On-Board Mining in the Sensor Web

Steve Tanner, S. Graves, M. Alshayeb, E. Criswell, A. McDowell, M. McEniry, and K. Regner
Information Technology and Systems Center
University of Alabama in Huntsville
S345 Technology Hall
Huntsville, Alabama 35899
stanner@itsc.uah.edu
www.itsc.uah.edu

The Information Technology and Systems Center (ITSC) at The University of Alabama in Huntsville is developing an innovative processing framework aimed at assisting science users in the use of the unique constraints and characteristics of the on-board satellite data and information environment. The Environment for On-Board Processing (EVE) system serves as a proof-of-concept of advanced information systems technology for remote sensing platforms. With EVE, data is processed as it is collected, enabling the production of custom data products on-board and in real-time. The web-based drag-and-drop EVE editor allows science users to build processing plans, which are compatible with the constraints of on-board computing environments. The EVE on-board, real-time processing infrastructure, will upload, schedule, and control the execution of these plans. Operations within the plans provide capabilities focused on the areas of autonomous data mining, classification and feature extraction using both streaming and buffered data sources. These will contribute to science research applications, including natural hazard detection and prediction, fusion of multi-sensor measurements, intelligent sensor control, and the generation of customized data products for direct distribution to users. A ground-based testbed has been created to provide testing of EVE and associated science applications in a heterogeneous, embedded hardware and software environment. Testbed components include platforms that represent both space based and ground based sensor platforms, including wireless sensor mesh architectures.[*]

1 INTRODUCTION

As sensor technology and the platforms they run on have evolved over the past few years, more and more computing resources have been placed on-board. This includes not just raw computing, but also memory and communications bandwidth. This has provided a new domain of research and development: How to manage and exploit these new resources, on-board and in real-time. According to NASA's Earth Science Vision, on-board processing will play a significant role in the next generation of Earth Science missions, providing the opportunity for greater flexibility and versatility in measurements of the Earth's systems. Such on-board processing can contribute to many Earth Science research applications, including natural hazard detection and prediction, intelligent sensor control, and the generation of customized data products for direct distribution to users. Ideally, the availability of custom processing, feature extraction, and data mining onboard satellites can allow end users to specify their own data products through the definition of a processing plan. On-board processing will reduce the volume of delivered data since only the data specified by the processing plan is transmitted to the user. This in turn will improve the accessibility and utility of Earth Science data sets.

The Information Technology and Systems Center (ITSC) at the University of Alabama in Huntsville (UAH) has begun the third year of a three-year research and development effort [ESTC 2001, ESTC 2002]. The goal of the effort is to build a prototype system which will allow research scientists to build processing plans and execute those plans in on-board processing environments.

This work, to address on-board satellite data processing, is based on the Center's existing custom processing, feature extraction, and data mining technologies. Therefore, the Environment for On-Board Processing (EVE) benefits from ITSC's experience with scientific data mining and knowledge discovery. Other current ITSC research projects deal with the distributed and heterogeneous nature of Earth Science data sets, as well as data integration, data fusion, and high-performance networking in Grid-based environments. Based on this background and a broad range of research affiliations, ITSC has designed and is now developing a new breed of processing system capable of handling the unique constraints and characteristics of the on-board sensor environment.

While ITSC has made significant progress in custom processing [1,2,3,4] and data mining technology [5,6], the current architectures were not designed to be optimal for on-board processing in an autonomous environment. In particular, the on-board processing environment includes significant hardware constraints, coupled with requirements for processing real-time streams of sensor measurements. Furthermore, the need for fusion of multi-sensor inputs, both on-board a single platform, and ultimately through communication between platforms, is of great importance. Therefore, EVE required a complete

[*] Financial and technical support for this effort is provided by NASA's Earth Science Technology Office.

re-engineering of current well-established software, ranging from fundamental changes in basic system architecture through new implementation of processing modules for greater efficiency. This approach reduced the risk, cost, and time associated with development of a full suite of on-board processing functionality. Integral to this effort is the simultaneous development of a ground-based testbed, which enables researchers to perform testing and certification in an environment simulating the expected on-board processing environment. This includes both on-orbit platforms as well as ground based sensors.

The ITSC has embarking on a phased approach for this research, which will advance the custom processing and data mining software from NASA's Technology Readiness Level (TRL) 2 to TRL 5 over a three-year period. This means that the system will progress from a purely theoretical research endeavor into a demonstrable system.

2 EVE System Requirements

The primary goal of the EVE project is to prototype a processing framework for the on-board satellite environment that includes data mining, classification, and feature extraction capabilities in order to support multi-sensor fusion, intelligent sensor control, and real-time customized data product research applications. These capabilities and applications will be tested and refined as they are exercised in the simulated on-board environment provided by the EVE ground-based testbed. To achieve this goal, the EVE team has defined a set of requirements and documented them in a System Requirements Specification.

These include the following primary items:

- Functional requirements: Support for real-time Linux; Use of pre-defined and user-defined operations; and Centralized repository of operations and documentation.
- Processing requirements: Execution of multiple processing plans simultaneously; Multiple communication protocols, including streaming data; Support for real-time polling and interrupts; and Initiation of operations in response to detected events.
- User Interface Requirements: Development of a user friendly interface that provides pre-mission plan editing and upload of operations; and System monitoring and control functions vital to mission success.

3 Major EVE Components

The EVE software architecture is comprised of six major components: Operations, Plans, Carts, a Ground Station, an On-Board System, and an Editor. The first three (Operations, Plans and Carts) represent the users' execution scenarios. The Ground Station and On-Board System are responsible for the execution of those scenarios, and the Editor is the means used to describe the scenarios.

The EVE system relies on the concepts of Operations, Plans and Carts (Figure 1) for users to describe how, what and when they want to execute software on-board. Operations are discrete prewritten routines that perform a specific stand-alone function, such as an image processing algorithm, a data mining routine, or other math or sensor based function. They are the executable building blocks, each representing a single executable stage, within the overall flow of a scientific process. These operations can be connected together to form a plan which, when coupled with sensor data, can perform a more abstract computation, such as cloud detection. A plan can be thought of as a structured network of operations that represents the end-to-end process of sensor data manipulation. A plan in turn can be placed into one or more carts, each representing a single real-time computing unit. These carts can then be uploaded to specific on-board systems, scheduled and executed.

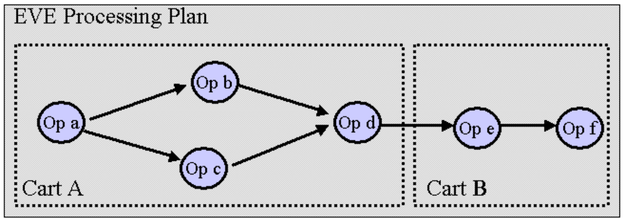


Figure 1: Operations, Carts and Plans

To accomplish the construction, testing and execution of these plans and carts, the EVE system uses three primary components: the editor; the on-board system; and the ground station (Figure 2). The editor is a web-based graphical interface, which can be used to write and validate the plans and carts, and is the user's primary interface with the EVE system. Once a plan is written and placed into carts, it is uploaded to the on-board system, where it is scheduled for execution. The on-board system executes the plan (via the carts) in real-time, based upon both user processing goals and system constraints. The ground station is the primary communication facility between the editor and on-board

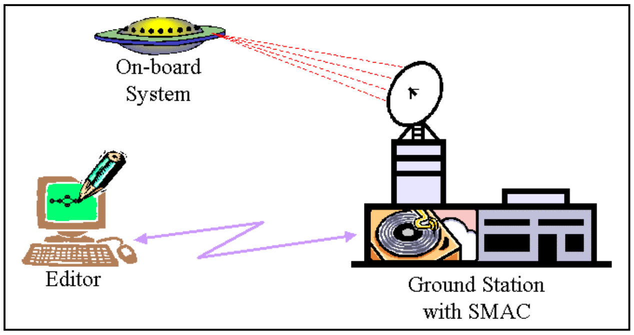


Figure 2: Major EVE Components

system. It also maintains a list of available operations and plans in the system as well as the on-board platforms available for use. The EVE system also has system monitoring and control (SMAC) functionality, which monitors all EVE components for malfunctions and also collects performance metrics.

4 Process Execution

The EVE editor allows a user to build a processing plan which specifies a set of operations and the data stream connections between them. These operations have been written in a general nature and are not aimed at a specific on-board platform. The concept of a cart is introduced in order to package such general operations for execution in specific real-time environments.

A cart is a subset of a plan, which holds a sequence of operations that can be executed as a single real-time unit. One cart may be used for an entire plan or a plan may be divided into several carts. Each cart contains a set of operations and provides a set of services to those operations. It holds metadata about the resources required by the operations and resource usage of the platform in which its execution is targeted. The cart also is responsible for tracking the ID tag of the plan and the other carts that comprise that plan.

4.1 Plans

Plans specify a set of operations and the connections between them. A plan specification forms a data flow diagram where the nodes are the operations themselves and the directed edges specify which operation consumes the output of another. Operations may have multiple inputs, multiple outputs, or both. A plan description must contain all of the data necessary to execute, such as:

- Information about the operations involved.
- Parameters for the operations involved.
- Connections for all of the operations' input and output data streams.
- Information about the graphical layout of the plan from the editor, to preserve the appearance when reloaded by a user at a later time.

4.2 Carts

In a real-time system, it is difficult to schedule tasks that depend upon multiple processes with a high degree of intercommunication. One method for dealing with this issue is to bundle all of the atomic operations of an entire plan into a single process image. However, this conflicts with the EVE goal of flexibility and modular plan building. An approach that offers a more flexible architecture is the construct of the cart. In effect, a cart represents several well-defined operations, bundled into a single process image. This bundling offers ease of communication, resource sharing, and task scheduling between the bundled operations within a cart. This model lets task planners worry less about blocking communication and process interruptions and focus more on constructing algorithms from the available operations.

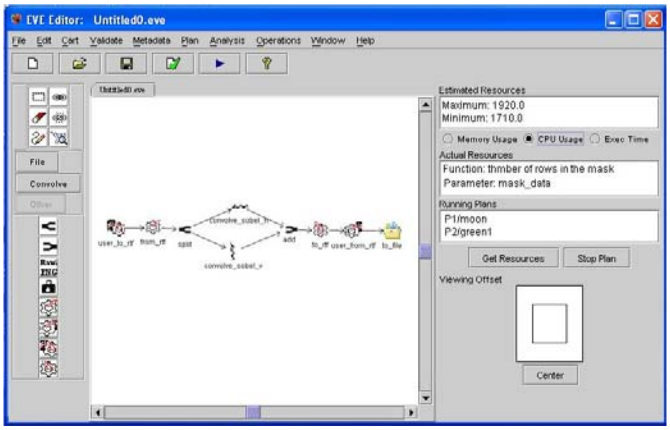
Another interesting benefit of the cart architecture is that it allows splitting up an entire plan into multiple sub-sections, with each sub-section having its own priority and resource usage profile and even its own targeted on-board platform. In this way the scientist designing the process can have extraneous but interesting portions of the process to be done on an auxiliary basis, without losing focus on the primary processing path and goals.

5 Editor

The EVE editor is a graphical web-based tool in which the user can develop their plans, combining operations together and grouping them into carts for real-time execution. They can also use the editor to upload, execute and monitor their plans.

The editor itself consists of a workspace for the manipulation of plans in a drag-and-drop manner, an area for the information on plan resource usage and monitoring, and a suite of editing tools and operations for user selection (Figure 3).

The operations provided to the user are stored in a library of prewritten routines. The user can choose operations by selecting associated graphical icons from the toolbar, and manipulating them within the editing workspace. When multiple operations have been placed within the workspace, the user can link them together, associating the output of one operation with the input of another, thus developing a process-flow for their plan. During this process-flow build up, the user can also specify the parameters associated with each operation.



Estimated Resources
Maximum: 1920.0
Minimum: 1710.0

○ Memory Usage ● CPU Usage ○ Exec Time

Actual Resources
Function: thmber of rows in the mask
Parameter: mask_data

Running Plans
P1/moon
P2/green1

Figure 3: The EVE Editor

To ensure plan correctness, the editor can perform several validations, including resources usage, connections between operations and data-flow checks. Once checked, the editor can generate a plan description file which can be used by the ground station and on-board platforms for execution.

As part of the plan validation, the editor can estimate the on-board resources a given plan will utilize. However, this is only an estimate. Once a plan has been uploaded to an on-board platform for execution, the editor can be used to monitor actual resource usage, providing the user with a tool to monitor their plan's progress.

While the EVE editor comes with a set of predefined operations, the team expects that users will want to add their own. Therefore, at any time the user can add and register new operations. This information will be incorporated into the editor, including the operation's parameters, how it is to be used, and what general group the operation should be place into. The user can also un-register existing operations, eliminating redundant, unnecessary or restricted operations from further use.

In addition to being able to open, edit, and save a given plan, users can also develop multiple plans, and edit them simultaneously. This, coupled with the ability to schedule and signal events, gives the user the ability to coordinate activities between plans.

6 Operations

To demonstrate the efficacy of the system, the team has implemented a few sample operations and inserted them into the EVE framework for use and testing. These have tended to fall into two categories: basic infrastructure tools and specific data mining tools. Basic infrastructure tools include support for streaming sensor data capture and storage, inter-cart communications, data splitting and data recombination. Operations that support data mining have thus far focused on the initial image processing steps, such as data filtering, edge detection and threshold signaling.

The current EVE testbed has live sensors in the form of video feeds, and several simulated sensors such as streaming data from NASA's Advanced Microwave Sounding Unit (AMSU-A) which is comprised of 18 frequency channels of Earth observation swath data.

Each sensor or simulated sensor requires its own operation for data capture. For the video system, the "vidop" operation is used for passing real-time image data through a plan one scan line at a time. For AMSU-A, another operation is available for reading data from disk and feeding it through a plan as streaming data.

Several operations are provided for inter-cart communication. These include operations for passing data between the real-time components and the non-real-time components as well as event notification and message passing.

Data splitting and recombination are provided so that multiple paths can be pursued. The split operation takes a data stream as an input and sends two copies of the data on two separate outputs, allowing parallel processing of the same information. The add operation takes two scan-line oriented inputs and combines the data by adding the pixel values, with an adjustable gain factor to aid in producing images that are easy to observe.

An example of filtering is an operation which takes AMSU-A swath data, extracts a single channel and thresholds it. Since different channels of data contain vastly different information due to frequency sensitivities, thresholding of several different channels can be of great help in feature extraction and event detection. Figure 4 shows the result of thresholding swath data from a full set of AMSU-A information.

Another filtering operation, convolution, has more general capabilities and is useful for a wide variety of image processing plans. The convolution operation is based on image input coupled with a convolution mask that determines the type of image processing to be accomplished. By changing the mask, many different frequency-domain manipulations of the data can be accomplished, such as blurring, sharpening, and edge-detection, as well as simulating near linear image degradation, such as motion blur or defocus.

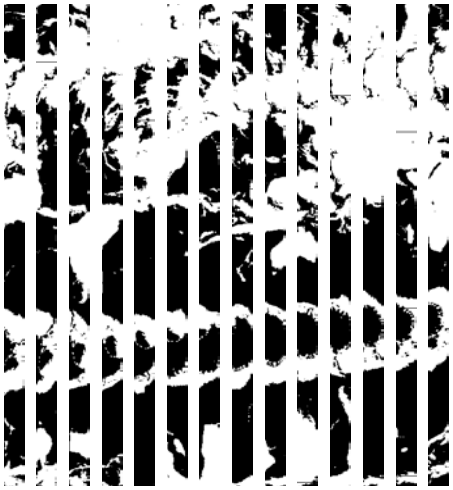


Figure 4: Thresholding Streaming AMSU-A Data

As an example, the split and add operations can be used in conjunction with two convolution operations for very fast edge detection. Using Sobel masks, one convolve can extract vertical edges while the other extracts horizontal edges. Taken together, the result is a combination of edges, which in turn can be used for downstream processing.

Figure 5 shows an example of real-time edge detection using the testbed's camera.

Two new operations currently under development detect activity in lightning point data. They're useful for finding areas of lightning activity either temporally or spatially.

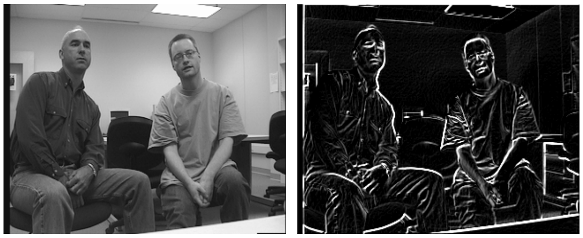


Figure 5: Real-Time Edge Detection

The first of these operations detects activity over a given time range, generating an event for each time interval in which a certain number of flashes occur. The parameters for this operation are latitude and longitude, a coverage radius, a start time and time increment, and a strike threshold. For example, starting at a given time with a time increment of 5 minutes, with a given latitude and longitude with a radius of 20 km and a threshold of 100 would cause messages to be generated for each 5 minute interval with more than 100 flashes occurring within 20 km of the location.

The second operation detects activity over that time range, generating an event for each location at which a certain number of flashes occur. The parameters for this operation are a latitude range, a longitude range, a start time and start increment, spatial granularities for latitude and longitude, and a strike threshold.

7 Software Engineering on EVE

The EVE team continues to follow a flexible software engineering process that proved to be successful during the first two years and is appropriate for this research development project. Early in the first quarter, the project's third year goals were defined and a high-level milestone schedule was developed. The team updated the system architecture and drafted a set of high-level requirements based on feedback from the second year reviews. A white board prototype led to hardware and software trade studies during the second quarter. With input from the EVE management team, the decision to port the distributed architecture to handheld computers with wireless networking capability was made. Existing software developer packages have been studied for suitability to the project and compatibility with the hardware. Software enhancements are being implemented on workstations within the EVE test bed environment, using existing software configuration management procedures. These products will be ported to and tested on the handheld target hardware during the third quarter. The test process consists primarily of scenario validation, user interface testing, capturing metrics, and regression testing. A technical review and preliminary capabilities demonstration will be convened late in the third quarter. Feedback from that demonstration will determine 4th quarter work towards product finalization.

8 Current and Future Project Plans

The EVE system is currently under development and testing. This effort will continue through the rest of the project, which ends in late 2003. During most of the remaining time, the emphasis will be on the use of EVE with full scale scenarios and a possible “flight of opportunity”, as well as support for sensor web and grid environments.

9 References

[1] Beaumont B., Helen Conover, and Sara Graves, 1996. “Information Systems Research at the Global Hydrology and Climate Center”, American Institute of Aeronautics and Astronautics Spaces Programs and Technologies Conference, September 24 - 26, 1996.

[2] Conover H. and Sara J. Graves, 1999. “Promoting Science Data through Innovative Information Systems”, American Geophysical Union, January 1999.

[3] Ramachandran R., H. Conover, S. Graves, K. Keiser, C. Pearson and J. Rushing, 1999. “A Next Generation Information System for Earth Science Data”, The International Symposium on Optical Science, Engineering and Instrumentation, Denver, 1999.

[4] Graves S., 1998. “Automating the Process of Information Extraction in Digital Libraries”, Panel Chair. IEEE Forum on Research and Technology Advances in Digital Libraries, Santa Barbara, CA, April 22-24, 1998..

[5] Keiser, K., J. Rushing, H. Conover and S. Graves, 1999. “Data Mining System Toolkit for Earth Science Data”, Earth Observation and Geo-Spatial Web and Internet Workshop (EOGEO)-1999, Washington, D.C., Feb 9-11.

[6] Graves S., 1998. “Creating an Infrastructure for Data Warehousing and Mining”, RCI-NASA Applications of Data Warehousing and Mining. Panel: Strategic Directions of Data Warehousing and Mining, Santa Barbara, CA April 20, 1998,..

[7] Stakem, P., “FlightLinux: A New Option for Spacecraft Embedded Computers”, 2001 Earth Science Technology Conference, Collage Park, MD, August 2001.

[8] Frank, J., A. Jonsson, R. Morris, D. Smith, “Planning and Scheduling for Fleets of Earth Observing Satellites”, Earth Science Technology Conference, College Park, MD, August 2001.

[9] Marinucci, T., A. Neelamegam, B. Tjaden, L. Tong, L. Welch, B. Goldman, G. Greer, D. Kaul, B. Pfarr, “Sensor Web Adaptive Resource Manager”, Earth Science Technology Conference, College Park, MD, August 2001.

[10] Tanner, S., H. Conover, S. Graves, K. Keiser, A Framework for Sensor Data and Product Processing, Workshop on Multi/Hyperspectral Technology and Applications, Redstone Arsenal, Alabama, February 6-7, 2002

[11] Tanner, S., K. Keiser, H. Conover, D. Hardin, S. Graves, K. Regner, and M. Smith, EVE: An Environment for On-Orbit Data Mining, IJCAI Workshop on Knowledge Discovery from Distributed, Dynamic, Heterogeneous, Autonomous Data and Knowledge Sources, Seattle, Washington, D.C., August 4-10, 2001.