Demo Abstract: Whac-A-Bee – A Sensor Network Game

Eugen Berlin * Pablo Guerrero † Arthur Herzog †‡ Daniel Jacobi †‡
Kristof van Laerhoven * Alejandro Buchmann † Bernt Schiele *
† Databases and Distributed Systems Group
‡ Multimodal Interactive Systems Group
\{lastname\}@dvs.tu-darmstadt.de
\{lastname\}@mis.tu-darmstadt.de
Department of Computer Science, Technische Universität Darmstadt, D-64283, Germany

Abstract
This paper illustrates both challenges and benefits found in expanding a traditional game concept to a situated environment with a distributed set of wireless sensing modules. Our pervasive game equivalent of the Whac-A-Mole game, Whac-A-Bee, retains the find-and-seek aspects of the original game while extending the location, the number of players, and the time-span in which it can be played. We discuss the obstacles met during this work, and specifically address challenges in making the game robust and flexible enough for large and long-term deployments in unknown territory.

Categories and Subject Descriptors
C.2.4 [Computer Systems Organization]: Distributed Systems—Distributed applications; K.8.0 [Computing Milieux]: Personal Computing—Games; C.2.1 [Computer Systems Organization]: Network Architecture and Design—Wireless communication

General Terms
Algorithms, Design, Performance

Keywords
Pervasive Game, Wireless, Sensor Network

1 INTRODUCTION
Pervasive games have been emerging ever since tiny computers started being attached to everyday objects [1, 2]. Prototypes have been developed which show that innovative gameplay can indeed be implemented on a set of embedded and networked devices that are mobile or distributed in the environment. However, the real challenges faced by these games, common to other sensor network applications, have to be addressed. To increase understanding in this area, we study the development of a particular pervasive game and describe the game design choices that led to the final system.

‡Supported by the DFG Graduiertenkolleg 1362, “Cooperative, Adaptive and Responsive Monitoring in Mixed Mode Environments”.

2 THE GAME
Whac-A-Bee is the pervasive, real-world equivalent to the arcade game Whac-A-Mole™, in which the player uses a mallet to hit moles as they randomly and suddenly pop up from their burrows for a short time. The wireless sensor network version described here consists of bees and hives instead of moles and burrows, respectively. The hives and bees are materialized with stationary sensor nodes and colored LEDs, while laser pointers are used to capture the bees by activating light sensors (replacing the mallet). Each player carries a node that provides feedback about the game state. Player teams compete with each other in one of three playing modes - a) capture limit, b) time or c) combination of both - to capture as many bees as possible.

Building a usable pervasive version of this game implies it must be playable. This is translated into the following requirements: robustness against communication and node failures; responsiveness in reacting to events and providing feedback to players; fairness among teams, and game parameters’ customizability. At the forefront of any sensor network application there is system lifetime, requiring low power operation. And lastly, decentralized operation was favored, only allowing centralized game configuration, start and stop commands, calling for scalable distribution of game logic.

3 SYSTEM ARCHITECTURE
In this work, power consumption is antagonist to all other requirements. Communication must be kept minimal, but robustness must not be jeopardized. Initially, game configuration parameters are propagated from the base station, building routes and triggering neighbor list creation. Later, during the game, route requests can be issued if a parent node drops or a communication failure occurs, resembling an AODV behavior. These routes are later used to update the game state at the server. Messages that need to be broadcast through the entire network, for example game events such as “bee pops up” or “bee hides”, are distributed using a combination of gossiping and Only One Transmits (O1T) [5]. Each game starts with one bee per team. The bee wanders from hive to hive hoping not to be whacked. The hive election must appear random to players (causing them to chase bees through rooms). This was implemented as a token-passing protocol following a random walk over node’s neighbor lists. A two-way handshake token-passing mechanism was necessary, as our empirical indoor packet loss rates were about 4%.

Copyright is held by the author/owner(s).
SenSys 09, November 4–6, 2009, Berkeley, CA, USA.
ACM 978-1-60558-748-6
The game has been implemented in SOS [3]. All sensor nodes are programmed with the same modules, thus the nodes can be deployed without prior configuration. A GUI is used to register the desired amount of players and teams, set up game parameters and start the game. One node is chosen as base station, bridging messages between the network and the PC running the GUI and an SOS server. When the game starts, one bee per team is injected at the base station. All further decisions on hive election are handled distributedly inside the hive network. Despite the two-way handshake token-passing, in isolated cases a bee token gets lost. Thus, the base station monitors the game flow, and, if necessary, injects a new bee token.

Hive nodes are visited by bees for a random, but bounded time period. A bee is either whacked timely, or it wanders elsewhere. Naturally, only hives with bees enable the light sensors. Events of global relevance such as game finalization (when the capture limit is reached) can be sent by any node, and are broadcast with O1T, minimizing wireless communication. No game finalization notifications are needed in a timed game, as every node knows when the game ends.

Nodes carried by players provide visual feedback whether their target bee is currently hiding or has popped up at a hive, as well as how far away it is. In order to provide this information, visited hive nodes initiate a simplified DV-hop [4] wave with themselves as anchors. When a player’s node is notified about its bee’s popping up somewhere, it starts requesting and displaying the proximity in terms of network hops. Proximity feedback is implemented as follows. Player nodes query direct (i.e., 1-hop) neighbor hive nodes periodically for the proximity toward the selected, currently visited hive node. The query includes a threshold, and only nodes located at a distance below this threshold reply back. With each new bee popping up, corresponding player nodes reset their threshold distance to infinite. From all replied distances, player nodes retain and show the shortest. If a player comes closer to the selected hive, the distance decreases. If the player moves away, no shorter reply will be received, causing the player node to re-query neighbors with an increased distance threshold. This gives the player a rough idea where to find the bee that has popped at a hive. This process continues until the bee hides or is whacked (either by the player, a team member or even by an opponent).

This distributed system approach aids the scalability of the system, since more nodes can be easily added to the hive network and initialized for a new game. Since 3 LEDs are available for player feedback, we limit the number of teams to 3, but with an unlimited and variable number of team members. This allows us to assemble balanced and fair teams with respect to the skills of the participants. In this way, an advanced player is able to compete against a team of novices.

4 DEMONSTRATION OVERVIEW

We have deployed the game at TU Darmstadt CS building’s testbed at several occasions, with up to 25 nodes spanning up to 8 offices and a corridor. At SenSys we intend to employ around 25 TelosB as hive nodes around the demo booth, complemented with a set of player nodes that conference attendees can use to play a short session in any of the gaming modes and configurations discussed in this abstract.

Whac-A-Bee is a game that contains all the ingredients a sensor network has to offer, and faces the same challenges other WSN applications deal with. This demonstration shows the suitability of the design decisions to implement a game that is distributed, robust and energy-efficient.

5 References