AD LOC: Collaborative Location-based Annotation

DEREK J. CORBETT[†] and DANIEL CUTTING[†]

AD LOC is a novel system for mobile devices to collaboratively tie persistent virtual notes to physical locations. Notes that are relevant to particular locations can be created and then cached using serendipitously formed one-hop wireless ad hoc network connections. The location provides an address to which the information is relevant and devices attempt to keep the information stored at devices which remain close to this address. In simulation experiments, even under conditions of nodal flux, it is possible to retrieve over 90% of the notes that have been stored.

1. Introduction

People are using mobile devices for an increasingly diverse collection of activities including gaming, storing and sharing contact details and schedules, purchasing products from vending machines, and even social organisation and coordination¹⁾. These devices are quickly becoming an important part of our lives, accompanying us everywhere from home and work, to movies, bars and bus stops.

Location-based services such as finding cafés in foreign cities or locating nearby train stations are appearing^{2),3)}, as are systems for annotating the physical environment with virtual “post-it notes”^{4)~6)}. Typically these require either server infrastructure deployed by providers into the physical environment, or mobile connectivity to the Internet available through expensive contracts.

A plethora of features are now offered in the latest mobile phones including cameras, high capacity storage cards, keyboards, large screens and wireless capabilities like WiFi and Bluetooth, which can be used for ad hoc connections to nearby devices without the cost of a service provider. Furthermore, technologies such as GPS and PlaceLab⁷⁾ are becoming more pervasive.

This paper describes AD LOC, a platform that enables users to tie persistent virtual notes to physical locations without the need for embedded infrastructure in the environment or access to the Internet[☆]. A typical usage scenario is outlined in Section 1.1 and a detailed system model is presented in Sections 2 and 3. In Section 4 an Internet object cache based upon AD LOC’s basic design is developed. This allows

users to save on mobile Internet costs by first probing AD LOC for content. Section 5 contains the evaluation of the system and highlights its advantages and disadvantages. This paper is concluded with related work in Section 6 and future work in Section 7.

1.1 Motivating Scenario

Imagine visiting a cinema and trying to decide which movie to see. You could use your mobile device to download short video trailers and reviews of the films that are currently showing, but it’s expensive to do so. Instead you first query AD LOC to see if anybody has published copies at the cinema’s location. Unfortunately nobody has yet downloaded the trailer you’re interested in, but you do find some reviews and it comes highly recommended. Your interest piqued, you download the trailer of the movie from the Internet and publish it as an AD LOC note for the benefit of future moviegoers. Based on the reviews and the trailer you decide to watch the movie; afterwards you vote for the AD LOC reviews that were the most helpful to you and add your own. As you leave the cinema, the notes you have published remain, cached on the devices of patrons arriving for the next session and other devices in the vicinity.

Figure 1 (a) shows user **A** downloading a movie trailer and publishing it along with her opinion of the movie in the foyer of her local cinema. Another user **B** is participating in the system and caches copies of the notes. When **A** leaves and user **C** arrives in Fig. 1 (b), he is able to retrieve the notes from **B**.

[†] University of Sydney, Australia

[☆] A version of this paper was presented at ICMU 2006⁸⁾.

(a) A publishes notes which are cached on bystander B. (b) A leaves. C arrives and retrieves all notes at this location.

Fig. 1 AD LOC devices cooperatively publishing, caching and retrieving notes at a physical location.

2. AD LOC

AD LOC is an infrastructure-free, localised, persistent and asynchronous platform for collaboratively annotating the physical environment with “notes”.

- Localised: notes are specific to physical locations.
- Persistent: notes remain in the environment.
- Asynchronous: notes are published and retrieved without requiring the concurrent presence of both publishers and consumers.
- Collaborative: anyone can publish or read any notes.
- Infrastructure-free: no servers or Internet connections are required.

AD LOC does not rely on any systems or services specifically installed at locations where users would like to publish notes. Not all devices are expected to have access to the Internet, so AD LOC does not make use of Internet-based servers to store the notes. Besides, storing the content on servers embedded either in the environment or on the Internet may make them susceptible to legal censorship or administrative bias. Instead, all notes stored in the AD LOC system are stored on the constituent mobile devices.

As users with mobile devices pass through locations, they automatically accrete notes that have been published there, and make them available to other users that wish to find and read them. As users leave, their cached notes can be transferred to other devices remaining at the location. AD LOC acts as a platform for storing and retrieving data over an inherently unreliable and dynamic substrate.

Each device is assumed to have the following necessary components: some form of localised, wireless communication technology

such as WiFi; a persistent storage cache of some size; and some way of sensing its location. There are no restrictions on how location is determined, just that it must be possible to convert it into global latitude and longitude coordinates.

2.1 Publishing

A *note* is the unit of annotation in the AD LOC system. It is composed of an identifier, a timestamp, a location described by an “area of relevance” or AOR (see Section 2.3), a descriptive subject, and a MIME data component:

$\langle id, timestamp, AOR, subject, data \rangle$

A note’s ID is constructed by creating a digest of the entire subject and data segments. Notes are conceptually stored at the user’s location at the time of publication where they persist for others to view, even after the publisher has left the area. A note is physically distributed to other AD LOC devices by caching, according to a cache replication policy (see Section 3).

2.2 Querying

A query is made by requesting notes matching a given template. The template can be empty, for a general *location query*, or constraints can be specified filtering the notes to be returned. For example, a timestamp can be given to ensure only recently published notes are returned, or a particular subject field can be required.

The device first examines its own cache, to determine if the query can be resolved locally. If no match is found, or the template is empty, the query is broadcast to all AD LOC devices within range. Each checks its cache and broadcasts a reply if matches are found.

Notes are considered matches if and only if they match the template and are considered relevant to the location of the querying or replying device. By including notes that are relevant to the location of the replying device, notes can be propagated to all devices that may have an interest in them. Section 2.3 details this mechanism.

The list of notes resulting from a query can be displayed to the user according to some ranking function (see Section 7). If only a single note matches a query, it can be automatically displayed to the user.

2.3 Area of Relevance

Notes are relevant to specific locations. However some notes are relevant to very small lo-

Fig. 2 An area of relevance growing from AB to CD.

cations, such as a storefront, while others are relevant to much larger areas, e.g., an entire football ground. Each note is thus associated with an *area of relevance* (AOR) defining this region.

The AOR of a note is not fixed. It begins small when a note is first published and can grow if more distant users also find the note to be relevant. In this way, the AOR of a note is able to approximate the size and position of the conceptual location originally intended by the publisher. The act of querying for a specific note or selecting and reading a note resulting from a general location query implicitly indicates it to be relevant to the user's current location, and hence grows the AOR.

The AOR is used to aid caching policies (see Section 3) and when responding to queries from other devices. A reply to a location query includes all notes that match the given template and that have an AOR including themselves or the query device.

A note's area of relevance is modelled by two coordinates that bisect a circle containing all points where users have indicated the note to be relevant. When a new note is published, its AOR is initially centered at the publisher, with a small, arbitrary radius. When another device outside the area receives the note and marks it relevant, the area is increased to minimally include both the original area and the new point. This results in an area which always encompasses all points that have ever been a part of it. **Figure 2** shows an area described by points A and B being increased to encompass a new point D (and henceforth being described by points C and D).

Since notes are cached on many devices, it is possible for some caches to retain stale AORs. This is remedied by stating that an AOR can never shrink in size. Any device that learns of a conflicting AOR for a note in its cache simply selects the largest as the "true" value.

3. Note Caching

The aim of AD LOC is to tie virtual notes to physical locations by physically caching them on mobile devices that are continually passing through. Since such a storage medium is so volatile, a good caching policy is of paramount importance for note availability.

3.1 Device Density

Clearly, in some situations it is impossible for AD LOC to guarantee note availability. For example, if all mobile devices move away from a location, there is no way for previously published AD LOC content to be available to devices that subsequently arrive. Similarly, if users sometimes refuse to cache notes or respond to queries, availability will suffer.

By assuming a uniform distribution of cooperative mobile users in a square area, it is possible to calculate the number of users required for a particular average neighbourhood degree. This in turn allows an estimate of the number of participants needed for AD LOC to effectively store notes (since all caching is achieved by broadcasting notes to neighbouring devices). It is worth noting that users in real scenarios are unlikely to be uniformly distributed; rather they would be expected to congregate in more popular locations. A uniform distribution does however provide an important minimum estimate for the number of necessary users. This Eq. (1) is adapted from work presented in Ref. 9):

$$n = \frac{6dl^4}{3r^4 - 16lr^3 + 6\pi l^2 r^2} \quad (1)$$

where n is the number of necessary users, r is the broadcast radius, l is the side length of the square area, and d is the average number of neighbours desired for each user. Using the parameters outlined in Table 3 this equation can be graphed to show how the average node degree increases as the number of nodes in the network is increased.

Under these conditions, at least 12 nodes are required for each to have an average of one neighbour, and 24 for an average of two neighbours. Thus, AD LOC is expected to perform poorly with less than 12 users in our experiments, since notes will not often have an opportunity to be replicated before devices move away from locations. Additionally, requests will go unheard as there may be no node to hear them. With at least 24 users, AD LOC should

exhibit good storage properties, leading to few lost notes and therefore good recall of published notes. As the density of the network increases, the ability of the network to retain notes published to it is expected to increase.

3.2 Cache Replication Policy

In general, the goal of the cache replication policy is to maximize the availability of notes and minimize the number of messages between devices. In AD LOC, users are decoupled as much as possible. This reduces the maintenance overhead, while promoting the availability of notes. To find an effective cache replication policy, there are two main decisions to make: when to broadcast notes, and which notes to broadcast. Several possible policies are identified below.

3.2.1 Basic

In the basic caching policy, which underlies all other policies, all messages between devices (including responses to queries) are broadcast such that any device overhearing the transmission of a note can cache it, even if it is not specifically intended to be a part of the exchange. The advantage of this approach is that nothing is published unless it is first requested (meaning notes that never match a query do not incur a broadcast). The disadvantage is that notes that do not match a query before the publisher leaves will not be persisted to that location.

3.2.2 Publish

The publish caching policy immediately publishes any new note via broadcast when it is generated, for caching on neighbouring devices. The advantage is that the note is highly replicated at the location immediately upon its creation. In this policy there is no need to wait for either a request or a periodic timer to replicate the message. However such an approach is highly dependent on user behaviour.

3.2.3 Periodic

In the periodic policy, each device periodically broadcasts some or all of their notes for caching by other devices. The subset to be broadcast is determined by the time at which they were cached; notes that have been recently received are less likely to be broadcast because they are likely to have also been recently overheard by neighbours. For locations with low user turnover, periodic broadcasting may result in unnecessary transmissions, and for high turnover locations the period must be frequent enough to ensure notes are not lost.

3.2.4 Location-aware Periodic

The location-aware periodic caching policy periodically broadcasts the subset of its cached notes which are relevant to the area within its broadcast radius. Any note that has an area of relevance that is at least partially within the broadcast radius of the device is selected to be sent. As with the periodic policy, the set of notes is pruned such that more recently received notes are not broadcast.

3.2.5 All

This caching policy combines both the publish and the location-aware periodic caching policies (as well as the default basic caching policy). The device will broadcast a note upon publication and also periodically broadcast location-relevant notes for caching.

4. The AD LOC Internet Cache

Section 2 described the basic AD LOC platform. As it stands, the system can be used to store any kind of unique user generated content. Some examples of the content that may interest users are; personal opinions, reviews and points of interest, all of which can be stored in the environment and made available to other users. In this section, a simple extension is presented to enable AD LOC to also function as an Internet object cache, saving users the cost of downloading content over expensive mobile Internet connections.

For some locations, there will be relevant material available on the Internet. As the popularity location based services continues to grow the amount of material generated specific to geographic locations will increase. The type of Internet content that could potentially be cached in AD LOC are non-personal location specific Internet objects. Examples of such content include; maps provided by location based map servers, websites for businesses and institutions located at their real world location, bus timetables located at bus stops, and trailers for movies located at cinemas. Any Internet data that is not of a personal nature, that contextually has a specific geographical location, is a candidate for caching in AD LOC. To use AD LOC as an Internet object cache, a user can simply download the object over their mobile Internet connection (if they have one) and republish it as a note for others to find (who either do not have an Internet connection or would prefer not to use it). Of course such a system is user driven, relying on users to tag data they are willing

to share with those around them. Ultimately, cooperation and sharing non-personal data will benefit all users participating in AD LOC.

The only real difference between an Internet object and any other note is that the same object can be published separately by many users. It is preferable to treat them as replicas of one another rather than distinct notes in order to save on storage and broadcast costs. This is trivially achieved by using the URL of the object as the subject of the note, which results in all notes published for a particular Internet object having the same identifier regardless of which devices publish it. Once notes have been published with the same identifiers, they are assumed to be replicas and will not be cached more than once on any particular device.

To use the Internet object cache, a user performs a query with the URL of the object they are trying to find as the subject. If the object has already been published in AD LOC it will be returned to them. Otherwise, they will need to download it themselves and if they choose tag it as non-personal and publish it at their location for others. Future requests for the same object by other users at its relevant location will now be able to be resolved via the local communications interface via AD LOC.

5. Evaluation

To evaluate the performance expected of AD LOC in a real world network, both a theoretical and an experimental evaluation are conducted. The different note-caching strategies are examined. To confirm the simulation model it is compared with the theoretical evaluation. The simulation and theoretical model exhibit similar performance trends.

5.1 Theoretical Evaluation

AD LOC is designed to provide the ability to publish notes that bear relevance to geographic locations. These notes should then remain available for other users who later pass through those locations to query for, and find. The goal of AD LOC is to maintain the notes' availability at their published locations as efficiently as possible. This theoretical evaluation is conducted to uncover the potential overheads associated with the different cache replication policies.

The overhead associated with the operation of AD LOC ($M_{overhead}$) can be divided into three underlying components. The query overhead (M_{query}) which is the number of query

messages sent to find notes. The reply overhead (M_{reply}) which is the number of reply messages sent in response to queries. The cache replication overhead (M_{cache}) which is the number of cache messages sent as a result of the particular caching policy.

As AD LOC queries are purely single hop, there is only one message broadcast per query. Each node sends a query every t_{query} seconds which is the average inter-arrival time of query messages. T is the total time the network is observed. The number of queries per node during the observation period is shown in Eq. (2).

$$M_{query} = \frac{T}{t_{query}} \quad (2)$$

It is much more difficult to estimate the number of messages broadcast in reply to queries. Other users will only respond to a query if they overhear it. Adapting from the Eq. (1), the average node degree and therefore the average number of neighbouring nodes (d) a query broadcast would be expected to reach can be determined. Of these neighbouring nodes, only those with relevant content notes will broadcast a reply. To reduce the number of replies, a neighbour will not send a reply containing content notes that were broadcast by other neighbours responding to the query. As the current state of the system relies heavily on the previous state it is difficult to determine what the average number of responses will be. This aspect can be more thoroughly explored under simulation, although both upper and lower bounds on M_{reply} can be determined.

In the best case, where there is at least one relevant content note available, only one neighbour sends a reply. Such a scenario could occur in two ways. Either only one neighbour has any relevant content notes to send in response to the request or it has all the relevant content notes and all other neighbours overhear its reply. Equation (3) shows the per node best case. When the average number of neighbours is below 1 it is normalised by a factor of d .

$$M_{(best)reply} = \begin{cases} M_{query} d & \text{if } d < 1 \\ M_{query} & \text{otherwise} \end{cases} \quad (3)$$

In the worst case all neighbouring nodes reply to a user's query. Such a scenario could occur in two ways. Either all neighbours have disparate content notes or not all neighbouring nodes are able to overhear each other's replies; leading

Table 1 Note-caching overheads.

Cache Replication	Overhead
Basic	$M_{cache} = 0$
Publish	$M_{cache} = \frac{1}{8} \frac{T}{t_{pub}}$
Periodic	$M_{cache} = \frac{T}{t_{per}}$
LA Periodic	$M_{cache} = \frac{T}{t_{per}}$
All	$M_{cache} = \frac{1}{8} \frac{T}{t_{pub}} + \frac{T}{t_{per}}$

Table 2 Total overheads.

$M(worst)_{overhead} =$	$M_{query} + M(worst)_{reply}$ $+ M_{cache}$
$M(best)_{overhead} =$	$M_{query} + M(best)_{reply}$ $+ M_{cache}$

to the duplication of content notes in replies. Equation (4) shows the per node worst case.

$$M(worst)_{reply} = M_{query} d \quad (4)$$

The overhead associated with replies can be bounded between the worst and best cases. In practice the overhead can be expected to fall between $O(1)$ in the best case and $O(n)$ in the worst case.

Table 1 summarises the overheads associated with different caching policies. The publish cache replication policies publish content notes via broadcast messages every t_{pub} seconds, which is the inter-arrival time of publish messages. Due to the use of the scenarios described in Section 5.2.1, a user will only be in a location to publish a note at best an eighth of the time. This factor of eight is adjusted for this in the equations. Periodic cache replication policies broadcast cache messages every t_{per} seconds, which is the inter-arrival time of periodic caching messages. In this coarse-grained theoretical evaluation there is no distinction between periodic and location-aware periodic. Essentially location aware periodic just selects fewer notes to broadcast cache, sometimes this may result in no broadcast at all, but in the worst case the location-aware periodic policy sends as many messages as regular periodic.

Finally, these equations can be used to estimate the total overhead of AD LOC per node, in the best and worst cases as presented in **Table 2**. In these equations M_{cache} is chosen for the appropriate caching policy. These equations bound the performance overhead that can be expected from AD LOC.

Figure 3 shows evaluation of the equations

Fig. 3 Estimated traffic overhead ratio.**Table 3** Experimental Parameters.

Duration	3,000 (s)
Number of Nodes (Variable)	5–100
Network Size	1,000 × 1,000 (m)
Physical Layer Transmission Range	176.3 (m)
Physical Layer Bandwidth	11 (Mbps)
Query Inter-arrival-time	Poisson 20 (s)
Publish Content Inter-arrival-time	Poisson 20 (s)
Replication Broadcast Period	100 (s)
Minimum Note Rebroadcast Age	20 (s)
Cache Flush	Normal Av: 500 (s) StdDev: 100 (s)
Lead in Time	1,000 (s)
Random Way-point Mobility Model	1 (m/s)
Initial Topology	Uniform random node placement

above using the parameters for experimental evaluation listed in **Table 3**. This figure shows the estimated range of traffic overheads that AD LOC can be expected to incur. M_{reply} dominates these functions as it is the only factor that potentially increases with the number of nodes in the network. In Section 5.2.3 these theoretical results are compared to the traffic overhead results found under simulation.

5.2 Experimental Evaluation

To fully investigate the complexities of AD LOC a simulation model was developed using the OMNeT++ network simulator¹⁰⁾ with the Mobility Framework extension, which allows for the simulation of mobile ad hoc scenarios. This simulation was conducted to understand what level of participation is required for AD LOC to be useful, i.e. the ability of AD LOC to retain and recall notes posted into the system. The overheads associated with the various methods

of note-caching are also examined.

5.2.1 Network Model and Simulation Parameters

The simulated network model consisted of mobile users with the ability to create and query for notes. These nodes are sporadically connected to each other as they move in the network via a wireless 802.11b based interface. In these experiments a simple random way-point model was used to model the mobility of the users.

The application layer generated queries based on a Poisson inter-arrival-time model. These queries were then broadcast to and resolved by neighbouring nodes. The simulation parameters are summarised in Table 3.

All periodic cache replication policies employed a 100s replication broadcast period. To avoid a broadcast storm these periods were randomly offset. Notes were not rebroadcast until at least 20s after they were received.

Each device in the simulation flushed its entire note cache based on a normal distribution centred at 500 seconds with a standard deviation of 100 seconds. On average a device erased its cache 6 times in each simulation run. This was done to simulate a turbulent environment, where users either leave the network entirely or choose to flush their device's cache. If a device were to never flush its cache it would eventually cache all available notes published in AD LOC (dependent upon its mobility and the availability of notes).

When a simulation begins, AD LOC may not be initially stable. To get a view of how AD LOC will perform in practice, measurements during the lead in time were not included in the results. All results presented here are the average of 10 runs for each experiment, and the graphed error bars represent a 90% confidence interval.

Both the effectiveness of AD LOC for publishing and storing content created by users *in situ* and its ability to cache Internet objects for limiting the use of Internet connections were investigated. In these experiments the essential difference between *Internet* objects and *user-created* content notes is that user-created notes are unique, and therefore must be discovered in their location whereas Internet objects can be specifically requested by URL.

AD LOC was experimentally evaluated according to two different scenarios while varying the number of nodes in the network. The sce-

narios that were evaluated were *City* and *Stadium*. The first scenario (City) simulated several city blocks, with 400 small locations of average radius 10m randomly placed to represent points of interest throughout the city. The second scenario (Stadium) comprised 4 large locations of average radius 100m, to represent 4 large points of interest. In each scenario the area covered by the locations was approximately an eighth of the total network area. A user would expect to be in one or more of these locations on average an eighth of the time. These locations were scattered uniformly and randomly throughout the network.

5.2.2 Metrics

The most important metric on which the performance of AD LOC is gauged is the availability of notes that have been published at a location. In these experiments note availability was calculated by keeping track of all the notes that had been published at each location, and comparing this to the notes obtained by users when they performed queries (either already in their cache or returned by other users). This ratio is called the *Recall* metric. It is the number of notes, found on query, relevant to a user's location divided by the number of unique notes that have been published that are relevant to the user's location. It is a value between 0 and 1 derived from Eq. (5).

$$Recall = \frac{Relevant\ Notes\ Found}{Published\ Relevant\ Notes} \quad (5)$$

Since AD LOC relies on proactively replicating cached content to neighbouring users, there is a potential overhead in terms of the number of messages the average user must send and receive. Obviously AD LOC is expected to exhibit increased traffic overhead compared to the ideal case of only querying Internet-based servers or servers embedded in the environment. This overhead is calculated by finding the *Traffic Overhead Ratio* (TOR), which is the ratio between the total number of messages a user sends as part of AD LOC (including queries, replies, and any other broadcast messages) and the number they would need to send if using only a client/server architecture (i.e., only the number of queries issued). This ratio is shown in Eq. (6).

$$TOR = \frac{Total\ Packets\ Sent}{Total\ Queries\ Made} \quad (6)$$

Ideally, AD LOC should exhibit a high recall

and low TOR, although the TOR is expected to increase as recall increases (since AD LOC trades Internet queries for local queries and caching). Recall should increase as the number of users increases, since more content should be cached and be more readily available. The TOR is also expected to rise. As the number of nodes in the network increases, the average number of neighbouring nodes a user will have will also increase. Therefore the average user will be required to cache notes and respond to queries increasingly more often than they perform their own queries.

One of the intended uses of AD LOC is caching Internet data relevant at particular locations. A key metric to evaluate its performance in this respect is the ratio of queries that can be fulfilled by AD LOC to the total number of queries made. This metric is referred to as the AD LOC *Satisfied Internet Queries* (ASIQ) ratio (Eq. (7)). There are two extremes to this ratio: 1 means all queries are satisfied by AD LOC and 0 means that all queries had to be resolved by the devices' Internet connections. This metric shows how well the system is able to cache content and reduce Internet traffic. The ASIQ ratio is expected to increase as more data is cached in AD LOC. Obviously an Internet object will only be available in AD LOC if it has been previously downloaded and cached by another AD LOC user. The number of Internet objects available in AD LOC will increase as more users participate in AD LOC and the longer AD LOC operates.

$$ASIQ = \frac{\text{Queries Resolved By AD LOC}}{\text{Total Queries}} \quad (7)$$

5.2.3 User Content Note Experiments

To make this experiment tractable the scenarios described above were used. When a node's publish timer fires, it initially checks if it is in one of the randomly defined scenario locations. If it is, it publishes a unique content note, otherwise the publish timer is simply rescheduled. In this way content notes are published by users at the predefined locations of interest throughout the network. If the users simply published notes at random, it would not simulate the hypothesis that notes would congregate at points of interest in the network.

This experiment measured the general availability of content notes created by users of the AD LOC system.

Fig. 4 Recall of user-created notes in the City scenario.

Fig. 5 Recall of user-created notes in the Stadium scenario.

Figures 4 and 5 show that regardless of which AD LOC caching policy is chosen, greater than 50% recall can be expected from just 10 nodes, and just 25 are needed for 90% recall. This shows that the underlying *Basic* policy where neighbouring nodes eavesdrop on replies to queries is enough to ensure recall of cache content notes at higher node densities. AD LOC's caching mechanisms are resilient to flux of the users in the network and their reliability increases with the number of participants. The smoother curve in the City scenario is an indirect effect of the smaller location areas.

The traffic overhead graphs in Figs. 6 and 7 show that the TOR increases logarithmically with the number of devices in the network. This increase is due to the increased number of replies a device must make as the density of its one-hop neighbourhood increases. These graphs show that there are two distinct bands of results. While protocols using a publish-on-creation caching policy (All and Publish) are better able to recall published content, they incur a traffic overhead penalty to do so.

Fig. 6 TOR for user-created notes in the City scenario.

Fig. 7 TOR for user-created notes in the Stadium scenario.

Fig. 8 TOR for user-created notes in the Stadium scenario.

Figure 8 shows the comparison between the theoretically estimated overhead range, shown shaded in grey, and the simulation outcomes. This shows that the outcomes of simulation fit with our hypothesis. The line $\logfit(x)$ shows a logarithmic trend line fitted to the *Basic* cache policy curve. The square of the correlation coefficient value for this fit is 0.9705. The initial hypothesis was that the overhead produced by the operation of AD LOC would be between $O(1)$ and $O(n)$ and indeed under simulation it is shown to be $O(\log(n))$.

5.2.4 Internet Cache Experiments

The Internet cache feature of AD LOC (Section 4) allows Internet objects to be stored at

Fig. 9 Internet caching in the City scenario.

Fig. 10 Internet caching in the Stadium scenario.

relevant locations in order to reduce users' mobile Internet costs. This experiment determined the degree to which Internet traffic was replaced by local AD LOC traffic as the number of users increased. We omitted the recall metric for this experiment as it was very similar to the results of the previous experiment.

The Internet cache experiments were modelled using the scenarios described above. Each location in each scenario had a list of data items that could be queried by a user. These data items were assumed to be the sum total of data items on the Internet relevant to the particular location. The user chose an item from this list based on a probability derived from a Zipfian distribution¹¹⁾. The total number of Internet data items was kept constant in each scenario. In the City scenario there were a Poisson average of 20 data items in each of the 400 locations, while in the Stadium scenario there were a Poisson average of 2,000 data items in each of the 4 locations. This allowed a variation in the number of data items in different locations and assumed that bigger locations would have more relevant data items.

Figures 9 and 10 show that as the number of nodes in the network increases, the fraction of queries that can be satisfied wholly within the AD LOC system also increases. This is a direct consequence of the fact that as the number of users grows, the number of requests for Internet objects grows, and more objects are cached.

Fig. 11 TOR for Internet objects in the City scenario.

Fig. 12 TOR for Internet objects in the Stadium scenario.

The City scenario had a low variance because each location had only approximately 20 Internet objects associated with it, such that it would not take long for all of these to be requested and cached for future users. This would lead to almost all queries in a location being resolved by AD LOC after an initial phase of requests. In the Stadium scenario there were around 2,000 objects for each location. Many of these would be requested only once and there would be many cache misses. As a result of the Zipfian distribution of requests, a few objects would be requested very often, resulting in many cache hits. The combination of these effects produced a very high variance, though on average the Internet cache was able to resolve around half of the Internet object queries in the Stadium scenario.

The ability of AD LOC to recall Internet data is similar to its ability to recall user-created notes. However, it incurs less of a traffic overhead penalty. This is shown in **Figs. 11** and **12**. The reason for the lower overhead is the non-uniqueness of Internet data and the use of a Zipfian distribution to model the probabilities of querying certain pieces of Internet data. This means that a device is more likely to get a cache hit for the data it is seeking than in the user scenario.

As AD LOC is essentially a loosely associ-

ated opportunistic network architecture, even as more nodes are added to the network it never reaches full recall or ASIQ. Still, AD LOC provides essentially free access to serendipitously available content with reasonably low network overhead.

6. Related Work

There is increasing interest in location-based services¹²⁾ and a critical mass of users with powerful and capable mobile devices¹⁾. There have been many systems that allow users of mobile devices to annotate the physical environment using the “post-it note” metaphor. One of the initial systems was Stick-e notes⁴⁾, but similar ideas have existed since the dawn of ubiquitous computing with devices such as the ParcTab¹³⁾ and Chameleon¹⁴⁾.

Stick-e notes present the notion of using context to trigger events. Note authors attach a contextual “trigger” (often including a location) to a note that describes under which circumstances it shall be displayed to the user. More recently, this concept has been extended to aid collaboration on city and worldwide scales^{5),6),15)} with public annotation of large tracts of physical space.

Other systems have taken a more introverted approach, using location-based services only to aid individual users. Web-enhanced GPS³⁾ uses GPS to find a user’s location, maps this via a database to a street address, then uses this as a search term to find relevant web pages.

Two recent studies^{16),17)} found that location-based services accessed by mobile devices would be well-received by users in general. They identified the importance of location in informal social activity and the importance of available topical, dynamic content. They also found that people liked and were willing to contribute additional content of their own. AD LOC’s ability to cache location-relevant web information and publish personal content fits this profile well. The E-graffiti¹⁸⁾ and CampusAware²⁾ projects found that many people enjoyed the social aspect of a collaborative location annotation system and contributed an unexpectedly high number of notes of advice or opinion specific to their locations.

The main difference between AD LOC and these similar systems is the storage medium. Most collaborative note systems thus far have focused on usability or the context- and location-aware aspects, and simply relegated

the storage to servers. In contrast, AD LOC focuses specifically on an alternative storage approach that requires no such infrastructure to be deployed.

Abiding Geocast¹⁹⁾ is a system similar to AD LOC. It offers a time stable geocast, which allows messages to be delivered to a designated region with a given lifetime. The system uses multihop ad-hoc networking to ensure that any node entering the region will receive the message, up until the message lifetime expires, or it is explicitly revoked. To achieve this one or more nodes are chosen as distribution servers in, or close to, the destination region.

As AD LOC is purely a single hop system for collaboratively tying notes to locations it improves on Abiding Geocast in two ways. Firstly as Abiding Geocast is built as a notification system it must keep track of any node entering or leaving a destination region and ensure that it is sent the geocast message, either through unicast or broadcast. This quickly becomes unscalable as the number of nodes grows. However the authors offer some optimisations, such as grouping nearby nodes together, to minimise this problem. AD LOC deals with this problem by shifting the onus for information retrieval to the user. Abiding Geocast is essentially a *push* protocol ensuring that data is pushed out to nodes when they arrive in a geocast region. By contrast AD LOC is a *pull* protocol, allowing users to query for information when they require it. This difference in approach is borne out by the difference in intended use. AD LOC has been designed primarily as a message cache that users can look up when interested, whereas Abiding Geocast has been designed as a mechanism to display important time critical traffic messages. AD LOC can emulate notification by having users periodically request information about their location. However in the common case it is expected that users will only query for information when they require it, and thus to push data to users would require more network traffic than is otherwise necessary.

The other improvement of AD LOC is that it does not require a multihop component. When creating a note a user must be at the location where the note is to be published. This allows AD LOC to better scale to large numbers of nodes and messages, as the effects of adding a new note are only felt locally, and not throughout the network as with a multihop geocast. This phenomenon is discussed in Ref. 20) and

shown in Ref. 19).

7. Conclusion and Future Work

The experimental results show that a critical mass of users is necessary for AD LOC to function in a useful manner. The results show this number is quite small. However, the simulations were conducted using an unobstructed free space signal propagation. In a real world scenario the penetration of WiFi may be much lower. This is both a blessing and a curse. With a lower broadcast radius, there is less interference and more users can be supported. The consequence is that more users will be needed in order to reach critical mass. The mobility and note publication/query models employed were also simplified. Future work could investigate more realistic models based on real mobility traces.

A possible criticism of AD LOC is that without environmental infrastructure or Internet-based servers to store notes, the reliance on transient devices will result in much published content being permanently lost when users leave a location. However, in many cases users are likely to repeatedly return to the same locations, e.g. workplaces or lunch spots. If their caches are not erased immediately upon leaving a location, much of the content will return with them the next day as they follow their ordinary routine. Indeed, location annotation systems have found users to often access the system from the same small subset of locations¹⁸⁾.

Another possible cache replication policy not discussed earlier is cache-on-departure. When a user is aware they are leaving the AOR of one or more of the notes in their cache, they can broadcast them for caching on other devices. The advantage of this approach is that notes that are not cached elsewhere at a location can be replicated before they are permanently lost, without the need for additional queries. However, mobile users may potentially move through many areas of relevance in a short period, triggering excessive broadcasts and there are subtle problems with determining when a user is departing an area²⁾.

The cache replication policy tries to widely cache as many notes as possible, but real caches are of limited capacity. A cache replacement policy decides which notes to eject when the cache becomes full. For this paper, caches are assumed to be large enough never to require replacement, but in practice the policy could use

the age of the note, the distance from its area of relevance, and the likelihood of the user returning to the location in future. Clearly the willingness of a node to participate is also important.

AD LOC cannot absolutely guarantee a note will be persisted for others to find, due to the inherently unreliable storage medium. Locations that need stronger guarantees could deploy a proxy server that acted like any other AD LOC device. The system's main strength is that such proxies are not required, though they can augment it if desired.

It may be beneficial to use a ranking function for presenting matching notes to users after a general location query. One possible way of doing this would be to have a voting system allowing AD LOC users to vote for notes they find, affecting their rank in the results of others' location queries. This would add an extra participatory component that could be combined with other aspects, such as the age of notes, to make the system more manageable for users in locations with a high number of users and notes.

The area of relevance technique will not work well for locations that are not approximately circular, as the area will need to encompass more and more irrelevant space to contain its extents. In the case of a river for example, the area of relevance would likely encompass vastly more area than it should. This is a known drawback of our approach, and will be addressed in future research.

There are some interesting properties of the AD LOC storage mechanism that would not occur with server-based approaches. For example, notes cannot be added to locations without physically being there which could present a problem if trying to seed the system with information. Also, notes will vanish from an area once everybody has left but may return the next day, for example, when people return with their caches intact. Curiously, it would even be possible for the same locations to harbour radically different sets of notes over the course of the day depending upon their changing social functions.

Acknowledgments The authors would like to acknowledge the ongoing support of the Smart Internet Technology CRC and also thank the reviewers of ICMU 2006.

References

- 1) Rheingold, H.: Smart mobs, *Shift: At The Frontiers of Consciousness*, pp.30–33 (Mar. 2004).
- 2) Burrell, J., Gay, G.K. and Kubo, K.: Context-aware computing: A test case, *Proc. UbiComp*, Göteborg, Sweden (2002).
- 3) Hariharan, R., Krumm, J. and Horvitz, E.: Web-enhanced GPS, Strang, T. and Linnhoff-Popien, C. (Eds.), *International Workshop on Location- and Context-Awareness, LoCA*, p.95 (2005).
- 4) Pascoe, J.: The Stick-e Note architecture: Extending the interface beyond the user, *Proc. conference on Intelligent user interfaces*, pp.261–264 (1997).
- 5) Dearle, A., Kirby, G.N.C., Morrison, R., McCarthy, A., Mullen, K., Yang, Y., Connor, R.C.H., Welen, P. and Wilson, A.: Architectural support for global smart spaces, *Lecture Notes in Computer Science 2574*, pp.153–164, Springer (2003).
- 6) Carter, S., Churchill, E., Denoue, L., Helfman, J. and Nelson, L.: Digital graffiti: Public annotation of multimedia content, *CHI extended abstracts on Human factors in computing systems*, pp.1207–1210 (2004).
- 7) Schilit, B.N., et al.: Challenge: Ubiquitous location-aware computing and the “place lab” initiative, *Proc. ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, pp.29–35 (2003).
- 8) Corbet, D.J. and Cutting, D.: Ad Loc: Location-based Infrastructure-free Annotation, *ICMU 2006*, London, England (Oct. 2006).
- 9) Corbett, D.J., Ruzzelli, A.G., Everitt, D. and O'Hare, G.: A procedure for benchmarking MAC protocols used in wireless sensor networks, Technical Report 593, School of I.T., The University of Sydney (Aug. 2006).
- 10) Varga, A.: The OMNeT++ discrete event simulation system, *European Simulation Multiconference*, Prague, Czech Republic (June 2002). <http://www.omnetpp.org>
- 11) Adamic, L.A. and Huberman, B.A.: Zipf's law and the internet, *Glottometrics*, Vol.3, pp.143–150 (2002).
- 12) Hazas, M., Scott, J. and Krumm, J.: Location-aware computing comes of age, *Computer*, Vol.37, No.2, pp.95–97 (Feb. 2004).
- 13) Want, R., et al.: The PARCTAB ubiquitous computing experiment, Technical Report CSL-95-1, Xerox Palo Alto Research Center (Mar. 1995).
- 14) Fitzmaurice, G.W.: Situated information spaces and spatially aware palmtop computers,

- Comm. ACM*, Vol.36, No.7, pp.39–49 (1993).
- 15) Mankins, M.: Location linked information: A framework for emergent, location-based content deployment, Master's thesis, Massachusetts Institute of Technology (2003).
 - 16) Kaasinen, E.: User needs for location-aware mobile services, *Personal and Ubiquitous Computing*, Vol.7, No.1, pp.70–79 (2003).
 - 17) Jones, Q., Grandhi, S.A., Whittaker, S., Chivakula, K. and Terveen, L.: Putting systems into place: A qualitative study of design requirements for location-aware community systems, *Proc. ACM conference on Computer supported cooperative work*, pp.202–211 (2004).
 - 18) Burrell, J. and Gay, G.: E-graffiti: Evaluating real-world use of a context-aware system, *Interacting with Computers*, Vol.14, No.4, pp.301–312 (2002).
 - 19) Maihöfer, C., Leinmüller, T. and Schoch, E.: Abiding geocast: Time-stable geocast for ad hoc networks, *VANET '05: Proc. 2nd ACM international workshop on Vehicular ad hoc networks*, pp.20–29, New York, NY, USA (2005).
 - 20) Li, J., et al.: Capacity of ad hoc wireless networks, *ACM/IEEE MOBICOM*, pp.61–69 (2001).

(Received January 22, 2007)

(Accepted April 6, 2007)

(Released June 6, 2007)

Derek J. Corbett received a Bachelor of Advanced Science with first class honours in Computer Science from the University of Sydney in 2003. Currently he is a Ph.D. candidate at the University of Sydney, under the supervision of Prof. David Everitt. In 2005 he was granted a DAAD stipendium for a six month exchange to RWTH University Aachen, Germany. He is currently working on the use of location awareness in ad hoc networks, with a focus on the interplay of location and both MAC protocols and service discovery protocols.

Daniel Cutting is a Ph.D. Candidate in Computer Science at the University of Sydney, Australia supervised by Drs. Quigley and Landfeldt. He received his bachelor degree from the University of New South Wales in 1998 and currently holds an Australian Postgraduate Award and Smart Internet Technology CRC scholarship. His main research interests are peer-to-peer networking, mobile ad hoc networking, operating systems and middleware.