Live Objects: A System for Infrastructure-less Location-based Services

Arata Miyamoto[1,2], Valerio Panzica La Manna[1], V. Michael Bove, Jr.[1]
[1]MIT Media Lab, Cambridge, MA, USA; [2]Toshiba Corporation, Tokyo, Japan
arata.miyamoto@toshiba.co.jp, vpanzica@media.mit.edu, vmb@media.mit.edu

***Abstract*—Location-based services are rapidly changing the way users explore and interact with the surrounding environment. They typically associate location information, provided by GPS or Bluetooth Low Energy devices, with remote content accessible through the Internet. However, Internet connectivity, especially in outdoor spaces, is not always available or may be too costly to be used for downloading large content (e.g. when users are abroad or have a limited data plan). In this paper we propose Live Objects, a system for infrastructure-less proximity-based services. Live Objects are pervasive media-server devices, which can be discovered by users and stream locally stored media content in proximity without the need for an Internet connection. Live Objects can be installed anywhere, even in places without power supply and communication infrastructure. We demonstrate a deployment of Live Objects in an outdoor test field. Using these devices, together with an Android app and a software middleware, users can have a self-guided tour of the area, discover the interesting places, and get associated information, particularly in the form of high-resolution video.**

I. Introduction

The high availability of modern smart devices is fostering the development of new location-based services that are changing the way users explore and interact with the surrounding environment. Examples of these services include augmented reality applications, which allow users to have different information overlays about the space around them; smart retail stores that can send coupons and media advertisements to interested users in proximity; and tourist-focused applications, where visitors can discover art pieces in a museum or interesting places in natural parks or historic sites.

These services associate a particular location with various forms of content. Location information is typically obtained by the GPS of user devices, or derived through lookup from tags returned by proximity-based devices, such as Bluetooth Low Energy (BLE) beacons deployed in the environment. The content is typically remotely stored in the Cloud and accessible through the Internet. However, there are still many scenarios where an Internet connection is not available, or it is not fast enough to download large remote content. This is especially true for outdoor spaces not covered by high-speed cellular network (LTE), but also for users with a limited or with no data plan, such as tourists in foreign countries. This is also true for some indoor public spaces, such as museums and shops, where a Wi-Fi network is available but the Internet access is restricted to the public.

In this paper, we propose *Live Objects*, a system for infrastructure-less location-based services. A Live Object consists of a new small device that can stream media content wirelessly to nearby mobile devices without an Internet connection. Live Objects are associated with real objects in the environment or with particular locations, such as an art piece in a museum, a statue in a public space, a historic building, or a product in a store. Accompanying middleware supports the development of Android apps that can discover, connect and interact with Live Objects in proximity. All the content related to a particular place is locally stored in the Live Object and transmitted to user devices with a high data rate and without the mediation of Cloud-based services.

We demonstrate the use of Live Objects with a first deployment in a real outdoor test field. We built four prototypes and installed them as part of documenting an environmental restoration project of a former cranberry farm. The prototypes have robust hardware and are equipped with solar panels and rechargeable batteries to enable continuous operation even in cloudy or dark conditions. Each Live Object is associated to a particular spot in the farm and contains the related content, mainly in the form of high-resolution video. On top of the software middleware, we developed a prototypical Android application where visitors of the farm can discover the Live Objects and receive streaming data. The application is used as a guide service for people exploring the farm, as well as to provide updates about the restoration project.

The paper is structured as follows. Section II provides an overview of the different components of the system. Section III describes the implementation of the demonstration. Section IV discusses some of the related work. Section V concludes the paper.

II. System Overview

The Live Objects System consists of two main components. The first component is the *Live Object*, a pervasive device that sends location-based content to user devices. The second component, called *Explorer*, is a middleware to be installed in user devices to interact with the Live Objects. Applications built on top of Explorer can discover nearby Live Objects and access their locally stored content.

A. Live Object

A Live Object works both as a beacon device and as a media server. As a beacon, it notifies its presence to Explorer in the communication range of radio signals. As a media server, it sends to user devices the content related to the location where it is installed. The content is typically in the form of large media data, such as audio/video files or a collection of pictures, which may not be suitable for real-time transmission over slow cellular or Wi-Fi connections. The data transmission is performed in peer-to-peer fashion, without any additional node or network infrastructure.

This work has been partially funded by Toshiba Corp. Thanks to Glorianna Davenport for helping us deploy and maintain our prototypes in Tidmarsh.

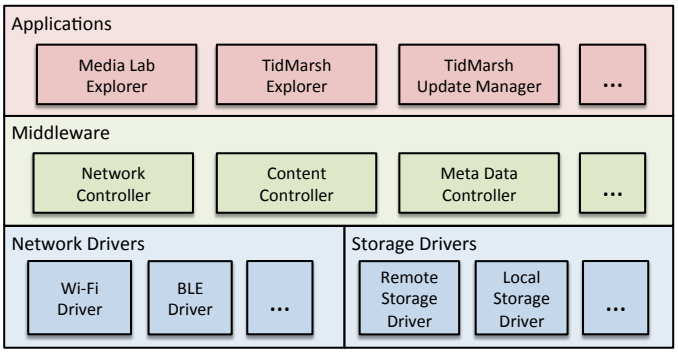

Fig. 1: Explorer Architecture Overview

To deliver media data to Explorer, a Live Object runs an HTTP web server. A Web API allows Explorer to download and upload a new file to/from a Live Object, or update an existing one. Live Objects support user device authentication through password.

In many scenarios, the content locally stored in the Live Object needs to be updated. The system offers two ways for content update according to the available infrastructure. If the Live Object can access the Internet, the Live Object can periodically enter *update mode* and synchronize to remotely stored content, which the administrator is able to update through a web application. This function has been implemented for scenarios in which Internet is only available to administrators and not to the general public. If no infrastructure is available, the administrator can use an *Update Manager* app, based on the Explorer middleware, and physically go in the proximity of Live Objects and update the content. Future work will focus on interconnecting Live Objects deployed in the same site using a mesh network protocol over Wi-Fi to facilitate remote content updates. With this mechanism, content can be propagated from one node to another until reaching the Live Object to be updated.

B. The Explorer middleware

The Explorer middleware eases the development of proximity-based services using Live Objects. The current implementation supports the Android operating system. The support for further mobile platforms is left as future work.

Figure 1 provides an overview of its modular, extensible, three-layered architecture. The upper layer is the application layer, containing the user apps. Each app implements the UI and the application-specific functionality for a specific deployment of the Live Objects or for a specific scenario. Section III presents an example of an app for the Tidmarsh Project. The middle layer is the middleware itself and provides the abstract functions to interact with a generic Live Object. It consists of three main components: the *Network Controller*, the *Content Controller*, and the *Metadata Controller*. The Network Controller provides two main functions: the discovery of Live Objects, and the connection. The discovery function allows Explorer devices to discover all the Live Objects in proximity. An Explorer device discovers a Live Object by listening to radio signals, such as Wi-Fi, Bluetooth, or a combination of both. The discovery function abstracts from the specific protocol used and its implementation is left to the lower driver layer. The discovery function also provides a way to retrieve additional information about the discovered Live Objects even before connecting to them (e.g. its name, or its location). The discovery function can be explicitly initiated by the application layer, when, for example, the user launches the app. It can also use a publish-subscribe mechanism, running in the background, that notifies the application when a Live Object is in proximity. After a Live Object has been discovered, the user can connect to it. This is performed by the connection function of the Network Controller, which, also in this case, hides the complexity of the lower-level protocol. This function takes also care of the authentication process, if the connecting Live Object requires it.

The Content Controller provides a way to access the content associated with the Live Object. This includes read, write, update and delete queries. The content can also be directly streamed from the Live Object once a connection has been established. The Content Controller optionally caches content previously downloaded from a Live Object in the local storage of user device. If a media file has already been cached, the middleware reads the content in its local storage without querying the connected Live Object. This function has been introduced to improve the performance of content streaming and reduce the load of concurrent requests in the Live Object.

The middleware can also assign metadata to a media file or a set of media files contained in a Live Object (e.g. a text description of the content, creator's name, data formats). The Metadata Controller provides the management of these metadata. Metadata are defined by JSON configuration files that are automatically downloaded once a Live Object is connected. These JSON files can contain arbitrary data that are application specific. Metadata are useful, for instance, to distinguish different types of content and to support multiple content formats on a single application.

The lower layer contains the *driver* modules implementing the interaction with the Android OS services. This layer contains the *Network Drivers* and the *Storage Drivers*. The Network Drivers implement the communication with the lower-level network protocols available in the user device in which Explorer is installed. The current implementation supports both Wi-Fi and BLE communication. Further drivers can be easily added to support other protocols, such as Wi-Fi Direct. The Storage Drivers provide the support for accessing the Live Object media content. The Remote Storage Driver implements an HTTP client invoking the Live Object API to query and retrieve the remotely stored content. The Local Storage Driver provides a way to retrieve content that is eventually cached on the file system of user devices.

III. Prototype Implementation

In this section we describe a first outdoor deployment of the Live Objects System in Tidmarsh Farms [1], a 577-acre private farm situated in Plymouth, MA, USA. The farm is hosting a restoration project that will transform 250 acres of the property from a cranberry farm into a natural wetland system. In this context, the Live Objects System is used as a non-invasive tool to guide visitors in exploring the farm and in having a better understanding of the different phases of the project, and its positive ecological impact [2].

[1] http://tidmarshfarms.com

[2] A demo video of this project is available at: http://shair.media.mit.edu/applications/liveobjects-at-tidmarsh

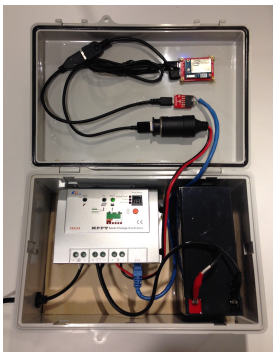

Fig. 2: Waterproof Live Object prototype.

Fig. 3: Installation on a pole.

A. Implementation of Live Objects

The outdoor area in which the Live Objects were installed not only does not provide high-speed Internet connectivity to users, but also it does not allow the devices to access a power supply. To minimize the solar panel and battery sizes and costs, energy efficiency of the devices is an important consideration.

The prototype consists of an Intel Edison board, equipped with Bluetooth and Wi-Fi modules, which is connected to a rechargeable battery and a solar panel via a charge controller. The charge controller converts voltages and regulates current between the solar panel, the battery, and the board. Figures 2 and 3 shows the internal of the prototype, and its final installation in poles. Finally, since the devices are likely to be exposed to rain, the board, the battery, and the charge controller have been enclosed in a waterproof box.

A Live Object is initially in *stand-by* mode, with the Wi-Fi module turned off to save energy if no user devices are in proximity. In this mode, only the BLE is on, advertising the Live Object presence through periodic beaconing. The Explorer on a user device discovers the Live Object from the BLE signal (Live Objects can be identified by a predefined prefix of BLE device name). Once a Live Object has been discovered, the user can decide to connect by sending a request to the Bluetooth module. If a Live Object receives a connection request to its BLE interface, it activates the Wi-Fi module listening for requests from Explorer. In this *active* mode the Explorer on the user device connects to the Live Object as a Wi-Fi client and queries stored media files. A selected item can then be streamed into the user device and shown in the user app. After a given timeout, if no connections have been established, the Live Object turns back to stand-by mode.

B. Implementation of the Visitor App

We implemented a prototype Android app over the Explorer middleware. With this app, a mobile user can get information about a location when the user comes close enough to one of the Live Objects. Figure 4 shows the main screen of the app that locates the discovered Live Objects on a map of the farm. The position data of a Live Object is encoded in the SSID of its Wi-Fi interface and in its BLE device name. Since these fields have a limitation on the maximum length (32 characters), the position data is shortened by quantization. Due to the self-advertisement of location data, our prototype does not require GPS services and thus it can be applied even in indoor scenarios where user devices cannot receive GPS signals. The app also leverages the background discovery offered by the middleware and propagates the notification of newly discovered Live Objects to the user device and eventually to other associated wearable devices, such as smart watches. Finally, through the app, the user can select a discovered Live Object and start watching the streamed content.

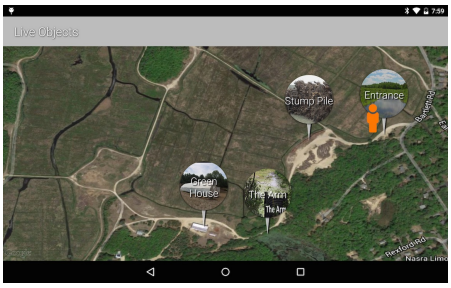

Fig. 4: A screenshot of the main screen of the Visitor App.

IV. Related Work

A variety of location-based services have been proposed that rely on Internet access to send data or messages from external web servers to mobile users [1] [3] [4]. In contrast to these services, Live Objects do not necessarily require Internet connections because they themselves work as stand-alone Web servers and establish single-hop networks with mobile users for direct content delivery. The ShAir middleware proposed by Dubois et. al. [2], offers a P2P content sharing functionality that is similar to the one offered by the Explorer. However, our focus is on infrastructure-less content sharing between the users and the surrounding environment, rather than between user devices. The use of Wi-Fi SSID for device discovery has been originally proposed by Muralidharan et al. [6] and adopted in our previous work [5]. This work extends this idea to support an energy efficient discovery, which combines both Wi-Fi and Bluetooth.

V. Conclusion

In this paper we have proposed a system, consisting of a new pervasive device, middleware, and an accompanying app, to support the development of proximity-based services in scenarios where Internet connection is not available. Moreover, our prototype shows how these services can be provided in outdoor scenarios without power supply, even subject to rainy conditions. Future extensions will look into making the system more social, exploring game applications such as media scavenger hunts built on top of the platform, and incorporating other types of media such as live and historical data from sensors associated with the objects.

VI. References

[1] A. Cupper, G. Treu, and C. L. Popien. Trax: a device-centric middleware framework for location-based services. *Communications Magazine, IEEE*, 44(9):114–120, 2006.

[2] D. J. Dubois, Y. Bando, K. Watanabe, A. Miyamoto, M. Sato, W. Papper, and V. M. Bove Jr. Supporting heterogeneous networks and pervasive storage in mobile content-sharing middleware. In *IEEE CCNC '14*, 2014.

[3] W. Husain and L. Y. Dih. A framework of a personalized location-based traveler recommendation system in mobile application. *International Journal of Multimedia and Ubiquitous Engineering*, 7(3):11–18, 2012.

[4] N. Marmasse and C. Schmandt. Location-aware information delivery with commotion, 2000.

[5] A. Miyamoto, D. J. Dubois, Y. Bando, K. Watanabe, and V. M. Bove Jr. Demo abstract: A proximity-based aerial survivor locator based on connectionless broadcast. In *IEEE PerCom '15*, 2015.

[6] K. Muralidharan, K. B. Dhanapal, and A. R. Chowdhury. BOWL: Design and implementation of a (connectionless) broadcasting system over wireless LAN. In *WOWMOM*, 2008.