Experience with C++11 in ArangoDB

SFBay Association of C/C++ Users, 11 March 2015

Max Neunhöffer

Max Neunhöffer

- I am a mathematician
- "Earlier life": Research in Computer Algebra
  (Computational Group Theory)
- Always juggled with big data
- Now: working in database development, NoSQL, ArangoDB
- I like:
  - research,
  - hacking,
  - teaching,
  - tickling the highest performance out of computer systems.

ArangoDB GmbH

- triAGENS GmbH offers consulting services since 2004:
  - software architecture
  - project management
  - software development
  - business analysis
- a lot of experience with specialised database systems.
- have done NoSQL, before the term was coined at all
- 2011/2012, an idea emerged:
  to build the database one had wished to have all those years!
- development of ArangoDB as open source software since 2012
- ArangoDB GmbH: spin-off to take care of ArangoDB (2014)

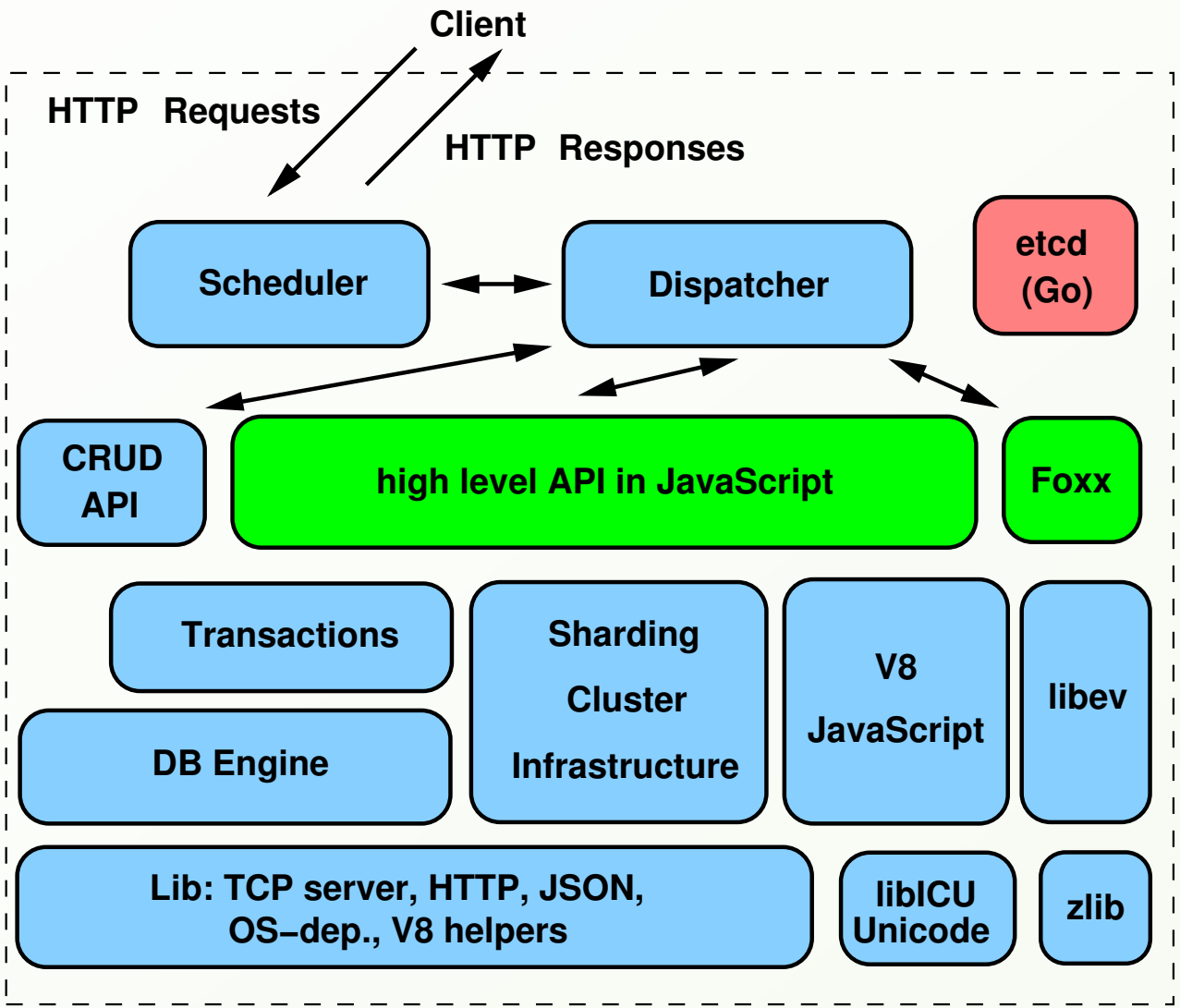
ArangoDB

- is a multi-model database (document store & graph database),
- is open source and free (Apache 2 license),
- offers convenient queries (via HTTP/REST and AQL),
- including joins between different collections,
- configurable consistency guarantees using transactions
- is memory efficient by shape detection,
- uses JavaScript throughout (Google's V8 built into server),
- API extensible by JS code in the Foxx Microservice Framework,
- offers many drivers for a wide range of languages,
- is easy to use with web front end and good documentation,
- and enjoys good community as well as professional support.

Organisation

- is developed by ArangoDB GmbH in Cologne (~ 12 people)
- the community helps, in particular with drivers
- is hosted on `https://github.com/ArangoDB/ArangoDB`
- runs on (at least) Linux, Mac OSX, Windows, raspberry Pi
- offers binary packages and virtual machines and containers
- the DB kernel is written in C++
- has Google's V8 embedded to use JavaScript for higher layers
- uses HTTP/REST and JSON as external API

ArangoDB Architecture



ArangoDB in numbers

- DB engine written in C++
- embeds Google's V8 (~ 130 000 lines of code)
- mostly in memory, using **memory mapped files**
- processes JSON data, **schema-less** but "shapes"
- library: ~ 123 000 lines (C++)
- DB engine: ~ 183 000 lines (C++, including 12 000 for utilities)
- JavaScript layer: ~ 1 106 000 lines of code
  - ~ 93 000 standard API implementation
  - ~ 318 000 Foxx apps (API extensions, web front end)
  - ~ 456 000 unit tests
  - ~ 239 000 `node.js` modules
- further unit tests: ~ 9 000 C++ and ~ 23 000 Ruby for HTTP
- plus documentation
- and drivers (in other repositories)

Platforms/architectures

We officially support

- **Linux** (binaries for CentOS, Debian, Fedora, RedHat, SLE, Ubuntu, gentoo, openSuse)
  with `GCC` or `clang`
- **Mac OSX** (all reasonably modern versions on Intel arch)
  with `GCC` or `clang`
- **Windows** (at least Windows 7 and later)
  with `Visual Studio 2013`
- raspberry pi
  with `GCC`

on the architectures

- Intel i386
- Intel/AMD x86_64
- ARM

→ emphasis on 64bit machines. (should compile fine on Posix.)

Switch to C++11

Before March 2014 we

- had a lot of C code (lower levels), and
- only used C++03.

For the 2.1 release (May 2014) and later we decided to

- compile most C code with a C++ compiler,
- change some to (proper) C++,
- adopt C++11, and
- withdraw support for systems without `-std=c++11`.

Since then we have eradicated all C-code.

Systems support for C++11

To get `-std=c++11` we need

- GCC at least 4.8 on Linux or older Mac OSX,
- clang at least Apple's 5.1 on Mac OSX $\geq$ 10.9
- Visual Studio 2013 on Windows
- GCC at least 4.9 on raspberry pi (Linux)

(We use the OpenSuse Build System.)

We had to drop support for Mandriva 2011.

On the following systems we had to do some "Eiertanz" to install our own compiler (GCC 4.8.2):

- CentOS-6
- Debian 6.0
- Debian 7.0
- raspberry pi
- RedHat 6
- SLE 11 SP3
- Ubuntu 12.04 (LTS)

Advantages from switching code from C to C++

- No more trouble calling C++ from C, because most is C++ now.
- Can use construction/destruction in more places.
- Can use STL in more places.
- C-compiler support in Visual Studio is rotting.

C++11 — the best (for us)

- `auto` and `decltype` save a lot of thinking and typing.
  (very occasionally it produces unforeseen results).
- `unordered_map` and `unordered_set` are useful.
  (We did not want to create a dependency on boost.)
- `shared_ptr` and `unique_ptr` work well and help.
- standardised atomic variables and operations are invaluable in a multi-threaded environment.
- explicit control over memory barriers and sharing
- nice helpers `to_string` and `stol`
- `emplace` useful
- Rvalue refs and move constructors good for clever classes
- range based `for` reads very nicely and is convenient
- lambda functions are useful for local, but repeated tasks
- `override` and `final` help the compiler to help

C++11 — less important (for us)

- `nullptr` instead of 0
- Strongly typed enums for better type checking
- no more annoying problems with > >
- `->` syntax for return types
- variadic templates
- `static_assert`

The move — what was the most work

- Actually use C++ features in C code.
- Fix all the build scripts with `-std=c++11` and sort out build environments.
- Bison/flex like to use `register` which now produces deprecation warnings.
- `make_pair` is now without template parameters.

Actual problems with C++11

We see very few actual problems:

- C++11 does not go far enough for multi-threading: for example no R/W locks.
- Some problems between different compilers (gcc/clang and Visual Studio): `constexpr` and `__thread_local`
- `make_pair<T1, T2>(a,b)` does no longer compile
- need GCC $\geq$ 4.8 or clang $\geq$ 3.4 or Visual Studio 2013
- `(std::min)` and `(std::max)` if `Windows.h` is included
- different namespace resolution in Visual Studio
- more casts necessary in Visual Studio (instead of implicit integer type conversions)
- `#include <functional>` necessary under Visual Studio
- double copy constructor not allowed under Visual Studio

We are currently hiring a senior C++ developer

`http://bit.ly/1s4t4qN`

Contact: `jobs@arangodb.com`