Towards Scalable Online Interactive Applications on Grids and Clouds

Sergei Gorlatch

University of Muenster, Germany
Real-Time Online Interactive Applications: ROIA

**Apps**: Online games, Interactive e-learning, Distributed simulations,…

- **Huge numbers** of concurrent users to a single application instance (> 40,000 simultaneous participants in Eve Online),
- **High update rate** of the application state (5-100 updates/sec),
- **Short response time** to actions (< 100ms), despite slow Internet links,
- **Variable load**: daytime-dependent,
- **Trust and security**: new challenges, e.g., cheating.

**Main Challenge – Scalability!**
**Runtime Characteristics of ROIA Services:**

- Multiple servers for a single application → **parallelism** + communication
- Clients are (re)connected to servers → **data (re)distribution** + scheduling
A model for Virtual Environment processing: Real-Time Loop

- Operated with high frequency (5-100Hz), in (soft) real time

1. Transfer user actions from the clients, receive in the servers
2. Process the new game state: apply game logic, move entities
3. Transfer the game state update to the clients
Current ROIA development:

- Choose a distribution/parallelization concept
- Create & maintain a customized run-time monitoring/controlling
  RTF liberates developer from these cumbersome tasks...
The Real-Time Framework (RTF) is a high-level development API & runtime environment that enables:

- Simplified app. development via parallelization and distribution techniques.
- Hiding and optimizing communication & synchronization (sockets in C++).
- Scalable and seamless use of resources (servers).
- Service adaptation and QoS for a changing user demand.
Distributing the ROIA State for Parallel Processing

- RTF supports a variety of distribution/parallelization approaches and allows the developer to combine different distributions.
• **RTF supports** all three distribution concepts *and their combinations*

• **Developer’s role:** to define zones and their possible replication

• **RTF’s role:** to automatically manage distribution, parallel computation of entities and all necessary communications
  – (re)assignment of zones to the servers during runtime in a seamless way for the developer and user, ensuring scalability (higher # users)
  – management tasks are performed by RTF, e.g. monitoring
• Tests with RTFDemo, a fast-paced First-Person-Shooter game:
  − Server real-time loop runs at 25 Hz
  − Clients are bots moving randomly and shooting at opponents in sight
  − Clients do not notice that they play in a multi-server environment

• Zoning scalability results:
  − RTF scales nearly linearly with the number of zones
  − Up to 1800 simultaneous clients with four zones/servers
  − Response time stays stable to up to ~450 clients per server

*Response time* = time between client’s action and server’s response to it
Evaluating RTF’s application scalability

• Replication scalability test: Quake 3 on top of RTF
  – RTF allows to scale Quake 3 and to increase the number of players
  – Players do not notice that they now play in a multi-server environment

• Zoning scalability test: Custom FPS with all typical game elements
  – Server real-time loop runs at 25 Hz
  – Clients are bots moving randomly and shooting at opponents in sight
OIA characteristics differ from classical Grid applications:

- **hanging user numbers** during runtime
- **dynamic user base** with changes of orders of magnitude
- **geographic distribution** of workloads due to users connecting from arbitrary world-wide locations but sharing the same application instance
- **single-hoster service provision** is often inflexible and cost-ineffective

**Approach:** Exploit characteristics of Cloud and IaaS:

- Provide dynamic server resources on-demand
- Scale applications using RTF to become massively multiplayer 
  **cost-effectively** on static Grids
That's what we want…

Short illustration…

Game Service Operator

Provides Game to Players

Resources within Cloud available on demand

Manually deploy & start

Automatically deploy & start

That's what we have…

Resources of Game Service Operator

Communication
ROIA Resource Management System for Clouds (ROIA-RMS)

esource Provisioning in Clouds with particular adaptation to **ROIA characteristics**

- Application-specific monitoring (RTF) reveals bottlenecks (potential QoS violation)

**ROIA-RMS** allocates resources w.r.t. cost-efficiency
### Industrial Cooperation

- **D Multi-User Online Cloud Authoring Platform (3D-MOCAP)**
  - Cooperation with *Spinor GmbH*, specialized on authoring systems and 3D engines (Shark 3D™) for computer games, simulations, etc.
  - Target: Cloud-based authoring system for multi-user 3D applications

<table>
<thead>
<tr>
<th>Low-entry barrier solutions, suitable to develop cost-effective 3D applications</th>
<th>Virtual Reality/Game software (e.g. Shark 3D™, Virtools™, Unity 3D™)</th>
<th>→ 3D-MOCAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>High-end libraries/engines, suitable to develop fully customized 3D applications (requiring major financial investments)</td>
<td>High-end engines (e.g. Unreal™)</td>
<td>MMO libraries (e.g. RTF™, Exit Games™) and engines (e.g. Bigworld™)</td>
</tr>
</tbody>
</table>
3D Multi-User Online Cloud Authoring Platform

- 3D multi-user online application with 3D-MOCAP
  - Application servers are running on **Cloud resources**
  - Application-specific assets are stored in **Cloud Storage Service**
  - **RTF** is used for communication and workload (re)distribution
  - Modifying applications at runtime by changing assets with **Authoring client**
A Demonstration of RTF

- Monitor the application service
  - Network characteristics
  - Event and client count
  - Saturation, etc.
- Add resources during runtime
- Redistribute load during runtime

The resource management is shown via a GUI which allows to:

- Clients have an avatar (robot) and move within the world
- They can shoot at each other’s avatar
- Zoning and replication within RTF is used to scale RTFDemo