

Using Context Awareness to Improve Quality of Information Retrieval in Pervasive Computing

Joseph P. Loyall and Richard E. Schantz

BBN Technologies, Cambridge, MA 02138, USA
{jloyall, schantz}@bbn.com

Abstract. Publish-subscribe-query information broker middleware offers great promise to users of pervasive computing systems requiring access to information. However, users of publish-subscribe-query information broker middleware face a challenge in requesting information. The decoupling of publishers and consumers of information means that a user requesting information is frequently not aware of what is available, where it comes from, and when it becomes available. *Too specific* a request might return no results, while *too broad* a request might overwhelm the user with a combination of useless and buried useful information. This paper investigates using *context*, such as a user's location, affiliation, and time, to automatically improve the quality of information brokering and delivery. Augmenting an explicit client request with contextual clauses can automatically prioritize, order, and prune information so that the most useful and highest quality among the information available is delivered first. The paper provides techniques for augmenting client requests with context, techniques for combining multiple contextual aspects, and experiments evaluating the efficacy and performance of those techniques.

Keywords: Context awareness, location based computing, quality of service, publish-subscribe-query information management

1 Introduction

Computing devices, including PDAs, GPSs, iPods, cell phones and laptops, as well as those embedded inside automobiles, appliances, and smart buildings, permeate our daily lives. Many of these computing devices are interconnected and more are becoming so, frequently through wireless connections and increasingly in an ad hoc manner. The devices and users are often mobile and it is not always obvious what devices are present and reachable. The ubiquity of computing and connectivity leads to unprecedented data access and variety confronting all users of mobile, embedded, and internetworked systems.

Advanced information brokers have emerged to help navigate the *Digital Whitewater*

This work was supported by the USAF Air Force Research Laboratory under ITT contract number SPO900-98-D-4000, Subcontract Number: 205344 – Modification No. 4.

of the expanding volume and variety of data available in ubiquitous systems. Based on publication-subscribe-query operations, these *information management services* offer active information management, enabling information providers to make information available (i.e., through publication) and information consumers to request future (i.e., subscriptions) or past (i.e., queries) information based on type, attributes, or content. A user requesting information is frequently not aware of what is available, where it comes from, and when it becomes available.

This presents new challenges in the area of traditional *quality of service (QoS)*, typically concerned with the management of resources such as CPU, bandwidth, and memory. In addition, the limited cognitive resources of users can be overloaded with the tasks of discovering, processing, and consuming the vast amounts of information available to them. Users of publish-subscribe-query information services face a challenge in crafting requests to discover and deliver needed information. *Too specific* a request might return no results, while *too broad* a request might overwhelm the user with information, burying the most useful results in a sea of less useful, even superfluous, information.

As an example, consider a user planning a route through an urban area and who needs maps or imagery of the possible routes he is likely to take as illustrated in Fig. 1. For this user, information that is recent, that is from trusted sources, and that is delivered in the order of closeness to his current location is much more likely to be of greater use than information that is older, less trustworthy, or more distant. However, with publish-subscribe-query systems, the user does not necessarily know what information is available, when it was created, by whom, or which locations it covers. It is challenging for the user to craft a request to get the best information available. If he crafts a request asking for any imagery available of the city, he might get a large number of responses, with a mix of imagery that is recent and old, for intermixed locations close to and distant from his location, and from a variety of sources that are known and unknown. Conversely, if he crafts a request that asks for imagery in a specific timeframe, location, or from particular sources, he might receive no information. This challenge affects the quality of information request and delivery in the following ways:

- Increases cognitive burden and effort on the user to examine a potentially large set of results to find the most useful ones or craft a set of precise requests that will be issued simultaneously or sequentially until useful results are received.
- Decreases resource efficiency, e.g., memory is needed on the user's device to hold a large set of results and bandwidth is used to transmit a large set of results or requests only some of which are useful.
- Adds delay due to the time required for returning and processing a large set of results or for issuing multiple

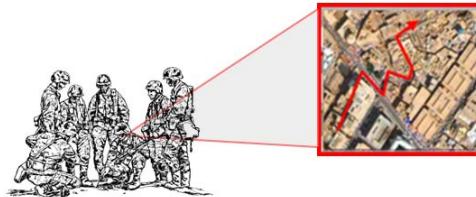


Fig. 1. A user planning a route through an area requires information about the area without knowing all the potential sources of the information.

increasingly refined requests and awaiting their responses.

Furthermore, when considering multiple aspects of information, the decision of what information is the most useful in response to a request is often not obvious. Returning to the example of urban route planning, which of the following images is more useful to the requesting user?

- An image that was collected today, but miles away from the user or another image that is of the location that the user is in, but is a year old.
- An image that is very recent, but taken by some unknown person's camera (and has gone through any unknown amount of doctoring), or an image much older but collected by official government or industry communication satellites.

Characteristics of the user and his situation form the *context* in which his explicit requests are made and are useful to help determine the characteristics of information that are the most useful. Contextual information about the user and his situation can affect what he needs, how he perceives a particular response, and information ordering. Contextual information can be used to supplement explicit requests, to refine broad requests or to broaden narrow requests, and to prune, prioritize, and sort information to better suit the user's situation and needs. Automatically gathering context and using it as part of information requests and dissemination is an important step towards a more proactive computing vision for embedded and ubiquitous computing environments.

Information Management Services (IMS). As a starting point frame of reference for the investigation into context aware capabilities, and to seed prototype experimentation, we use a US Air Force reference implementation of IMS [1, 2, 7] developed for tactical and enterprise net-centric operations. Information is in the form of typed *Managed Information Objects (MIOs)* consisting of a data payload and metadata describing the object and its payload. Requests for information, i.e., subscriptions or queries, are expressed using *predicates* over MIO types and metadata values, expressed in query languages such as XPath or XQuery.

The core set of IMS, illustrated in Fig. 2, includes the following:

1. *Submission service* receives MIOs published by clients.
2. *Predicate evaluation* matches the metadata of MIOs to predicates registered by information subscribers.
3. *Archive service* inserts MIOs into an MIO data store.
4. *Query service* processes queries consisting of predicates over the metadata of archived MIOs.
5. *Dissemination service* sends query results to querying clients and MIOs that match subscription predicates to subscribing clients.

There are three places in the IMS architecture in which contextual information can be used to order and filter MIOs to improve the quality of information dissemination:

- *Query service operations to affect the order and number of query results.* Contextual information can be used to determine which results to return to the query client and the order in which to return them to be of most use, i.e., of highest quality, to the client. Once a sufficient number of high quality results have been returned, the remainder can be pruned.

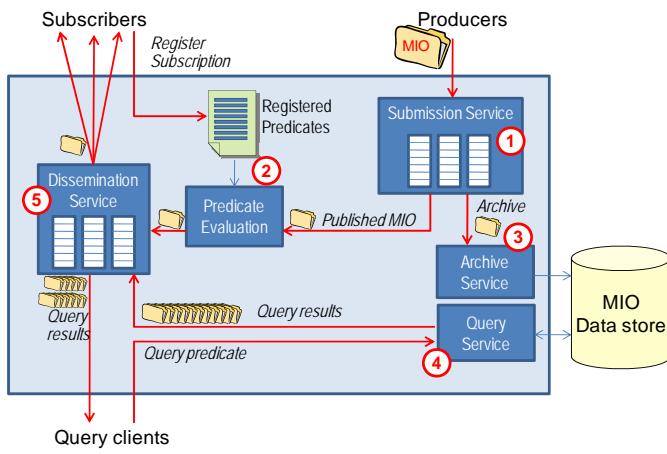


Fig. 2. The services involved in information management that can benefit from using context information.

- *Dissemination service queues to affect the order of delivery of MIOs.* When the outgoing bandwidth is insufficient to keep up with the number of MIOs that need to be disseminated and the number of clients to whom they need to be disseminated, the dissemination service enqueues the MIOs. Contextual information can be used to order and filter the MIOs in these queues.
- *Registered subscription predicates to prioritize and refine predicates improving the quality of MIO brokering.* Just as contextual information can be used to refine and order the results of queries, context can be used to refine and order predicates to be evaluated.

Aspects of Context. Although there are many elements of context that can be considered, the following are particularly relevant for ubiquitous computing and mobile, tactical users, and drive our initial investigation:

- *Location* – A user's position or the position of an area of interest, frequently expressed as latitude, longitude, and altitude. Many existing devices include the ability to detect their locations, using Global Positioning System (GPS) and Enhanced Observed Time of Difference (E-OTD), which provide absolute position, and WLAN and RFID, which provide relative position.
- *Time* – Whereas location indicates a user's position in space, there is a corresponding position in time. More recent information is likely to be more useful in general, while a user's time zone and related aspects (night or day, winter or summer) can have an effect on QoS requirements and perception.
- *Affiliation* – A user's affiliation can also affect QoS requirements and perception, e.g.,

in terms of trust. Information from an affiliated source, or closely related source, is more likely to be trusted. For example, in a military situation, a warfighter is more likely to trust information from his own unit or command than from a coalition partner, and more likely to trust a coalition partner than an unknown or opposing source.

These three contextual attributes are easily quantified and represent core attributes upon which to build a context aware approach for QoS. Location is increasingly being exploited in *location-based services* and *location aware computing* [3, 4]. Time corresponds closely to the core QoS property of *timeliness* (i.e., latency and performance), and affiliation contributes to the core QoS property of *trust*.

Other contextual attributes are less easily quantifiable and less directly mapped to QoS characteristics, but are useful to consider in future work as we expand our vision into the notion of generally intelligent and anticipatory information management capabilities. These include *user activity* (what a user is doing), *user mood* (e.g., is the user distracted, fatigued, impatient, or angry), *situation awareness* (the things going on around the user), *equipment* (what the user is carrying or has), *user attention* (the amount of attention that the user is able to provide), *user preferences* (what a user prefers, wants, or needs in his or her current situation), and *administrative domain* (the domain that a user is in or connected to).

The challenges associated with inserting context awareness toward improving quality of service for information management include where and how to incorporate context awareness, selecting the aspects of QoS to incorporate, establishing the appropriate granularity of contextual values, determining the added value of combinations of contextual attributes, and how to (and how frequently to) automatically gather or update the contextual values themselves. The remainder of this paper touches upon some of these subjects as a first step in exploring this rich and emerging field of research.

2 Incorporating Context into Information Requests

In this section, we explore techniques for automatically converting explicit information requests originating from a user into augmented requests that prioritize results based on just a single dimension of context, e.g., time (most recent results first), location (closest results first), or affiliation (most trusted results first). We then discuss techniques for combining contextual aspects.

Request Expansion. A straightforward technique, *Request Expansion*, converts a single explicit user request into a set of requests, each with extra clauses that match ranges of contextual values. The full set of new requests would match all (or approximately all) the MIOs matched by the original request, but each of the new requests would return only a subset of the matches, with the first of the new set returning those MIOs that are most appropriate or desirable based on the added context, with each subsequent request in the new set returning the next set of most appropriate MIOs. For example, a single user supplied request can be turned into multiple context added requests that additionally match on

- Time, each predicate representing a range of time, e.g., an hour, day, week, or month.
- Location, each predicate representing a particular distance, e.g., a mile, from a point.

This approach is easy to implement and it can be efficient. Once an expanded request has returned results, no subsequent results are going to be better from the perspective of that context, so the processing of the remaining expanded predicates can often be avoided, especially if the results completely satisfy the user's need, the resources can be better used for other requests or operations, or there is a need to reduce the cognitive overload of too many results.

However, the approach comes with tradeoffs. The efficiency and effectiveness depend on the granularity chosen for the contextual attribute. This technique provides only an approximation of the optimal order of contextual value, more or less optimal based on the granularity of the contextual ranges chosen. Finer granularity results in closer to optimal order of results, but more predicates to process (resulting in lower performance). Coarser granularity takes less time to execute because there are fewer predicates to process, but the results are also more coarsely ordered.

As an example, Fig. 3 shows an expanded set of requests based on location that expands from the requesting client (represented by the "R" in the center of the image) one latitude and longitude minute at a time. Any information object that matches a predicate representing an inner bounding box will be roughly closer in proximity to the requesting client than an information object matching a subsequent predicate representing an outer bounding box, and will be considered higher quality with regard to the location context.

The request expansion technique addresses several of the challenges described in Section 1, namely (1) the user provides a request only about *what* he needs, limiting the complexity of his request; (2) some results will be returned to the client, if any are available within the ranges specified; and (3) if a *best* set of results are



Fig. 3. A subscription by a client (R) can be turned into multiple subscriptions, each matching on the original predicate and location inside a bounding box. The closer the bounding box to R, the higher quality the resulting matched information is.

available as determined by the added context, e.g., very recent, close, or trusted, they can be prioritized accordingly.

Ordering as a Feature of Query Languages. Another technique uses existing query languages, e.g., *XQuery*, to prioritize and sort information. We automatically add clauses to an explicitly made user request that order on ranges of contextual values. XQuery provides an “order by” operator which uses capabilities of the underlying database to sort the responses.¹ This needs to be applied in a place in the overall system design where there are a set of MIOs to sort, such as in the archive repository, dissemination service queue, or submission service queue. It is less useful in the predicate evaluation service, where only one MIO is available at a time.

In contrast to the request expansion approach, this technique produces an optimal ordering among the MIOs available for sorting but is less universal.

Context Templates as an Automation Mechanism. We developed a set of templates as a proof of concept approach serving two purposes. First, the templates provide a means for automatically extending explicitly provided user requests with the additional contextual clauses, using either of the approaches above. Second, the templates are composable, so that we can also easily combine multiple contextual attributes in an automated fashion. The templates take arguments indicating the predicate being extended, a Boolean indicating whether the call to the template is the first or is nested in a composed set of template calls, and additional arguments indicating granularity and starting points (e.g., current location or time).

Combining Contextual Attributes. While using a single contextual attribute to focus information brokering and dissemination can improve the quality of information, the more expected case will utilize multiple aspects of context. For example, it might be useful for a user to have information that is close in location to his position, but not if it is out of date or from an untrusted source. Likewise, recent information that is distant in location from the user's position might be less useful than something a little older but much closer. What a user typically needs is information with the right combination of contextual attributes that indicates that it is of the highest value to his current goals. Supporting this, however, requires being able to key on multiple simultaneous aspects of context and to be able to compare and sort the combinations in a meaningful way.

A key tradeoff in combining contextual aspects is deciding how the aspects are weighted relative to one another, i.e., whether one or more contextual aspects are more important than others, whether they are equal in importance, whether the relative impor-

¹ In practice, the information management services that we use utilize XPath [12] for the predicate language, instead of its superset XQuery [13], because of the latter's ability to modify or delete archived information, a security risk that breaks the semantics of the information broker in which subscribing and querying clients only have *read* access to the published and archived MIOs. If this restriction is relaxed, or enforced by other means such as examining requests for specific language clauses that indicate modification or deletion, then features of XQuery, or another expressive and Turing complete [5] predicate language, can be used to order and filter results based on complex relationships, such as contextual information.

tance is unknown, or whether the relative weighting is itself “context sensitive”.

If there is an order of importance, then the composable templates described above can be used to sort on multiple contextual attributes sequentially. The template is called multiple times in order, sending the original user request to the first call and the result of the first call to the second, etc. This automatically generates an extended predicate that orders by the first contextual attribute followed by ordering by the second contextual attribute, and so forth². This works well when the contextual attributes can be totally ordered and when they are have coarse granularity. Finer grained attributes that have only one or a very few elements in each unique value will not have any effect when they sort on the additional attributes.

When contextual attributes are of equal (or unknown) importance, one can examine the intersection of the matching sets for individual contextual aspects. That is, execute multiple, interleaved requests, each incorporating a single contextual aspect, and then aggregate the order of delivery of MIOs across the sets. This approach works with either the request expansion or query language ordering approach, but requires running multiple predicate evaluations or executing multiple queries, essentially trading off more execution time for a smaller, more tailored result set. This is a suitable tradeoff when CPU at the server is plentiful, but bandwidth to the requesting client is not, a situation to be expected in mobile, tactical environments. Furthermore, evaluation of the requests can be halted once the intersection set is large enough.

3 Example and Evaluation

To illustrate and evaluate incorporating context into requests, we created an operationally relevant experimental scenario, shown in Fig. 4, and database of MIOs. We developed a requesting client and a set of ten publishing clients, each simulating a moving sensor (e.g., an unmanned aerial vehicle, UAV) placed randomly within the confines of Baghdad, Iraq, heading in a randomly chosen direction “traveling” approximately 30 mph, and publishing imagery information of the location over which the UAV is “flying” every 12 seconds. The requesting client was considered a “US Force” and each UAV was assigned an *affiliation* randomly from a set of 14 ordered choices (e.g., US, UK, or Iraqi force). The requesting client and each published MIO were tagged with time, location, and affiliation contextual values. Fig. 4 shows the simulated location of the requesting client (“Requester”), the starting location of the 10 publishers, the direction each is traveling, and their affiliations.

² For example, the following pseudocode uses the templates to order by Time context to the “month” granularity, followed by ordering by Location to the finest granularity (Lat-Lon second):

```
tmpredicate = tx.extendPredicateWithTimeContext(orig_predicate, "month", true);
tmlocpredicate = ctx.extendPredicateWithLocationContext(tmpredicate, client_lon, client_lat, false);
...
executeQuery(tmlocpredicate);
```

Fig. 5 and Fig. 6 show the results from automatically incorporating location context into the information search request using the request expansion and query language ordering approach, respectively. The results of request expansion track the optimal results closely, but not perfectly due to the selection of granularity of the expanded requests. Likewise, the query language ordering technique tracks the optimal distance context closely, but not perfectly. In this case, the difference is because the XQuery ordering we used is based on the *Taxicab* distance calculation [6] rather than the optimal Euclidean distance.

While the inclusion of context can increase the amount of time (between 1% and 20% longer to execute the requests in our experiments) to get all the results, a more insightful question to ask instead of how long it takes to get all results is how quickly a sufficient set of results useful to the requester are available (or, conversely, how many useful results can be retrieved within a specific deadline). As the graph in Fig. 7 shows, the 100 closest, most recent, or most trusted MIOs are available in less than six seconds in any of the methods.

Fig. 8 shows an approach to provide a set of results with the best overall combination of the time, location, and affiliation contextual attributes, where *best* is defined to be the intersection of the result sets for each individual contextual attribute delivered in order of the normalized summed contextual value. After 325 results ordered by the individual contextual attributes, there were 34 MIOs common to each set. Fig. 8 shows these graphed in

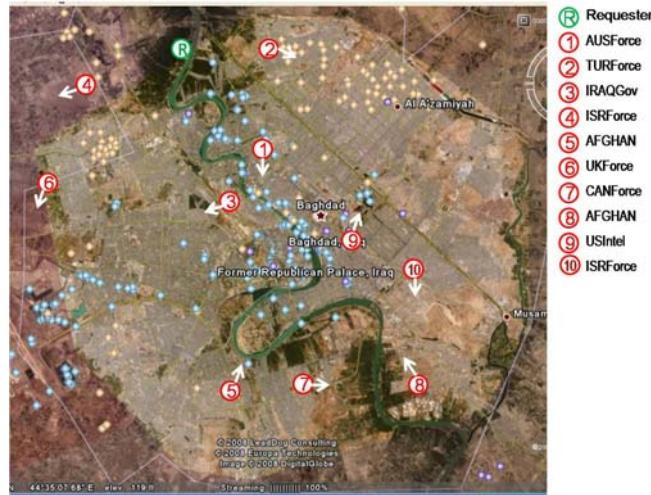


Fig. 4. The experimental database was populated by ten publishers located roughly within the confines of Baghdad, Iraq, moving in one of twelve directions, publishing simulated imagery MIOs.

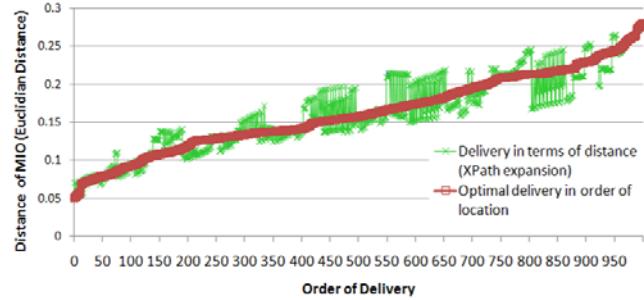


Fig. 5. XPath expansion by location graphed against optimal distance.

the order of their normalized summed values. The thick purple line (marked with “ \times ”) is the combined normalized value, with the other lines representing the individual normalized attribute values. These 34 MIOs represent the order of delivery of MIOs with the *best combination of* recentness, distance from the requesting user, and trust. The intersection technique to create this combination is a promising technique for combining even significantly independent contextual attributes like the ones that we investigated.

4 Related Work

Other work in context awareness has targeted pervasive computing environments and database queries. *Context Query Language, CQL*,

is a specific query language [10] for gathering contextual information in pervasive computing environments, and could therefore be used to collect the contextual information used to enhance the predicates in our approach. In [8], Mansour et al fashion queries in mobile applications as relational algebra, including tags for contextual information. Context is defined as XML documents and queries are automatically formed by middleware that inserts values from the XML context into the queries. In [11], Seher et al build an ontology of contextual information that is used to tag queries to refine them. In [9], Norrie et al also

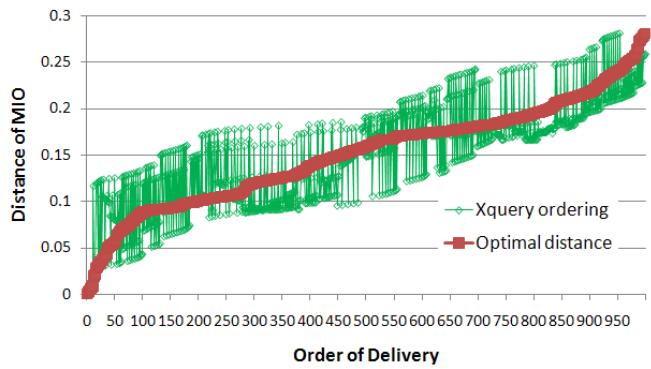


Fig. 6. The order of MIO delivery using an XQuery request with location context graphed against the Euclidean distance of each MIO from the requesting client (smaller is better).

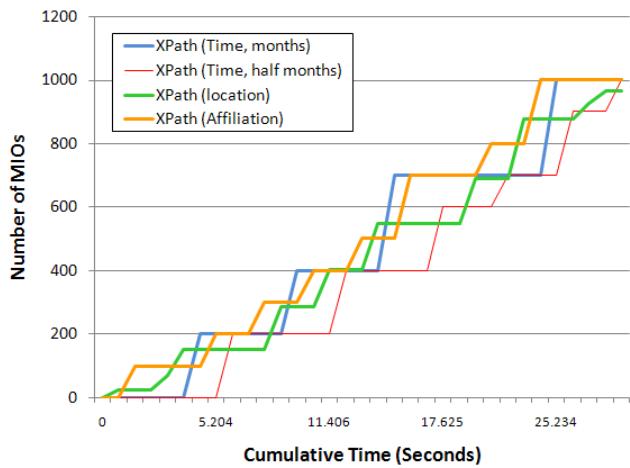


Fig. 7. Cumulative time it takes for MIOs to be delivered in the context-based request expansion.

deal with context in terms of database queries, but define “context” as the part of the schema needed to interpret a query in a local database that needs to be distributed with a query to enable it to be interpreted across federations.

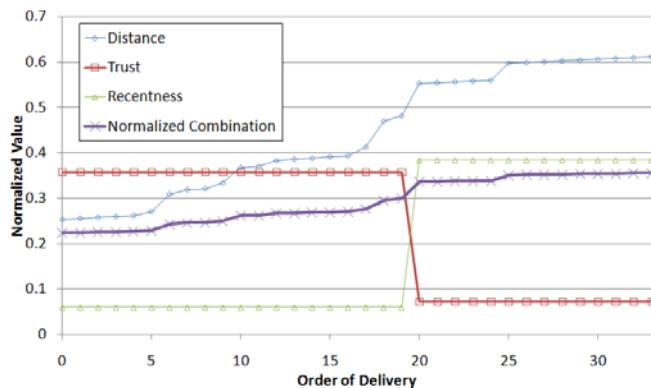


Fig. 8. The results if the intersection of the first 325 results ordered by distance, recentness, and trust are returned in order of their normalized summed values.

5 Conclusions

This paper explores the space of using context awareness to improve the quality of information management. It provides the following major contributions:

- Identification of ways in which context can be used to automatically streamline and improve information retrieval and dissemination.
- Strategies for automatically using context awareness to improve the quality of processing information requests and quantitative evaluations of the strategies.
- Approaches and metrics for combining context attributes.

Context awareness and the use of contextual information as part of an overall set of information management services can provide the following improvements in the quality of information discovered, brokered, and disseminated:

- Reduce the manual burden of crafting appropriate information requests.
- Reduce the cognitive load on the user to choose results that are most useful.
- Improve the speed by which useful results get to the user and ensure that the earliest results are more likely to be the most useful.
- Reduce demand on resources and increase the effectiveness of resource usage.

Our investigations illustrated that some (maybe many) context attributes are independent and have other aspects (such as granularity) that make it challenging to combine them into a single measure of context useful for improved quality. We presented a few techniques for combining attributes that are promising, including composable templates that support sorting on multiple keys. The intersection approach we present is more versatile (supports any granularity and doesn't require ordering the context attributes) but comes with the tradeoff of requiring processing of more requests.

The next steps for this research include applying and building upon the results of this study to incorporate context awareness into emerging information management (IM) ser-

vices. An overarching result of our investigation is evidence that context awareness can improve the usefulness of information brokers and reduce the burden on the users of IM services. The results of this investigation can be used to influence the design, configuration, and implementation of IM elements to make them context aware. Contextual information can be incorporated in MIOs and in MIO database indexes, so there is ready access to context information and faster retrieval and ordering based on context. Similarly, the structures underlying predicate registration and predicate matching can be organized to support context-aware pattern matching, prioritization, and filtering. Finally, context awareness can be built into the IM brokering services, making IM operations more context aware, responsive to client needs and preferences, proactive, and effective.

References

1. V. Combs, R. Hillman, M. Muccio, R. McKeel. Joint Battlespace Infosphere: Information Management within a C2 Enterprise. Tenth International Command and Control Technology Symposium (ICCRTS), 2005.
2. R. Grant, V. Combs, J. Hanna, B. Lipa, J. Reilly. Phoenix: SOA Based information Management Services, SPIE Conference on Defense Transformation and Net-Centric Systems, Orlando, FL, April 13-17, 2009.
3. Anamika Das. Location Based Services, A Status Report of South East Asia. Location, July-August 2008.
4. Sanjay Deshmukh, John Miller. Location-Aware Computing. Deviceforge.com (<http://www.deviceforge.com/>), December 3, 2003.
5. Stephan Kepser. A Simple Proof for the Turing-Completeness of XSLT and XQuery. *Proceedings of Extreme Markup Languages*, Montreal, Quebec, Canada, August 2-6, 2004.
6. E.F. Krause. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. NY: Dover, 1986.
7. M. Linderman, B. Siegel, D. Ouellet, J. Brichacek, S. Haines, G. Chase, J. O'May. A Reference Model for Information Management to Support Coalition Information Sharing Needs. Tenth International Command and Control Technology Symposium (ICCRTS), 2005.
8. E. Mansour, H. Höpfner. Towards an XML-Based Query and Contextual Information Model in Context-Aware Mobile Information Systems. 10th Intl. Conf. on Mobile Data Management: Systems, Services and Middleware, 2009.
9. M.C. Norrie, D. Kerr. Universal Contextual Queries in Database Networks. 3rd Intl. Conf. on Cooperative Information Systems, 1995.
- 10.R. Reichle, M. Wagner, M.U. Khan, K. Geihs, M. Valla, C. Fra, N. Paspallis, G. Papadopoulos. A Context Query Language for Pervasive Computing Environments. Sixth Annual IEEE Intl. Conf. On Pervasive Computing and Communications, 2008.
- 11.I. Seher, A. Ginige, S.A. Shahrestani. A Domain Identification Algorithm for Personalized Query Expansion with Contextual Information. 6th IEEE/ACIS Intl. Conf. on Computer and Information Science, 2007.
- 12.World Wide Web Consortium, *XML Path Language (XPath) Version 1.0*, W3C Recommendation, November 16, 1999, <http://www.w3.org/TR/xpath>.
- 13.World Wide Web Consortium, *XQuery 1.0: An XML Query Language*, W3C Recommendation, January 23, 2007, <http://www.w3.org/TR/xquery/>.