Arquitectura de segmento terreno satelital adaptada para el control de límites de telemetría dinámicos

Pablo Soligo, Jorge Salvador Ierache

Universidad Nacional de La Matanza
Departamento de Ingeniería e Investigaciones Tecnológicas.
Florencio Varela 1903, B1754JEC San Justo, Buenos Aires.
psoligo@unlam.edu.ar
http://www.unlam.edu.ar/

Resumen El presente trabajo muestra la implementación de un prototipo funcional para el monitoreo del estado de salud de un satélite artificial científico. Aunque aplicable a otros sistemas, la implementación inicial ha sido probada utilizando como fuente de datos telemetría satelital y como prototipo inicial de desarrollo el Prototipo de Segmento Terreno de la UNLaM (UGS) derivado de la misión académica Formador Satelital 2017 (FS2017). Utilizando herramientas de estantería en cualquiera de sus variantes Off-the-Shelf (OTS), Commercial-Off-The-Shelf (COTS) u Open Source Software (OSS), el prototipo analiza los datos históricos para obtener patrones de comportamiento que permitan ajustar los límites de control y efectuar un monitoreo mas cercano y preciso de la telemetría, en tiempo real, manteniendo los costos operativos acotados y utilizando hardware de propósito general.

1. Introducción

Existen tres métodos extendidos para analizar la telemetría de un satélite y controlar el estado de salud; control de límites, sistemas expertos y sistemas basados en modelos [1].

Control de límites: El control de límites es el sistema más simple y difundido, consiste en establecer, con la ayuda de un experto, los valores máximos y mínimos aceptables para un sensor (temperaturas, voltajes, corrientes) o conjunto de sensores y verificar que los mismos no sean violados. Estos valores usualmente deben ser ajustados con el sistema en vuelo y la cantidad de sensores puede volver impracticable por este método el control de todos los sensores. La simplicidad vuelve a este sistema el mas popular, aunque se puede deducir que el mismo es absolutamente insensible al contexto.

Sistemas expertos: Durante los 80 y inicio de los 90 sistemas expertos han sido probados en diferentes misiones (Ej: GEOTAIL, NOZOMI y HAYABUSA). Estas técnicas si bien han mejorado el rendimiento de los sistemas

basados exclusivamente en control de limites no están exentas de problemas. Los sistemas expertos no pueden encontrar desperfectos desconocidos o no contemplados y requieren una laboriosa configuración y el conocimiento incorporado puede volverse inconsistente por cambios de diseño o por desconocer el comportamiento del satélite después del lanzamiento.

Sistemas basados en modelos: En esos sistemas la idea principal es detectar las anomalías y sus razones comparando simulaciones obtenidas por modelos computacionales contra el comportamiento actual del sistema.[1].

Cualquiera de los métodos mencionados requieren un asesoramiento casi continuo del experto en el área, ya sea actualizando límites, generando reglas o parametrizando simulaciones.
Por otro lado el aprendizaje automático ofrece un amplio rango de posibilidades para la predicción de comportamientos. Las técnicas de aprendizaje automático intentan servirse de la datos disponibles para predecir comportamientos futuros. En lugar de requerir a personal experto que infiera reglas y construya modelos, el aprendizaje automático ofrece una forma mas eficiente para capturar el conocimiento, ha salido de los laboratorios y hoy es utilizado en el reconocimiento de voz e imágenes, en la búsquedas en Internet, en el ofrecimiento de productos y las fronteras de aplicación todavía no están establecidas.
Específicamente en el área de machine learning y telemetría satelital existen algunos antecedentes que abren el camino a estos estudios, los trabajos de [2], [1] y mas recientemente [3] ofrecen alternativas distintas para la solución del mismo problema.
Los trabajos citados se concentran establecer conclusiones, mayormente prometedoras, de las posibilidades que tiene el aprendizaje automático en el monitoreo del estado de salud de un satélite sin especificar una implementación práctica, objetivo principal de este trabajo.

2. Antecedentes

Durante la experiencias obtenidas durante la primera cohorte de la Maestría en Desarrollos Informáticos de Aplicación Espacial (MDIAE) (2015-2017) se genero el primer prototipo funcional del segmento terreno ahora denominado UGS para la misión FS2017. Se utilizo como segmento de vuelto un modelo de ingeniería de cubesat de 2U de la compañía ISIS (`https://www.isispace.nl/`) [4], [5] y [6]. En particular para el control del estado de salud se utilizo un sistema de control de límites, durante la MDIAE se tuvieron experiencias con multiples sistemas del área espacial que hacen uso de esta técnica. Se han priorizado sobre las decisiones de diseño las recomendaciones que publicara en 2003 el Instituto Norteamericano de Aeronáutica y Astronáutica, del inglés Institute of Aeronautics and Astronautics (AIAA) [7], intentando cumplir los siguientes objetivos:

- Independencia de la arquitectura de hardware
- Independencia del fabricante del hardware
- Tolerancia a fallas

- Implementable en computadoras personales comerciales
- Independencia del sistema operativo

Con estas premisas y priorizando soluciones comúnmente utilizadas en los sistemas de propósito general se ha optado por:

1. Utilizar interpretes de propósito general para la decodificación de telemetría y los scripts de comandos en lugar de desarrollar interpretes propios (Solución extendida en el área espacial)
2. Utilizar base de datos relacionales (Sistema de base de datos relacional, del inglés Relational Database Management System (RDBMS)) y no relacionales para el almacenamiento de datos y metadatos.
3. Utilizar herramientas de máxima penetración para la distribución y ejecución de tareas.
4. Utilizar capas de acceso a los datos (Object Relational Mapper (ORM)) que faciliten la portabilidad entre motores de RDBMS y otras tecnologías de almacenamiento.
5. Utilizar protocolos de comunicación a nivel de aplicación comunmente aceptados en el ámbito informático de propósito general.

El UGS, desde su primera versión, opta por el uso de un lenguaje de propósito general para el procesamiento de telemetría como para la generación de scripts de comandos. Por popularidad [8], productividad y facilidad de uso la opción elegida fue python. Al tratarse de un interprete, se ha utilizado las capacidades de reflexión de software como herramienta para la carga dinámica de procedimientos y posterior el procesamiento en tiempo real de los datos satelitales [5] y [4]. Puntualmente respecto al uso de lenguajes de propósito general, ya en el 2008, trabajos en la industria privada explicitaban las ventajas de este enfoque, considerando: [9]:

Procesamiento: XML, SOAP, XTCE parsing
Comunicaciones: Sockets, Acceso a Internet, Llamada a procedimiento remoto, del inglés Remote Procedure Call (RPC), Correo Electrónico
Mediciones de performance: Contadores, acceso al hardware y sistema operativo.
Otros: Acceso a bases de datos, funciones matemáticas, compresión de datos, hilos, criptografía.

Procesar telemetría utilizando carga en tiempo de ejecución y reflexión de software y la opción de utilizar un lenguaje e interprete de propósito general habilitan a una fracción de costo la incorporación de bibliotecas para la minería de datos y el aprendizaje automático. Todo esto suma un argumento mas a la lista expresada en [9].

3. Arquitectura e implementación

Originalmente el primer prototipo desarrollado de segmento terreno desarrollado para el FS2017 [6] contaba de un servidor de base de datos (RDBMS), una

aplicación web y una serie de productos a ejecutar manualmente.
Actualmente el sistema trabaja sobre procesos distribuidos. Un subconjunto variable de procesos se ejecuta de manera distribuida dentro de la infraestructura de hardware disponible. Cada nueva ingesta de telemetría, independientemente de la fuente, dispara un proceso distribuido encargado de procesar el paquete, generar y persistir las variables de telemetría que el paquete contenga. Se generan todas las variables de telemetría que el satélite tiene configuradas, activas y se corresponden con el tipo de paquete. Actualmente el prototipo puede trabajar con telemetría de múltiples misiones incluida Satélite Argentino Científico - D (SAC-D) ([10]) utilizado en esta prueba. Desde su primera versión, el prototipo verifica el estado de salud del satélite mediante un control de limites (1), tanto para sus variables directas como derivadas, estas últimas son variables que se crean artificialmente a partir de la combinación de otras.
La adaptación mostrada en este trabajo propone crear una entidad asociada al tipo de telemetría donde almacenar un modelo de predicción automáticamente creado. La figura 1 muestra el diagrama de clases (Solo se muestran los campos sensibles al análisis) donde se pueden encontrar los campos para almacenamiento del modelo (Clase **TlmyPrediction**, atributo *data*) y fecha de expiración (Clase **TlmyPrediction**, atributo *expiration*), esta entidad esta asociada al tipo de telemetría **TlmyVarType**.

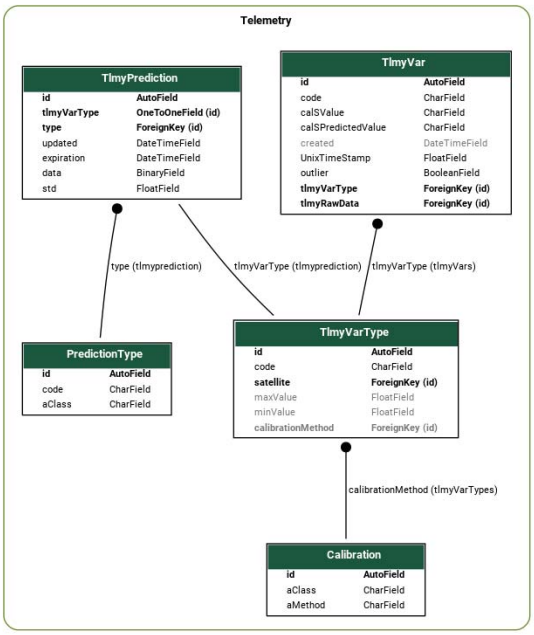


Figura 1: Diagrama de clases

La predicción puede no estar disponible, no ser conveniente o no presentar relevancia para el tipo de telemetría, por esta razón la relación no es obligatoria. Si se recibe un nuevo paquete de datos (independientemente de la fuente) y este contiene un tipo de telemetría determinado y este tipo tiene un modelo para la predicción, el mismo es aplicado modificando los limites máximos y mínimos de ese tipo de telemetría en un valor desplazado desde el valor predicho, positivamente y negativamente una distancia configurada. En las pruebas desarrolladas se ha tomado un σ (Sigma) como desplazamiento (1 y 2).

$$max = p + \sigma \tag{1}$$

$$min = p - \sigma \tag{2}$$

Estableciendo los máximos y mínimos dinamicamente en función de una predicción se pueden obtener controles sensibles al contexto actualizado del satélite.
En el caso de análisis de este trabajo se ha creado una variable derivada denominada **eclipseElapsedTime**, esta variable contiene el tiempo en segundos que el satélite permanece eclipsado o la cantidad de segundos (negativos) hasta el próximo eclipse. La figura 2 muestra las variable directa **vBatAverage** (Voltaje medio de batería) del SAC-D y la variable derivada **eclipseElapsedTime** calculada en *runtime* dentro del mismo sistema utilizando pyephem [11](Esta última variable podría ser provista por los servicios del área de determinación orbital en una agencia espacial [12]).
En la figura 2 se puede observar como la tensión decrece (variable **vBatAverage**) durante el eclipse, los paneles solares quedan parcial o totalmente tapados por la tierra durante algunos minutos.

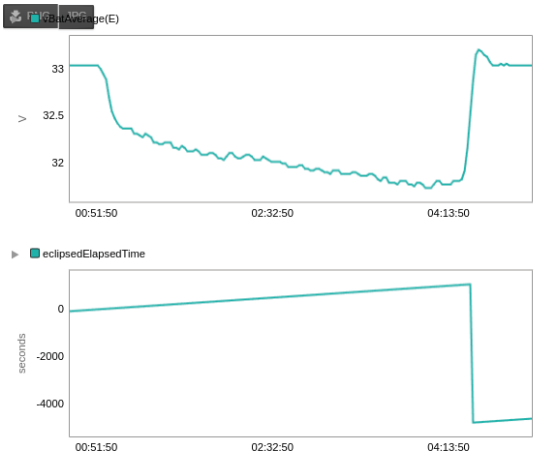


Figura 2: Caída de tensión al entrar en eclipse, ambos gráficos apilados muestran el mismo periodo de tiempo. La figura superior muestra el voltaje y la figura inferior la variable derivada **eclipseElapsedTime**.

Una vez que los paneles vuelven a tener la iluminación solar ocurre una rápida recuperación durante el periodo entre umbra y penumbra. La figura 3 muestra que el patrón se repite durante varios eclipses.

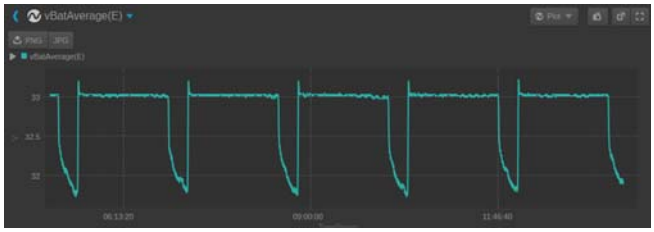


Figura 3: Variable directa **vBatAverage** visualizada por OPENMCT

Utilizando como técnica el control unicamente límites estáticos(1), para el ejemplo dado, se deberían establecer límites máximos y mínimos del orden de los 30v y 34v respectivamente. Mediciones de 31v durante los periodos entre eclipses no serian tomadas como una anomalía cuando en realidad se podría estar presentando un problema grave en la carga y un comportamiento anormal. Un conjunto de reglas explicitadas por un experto pueden salvar esta situación, aunque es necesario el experto y por otro lado los valores finales solo se conocen con el sistema en vuelo.
En el sistema propuesto, un proceso distribuido (Figura 4, Tareas Periódicas) revisa los tipos de telemetría que tienen modelos de predicción vencidos, es decir, superan su fecha expiración (propiedad *expiration*, entidad **TlmyPrediction**). Previamente ese tipo de telemetría fue configurado como sensible a predicción. Los modelos tienen una fecha de vencimiento que puede forzar su regeneración. Para cada tipo de telemetría en esta situación se realiza una nueva preparación de datos según configuración, se genera un nuevo modelo de predicción y en el caso de ser lo suficientemente preciso se persiste, caso contrario se informa mediante una alarma.
Al utilizar un lenguaje de propósito general para el procesamiento de telemetría se puede hacer uso de las múltiples bibliotecas disponibles para aprendizaje automático y normalmente existen herramientas para la persistencia (Almacenamiento en el algún medio permanente) de los modelos de predicción. Cuando un nuevo paquete de telemetría ingresa al sistema mediante la Interfaz de programación de aplicaciones, del inglés API: Application Programming Interface (API) se procesa el paquete y se llama a la función de calibración para cada uno de los tipos de variables de telemetría configuradas para ese paquete (Figura 4, Procesador de telemetría). Si el tipo de telemetría tiene asociado un modelo de predicción entonces este también se aplica y se utilizan el resultado para ajustar los máximos y mínimos tolerables, si el valor calibrado esta por fuera de los limites dinámicos entonces se genera una alarma siguiendo el flujo establecido en el sistema.

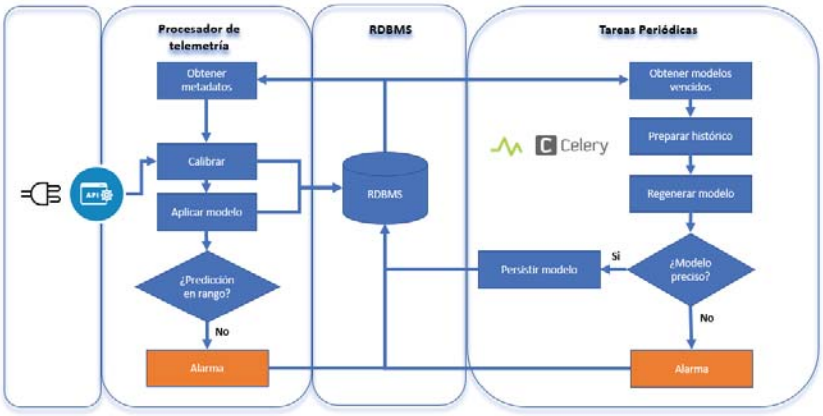
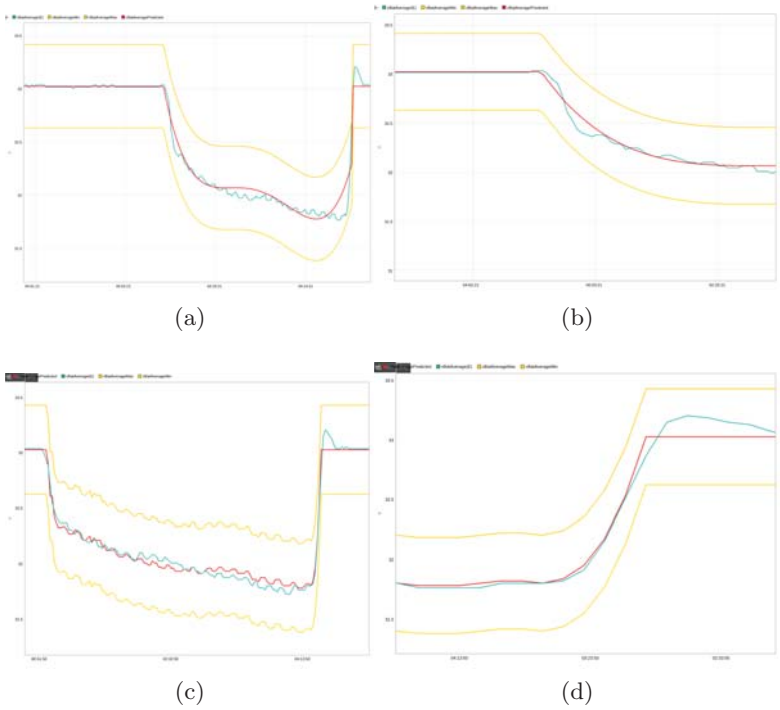


Figura 4: Arquitectura conceptual para la verificación de límites en la telemetría

4. Resultados

La figura 5 muestra los resultados aplicando regresión polinómica y aplicando un árbol de regresión lineal (Biblioteca scikit-learn [13]). En ambos casos los valores predichos están dentro del rango de un σ establecido. Se puede observar tanto en los periodos entre eclipses como dentro del eclipse como los máximos y mínimos se ajustan permitiendo un control mas estricto y por sobre todo, sensible al contexto.

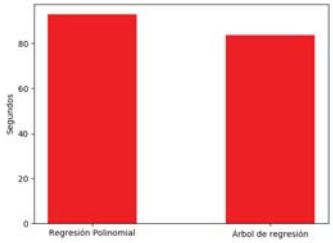


(a) (b)

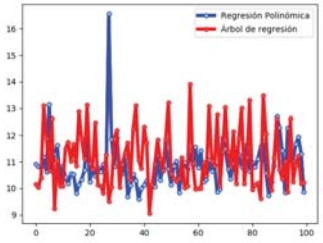
(c) (d)

Figura 5: (a) Predicción utilizando regresión polinómica (Eclipse completo) (b) Predicción utilizando regresión polinómica (Entrada en eclipse) (c) Predicción utilizando árbol de regresión (Eclipse completo) y (d) Predicción utilizando árbol de regresión (Salida del eclipse), donde celeste es valor real, rojo valor predicho y amarillos máximos y mínimos

Los máximos y mínimos son ahora sensibles al contexto, entendiendo por contexto a la variable derivada que indica el periodo entre eclipses en segundos. La figura 6 (a) muestra los tiempos de generación del modelo incluyendo la búsqueda de los datos (12 horas), la preparación y persistencia de los datos para alimentar el algoritmo y la generación y persistencia del modelo en sí.
La figura 6 (b) muestra (En milisegundos) los tiempos necesarios para la generación, utilizando el modelo persistido, del valor predicho y el ajuste de los limites para la variable **vBatAverage**. Se muestran los tiempos para 100 paquetes SAC-D, los tiempos no contemplan el control de límites que es independiente de si el límite es modificado dinamicamente o establecido por configuración.



(a) Tiempo de generación del modelo de predicción



(b) Tiempo en milisegundos para la predicción de variables **vBatAverage** pertenecientes a satélite SAC-D (100 muestras).

Figura 6

5. Conclusiones

El presente trabajo muestra una propuesta de implementación que permite aplicar las herramientas contemporaneas de aprendizaje automático a un segmento terreno desarrollado bajo la premisas expresadas en trabajos anteriores [6], [5] y [4]. La capacidad añadida no implicó modificación alguna al núcleo del segmento terreno original y su implementación es transparente. Como se mostró en las figuras 6 (a) y 6 (b), sin generar un coste relevante en cuanto a tiempo de procesamiento, la idea permite un control mas cercano de la telemetría de housekeeping sin necesidad de intervención humana durante la operación. El costo y riesgo de este tipo de misiones justifican el desarrollo estas técnicas y suman un argumento adicional a favor del trabajo con herramientas de propósito general en el área espacial .

6. Trabajo a Futuro

La propuesta presentada en este trabajo no esta exenta de problemas o puntos a profundizar. La búsqueda de relaciones, o variables contextuales aun requieren del asesoramiento del un experto para su correcta identificación. En próximos trabajos se abordará esta problemática en función de encontrar correlaciones y explotarlas de manera automática, buscando explotar tanto como sea posible el poder de computo actual, no como remplazo del experto, sino como un aliado en la búsqueda e identificación de patrones ocultos.
Por otro lado, aunque se ha contemplado la eliminación de *outliers* durante la preparación de los datos de entrenamiento, la aparición de estos durante la operación y su impacto en la generación de falsos positivos no ha sido tenida en

cuenta en esta primera versión, la incorporación de un filtro que detecte *outliers* en tiempo real esta pendiente.

Referencias

1. Takehisa Yairi, Yoshinobu Kawahara, Ryohei Fujimaki, Yuichi Sato, and Kazuo Machida. Telemetry-mining: a machine learning approach to anomaly detection and fault diagnosis for space systems. In *Space Mission Challenges for Information Technology, 2006. SMC-IT 2006. Second IEEE International Conference on*, pages 8–pp. IEEE, 2006.
2. Takehisa Yairi, Minoru Nakatsugawa, Koichi Hori, Shinichi Nakasuka, Kazuo Machida, and Naoki Ishihama. Adaptive limit checking for spacecraft telemetry data using regression tree learning. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, volume 6, pages 5130–5135. IEEE, 2004.
3. L.Amoruso C.Brighenti, A.Brighenti D.Evans, M.Ricatto D.Moretto, and M.Carbone F.Ferrari. Advances in context aware spacecraft telemetry checking. In *2018 IAC*. IAC, 2018.
4. Pablo Soligo and Jorge Salvador Ierache. Segmento terreno para misiones espaciales de próxima generación. *WICC 2019*.
5. Pablo Soligo and Jorge Salvador Ierache. Software de segmento terreno de próxima generación. In *XXIV Congreso Argentino de Ciencias de la Computación (La Plata, 2018).*, 2018.
6. Pablo Soligo, Ezequiel González, Eduardo Sufán, Emmanuel Arias, Ricardo Barbieri, Pablo Estrada, Alfonso Montilla, José Robin, Javier Uranga, M Cecilia Valenti, et al. Misión cubesat fs2017: Desarrollo de software para una misión satelital universitaria. *WICC 2017*, page 843.
7. Ken Galal and Roger P Hogan. Satellite mission operations best practices. 2001.
8. TIOBE Software. TIOBE programming community index, september 2017, 2017. [Online; accessed 26-September-2017].
9. Gonzalo Garcia. Use of python as a satellite operations and testing automation language. In *GSAW2008 Conference, Redondo Beach, California*, 2008.
10. Amit Sen, Yunjin Kim, Daniel Caruso, Gary Lagerloef, Raul Colomb, Simon Yueh, and David Le Vine. Aquarius/sac-d mission overview. In *Sensors, Systems, and Next-Generation Satellites X*, volume 6361, page 63610I. International Society for Optics and Photonics, 2006.
11. Brandon Craig Rhodes. Pyephem: astronomical ephemeris for python. *Astrophysics Source Code Library*, 2011.
12. S. Hackel, Y. Wasser, M. Meinel, and R. Kahle. Flight dynamics microservices. In *Proceedings of the International Astronautical Congress, IAC*, volume 2018-October, 2018.
13. scikit-learnmachine learning in python. `https://scikit-learn.org/stable/`. Accessed: 2019-23-07.