Service Oriented Architecture (SOA) and Space Analysis INtegration Toolkit (SAINT) White Paper

DSoft Technology is a leading authority and management agency of the AFSPC A9 Astrodynamic Standards (AS) repository. DSoft Technology is well-versed in using and integrating these AS, having worked with the AS since 2005. Since then we have performed CM, testing, integration, and distribution of the AS, and understand the complexity of their integration, constraints and use.

The diagram below provides the overall design for SAINT, a DSoft-developed tool for AFSPC/A9 built on a Visual Studio toolkit (.NET) and framework that allows users **to create new space applications by combining components** (2D and 3D visualization, robust data grids, export) and assemblies that include the AS. SAINT is a **Government Off-The-Shelf (GOTS)** product that combines algorithms, data structures and visualization components using Microsoft's Model-View-Controller architecture pattern. The SAINT framework can easily be applied in a Service Oriented Architecture (SOA) where the different .NET Services (and thus the underlying algorithms) can be made available on the network. Any application with access to the SOA Services can then invoke these.

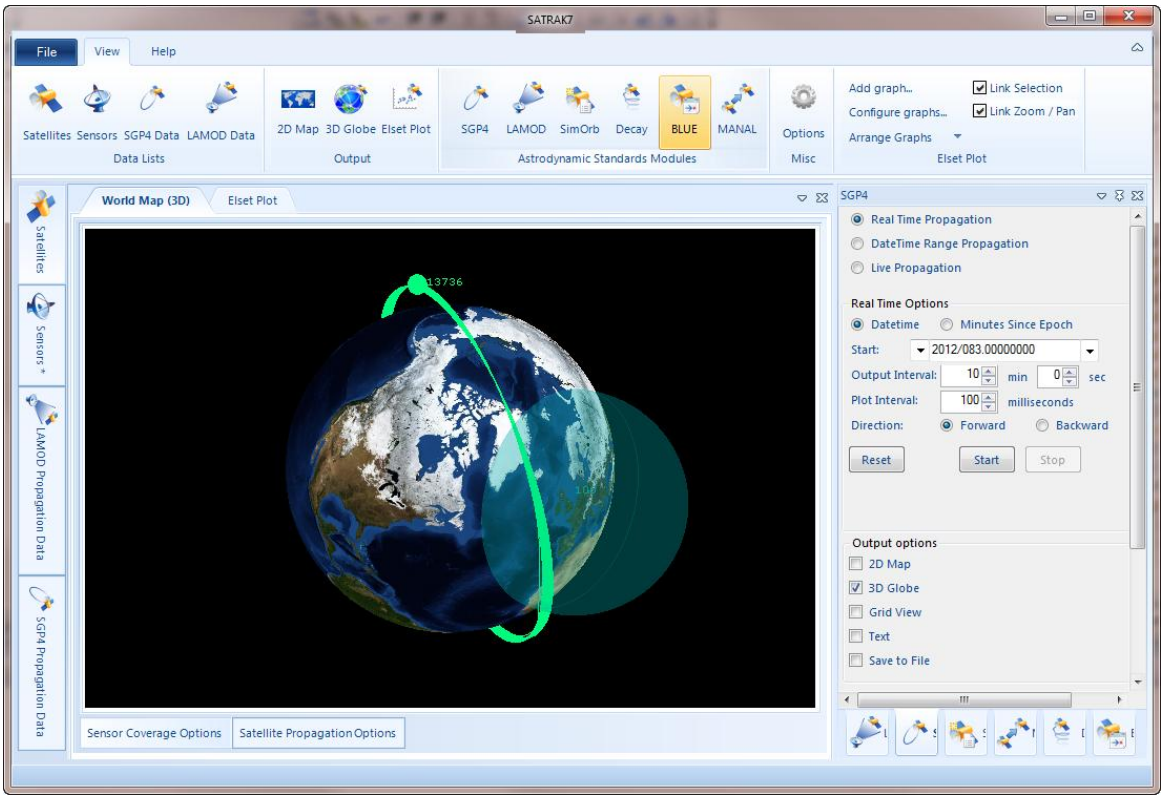

Exhibit 1: DSoft Technology's AS and M&S capabilities provide excellent functionality with no licensing cost.

A virtualization architecture would allow for multiple server instances to run on a single server platform where high Input/Output (I/O) is not a priority. Dedicated server instances can be optimized for higher I/O where needed. A client / server model may be also utilized where simulation tools can be run on user workstations and data managed from a central repository. **Integrating SAINT into candidate SOA, would allow multiple users access to tools and data simultaneously.** A SOA approach is one where application design and development is based on the concept of services. It applies successful concepts

proved by object oriented development, component based design, and Enterprise Architecture integration technologies.

SOA is an architecture that represents SW functionality as discoverable services on the network. Using SOA, data can be sent as asynchronous messages from one service endpoint to another. These service endpoints can be client applications, a continuous service running on a dedicated server, or any other client / server that requests / provides data. The data messages can be as simple as single values, XML or even a complex stream of binary data. Messages sent between the different service endpoints can be encrypted. Therefore, a SOA enables the design of SW systems that provide services to other applications (models and SW tools) through published and discoverable interfaces, and where the services can be invoked over a network. When implementing an SOA, we would create a new way of building and organizing M&S applications within a more powerful, flexible programming model, resulting in faster development and lower ownership costs and implementation risk.

As an example, the Special Perturbations (SP) propagator AS can be very computationally intensive. By providing this algorithm as a SOA service it can be called from a low-end client, executed on a high-performance server and return the results back to the client for visualization or further analysis. The server can be more easily scaled-out if demand increases than upgrading all client workstations.

The Algorithm Dynamic Link Library (DLL) integrated in SAINT can be written in any language as long as it is compiled as a 32 or 64-bit DLL. DSoft Technology has successfully integrated DLLs written in C# and FORTRAN and foresee no issue in integrating DLLs written in MATLAB. (MATLAB Compiler™ converts MATLAB code into a shared library and execution of that library is provided by their freely distributable runtime engine, MATLAB Compiler Runtime).

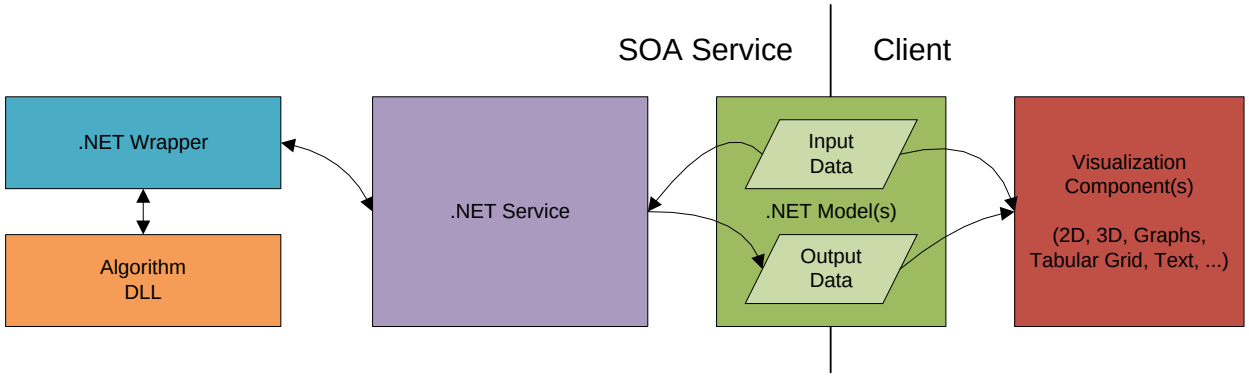

Exhibit 1: A Service-based Architecture Like the One Above will Likely Provide Significant Benefits to Reusing SAINT

The first step in integrating the DLL into the SAINT framework is to create a lightweight wrapper that can interface with the static methods provided by the algorithm DLL. The wrapper's only function is to provide entry points into the DLL and will not manipulate any data. That is the role of the .NET Service which will take input data of any form, convert it to the correct format and provide it to the algorithm through the .NET Wrapper. Output data from the algorithm is then routed back to the .NET Service which provides this data to any of the visualization components. If needed, the .NET Service converts the algorithm data to the correct format before providing the data to the components. Different .NET Services communicate to each other through data, meaning output data from one .NET Service can be used as input data for another. This provides a loose coupling mechanism that prevents tight and complex dependencies between .NET Services as well as a limitless chaining of .NET Services to support complex scenario's. New algorithms integrated into SAINT can take advantage of the existing algorithms (AS) and components (2D, 3D, Graphs, Grid, Export, etc.). Exhibit 2 shows sample output of the graphs in SAINT.

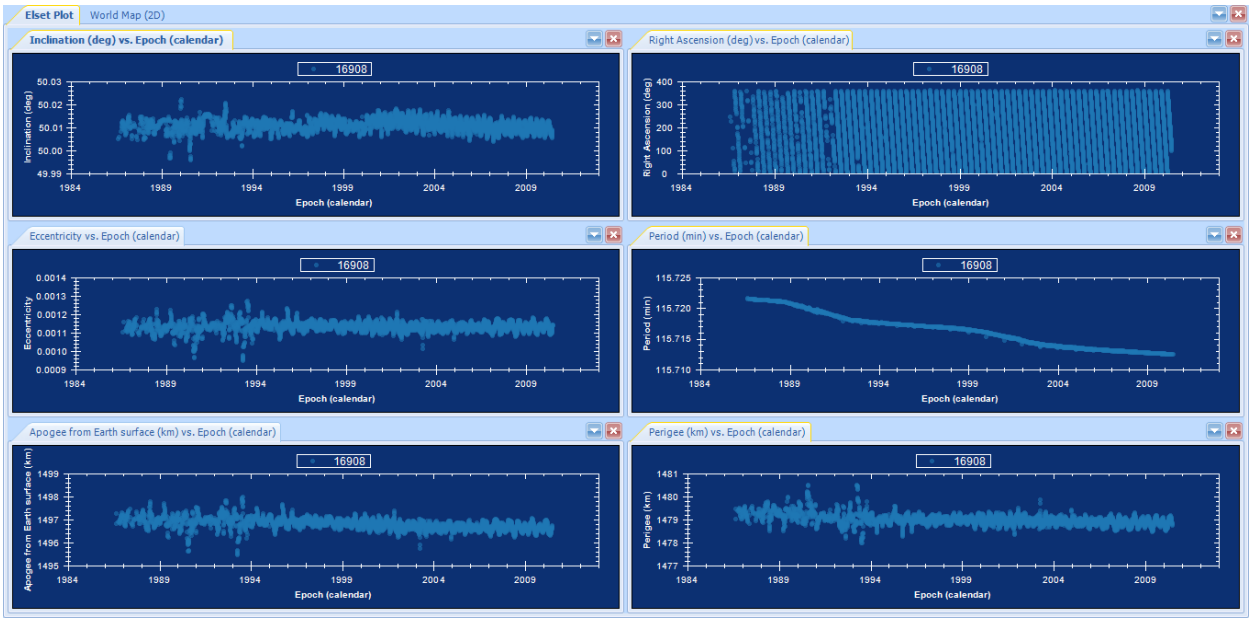

Exhibit 2. Elset Data Plotted Against Epoch Time In SAINT

The framework allows for rapid integration of any algorithm (or other type) code by creating light-weight wrappers. **Any type of language can be used to code the algorithm as long the code can be compiled into DLLs.** Currently SAINT has algorithms integrated that were written in FOTRAN95 and C#, all of them part of the AS and Algorithms Library (ASAL). SAINT itself is compiled into a set of DLLs that can be used from any other programming language. As an example, the Maneuver Detection and Recovery (MDR) module for System Effectiveness Analysis Simulation is written in MATLAB and calls the SAINT Look Angle Module (LAMOD) and AstroFunc algorithms to help detect maneuvers for a given set of satellites. MDR initially integrated older versions of the AS standalone executables. With the replacement using the SAINT DLLs, improvements in both accuracy and performance were realized.

Algorithms developed in other applications may be extracted (if design permits) and integrated into SAINT. A preliminary investigation of other AS algorithms integrated into the Astrodynamic Support Workstation was performed. Most of these algorithms are written in C or F77. DSoft Technology has

extensive experience in taking legacy Fortran 77 code (Orbit Simulator (SIMOrb) and BLUE) and rewriting it or putting wrappers around the code so that it can be integrated into SAINT.

Another important part of the SAINT architectural framework is the visualization components that allow the user to present input and output data in various ways. Any data can be presented as raw text, a tabular grid with advanced sorting, filtering, grouping, and formatting, and a 2-dimensional graph that can visualize two additional dimensions, using color and size of the data points. Location-aware data can also be visualized using a 2-dimensional map and or 3-dimensional globe representing the Earth. Although not as powerful as the STK tool from AGI, it provides much of the base functionality like rendering satellite orbits, ground sensors, sensor coverage overlay, look angles, etc. with no licensing cost.

As a prime example of how SAINT can integrate various algorithms and visualize the data, DSoft Technology developed the next-generation version of the Satellite Trajectory and Attitude Kinematics (SATRAK) tool. It combines the core algorithms from the ASAL (Simplified General Perturbations 4 (SGP4), LAMOD, SimOrb, Orbit Decay, Maneuver Analysis) with the various visualization components allowing the end-user to rapidly model orbits for given set of satellites, calculate look angles and sensor coverage against a list of sensors and provide quick situational awareness. SATRAK has recently been approved by Air Force Network Integration Center to operate on both the Non-Secure and Secure Internet Protocol Router networks.

The SAINT architecture supports selecting which algorithms to integrate into applications. It was decided to only provide certain algorithms in SATRAK however SAINT currently has higher fidelity AS algorithms integrated into it that can be used by AFSPC/A9 These include Standard Special Perturbations (SP), Standard Special Perturbations (SP) Propagator Satellite State Vector (SPVEC), Report Observation Association (ROTAS) and Observations (OBS). Work on integrating Computation of Miss Between Objects (COMBO).

Integration of other algorithms used by the missile community has been discussed that would take advantage of the visualization and analysis tools already implemented in SAINT. Integration of these algorithms can be done at a fraction of the cost using SAINT to create new applications. MATLAB algorithms written in BLADE and MASTR, two engineering models used by AFSPC/A9 could be easily integrated into the SAINT framework in one or two days that would immediately allow for the visualization and analysis of missile threats with radar coverage.

M&S code should be designed to be executed on commercially available high speed computing servers, distributed applications within advanced developer environments using object-oriented languages that emphasize reuse of code, flexibility and platform independence. Using the AS provides tested, trusted, and centrally maintained compatible code to ensure interoperability and meet the AFSPC Instruction 60-102 directives. Use of these trusted standards ensures compatible calculations across new and existing AFSPC systems for satellite trajectories and related data in support of the Air Force Space Control Center. The benefits for using the AS ensures accuracy, reduces risk and cost, and has the advantage of new releases/updates as they become available. Updates may be directly ported into the delivered SAINT architecture with minimum development cost exposure.

Although SAINT and SATRAK may not be as robust as other tools such as Satellite Toolkit™, they provide similar capabilities, support a very flexible architecture…**and there are no annual licensing costs!** Additional information may be found on the DSoft Technology website at: http://www.astrodynamicstandards.com/standards. AS SW compatibility across multiple platforms may be found at: http://www.Astrodynamicstandards.com/Resources/ Astro_Stds_List.pdf.