Web Based Space Mission Visualization System

Daniel O'Neil
daniel.a.oneil.@nasa.gov

Introduction

Free web based maps and digital globes enable people to explore area, plan trips, find a variety of services, and get directions from point A to point B. We need a similar system for the Solar System. For centuries, mechanical orreries enabled people to view and discuss the structure of the Solar System and make predictions about eclipses and other celestial events. In the 21st century, a web based orrery can serve as a tool for astronomers, educators, entrepreneurs, mission planners, game developers, authors, and the general public. This proposal describes use cases, functions and features, a development approach, and programmatic aspects of a Web-based Mission Visualization System (WMVS).

Use Cases

An orrey is a solar system model that can be viewed from several points of view and manipulated to present the positional relationships among celestial bodies. Figure 1 depicts a mechanical orrery. If we had a general purpose web-based mission visualization system what could we do with it? The following subsections explain a variety of use cases.

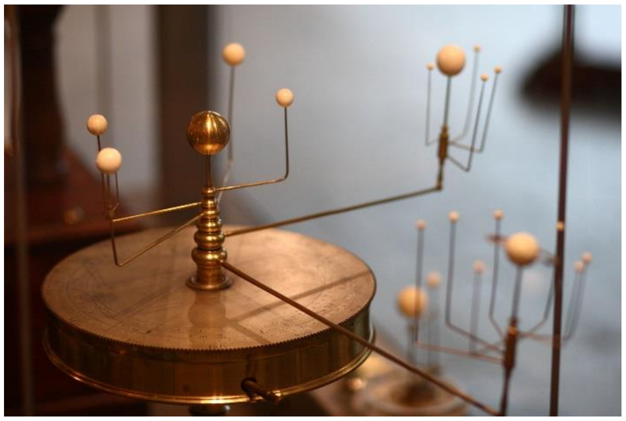
Figure 1 A 1766 Benjamin Martin Orrery, used at Harvard, source: Wikipedia

Education

Educators and students could explore the solar system through the orrery part of the WMVS. A user interface with time line and dial for accelerating time could provide the capability to view celestial events such as solar eclipses, visits from Halley's comet, or the Tunguska event. Other user interface widgets could provide a capability to measure distances between various celestial bodies and estimate travel times to get from point A to point B at various velocities.

Science and Exploration Mission Visualization

Mission planners could use the WMVS to identify way points and orbital maneuvers. A 3D model repository could enable planners to either select existing spacecraft models or upload a new model. By assigning velocities to the mission plan and selecting a cockpit view, mission planners could simulate the mission and see the Solar System from the perspective of the spacecraft as it moves among the way points. Mission planners could save the way points and make them public or limit access to a list of people by created an e-mail distribution list. Mission plans saved for public viewing can educate the general public. A feature of the mission plans could include rationale for maneuvers and way points to provide insight and a basis for discussion. Space agencies could incorporate interactive mission visualizations into their websites.

Emerging Space Industries

Space entrepreneurs could use the WMVS to augment their business plans by depicting service routes, traffic models, and service expansion. A digital globe would be better suited for depicting commercial activities in Low Earth Orbit, such as satellite servicing, orbital debris collection, or orbiting hotels. A WMVS would better serve depiction of commercial space travel to the Moon, asteroid mining, or commercial transportation among space colonies. Space entrepreneurs could use the 3D model repository to deploy spacecraft and the missing planning tools to depict anything from assaying asteroids to deploying space infrastructure for communications, power, and manufacturing. Corporations could integrate interactive space operations models into their websites.

Serious Games

The entertainment industry could use the WMVS code base as a starting point to develop serious games about space colonization, resource prospecting, Solar System wide economic simulations, space piracy, space battles, and races through the Asteroid belt. An open-source license that allows the use of the code in commercial products would encourage companies to adopt the vetted technically accurate orrery. As the serious games flourish, the companies can give back to the open source repository by providing user interface widgets, 3D models, and an improved look-and-feel of the user interface.

The Web Based Orrery

The Use Cases identified a few WMVS functions including mission planning tools, various camera views, a time line, a time acceleration widget, and a 3D model repository. Features of the WMVS include native execution with a modern web browser, an Application Programming Interface (API) the enables the development of WMVS based applications, and an open source development approach that enable contributions from everyone. The following subsections provide details about these functions and features.

Native Execution within a Web Browser

The capability to use the WMVS without the requirement to download a thick-client, install a plug-in, or update a virtual machine will make the system more attractive to the casual user. Programming language such as JavaScript and WebGL execute natively within popular web browsers and take advantage of graphics processors.

Planets, Moons, and Asteroids

An ideal web-based orrery would include all of the known planets, moons, and asteroids; however, rendering detailed models of the 600,000 known asteroids and objects within the rings of Saturn could overwhelm a web browser. Menu items or check boxes could provide a capability to configure orrery so that it presents the most interesting celestial bodies to the individual. One website visitor may want to view Earth and a selected set of Potentially Hazardous Asteroids (PHA). Another visitor may want to view Jupiter and its moons.

Level of Detail

Managing level of detail can reduce the load on the graphics processors. From a distance, the asteroid belt could appear as particles and the planets could be rendered with procedural textures. As the point of view moves closer to a planetary surface, a texture library could provide a more technically accurate texture map.

Asteroids that have shape files could have corresponding 3D models. From a distance, a generic ovoid shape could represent the asteroid. As a camera moves closer to the asteroid, a more accurate 3D model could be substituted for the generic model.

Future Functions

Initially, the WMVS ought to focus on a few functions and features that support education and presenting mission concepts. As the project attracts more developers, additional functions and features could be integrated; the following subsections identify three functions that could wait for a later release.

- *Particle Dynamics* – depicting engine plumes and dust plumes could wait. An initial implementation could be to depict the asteroid belt from a distance.
- *Gravitational Effects* – Initially, the orbits could simply propagate the orbital elements. Periodically, an interface could update the orbital elements from JPL and other organizations that track celestial bodies. A Future version of the system could address gravitational effects that perturb orbits.
- *Surface Landing* – Landing on the planetary surface ought to be done with a digital globe. A future version of the WMVS could include interfaces to digital globes such as NASA World Wind, Google Earth, and Cesium to hand-off a mission visualization. The orrey could focus on the in-space part of the mission and the digital globe could depict the landing. A website could include frames to present the overview with the orrery and the surface landing with a digital globe.

Time and Navigation Widgets

User interface widgets for managing time and selecting points of view will prevent the visitor from getting lost in the orrery. An initial set of widgets include:

- *Time Line* – A configurable timeline could present minutes for an approach to an asteroid, weeks for the depiction of a spacecraft conducting an orbital maneuver, months for a mission to Mars, years to depict comet and meteor trajectories, centuries or millennia to show long term behaviors of the Solar System.
- *Time Accelerator* - A widget, such as a dial or slider can provide a visitor with a capability to accelerate a mission visualization. Depending on the implementation orbital propagator(s), time acceleration could introduce errors.
- *Way Point Selection* – Another method for time acceleration could be a widget or menu for selecting previously defined way points. When a visitor selects a way point, the system could reset the orbital propagator(s) to mitigate the accrual of errors.
- *Camera and View Selection* – A function for establishing and pointing cameras could enable the presentation of a mission concept from several perspectives. When a visitor selects a spacecraft model the system could establish a camera at the bough of the ship to

depict the spacecraft's point of view. If the mission involves multiple spacecraft, each ship could have a selectable camera.

Mission Planning

The Use Cases section describe how mission planners and space entrepreneurs could use the mission planning functions to establish way points to present a mission plan. Mission planning widget and screens include:

- *Map Views* – Working in 3D can be difficult because the two-dimensional computer monitor is presenting a three dimensional volume. Selectable 2D map views can simplify the process of placing way-points. A visitor could select two map views and place the point on those two views to generate the 3D location. The map views ought to include selectable center points and distance scales. A visitor could start with Earth as a center point and after establishing a few points, he or she could select Mars as the center point to establish additional way points.
- *Way Point Definition* – A way point identifies a location in space where a spacecraft will start, travel through, or complete a mission. Way points could be represented with a variety of user selectable colored 3D icons. When creating a way point, a visitor could enter paragraphs to explain the rationale for placement and links to supporting information.
- *Trajectory Trace Path* – As a mission simulation executes, a spacecraft moves along a defined path through a set of way points. A detailed analysis of orbital trajectories could be done beforehand with design and analysis tools like NASA's General Mission Analysis Tool (GMAT) and Analytical Graphics Incorporated's System Tool Kit (STK). The WVMS could offer a simple orbital propagator to move a spacecraft along a trajectory provided that the visitor supplies the orbital elements for each leg of a mission plan. If the visitor does not supply the orbital elements, then the WVMS could offer straight lines or Bezier curves between way points. During the simulation, the WVMS could trace the trajectory paths with visitor specified colors, thicknesses, and line types.

Model Library

A model library could include 3D geometric meshes of asteroids, spacecraft, and other objects. Features of the model library include a 2D image, description, and dimensions of the models in the library.

- Import Articulated 3D models – a spacecraft model may need to reconfigure itself during a mission, for example, it could deploy solar panels and antenna. Computer modeling programs, such as Blender, provide a capability to create models with rigs and articulated movements. The import function ought to support a file format that includes the rig and articulations. The mission planning way point editor could include an option to activate the model's articulated behavior.
- Import Texture Maps – The file format of the 3D model ought to include texture maps so that the mission planner does not have to manipulate the texture maps within the WVMS. There exists a variety of powerful free 3D modeling programs, Blender for example, which can export a COLLADA file that includes texture maps.

Application Programming Interface

The public WVMS ought to provide a core set of functions and features with an Application Programming Interface (API) that enables developers to extend system capabilities and to create new applications that use the WVMS code base. The following subsections describe interfaces to external databases with data about celestial bodies, usage of existing computer graphics code libraries, and access to the Solar System scene-graph.

Interfaces to Celestial Body Databases

The Jet Propulsion Laboratory Near Earth Objects (NEO) Project Office provides access to a database of orbital trajectories for PHAs. The Asterank website provides Representational State Transfer (REST) protocol interface to its database. The WVMS API ought to provide functions that for calling these external databases and other sources of celestial body data. Considering that accessing these external sources during a simulation could slow down the system, the WVMS may need a local database that is periodically updated as JPL and other organizations update orbital element data. These external database data request functions could provide developers the capability to get the latest data during an initialization phase.

Interfaces to JavaScript & WebGL libraries

Instead of creating a new 3D graphics code library, the WVMS ought to build upon an existing code library, such as X3Dom or Three.js. If the development team can identify multiple compatible 3D graphics, data visualization, and user interface code libraries, these API functions could provide a capability to program at a higher level. Consider the mission planning tools, there could be calls to three or four different code libraries, a WVMS API function enable the developers to work in an object oriented fashion to establish map views, way point editors, and trajectory trace paths.

Solar System Scene Graph

A scene-graph is a data structure that defines an integrated collection of 3D objects. A Solar System scene-graph at the heart of the orrery would include the planets, moons, and asteroids. A user interface to configure the scene-graph to add or remove objects. Functions in the WVMS API could enable a code developer to navigate, read, or modify the scene-graph for the purpose of adding data or updating data visualization overlays.

Celestial Object Builder Utility

A future set of functions could provide a capability to build simple planetoid models. Presently, only a few shape files of asteroids exist. To provide a more sophisticated model of an asteroid than the simple ovoid shape, the celestial object builder utility functions could enable a visitor code developer to import 3D models and specify landing spots or locations of desirable resources.

Skills to Components Mapping

A web based mission visualization system will have many parts and require a variety of skills to implement the parts. The following table maps the needed component development to the types of skills needed to implement the components.

Application Component	***Needed Skill(s)***	***Rationale***
Orbital Dynamics Functions	Orbital dynamics expert, Physicist, or Mathematician	Kepler's equations specify the motion of celestial bodies. Required skills involve translating equations into code for a library.
Thrust equation and orbit transfer functions	Aerospace engineer, Physicist, or Mathematician	Equations for modeling thrust and knowledge about orbital maneuvers, such as the Hohmann transfer enable the development of spacecraft mission simulations. Needed skills involve translating various maneuvers into algorithms that a programmer can implement as a code library.
Solar System Model or Orrery	Computer graphics programmer and 3D graphics modeler. Physicist consultant	A reusable model of the solar system, implemented as a hierarchical scene-graph with rotating texture mapped planets serves as the mission visualization environment. Required knowledge and skills include 3D graphics programming, 3D modeling, and texture mapping. Consulting with a Physicist will help ensure that planets are spinning correctly.
Way-point mapping visualization	Computer graphics programmer	Waypoint and orbital trajectory visualization will result from mission planning. Required knowledge involves an understanding of 3D computer graphics algorithms and scene-graphs.
Interactive Visualization Widgets	User Interface Designer and computer scientist	Widgets, such as menus, dials, sliders, and timelines, will enable an end-user to select camera views, accelerate or decelerate the simulation, select a point of time within a mission, and manipulate the point of view. Skills include designing the graphical user interface and mapping the controls to the appropriate functions within the application.
Model Library	Graphics expert(s) in 3D modeling and computer scientist	A 3D model library will enable mission designers to upload or select models for mission visualizations. Required skills include some user interface design, file management code development, and functions to integrate the models into the orrery scene-graph.
Application Programming Interface library	Computer scientist and technical writer	An API library enables experts to utilize the orbital dynamics, thrust, and orbit transfer functions to create mission visualizations. Required skills involves writing descriptions of the functions and producing web documentation.
Orbital element data access functions	AJAX, RPC, or REST code developers	Accessing orbital element data about asteroids, planets, and other celestial bodies requires occasional remote access to databases. Required skills involve remote access protocols, such as Representational State Transfer (REST).
Mission Planning User Interface	Graphics expert(s) in 2D and 3D	Mission planning involves 2D and 3D views of the solar system, way-point specification, and orbital maneuver selection. Required skills involve user interface design, translating 2D map way-points to 3D points within the scene-graph, and generating scripts of orbital maneuvers that move 3D models of spacecraft.

Development Approach

An open source development approach can improve the chances of wide-spread adoption of the WVMS. Establishing the credibility of the system requires involvement of world-class orbital dynamics experts, mission planners, and code developers. The following list of features describe a development approach to engage experts through-out NASA, the aerospace industry, and the world of independent code developers.

- *Agency Wide Team* – Establishing an agency wide team will ensure wide-spread adoption within the agency and build a community with expertise in orbital dynamics, spacecraft design, mission planning, 3D graphics, and web-app programming
- *Verifying and Validating Orbital Propagator(s)* - Vetting the orbital propagators is essential to establishing credibility. Reaching out to the general aerospace and astronomical communities to establish independent review teams can ensure that the orbital propagators have been properly implemented.
- *Verify and Validate Code* – Open source development projects typically use bug tracking systems and volunteers to work out the bugs. Establishing a development team for core functions and features could lead to a more formal development and review process that leads to better quality software. As the volunteer workforce builds upon the core code base, the core team could establish regression test procedures and discussion forums for code walkthroughs to verify the code.
- *Free Open Source Code* – Selecting a flexible open-source code license could motivate the commercial game industry to build upon the code base and contribute to it. The license ought to allow usage of the code base in commercial products. The terms of the agreement ought to include a requirement that companies that benefit from the code base contribute some code or improvements.
- *Engage Programming Community* – Establishing contests and prizes are ways to attract independent developers and small businesses. Writing and presenting conference papers provides visibility for the project and can attract students who are looking for independent study projects. The following list provides three examples of forums for engaging the programming community.
    - *TopCoder* – With a world-wide community of over 300,000 programmers and a standardized contest driven life-cycle, TopCoder could provide a development team to jumpstart the project
    - *International Space Apps Challenge* – An week-end challenge that attracts thousands of independent programmers. A set of challenges could request software objects that perform specific functions or provide particular features.
    - *Datanauts* – A data visualization team that builds reusable components for NASA open source projects.
    - *Simulation Conferences* – Presenting the WVMS progress at simulation conferences could attract independent developers or companies to the project.

Schedule

Milestones for a software project include requirements definition, architectural design, user interface mockups or wireframes, code development, demonstrations, testing, validation, etc. A detailed schedule could be developed by the core development team.

Cost Expenses may include charge codes for civil servants, prize money for contests, and grants to universities. If implemented as Datanaut and Space Apps challenges, the costs will be the civil-servant time to provide technical guidance and socialize the application. A detailed cost breakdown can be developed by the core development team.