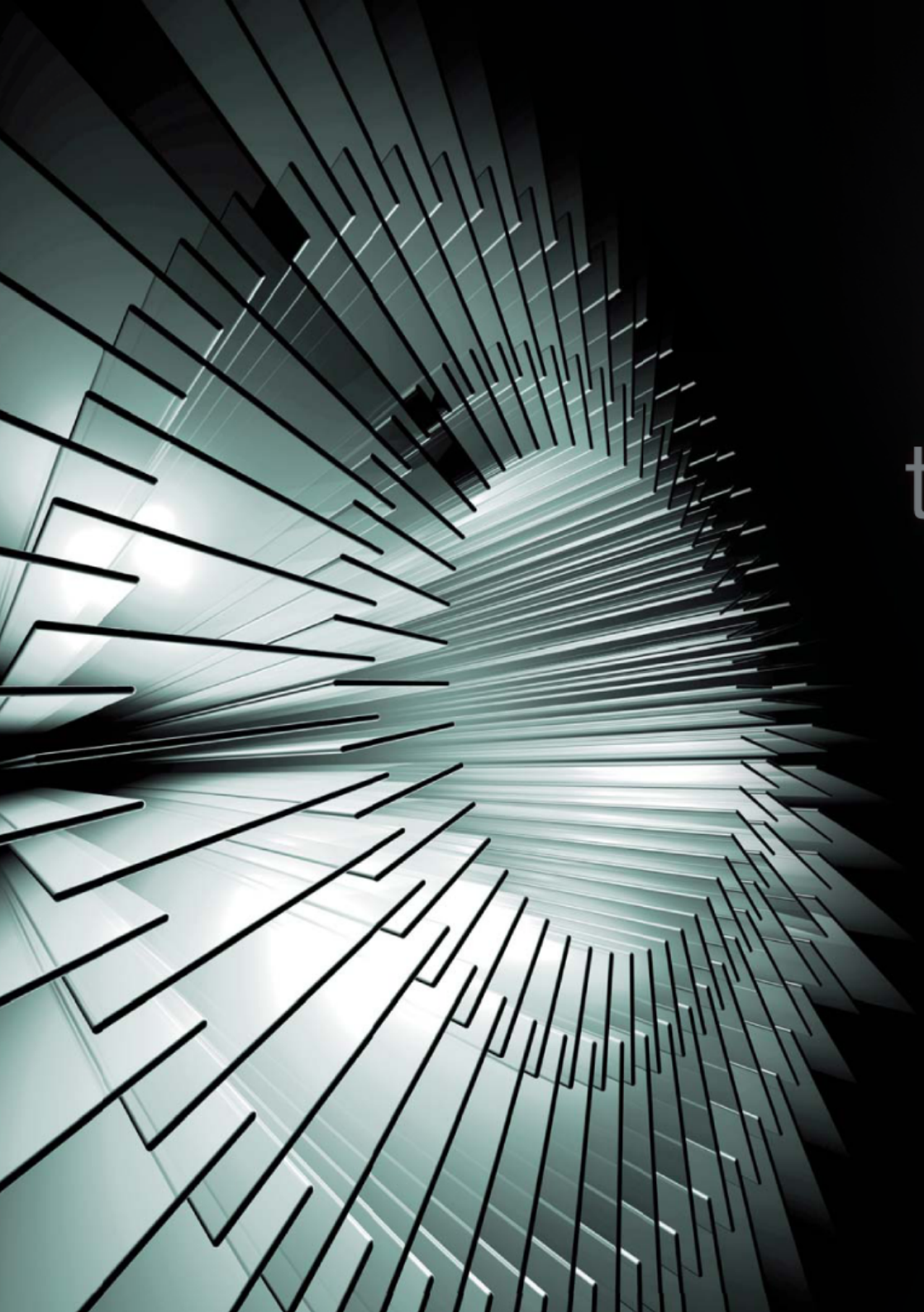


BLACKRAY

the open source data engine

www.blackray.org

softmethod

Presentation Agenda

- Brief History
- Technology Overview
- Positioning towards other Projects
- Roadmap
- The Team
- Wrap-Up

Brief BlackRay History

What is BlackRay?

- BlackRay is a relational, in-memory database
- SQL Support
- Fulltext (Tokenized) Search in Text fields
- Object-Oriented API Support
- Persistence via Files
- Scalable and Fault Tolerant
- Open Source, Open Community
- Dual licensed: GPL and Proprietary

BlackRay History

- Concept of BlackRay was developed in 1999 for a Web Phone Directory at Deutsche Telekom
- Development of current BlackRay started in 2005, as a Data Access Library
- Production Use at Deutsche Telekom in 2006
- Evolution from Library to Data Engine in 2007 and 2008
- Open Source under GPLv2/Dual License since June 2009

Why Build Another Database?

- Rather unique set of requirements:
  - Phone Directory with approx. 80 Million subscribers
  - All queries in the 10-500 Millisecond range
  - Approximately 2000 concurrent users
  - Over 500 Queries per Second (sustained)
  - Updates once a day may only take several minutes
  - Index needs to be tokenized (SQL: CONTAINS)
  - Phonetic search
  - Extensive Wildcard queries (leading/midspan/trailing)

Options available 1999

- Pretty much no options available:
  - ORACLE and DB2 did not support Tokenized Index
  - MySQL would not support the table size required

The first implementation in C was an embedded version of what is now called BlackRay.

The implementation was done on Solaris with 64 Bit Ultrasparc (Sun Ultra 10)

Options evaluated in 2005

- ORACLE 10g with Oracle Text
  - Worst case search performance in the tens of minutes
  - Updates of the 1TB+ Index takes hours or days....
- FAST Search and Transfer (now Microsoft)
  - Extremely hardware intensive, 48 Sun V280 Servers required
  - Not transactional, loses data on indexing
- PERST by McObject
- Thunderstone
  - Well, lets say Deutsche Telekom could not afford it.....

Decision to Implement BlackRay

- Decision was formed in Mid 2005
- Designed as a lightweight data access engine
- Implementation in C++ for performance and maintainability
- APIs for access across the network from different languages (Java, C++)
- Feature set was designed for our specific business case in a particular project

Current Release

- Current release 0.9.0 released on June 12th, 2009
- Production level quality
- All relevant index functions for large scale search applications
- APIs in C++ and Java fully functional
- SQL still only a small subset of the API functionality

Technology Overview

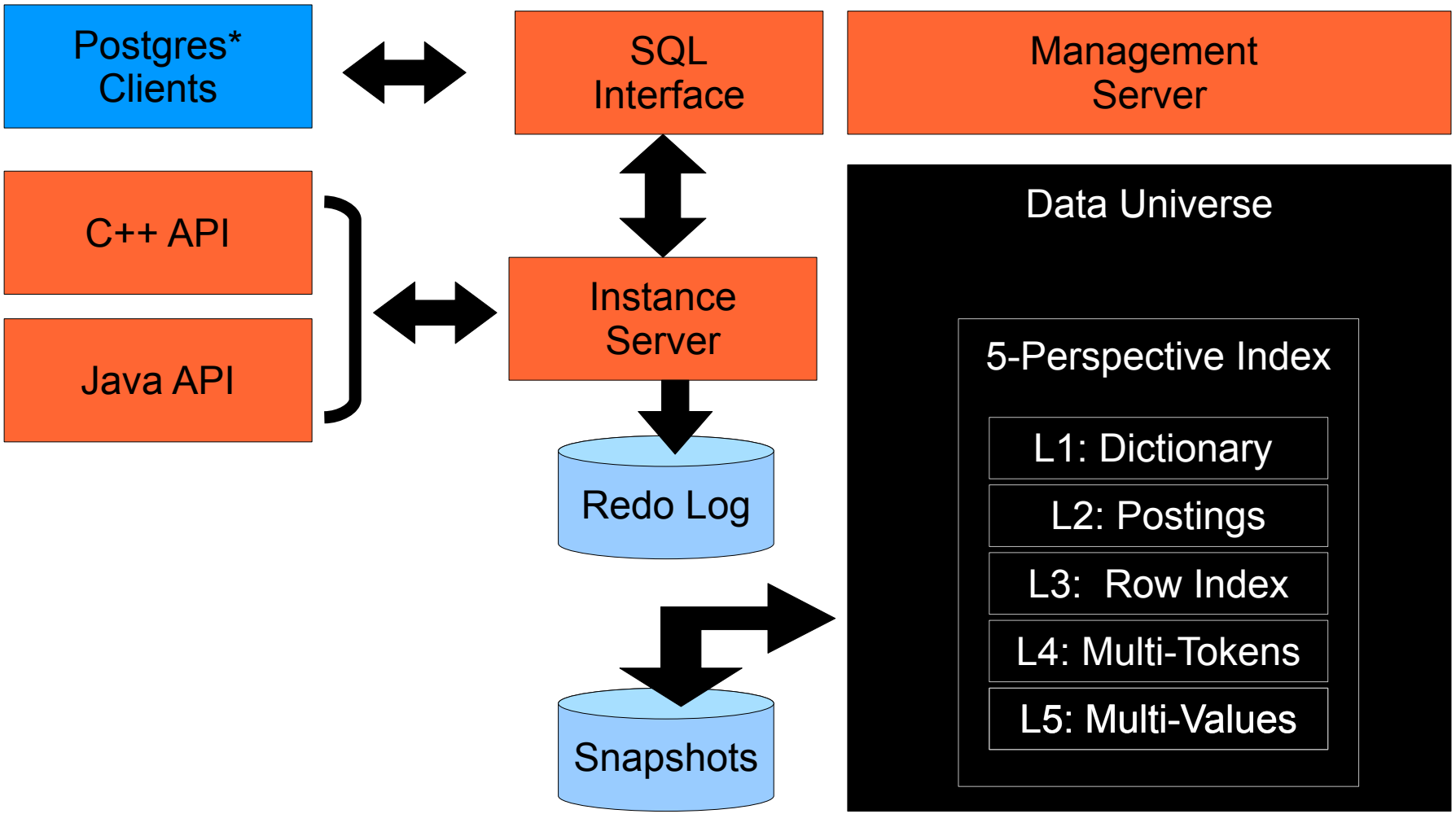
Why call it Data Engine?

- BlackRay is a hybrid between a relational database and a search engine → thus we call it „**data engine**“
- Database features:
  - Relational structure, with Join between tables
  - Wildcards and index functions
  - SQL and JDBC/ODBC
- Search Engine Features
  - Fulltext retrieval (token index)
  - Phonetic and similar approximation search
  - Extremely low latency

Why call it Data Engine?

- BlackRay supports the combination of Search Engine and Database features:
  - Full-text queries with wildcards and phonetics....
  - Multiple columns with different data types
  - Joining tables to better organize data

BlackRay Architecture



Hierarchical Model

- Each BlackRay node can hold many Instances
- One Management process per node
- Each Instance runs in a separate Process
- Instances are completely separated from each other
- Snapshots (for persistence) are taken on an Instance level
- In an Instance, Schemas and Tables can be created
- Queries can span across Tables and Schemas

Getting Data Into BlackRay

- Once an Instane is created, data can be loaded
- Schemas, Tables, and Indexes are created using an XML description language
- Standard loader utility to load CSV data
- Bulk loading is done with logging disabled
- Indexing is done with maximum degree of parallelism depending on CPUs
- After all data is indexed, a snapshot can be taken

Basic Load Performance Data

- German yellowpage and whitepage data
  - 60 Million subscribers
  - 100 Million phone numbers
  - Raw data approx 16GB
- Indexing performance
  - Total index size 11GB
  - Time to index: 40 Minutes, on dual 2GHz Xeon (Linux)
  - Updates: 300MB, 200K rows in approx 5 minutes
- Time to load snapshot: 3.5 Minutes for 11GB

Data Universe

- BlackRay features a 5-Perspective Index
  - Layer 1: Dictionary
  - Layer 2: Postings
  - Layer 3: Row Index
  - Layer 4: Multi-Token Layer
  - Layer 5: Multi-Value Layer
- Layer 1 and 2 comprise a fully inverted Index
- Statistics in this Index used for Query Plan Building

Snapshots and Data Versioning

- Persistence is done via file based snapshots
- Snapshots consist of all schemas in one instance
- Snapshots have a version number
- To make a backup of the data, simply copy the snapshot file to a backup media
- It is possible to load an older snapshot: Data is version controlled if older snapshots are stored

Transactions in BlackRay

- BlackRay supports transactions via a Redo Log
- All commands that modify data are logged if requested
- In case of a crash, the latest snapshot will be loaded
- Replay of the transaction log will then bring the database back to a consistent state
- Redo Log is eliminated when a snapshot is persisted
- For better performance snapshots should be taken periodically, ideally after each bulk update

Query APIs

- C++ and Java Object Oriented APIs are available
- Built with ICE (ZeroC) as the Network and Object Brokerage Protocol
- Query Objects are constructed using an Object Builder
- Execution via the network, Results as Objects
- Load balacing and failover built into the protocol
- Bindings for Python already done, C#, Ruby and PHP in the works

Query APIs

- Queries can use any combination of OR, AND
- Index functions (phonetic, sysnonyms, stopwords) can be stacked
- Token search (fulltext) and wildcard are also supported
- Advantage of the APIs:
  - Minimized overhead
  - Very low latency
  - High availability

Management Features

- Management Server acts as central broker for all Instances
- Command line tools for administration
- SNMP management:
  - Health check of Instances
  - Statistics , including access counters and performance measurements

Administration Requirements

- In-Memory Databases require very little administrative tasks
- Configuration via one file per Instance
- Disk layout etc all are of no importance
- Backups are performed by copying snapshots
- Recovery is done by restoring a snapshot
- No daily administration required
- SNMP allows remote supervision with common tools

Some more details

- Written in C++
- Relies heavily on boost
- Compiles well with gcc and Sun Studio 12
- Behaves well on Linux, Solaris, OpenSolaris and MacOS
- Complete 64 Bit development
- Use of cmake for multi platform support

Positioning BlackRay

BlackRay and other Projects

- BlackRay is being positioned as a Query intensive database addition
- Ideally suited where updates are done in Bulk and searches outweigh updates by many orders of magnitude
- Wildcards come with little overhead: No overhead for trailing wildcard, some overhead for leading and midspan wildcard
- Good match when index functions such as phonetic or word rotation/position search combined with relational data are required

OS Projects with similar Goals

- Relational Databases (the usual suspects):
  - MySQL, MariaDB, Drizzle....
  - PostgreSQL...
  - Many more alternatives, including embedded etc....
- Fulltext Search:
  - Sphinx
  - Lucene
- In-Memory Databases:
  - FastDB
  - HSQLDB/H2 (Java → Garbage collection issues....)

Commercial Alternatives

- Commercial In-Memory Databases
  - ORACLE/TimesTen (Acquired by ORACLE in 2005)
  - IBM/SolidDB (Acquired by IBM in 2007)
  - VoltDB (No real data available as of yet)
  - eXtremeDB (embedded use only)

- Dual-Licensed Alternatives
  - CSQL (Open Source Version is severely crippled)
  - MySQL with memcached

Is it the right thing for me?

- BlackRay is not designed as a 100% RDBMS replacement
- Questions to ask:
  - Do I need ad-hoc data updates, or are updates done in bulk?
  - How important are fulltext search and extensive wildcards?
  - How large is my data? Gigabytes: OK. Terrabytes: Not yet
  - Do I need a relational data model?
  - Is SQL an important feature?

BlackRay will fit you well when...

... searches outweigh updates

... data is updated in bulk

... needs to be available quickly

... you have Gigabytes not Terrabytes

... you need lots of SELECTs

... SQL is necessary

... a relational data model is required (JOIN)

... source code must be available

Project Roadmap

Immediate Roadmap

- Pending immediate release:
  - BlackRay Admin Console (Remora) 0.1.0
- Upcoming 0.10.0
  - Rewrite of SQL Parser
  - PostgreSQL client compatibility (via network protocol) to allow JDBC/ODBC... via PostgreSQL driver
  - Rewritten CLI tools
  - Some bugfixes (potential memory leaks)
  - Authentication for Instances

Midterm Roadmap

- Data manipulation/Query Features
  - Ad-hoc INSERT/UPDATE/DELETE support
  - Aggregate functions for SELECT
- Scalability Features
  - Sharding & Partitioning Options
  - Federated Search
- Security Features
  - Improved User and Access Control concepts
  - SSL for all connections
- Improved Statistics Module

Longterm Roadmap

- Integration with other DBMSs
  - Storage Engine for MariaDB ⟶ Depends on potential modification needs of the storage engine interfaces
  - Trigger-based updates to support BlackRay as a Query cache instance over a regular DBMS
- Standalone Engine Improvements
  - SQL92 compliance: SUBSELECT/UNION support
  - Triggers in BlackRay

The Team behind BlackRay

Development Team

- SoftMethod Core Team
  - Thomas Wunschel – Lead Developer
  - Felix Schupp – Project Sponsor
  - Andreas Meyer – Documentation, Porting
  - Frank Fiedler
- Outside Contributors
  - Mike Alexeev – Senior Developer
  - Andreas Strafner – Developer, Porting to AIX

Thomas Wunschel

- Director of Development, SoftMethod GmbH
- Almost 10 years of development experience
- Involved with BlackRay and its applications since 2005
- Currently involved in the Network Protocol Stack
- Lead Designer and Decision Lead for new Features

Mike Alexeev

- Senior Software Developer
- Over 10 years of C++ experience
- First outside committer to BlackRay
- Currently involved in rewriting the SQL grammar to support all index features available via the APIs

Felix Schupp

- Managing Director, SoftMethod GmbH
- Over 10 years of commercial software development
- Designer of first BlackRay predecessor in 1999
- Project sponsor and spokesperson
- Responsible for funding and applications
- Guide and coach in the development process

Wrap-Up

What to do next

- Get BlackRay:
  - Register yourself on http://forge.softmethod.de
  - SVN checkout available at http://svn.softmethod.de/opensource/blackray/trunk
- Get Involved
  - Anyone can register and create tickets, news etc
  - We have an active mailing list for discussion as well
- Contribute
  - We require a signed Contributor agreement before being allowed commit access to the repository

Contact Us

- Website: http://www.blackray.org
- Twitter: http://twitter.com/dataengine
- Facebook http://facebook.com/dataengine
- Mailing List: http://lists.softmethod.de
- Download: http://sourceforge.net/projects/blackray

- Felix: felix.schupp@softmethod.de
- Thomas: thomas.wunschel@softmethod.de