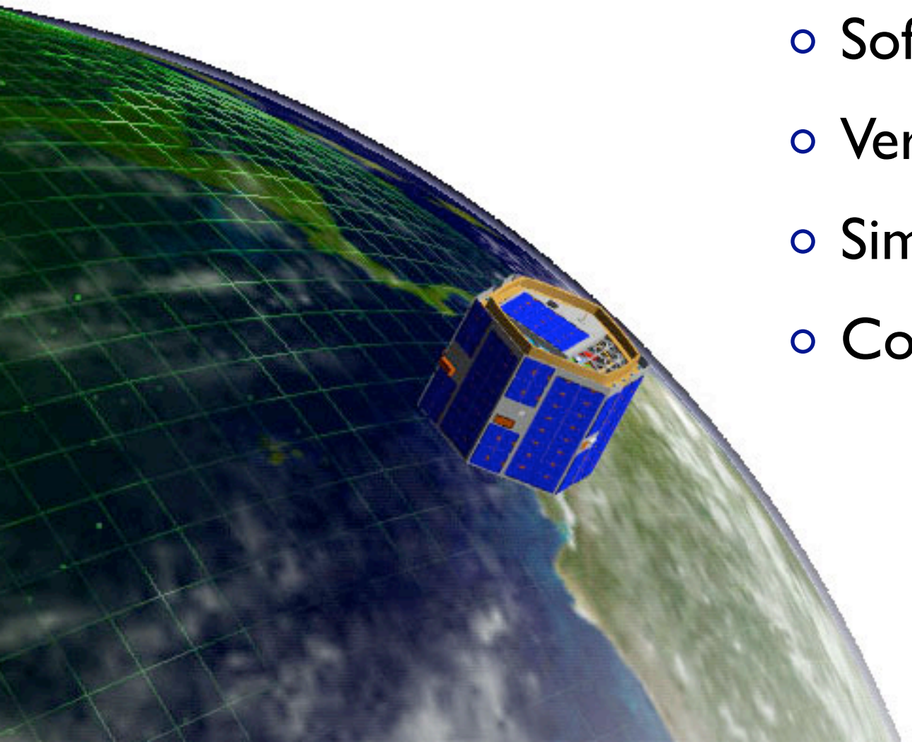
Open-Source, Extensible Spacecraft Simulation And Modeling Environment Framework

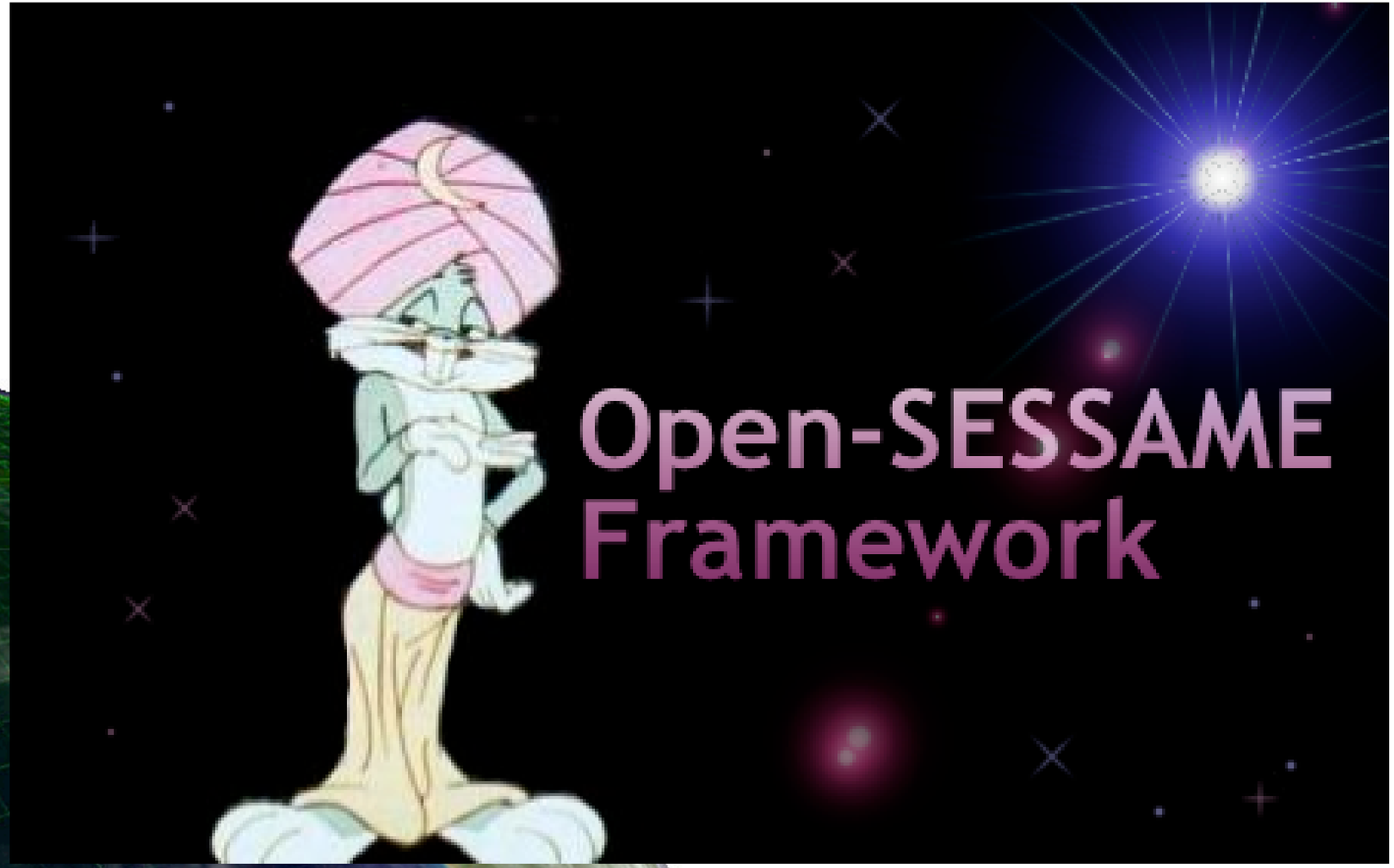
Andrew Turner
Masters Thesis Presentation
July 17th, 2003

Outline of Presentation

- Introduction
- Background
- Purpose
- Architecture Design
- Software Implementation
- Verification and Validation
- Simulation Examples
- Conclusions



Introduction



Adapted by J. Schwartz from http://www.jenn98.com/bugs/1957-1.html

- Open-SESSAME Framework
  - Open-Source
  - Extensible
  - Spacecraft
  - Simulation And Modeling
  - Environment
  - Framework

Section: Background

- Spacecraft Simulation
- Software Alternatives
- Project Background
- Purpose
- Object-Oriented Programming

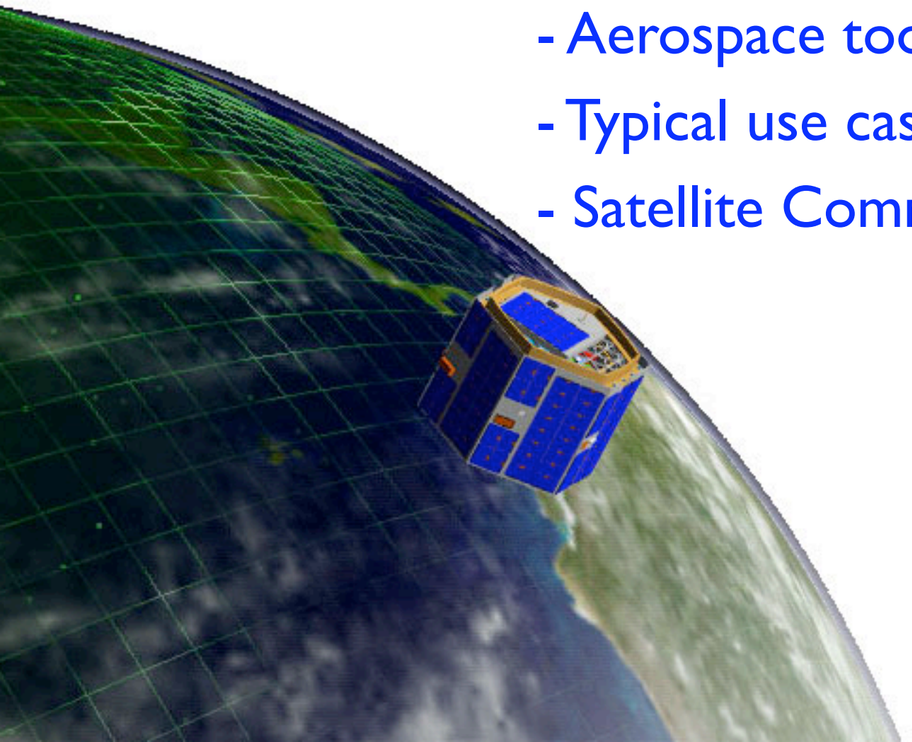
Spacecraft Simulation

Aerospace Engineering

- Attitude dynamics
- Orbit dynamics
- Satellite environment
- Terminology
- Aerospace tools
- Typical use cases
- Satellite Community

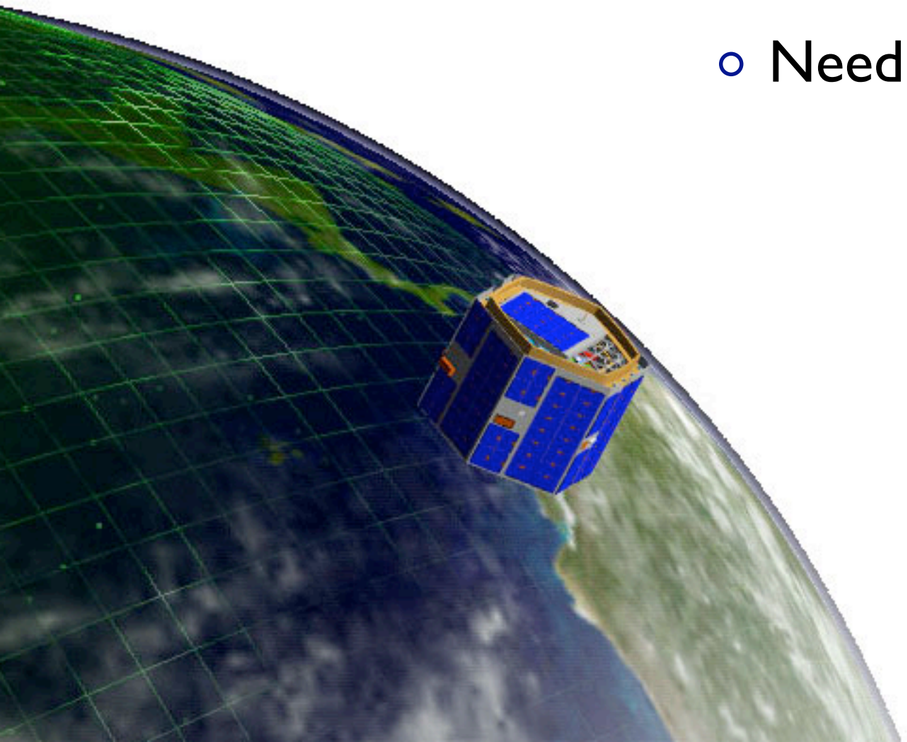
Computer Science / Software Engineering

- Object-Oriented Design
- Computational Speed
- Networking
- Data Handling
- Extensibility
- Software development tools
- Open-Source Community



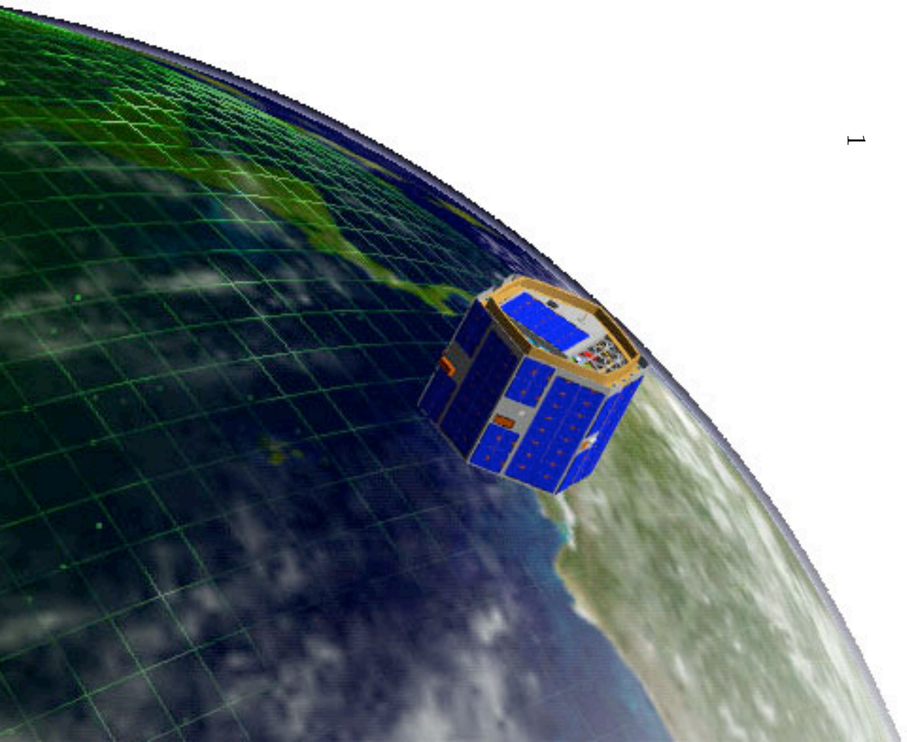
Project Background

- Developing attitude control algorithms
- No current solution
  - Performs necessary tasks
  - Understandable and inspectable
- Need to redevelop solution every time



Software Alternatives

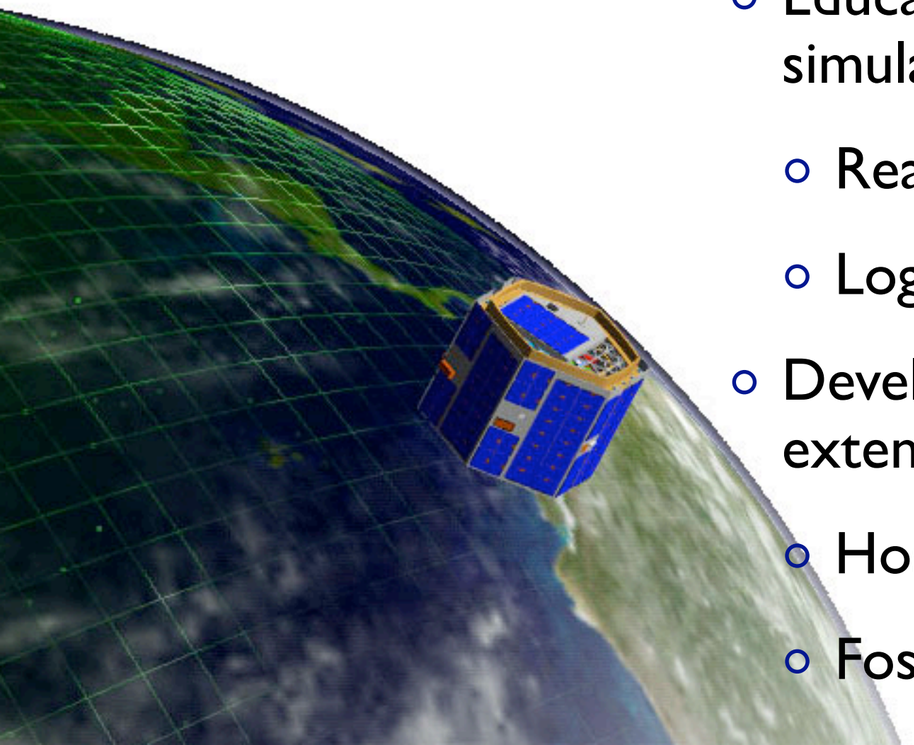
Package	Manufacturer	Benefits	Negatives
AutoCon	NASA Goddard	Heritage	Not maintained, not readily available
DSHELL dshell.jpl.nasa.gov	NASA JPL	Free to academic institutions	Not readily available
Formation Flying Testbed	NASA Goddard	Well supported, ties in with hardware	Expensive, limited availability, aimed towards formations
FreeFlyer www.ai-solutions.com	a.i. solutions	Scriptable, good user interface, heritage, 2D/3D visualization	Expensive, limited integration with existing software
MultiSatSim www.psatellite.com	Princeton Satellite Systems	Good graphics, easy to use interface, scriptable	Apple hardware only, limited to 8 satellites, limited expandability
ORSA orsa.sourceforge.net	Open-Source	multiple platforms, active development	Not complete, limited functionality, orbits only
Open-SESSAME spacecraft.sourceforge.net	Open-Source	multiple platforms, extensible, well documented, active development, orbit and attitude, seperate libraries	No graphical user interface, requires knowledge of C++ programming
SATCOS www.saic.com/space/	SAIC	Heritage	Orbit and constellations only
SaVi savi.sourceforge.net	Open-Source	Good user interface, in development	Only models orbits, made for constellations, single developer
SC Modeler www.avmdynamics.com	AVM Dynamics	-	No longer supported
Spacecraft Control Toolbox www.psatellite.com	Princeton Satellite Systems	Well developed, documented	MatLab only, attitude only
Satellite ToolKit www.stk.com	Analytical Graphics	Easy to use, 2D/3D visualization, heritage, large number of modules, good support, external communcations, variety of operating systems	Expensive, not extensible, complex
Swingby	Computer Sciences Corporation	–	–
WinOrbit www.satnet.com/winorbit/	Carl Gregory, Univ. Illinois	Free of cost, graphical user interface	Windows only, not extensible, user interface difficult



1

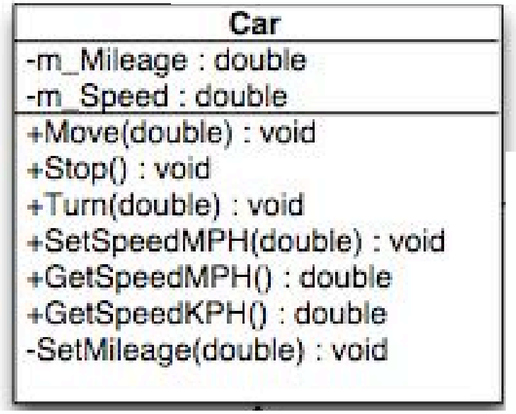
Purpose

- Provide a useful spacecraft simulation modeling tool to engineers
  - Complement existing engineering tools
  - Include underdeveloped methodologies
  - Allow for infinite extensibility
- Educate students and other users how to design and implement simulation in software
  - Readable and useful documentation and software
  - Logical architecture design
- Develop an open-source framework that can be maintained and extended longer than the original project's lifetime
  - Host on a public repository and include developers in the community
  - Foster communication between developers and users



Object-Oriented Programming

- Objects = Classes
  - Encapsulate Physical Objects
  - Abstract Functionality
  - Data-Hiding
- UML Diagram
- Object-Oriented Design
  - Composition
  - Inheritance

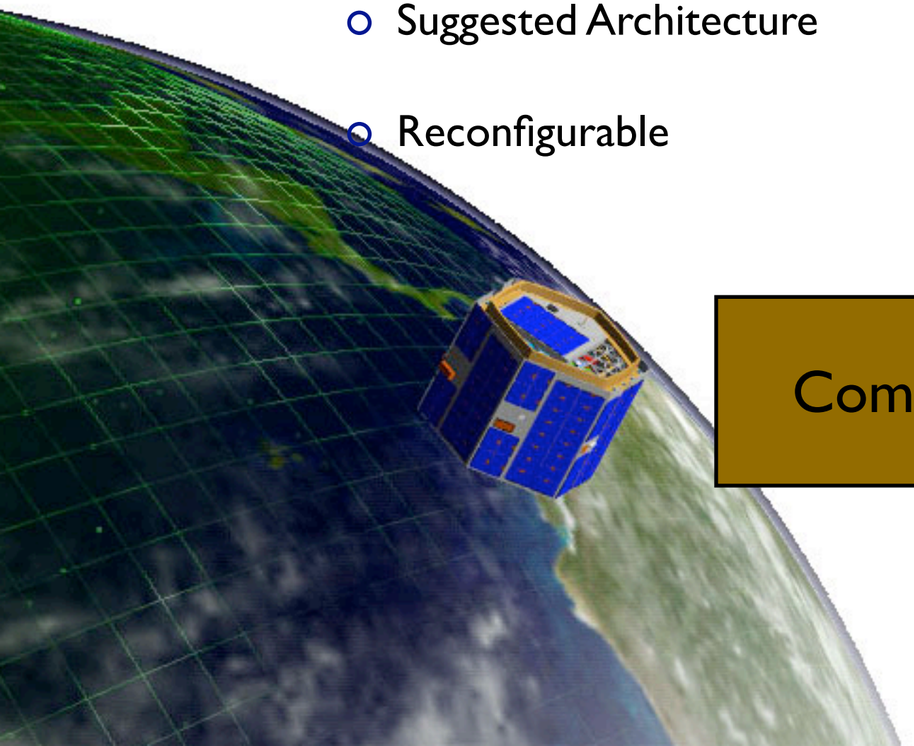


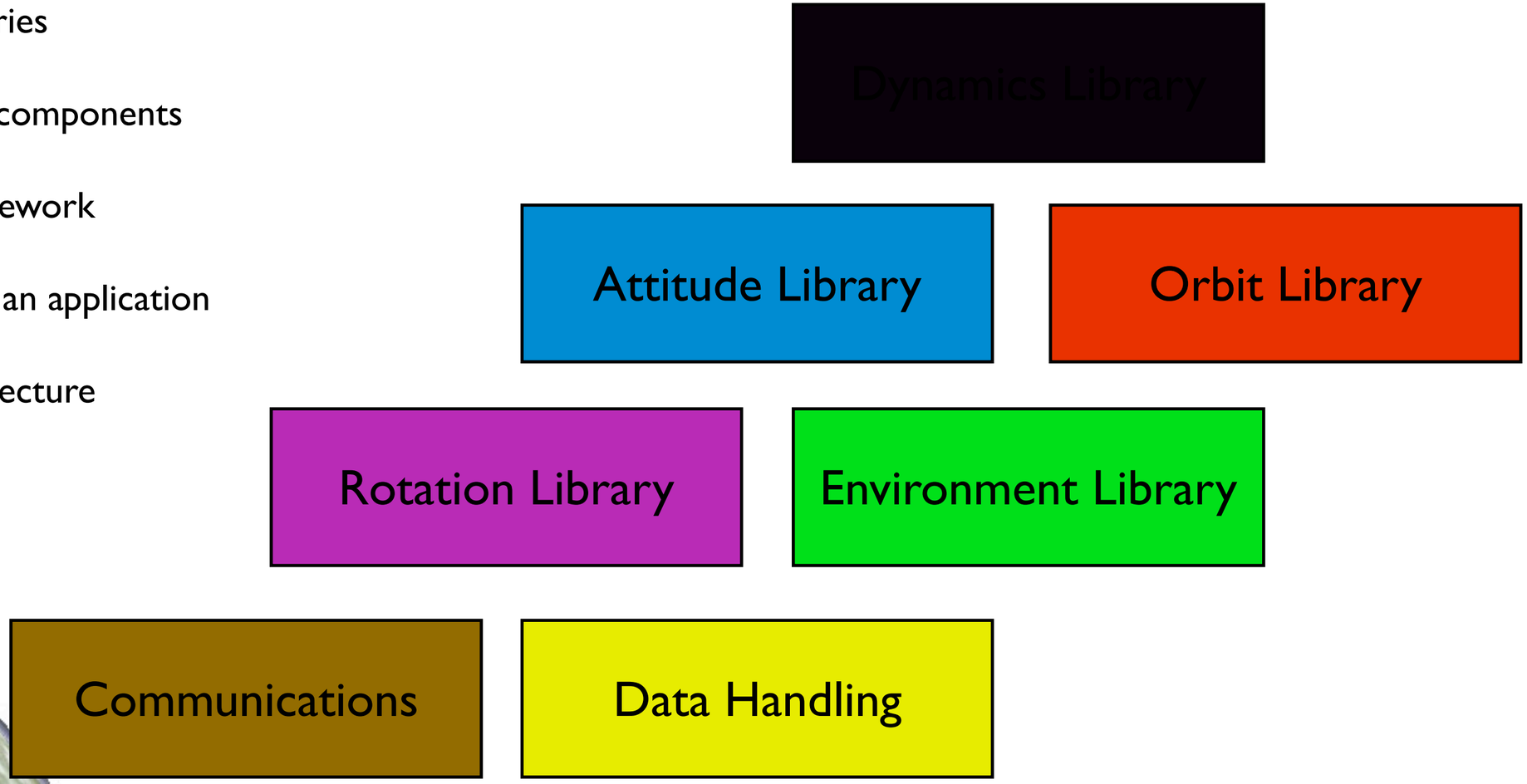
Section: Architecture Design

- Software Architecture
- Component Libraries
- Building an Application

Architecture Design

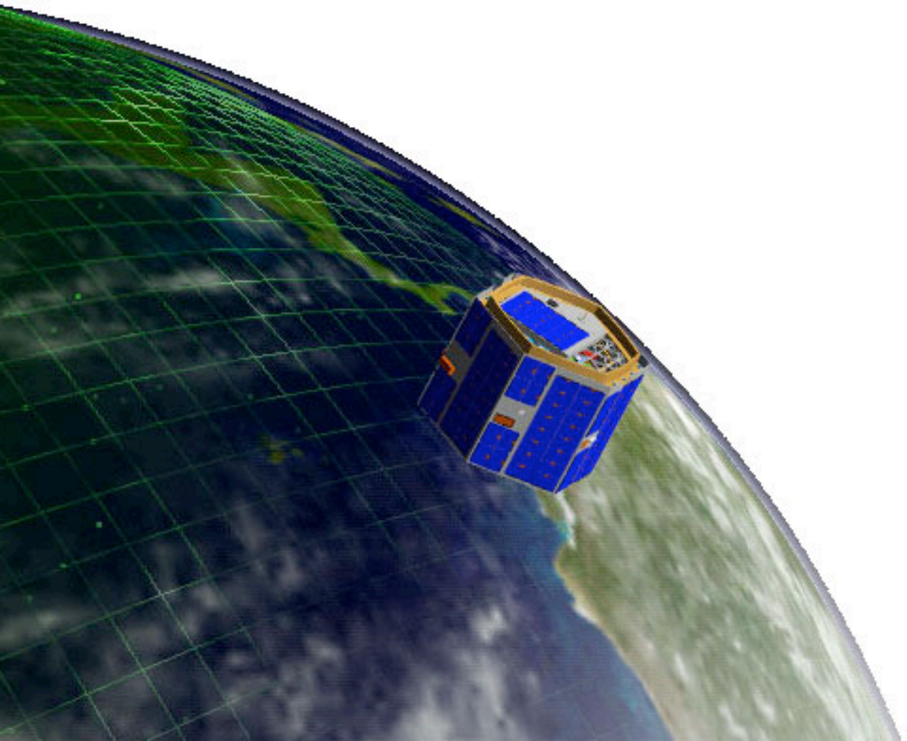
- Collections of libraries
  - Interchangeable components
  - Application framework
- ‘Assemble’ to build an application
  - Suggested Architecture
  - Reconfigurable

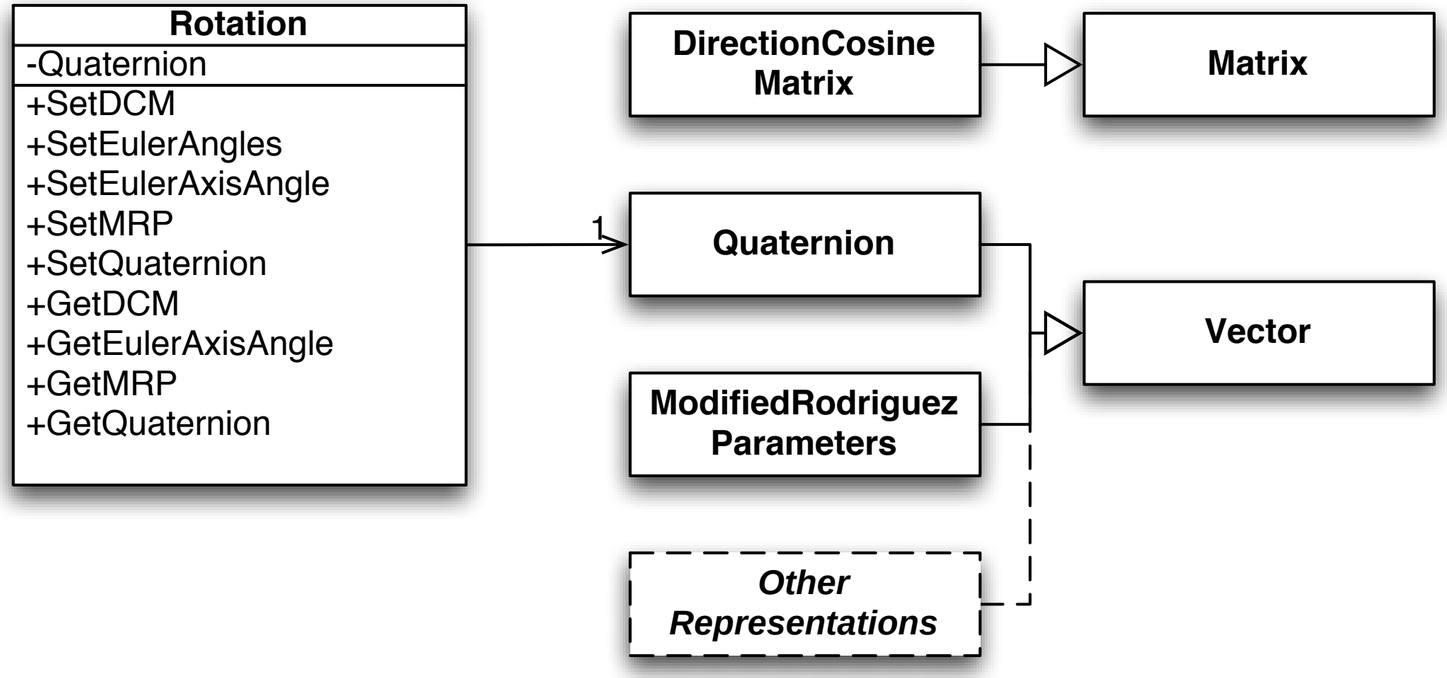




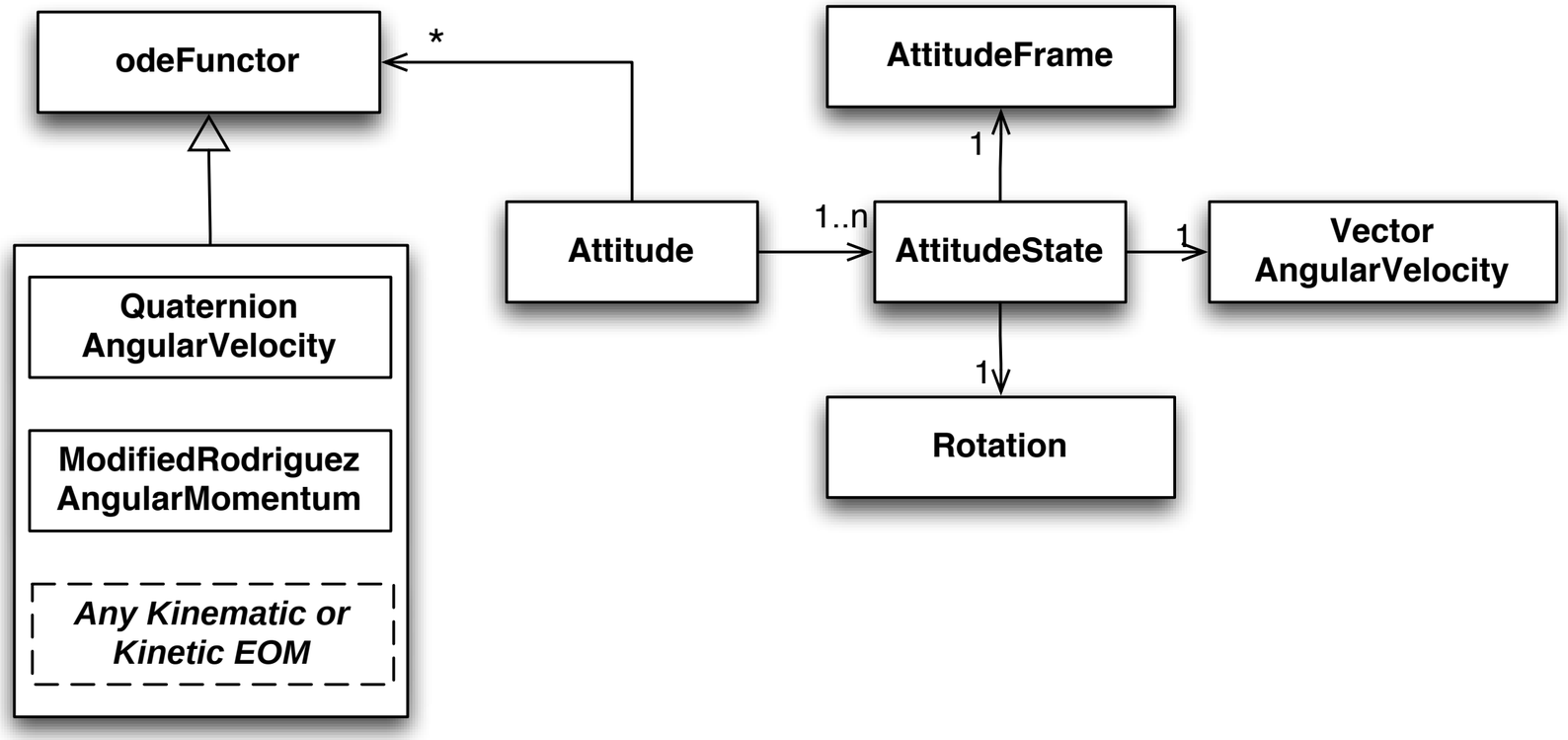
Rotation Library

- **Rotation** Class
  - Abstract
  - Conversion Routines
  - Stored as a **Quaternion**





Attitude Library

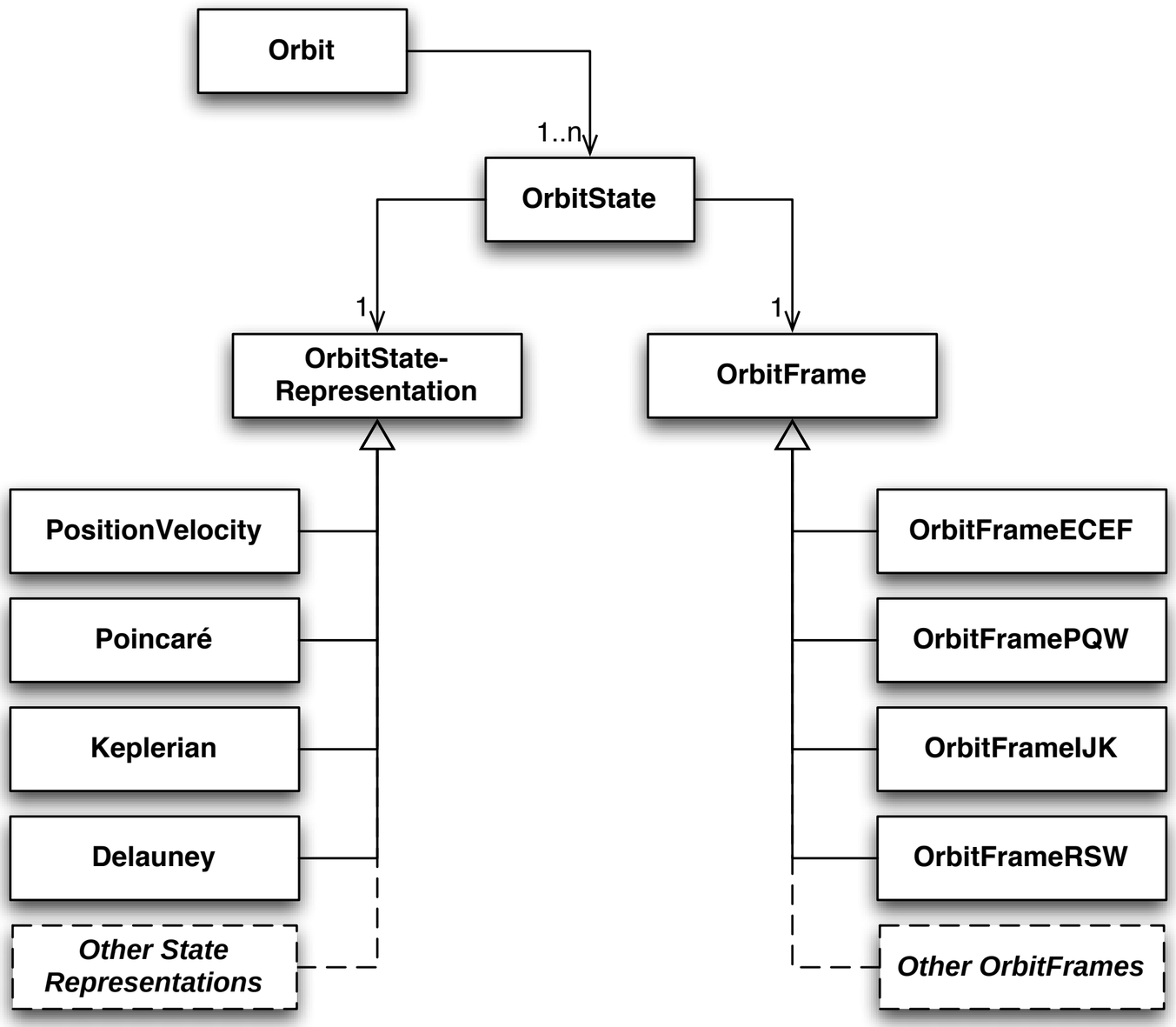


Kinematic & Kinetic
Equations of Motion

- **AttitudeState**
  - Composition
  - Rotation, Frame, AngVel
- **Attitude**
  - Larger composition
  - AttitudeState
  - Kinematics & Dynamics
  - History

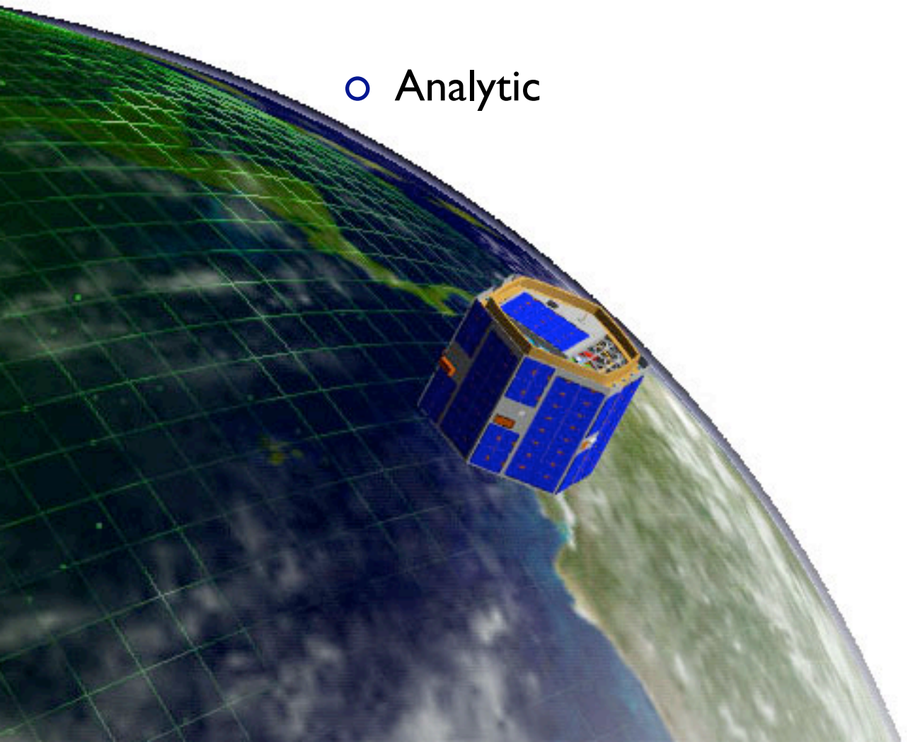
Orbit Library

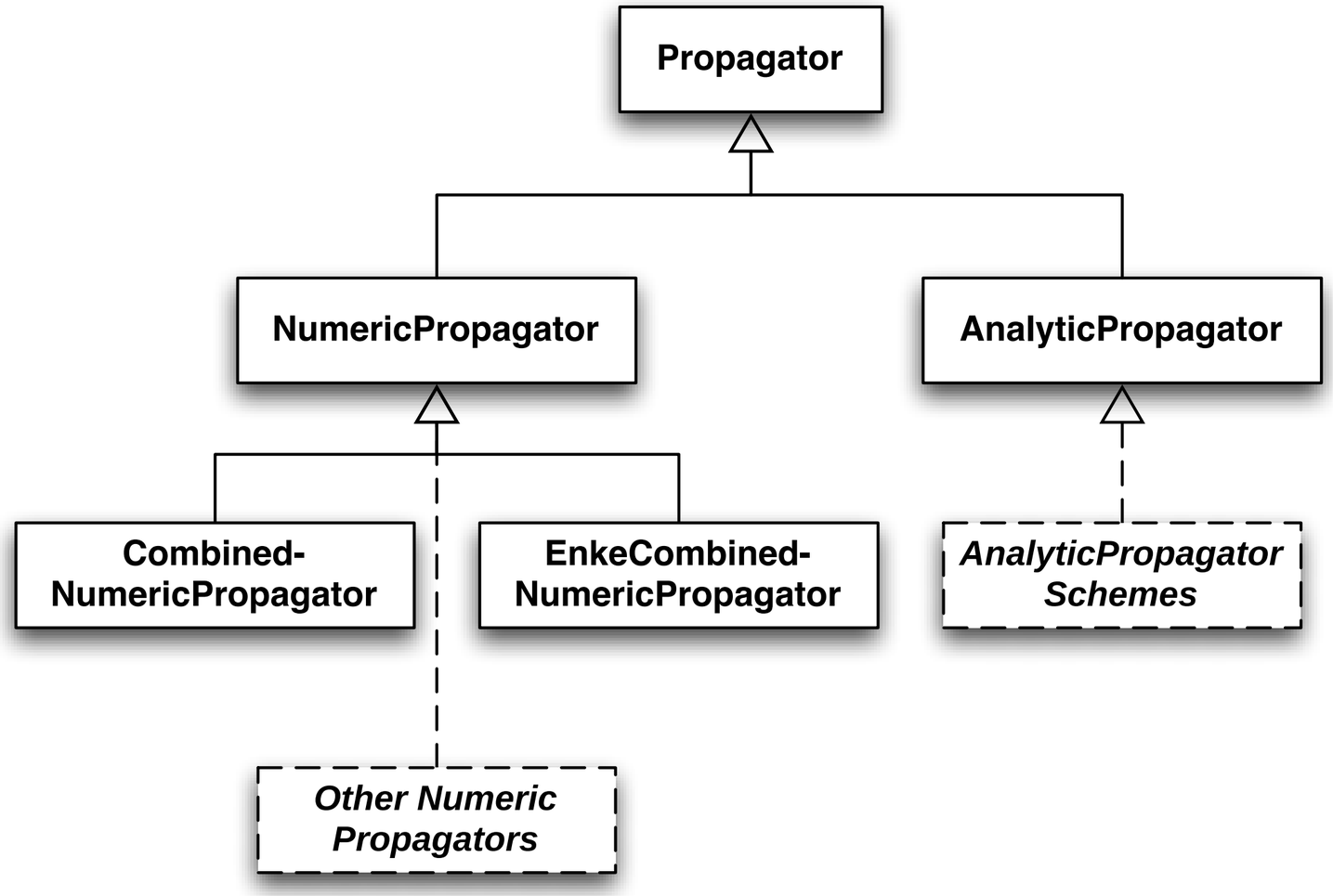
- **OrbitState**
  - Local Composite
  - **OrbitStateRep**
  - **OrbitFrame**
- **Orbit**
  - Larger Composite
  - Dynamics
  - History



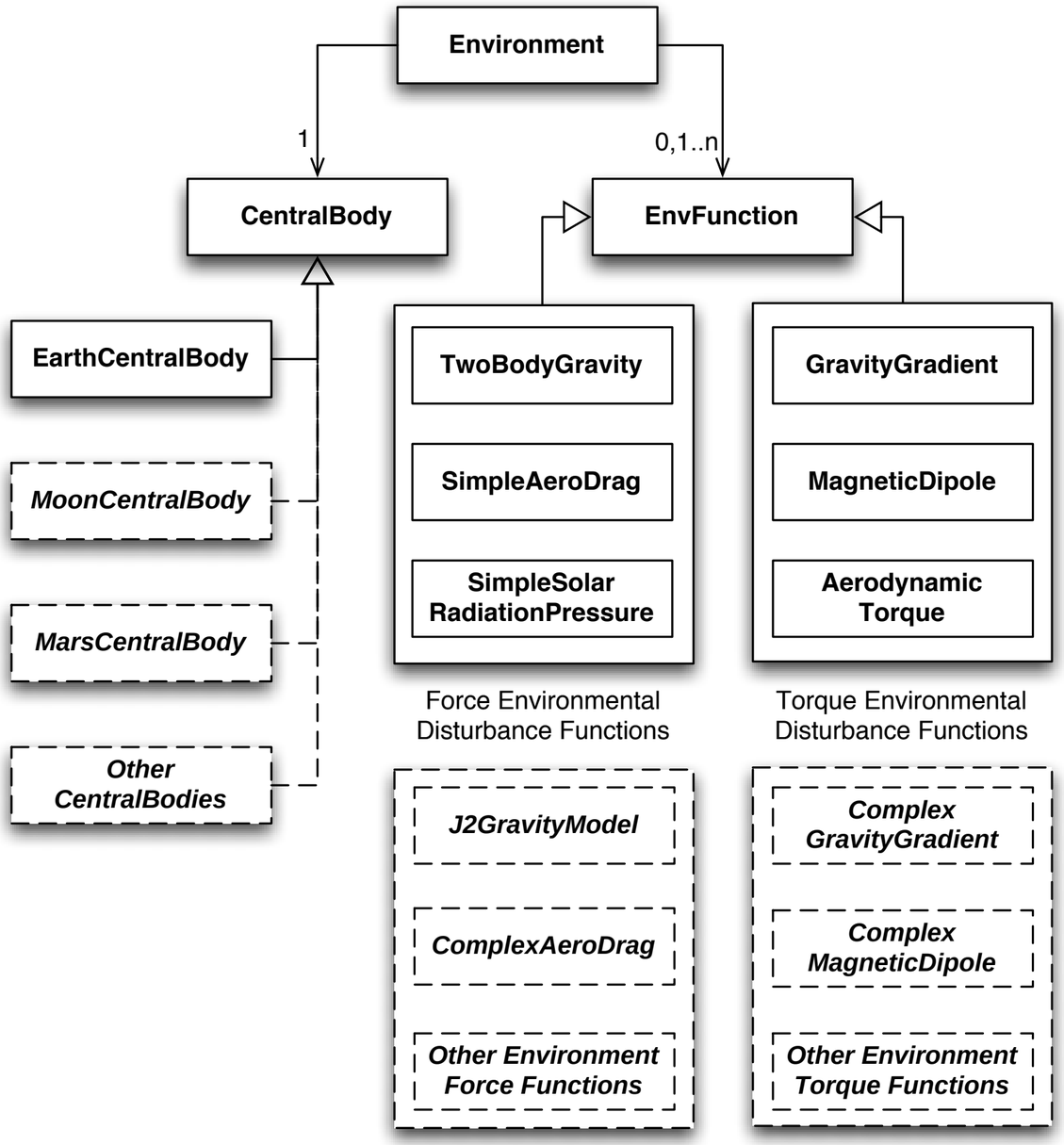
Dynamics Library

- **Propagator**
  - Abstract interface
- Types of propagators
  - Numeric
  - Analytic





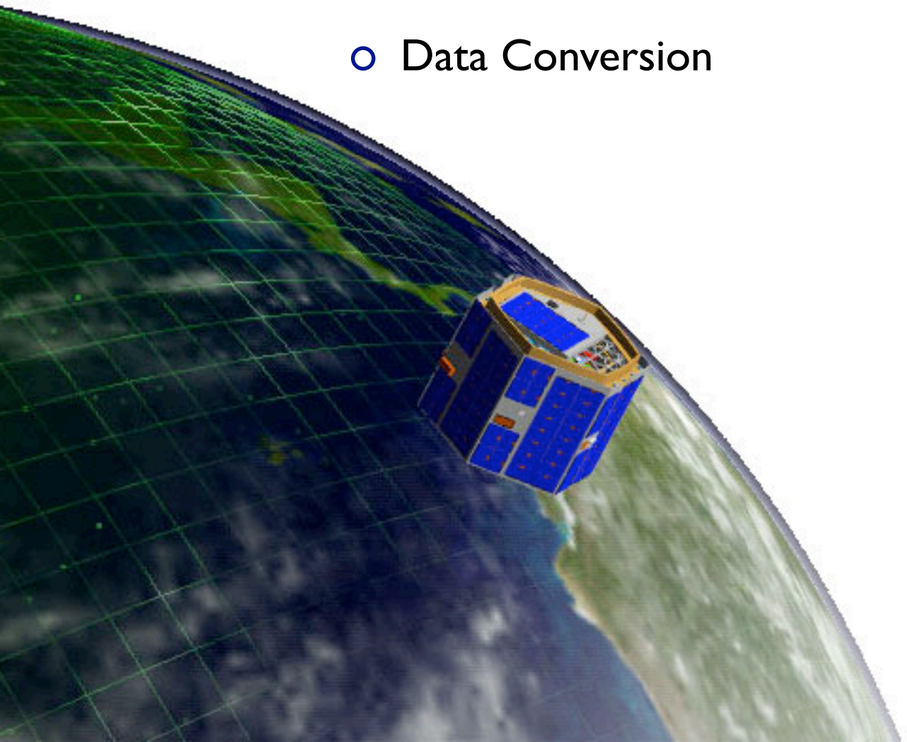
Environment Library

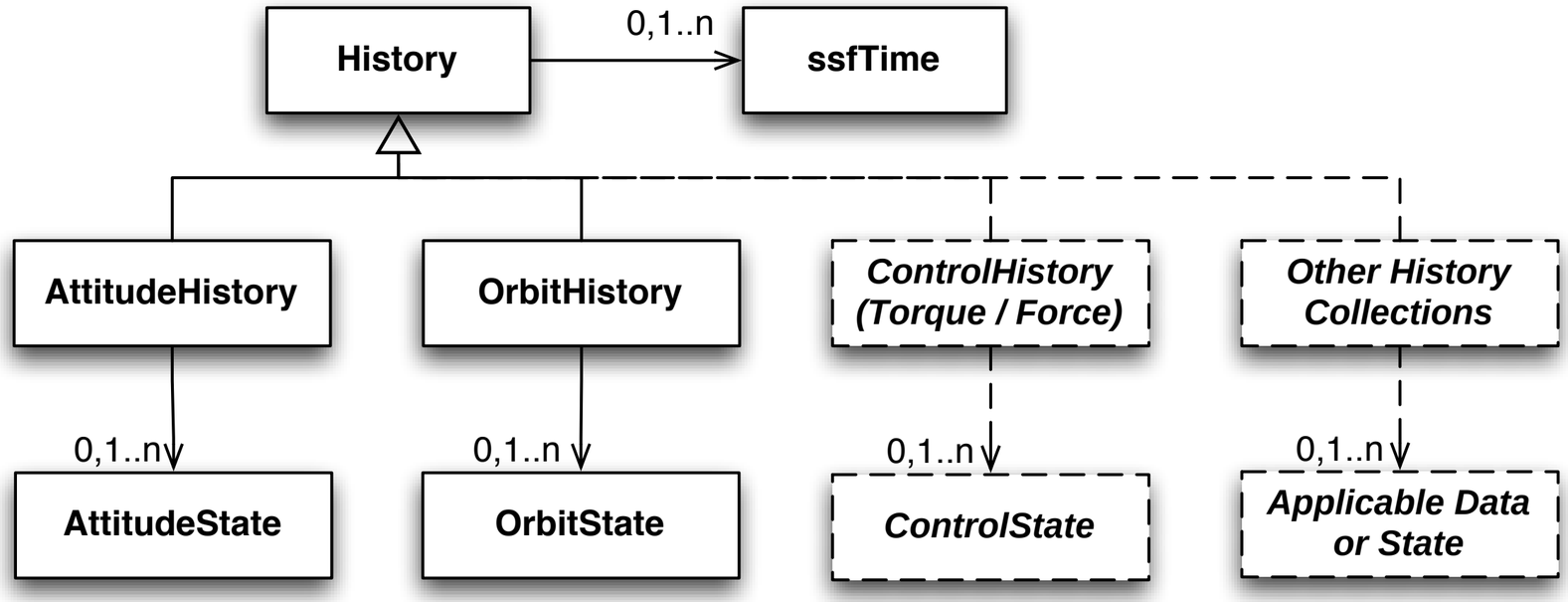


- **Environment**
  - Composition and interface
- **CentralBody**
  - Physical parameters
  - Defines frames
  - Location with respect to other celestial bodies
- **EnvFunction**s
  - Interface to force and torque functions
  - Parameters: time, orbit state, attitude state

Data Handling Utilities

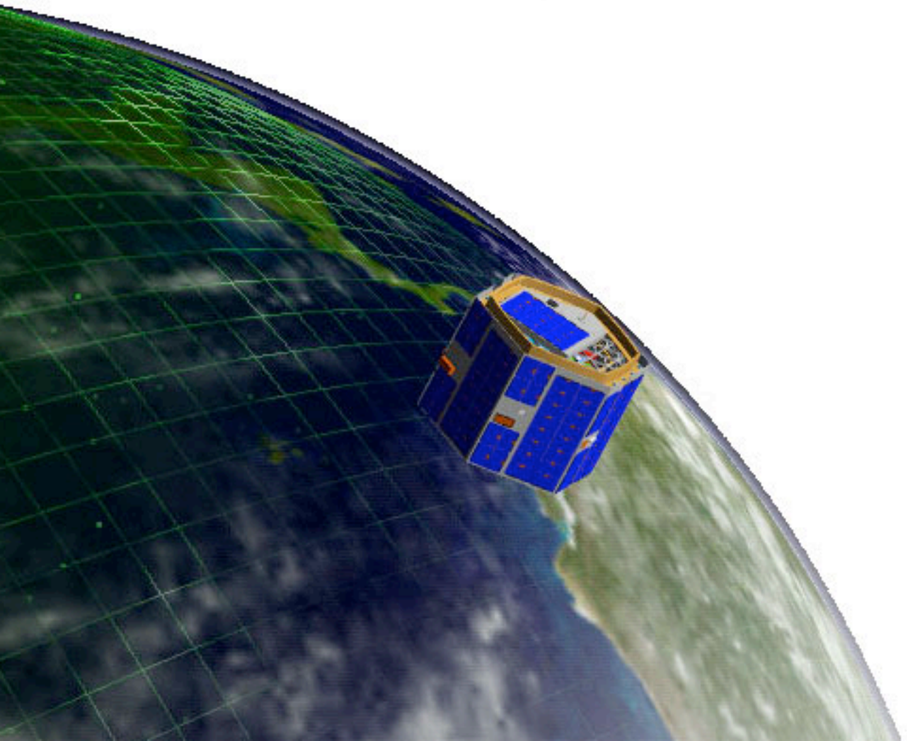
- **History**
  - Vectors of data
  - Associated with times
  - Interpolation
- Data Conversion

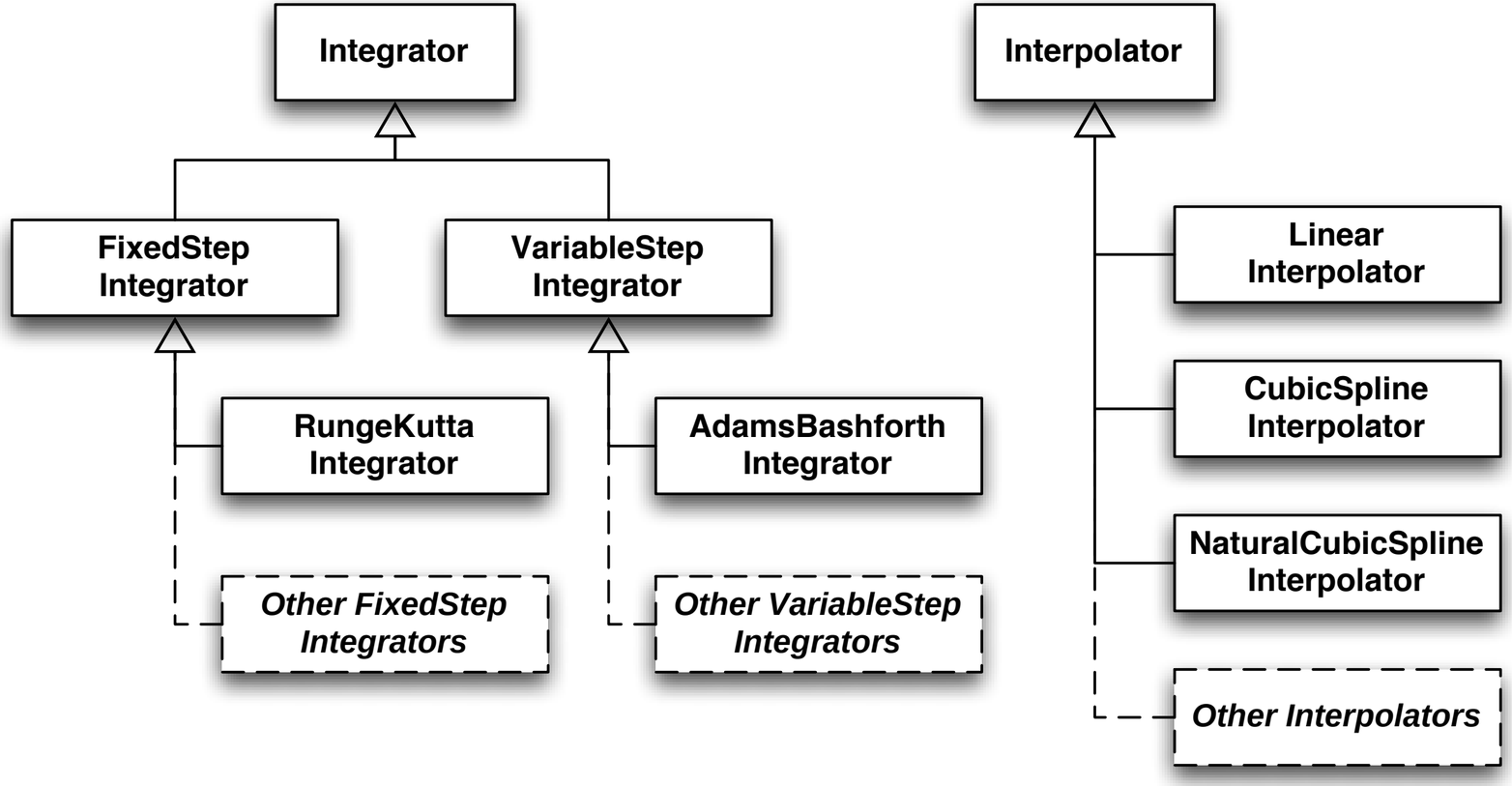




Math Library

- Numerical Integration
  - Fixed step
  - Variable step
- Data Interpolation





Open-SESSAME Application

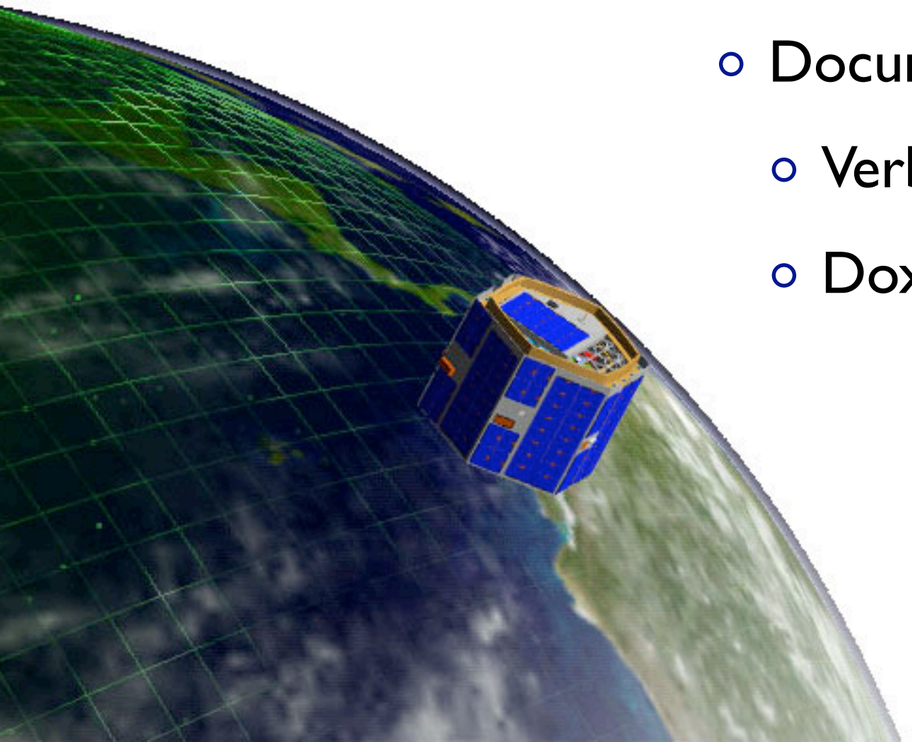
1. Choose Equations of Motion
2. Add Dynamic Components
3. Choose Orbit Characteristics
4. Choose Attitude Characteristics
5. Add Environment Effects
6. Add History
7. Choose Interpolators
8. Simulate!

Section: Software Implementation

- Software Implementation
- Code Metrics
- Implementation Artifacts
- Documentation

Software Implementation

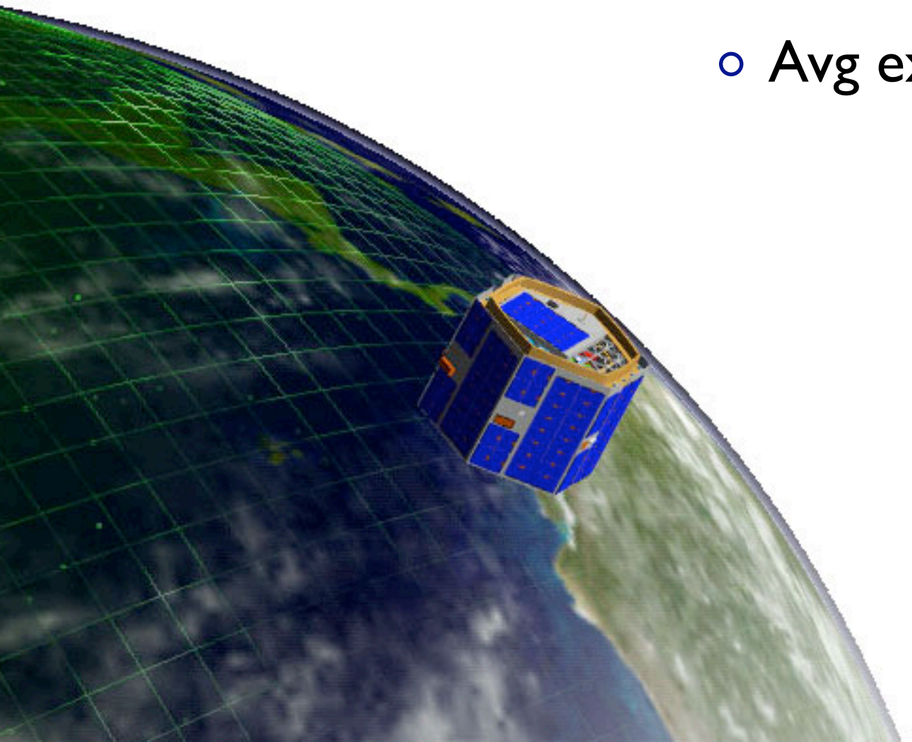
- C++ software language
- Open-source libraries and toolkits
  - Matrix
  - Network Communications
- Documentation
  - Verbose coding
  - Doxygen



Code Metrics

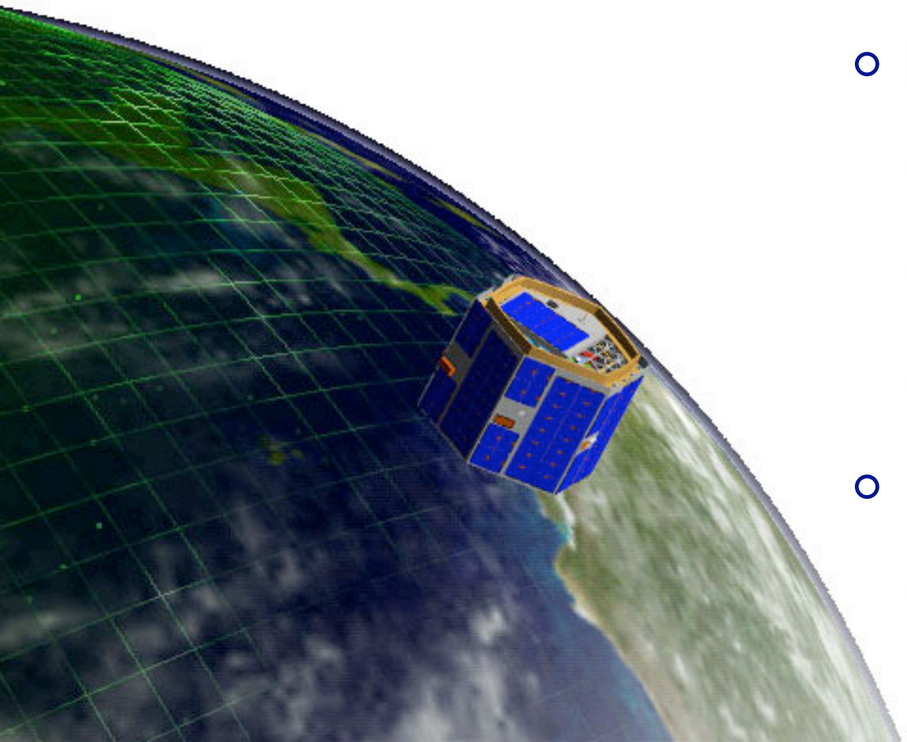
- Lines of code: 4810
- Lines of documentation: 8112
- Number of classes: 44
- Functions: 498
- Avg executable size: ~4 MB

Library	Size
Attitude	335k
Data Handling	749k
Dynamics	697k
Environment	518k
Math	430k
Matrix	1.88M
Orbit	658k
Rotation	255k
Utils	223k



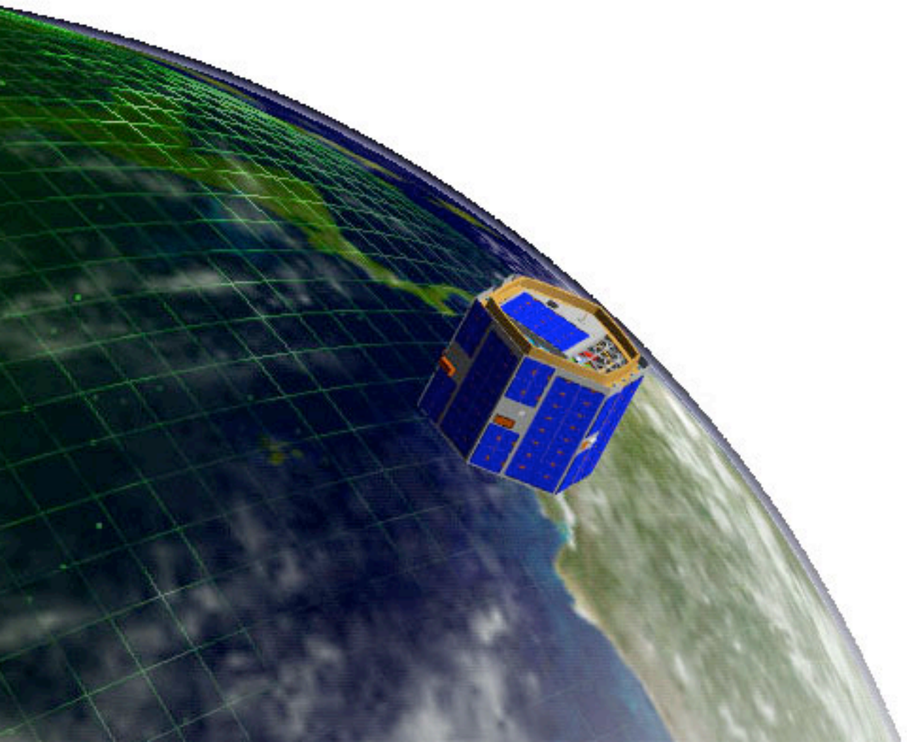
Implementation Artifacts

- Framework bundle
  - 9 libraries
  - Source code
  - Example applications
- Documentation
  - User's Manual
  - Master's Thesis
  - Conference paper
- Public repository
  - Forums, documentation, Source (CVS)



Documentation

- Doxygen
  - Automatic document generation
  - Produces html, xml, latex (pdf)
  - Includes equations
- Verbose coding
  - ‘Hungarian Notation’
  - Inline comments
- Cite references
- Programming primer

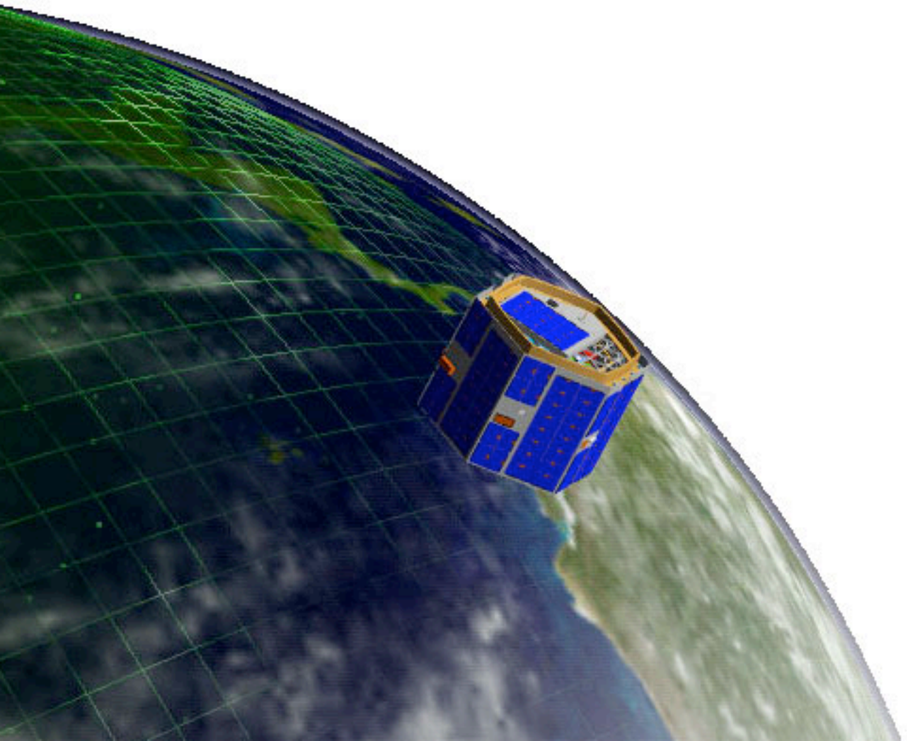


Section: Verification and Validation

- Verification
  - Purpose
  - Methods
  - Results
- Validation
  - Purpose
  - Methods
  - Results

Verification

- Show that the software is implemented correctly - *Verify*
  - Computational results from software match analytical analysis
  - Output is compared to alternative, industry accepted, solutions



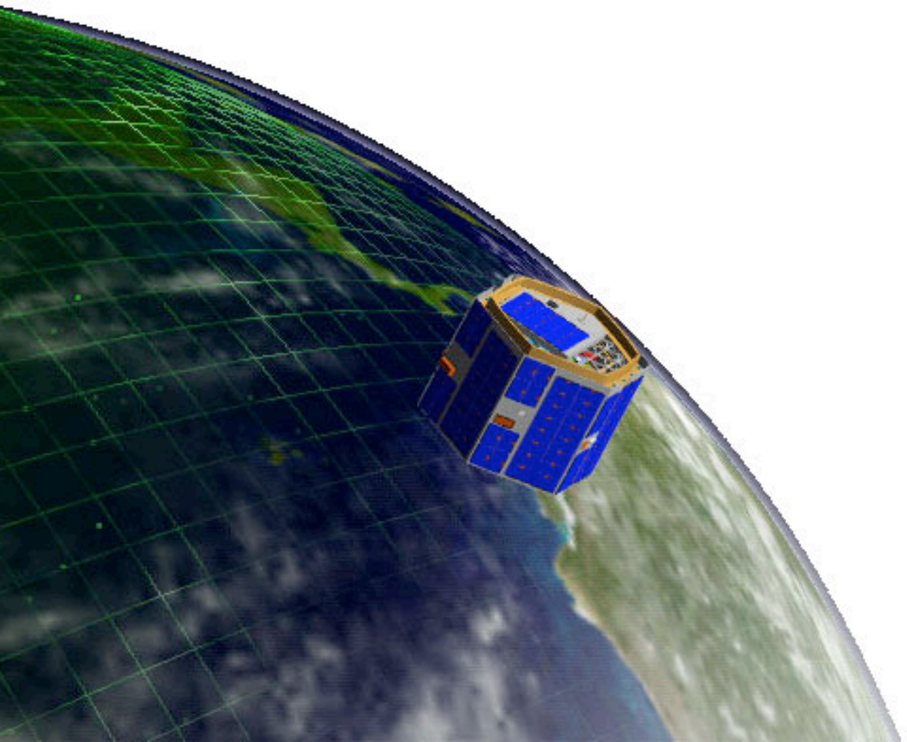
Orbit Verification

Conservation of Angular Momentum

$$\vec{\mathbf{h}} = \vec{\mathbf{r}} \times \vec{\mathbf{v}} = constant$$

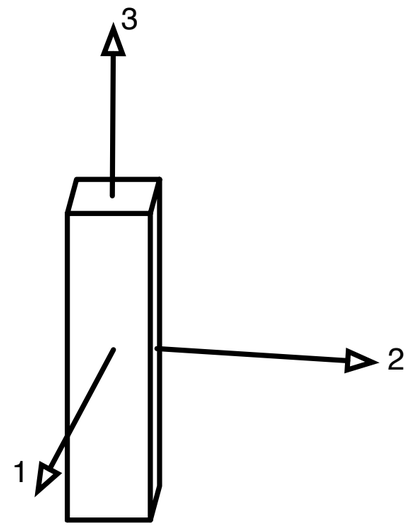
Keplerian Equation

$\mathbf{r}_0^i$	$[1131.34, -2282.343, 6672.423]^T km$
$\mathbf{v}_0^i$	$[-5.64305, 4.30333, 2.42879]^T km/s$
t	2400 sec
Integrator	RK-4(5), tol: 10×10^{-9}
Output	
Running Time	0.59 sec
$\mathbf{r}_{Kep}^i(t)$	$[-4219.853, 4363.116, -3958.789]^T km$
$\mathbf{v}_{Kep}^i(t)$	$[3.689736, -1.91662, -6.112528]^T km/s$
$\mathbf{r}_{Sim}^i(t)$	$[-4219.752, 4363.029, -3958.766]^T km$
$\mathbf{v}_{Sim}^i(t)$	$[3.689865, -1.916734, -6.112511]^T km/s$
RMS error	$\Delta\mathbf{r}_{RMS} = 3.1493 \times 10^{-5}$
	$\Delta\mathbf{v}_{RMS} = 6.95115 \times 10^{-5}$

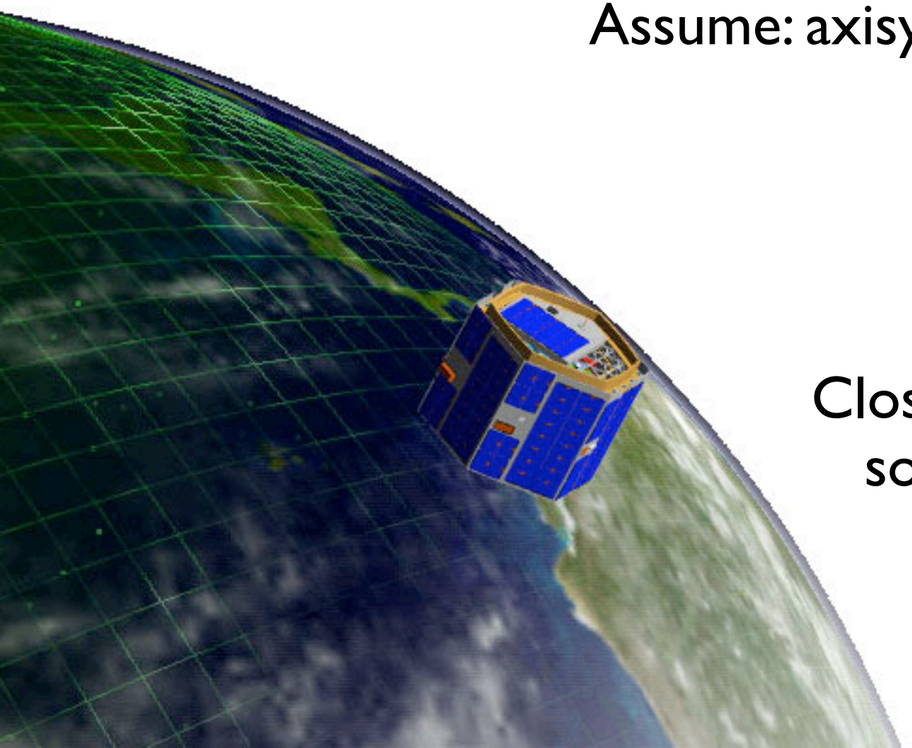


Closed-Form Attitude Solution

Euler's Equations: $\mathbf{I}\dot{\omega} = \mathbf{g} - \omega \times \mathbf{I}\omega$



Assume: axisymmetric $I_T = I_1 = I_2$



Closed-form solution:

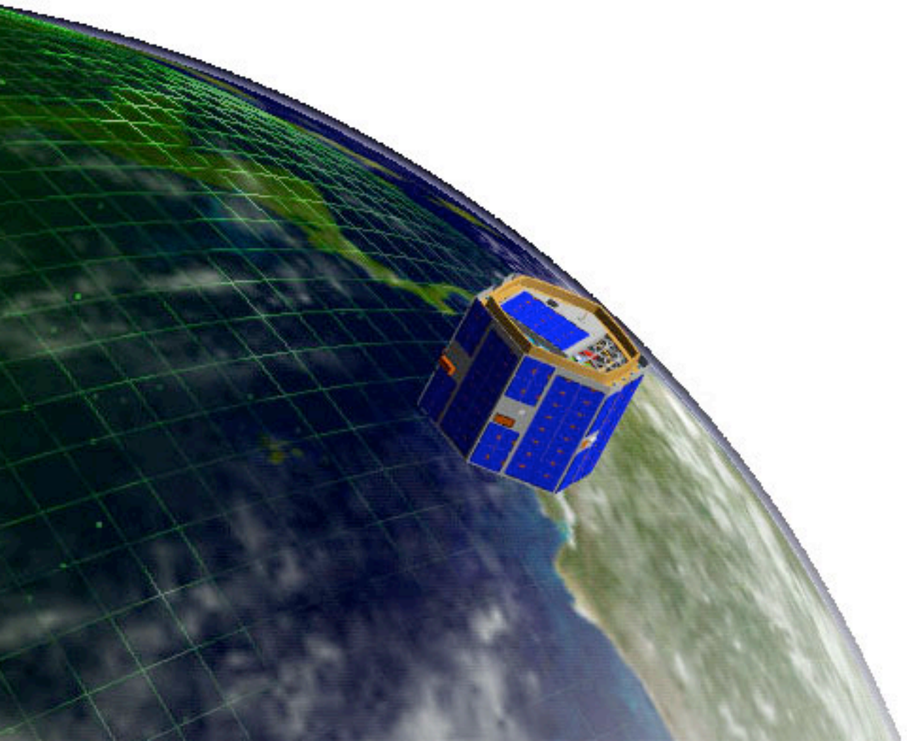
$$
\begin{aligned}
\omega_1 &= \omega_{01}\cos\omega_p t + \omega_{02}\sin\omega_p t \\
&\quad + \frac{1}{I_T}\int_0^t \left[g_1(\tau)\cos\left(\omega_p(t-\tau)\right) + g_2(\tau)\sin\left(\omega_p(t-\tau)\right)\right]d\tau \\
\omega_2 &= -\omega_{01}\sin\omega_p t + \omega_{02}\cos\omega_p t \\
&\quad + \frac{1}{I_T}\int_0^t \left[-g_1(\tau)\sin\left(\omega_p(t-\tau)\right) + g_2(\tau)\cos\left(\omega_p(t-\tau)\right)\right]d\tau \\
\omega_3 &= \omega_{03}
\end{aligned}
$$

Attitude Verification

ω_0	$[0.3, -0.4, 0.7]^T rad/s$
I_t	100 kg m^2
I_3	150 kg m^2
$g(t)$	$\begin{bmatrix} 0.2\sin t & -0.4\sin t & 0 \end{bmatrix}^T Nm$
t	300 sec
Integrator	RK-4(5), tol: 10×10^{-8}
Output	
$\omega(t)$	$\begin{bmatrix} -0.46134984 & -0.194318263 & 0.7 \end{bmatrix}^T rad/s$
ω_{Sim}	$\begin{bmatrix} -0.46134984100 & -0.193182629296 & 0.7 \end{bmatrix}^T rad/s$
RMS error	$\sqrt{(-1.15712 \times 10^{-9})^2 + (2.942222 \times 10^{-9})^2}$ $= 3.165158 \times 10^{-9} rad/s$

Validation

- Show that the correct software is implemented - *Validate*
  - Does the code meet requirements of use?
  - Does the implementation improve upon existing solutions?
  - Can the software easily demonstrate its purpose?

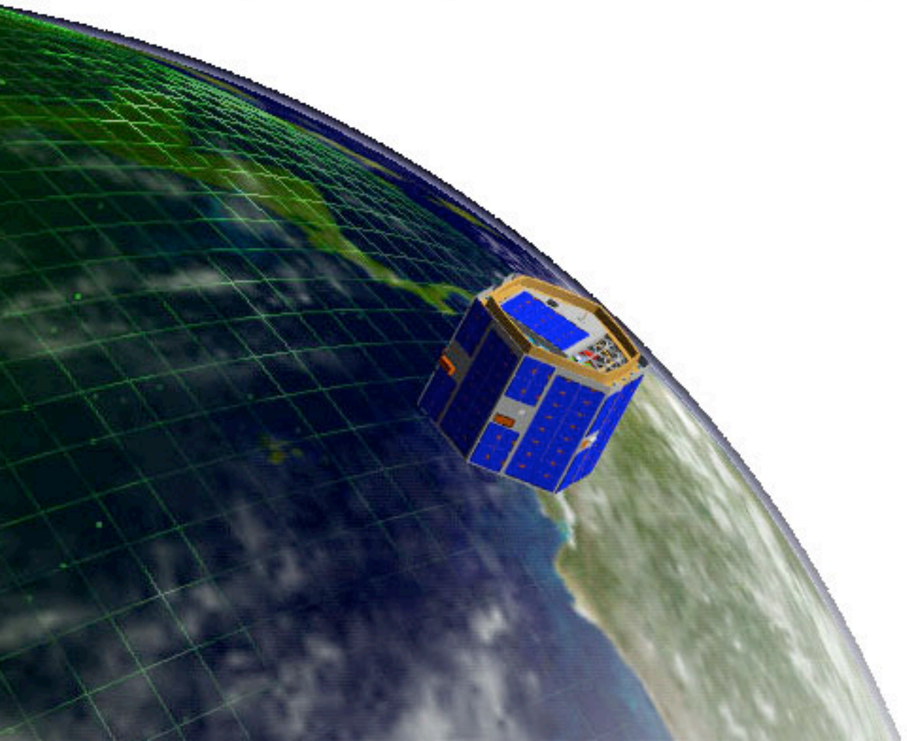


Section: Simulation Examples

- Attitude Integration
- Orbit Integration
- Coupled Simulation
- Hardware-in-the-loop
  - HokieSat Orbit/Attitude Coupled Simulation Server
  - HOACSS

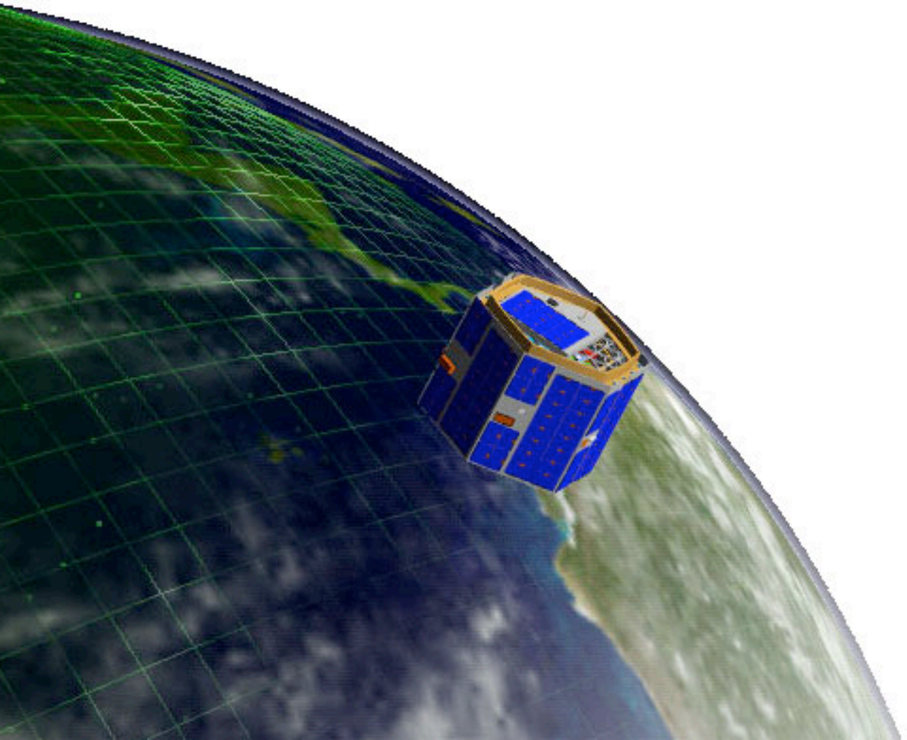
Attitude Integration

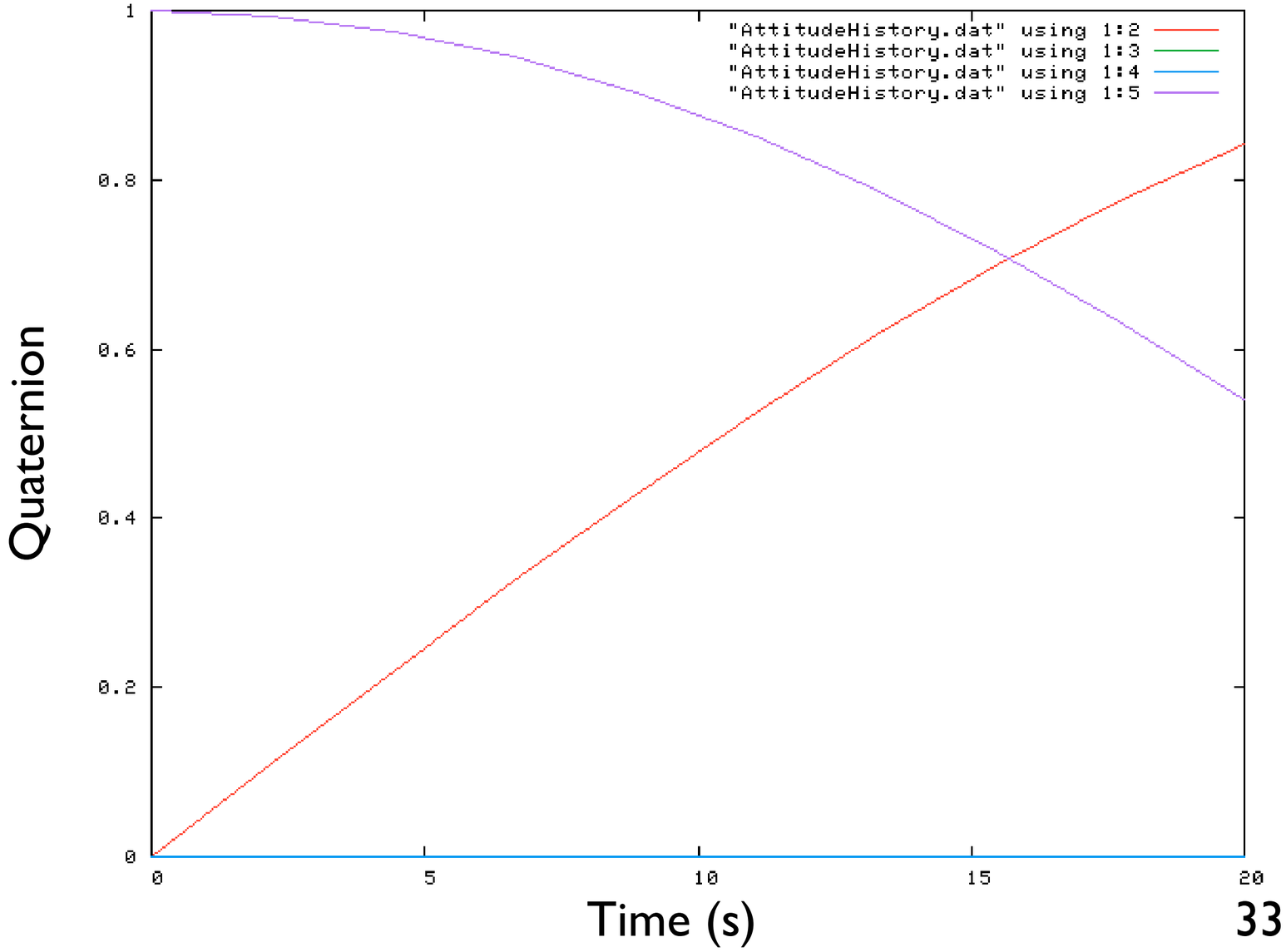
1. Code the attitude dynamics equation
2. Code the disturbance torque function
3. Assign function parameters
4. Create an initial attitude state
5. Create and initialize integrator
6. Integrate the equations
7. Graph or output the state history



Attitude Simulation Output

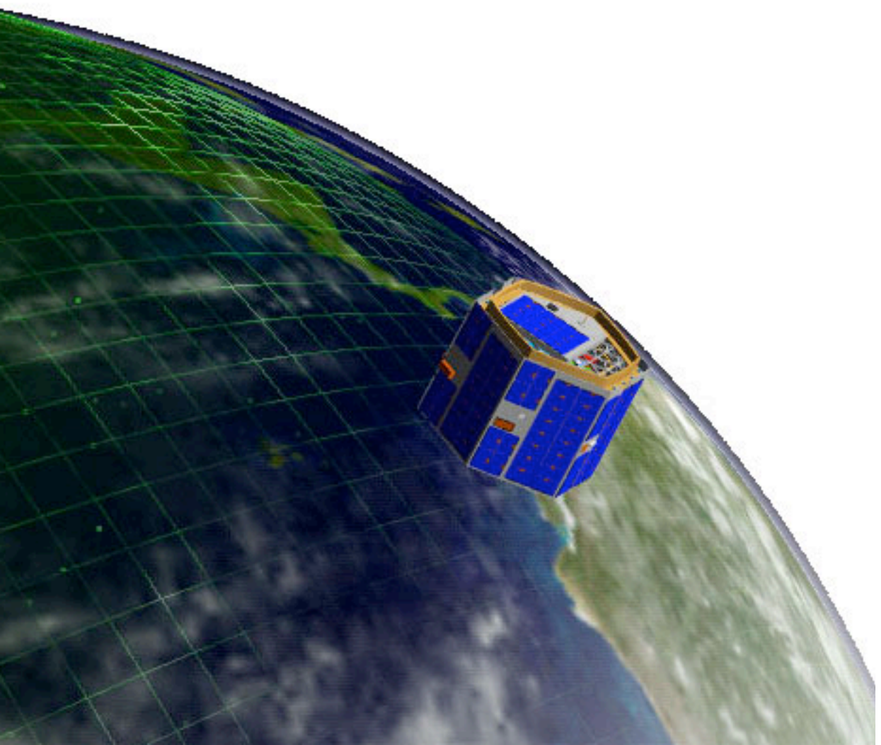
- Gnuplot
- Save/Export
- Dynamics display

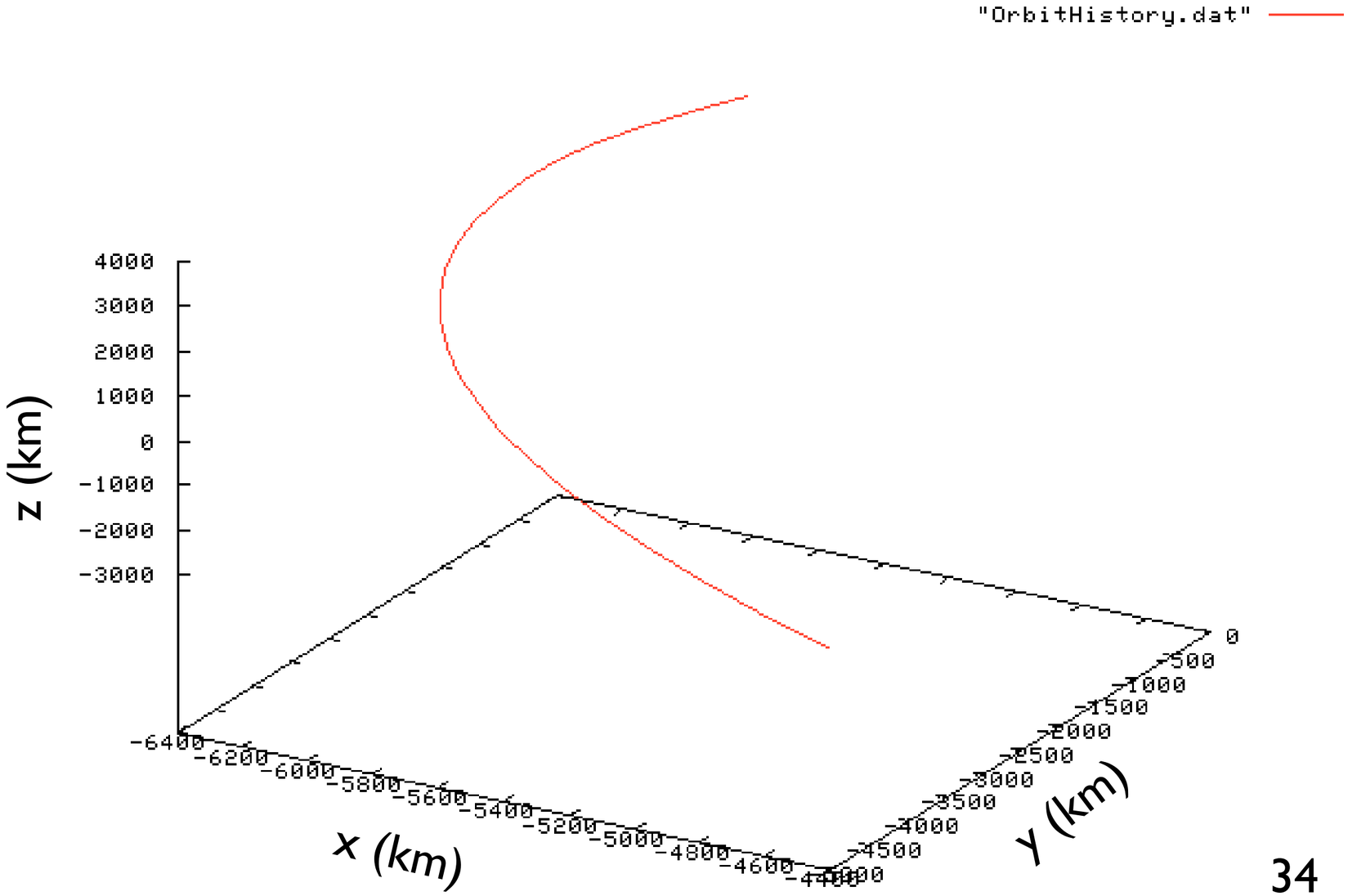




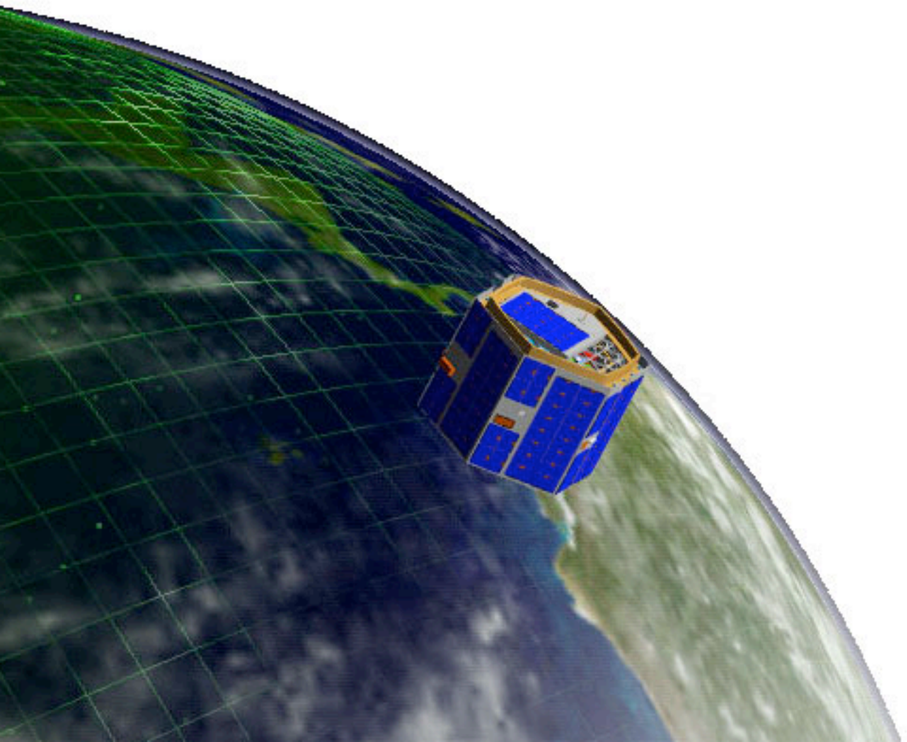
Orbit Simulation Output

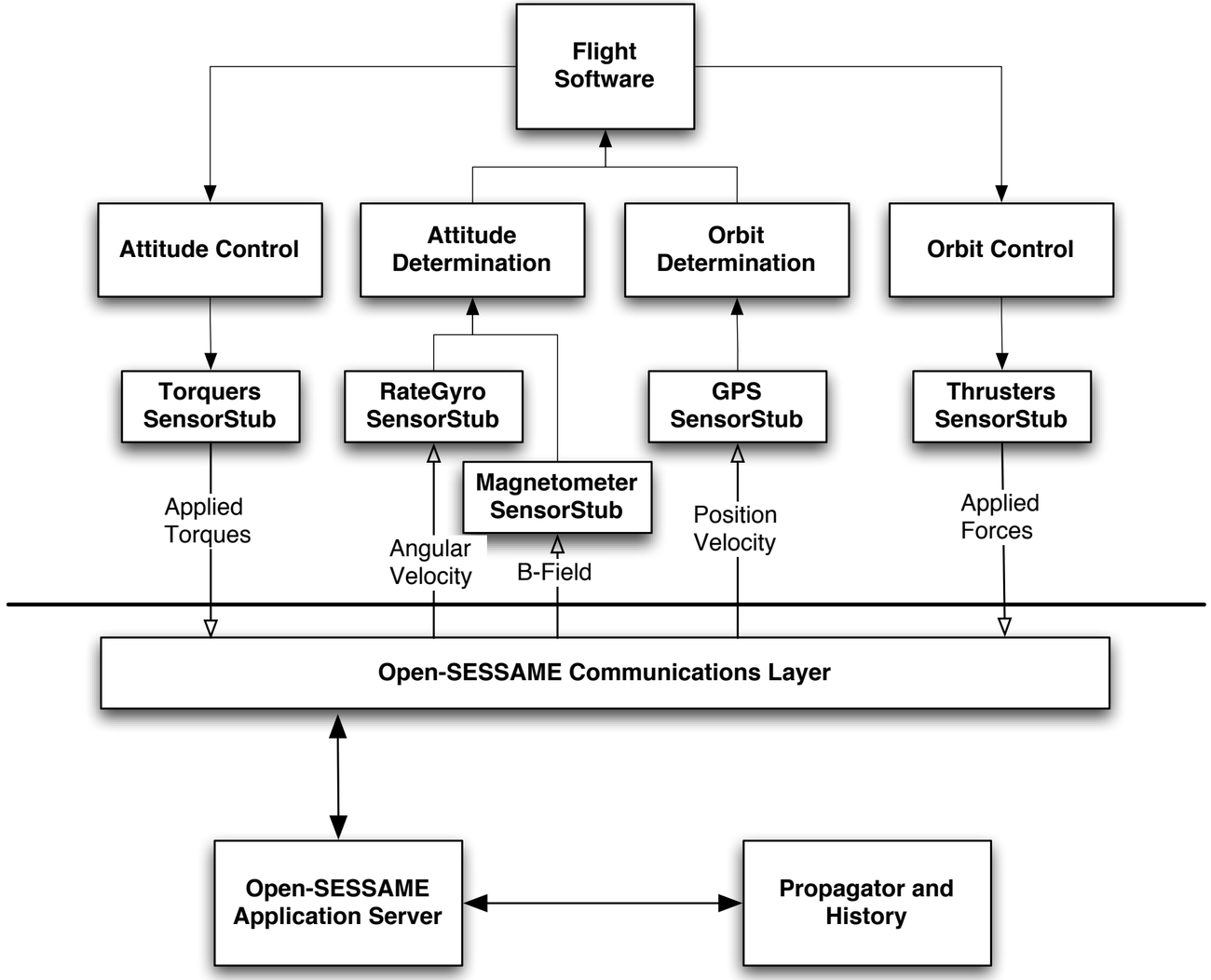
- Similar application construction
- Output orbit position





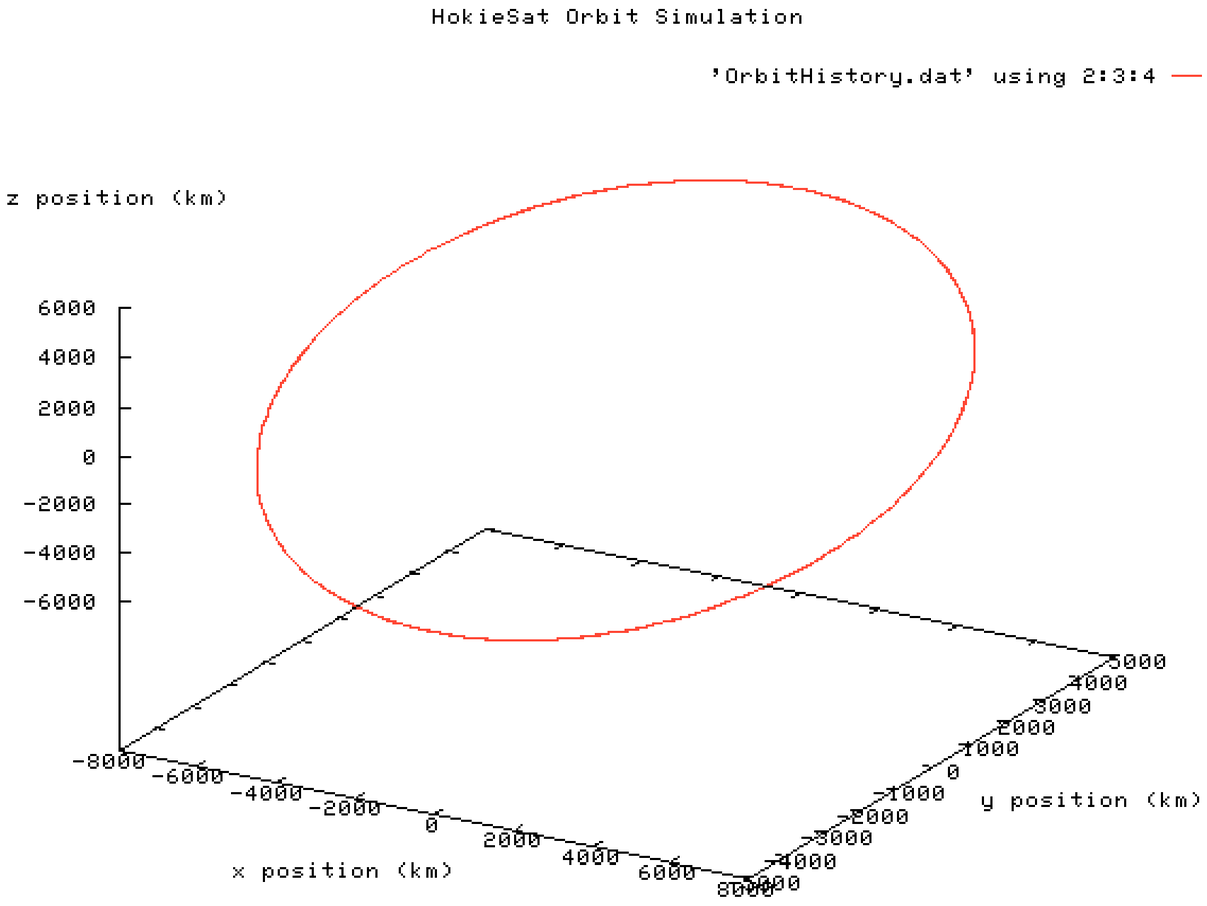
Hardware-in-the-Loop



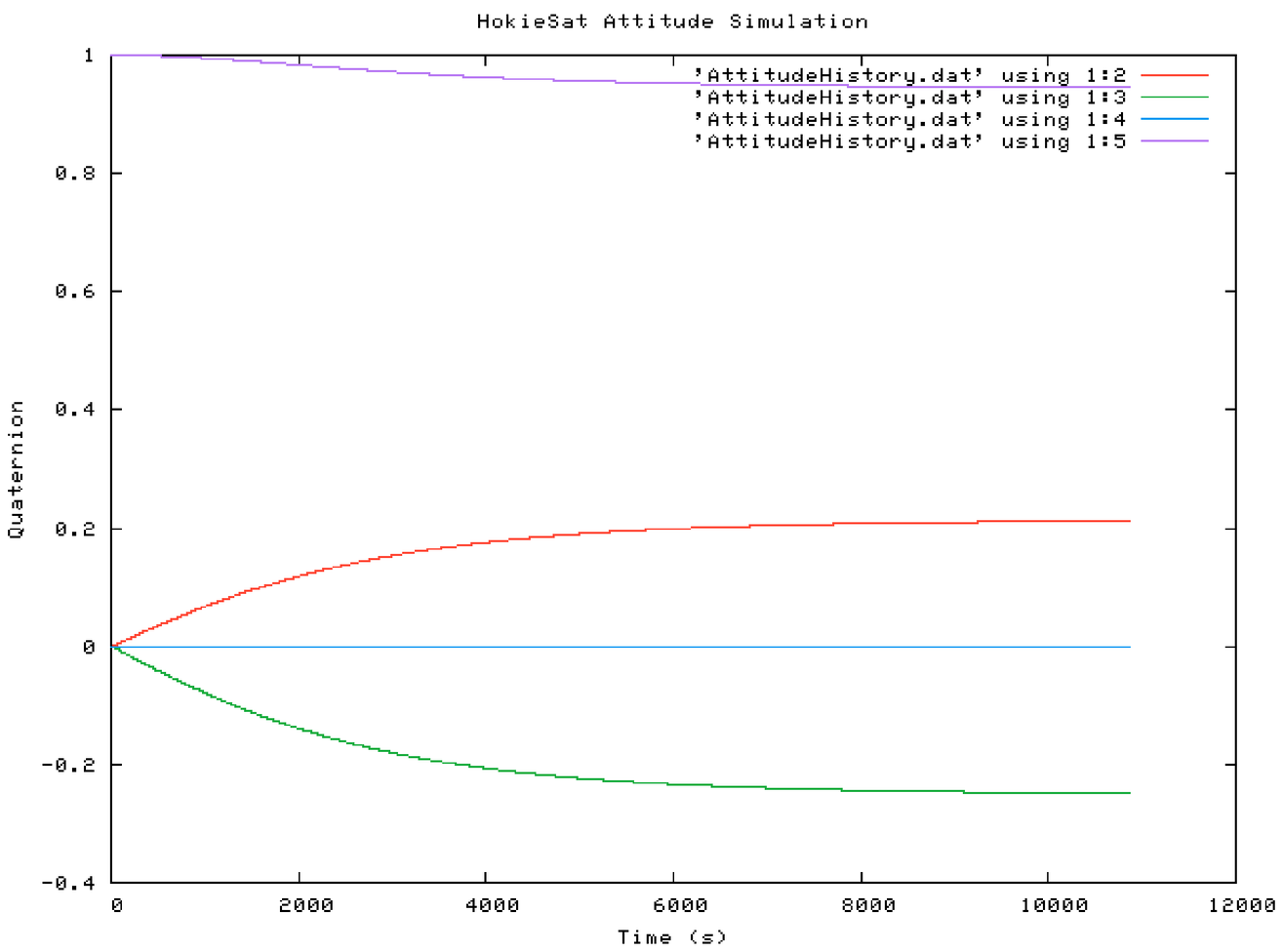


HokieSat Simulation

Orbit Plot



Attitude Plot



Conclusions

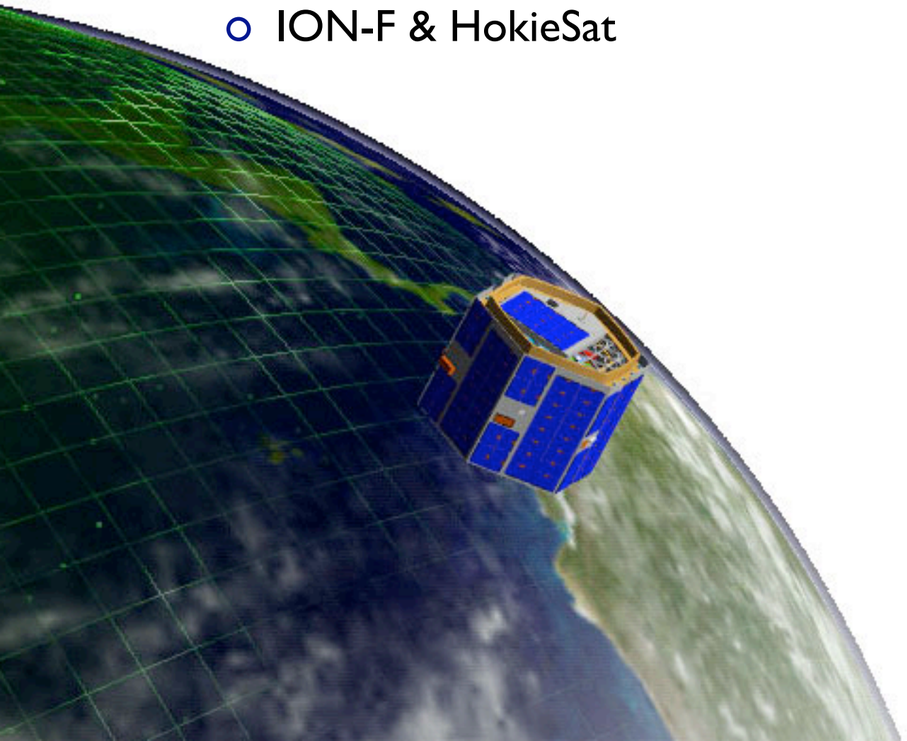
- No prior solution for configurable, extensible, coupled spacecraft simulation
- Open-SESSAME fulfills goals and requirements
  - Speed and accuracy comparable to commercial packages
  - Used to ease development and testing of HokieSat flight code
- Future
  - Multiple extension points for added functionality
  - Integrate software into curriculum
  - Publish use cases and include in lab activities
  - Increase connectivity with other programs
  - Optimize operation

Thank you

- Space Systems Simulation Laboratory
  - Jana Schwartz
- Dr. Christopher Hall
- Sourceforge.net
- ION-F & HokieSat

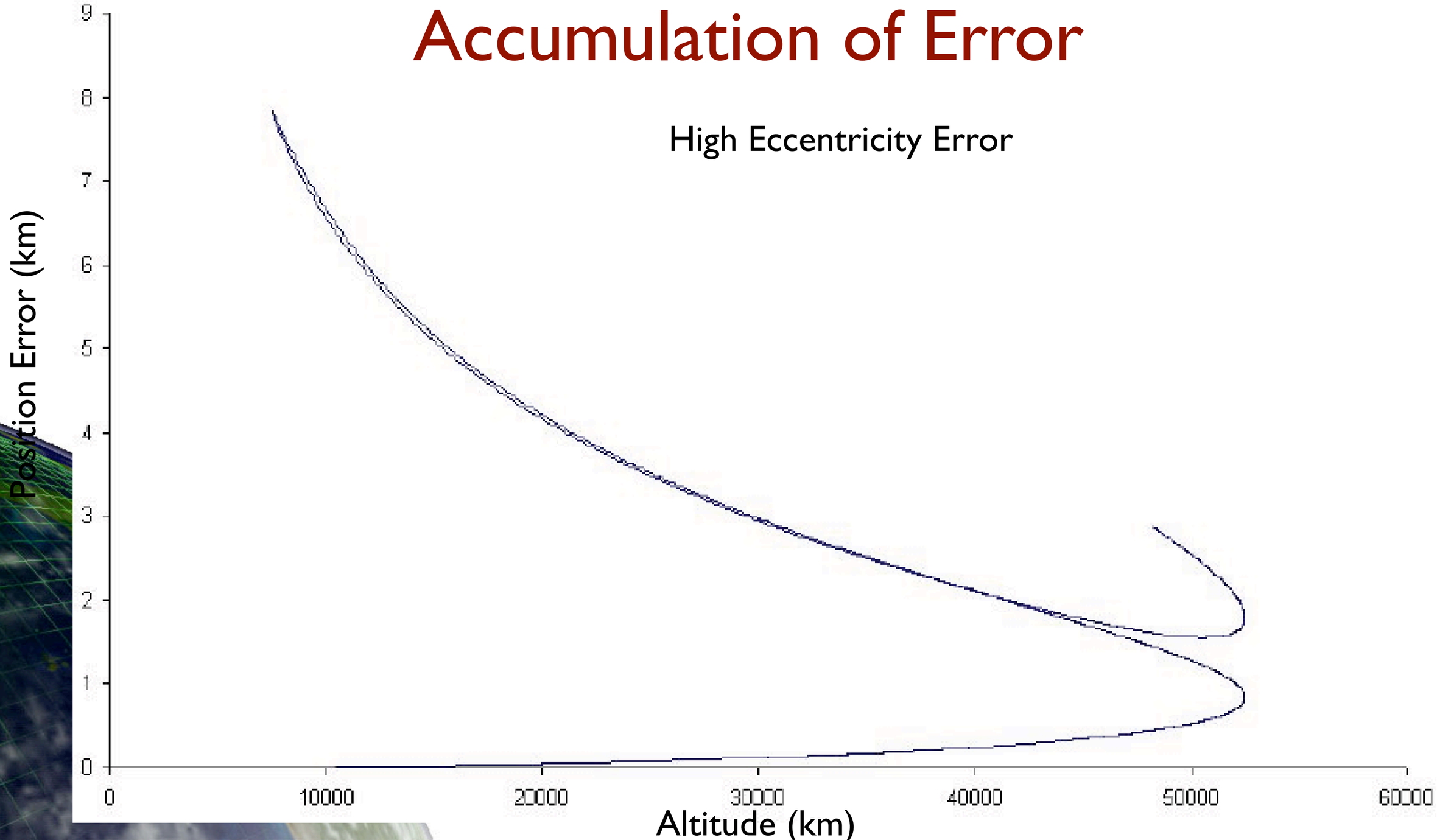
the Space Systems Simulation laboratory

Virginia Tech Department of Aerospace & Ocean Engineering



Questions?

Accumulation of Error



High Eccentricity

Parameter	Value		
Initial Orbit State	$\mathbf{r}_0 = \begin{bmatrix} 7500 & 0 & 0 \end{bmatrix}^T$		
	$\mathbf{v}_0 = \begin{bmatrix} 0 & 6.819333903 & 6.819333903 \end{bmatrix}^T$		
Disturbances	Two-Body Gravity		
Integrator	Runge-Kutta-Fehlberg 4(5)		
Simulation Time	20 days		
Output	**O-SESSAME**	**STK**	**FreeFlyer**
Final Orbit State	$\begin{bmatrix} -46724.94751 \\ -8276.177337 \\ -8276.177337 \\ 1.339061791 \\ -0.85741544 \\ -0.85741544 \end{bmatrix}$	$\begin{bmatrix} -46727.35881 \\ -8275.048124 \\ -8275.048124 \\ 1.338824 \\ -0.857447 \\ -0.857447 \end{bmatrix}$	$\begin{bmatrix} -46713.96349 \\ -8283.620519 \\ -8283.620519 \\ 1.340490327 \\ -0.857151194 \\ -0.857151194 \end{bmatrix}$
Running Time	15 sec	4 sec	48 sec
RMS Error	-	2.72×10^{-4}	1.73×10^{-3}