HIGH AVAILABILITY AND DISASTER RECOVERY FOR IMDG

VLADIMIR KOMAROV, MIKHAIL GORELOV
SBERBANK OF RUSSIA
**ABOUT SPEAKERS**

**Vladimir Komarov**  
Enterprise IT Architect  
vikomarov@sberbank.ru

In Sberbank since 2010. He realized the concepts of operational data store (ODS) and retail risk data mart as a part of enterprise data warehouse. In 2015 performed the testing of 10+ distributed in-memory platforms for transaction processing. Now responsible for grid-based core banking infrastructure architecture including high availability and disaster recovery.

**Mikhail Gorelov**  
Operations expert & manager  
magorelov@sberbank.ru

In Sberbank since 2012. He is responsible for building the infrastructure landscape for the major mission-critical applications as core banking and cards processing including new grid-based banking platform. Now he acts as both expert and project manager in "18+" core banking transformation program.
ABOUT SBERBANK

SBERBANK

The largest bank in Russian Federation

- 16K+ offices in Russia, 11 time zones
- 110M+ retail clients
- 1M+ corporate clients
- 90K+ ATMs & POS terminals
- 50M+ active web & mobile banking users
**Availability** = \[
\frac{Total \ time - Downtime}{Total \ time}\] \times 100 \%

<table>
<thead>
<tr>
<th>Availability</th>
<th>Yearly downtime</th>
<th>target for 2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>99 %</td>
<td>3d 15:39:29.5</td>
<td></td>
</tr>
<tr>
<td>99.9 %</td>
<td>8:45:57.0</td>
<td></td>
</tr>
<tr>
<td>99.99 %</td>
<td>0:52:35.7</td>
<td></td>
</tr>
<tr>
<td>99.999 %</td>
<td>0:05:15.6</td>
<td></td>
</tr>
<tr>
<td>99.9999 %</td>
<td>0:00:31.6</td>
<td></td>
</tr>
</tbody>
</table>
OUR METHODS

• additional **control and checking tools**;

• **monitoring** improvement:
  • new metrics design;
  • new visualizations;

• **continuous testing**:
  • operational acceptance tests;
  • performance tests;
  • 45+ scenarios of destructive testing;

• keeping **incident response plan** up-to-date.
## Threats and Facilities

<table>
<thead>
<tr>
<th>Datacenter loss</th>
<th>DC interconnect failure</th>
<th>Application bugs, admin errors</th>
<th>User data corruption</th>
<th>HW/OS/JVM failures</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-disk data persistence</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Data redundancy</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Distributed cluster</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Data snapshots</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Point-in-time recovery</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Health self-check</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Data replication</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
THE LEGACY GRID-ENABLED ARCHITECTURE

Application servers
compute

In-memory data grid
caching & temporary storage

Relational DBMS
persistence & compute

Strengths

• Robust and stable persistence layer
• A grid hasn’t to be highly available

Weaknesses

• The write performance is limited by database
• The persistence layer is not horizontally scalable
• Data need to be converted from object representation to relational model
• Database and grid can become inconsistent if data is changed directly in the database
• The database requires high-end hardware
SBERBANK CORE BANKING PLATFORM ARCHITECTURE

Application servers
compute

In-memory data grid
compute & data persistence

Opportunities

• Fully horizontally scalable architecture on commodity hardware
• The data is stored as objects, no conversion required
• The only instance of the data

Challenges

• The grid has to persist the data
• The grid has to be fault tolerant
SERVICE CONTINUITY THREATS

- The above tree does not consider security issues
- Application and user issues cannot be solved at platform level
- Let’s focus on system issues!
THE CONCEPT OF SERVICE PROVIDER INTERFACE (SPI)

**API vs. SPI**

<table>
<thead>
<tr>
<th></th>
<th>API</th>
<th>SPI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Defined by</strong></td>
<td>Platform</td>
<td>Platform</td>
</tr>
<tr>
<td><strong>Implemented by</strong></td>
<td>Platform</td>
<td>System software (custom code)</td>
</tr>
<tr>
<td><strong>Called by</strong></td>
<td>Application (custom code)</td>
<td>Platform</td>
</tr>
</tbody>
</table>

Sberbank implements GridGain SPI:

- TopologyValidator
- AffinityFunction
THE CONCEPT OF AFFINITY

nodeFilter
the property of the cache that defines the set of nodes where the cache’s data can reside

partition()
the fast, simple and deterministic function usually division reminder mapping object to the partition (chunk)

assignPartitions()
the function distributing partitions (chunks) across the nodes
**Sberbank’s affinity implementation**

**Datacenter 1**

1 2 3 4 5 6 7 8

**Datacenter 2**

8 7 1 2 4 5 6 3

- The grid is distributed across 2 datacenters.
- Data connectivity is **limited to 8 nodes (a cell)**.
- Every partition has the master copy and 3 backups.
- Each datacenter has 2 copies of a partition.
- Both datacenters are active.

---

**Broken node:**

more nodes in the cluster → faster recovery

**Semi-broken node:**

more linked nodes → stronger performance impact

Find a balance!
• Nodes of a cell reside in different racks.
• Clos network provides stable high-speed connectivity.
• Doubled datacenter interconnect reduces split-brain probability.
• Every server contains NVMe flash and HDDs.
LET’S SPEAK ABOUT NETWORK FRAGMENTATION…

Regular operation

Fragmentation type 1

Datacenter loss

Fragmentation type 2

DC interconnect loss

Fragmentation type 3
HOW DOES GRIDGAIN RECOVER A BROKEN CLUSTER?

- End (commit or rollback) all the active transactions.
- Choose new cluster coordinator
- Call `TopologyValidator.validate()`
  - true (default)
    - Continue normal operation
  - false
    - Go read-only mode
**Decisions possible:**
- RW (read–write): continue normal operation
- AW (admin wait): freeze the cluster and wait for admin interaction

<table>
<thead>
<tr>
<th>DC1</th>
<th>DC2</th>
<th>Data</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Partial</td>
<td>✓</td>
<td>RW</td>
</tr>
<tr>
<td>All</td>
<td>Partial</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>None</td>
<td>✓</td>
<td>RW</td>
</tr>
<tr>
<td>All</td>
<td>None</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Partial</td>
<td>All</td>
<td>✓</td>
<td>AW</td>
</tr>
</tbody>
</table>
DECISION AUTOMATION USING QUORUM NODE

Datacenter 1

Datacenter 2

Quorum datacenter

Quorum node

RW

STOP

STOP
LOCAL FILE STORE (LFS)

RAM

Trx processing

Sync. write

Paged memory

HDD

Write-ahead log

Sync. write

Async. write

Paged disk storage (files)
**Current**

- **Snapshot** to local disk (full/incremental/differential)
- Snapshot **catalog** inside the data grid
- Copying to NAS **using NFS**
- Restoring on **arbitrary grid topology**

**Future**

- **Point-in-time recovery** using snapshot and WAL;
- **External backup catalog** in relational DBMS;
- Copying to SDS using **S3/SWIFT**;
- ...and more!
THANK YOU!

Vladimir Komarov <vikomarov@sberbank.ru>
Mikhail Gorelov <magorelov@sberbank.ru>