# New in Firebird 2.5: SuperClassic Architecture

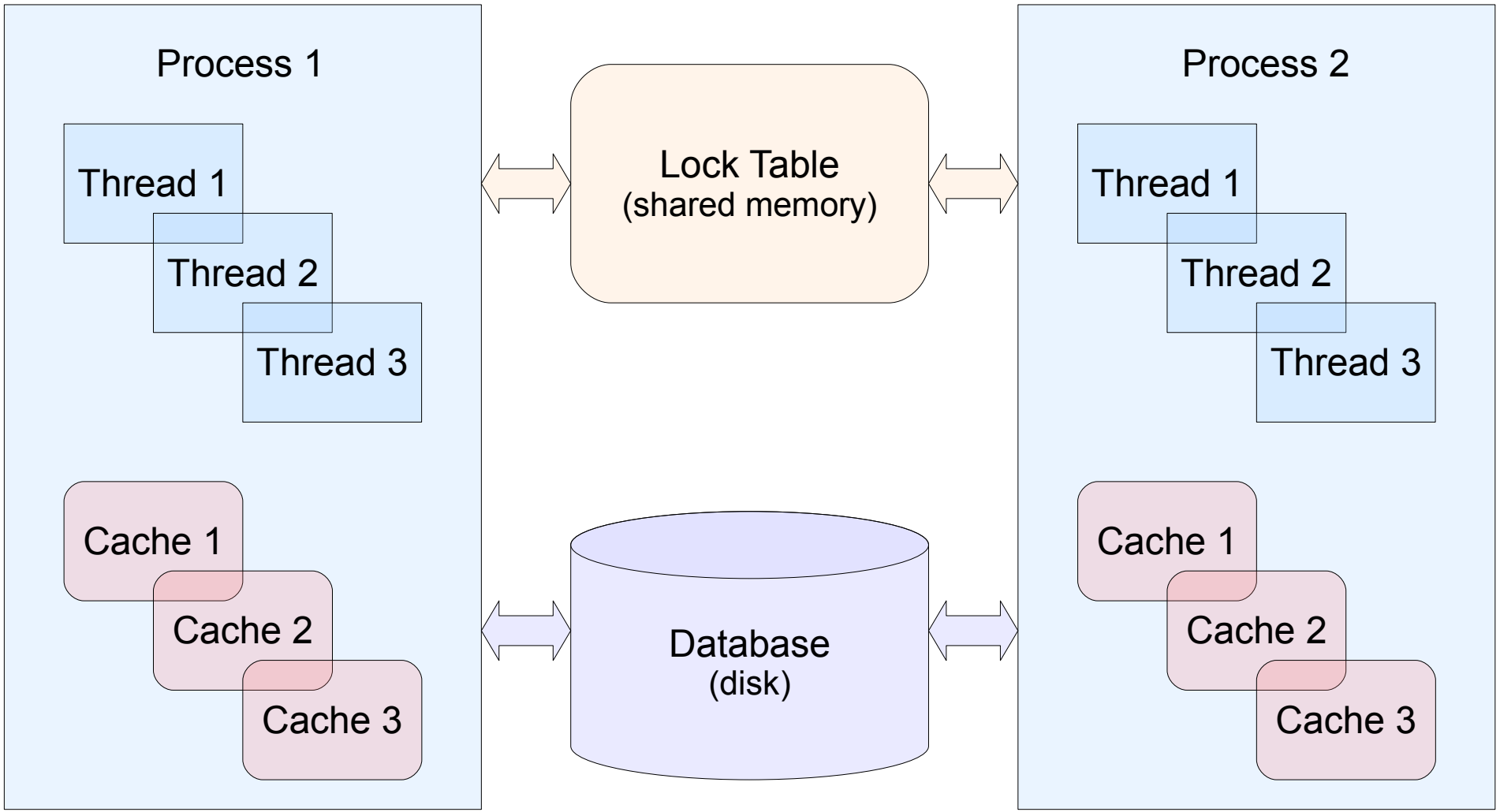
Dmitry Yemanov

Firebird Project
http://www.firebirdsql.org/

# Historical background

- **Vulcan project and SAS Institute**
  - «No-shared-cache» mode
- **Key ideas**
  - Multi-threaded multi-process server
  - Shared database access
  - Separate caches per connection
  - Fast communication (e.g. lock management) between local connections (no IPC involved)

# SuperClassic: architecture



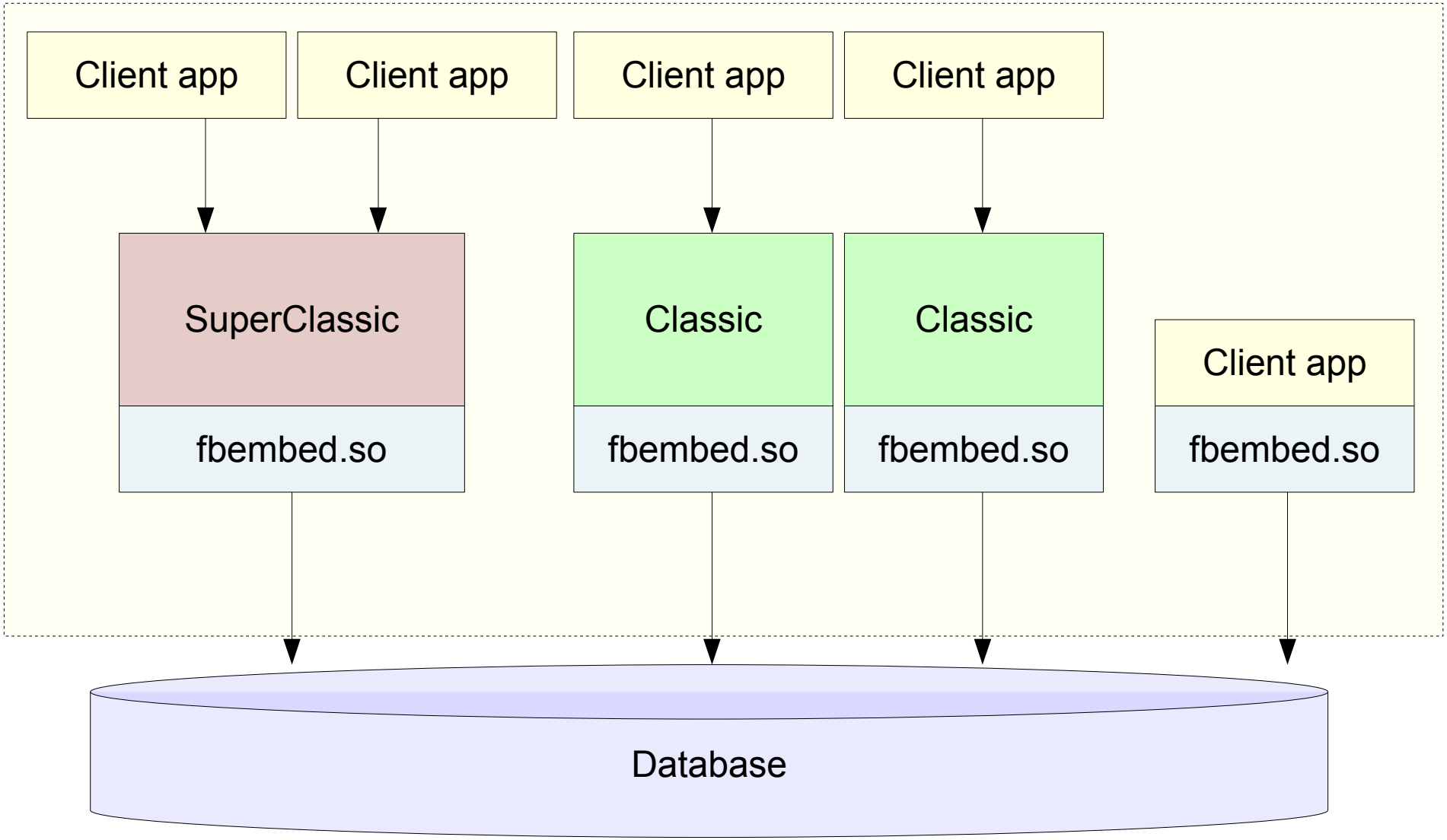
# Behind the scenes

- **Threadability as primary goal**
  - Common threading platform across all supported architectures
  - Overall thread safety (including client and embedded libraries)
  - Improved synchronization logic,
    say «goodbye» to the global mutex
  - Asynchronous port cleanup, sweep, services and statement/attachment cancellation

# Behind the scenes

- **Embedded engine uses new architecture**
  - Host applications may still have multiple connections to the same database
  - Different host applications may safely access the same database simultanously
  - Official utilities (gbak, gfix, etc) and 3rd party tools (DBWorkbench, IBExpert, etc) can be used in parallel with your application as well

# Shared database access



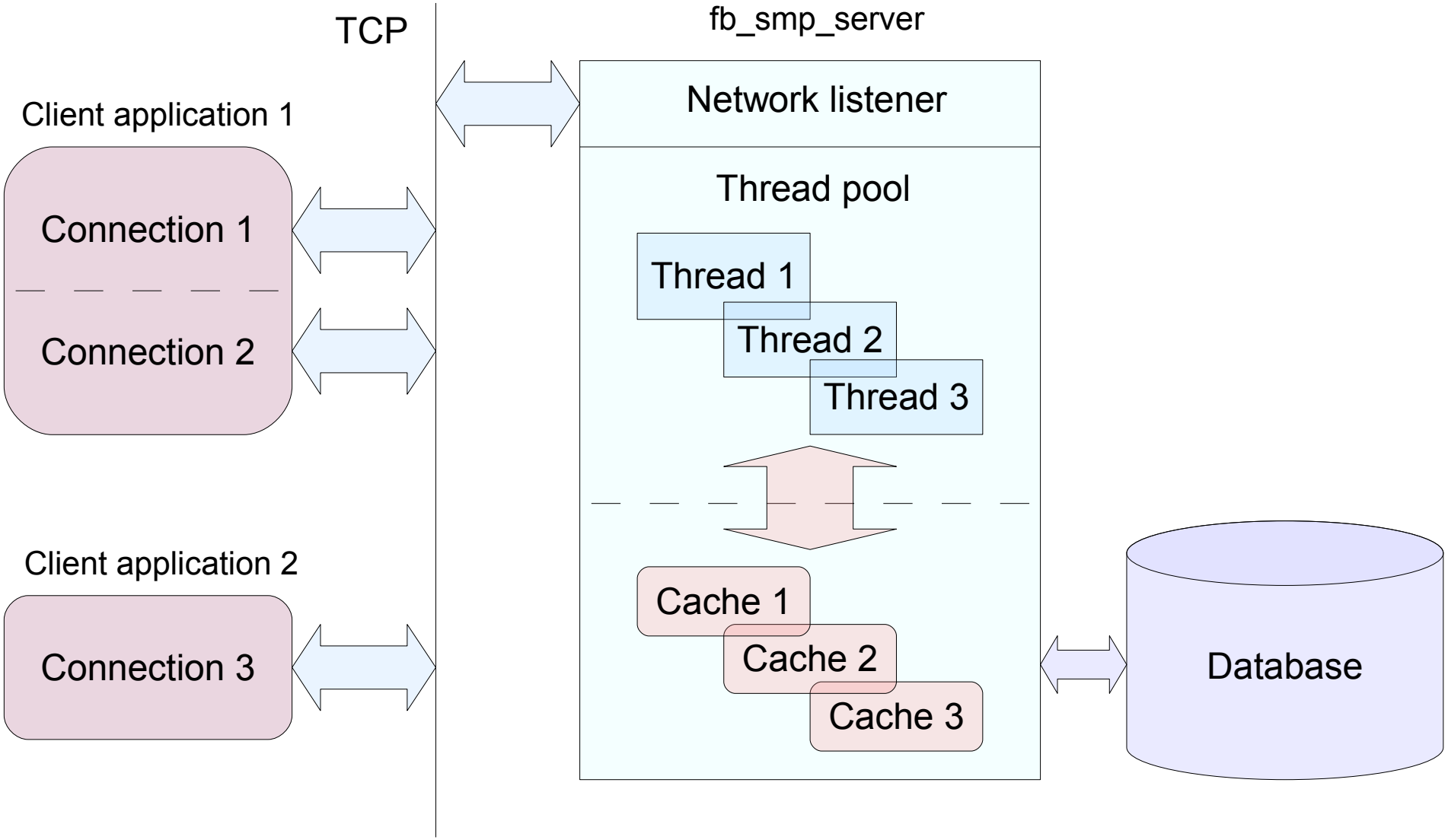
# SuperClassic as updated Classic

- **General**
  - Both are backed by the embedded engine
  - The same thing but with different visible behavior
- **Classic**
  - Dedicated listener process (xinetd or native), worker process per user connection
- **SuperClassic**
  - Single listener and worker process, thread pool to operate user requests

# Using SuperClassic

- **Windows**
  - Same executable as for the regular Classic
  - Run as application: fb_inet_server -a -m
  - Run as service: instsvc install -m
- **POSIX**
  - New executable: fb_smp_server
  - Installed as a daemon (similarly to SuperServer), no need in xinetd

# SuperClassic: application



# SuperClassic benefits

- **As compared with Classic**
  - Better scalability (number of connections)
  - Slightly better performance
  - Server can be safely shutdown as a whole
  - Possibility to enumerate attached databases/users
  - Security database connection is cached
- **As compared with SuperServer**
  - Better concurrency on SMP / multi-core hardware
  - Better scalability (connections are not limited)

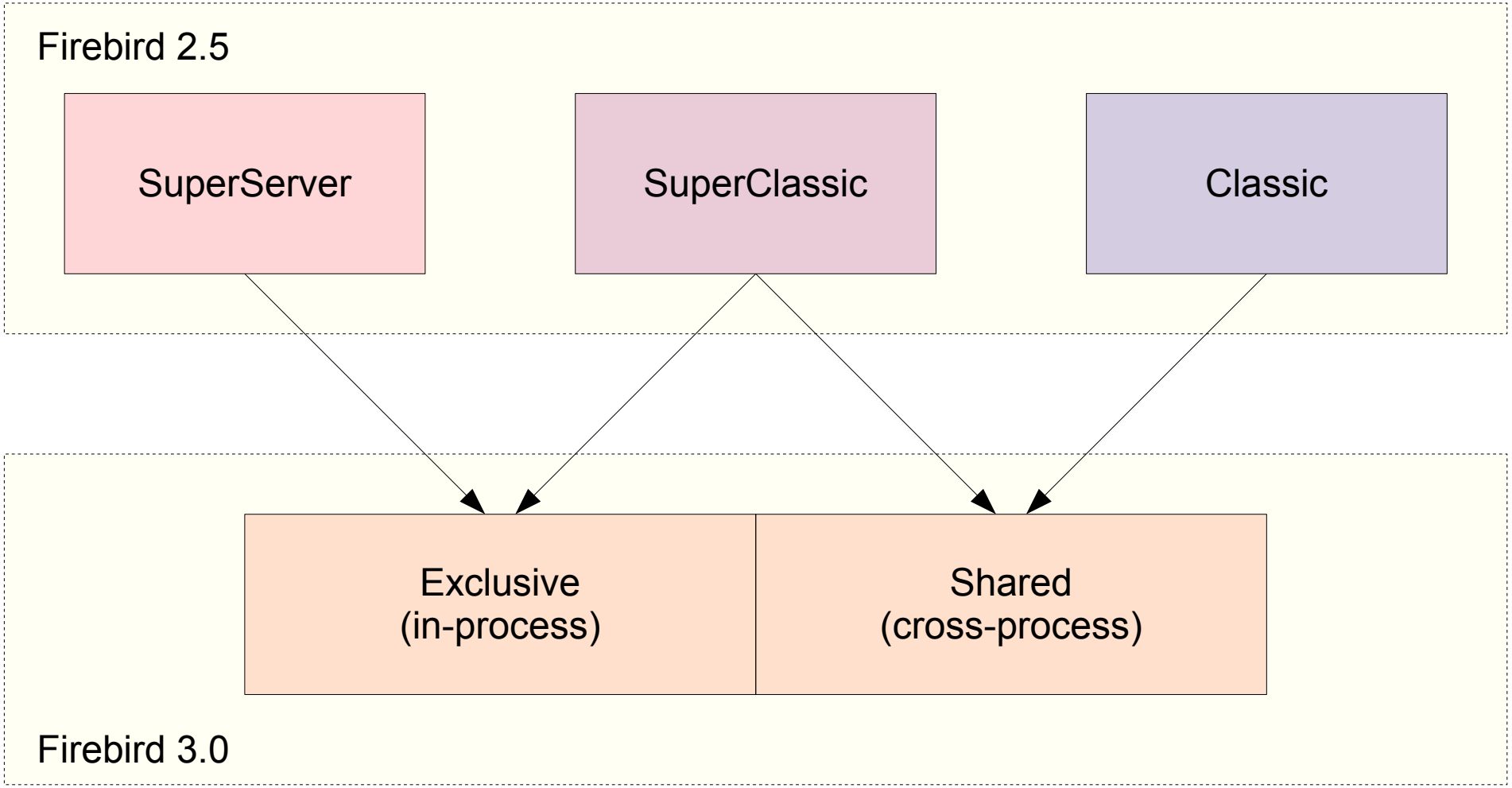
# SuperClassic drawbacks

- **As compared with Classic**
  - Server crash affects all user connections
- **As compared with SuperServer**
  - Somewhat ineffective memory usage and extra I/O overhead
  - High lock table contention (page locks), requires careful tuning the lock manager
- **General**
  - Doesn't make much sense on 32-bit systems

# Future evolution

- **Overview**
  - Single architecture supporting both in-process and cross-process interaction
  - Shared caches per database per process
  - Exclusive database access: single process, multiple threads, local synchronization
  - Shared database access: multiple processes, multiple threads, combined synchronization
  - Choice between process-per-connection mode and the thread pool

# Future evolution



# Questions?

**Recorded webinar will be available at**
**http://www.MindTheBird.com/**