Multiplayer computer games, mobile phones and beyond

Michel Simatic

CSC5004

January 16th, 2009
Outline

1 Background .................................................................................. 3
2 Introduction .................................................................................. 5
3 Main constraints brought by video games........................................... 14
4 Peer-to-peer architecture ................................................................. 25
5 Client-server architecture ................................................................. 36
6 Server-network architecture ............................................................... 46
7 Mobile games ............................................................................... 53
8 Pervasive and ubiquitous games......................................................... 65
9 Game communication middlewares .................................................... 78
10 Conclusion .................................................................................. 96
1 Background

1.1 About the speaker
1 Background

1.1 About the speaker

- Worked for Alcatel from 1990 to 2002
- Since 2002, lecturer-researcher at TELECOM & Management SudParis
- Research on video games on mobile phones.
  
  ◆ Former leader of JEMTU project
  - Goal: Contribute to platforms for games on mobile phones following a technological axis and a usage axis
  - 11 researchers from Institut TELECOM
  - Support by TELECOM Foundation (Alcatel-Lucent, BNP Paribas, Orange and SFR)
  - Dates: 01/01/2006 - 31/12/2008
  
  ◆ Co-leader of PLUG project
  - Goal: Study embedded technologies to develop pervasive and ubiquitous games and their acceptability form a sociological, cultural, economical and industrial point of view
  - 4 academics, 3 industrials and 1 museum
  - Support by ANR
  - Dates: 01/01/2008 - 31/12/2009
  
  ◆ Collaboration with CNAM and ENJMIN
  
  ◆ Member of Video games workgroup of Cap Digital industrial pole
2 Introduction

2.1 Few figures about video game industry ........................................... 6
2.2 What makes multiplayer game such an interesting distributed application .... 10
2.3 Different types of games ................................................................. 11
2.4 Two levels of functionalities in multiplayer games ............................. 13
2.1 Few figures about video game industry

- In 2007 [Ducourtieux, 2007]
  - World (USA + Europe + Japan)
    - Software: 20.7 billion €
      - Consoles: 51.7%
      - Portable Consoles: 18.8%
      - PC: 17.8%
      - On-line: 5.8%
      - Mobile phones: 5.3%
    - Consoles: 9.8 billion € (5.6 billion € for non-portable consoles)
    - Total is at the same level as music industry
  - France
    - Total = 2.4 billion € (+ 65% compared to 2006)
    - In 2007, 23% of French families bought a video game product (+ 5%) for an average cost of 400 € (+ 27%) [Agence Française pour le Jeu Vidéo (AFJV), 2008c]

- In S1 2005, French sales podium [Agence Française pour le Jeu Vidéo (AFJV), 2006]
  1. Game *Gran Turismo 4*: 19 million €
  2. DVD *Shrek 2*: 16 million €
  3. CD *Les enfoirés*: 15 million €
  4. Book *Da Vinci Code*: 11.8 million €
Few figures about video game industry (2/4)

- Casual game revolution
  ♦ Aging seasoned gamers do not have enough time budget left to play Diablo, Warcraft, Heroes of Might and Magic...But they still want to play [Michaud, 2008]
  ♦ People play at work [Renou, 2008]
  ⇒ Akinator, Monlegionnaire, World of Goo...
  ♦ In 2008, casual games = 1/3 of gaming revenues (9.2 B€)

- The free to play revolution [Events, 2008]
  ♦ 2 B$ with games which are free to play (but you must buy contents)
  ⇒ Runescape (70 M$ revenues), Club Penguin (for 8–12 years old), Battlefield Heroes...
  ♦ Average Revenue Per User (ARPU)
    ▶ 1$ for free players (thanks to advertising)
    ▶ 15$ for paying users
    ▶ >100$ for big spenders
  ♦ How much does it cost you to get customer’s payment?
    ▶ PayPal: 5%
    ▶ Prepaid card: 20%
    ▶ Cell phone: 65%
  ♦ Web games will become mainstream (like TV became mainstream way ahead cinema)
Few figures about video game industry (3/4)

■ Distribution of skills in a development team [Capital Games et al., 2006]
  ♦ Programming: 41 %
  ♦ Animation: 28 %
  ♦ Game/Level Design: 11 %
  ♦ Project management: 8 %
  ♦ Administration: 12 %

■ In 2007, in France [Maman, 2007]
  ♦ 330 companies concerned
    ▶ 114 development studios
    ▶ 64 editors
    ▶ 28 middleware producers
    ▶ 23 distributors and hardware specialists
  ♦ 2,000–2,500 full-time jobs
  ♦ Around 8 employees per actor
  ♦ Since 2005, number of created start-up increased by 55%
Few figures about video game industry (4/4)

- What about multiplayer games? [SirBruce, 2007]
2.2 What makes multiplayer game such an interesting distributed application

- A multiplayer game is a distributed application, i.e. a set of autonomous machines which are communicating and make the game application evolve by exchanging messages [Natkin, 2002]

- Compared to classical distributed applications, multiplayer games have constraints which are:
  1. Classical: Consistency, deployment...Thus we reuse classical solutions
  2. Original: Reaction time, number of players (users), security. These require specific solutions
     Note: These problems are similar to the problems observed in distributed simulation applications and collaborative applications (e.g. e-learning)

- Mobility of terminals introduces additional constraints
2.3 Different types of games

- Classification based on [Saha et al., 2002, Nokia, 2003a]

- First-Person Shooter (FPS)
  
  Player has a gun and must fulfill a mission (either alone or within a team)
  
  Examples: Quake, Counter Strike...

- Simulation
  
  Player controls a team (football, hockey) or pilots an engine
  
  Examples: FIFA, Collin Mac Rae...

- Massive Multiplayer On-line Role Playing Game (MMORPG)
  
  Player controls an avatar in a virtual world
  
  Examples: World Of Warcraft (WoW), Dark Age of Camelot, Lineage (180,000 simultaneous players)...
Different types of games (contd)

- **Real-Time Strategy (RTS)**
  Player controls an army dealing with enemies
  Examples: Age of Empire, Command and Conquer...

- **Tour-based games**
  ♦ Each player plays one after the other (chess, poker, Heroes...)
  ♦ Player gives its orders to a server during a game turn. Then the server solves all of the movements
2.4 Two levels of functionalities in multiplayer games

One must provide 2 levels of functionalities in multiplayer games

♦ Customer management level
  ▶ Managing and animating the community of players
  ▶ Access portal (e.g.: http://www.goa.com)
  ▶ Forums
  ▶ High score storage
  ▶ Game downloads (e.g.: http://www.metaboli.fr or http://www.gallery.fr)
    NB: Eventually all games will be downloaded (e.g. TMNations 500 MB, Facade 800 MB...)

♦ Middleware level
  ▶ Communication and synchronization between the different modules of the game

Although “the game is 2% of the effort, the rest is customer management...” [Events, 2008], this presentation concentrates on middleware level after describing the problems brought by games to this level
3 Main constraints brought by video games

3.1 Building a consistent state ....................................................15
3.2 Performances ................................................................ 16
3.3 Scalability ................................................................... 22
3.4 Fault tolerance ...............................................................23
3.5 Security ..................................................................... 24
3.1 Building a consistent state

- At the same time, two computers may have different view of the current state because [Natkin, 2002]
  - Message propagation time is not bounded
    - Example: Car race [Pantel and Wolf, 2002a]
  - Messages can be lost
    - Example: Why a dead man may be able to shoot [Mauve, 2000]

- The virtual universe has its own measure of time. Thus, even though current date/time can be different on the different copies of the game, these copies must handle the commands at the same virtual date/time.
3.2 Performances

3.2.1 Bandwidth usage ........................................................... 17
3.2.2 CPU required to handle messages ........................................... 18
3.2.3 Taking into account latency and jitter ....................................... 19
3.2.4 Fairness requirement ........................................................ 21
3.2.1 Bandwidth usage

- Corresponds to the quantity of used network [Smed et al., 2001, Natkin, 2002]
- Seems not to be a problem anymore in fixed network (LAN, ADSL) although... There are still some 56 kbps modems
- May be a constraints in mobile networks
  Example: I-Mode specifications require application size to be lower than 30 Kb (in order to limit download time). Note: This limit is a *marketing* limit
3.2.2 CPU required to handle messages

[Bauer et al., 2002] reminds that it is not the used bandwidth which is the major constraints with network messages, but:

- Used CPU cycles
- Memory bandwidth
- Volume of input-output

used to handle the arrival of each message

- A farm cannot handle more than 100,000 HTTP requests per second [Bauer et al., 2002]
- Network traffic generated by a distributed application handling 100,000 objects can require up to 80% of the CPU of a 500 MHz processor [Smed et al., 2001]
- In Google platform, 15,000 PC handle... 2,000 requests per second
3.2.3 Taking into account latency and jitter

- Latency = Response time between a command and the materialization of its effect on all computers of players [Natkin, 2002]
- Jitter = Variance of latency during time [Smed et al., 2001]
- Latency (and jitter) are inherent to networks:
  - Internet
    - Atlantic crossing ≤ 80 ms (theoretical limit > 25 ms) [Smed et al., 2001]
    - Most important German ISP: 342 ms [Pantel and Wolf, 2002a]
    - Depends upon the number of crossed routers/providers [Bauer et al., 2002]
  - Wireless Area Networks (WAN)

<table>
<thead>
<tr>
<th></th>
<th>Throughput (in kbps)</th>
<th>Latency (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPRS (2G)</td>
<td>~ 40</td>
<td>≫ 500</td>
</tr>
<tr>
<td>EDGE release 4 (2,5G)</td>
<td>100 – 200</td>
<td>&lt; 500</td>
</tr>
<tr>
<td>UMTS (3G)</td>
<td>384</td>
<td>300</td>
</tr>
<tr>
<td>HSDPA (3G+)</td>
<td>1000 – 3000</td>
<td>100 – 300</td>
</tr>
</tbody>
</table>
Taking into account latency and jitter (contd)

- Constraints on latency/jitter depends upon the game
  - FPS: latency < 100 ms [Smed et al., 2001]
  - Car race: latency may go up to 150 ms (cockpit view) [Pantel and Wolf, 2002a]
  - RTS: latency may go up to 500 ms as long as jitter remains low [Bettner and Terrano, 2001]
  - To summarize for a mobile environment [Nokia, 2004]

See [Claypool and Claypool, 2006] for a more recent study

“A game can be considered playable if its users find its performance acceptable in terms of the perceptual effect of its inevitable inconsistencies” [Brun et al., 2006]
3.2.4 Fairness requirement

- Some games based on bids, bets or quiz (e.g. *Questions for a champion* where the player must press his buzz before his competitors) require guarantees concerning fairness of latency despite
  - Differences in performance of the terminals
  - Differences in performance concerning access to the network
- This requirement does not influence the architecture. It requires another software layer to introduce delivery delays
- This is still a research subject
  - Fair message exchange framework [Guo et al., 2003]
  - It is always possible to increase the level of inconsistency in a game by artificially delaying information. This technique enables the equalization of inconsistencies among players, effectively improving game fairness at the cost of overall playability. [Brun et al., 2006]
3.3 Scalability

- Game architecture must be able to [Natkin, 2002]
  - Reconfigure dynamically itself according to the number of players
  - Accept an important number of players
3.4 Fault tolerance

- Keep the service available despite failure of some of the elements [Natkin, 2002]
3.5 Security

- Classification according to [Natkin, 2002]
- Players authentication
- Protection in terms of confidentiality and integrity
- Protection of author rights
- Respect of game rules [Smed et al., 2001]
  - Motivated by a vandalism or domination spirit, some players cheat
  - Protections must be put in place to make cheating as difficult as possible
  - Cheating techniques:
    - Packet and traffic tempering
    - Information exposure
    - Design defects
    - Collusion
- In the remainder, we will focus on this last aspect (specific to games)
4 Peer-to-peer architecture

4.1 Overview ................................................................. 26
4.2 Evaluation ................................................................. 27
4.3 Using total order algorithms ........................................ 30
4.4 Interest management .................................................... 31
4.5 Interest management and Content-based publish-subscribe 33
4.6 Dead reckoning .......................................................... 34
4.7 Consistency algorithms ................................................ 35
4.1 Overview

- Principles [Natkin, 2002]
  - Each terminal sends to all other terminals all of the commands executed by the player
  - Each terminal has a full copy of the game. It computes the current state of an on-going game thanks to the received messages

- Examples
  - Age of Empire [Bettner and Terrano, 2001]
  - Six in the City [Mitchell et al., 2003]

- Comments
  - Architecture well-adapted to local networks because they allow to quickly broadcast a message
  - But it is also used in the case of terminals connected through Internet (mostly for games allowing a high latency)
  - It is used by a lot of games because it is a natural extension of the monoplayer version of a game (except for the problem of random events [Bettner and Terrano, 2001])
4.2 Evaluation

- **Building a consistent state**
  
  There is no central reference: A synchronization mechanism must be put in place between the different replicas
  
  ⇒  ♦ “Consistency algorithms” (*Algorithmes de cohérence*)
  ♦ “Total order algorithms”

- **Bandwidth usage**
  
  ♦ Requirements are low if number of players is low [Pantel and Wolf, 2002a]
    
    Example: MiMaze game broadcasts 52 bytes per player with a frequency of 25 Hz ⇒ $52 \times 8 \times 25 \approx 10$ kbps [Gautier and Diot, 1998]
  
  ♦ If bandwidth is too low (e.g. Internet access through modem), the game must limit the number of messages sent
  
  ⇒  ♦ “Interest management” (*Filtrage sémantique*) possibly coupled with
    “Content-based publish-subscribe”
  ♦ “Dead reckoning” (*Prévision d’état*)
Evaluation (2/3)

- CPU required to handle messages

  Game must limit number of messages sent (so that there is still CPU to handle processing related to the game)

  ⇒ Same solutions as Bandwidth usage

- Latency and jitter

  ⇒ ♦ “Dead reckoning” (*Prévision d’état*)
    ♦ “Consistency algorithms” (*Algorithmes de cohérence*)

- Scalability

  ♦ Limited by CPU required by the least powerful machine to handle messages
    [Bettner and Terrano, 2001]

  ♦ Thus at most few dozens of players for FPS

- Fault-tolerance: OK
Security

- Packet and traffic tempering cheating techniques: KO
- Information exposure
  - A terminal can multicast its messages only to terminals concerned by the changes in its instance of the game
  ⇒ Content-based publish-subscribe algorithms
- Design defects: KO
4.3 Using total order algorithms

- Problem: Prevent consistency problems between the different game replicas
- Solution: Use a total order algorithm
  - Any broadcast message is received in the same order by all of the replicas
  - Example: Arrow multicast [Herlihy and Mohan, 2003]
4.4 Interest management

- Problem: Limit the number of messages sent by the different terminals
- Solution overview [Smed et al., 2001]
  - It is not necessary to send to each player a copy of all of the actions of other players
  - You only need to consider the part of the universe the player can observe and which it can influence
  - Game engine can determine the set of objects perceptible by player’s avatar (via seeing, hearing or touching) = Focus
  - Game engine can also determine the space within which an object can be perceived = Nimbus
  - Messages must be exchanged between players A and B if and only if there is an intersection between one’s focus and the other’s nimbus
Interest management (contd)

- Example of a hide-and-seek game
4.5 Interest management and Content-based publish-subscribe

- Interest management can be coupled with *Content-based publish-subscribe*
- *Content-based publish-subscribe*: What's that?
- Example of Mercury [Bharambe et al., 2002]
  - Have a subscription language as flexible as *Gryphon* or *Siena*
  - Be as scalable as *Herald* or *Scribe*

Problems close to the ones observed in “Semantic P2P”
4.6 Dead reckoning

- Problem: Limit the number of messages sent by the different terminals
- Solution overview [Pantel and Wolf, 2002b]
  - A player broadcasts the position and speed vector of its avatar
  - It broadcasts again these information only in the case they have changed significantly
  - Upon receiving this message, the other players take into account this new position for the avatar copy they hold and memorize the speed vector
  - From then on, they move, by themselves, the avatar copy using the speed vector
4.7 Consistency algorithms

- Problem: Dead reckoning may lead to inconsistencies
- Please refer to [Chabridon and Kodrnja, 2005, Khan et al., 2007] for overview of solutions
## 5 Client-server architecture

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Overview</td>
<td>37</td>
</tr>
<tr>
<td>5.2</td>
<td>Evaluation</td>
<td>38</td>
</tr>
<tr>
<td>5.3</td>
<td>Use farms</td>
<td>42</td>
</tr>
<tr>
<td>5.4</td>
<td>Add intelligence to the network</td>
<td>43</td>
</tr>
<tr>
<td>5.5</td>
<td>Dynamic multicast</td>
<td>44</td>
</tr>
<tr>
<td>5.6</td>
<td>Example of an architecture: EVE Online</td>
<td>45</td>
</tr>
</tbody>
</table>
5.1 Overview

Principles [Natkin, 2002]
♦ Game is running on a central site (the game server) which receives all of the commands of all of the terminals of the players (the clients)
♦ Server does all of the computation (except aspects related to graphics and sounds)
♦ It returns the result to all of the terminals

Examples
♦ Quake
♦ Dark Age of Camelot
♦ Samuraï Romanesque [Potel, 2003]
♦ TibiaME [Nokia, 2003b]

Comments
♦ Architecture well-adapted to Internet network where there is no broadcast capability
♦ Natural architecture in the case of games based on a virtual world
5.2 Evaluation

- Building a consistent state: As the server is the reference, one could think there is no consistency problems. But there are...

- Bandwidth usage
  - ♦ Low on client side
  - ♦ Can reach saturation on server side (because of CPU)
Evaluation (2/4)

- CPU required to handle messages
  - Server = Bottleneck
  - Example: Quake

→ ♦ Interest management
   ♦ Dead reckoning
   ♦ Use computer farms (example: Quake on a grid)
   ♦ Add intelligence to the network
   ♦ Dynamic multicast

- Latency and jitter
  Additional latency can be introduced because of the number of messages handled by the server (see previous point)
Evaluation (3/4)

■ Scalability
  Server = Bottleneck (example, in TibiaME MMORPG, 1,000 players per server)
  ⇒ ♦ Increase number of servers, each one handling its own instance of the virtual world (and each player being affected to a given server, once for all)
  ♦ Share the management of the virtual world between different machines (each player migrating from one machine to another according to his position in the virtual world)
  ♦ Use “server network” architecture

■ Fault-tolerance
  Server = Bottleneck
  ⇒ ♦ Use computer farms
  ♦ Use fault-tolerant servers
  ♦ Use “server network” architecture
Security

♦ Packet and traffic tempering cheating techniques
  Installation, on the server side, of code checking consistency of information sent by the client

♦ Information exposure
  Server sends only relevant information to the different clients (see last versions of Quake)

♦ Design defects
  Installation, on the server side, of code checking consistency of information sent by the client
  Example: “Runtime verification” (e.g. evaluate how fast a player earns money [DeLap et al., 2004])
5.3 Use farms

- Problems:
  - On the server side, cope with CPU requirements
    - Example: An MMORPG with 100,000 players would lead to a traffic of 200 transactions per second on a database
  - Do it dynamically so that unused resources can be shared with other games

- Solution: Use a farm

- Example of product (project?) Butterfly [Butterfly.net and IDC, 2003]
  - Cluster of 50 IBM eServer
  - Interconnection with high speed optical links
  - Cluster management with Globus
  - A player is transparently affected to one of the eServers
  - DB2 database with mechanism for migrating data concerning a player from one eServer to the other
5.4 Add intelligence to the network

- Problem: Reduce traffic handled by the server
- Solution: Add *booster boxes* to do that as soon as possible [Bauer et al., 2002]
  - Cache management: *Booster box* caches non real-time information and answer for the server
  - Aggregation: Events of 2 (or more) clients are aggregated in a single message
  - Intelligent filtering: depending on the game state, some events may be no more significant. *Booster box* filters them
  - Intelligent routing: If server is distributed on several machines, *booster box* forwards a message to the concerned machine

*Booster box* is a configurable router adapted to some games.
5.5 Dynamic multicast

- Problem: Reduce traffic handled by the server
- Solution = Dynamic multicast (*Arbre de multicast*) [Ramakrishna et al., 2003]
  - All of the nodes are linked together according to a tree structure
  - Each node is either parent for other nodes (its children) or a leaf
  - Server is the root of the tree
  - Each node sends its messages to its parent. This parent aggregates messages from its different children before forwarding them to its own parent
  - Messages issued from the server follow the reverse path

- Experience on DOOM with 8 players

<table>
<thead>
<tr>
<th></th>
<th>P2P</th>
<th>client-server</th>
<th>Dynamic multicast</th>
</tr>
</thead>
<tbody>
<tr>
<td># packets sent out by players</td>
<td>56</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td># packets in the network</td>
<td>207</td>
<td>40</td>
<td>26</td>
</tr>
</tbody>
</table>
5.6 Example of an architecture: EVE Online

- 400 GHz CPU / 200 GB RAM [Brandt, 2005]
- 2 Routers (CISCO Alteon)
- 14 Proxy servers (IBM Blade)
- 55 Sol servers (IBM x335)
- 2 DB servers (clustered, IBM Brick x445)
- FastT600 Fiber, 56 x FC 15k disks, DS4300 + 3*EXP700
- Windows 2000, MS SQL Server
6 Server-network architecture

6.1 Overview ................................................................. 47
6.2 Evaluation .................................................................. 48
6.3 Mirrored server ......................................................... 50
6.4 Zone server ............................................................... 51
6.5 Generic Proxy System ............................................... 52
6.1 Overview

- **Principles [Smed et al., 2001]**
  - In this architecture, several servers are interconnected
  - A client is connected to a given server. This server is communicating with all other servers and thus all of their clients

- **Examples (research field)**
  - Quake with mirrored servers [Cronin et al., 2002]
  - Zone-server [Riera et al., 2003]
  - Generic proxy system [Mauve et al., 2002]

- **Comments**
  - Servers work according to a P2P architecture
  - A client works with its server according to a client-server architecture
6.2 Evaluation

- Building a consistent state
  - See client-server for client-server part
  - See P2P for P2P part
- Bandwidth usage
  - On client side: Limited
  - On servers side: Distributed
- CPU required to handle messages
  Load distributed among servers
- Latency and jitter
  - Better than client-server (client is nearer its server)
- Scalability
  Load distributed among servers
Evaluation (contd)

- Fault-tolerance
  - ♦ If one server stops, game is over for its clients (papers propose solutions to migrate this clients)
  - ♦ But game goes on for majority of players

- Security
  Better than client-server (The servers are less loaded. Thus more controls can be put in place)
6.3 Mirrored server

- In this architecture [Cronin et al., 2002], servers are interconnected with a private high-speed network dedicated to data exchange between them.
- Inconsistencies are handled with TSS algorithm.
6.4 Zone server

- This architecture is dedicated to mobile terminals [Riera et al., 2003]
- Several terminals have the role of *zone server* (there must be at least 2 *zone servers* for fault tolerance)
  - They serve terminals which are plain clients
  - They synchronize with other *zone servers* in case of connectivity loss with a client
  - A *Zone server* receives data from its own client and transmits them to other *Zone Server* (There is no consistency mechanism, but authors believe TSS algorithm could be a good answer)
- Future work = Selection of the different *Zone servers*
6.5 Generic Proxy System

- In this architecture [Mauve et al., 2002], a *central server* is responsible for keeping the game state.
- It is working with several *proxy* to whom it delegates some of its functions.
- A client connects to the server or to one proxy.
- When a proxy receives a message from a client, it forwards it to the server and the other proxies.
- This architecture improves:
  - Congestion control
  - Robustness
  - Latency
  - Equity
  - Security (as more controls can be done)
- Research subject: Adapt this architecture to an environment with mobile terminals.
7 Mobile games

7.1 Few figures about mobile game industry (1/3) ........................................... 54
7.2 Key issues ........................................................................................................ 60
7.3 Example of a mobile multiplayer game .............................................................. 61
7.4 Architectures for mobile games ...................................................................... 64
7.1 Few figures about mobile game industry (1/3)

- Console game = 50 $ versus Mobile game = 5-7 $
- Development: Console game in M$ versus Mobile game = 100-400 k$
- Huge number of potential customers
  - Game consoles: Gameboy, Nintendo DS (On 31/12/2007, 68m units since end 2004), N-Gage, PSP (on 31/12/2007, 25m units sold since end 2005)...
  - PDA (20m units sold in 2007)
  - Mobile phones (1.100m units sold in 2007)
- Prediction that mobile games could generate 11.2 billion $ by 2010 [Patterson and Dredge, 2006]

<table>
<thead>
<tr>
<th></th>
<th>2007</th>
<th>2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia</td>
<td>1,8 B$</td>
<td>4,6 B$</td>
</tr>
<tr>
<td>Europe</td>
<td>1,5 B$</td>
<td>2,0 B$</td>
</tr>
<tr>
<td>USA</td>
<td>0,7 B$</td>
<td>1,7 B$</td>
</tr>
<tr>
<td>Total</td>
<td>4,3 B$</td>
<td>10,0 B$</td>
</tr>
</tbody>
</table>

- Other study [Clech, 2007]: +50% in 2006 with:

- April 2005: Jamdat bought Tetris rights for 15 years for 137 million $ (3Q04, Tetris = 4,1 million $ revenues) [Patterson and Dredge, 2006]
- “Times will be hard for small mobile game developers in the near future” [Koivisto, 2007]
Few figures about mobile game industry (2/6)

- In France, in 2006 [Séfrin, 2007]:
  - 10 Million downloads of games
  - 3–5 € per game
  - About 300 games offered by 50 editors
  - 46% of gamers are female
  - Contents market = 500–700 M€ (to be compared to 20 B€ of voice market...)
  - 51,7 Million customers
  - 500,000 Orange customers watch mobile TV (43 minutes/month/subscriber)
  - Orange launched http://www.ticket-game.com

- [Agence Française pour le Jeu Vidéo (AFJV), 2008a] synthetizes French 2007 sales for mobile contents:

<table>
<thead>
<tr>
<th></th>
<th>Units (in Million)</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Games</td>
<td>13,3</td>
<td>56,7 M€</td>
</tr>
<tr>
<td>Videos</td>
<td>6,0</td>
<td>15,6 M€</td>
</tr>
<tr>
<td>Images</td>
<td>20,0</td>
<td>56,0 M€</td>
</tr>
<tr>
<td>Ringtones</td>
<td>22,0</td>
<td>54,4 M€</td>
</tr>
<tr>
<td>Full Track Music</td>
<td>15,9</td>
<td>19,0 M€</td>
</tr>
</tbody>
</table>
Few figures about mobile game industry (3/6)

- Mobile game sales could be a lot better [Morrisson, 2008]
Few figures about mobile game industry (4/6)

- Top 10 2007 best mobile games sales (in France)
  [Agence Française pour le Jeu Vidéo (AFJV), 2008a]

<table>
<thead>
<tr>
<th>Rank</th>
<th>Title</th>
<th>Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tetris</td>
<td>Electronic Arts</td>
</tr>
<tr>
<td>2</td>
<td>Block Breaker Deluxe</td>
<td>Gameloft</td>
</tr>
<tr>
<td>3</td>
<td>The Sims 2</td>
<td>Electronic Arts</td>
</tr>
<tr>
<td>4</td>
<td>Brain Challenge</td>
<td>Gameloft</td>
</tr>
<tr>
<td>5</td>
<td>Worms Forts</td>
<td>THQ Wireless</td>
</tr>
<tr>
<td>6</td>
<td>Deal or not deal</td>
<td>Gameloft</td>
</tr>
<tr>
<td>7</td>
<td>New York nights: Success in the city</td>
<td>Gameloft</td>
</tr>
<tr>
<td>8</td>
<td>Desperate Housewives</td>
<td>Gameloft</td>
</tr>
<tr>
<td>9</td>
<td>Pac-Man</td>
<td>Namco</td>
</tr>
<tr>
<td>10</td>
<td>The Simpsons: Minutes to meltdown</td>
<td>Electronic Arts</td>
</tr>
</tbody>
</table>

- Mobile is very well adapted to casual gaming [Michaud, 2008]
Few figures about mobile game industry (5/6)

- [Koivisto, 2007] shows that:
  - 60% gamers play at home
  - 60% gamers play more than once a week
  - Average play session = 15–28 minutes
  - 45% of mobile game players play multiplayer mobile games at least once per month
Few figures about video game industry (6/6)

- Mobile game value chain [IGDA Online Games SIG, 2005]
7.2 Key issues

- Unlimited data plan [Michaud, 2008]
- Make download really easy
- Operator is too strong
  - Low quality of games
  - Little development of multiplayer games

- Portability
  - Around 700 different mobiles available in the world
  - For *Prince of Persia*, Gameloft made 100 different porting
  - Gameloft has 2,000 people. 80-90% are working on porting
  
  Solution = MIDP3? Brew? Wipi? Doja?

- Other aspects (FYI)
  - Limited capacities of virtual machines (e.g. threads)
  - Limited energy resources [Mascolo et al., 2002, Nokia, 2007]
  - Limited screen size [Nokia, 2003a], but display could be on a big screen [Rodriguez et al., 2008]
  - Screen size varies widely [Palm, 2003]
  - Colors and sounds are limited [Nokia, 2003a]
  - Input interface is limited [Nokia, 2004], but dedicated devices like Zeemote [Zeemote Inc, 2009], touch screens [Agence Française pour le Jeu Vidéo (AFJV), 2008b] and accelerometers may reduce this limit
  - Deployment [Rawat and Simatic, 2007]
7.3 Example of a mobile multiplayer game

- TibiaME (Q2 2003) [Nokia, 2003b]

  - German adaptation to mobile phone of PC MMORPG
  - Main challenges: Size (150 KB to download and 390 KB RAM), Latency (up to 4 seconds) and User Interface
  - Server side: 90% of PC version code (C-C++/Linux) reused (it was already based on 2D sprites). Uses a PostgreSQL DB
  - Client-side: Developed from scratch for C++/Symbian OS client-side
  - Development = Team of 5 people in 6 months.
  - 1 server handles 1,000 players
Example of a mobile multiplayer game (2/3)

- Business model (in 2003) [Nokia, 2003b]
  - No cost to acquire the client application and no monthly subscription fee
  - Users pay for the data traffic they create over (T-Mobile’s) GPRS network.
    Typical player generates traffic 400 KB/hour ⇒ costs 0.72-3.7 € depending on deal with network provider.
  - Game provider has a three-level hierarchy of >100 volunteer game masters and game counselors, allowing continuous online support.

- In 2008 [CipSoft GmbH, 2008]
  - Premium access subscription costs 9,99 € for 4 months
  - Episodes are available (30–60 minutes playing time)
Example of a mobile multiplayer game (3/3)

- More information on 2008 version [Zuckerer, 2008]
  - About 3,500 active Premium subscribers and about 20,000 active players altogether.
  - About 18 simultaneous gameworlds, each capable of having 250 players online at the same time.
  - Can go up to over 1000 players per server (Xeon systems [2 CPUs with 2,0 Ghz] with 2 GB RAM). The reason for the low amount is that the game feels crowded on the mobile phone with this amount of players. The screen size is small and a few players just fill up a city very quickly. CipSoft is constantly adding new content to the game to relax the situation and will eventually lift up the player capacity.
  - On 24h average, 1,850 simultaneous players over the 18 worlds.
- Statistics on countries from where people are playing TibiaME
  - Most of players come from Brunei, Russia, Poland and Indonesia. The strong regions on the market are in Far-East Asia and East Europe. Countries with cheap mobile internet access usually have a high interest in playing the game.
- Most of players play more than 4 hours in average per day.
- Development team
  - “As we also work on our main product Tibia (PC-Game) some people work on both teams. The dedicated TibiaME team has got 5 people working on it. At the moment we are pushing new features and the development of the game in general. We have got two big updates every year, adding new content and new features to the game. Between the updates we publish small content episodes for the players.”
7.4 Architectures for mobile games

- **Peer-to-peer**
  - Well adapted to consoles and PDA (or even mobiles on WLAN), because limited number of terminals (typically 8) and oad sharing
  - Not realistic for mobiles on WAN (client/server communications are mandatory!)

- **Client-server**
  - No load sharing as with Peer-to-peer
  - Natural architecture with Bluetooth networks and mobiles on WAN

- **Server-network**
  - Sounds interesting!
  - Imagine to have several mobiles connected with Bluetooth, one of these mobiles, the *zone server*, being connected to the server via WAN
8 Pervasive and ubiquitous games

8.1 Definitions ................................................................. 66
8.2 Product examples .......................................................... 67
8.3 Research examples ......................................................... 68
8.4 Zoom on “Plug: Secrets of the Museum” (PSM) .................... 72
8.5 Business and revenue issues ............................................. 76
8.6 Specific technical issues .................................................... 77
8.1 Definitions

- Pervasive game = Game enriched with computing and communication technologies in order to combine transparently physical and numerical spaces of the player [Schneider and Kortuem, 2001]
- Ubiquitous = Game which can be played anywhere [Schneider and Kortuem, 2001]
- ARG = Alternate Reality Games: Games where border between reality and game universe is blurred [Le Bourlout, 2007]
8.2 Product examples

- In Memoriam - Le Dernier Rituel [Ubisoft, 2006]

- ViaTemporis [Scrinéo, 2005]
  - 29 €

- REXplorer [Walz et al., 2006]
  - 10,000 - 15,000 players per year
  - Hardware renting for 1.5 hours
8.3 Research examples

8.3.1 Research examples in limited physical space ........................................ 69
8.3.2 Research examples in unlimited space ............................................. 70
8.3.3 Research examples with augmented reality ........................................ 71
8.3.1 Research examples in limited physical space

- “The Drop”: A spy game taking place in a commercial center [Smith et al., 2005]
- “Save the Princess!”: A D&D game taking place in an office (actually a demo game for TinyLime) [Mottola et al., 2006]
- “Pac-Lan”: An arcade game taking place in a neighborhood [Rashid et al., 2006]
- “Plug: Secrets of the Museum”: A virtual card game taking place in a museum (see zoom in slide 8.4).
8.3.2 Research examples in unlimited space

- BotFighters [PriXArs, 2005]

- Mogi [Licoppe and Inada, 2005]

- Conqwest [Qwest Communications International Inc., 2004]
8.3.3 Research examples with augmented reality

- Human Pacman [Cheok et al., 2003]

- ARQuake/Tinmith [UniSA, 2002]

- 3D Sky Vaders (A_RAGE: Augmented Reality Active Game Engine) [A_RAGE Pty Ltd, 2005]
8.4 Zoom on “Plug: Secrets of the Museum” (PSM)

8.4.1 Presentation of PSM ....................................................... 73
8.4.2 Technical challenges ........................................................74
8.4.3 Pre- and Post-experimental remarks .................................75
8.4.1 Presentation of PSM

- Pre-product demo of PLUG project [Cédric et al., 2009]
- Designed to let players discover, in a different way, some of the objets of a museum [Astic et al., 2009]
- Players are equipped with NFC-enabled mobile phones
- Museum is equipped with RFID tags located besides key objects of the museum
- Goal = Collect families of virtual cards associated to these key objects
- To do so, players exchange virtual cards either with the RFID tags or with other players (through NFC peer-to-peer communication).
- PSM includes a hint function: It gives a player an indication of an RFID tag which contains a virtual card interesting for his collection.
- PSM is currently implemented for 8 players and 16 tags in the context of the musée des Arts et Métiers
- A PSM game session lasts 55 minutes
- Measures made during experiments showed that players take an average time of 90 seconds to move from one tag to another
8.4.2 Technical challenges

- Communications between mobiles are only local \( \Rightarrow \) There cannot be a central server taking care of the game data

\( \Rightarrow \) 5 technical challenges [Simatic, 2009]
1. At the beginning of a game session, all of the mobile phones must be initialized with the virtual cards they are hosting
2. Each RFID tag must be initialized with the virtual card it is hosting
3. The hint function must be implemented
4. All playing mobiles must be notified of the end of a game session at the same time
5. The game must tolerate faults concerning the game application on a mobile, a mobile itself, and faults related to RFID tags
8.4.3 Pre- and Post-experimental remarks

- Importance of in-game communications
- Pedagogy versus Fun
- Reaching consensus in the definition of player’s main goal
- Environment design is capital
- Have (qualified) usage people in your project to get correct feedback

Next sessions

- Futur(s) en Seine: 29/05/09 – 07/06/09
  - 1 PI20 will contribute by developing a synthesis application
- PSM version 2: November 09
  - With iPhones/iPod touch or Samsung Player Addict & Wiimote & biometric sensors
  - 1 PI20 is contributing to the game design
8.5 Business and revenue issues

- Elements from [Söderlund et al., 2005]
  - Market structure: Towards pervasive advergaming
  - Revenue model: Dominant model should be pay-per-use (games as services rather than as products)
  - Production process: Game release will be followed by numerous new releases; This development forces game development companies to radically rethink their production strategy.
  - Marketing: What producer(s) of pervasive games want future consumers to define as “pervasive games”?
8.6 Specific technical issues

- Context management in ubiquitous environment
  - See Denis Conan’s presentation

- Development tools
  - mscape [mscape, 2007]
    - Single player
    - Only location-based (with GPS) games/applications
    - Programming of “mediascapes”: A mediascape is a collection of media fragments associated with positions in space. You experience the media fragments as you walk around the space
  - Open-MEE [Biswas et al., 2006]

- Tools of robotics community
  - Microsoft Robotics Studio [Microsoft, 2008]
  - Lego mindstorms NXT software [LEGO, 2008]

- Issue: Monolithic aspect of current Multiplayer Mobile Game engines [Rawat and Simatic, 2007]
9 Game communication middlewares

9.1 Few examples of middleware ................................................................. 79
9.2 Core middlewares used by game communication middlewares ............. 81
9.3 Zoom on Darkstar ............................................................................. 82
9.4 Zoom on GASP ................................................................................. 83
9.1 Few examples of middleware

- **Open source**
  - Darkstar (see zoom in slide 9.3)
  - GASP (CNAM-CEDRIC, Filao, INT) (see zoom in slide 9.4)
  - MUPE [Suomela et al., 2005]
  - Massiv [Massiv, 2005]
  - Open NeL (Nevrax) [NeL community, 2009]
  - ICE (ZeroC) [Henning, 2004]
  - OpenTNL (Torque) [OpenTNL, 2007]
  - Grizzly (Sun) [Sun Microsystems, 2009]

- **Commercial products**
  - Neutron (Exit Games) [Exit Games, 2009]
  - Terraplay (Terraplay) [Terraplay, 2003, Terraplay, 2002b]
  - NetZ/Eterna (Quazal) [Quazal, 2009]
  - SNAP Mobile (Nokia) [Nokia, 2009]
Few examples of middleware (contd)

○ R&D products
  ♦ Continuum (FT R&D) [Dang Tran et al., 2003, Dang Tran et al., 2002]
  ♦ Skype4Games (University of Mannheim) [Triebel et al., 2007]
  ♦ Real-Time Framework/RTF (University of Münster in the context of edutain@grid STREP) [Ploss et al., 2007]
9.2 Core middlewares used by game communication middlewares

- **CORBA**
  - Continuum is using Jonathan (ObjectWeb) [Dang Tran et al., 2003]

- **Message-Oriented Middleware (MOM/Publish-subscribe)**
  - Terazona uses a JMS-compliant MOM (C or Java interface) [Zona, 2002]
  - Terraplay offers a MOM with reliable (or non-reliable) delivery, towards a machine or a group, with dynamic subscription capabilities [Terraplay, 2002b, Terraplay, 2002a]
### 9.3 Zoom on Darkstar

- **Open Source Project** [Waldo, 2008, Sun Microsystems, 2008]
- **For games based on a client-server architecture**
- **Philosophy:** Make server-side game code reliable, scalable, persistent, and fault-tolerant in a manner that is transparent to the game developer.
- **In Darkstar, clients can send messages** [Sun Microsystems, 2007a, Sun Microsystems, 2007b]
  - Directly to the server: Once connected to the server, each client has a dedicated communication medium between server and itself
  - Through a **Communication channel**
    - A **Communication channel** is created by the server (which defines the name of this channel, as a string)
    - Server registers/unregisters clients, either spontaneously or following a client request
    - Once a client is registered to a **Communication channel**, it receives any message sent to this **Communication channel** by itself or another client
- **Note:** The multi-node version is not in the current release (which contains only the single-node server)
9.4 Zoom on GASP

9.4.1 Java 2 Micro Edition (J2ME) ............................................................... 84
9.4.2 Typical modelization of a multiplayer game on J2ME .................................. 85
9.4.3 Standardization of multiplayer game platforms by OMA .................................. 86
9.4.4 GASP: Objectives .................................................................................. 87
9.4.5 GASP: Data model .................................................................................. 88
9.4.6 GASP: Architecture ................................................................................. 89
9.4.7 GASP: Event model .................................................................................. 90
9.4.8 GASP: MooDS protocol .......................................................................... 92
9.4.9 GASP: MooDS versus SOAP .................................................................... 94
9.4.10 GASP: status and perspectives .................................................................. 95
9.4.1 Java 2 Micro Edition (J2ME)
9.4.2 Typical modelization of a multiplayer game on J2ME

1. Thread Game logic execution
2. Thread Paint
3. Thread Keyboard event listener
4. Thread Communication with the platform (game server)
9.4.3 Standardization of multiplayer game platforms by OMA

- **Games Services** workgroup of *Open Mobile Alliance* (OMA) proposes a general architecture and functional specifications [Open Mobile Alliance, 2007]
9.4.4 GASP: Objectives

- Implementation, in Java, of OMA/GS specifications for games platforms
- Open Source
- Target
  - J2ME mobiles, MIDP and Doja profiles
  - Games with quick interactions (beyond turn by turn), despite http protocol over WAN
  - Small editors
- Take into account development constraints on mobile (code size, message size)
9.4.5 GASP: Data model

![Class diagram for GASP data model]

- **Platform**: 1
- **MasterApplicationInstance**: 0..*
- **ApplicationInstance**: 0..*
- **Actor**: 0..*
- **Application**: 1
- **Session**: 1
- **ActorSession**: 0..*
- **Rights**: 1
- **User**: 0..*

Legend:
- **Base de données**: Gray
- **Classes**: Blue
9.4.6 GASP: Architecture

MOBILE
- GASP Client
- Client Game Logic

OTA

GASP
- GASP DB

PLATFORM REPRESENTATION
- Platform
- Services
- DEManager
- IDManager
- MasterApplicationInstance
- ApplicationInstance
- ActorSession
- GAME Representation
- GAME SESSIONS

SERVLETs container
- FirstLogin
- Login
- Lobby
- Join
- Create
- InGame
- Name
- StartAI
- EndAI
- QuitAI
- Quit

GAME SERVER
- GASP Server
- Server Game Logic

extend
com/appel
association
9.4.7 GASP: Event model

- 5 event types:
  - JoinEvent ⇒ A new player joins the game
  - StartEvent ⇒ Game start
  - EndEvent ⇒ Game end
  - QuitEvent ⇒ A player quits a game
  - DataEvent ⇒ Data sent during game time by a player or server game logic

- 3 types of event listeners:
  - ApplicationInstance (Game sessions)
  - ActorSessions (The players)
  - GASPServer (Interface extended by server game logic)
GASP: Event model (contd)
9.4.8 GASP: MooDS protocol

- MooDS = Mobile Optimized Objects Description & Serialization
  - Goal: Improve communication speed by reducing size of messages exchanged in game
  - Algorithm: Transport game messages by value (primitive types)

```
<xsd:schema ... />
<xsd:complexType name="Root">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element ref="gmr:update"/>
      ...</xsd:choice>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Update">
  <xsd:attribute name="id" type="xsd:short"/>
  <xsd:attribute name="posX" type="xsd:int"/>
  <xsd:attribute name="posY" type="xsd:int"/>
  <xsd:attribute name="time" type="xsd:long"/>
</xsd:complexType>

...</xsd:schema>
```

```
MooDS Generator

MooDS = Mobile Optimized Objects Description & Serialization

- void encodeUpdate(DataOutputStream dos)
- Update decodeUpdate(DataInputStream dis)
+ void encodeData(Hashtable h, DataOutputStream dos)
+ Hashtable decodeData(DataInputStream dis)

gamePackage.CustomTypes

CustomTypes.java

`types.xsd`
GASP: MooDS protocol (contd)

```
Update
aSID:short
posX:int
posY:int
time:long

Hashtable
Update

Hashtable
Update
```

```
CustomTypes
encodeUpdate(dos,update)
void encodeData(dos,hashtable)

time posY posX aSID UType

Hashtable decodeData(dis)
```

```
CustomTypes
Update decodeUpdate(dis)
```
9.4.9 GASP: MooDS versus SOAP

See [Pellerin, 2007] for more information.
9.4.10 GASP: status and perspectives

- GASP is an Open Source Java platform for multiplayer games on mobile
- **Status**
  - Publication (ObjectWeb [Pellerin et al., 2007]) of GASP
  - Publication of BomberGASP as demonstrator of GASP
  - Publication of MooDS version 2.0
    - Accelerate multiplayer game development (code generation)
    - Management of several communication models: Unicast, multicast and broadcast
    - Data persistency service
  - uGasp
- **Perspectives**
  - Cross-Platform: C# management (Interface MIDP, Doja, Flash and Windows Mobile client in the context of the same game)
  - Online/Bluetooth GASP: Development of a link between online and Bluetooth versions (several groups of users will be able to play the same game)
  - GASP for pervasive games: uGasp at ICPS & uGasp in “Plug: Secrets of the Museum” version 2
10 Conclusion

- A multiplayer game is a distributed application with specific constraints: consistency, reaction time, number of players (users), security...
- Use of mobile terminals amplifies these constraints or adds some new ones (e.g., latency)
- Classical architectures are able to answer to these constraints
- But they must be completed with solutions (possibly specific to games): interest management, dead reckoning, consistency...
- Multiplayer mobile games business will explode as soon as:
  - The costs of playing become low ???
  - Pervasive and ubiquitous games are more developed
  - Flash is more and more available on mobile (Why not Runescape or MapleStory on mobile? [Gamasutra, 2007])
Références


Multiplayer computer games, mobile phones and beyond


Multiplayer computer games, mobile phones and beyond


Multiplayer computer games, mobile phones and beyond


Multiplayer computer games, mobile phones and beyond


Multiplayer computer games, mobile phones and beyond


self-optimizing multiplayer gaming architecture. In Autonomic computing workshop, the fifth annual international workshop on Active Middleware Services (AMS 2003).


Multiplayer computer games, mobile phones and beyond


Multiplayer computer games, mobile phones and beyond


