JSatOrb: ISAE-SupAero’s Open-source Software Tool for Teaching Classical Orbital Calculations

Thibault Gateau, Julio Hernanz-Gonzalez, Theo Koudlansky, Lucien Sénaneuch and Patrice Labedan

2018 November 7
Teaching support

- Space Mechanics
- Mission Analysis
- Thermal Architecture
- Space Telecom
- Power system
- AOCS
- Launchers
- Project Management

...
ISAE-SUPAERO: Doing Space Stuff

Research & Support
- Engineering Projects
- Research Projects

Facilities
- UHF/VHF antenna/station
- S-Band antenna/station
- Control Center
- Clean Rooms ...

thibault.gateau@isae.fr
3U Nanosats projects on the way (more than these 3!)

- Entriesat
  - Phase D
- Eyesat
  - Phase D
- Nimph
  - Phase B1
1. ISAE-SUPAERO Software Legacy
   - Satorb
   - Simusat
   - Current Statement

2. JSatorb: a Satorb possible evolution
   - JSatorb: Increasing Modularity, First Attempt
   - JSatorb: Service Oriented Architecture

3. Conclusion & Questions
1. ISAE-SUPAERO Software Legacy
   - Satorb
   - Simusat
   - Current Statement

2. JSatorb: a Satorb possible evolution

3. Conclusion & Questions
ISAE-SUPAERO Space Software Suite

People involved

- Initiator (before 2001!):
  - Christian Colongo

- Current Dev Team:
  - Patrice Labedan
    - Guillaume Garrouste
    - Thibault Gateau

- Lot of support from:
  - Students Projects
  - Internships
  - PhD students
  - Collaborations (TUM)
Functionnalities

[Hernanz-Gonzales 2017]

1. Objects Creation
   - Satellites
   - Ground stations
   - Links

2. Analysis
   - Ephemeris
   - Manoeuvres
   - Coverage
   - Access
   - RF Links
   - Attitude

Intuitive GUI

- Adapted to students
Simulaunch
Kerbal Space Program before Kerbal Space Program...
ISAE-SUPAERO Software Legacy

Pros: Homemade
- Fit exactly to ISAE-SUPAERO needs
- Adapt what we want
- No intellectual property issue
- Short dev cycles

Cons: Homemade
- Not open-source, no community behind
- Costful for internal developers
- Not cross platform
- Validation by hand
ISAE-SUPAERO Software Legacy

JSatorb: a Satorb possible evolution
- JSatorb: Increasing Modularity, First Attempt
- JSatorb: Service Oriented Architecture

Conclusion & Questions
Create the germ of JSatOrb, a version of SatOrb in Java

- Student-focused learning tool
- Professional software for researchers

Implement a clear division in the coding

Calculations

User interface
## Choosing an Astrodynamics Library

<table>
<thead>
<tr>
<th>Feature type</th>
<th>Software Features</th>
<th>SatOrb</th>
<th>OreKit</th>
<th>JAT</th>
<th>TUDAT</th>
<th>CelestLab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visualization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Terminal</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>2D (planisphere)</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>3D (Earth)</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td>Ephemeris</td>
<td>Position/Velocity</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Keplerian Parameters</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Eclipses</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Manoeuvres</td>
<td>Impulse</td>
<td>-</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Continuous</td>
<td>-</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Time</td>
<td>UTC</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>TAI</td>
<td>-</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Julian</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>NORAD</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Propagators</td>
<td>Kepler</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Brouwer</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Mosaif</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>SGP4/SDP4</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Central</td>
<td>-</td>
<td>✔</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Lyddane</td>
<td>-</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Eckstein-Hescher</td>
<td>-</td>
<td>✔</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Cohessy Wiltshire</td>
<td>-</td>
<td>-</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Others</td>
<td>TLE Format</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Coverage</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Constellations</td>
<td>✔</td>
<td>-</td>
<td>✔</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Ground stations</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Links</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Language</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>VisualBasic</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Java</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>C++</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Scilab</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>
JSatorb: Full Java solution? - JavaFX version
What about modularity?

Pros

- Fast to setup
- Fully Java oriented
- Classical testing/validation process
- Known territory...

Cons

- One heavy client
- Fully Java oriented
  - How to adapt it with other libraries?
  - Dependencie to the JVM...
- Still a heavy development process
Why developping “JSatorb”? 

**Satorb**
- Pros: Homemade
  - Fit already with our needs
  - Years of support XP on it
- Cons: Monolithic block
  - Huge validation to do
  - Not cross platform
  - Data flow interface
  - Not standardized

**JSatorb**
- Cons: New Soft
  - Development Effort
  - Architecture evolution
- Pros: Linked to the world
  - Cross-plateform
  - Modularity
  - Long term compatibility
  - Using ”real“ Astrodynmamic libraries
Switching to a Service Oriented Architecture

Web oriented architecture
- Front-end for the user interface
- Back-end for the calculus stuff
Switching to a Service Oriented Architecture

Web oriented architecture
- Front-end for the user interface
- Back-end for the calculus stuff

Advantage
- Modularity (components oriented, user interface separated)
- Portability (browser)
Switching to a Service Oriented Architecture

Web oriented architecture
- Front-end for the user interface
- Back-end for the calculus stuff

Advantage
- Modularity (components oriented, user interface separated)
- Portability (browser)

Limits
- Wide implementation choices to do
- Re-do it all!
What’s a REST API?

REST: REpresentational State Transfer

- Based on HTTP but not a standard:
  - Respect of conventions
  - Good practices

What's a REST API?

REST: REpresentational State Transfer

1. Based on HTTP but **not** a standard:
   - Respect of conventions
   - Good practices

2. Five rules:
   - URI as resource identifier
   - HTTP verbs as operation identifier
   - HTTP response as a resource representation
   - Links as relations between resources
   - A parameter as an authentication token

What’s a REST API?

**Rule 1: URI as ressource identifier**

- `http://mywebsite.com/books`
What’s a REST API?

Rule 1: URI as resource identifier

- http://mywebsite.com/books/comments/87
- http://mywebsite.com/books/87/comments
Our Current Operating Ground Station (SCC)
JSatorb: Increasing Modularity, First Attempt
JSatorb: Service Oriented Architecture

Front-end

- «Web Browser»
- User Interface

Back-end

- «Component»
- Facade / Gateway

- «Component»
- Coverage
- «Component»
- Visibility
- «Component»
- Propagation
- «Component»
- TLE

Text

thibault.gateau@isae.fr

ICATT 2018 - DLR Oberpfaffenhofen
Frontend development: Theo Koudlansky
JSatorb: Typical Use Case Scenario

User through his UI → Gateway → Propagator Services

Add satellite → Ephemerids calculus request → Ephemerids response

Display ephemerids

Add a ground station → Visibility calculus request → Visibility response

Display visibility

Coverage Map Windows

Coverage calculus request → Coverage response

Display Coverage map

Display ephemerids

Ephemerids calculus request

Ephemerids response

Visibility calculus request

Visibility response

Coverage calculus request

Coverage response
1 ISAE-SUPAERO Software Legacy

2 JSatorb: a Satorb possible evolution

3 Conclusion & Questions
Nowadays academics requirements

- Open-Source
- Cross-platform
- Standardized
- Documented

Take home Message

- Still cover **teachings** requirements
- Still allow customization for **research**
- Intercompatibility - Input/Outputs Standardization
- Open-Source under MIT licence
  (https://sourceforge.isae.fr/projects/jsatorb/repository)
Thank you for your attention!

Any question?