Firebird development process

Past, present and future

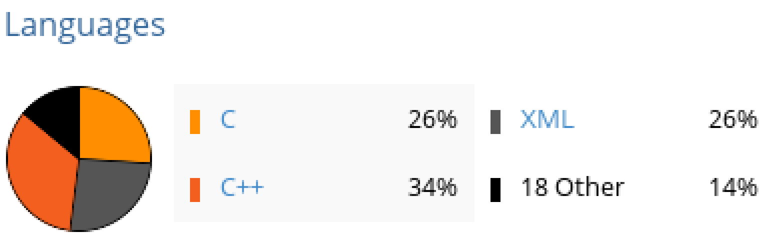
Pavel Cisar, IBPhoenix

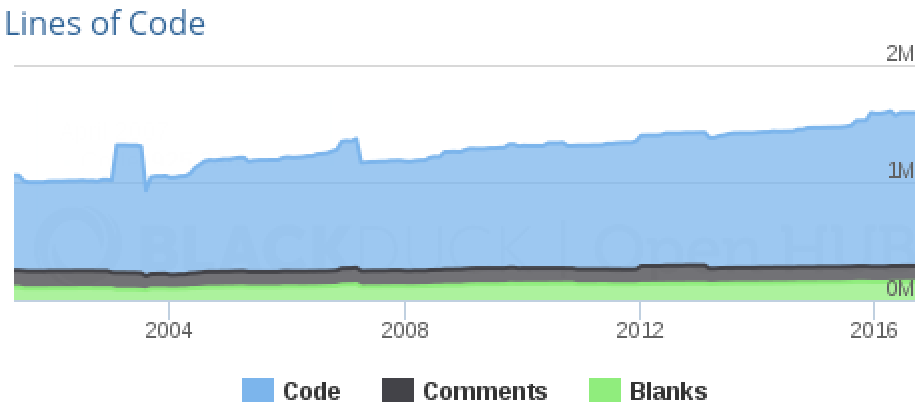
Firebird Conference 2016, Prague

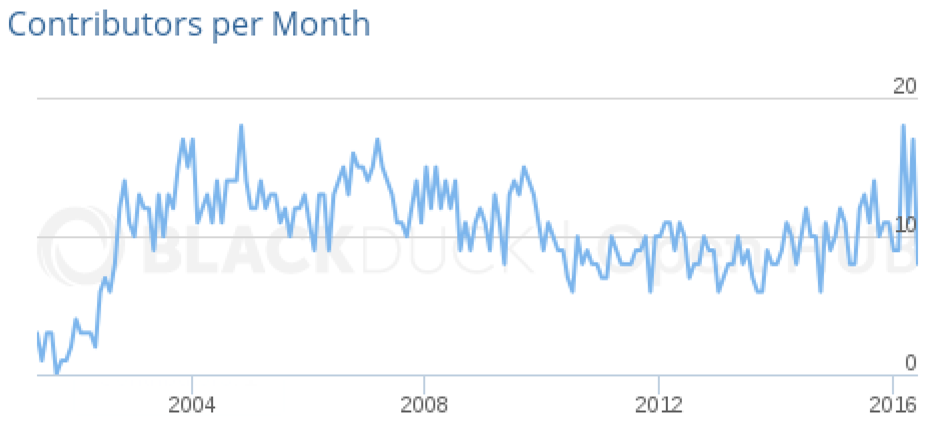
Few facts about Firebird

In a Nutshell, Firebird...

- was established on 31th July 2000
- has had 55,533 commits made by 70 contributors representing 1,306,723 lines of code
- is mostly written in C with a very low number of source code comments (6%)
- has a well established, mature codebase maintained by a large development team with stable Y-O-Y commits
- released 6 major versions and 25 maintenance versions of Firebird engine in 65 releases
- has 6 sub-projects (Java, .NET, ODBC and Python drivers, Documentation and QA)
- Main platforms are Windows and Linux, with extended support for MacOS
- Also (some versions) available on Solaris, FreeBSD, AIX, HP-UX, Sinix-Z and Android







From the tracker...

Totals

Resolution	Bugs	Improv.	New feature	Total
Fixed	2275	324	171	2770
Unresolved	616	360	234	1210
Won't Fix	553	93	71	717
Duplicate	166	35	53	254
Incomplete	42	3	4	49
Cannot Reproduce	191	3	1	195
Total	3843	818	534	5195

Processed in releases so far

Resolution	Bugs	Improvement	New feature	Total
Fixed	2077	310	164	2551

Fixed in average 39 issues per release, 14 per month.

Unscheduled and affecting v2.1 and up

Resolution	Bugs	Improvement	New feature	Total
Fixed	8	0	0	8
Unresolved	395	3	17	415
Total	403	3	17	423

Issues for 4.0

Resolution	Bugs	Improvement	New feature	Total
Fixed	99	13	4	116
Unresolved	1	15	10	26
Total	100	28	14	142

Development process in the past

2000-2002: Initial stage

- Loosely organized group of enthusiasts
- Uphill battle for survival (FB 1.0 released 12th March 2002)
- Defining Identity (logo, brand, community presence etc.)
- Competition from Borland InterBase and Yaffil
- Work on Firebird 1.5 started (alpha in October 2002)

2003: Turning the tide

- Firebird Foundation formed
- Fight with Mozilla over Firebird name
- Firebird 1.5 in RC stage (RC7 in November)
- Work started on Firebird 2.0
- Firebird 1.0.3 (3th June, last form 1.0 line)
- Yaffil merges with Firebird (2nd December 2003)
- Jim Starkey starts working on Vulcan (17th December)
- QA initiative started (using QMTest)

2004-2005: Hope & Strife

- Firebird 1.5 released 21th February 2004 (after 9 RC's)
- QA around QMTest evolved into useable state
- Vulcan and VulcanJ (QA)
- Work on 2.0 continues (with hopes to include results from Vulcan development)
- Continuous problems with quality, efficiency and interpersonal relations

2006: Consolidation

- Jim leaves Vulcan for MySQL (18th February)
- Nikolay leaves for Red Soft
- Firebird 2.0 released on 12th November
- JIRA implemented (June) and website moved

2007-2008: Down to Earth

- David S. Rushby died in July 2007
- Firebird 2.1 development and release (18th April 2008)
- Firebird 2.5 started (Alpha in September 2008)
- Bold plans (January 2008):
  - Q1: v2.1 RC and Final, v2.0.4, v2.5 Alpha
  - Q2: v2.1.1, v2.5 Beta
  - Q3: v1.5.6, v2.0.5, v2.5 RC
  - Q4: v2.1.2, v2.5 Final, v3.0 Alpha

- Technical Task Group (TTG) formed (December 2008)

2009-2010 It's getting complicated

- kinterbasdb 3.3.0 released (January 2009) that works with FB 2.1 but had problems with 2.5. Firebird QA in serious jeopardy.
- QMTest abandoned in favor of home-made system fbtest (October 2009)
- Big Infrastructure Crash (December 2009)
- Big migration from CVS to Subversion (2009-2010)
- Building build farm, take one
- Firebird 2.5 released on 4th November 2010

2011-2016 What the Hell

Bumpy road to v3.0

- New website (2011)
- Build farm for daily snapshot builds (Windows and Linux)
- Problems with QA builds up. It's time for new driver - FDB (initial release 2012).
- Number of FB versions in development reduced from 4 to 3 (2.0.7 from April 2012 was the last one from 2.0 line, decided in July)
- June 2013 - First Alpha for 3.0
- August 2013 - Pavel Zotov joined QA
- January 2014 - FB 3.0 Alpha 2
- April 2014 - Checkpoint for 3.0 and post 3.0 development. Maintenance release policy changed.
- October/November 2014 - Conference & 4.0 feature set planning & 3.0 Beta 1
- 2015 - Year of the 3.0 release. We almost made it.
- 2015 - First big financial crisis
- 2016 - 3.0 is out, work started on 4.0.
- 2016 - Second big financial crisis.
- 2016 - Migration from SF Subversion to GitHub.

Fixed 1062 bugs, 183 improvements, 53 new features:

- 2011: 2.1.4, 2.5.1
- 2012: 2.0.7
- 2013: 2.1.5, 2.5.2, security updates to 2.1.5 and 2.5.2, 3.0 Alpha 1
- 2014: 2.1.6 and 2.1.7, 2.5.3, 3.0 Alpha 2 and Beta 1
- 2015: 2.5.4 and 2.5.5, 3.0 Beta 2 and RC1
- 2016: 3.0 RC2 and Release, 2.5.6, 3.0.1

Technical Task Group (TTG)

- Developers and project members & Foundation sponsors and members
- For discusion about development plans and priorities
- Founded 2008, 4 meetings (2x 2008, 2010 and 2015/16)

Lessons from our development process

What is it about

- People
- Communication
- Infrastructure
- Process
- Funding

People (The Bright Side)

- We have a great team:
  - skills
  - motivation
  - dedication
  - responsibility
  - interpersonal relationships
  - initiative and able to act both independently and as a team
- Cooperation with RedSoft

People (The Dark Side)

- Group is too small - 3.5 full-time core developers, 3 part-time QA workers, skeleton staff for sub-projects and supporting tasks (~15 people in total)
- Some tasks are not sustainably filled or filled enough or at all
- Building overworking and burnout syndrome (most people are with project for 12 and more years)

Communication (The Bright Side)

- Well established channels (private and public)
- Good communication culture
- Intragroup communication and small scale cooperation (2-5 people) works very well

Communication (The Dark Side)

- Group communication and decision making is slow
- Language barrier
- No dedicated personel
- Communication rings are not connected very well
  - Group to group communication
  - Public records and "broadcasting"

Infrastructure (The Bright Side)

- Web(s)
- Tracker (JIRA)

- Subversion & Git
- Build farm
- Download central

Infrastructure (The Dark Side)

- Very outdated JIRA (decommissioned 5 years ago) in semifragille state
- Reocuring problems with infrastructure hosted at BroadView Software in Canada
- Fragmentation
- No dedicated personel

Process (The Bright Side)

- Clear distribution of responsibilities
- Functional routine to create a release
- Functional problem tracking
- Functional QA to keep regressions under control
- Minimal bureaucracy

Process (The Dark Side)

- **No formal process** (with schedules, checkpoints, "meetings", checklists)
- Plans only at basic level, no fixed routine for planning and problem resolution
- Actual status and prospects of future are not well communicated to all concerned. Important information are unevenly distributed and are not shared very well.
- Opaque nature of work leads to internal inefficiency and increases entry barrier for new people.

Funding (The Bright Side)

- Firebird Foundation facilitates funding from (corporate and private) sponsors
- Enough funds and resources to operate at **fundamental** level

Funding (The Dark Side)

- Not enough funds and resources to grow
- Occasional shortage of takings (so far always solved)
- Foundation is utterly passive in relation to the Project and it's own mission

New Development Process

It's time for change. We have to adapt and evolve.

People

- We need at leats 5 more people in supporting jobs (QA, infrastructure, small coding tasks, assistance etc.) but 10-15 people would be best.
- We need at least one junior core developer.
- We need volunteer(s) that would actively look for new people, resources and sponsors (ideally operating under Foundation's roof).
- Opportunity for individuals or employees of sponsor to help even as part-time job.

Communication

- We need coordinators and "communication officers" for the project itself and for the community overall. Someone who would make sure that important infromation is shared to all concerned.
- We have to find and get used to new methods how to communicate efficiently (maximum gain with minimal effort).

Infrastructure

- We need help with JIRA upgrade, hosting and maintenance.
- We need someone who would focus on looking after infrastructure.

Process

- Constraints and conditions:
  - New development and maintenance done in parallel.
  - Minimal level of required bureaucracy and group decisions, use of clear and simple policies.
  - Use of open and preferably free technical resources.
- Next is an overview of proposal that's currently discused...
- Whole process is structured into continuous sequence of unified iteration cycles of fixed length. Iteration cycle is 6 weeks long (~1.5 month).
- Whole development cycle for new version consists from 10 iterations (~14 months in total) divided into 4 stages (more about that later). It's possible to extend it with few more iterations (for example due to technical reasons - more complex features, increased maintenance etc.).

Iteration templates - a list of task types and assignments necessary for specific iteration purpose:

- regular iteration (new development + maintenance) in several variants (3+3, 2+2/1+1, 3x1+3 etc.)
- RC iteration (bug fixing, preparation for new big cycle)
- initial iteration (research + increased maintenance)
- "emergency" maintenance iteration (as long as necessary)
- "idle" iteration (variable time, for vacations, conferences, "tool sharpening" etc. where we'll focus on other things than development itself)
- others that we would identify as reoccurring
- Allows focusing on single task, individually and collectivelly, long enough to achieve visible results. Both new development and bug fixing require different methodology and approach to coordinated work between developers and supporting staff (QA, doc writers etc.) and may require/use different resources.
- It is easier to keep track of the current status for all participants, as it doesn't require additional information exchanges beteen peers. I.e. we basically know what others are doing at the time.
- It's easier to evaluate progress, and check to what extent our expectations are met.
- Planning is easier, including synchronization of personal schedules (vacations etc.) with group schedule.
- **Initial stage.** 3 iter. Research and specification of technical realization (written skeleton documentation) and initial / prototype implementation. Result is Alpha release.
- **Alpha stage.** 3 iter. Finalization of implementation, documentation, testing, verification of correctness. Result is Beta release.
- **Beta stage.** 3 iter. Stress testing, testing in real world scenarios, public testing, optimization and bug fixing. Result is RC release.
- **Release Candidate stage.** 1 iter. Final public testing and (critical only) bug fixing with 2 week checkpoints. Final planning of new version, preparation for new big development cycle (branching etc.).

Collaboration could be managed mostly via facilities we currently use:

- JIRA tracker: Targets and task records, planning, assigning, centralisation of task-related information, public comments and discussions about task/target.
- Mailing lists: as used now.

- Web. Instructions and guidelines, procedures, rules, links to resources. If possible automatically aggregated status from other sources.
- Callendar. We can use Google calendars or install our own callendar server (there are several open source options). In both cases it's possible to use various calendar clients of individual preference.

Funding

Responsibility of Firebird Foundation - Stabilize income flow that would cover essential grants, plus would create monetary reserves in FF accounts to cover essential expenses for at least 3 months.

- Solution for monetary fluctuations (basket of currencies)
- Revision of grant allocations
- **Active fund-raising**

Thanks for your attention. Questions?

contact: pcisar@ibphoenix.cz www.ibphoenix.com