A Virtual World Distributed Server
developed in Erlang as a Tool for analysing Needs of Massively Multiplayer Online Game Servers

Erlang/OTP User Conference
Stockholm on November 10, 2005

Michał Ślaski
michal@erlang-consulting.com
Contents

• Massively Multiplayer Online Games
• Problems to solve
• Prototyping
• Using Erlang/OTP for the prototype
• Client and server applications
• Determining the maximum number of players
• Architectures with single and double-function nodes
• Synchronisation of the game’s state
• Conclusions
Massively Multiplayer Online Games

- Significant increasing in number of active subscribers
- Fast growing online games market

Source: MMOCHART.COM

Source: IGDA, "2004 Persistent Worlds White Paper"
Problems to solve

- Big amount of data to process

- Frequent synchronisation
  - acceptable delay 150 - 500 ms

- Adaptation to the connection quality

- Limited opportunity for full testing with thousands of players during early phase of development
Prototyping

- Experimenting with the functional and technical aspects
- Determining the scalability of the architecture
- Testing new solutions
Using Erlang/OTP for the prototype

- Open source
- Compatible with various platforms
- Concurrent oriented programming
  - one process for every connected player
- Efficient internal communication
- Supporting distributed applications
- Distributed database Mnesia
- Distributed load testing tool IDX-Tsunami
Client application

- **Java + Java3D**
- **Actions:**
  - moving
  - collecting objects
  - magic
  - chat
- **Player’s state kept on the server**
Server application

- The game terrain divided into zones
- Players placed within the same zone are processed on the same node

- Server distributed on a cluster of machines

![Diagram showing server application with zones and nodes processing user actions and network traffic.](image)
Determining the maximum number of users

The server was overloaded in the 60th second of the test.
Architectures with double-function nodes

The graph illustrates the relationship between the number of nodes processing network traffic and player's actions, and the maximum number of players that can be handled by the architecture. The legend indicates that the red dots represent the maximum number of players handled by the architecture.
Architectures with single-function nodes

Legend
- • maximum number of players
- ▲ action processing nodes' CPU utilisation
- ▼ network processing nodes' CPU utilisation

CPU utilisation [%]

number of nodes processing player's actions (2 network nodes)

number of players

0 1 2 3 4 5 6

100 200 300 400 500 600 700

0 10 20 30 40 50 60 70 80 90 100 110 120

Legend
- • maximum number of players
- ▲ action processing nodes' CPU utilisation
- ▼ network processing nodes' CPU utilisation
Synchronising issue

Client A

Try to pick up a mushroom

Action status - OK

mushroom collected

Action status - FAIL

Client B

Try to pick up a mushroom

Server
Desynchronisation of the game’s state

- **Server is not overloaded**
- **More successful actions**
Desynchronisation of the game’s state

- Server is overloaded
- More actions fail

Legend
- successful actions
- not successful actions

\[ y = 0.04855 \text{ (successful)} \]
\[ y = 0.0009119 \text{ (not successful)} \]
Conclusions

- Erlang/OTP is appropriate for developing prototypes of the distributed MMOG systems

- Rapid development of the prototype allows examining different algorithms in a relatively short period of time

- Future work
  - algorithms not dependent on the player’s position
Contents

- Massively Multiplayer Online Games
- Problems to solve
- Prototyping
- Using Erlang/OTP for the prototype
- Client and server applications
- Determining the maximum number of players
- Architectures with single and double-function nodes
- Synchronisation of the game’s state
- Conclusions