P2P Game Network Framework

Communications and Context Framework for Building P2P Multiplayer Games

Andrés Felipe Castaño Henao (acastano@gmail.com)
Juan David Hincapié Ramos (jhincapie@gmail.com)
Professor: Edwin Montoya (emontoya@eafit.edu.co)
Date: June-2007

Abstract

This work presents an approach to designing a software framework for developing multiplayer games: the P2P Game Network Framework, centered in network communications and interaction with the context. This general design assumes a group of devices organized in a P2P network, implying the necessity of tools for self-organization of the group and group synchronization. These tools aim to guarantee that all the peers within a group share identical values for state variables and other mechanisms for dealing with failure or unexpected abandoning of the group by any of its members, minimizing the impact for the whole group. Working under the P2P model brings gains in flexibility and deployment costs for multiplayer games compared to other server based solutions. This project has also implemented the defined framework on the J2ME platform using Bluetooth for the transport layer.

Key Words

Java Micro Edition (J2ME), Bluetooth, Multiplayer Games, Mobile Computing, P2P, Framework

Introduction

Mobile devices development has been accelerated by the improvements of the hardware components, the rising of new technologies and the increased market penetration. These devices range from notebooks to PDAs to mobile phones, with every time more and better services like WiFi, Bluetooth, UMTS, high-res screens, high performance operative systems, etc. And specially for mobile phones we are seeing penetration rates of up to 109% or 108% for countries like Sweden and Italy among others[1]. At the same time, it's remarkable the way how wireless communications technologies have increased their capacities, transmission rates, coverage ranges, and still continue reducing the cost per hardware component.

Innovations in the mobile devices and wireless networks fields have led the development of new kind of applications according to the new scenarios. Fields like the Pervasive or Ubiquitous Computing stand out by using all these new non-conventional devices and scenarios for providing new innovative services. And once again it's entertainment services and applications the ones leading the way: downloads for mobile phones, resource sharing among users and specially mobile games (single or multi player) try to make extensive use of the new possibilities[4].

This industry has grown up from the first snake game to the latest 3D games in the last years, creating an industry that pushes for the continued development of the devices in terms of processing capacity, interactivity and connectivity. The first mobile games (single user) were developed for the so called “casual player”[1], leaving out the “hardcore player”[2] due to the restricted capabilities of the first devices. The vast variety of devices now in the market, their processing power, their GUI capacity and communication capabilities are bringing the “Hardcore Players” into the field, seeking to exploit the newly available options like mobility and ubiquity, and at the same time keep the already existing advantages of the console and PC games like the graphics and the

1 Casual players are those that don't really spend a significant amount of time playing or/and that prefer easy and not time consuming games.
2 Hardcore players defines those players who spend most of their spare time to video games, it could be playing, reading or discussing about them.

multiplayer environments.

Constructing multiplayer games in these new scenarios brings to the table many challenges. In some cases there would be connectivity enough to reach the Internet and therefore the high capacity servers[5]. Some others cases would be better described as groups of devices disconnectied from the Internet but robust enough to establish a multiplayer game session. Besides the communication capabilities of the devices, new elements are now available in the interaction of devices with themselves, the group and the environment. These elements collect information from the context[3] of the device.

It's then relevant for the academy and the industry to start moving towards the creation of frameworks (both methodological and software libraries) to leverage the development of applications within these new technological scenarios. Primarily pushing the gaming and entertainment industry and later by integrating them into the corporate, health care and educational world, etc. And of course, not only restricted to mobile devices[4].

Multiplayer Games

Are the ones that allow multiple player to play simultaneously. In our case, the interaction is done by using computing devices.

Multiplayer Games Break Down

Games can be classified by gender(action, adventures, strategy, shooter, role, races, combat, sports, puzzles, cards, music, board, etc.) or by their technical/theoretical characteristics. This second case is what interests us:

- Solo Games: Each round every player has a complete game session, independent from the other players' sessions. A winner will be chosen at the end by comparing their scores.
- Turn-Based Games: The game session is divided in discrete turns. The game will either have a single player per turn or all the players acting in every turn.
- Act-Whenever Games: This games usually last for a long time (months or years) and every player can come in at any moment and interact.
- Slow-Update Games: The agents configured by the player stay constantly interacting with the game according to some behavioural rules given by the player.

Device Architectures

As every player has a different device, the connection between devices is usually given by the following patterns:

- Dedicated Server: There is a fixed given server or list of servers hosting the multiplayer features. The handling of the different features can also be done in different servers: billing server, lobby server, game-play server, messaging server, resources server, etc.
- Servers Cluster: Similar to the previous one. In this case each server can be a cluster of server, providing the game provider with greater flexibility, higher QOS, etc.
- Ad-Hoc Server: Each client application is fully enabled to act as a local network multiplayer server and host a game session. In a future session any other client application can act as the server.
- P2P Networks: They are based on the concept of grouping and sharing the resources of every device member of the net, abolishing any distinction between servers and clients. Once in the P2P network the devices (aka peers) will exchange messages (either directly or by proxy) and it is up to every single game instance to control the interactions.

---

3 Context are the circunstances in which a device is being used according to the Systems Enginnering.

4 According to Gartner, in 2005 the corporate investment in wireless technologies was TOP 3 in the technological strategies of European companies - http://www.gartner.com/press_releases/asset_125194_11.html.

Communication Technologies

The following list shows the most commonly used technologies for the transport and communications layer for multiplayer games:

- Serial Port: One to one connections via modem or serial port. It isn't popular anymore due to the bandwidth limitations, the high costs related to using a modem and the low scalability.
- Infrared: Wireless one to one connections with reduced mobility. Its use by providers is very restricted due to the low bandwidth, signal range and the necessity to have “visual” contact from one to another device.
- UDP: Stateless connections with reduced network overhead.
- TCP: State-full and trusted connections creating some network overhead.
- HTTP Polling: Repetitive http request to a Game Server. Each request optionally has a message about the player's state or move and expects an update on the game state from the server.
- Bluetooth: Limited number of devices wirelessly connected and highly mobile in short range areas.
- SMS/MMS: Asynchronous connections between multiples devices or between the devices and a game server. There is a considerable delay for the reception of the messages and it's not guarantied. The packet length is also limited (at least in SMS messages).

There are many combinations between the devices architecture and the communication technology being used in the market. The chosen combination for a game depends on the target device and the business model of the game developer.

Common Problems

The following problems are to be controlled as much as possible due to their negative impact in the user experience:

- Cheating: Any kind of cheating can be trace down to errors or misconceptions in the design of the game software. The game providers have developed different tools to deal with this kind of problems like monitoring systems, continuous updates and even penalties to the players.
- Lag: The interaction of players with differences in network capacity usually results in giving advantages to the one with the higher specs.
- Packet Loss: A high packet loss rate can cause the game to be inoperable.
- User Drop: For games that rely on a high player-to-player interaction the unexpected disconnection of one player can cause a deadlock in the game or even the end of the game session.

Why P2P Game Network Framework?

Every multiplayer game development project has to design and implement some features common to all of them like discovery of devices and game instances, connection, reconnection, packet exchange, game session management, etc. The P2P Game Network Framework provides a set set of services and protocols that standardize the way these common functionalities should work, presenting however lose coupling requirements to the game application. The services (and protocols) standardize the following functionalities:

- Message handling and delivery.
- Peer and game session discovery.
- Peer monitoring during the game session.
- Game session access control.
- Object synchronization between peers.
- Game continuity assurance.

The design of the framework is such that it could be implemented in almost any computing platform:

- There are no requirements about the transport network technology or it's topology.
- There are no requirements on the programming language or the runtime for the implementation.

P2P Approach

The traditional client/server approach to multiplayer games has the following disadvantages [6]:

- Network access and message processing bottlenecks.
- Unique failure point.
- High maintenance costs.
- Requires connectivity from the client to the server.

All these disadvantages are important when proposing any multiplayer gaming solution, especially as mobile devices become more and more present in our lives and are to be used as gaming platforms (mobile devices have increasingly big processing power and local area communication capabilities, though connecting to an Internet server is still very slow and costly). A P2P approach provides an interesting alternative to cope with all these situations at once.

There are many definitions to what is or not P2P, we will take the one from the openP2P[7] project group which says that an application is considered to be P2P always that answers positively the next two questions:

- Is variable connectivity and temporary network addresses considered as the norm?
- Gives enough autonomy to all the peers on the edge of the network?

P2P Game Network Framework has been designed to fully embrace the P2P paradigm.

Services and Protocols

The framework is formed by four services that altogether implement the aforementioned functionalities. The framework defines the services in terms of the relations between them and with the game application. However there are some details that are left out to be defined when implementing it due to the high dependency with the development/runtime platform:

- Connection Service: Handles the delivery of messages between peers. It makes use of the Connection Protocol (CP) to package messages coming from higher layers. When the peer is acting as the session master this service performs the message-forwarding to all the peer in the game session.
- Discovery Service: Handles the session discovery and peer monitoring functionalities. The peer discovery and monitoring are not specified by the framework due to the high dependency to the runtime/transport technologies. Once the peers are discovered the game session discovery routine starts by using the Discovery Protocol (DP).
- Group Management Service: Guaranties the peers group coordination by controlling the access to the game session and the synchronization mechanisms, both by means of the Group Management Protocol (GMP).
- P2P Network Service: This service is the entry point for the game application to the framework. It encapsulates all the functionalities and offers access points to them.

Master and Slaves

The framework's approach is that of an hybrid P2P network. That entails the existence at some point of time of a master peer and zero or many slave peers. The master is required to be reachable by all the peers and performs the following tasks:

- Forwards messages from any peer to the rest of the peers in the game session.
- Controls the access of new peers to the game session.
- Initiates and orchestrates the game session synchronization process.
- Organizes the “game session continuity vector” which is a prioritized list for who would be the next master in case the actual one leaves or fails. In this vector are listed all the peers in the game session.

The slave peers have a direct interaction with the master peer in terms of:

- Message passing to the group through the master.
- Reception of messages from the group through the master.
- Synchronization process initiation request to the master (it does not forces the master to start the synchronization process).
- Taking part of the synchronization process after being initiated by the master.

The prioritization of the “game session continuity vector” depends on information gathered from the context. This information can be the number of peers in the session that a peer can reach, the available bandwidth of the device, the variability of the location of the device, the remaining battery charge, etc.

Game Session Continuity/Recovery

Throughout a game session it could happen for a certain peer or for a group of them (maybe all) that the master peer is unreachable due to some failure or because is out of range. At that moment the framework should trigger a process to recover the game session and allow the player to continue playing, trying to keep the maximum number of former players in the new session. From this recovery process results one or more new masters, each with a game session and a group of peers. The recovery is performed according to the next algorithm:

1. If the next peer in the “game session continuity vector” is the local peer, then the local peer turns into the game session master and creates a new session identifier. If there are some queued requests as described in step 3 then they are given a positive answer and returns.

2. For each peer in the “game session continuity vector”, if it's reachable (this is known due to the context monitoring tools) then try to join its session with JoinSessionMessage from GMP.
    1. If the answer is negative and there are more peers in the vector then goes back to step 1.
    2. If the answer is positive then gives a negative answer to all the queued requests and returns.

3. If while executing step 2 the peer receives a JoinSessionMessage it gets queued up for later response.

4. If the peer has not started and session recovery process and a JoinSessionMessage is received then it assumes that there is some problem with the master peer and proceeds to check the connection with it:
    1. If the local peer has also lost connection to the master peer, the request is queued up and goes to step 1.
    2. If the peer still has connection to the master it answers negatively to all the queued requests and returns.

J2ME/Bluetooth Implementation

The framework has been implemented on J2ME according to the general specification and implementing the device discovery and monitoring through the Bluetooth JSR 82 API. Some annotation regarding the implementation are:

- Coverage Area: By definition the coverage area of a Bluetooth network is 10 meters.
- XML: The processing of XML messages requires a high processing and memory load.

Future Work

There is the opportunity to continue the project's development in the following areas:

- Change device monitoring to application monitoring: The device monitoring does not guaranty that the application is still running inside the device (it could have hung or be stopped by the user).
- Wireless Mesh Networking: Removal of a master peer to achieve gains in issues like coverage area and load balancing.
- Statistics and System Logs: The framework should collect information about its functioning in things like network status, jitter, latency, network coverage, performance, etc. All this collected both in runtime variables and system logs for error tracking.

- J2ME/Bluetooth Implementation Improvements: The current J2ME implementation can be improved considerably in the following issues:
  - Synchronization Rate: Find the optimal synchronization rate while operating normally and determine the right events that trigger the synchronization process.
  - Simultaneous Connections: Implement a connections pool from the maximum number of connections supported by the Bluetooth implementation of the device.
  - Library Footprint: The footprint (both physical and runtime) can be reduced by using obfuscation techniques and garbage collection mechanisms.
  - Error Management: This can be improved and potentially included in the frameworks general design.
- Other Implementations: The framework can be implemented on other runtime and transport technologies like Windows Mobile over GPRS or Linux over TCP/IP.

Conclusions

The P2P Game Network Framework proposes a basic structure for building communication and context frameworks used in the construction of multiplayer games and potentially other communication applications.

The framework takes an hybrid P2P approach which gives advantages like a game session recovery system to provide a more robust infrastructure to avoid an unique failure point scenario, enables a cheaper deployment (there are not installing and maintenance cost for intermediate game servers) and provides mechanisms for game session continuity despite the exit of other peers.

The actual framework implementation on the J2ME/Bluetooth platform presents some limitations which can be directly linked to the limitations in the java virtual machine and Bluetooth API implementations. It's expected the future implementations of these two technologies to be more stable and adjusted to the specifications.

Application developers can use the J2ME/Bluetooth implementation of the framework to build applications. For the test applications developed for the project only the 7% of the code was application specific, the rest was the framework's code.This fact show how the framework can decrease the costs and time invested developing the projects.

Last, the framework provides the bases for building implementations of it on other platforms. That way a game can be developed in several platforms without changing neither its programming model nor its functionalities.

References

1. Ibrahim, J. 4G Features, Bechtel Telecommunications Technical Journal.
2. Organization for Economic Co-operation and Development. Digital Broadband Content: Mobile Content, New Content For New Platforms, 2005. http://www.oecd.org/dataoecd/19/7/34884388.pdf
3. David Linner, Fabian Kirsch, Ilja Radusch, Stephan Steglich. "Context-aware Multimedia Provisioning for Pervasive Games," ism, pp. 60-68, Seventh IEEE International Symposium on Multimedia (ISM'05), 2005.
4. Alf Inge Wang, Michael Sars Norum, Carl-Henrik Wolf Lund, "Issues related to Development of Wireless Peer-to-Peer Games in J2ME," aict-iciw, p. 115, Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06), 2006.
5. http://eqplayers.station.sony.com/index.vm, EverQuest.
6. Abdulmotaleb El Saddik, Andre Dufour. "Peer-to-Peer Suitability for Collaborative Multiplayer Games," Multimedia Communications Research Lab (MCRLab) University of Ottawa, Ottawa, Canada, K1N 6N5
7. http://www.openp2p.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html?page=2, "What is P2P… and What Isn't [Nov. 24, 2000]"