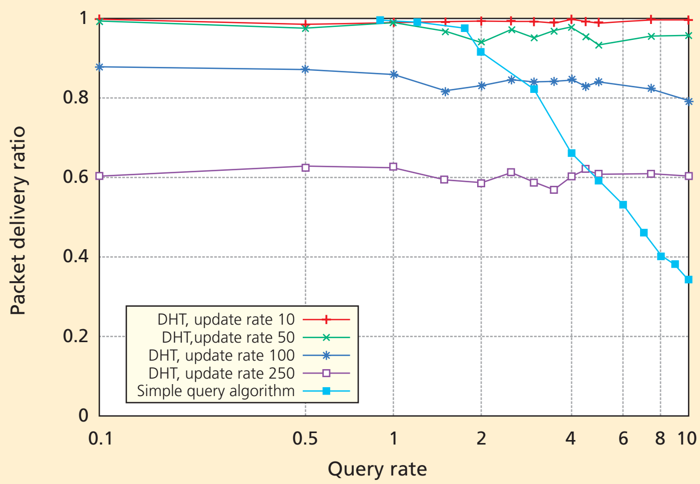

Figure 5 Packet delivery ratio

overhead. However, we note that the DHT algorithm was not optimized for an ad hoc network setting and we believe that an improved algorithm (which, for example, determines which sensor should be superpeer or leafnode) can achieve the same performance with a significantly reduced overhead. More importantly, the DHT algorithm avoids a concentration of data around any one node and is able to maintain high delivery rates for much higher query rates than the centralized approach. As a conclusion, the DHT algorithm is a good solution for high query rates (e.g., 10 queries per second), given that the updated rate is below 100 updates per second. This provides more flexibility concerning the scenarios the architecture can be used for and fits well with the inherently local generation of location data.

5. Conclusion

In this article, we described a distributed system architecture for tracking mobile objects based on a hierarchical structured P2P system and a real-time location system. We demonstrated that this distributed solution to mobile object tracking matches the inherent decentralization stemming from the environment itself very well and is, therefore, more appropriate than centralized solutions in the considered scenario.

References

[1] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann and F. Silva: "Directed Diffusion for Wireless Sensor Networking," Transactions on Networking, 2003.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci: "A survey on sensor networks," IEEE Commun. Mag., Vol.40, No.8, pp.102-114, 2002.

[3] S. Zoels, Z. Despotovic and W. Kellerer: "Cost-Based Analysis of Hierarchical DHT Design," in proc. 6th IEEE Intl. Conference on P2P Computing, 2006.

[4] I. Stoica, R. Morris, D. Karger, M. Kaashoek and H. Balakrishnan: "Chord: A scalable peer-to-peer lookup service for internet applications," in SIGCOMM, 2001.

[5] M. Michel, Z. Despotovic, W. Kellerer, Q. Wei, J. Widmer, N. Ishikawa, T. Kato and T. Osano: "P2P search routing concepts for mobile object tracking," in MOBIQUITOUS 2007 (Poster), 2007.