Wireless Ad Hoc Podcasting

Vincent Lenders[a] vlenders@princeton.edu
Martin May[b] may@tik.ee.ethz.ch
Gunnar Karlsson[c] gk@ee.kth.se
Clemens Wacha[b] cwacha@ee.ethz.ch

[a]Princeton University, NJ, USA
[b]ETH Zurich, Zurich, Switzerland
[c]KTH, Stockholm, Sweden

Podcasting has become a very popular and successful Internet service in a short time. This success illustrates the interest for participatory broadcasting, in its actual form however, podcasting is only available with fixed infrastructure support to retrieve publicized episodes. We aim at releasing this limitation and present herein our podcasting system architecture together with a prototype implementation based on opportunistic wireless networking that allows us to extend podcasting to ad hoc domains.

I. Introduction

Podcasting has become very popular for dissemination of audio and video content over the Internet. It is based on user subscriptions where software clients query servers for updates of subscribed content feeds using an Internet syndication protocol like RSS [3] or Atom [6]. Podcasting is often used for downloading of contents to a mobile media player that people play back on the move. A main limitation with the current system is the inflexible separation of downloading to a docked media player and expending of the data when on the move. This clearly limits the usefulness of the service since there could be hours passing between downloading opportunities.

This paper describes our efforts to broaden the concept of traditional podcasting for wireless ad hoc delivery of contents among mobile nodes. The idea is supported by the high penetration of short-range wireless communication capabilities (mostly WLAN and Bluetooth) in modern media players like iPods, Zunes, or smart phones. Without relying on any infrastructure support, our wireless ad hoc podcasting service enables the distribution of content using opportunistic contacts whenever podcasting devices are in wireless communication range. We envision such a dissemination mode to be particularly successful in urban areas, where people meet in public transportations, at sport, art, or music events.

II. System architecture

The wireless ad hoc podcasting we propose herein expends the traditional podcasting concept for device-to-device delivery of contents amongst mobile nodes. New contents are provided in an ad hoc wireless podcasting area either by access points connected to the Internet or by the mobile nodes themselves. The mobile nodes could have been filled with contents off line by means of docking or, more interestingly, have generated the contents like audio, photo, or video recordings on their way. In the first case, each access point fetches contents from podcast servers across the Internet and forwards them to mobile nodes within its range; this is done in the traditional podcasting mode. In the second case, a mobile node that has contents to share provides data to other mobile nodes that pass in radio range. This is the new content distribution mode that we add to the existing one.

Our ad hoc podcasting mode brings the following advantages. First, it provides nodes with contents when they are not connected to the Internet; second, it provides a new ad hoc broadcasting domain when also the sources of the data could be the mobile nodes.

III. Content structuring

Key for the interoperability of different podcasting devices is the way content is organized. For our implementation, we propose a solution by extending the standards of existing podcasting and the corresponding syndication protocols. Specifically, ad hoc podcasting organizes content in the ad hoc domain in *channels*. This facilitates the search and allows users to subscribe and automatically receive updates for contents that they are interested in. Following the approach of RSS [3] and Atom [6], we further structure channels into *episodes* and *enclosures*. To make efficient use of contacts with a small duration, we divide enclosures into *chunks*, transport-level small data units of a size that can typically be downloaded in an individual node encounter. Note that the chunks do not have to be received in order, i.e., chunks can be downloaded opportunistically from different peers. Each chunk is further divided into *pieces*, the atomic transport unit of the network. As an example, channels could be sports, new channels, or music channels. An episode would then be a published event, e.g. a particular soccer game, within such a channel. One or more video, audio or text description files belonging to an episode are the enclosures. Chunks would be for example data units of size 250KB and pieces would be set to the MTU size of the link layer to avoid fragmentation at the MAC layer.

IV. Ad hoc podcasting protocols

We propose an opportunistic pair-wise synchronization scheme for contents exchange between the mobile nodes. That is, when two nodes happen to move into wireless range, they associate together and start soliciting for episodes from subscribed channels. This simple approach has several advantages over trying to establish a multi-hop network over the nodes. First, there is no overhead associated with discovering the network topology and maintaining routes up-to-date. This is particularly critical when the node contacts are of short durations and the network is often disrupted as we expect to be in typical deployments. Second, as data is not pushed, the nodes have complete control over the data they carry and forward. Third, not all the MAC technologies support multi-point connections and by relying on pair-wise associations, we leverage almost any wireless MAC. Last but not least, this approach is simple and hence allows for efficient implementations that can run on devices with limited resources.

In the absence of multi-hop routing, we do not need a network layer in our system architecture and we therefore implement our protocols directly on top of the MAC layer. We identify three basic protocols for (i) device discovery, (ii) identification of the contents available at the remote peer, and (iii) contents downloads. Device discovery is achieved through broadcast beacons sent periodically which inform neighboring devices about a node's presence. In addition, these beacons include a time stamp that is set to the last time a node has downloaded new contents. This field helps reducing the synchronization attempts when two nodes remain in vicinity for longer periods. The second protocol serves to identify channels and episodes at the remote peer that one is subscribed to but does not have yet. Instead of querying every single subscribed channel, the devices first exchange a bloom filter hash index [2] that contains all channel IDs a device offers. Both devices then test their locally subscribed channels against the bloom filter hash index of the other device and identify matching channels. Employing bloom filters reduces the amount of information that has to be exchanged for this process. If at least one device has found a matching channel, it starts querying for episodes. We currently support three different types of queries that are employed depending on the channel policy:

1. The peer requests any random episodes within a channel that the remote peer offers. This is particularly useful if the episodes are independent of one another (like a selection of poems).
2. A peer requests any episodes which are newer than a given date starting with the newest episode. This is particularly useful for news feeds in which the latest news feed has the most value.
3. The peer requests any episodes that are newer than a given date starting with the oldest episode. This type of retrieval is useful for episodes which depend on each other like a series.

When a missing or incomplete episode is identified, the third protocol is initiated. This protocol downloads the identified episode(s) chunk by chunk. The download protocol is resilient against download interruptions by having unique ids for each chunk. In case an association is interrupted, the missing chunks can be download in the future without having to download the successfully downloaded chunks again. The chunks are downloaded in a random order similarly to the download process in BitTorrent [1]. This allows for a better mix of chunks among the nodes and generally decreases the probability of having the same incomplete set of chunks at different nodes.

Both nodes in an association share the download link capacity in a fair manner without enforcing any strict guarantees. Furthermore, each node performs the described protocols in an asynchronous way. It is therefore possible that one node is still downloading while the other is finished. When a device has finished downloading (because the remote peer has no further contents of interest), a device could simply wait, or better, start soliciting contents from channels other than the ones the user is subscribed to. This behavior increases the probability of having contents to share to other nodes in future encounters and generally increases the overall performance. We have investigated different types of solicitation strategies based on the channel popularity in [4].

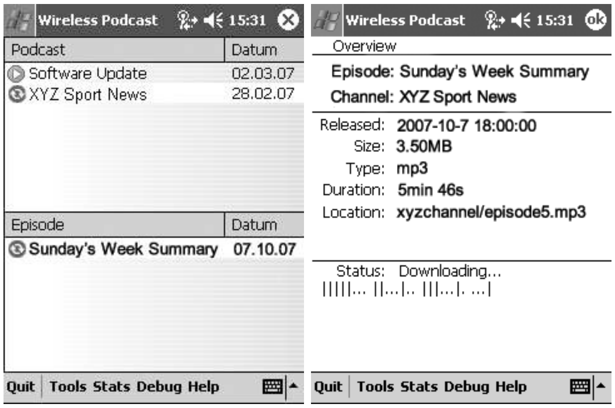

Figure 1: Application running on a handheld device.

V. Implementation

We have implemented the described protocols and a corresponding podcasting application in C++. Our implementation runs on Laptops as well as on handheld devices like HP Ipaqs or even cell phones. We have successfully tested our implementation on various operating systems including Windows Mobile Edition, Windows XP, MAC OS, and Linux. We have used the IEEE 802.11b MAC turned in ad hoc mode but almost any other MAC technology (e.g., UWB, Bluetooth, infrared, Zigbee, etc.) could potentially be used given that our design only relies on a point-to-point communication abstraction at the MAC layer.

Figure 1 (left) represents a snapshot of the podcasting application running on a handheld device. The upper part of the screen lists the channels that a user is subscribed to. The lower part of the screen shows individual episodes that the node has already or is currently retrieving for the selected channel in the upper part. Figure 1 (right) represents the download view. The display shows general information about the episode that is currently being downloaded. The lowest part of the screen with bars and dots represents the chunks that have already been received. A bar represents a downloaded chunk whereas a dot indicates a missing chunk.

We have performed various experiments with our implementation to understand the challenges and bottlenecks of the system and how content propagates with user mobility. The overall distribution of podcasts is mainly dictated by the mobility and density of the nodes, two factors that we cannot influence in the real world. However, one observation is that the download performance significantly depends on the link quality between two devices. In order to discriminate nodes based on the link qualities, we have found it useful to count the number of received discovery packets from neighboring devices and associate with those that have the highest count. We have further measured that for typical episode sizes of 3-5MB, it is possible to download a complete episode in a single contact when one node is fixed and the other node is crossing its range at pedestrian speed. This even indoors when the wireless range is limited by the structure of the buildings. More detailed measurement results with our implementation can be found in [5].

VI. Conclusions

We have proposed to extend the traditional server-based podcasting delivery mode with an ad hoc delivery mode among mobile users. Our approach for ad hoc synchronization between mobile nodes relies on a pair-wise and receiver-driven solicitation protocol. We do not perform explicitly multi-hop routing at the network layer. Rather, routing is achieved through a combination of user mobility and application-layer solicitation. In fact, our system architecture does not include a network layer and we implement the transport layer directly on top of the MAC layer. With the proposed approach, we hope to generally broaden the concept of podcasting and to see a more open and less restricted broadcasting mode in the wireless domain than what we have today.

References

[1] BitTorrent.org. Protocol Specification. http://www.bittorrent.org/protocol.html, 2006.

[2] B. Bloom. Space/time tradeoffs in in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970.

[3] RSS Advisory Board. RSS 2.0 Specification. http://www.rssboard.org/rss-specification, June 2007.

[4] Vincent Lenders, Martin May, and Gunnar Karlsson. Wireless Ad Hoc Podcasting. In *Proceedings of IEEE SECON*, San Diego, CA, June 2007.

[5] Martin May, Clemens Wacha, Vincent Lenders, and Gunnar Karlsson. Wireless Opportunistic Podcasting: Implementation and Design Tradeoffs. In *Proceedings of the ACM Mobicom CHANTS Workshop*, Montreal, Canada, September 2007.

[6] M. Nottingham and R. Sayre (Eds.). The Atom Syndication Protocol. IETF RFC 4287, December 2005.