Distributed Emulation for the Design of Active Tag Based Systems

Razvan Beuran[1,2], Junya Nakata[1,2], Takashi Okada[2,1], Tetsuya Kawakami[3], Ken-ichi Chinen[2,1], Yasuo Tan[2,1], Yoichi Shinoda[2,1]

[1] National Institute of Information and Communications Technology, Ishikawa, Japan
[2] Japan Advanced Institute of Science and Technology, Ishikawa, Japan
[3] Panasonic Corporation, Kanagawa, Japan

***Abstract*—Several choices have to be made during the design process of active tag based systems. Since the number of properties that must be decided before production and wide-scale deployment is relatively high, the use of real experiments in the design phase may be prohibitive. We propose the use of emulation for performing large-scale experiments with active tag based systems easily and in a repeatable manner. Such experiments can be used to validate the behavior of the system, and to decide the values for various system parameters. We illustrate this approach by experimental results obtained with an emulation framework that we designed and implemented for a pedestrian localization active tag based system.**

Keywords—active tag system; distributed execution; wireless network emulation; processor emulation; pedestrian localization

I. Introduction

The design of active tag based systems has several phases until a product is ready for mass production. Hardware design is usually done in parallel with firmware design, and the system is first produced as a prototype. Typically, several parameters of the resulting system are configurable, and their values are decided by a combination of theoretical analysis and tests done using the prototype system. However, for systems with a reasonable level of complexity the number of parameter values to be decided can be significantly high. In such a case exploring the parameter space through repeated real-world experiments may be prohibitive.

In this paper we propose the use of emulation for a thorough analysis of the performance and behavior of active tag systems in realistic conditions. The pedestrian localization system that we emulated uses the data communication and processing features of active tags so as to provide to a central pedestrian localization engine the information needed to automatically calculate the trajectory and the current position of the active tag wearer. The setup also includes a number of tags with known position: fixed c-tags and gateway c-tags. Gateway c-tags can transfer information between them and to the outside world using the 802.11j standard. The data provided by gateways is used by the pedestrian localization engine.

Several real-world experiments were carried out by using a prototype localization system, as reported in [1]. The experiment consisted in the orchestrated movement, both indoors and outdoors, of 16 pedestrians wearing prototype tags, which are nicknamed communication tag, or *c-tag*. A series of hindrances were identified during the real-world trials, such as: (i) battery depletion was relatively fast and caused signal to weaken during and between experiments; (ii) orchestrating a real-world trial using even as few as 16 people was time consuming: a 15-minute trial needed hours of preparation.

This motivated the design and implementation of an alternative experiment platform using emulation, intended for the system design and development phases. This platform is based on existing tools, namely the wireless network emulator QOMET [2], and the experiment-support software RUNE [3]. Our emulation platform uses a distributed approach to achieve scalability for live execution of large scenarios.

Active tags were so far mainly studied through simulation (e.g., [4]). Some of the experiment tools for ubiquitous systems focus on the operating system level, such as TOSSIM [5], which is a TinyOS simulator. ATEMU [6] is able to emulate TinyOS applications at processor level. The manufacturer of the active tag processor (microcontroller) used in our prototype, *Microchip*, only provides two alternatives for system development: real-time emulation in hardware, or processor simulation [7]. Since none of these solutions are appropriate for our purpose, we developed our own real-time processor emulator running on standard PCs. Wireless communication emulation is currently mainly done in relation to WLANs. TWINE [8] uses computer models for performing real-time experiments so as to avoid undesired interferences and side effects. QOMET uses a similar approach, and we extended it so as to meet active tag communication emulation requirements.

An initial version of our emulation system was presented in [9]. Here we review the main system properties and improvements, and give a thorough description of how the emulation system was used for making design choices for the active tag based pedestrian localization system.

II. Emulation Framework

We use the technique of emulation to carry out experiments in a wide-range of controllable conditions, and in a repeatable manner. This implies creating a virtual environment in which the movement of pedestrians, the communication, and the behavior of active tags are reproduced in real time. Our system makes it possible to run the active tag firmware in such an emulation environment.

A. System Overview

Our system is run in a distributed manner using the experiment-support software RUNE (Real-time Ubiquitous Network Emulation environment) [3]. The experiment itself is performed using standard PCs that are part of StarBED, the large-scale network experiment environment at the Hokuriku Research Center, National Institute of Information and Communications Technology (NICT), in Ishikawa, Japan [10]. The software described in this paper can be downloaded from the following location: *http://www.starbed.org/download/*.

The pedestrian localization prototype system that we emulated uses AYID32305 tags from Ymatic Corporation, also known under the name S-NODE [11]. S-NODEs use as processing unit the PIC16LF627A microcontroller with a frequency of 4 MHz. The wireless transceiver of the prototype system operates at 303.2 MHz, and the data rate is 4800 bps (Manchester encoding), which results in an effective data rate of 2400 bps. According to Ymatic Corporation, the error-free communication range of S-NODEs is 3-5 m depending on the antenna used. These tags are emulated in our testbed both from the point of view of the wireless communication, and also at processor level.

B. Wireless Communication Emulation

The active tags employed in the pedestrian localization experiment use wireless communication to exchange information with each other. We extended the WLAN emulator QOMET [2] to support the wireless transceiver used by active tags, a task facilitated by its modular architecture. Our current model for active tag communication establishes the relationship between the distance between two nodes (a real-world scenario parameter) and the Frame Error Rate (FER, a data link layer parameter). This conversion is done based on measurements we carried out in an RF shielded room with a helicoidal antenna.

We modeled the FER versus distance dependency by fitting a second degree equation on the measurement results, done with 4-byte payload packets. Furthermore, in order to extend the communication range of the wireless network model, we further introduced the constant *C*, a scaling factor, as shown in equation (1). Note that the FER given by equation (1) must be scaled accordingly for frames with payloads other than 4 bytes, and must be limited to a maximum value of 1, since it represents a probability.

$$FER_4(d) = \begin{cases} 0, & \text{if } d/C < 0.5\,m; \\ 0.1096(d/C)^2 - 0.1758\,d/C + \\ 0.0371, & \text{otherwise}. \end{cases} \qquad (1)$$

For active tags, FER is the only communication parameter that needs to be computed. FER is equivalent to packet loss in the absence of a network layer. Communication delay and bandwidth limitation are directly recreated in the data transmission & reception process, hence they need not be emulated.

Given that the active tags are not generating IP traffic, we could not use a wired-network emulator for introducing network layer effects to traffic, as previously done when using QOMET [2]. Therefore we implemented a channel emulation system, named *chanel* (CHANnel Emulation Library). The main role of *chanel* is to recreate scenario-specific communication conditions based on the FER probabilities computed by QOMET. This function is similar to that of any wired-network emulator, such as *dummynet* [12]. A second function of *chanel* is to make sure the data is communicated to all the systems that would receive it during the corresponding real-world wireless scenario.

C. Processor Emulation

One of the advantages of network emulation is that already-existing network applications can be studied through this approach so as to evaluate their performance characteristics. Although this is relatively easy for typical network applications that run on PCs, the task is more complex when the network application runs on a special processor. In order to run the active tag application unmodified on our system, we emulated the active tag processor so that the same firmware that runs on active tags can be run in our emulated environment without any modification, nor recompilation.

When emulating active tag applications such as ours it is important to introduce cycle-accurate processor emulation. In our case active tags use the time information contained in messages to synchronize with each others autonomously. Incorrect time information may introduce problems, such as time desynchronization, and potentially communication errors, therefore it must be avoided. We tackle this issue implicitly by our distributed emulation approach: each PC in our testbed has to emulate only a few active tags; therefore we could achieve live emulation execution for over 130 active tags.

The processor emulation module is responsible for:

- Instruction execution emulation: all needed PIC instructions are supported by our processor emulator;
- Data I/O emulation: the only I/O access method used by the active tag application is USART (Universal Synchronous Asynchronous Receiver Transmitter);
- Interrupt emulation: all interrupts necessary for the active tag application are supported.

III. Emulation Experiments

We performed several emulation experiments on StarBED with the system that we developed. Some experiments focused on analyzing and extending the 16 pedestrian prototype trials carried out by Panasonic [1]. Other experiments aimed at validating our system by comparing the results obtained in an outdoor real-world trial with 2 pedestrians, and an equivalent emulation experiment on StarBED. In both these cases we could compare the results of the real experiments and those of emulation. We also made several experiments that were performed exclusively through emulation, such as several 100-pedestrian scenarios in a virtual environment representing a 300 x 300 m area in Kawasaki city (pedestrian motion patterns were created using a realistic pedestrian trajectory generator that we designed and implemented). Experiments with 100 pedestrians–a total of over 130 emulated tags–are an important achievement, since such experiments could not have been done easily as real-world trials.

In this paper we show some experimental results obtained in the 16-pedestrian and 2-pedestrian experiments that illustrate the usability of our approach. These results show how our emulation system can be used in the design process of the active tag based pedestrian localization system. The time granularity used when computing communication conditions and during real-time execution was of 0.5 s. RUNE was used to configure the host PCs and to run the experiment.

For the 16-pedestrian experiments, the initial positions of the pedestrians and their movement, the locations of the 4 fixed c-tags and 3 gateway c-tags, and the building topology were all described by converting the real-world experiment information to QOMET scenario description. Some emulation experiments were intended to analyze the influence of communication range on localization accuracy. Transmission range was modeled in our system by using equation (1), and it can be varied in the real active tag system by configuring the transmission power. Communication range is an important parameter, since it is in a direct relationship with power consumption, hence with operating time.

The results shown in Fig. 1 were obtained by considering 5 transmission ranges, between 3m and 15 m in increments of 3 m. The mean localization error shown in the figure is the average for all pedestrians; a series of 3 experiments was done for each transmission range. It can be seen that transmission range influence localization error, and an optimum is achieved for 9 m communication range. Although it was known that smaller ranges introduce errors because of a smaller communication probability, and larger ranges introduce errors because of the imprecision induced by a larger coverage area of each tag, the optimum value of the transmission range was not known. Determining this optimum value was relatively easy to be done by emulation compared to the alternative of making real-world experiments.

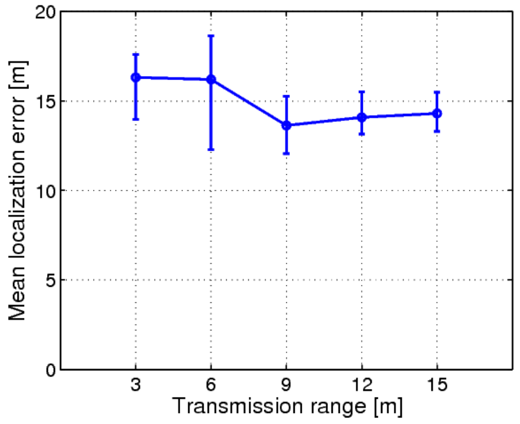

Figure 1. Mean localization error versus transmission range.

The c-tag protocol employs time-division multiplexing. Another parameter that we studied for the 16-pedestrian experiments was the number of communication slots in the time-division multiplexing scheme that provides the best overall localization performance. The prototype active tag system used 9 slots for communication, preceded and followed by one guard slot. During emulation, for two communication ranges, 3 m and 9 m respectively, we configured the active tag firmware to use 3, 6, and 9 communication slots.

Fig. 2 shows that there is a performance gain when increasing the number of slots, but this gain is not so evident for the 3 m range, since the communication opportunities are rather low compared to the case of the 9 m range. For 9 m range, the advantage of using more than 3 slots is quite visible, and an optimum performance level seems to be reached for 6 slots already. The advantage of using a smaller number of slots is that the active duration of the tag is decreased, and therefore battery life is potentially increased by 30% (when using 6 slots), or even by more than 50% (when using 3 slots). The disadvantage is that, when several tags want to communicate with each other, a smaller number of slots available for communication leads to a higher collision rate, and impedes information exchanges. From Fig. 2 we conclude that for 3 m communication range 9 slots have to be used to get the best performance, but in the case of 9 m communication range performance is significantly increased even if using only 6 slots, therefore a 30% battery life increase could be achieved.

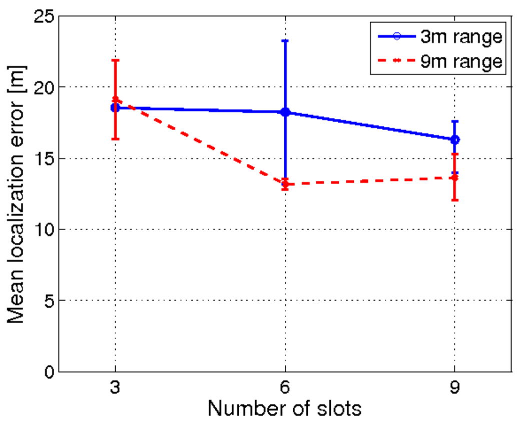

Figure 2. Mean localization error versus the number of slots used in time-multiplexed communication.

The c-tags create records of their meetings with other c-tags and store them in the internal memory until the records can be uploaded to an encountered gateway. Due to memory limitations (only 16 records can be stored), the meetings that occur within a certain time period are merged in a unique record. The length of this time period is controlled by a parameter that indicates this duration in units equal to c-tag time intervals. For the default transmission frequency and number of slots in the time-multiplexed communication, one c-tag time interval is equal to approximately 2 s. For example, if the number of time intervals for which merging occurs is 1 (0x01 in hexadecimal), the ids of all tags met during approximately 2 s are merged into one record. However, if the number is 31 (0x1F in hexadecimal), all the ids of the tags met during the corresponding 62 s period become one record. Using longer time periods allows for storing more records, but the accuracy of identifying the time (and therefore the place) where a meeting event occurred decreases. On the other hand, a shorter time period for merging increases both accuracy localization accuracy and memory utilization. Records are deleted in chronological order when memory becomes full, hence c-tag meeting history is lost. Depending on applications and deployment topologies, a trade-off must be made between memory usage and localization accuracy by selecting the appropriate value of the time interval used for merging.

Fig. 3 shows the results of the investigation we did for different values of the merging time period in the case of a 2-pedestrian experiment. The motion pattern in this experiment was relatively simple, following a T-shape trajectory with fixed tags and gateways placed at the trajectory ends and intersections. We show the average localization error in 3 experiments for each of the two pedestrians and for 5 different values of the number of time intervals used for merging.

The results in Fig. 3 indicate that for more than 7 time periods (approximately 14 s) good accuracy is obtained; as expected, and localization error decreases as the number of time intervals used when merging is smaller. However, for values of 3 or 1, the memory becomes insufficient for storing all the information until the tags arrive at the gateway located at the destination. This makes localization error increase sharply. In this experiment, by using emulation it was possible to determine, given a certain amount of memory, what is the optimum value of the number of time intervals used during merging in order to achieve best localization performance.

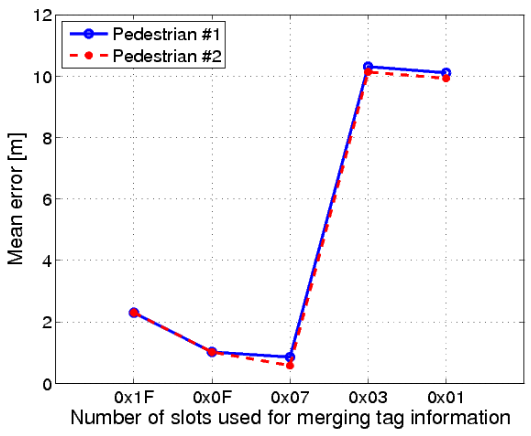

Figure 3. Mean localization error versus of the number of time intervals used for merging tag information.

IV. Conclusion

We presented our approach of using emulation in the design and development phase of active tag based systems. We illustrated this approach by a detailed discussion of the emulation framework that we developed for a pedestrian localization system using active tags. Our framework is currently used by Panasonic for the development of the pedestrian localization system, and to simplify the implementation and testing procedures of the localization engine. Employing this approach in the development phase is significantly more flexible and versatile than making real-world experiments, and helps reducing the number of real-world trials that must be carried out with the prototype system. This allows a faster product development cycle, and a quicker time to market.

In our experimental platform emulation plays an essential role at two points: (i) recreate in real time the wireless communication conditions between active tags using a simple frame error rate versus distance model; (ii) emulate in real time the active tag processor so that firmware can be run directly, without modification or recompilation. Recreating realistic experimental conditions makes it possible to validate the behavior of the system in a wide range of scenarios. In addition, one may explore with relative ease the parameter space of the system under test, so as to determine the optimum values of the parameters that produce the best overall performance under specific conditions and scenarios.

Our future work has several directions, such as: (i) improve the scalability of the system to enable emulation experiments with pedestrian groups as large as 1000; (ii) improve the realism of the wireless communication emulation by using more accurate models for electromagnetic propagation.

References

[1] Y. Suzuki, T. Kawakami, M. Yokobori, K. Miyamoto, "A Real-Space Network Using Bi-Directional Communication Tags – Pedestrian Localization Technique and Prototype Evaluation", *IEICE Forum on Ubiquitous and Sensor Networks*, tech. report, October 30-31, 2007.

[2] R. Beuran, L. T. Nguyen, K. T. Latt, J. Nakata, Y. Shinoda, "QOMET: A Versatile WLAN Emulator", *IEEE 21st Int. Conf. AINA-07*, Niagara Falls, Ontario, Canada, May 21-23, 2007, pp. 348-353.

[3] J. Nakata, T. Miyachi, R. Beuran, K. Chinen, S. Uda, K. Masui, Y. Tan, Y. Shinoda, "StarBED2: Large-scale, Realistic and Real-time Testbed for Ubiquitous Networks", *TridentCom 2007*, Orlando, Florida, U.S.A., May 21-23, 2007.

[4] A. Janek, Ch. Trummer, Ch. Steger, R. Weiss, J. Preishuber-Pfluegl, M. Pistauer, "Simulation Based Verification of Energy Storage Architectures for Higher Class Tags supported by Energy Harvesting Devices", *Conf. on Digital System Design Architectures, Methods and Tools (DSD 2007)*, Lubeck, Germany, Aug. 29-31, 2007, pp. 463-462.

[5] P. Levis, N. Lee, M. Welsh, D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications", *Proc. of the First ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, California, U.S.A., November 5-7, 2003.

[6] J. Polley, D. Blazakis, J. McGee, D. Rusk, J. S. Baras, "ATEMU: A Fine-grained Sensor Network Simulator", *Proc. of the First IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, Santa Clara, California, U.S.A., October 4-7, 2004.

[7] Microchip Technology, Inc., *http://www.microchip.com/*.

[8] J. Zhou, Z. Ji, R. Bagrodia, "TWINE: A Hybrid Emulation Testbed for Wireless Networks and Applications", *IEEE INFOCOM 2006*, Barcelona, Spain, April 23-29, 2006.

[9] R. Beuran, J. Nakata, Y. Suzuki, T. Kawakami, K. Chinen, Y. Tan, Y. Shinoda, "Active Tag Emulation for Pedestrian Localization Applications", 5th International Conference on Networked Sensing Systems (INSS08), Kanazawa, Ishikawa, Japan, June 17-19, 2008, pp. 55-58.

[10] T. Miyachi, K. Chinen, Y. Shinoda, "Automatic Configuration and Execution of Internet Experiments on an Actual Node-based Testbed", Proc. of Tridentcom 2005, Trento, Italy, February 2005, pp. 274-282.

[11] Ymatic, Inc., http://www.ymatic.co.jp.

[12] L. Rizzo, "Dummynet FreeBSD network emulator", *http://info.iet.unipi.it/~luigi/ip_dummynet/*.