From low level toolbox to orbit determination: handling users requests in Orekit

Luc Maisonobe

Pascal Parraud
AGENDA

- Orekit history
- Community
- Interacting with users
- Customers users
- Users requests topics
- Conclusion
Orekit history (1/2)

- 2002: inception
  - Orekit intended as a basis for ground segments bids
- 2006: version technically complete
  - Exceeds technical expectations, why not propose it by itself?
- 2008: failure of the commercial approach
  - Nobody wants a critical component to be third party controlled
- 2008: open-source!
  - Permissive license
  - Very good reception by space community
Orekit history (2/2)

- 2008-2011: cathedral development model
  - Development behind closed doors
  - Project teams decides when to release
- Since 2011: bazaar development model
  - Collaborative tools
    - Forge, mailing lists, issue tracker
    - Decentralized Source Code Management system
  - External committers (write access)
- 2012: open governance
  - Meritocratic model inspired by Apache Software Foundation
  - PMC representatives from agencies, academics, private companies
Community

- Changing a commercial failure into successful project
  - Key factor 1: choice of open-source model
    - Apache Software License V2: a permissive license
    - Users get the control back for critical component
  - Key factor 2: community
    - An open-source project without a community is a failed project
    - Bad experience 20 years ago
      - Innovative attitude simulation library (still innovative 20 years later)
      - Operationnaly validated
      - Put it online ... wait for users ... still waiting
The Cathedral and the bazaar

- Analysis of open-source development models
- 19 lessons learned
  - Lesson 7 (the most widely known)
    - Release early, release often. And listen to your customers.
  - Lesson 6 (in chapter “The importance of having users”)
    - Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging
Interacting with users (1/2)

- First phase: point-to-point communication
  - Used in the early years
  - Only static web site was available
  - During cathedral development model phase

- Already some interactions, despite less than ideal situation!
  - Questions (validation, how-to, ...)
  - Features requests
  - Bugs reports
  - Contributions!
• Second phase: collaborative tools
  • Used since 2011
  • Forge, git, mailing lists ...
  • During bazaar development model phase
• More interactions, users becomes actors
  • Questions (validation, how-to, ...), **and answers**
  • Features requests, **and implementations**
  • Bugs reports, **and fixes**
  • External committers!
Customers users

• Special case
  • CS-SI is a private commercial company
  • Customers pay and have requirements
    - They don’t want to see their know-how disclosed
    - They want to have a maximal return on investment
  • Let’s make a deal!
    - Generic classical features contributed to Orekit
    - Value-added development on top remains customer property
  • What is the gain for customers?
    - Benefit from users fixes and enhancements
      • Reduced cost, improved maintenance
    - Always in synch with upstream library
    - Avoid competing contributions that will generate additional costs
Users requests topics

- Stabilizing phase
  - Questions about validation
  - API requests
  - Few features requests
    - Library was already quite rich
    - Users were discovering it and building new basic applications

- Established tools phase
  - High level feature requests
    - Mission-related
    - Often show very good grasp on Orekit internals
Ellipsoid tessellation

- Status in Orekit 7.0
  - Geographic zones, but only for fly-over
  - Sensors zones, but only for point targets
- A feature with multi-users history
  - New user: create tiles on ground
  - New user: compute DOP over a region
  - New user: ground zones visibility in FoV
- All within 2 months!
- Featured prepared for 7.1
  - One external committer expanded on it!
Users are important!

A good community is the best asset of an open-source project