

CSC2231 PROGRESS REPORT 1

This report will give a brief description of the system we have designed using a system architecture diagram showing the various components and how they interact. Rationales behind design choices are presented and a preliminary UI solution to the distributed two-phase commit algorithm is presented. Milestones are included for the next six weeks.

THE PROPOSED SYSTEM

Shopping requires frequent visits to too many locations. In this project, we propose a system to allow friends in an online social network to share “to do” lists so that they can coordinate their efforts in the real world. This system will tie the social network’s master “to do” list to location based services so that group members know when they are in a position to help out a neighbor in the network. In this architecture, a distributed locking algorithm is unreliable and slow. Instead, a novel user interface will be developed so that users can participate in a distributed, two-phase commit system; the awareness of the status of other collocated users facilitates the coordination among them to complete tasks on the shared to-do list without duplication of effort.

SYSTEM ARCHITECTURE OVERVIEW

There are three major components in our proposed system: Cellphones, Beacons and the Social Network Server. Figure 1 shows an overview of the components of our system. Cellphones and Beacons are connected over Bluetooth, and Beacons and Social Network Server communicate with each other through the Internet. The arrows represent the flow of messages. For the details of how message passing in our system works, please see the USER SCENARIO section.

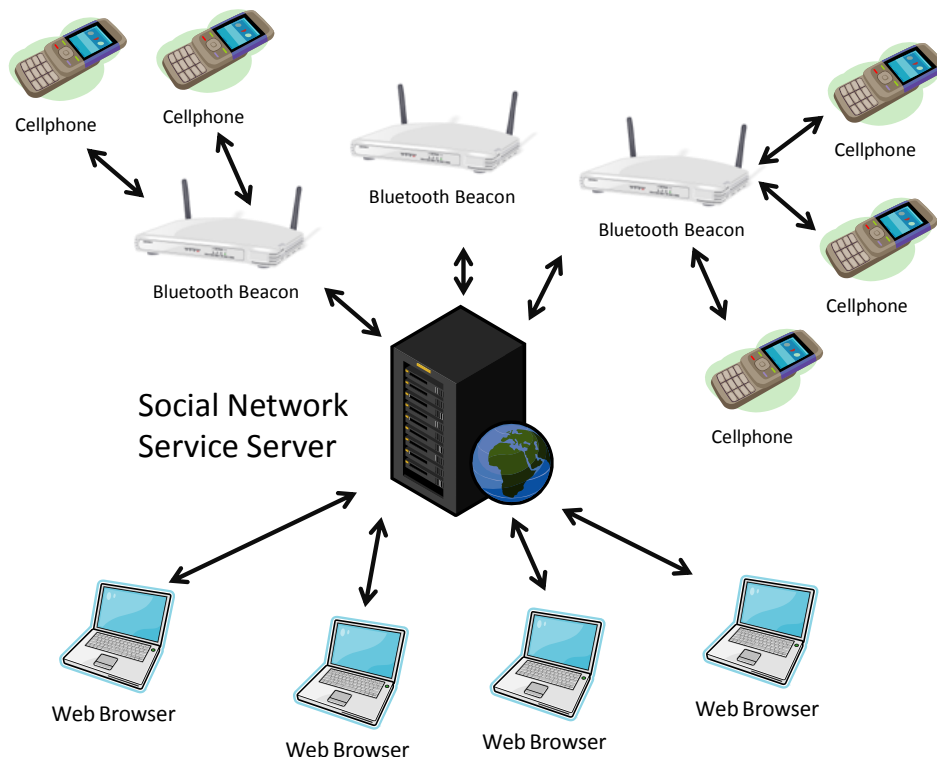


FIGURE 1 - SYSTEM OVERVIEW

USER SCENARIO

We present a simple user scenario to see how our proposed system works. In this scenario, Mary wants a Starbucks Latté and John is going to a Starbucks store soon.

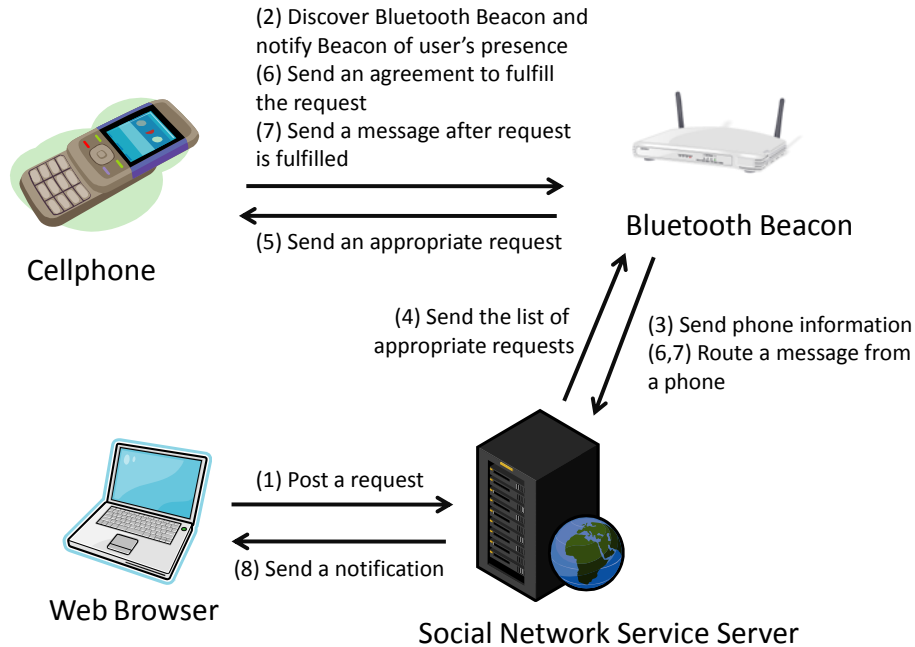


FIGURE 2 - SYSTEM ARCHITECTURE

1. Mary logs into the Social Network Server website and enters a request for a Starbucks Latté (Figure 2(1)).
2. John, one of Mary's friends, walks into a Starbucks, and his Bluetooth-enabled phone registers his presence with the Bluetooth beacon that is present at that location (Figure 2(2)).
3. This beacon passes John's phone ID, along with its own location information to the social network server (Figure 2(3)), which replies with a list of open requests that are applicable to that beacon's location and Mary's friendship (Figure 2(4)).
4. The beacon examines the list of requests, and finds that John's phone is appropriate to notify about Mary's request. The beacon contacts John's phone and tells him about this drink that Mary wants (Figure 2(5)).
5. John decides to purchase the drink for Mary and confirms this through the cell phone's UI. A message is sent to the Social Network Server via the beacon and Mary's request is locked so that no other people will buy Mary a Starbucks Latte (Figure 2(6)).
6. John approaches the counter, orders, pays and receives the drink.
7. John then taps the appropriate button on his phone to confirm with the system that he has the drink in his possession. The beacon is notified, which informs the Social Network Server, which updates the status of the drink order (Figure 2(7)). Mary is notified accordingly (Figure 2(8)).
8. John brings the drink to Mary and the necessary payment is given.

DESIGN RATIONALES: PROBLEMS AND SOLUTIONS

We foresee some problems during the development of our proposed system. Some have been addressed or mitigated, while others remain open and unresolved.

PROBLEM: PHONES MAY WANDER IN AND OUT OF A VENUE AND THE BEACON IS THE ONLY RELIABLE PIECE OF THE ARCHITECTURE AT THE VENUE.

Solution: The beacon must continuously ping the phones, and ensure they are still in range. If a phone leaves, the beacon is the only “reliable” device in the venue that can successfully unlock the request on the SNS. Agreed-to requests can be restarted by the beacon if a phone returns into range and the request has not been fulfilled. Users who agree to requests, leave, and then return to venues must be notified if their task has been fulfilled by another.

PROBLEM: BEACONS ARE LIKELY SITTING ON PUBLIC NETWORK CONNECTIONS AND ARE UNABLE TO ACCEPT INBOUND CONNECTIONS FROM THE SNS.

Solution: The beacon logic must initiate all communications to the SNS and keep the entire system state consistent. The beacons must be aware of all active phones in the area, and any active requests that have been broadcasted to the phones. The system will use HTTP GET/POST requests with high timeout to simulate the “push” functionality. As soon as new requests are posted to the SNS, the GET requests will be filled by the webserver, and the beacons will be instantly aware of updated requests.

PROBLEM: USERS MAY NOT WANT TO PARTICIPATE, OR NEED TO TURN OFF REQUESTS AS NECESSARY. BLUETOOTH IS BAD FOR BATTERY LIFE.

Solution: Phones are proactively and constantly looking for beacons, and actively seek requests to fulfill. This ensures that the user is in complete control and can opt-out if necessary. If a user does not wish to participate in this exercise, the particular functionality can be easily controlled by the end user. This ensures the user will not be bombarded with requests. Additionally, the SNS can limit the number of requests sent to a single user in a specified period (e.g. one per day).

PROBLEM: PEOPLE MAY COMMIT TO BUY A DRINK AND THEN DECIDE NOT TO OR MAY DECIDE TO PURCHASE DRINKS ONLY IF NO ONE ELSE IS AROUND. SIMPLE 2-PHASE COMMIT IS NOT ENOUGH SINCE WE CAN'T CONTACT PHONES SEQUENTIALLY – WE'D BE MISSING OUT ON TOO MANY OPPORTUNITIES. SOME USERS MAY REQUIRE ADDITIONAL CONTEXT INFORMATION TO MAKE PURCHASING DECISIONS.

Solution: We are developing a status UI that will allow 2 or more users to “negotiate” who will purchase the drink. By looking at the display, any notified user will see what the “other” users are doing; is someone else more likely to buy the drink? See the “UI Sketches” section below.

PROBLEM: COMMUNICATION BETWEEN BEACON AND SOCIAL NETWORK SERVER MAY BE UNRELIABLE.

Potential solution: Caching and merging logic may be necessary on the beacon. Regardless, system should ensure a state of correctness in the face of network troubles.

PROBLEM: USERS WHO REQUEST PURCHASES MAY NOT CHECK SOCIAL NETWORK WEBSITE CONSTANTLY.

Potential solution: Server may notify users directly via email if there are status updates for their requests.

WHAT'S BEEN BUILT

- Detailed system architecture has been documented, along with Java interface classes set up on CVS
- Obtained development cell phones and installed Java-based SDK for cell phone development (S60 based Nokia N93 and N95 phones)
- Written simple "Hello World" Bluetooth Discovery application for cell phone
- Written simple "Hello World" Forms-based UI for cell phone
- Set up Social Network Server HTTP server with backend MySQL

SCHEDULE FOR COMPLETION OF REMAINING COMPONENTS OF PROTOTYPE

2nd week, November (4th ~ 10th)

- Building the user registration system and the friend list system in Social Network Server
- Implementing Bluetooth service on the Beacon and Bluetooth discovery on the cell phone
- Implement cell phone UI in Java for request accept/decline/confirm

3rd week, November (11th ~ 17th)

- Building the request posting system in Social Network Server
- Enabling Social Network Server and Beacon to communicate with each other via HTTP
- Enabling Beacon and cell phone to communicate via Bluetooth once service is discovered

4th week, November (18th ~ 24th)

- Implementing minor functions (like notification via email) in Social Network Server
- Refining the algorithm for picking up appropriate requests

5th week, November (25th ~ 1st December)

- Debugging the system in Social Network Server
- Writing the second progress report

1st week, December (2nd ~ 8th)

- Testing the whole system

2nd week, December (9th ~ 15th)

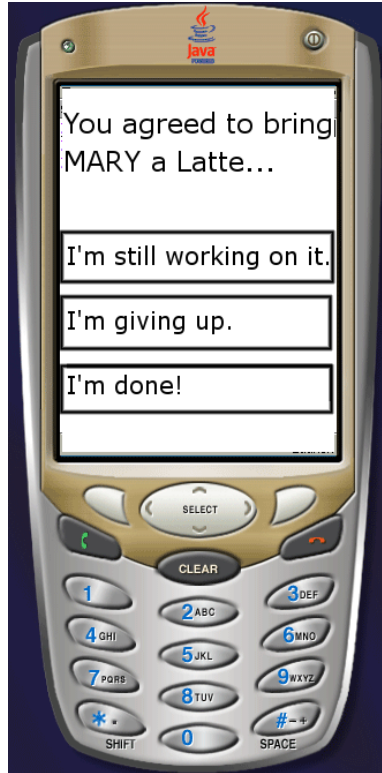
- Taking a demonstration video
- Finishing the final report
- Preparing the presentation

UI SKETCHES

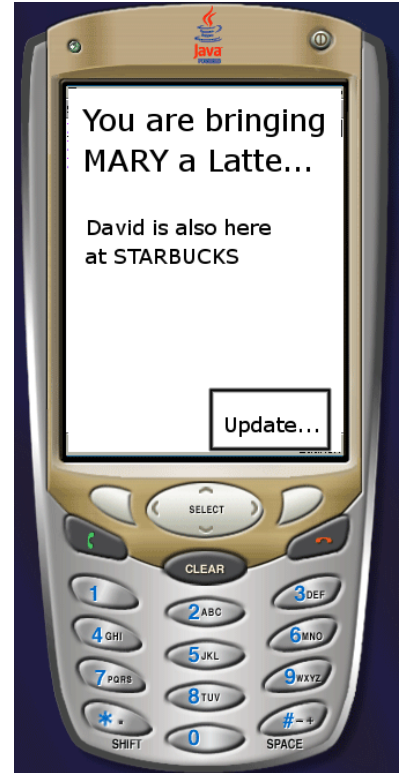
The cell phone UI is very minimal so that interactions with the system are lightweight. There are two states in which the cell phone polls the user for an answer, when a request is first made and when a confirmation is sought by the beacon. While a user is fulfilling a request, a status screen shows the details of the request as well as information about friends in the same location.



REQUEST SCREEN



CONFIRMATION SCREEN



STATUS SCREEN