A Geospatial Complex Event Processing Engine for Abnormal Vessel Behavior Detection Suitable for Maritime Surveillance

Manolis Tsogas*[a], Polyzois Parthymos[a], Marios Moutzouris[a], Nektarios Patlakas[a], George Karagiannis[b], Antonis Kostaridis[a], Dimitris Diagourtas[a]
[a]Satways Ltd., 3 Christou Lada, Athens, Attiki, Greece 15233; [b]Hellenic Navy General Staff A4/IV, Hellenic Ministry of Defense, 227 Mesogeion, Attiki, Greece 11525

ABSTRACT

Modern maritime surveillance systems provide a huge amount of data for near real time processing due to the large number of ships sailing across the seas. Thus, it is necessary to be able to handle large amounts of information accurately and efficiently. This can be done by utilizing a Complex Event Processing (CEP) engine which can process streams of data available from different sensors. In this paper, we present an advanced computational rule-based engine (TRITON) that analyzes ship positions for the automatic detection of abnormal and/or specific vessel behavior following the CEP paradigm. The purpose of the system is to support maritime authorities' surveillance functions, by providing an enhanced situational picture in near real time. The system is able to analyze ship position reports using data from available tracking systems such as Maritime radars, Long Range Identification and Tracking (LRIT) systems, Terrestrial-AIS, Satellite-AIS, Vessel Monitoring System (VMS) and Earth Observation satellites, among others. Patterns, such as entering an area of interest, encounters at sea, approaches to shore, drifting and deviations from usual routes, are detected and operators can be automatically alerted in real time. The system reasons on over 20 different rules and is designed in such a flexible way that can be expanded very easily according to user needs.

Keywords: Complex event processing, geospatial, maritime, abnormal behavior, patterns, anomaly detection

1. INTRODUCTION

Vessel Traffic Services (VTS) are used for maritime traffic monitoring and are operated by harbor, port or national authorities in order to ensure that ships moving in areas of interest are following established traffic and safety rules. Over the last years, instead of using human operators to detect abnormal vessel behavior, which is a difficult task considering the huge amount of vessel traffic, it is more efficient to use automatic detection systems that are less prone to errors. Maritime domains that can benefit from such systems include safety, when detection and early warnings for vessels posing a potential risk [1], arriving at specific areas or following unusual routes are of great importance. Also, another domain is security, when it is necessary to track vessels involved in illegal activities, smuggling prohibited substances, or breaking embargos. Border protection is also another interesting field, where vessels crossing sea borders transferring goods or persons illegally must be detected. Finally, detection of illegal fishing activities within regulated zones is something that requires attention considering the depleting stocks.

Information flow processing models have become an important approach in order to cope with time constraints in a wide range of environments [2]. Complex event processing was originally developed as a set of tools and methodologies for detecting situations of interest under high throughput data streams in applications such as network intrusion detection, industrial control systems [3] and real time analytics. Over the last years, huge volumes of data are continuously generated due to the increasing number of applications especially in the maritime environment [4]. Therefore, efficient methods are required to determine the event patterns of interest and manage highly dynamic events in real-time. A methodology for applying CEP in the maritime environment is described in [5] where the authors have created a basic rule-based system using an event model for handling incoming and outcoming information for detecting simple anomalies produced from ships.

*m.tsogas@satways.net; phone +30 210 6840036; fax +30 210 6840037; satways.net

In this paper we present a novel abnormal vessel behavior detection service, which is called Triton and is implemented using a Geospatial Complex Event (GCEP) processing architecture, in order to identify and analyze motion patterns of vessels that indicate an ongoing situation that needs attention. Geospatial complex event processing combines geospatial information and detections from different and distributed sources of information to infer events or patterns that describe the current situation. It offers flexibility in applying rules only in specific areas of interest or in specific vessels according to their type or identifier and it allows to combine rules in order to identify more complex motions. Thus, the user can focus exactly where it is necessary, less processing power is required, and a broader set of abnormal vessel behavior patterns is supported.

2. DETECTION OF ABNORMAL BEHAVIOR

2.1 Complex Event Processing

Complex event processing (CEP) is the method for processing large volumes of incoming events in real time and detecting when there is an abnormal behavior using specified rules. In a system implementing CEP, events are arriving in a streaming fashion and the goal is to detect certain patterns defined by domain experts. Typical examples of applications where CEP can be used are in the financial sector for fraud detection, algorithmic trading and risk management. Also, it can be used in network and application monitoring for the detection of intrusion attempts and monitoring of service-level agreement (SLA). In the business process management and automation, it can be used for relationship management, workflow applications, process tracking, operational intelligence and reporting exceptions. And finally, it is useful for sensor network applications such as RFID reading, scheduling and control of production lines, analysis and management of public transport, air traffic and maritime surveillance.

The open source Esper platform was used [6] to provide native complex event processing capabilities during the implementation of the TRITON service. Esper offers low latency and high throughput, compliance with standards and is light-weighted in terms of memory, CPU and IO-usage. It is highly scalable, memory efficient, with low latency and the most important is real-time streaming-capable so it can be used for online and real-time arriving data and high-variety data, as well as for historical event analysis. Esper's compiler and runtime are not limited to running on a single machine and can be part of a distributed stream processing framework. Both modules can run in any architecture and any container, as they have no dependencies on external services and do not particularly require any threading model or model of how time advances and do not require any external storage. Also, Esper offers a scripting language which is called Event Processing Language (EPL), that implements and extends the SQL-standard and enables rich expressions to be applied in events over time. EPL works well with event-time and watermark-based time management.

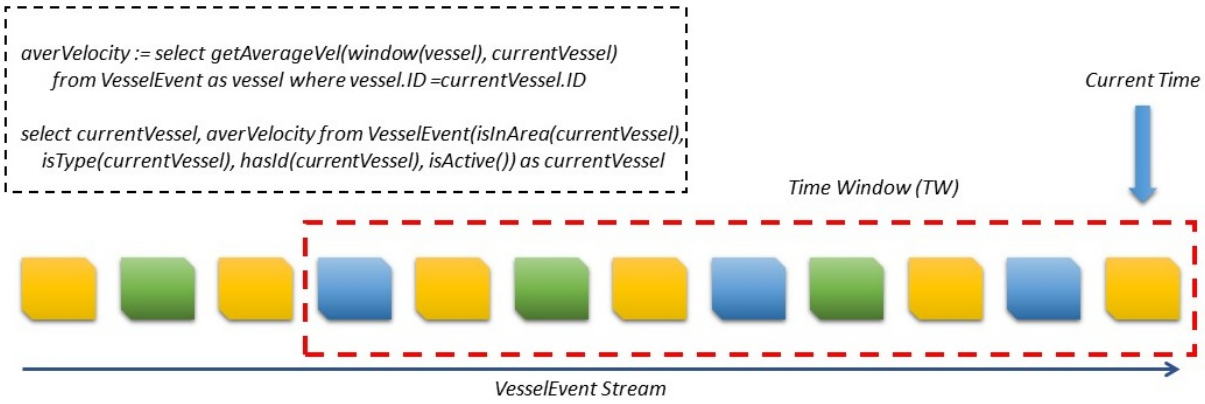

Figure 1. Rule example using EPL language

Figure 1 shows a rule example using the Event Processing Language to form a query in order to calculate the average velocity for every vessel. Every update in ship's state is represented by a new 'Vessel' event (ship ID is distinguished using different color in events). In order to find the average velocity a subquery (averVelocity) has been created, which

filters out only the vessels having the same id as the current one, using a time window in the past. A custom aggregation function (getAverageVel) has been created for estimating the average velocity using the filtered events. Finally, this subquery is part of a standard 'select' query that has been created for reading from the stream only events for vessels located in the user-specified area (isInArea) or having the requested id (hasId) and when this rule is scheduled to run (isActive).

2.2 Geospatial Complex Event Processing Service (TRITON)

TRITON is a service that processes data in real time providing enhanced situational awareness and fast response in critical situations. This is accomplished by producing alerts from various patterns which are manually or automatically enabled. The maritime domains in which this service can be applied include border control, counter smuggling, counter human trafficking, counter piracy, counter terrorism, naval and joint operations, traffic safety, crisis management, maritime search and rescue, pollution and safeguarding of the environment, protection against illegal fishing and port control.

The service can detect patterns and relationships between events using the EPL language. EPL provides out-of-the-box capabilities like pattern matching and detection, filtering, transformation, aggregation, event hierarchies, detecting relationships, thus it allows easier modelling of rules and detection of abnormal vessel behavior, when applied to incoming data streams of AIS, radar and fused tracks. Furthermore, a great number of custom aggregation functions have been implemented using the JTS Topology Suite which provide TRITON with geospatial capabilities and advanced features for handling complex scenarios. The user can select and apply multiple rules in specific vessels (selected using their type or their identification number), routes or sea areas or combine rules for more complex situations. The core of the service provides multiple algorithms such as Bayesian Inference Systems for utilizing prior knowledge.

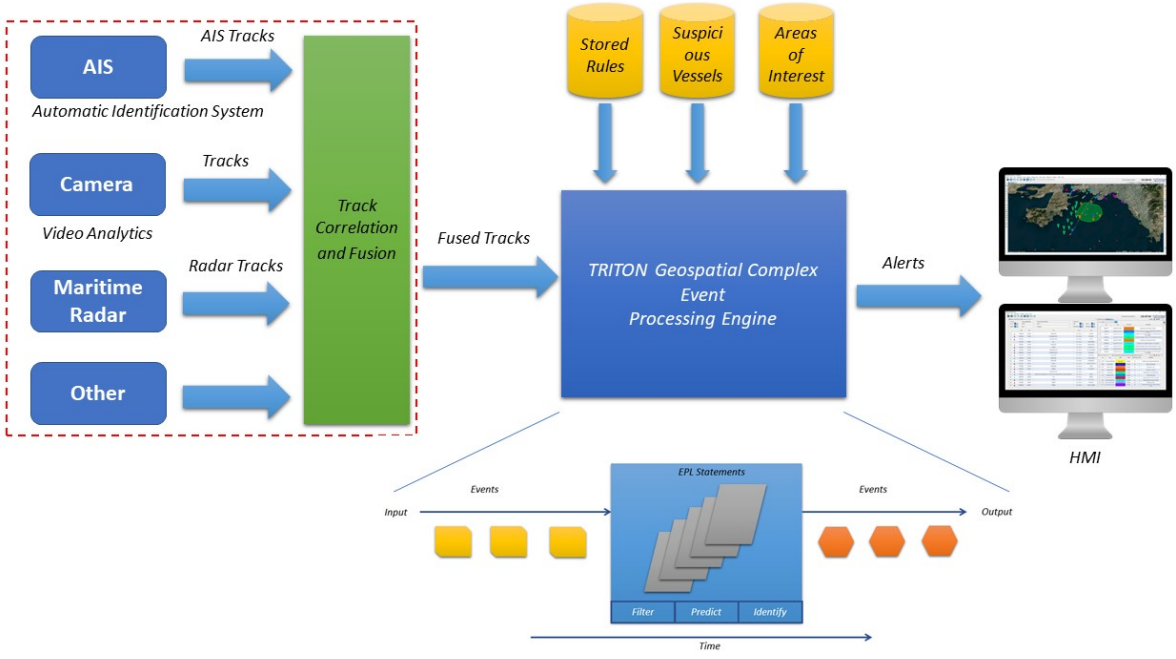

Figure 2. Architecture of Triton Service

The architecture of the engine is shown in figure 2. The input consists of vessel information (position, velocity, heading, identity, dimensions, etc.) received from various sources of information, like the Automatic Identification System (AIS) and maritime radars. If necessary, information about suspicious vessels can also be retrieved from the database, which also provides the coordinates of all the areas, borders and locations which are useful for identifying vessel behavior patterns in respect with their position in the sea environment. Due to the multiple sources of information it is necessary

to apply a fusion preprocessing step in order to correlate detections from different sensors and acquire a reliable and more accurate vessel state. Then, each vessel update is considered as an event, which feeds the input stream of the engine. The next step is to process the event stream by applying a set of rules, which define specific patterns to be detected, according to user preferences. The operator can enable and disable rules according to the current situation and apply them to specific zones in the monitored sea environment or exclude specific zones that are of no interest, through the user interface.

2.3 Abnormal Patterns

A list of patterns indicating abnormal or unwanted behavior from vessels can be found in [7]. TRITON provides a wide range of rules that can be used to detect those anomalies. Existing rules associated with geofence operations (Figure 3) include detection of vessels entering, exiting, crossing, moving away or approaching areas. Also, crossing, approaching or moving away from borders is supported in addition to approaching or moving away from single location points. These rules allow the user to completely monitor traffic near critical infrastructures like ports and drilling platforms, or detect vessels crossing country borders for disrupting irregular immigration. Finally, it is possible to detect a vessel which is approaching a specific area and has departed from a specific port or zone.

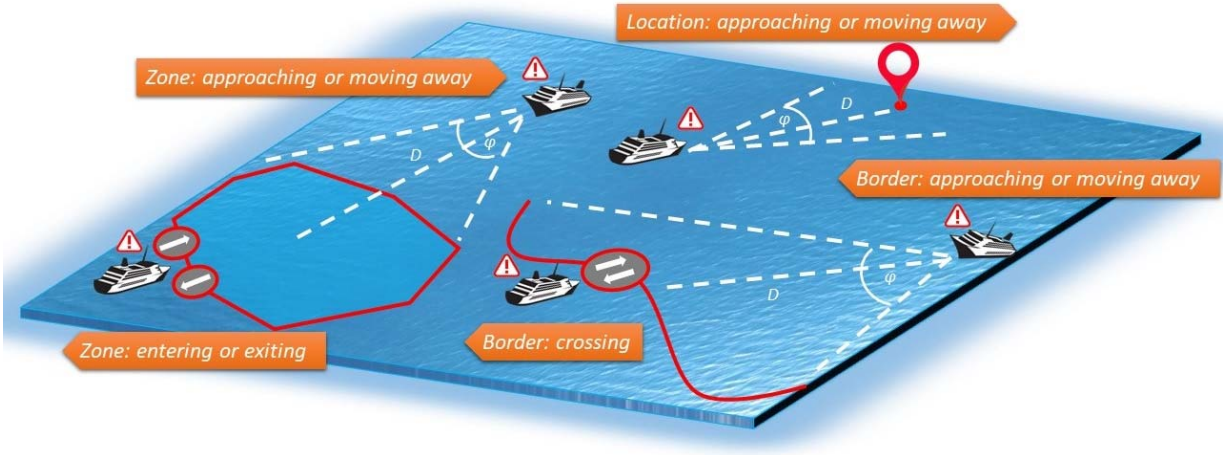

Figure 3. Geofence rules supported from Triton

Furthermore, TRITON provides a set of rules for recognizing specific patterns related with vessel interaction, such as a small boat approaching a larger vessel, two vessels sailing together in parallel, vessels rendezvousing and vessel continually crossing the route of another vessel. These rules are useful for identifying illegal activities such as smuggling scenarios. Also, rules related to ship safety, such as detection of an imminent collision and vessel domain violation are also provided, when it is necessary to know if two ships are too close together or following their current course will end up in a collision between them. There are also rules helping operators to identify possible spoofing or suspicious scenarios such as track splitting, inconsistent kinematic attributes, space-time incompatibilities, suspicious identity, AIS transmission stopped and contact lost. Finally, other rules provided by the TRITON include the detection of abnormal speed or course change, exceeding speed limit, vessel having an average velocity above or below user-defined threshold, detection of loitering ships and deviation from shipping lanes.

But the most important feature of TRITON is that it offers the ability to detect complex scenarios using combined rules. For example, a smuggling scenario would include the following actions. A cargo vessel transmitting AIS information is approaching a designated area near an island, which is isolated and where similar illegal actions are known to be taking place. Before the ship alters its course, it switches off its AIS transmitter (it is still detected by a radar station) and then it is making an abnormal course change and heading to the shore. When it is sufficiently near the coast it halts and is waiting for a small boat which is approaching it. The boat is rendezvousing with the cargo vessel, they stay side by side, completely stopped (transfer of illegal items may be taking place) and then the boat departs for the coast. The same

procedure may be repeated multiple times or several small boats to be used for transferring items to the shore. When their intended actions are completed, the cargo vessel departs and resumes its initial route, switching on its AIS transmitter. Using TRITON, one would be able to set multiple rules, which cover a specific test case like the one described, and to trigger a single alarm when this complex pattern is detected. In our case, the user should create a complex rule consisting from a geofence rule for approaching a specific area, a geofence rule for entering an area, an abnormal course deviation rule, a rule for AIS transmission stopped, a loitering rule and a small vessel approaching a bigger vessel rule.

3. RESULTS

3.1 Simulating data

TRITON service has been evaluated using large scale simulated data for reconstructing vessel traffic in Greek territorial waters. For this purpose, a simulator was developed as an extra service which have been adapted in the input of the Complex Event Processing Engine. The simulator can produce track updates based on predefined routes which are loaded from a database or created from the user of the service. Also, it is able to produce both AIS vessels (refresh rate is not constant and usually low) or vessels simulating radar detections (constant high refresh rate).

At the end, the simulator was able to produce traffic for all kinds of ships, such as passenger vessels travelling from major mainland ports (Piraeus) to Greek islands, cargo vessels and tankers crossing the Aegean and following the main trading shipping lanes, pleasure crafts and sailing boats making short trips from the Greek islands, fishing boats following random paths in specified areas and unknown contacts from radar stations. In order to increase the traffic load directed in the event processing service, the simulator was able to produce also stationary random vessels in all Greek ports. For the purpose of the evaluation we have used publicly available routes of Greek ferry boats around Greece and we have created additional routes for representing the trading shipping lanes, the paths of pleasure crafts and the random paths of fishing boats.

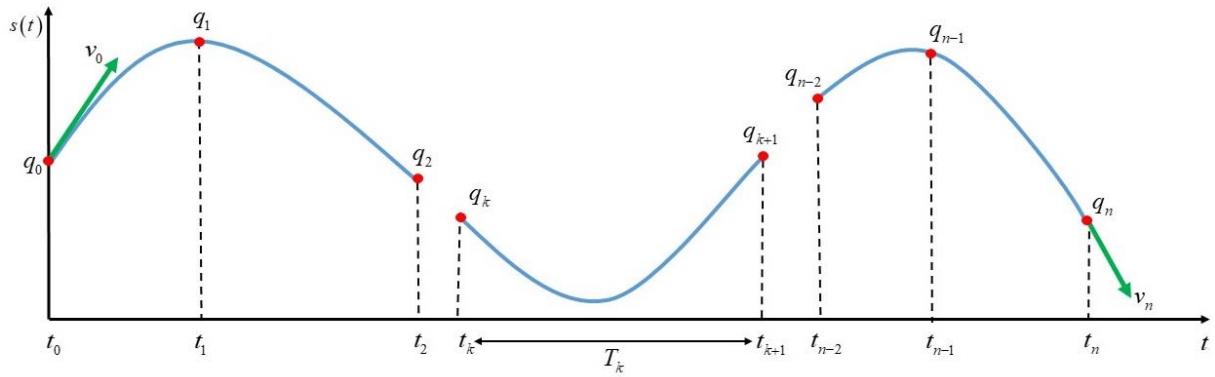

Figure 4: Definition of spline for interpolating ship trajectory

The core engine of the simulator is using an interpolation method based on cubic splines [8] in order to reconstruct the motion of a vessel following a specific path which is provided as a set of points in space (route). When $n+1$ points are given it is possible to use n polynomials of degree p, each one defining a segment of the trajectory. The overall function $s(t)$ defined in this manner is called spline of degree p. The desired degree of continuity between segments dictates the value of p. In our case, we want to ensure the continuity of velocities and accelerations at the time instants t_k, where the transition between two consecutive segments occurs. In that case it is sufficient to use polynomials of degree p =3 (cubic spline).

$$s(t) = \{q_k(t), t \in [t_k, t_{k+1}], k = 0, \ldots, n-1\} \quad (1)$$

$$q_k(t) = a_{k0} + a_{k1}(t - t_k) + a_{k2}(t - t_k)^2 + a_{k3}(t - t_k)^3 \quad (2)$$

So, in order to estimate the overall trajectory, it is necessary to compute 4 coefficients for each polynomial, which means $4n$ coefficients for the whole trajectory. In order to solve this problem, the following conditions must be considered: $2n$ conditions for the interpolation of the given points, since each cubic function must cross the points at its extremities, $n-1$ conditions for the continuity of the velocities at the transition points and $n-1$ conditions for the continuity of the accelerations at the transition points. In this way, there are $4n-2$ conditions and therefore the remaining degrees of freedom are 2. So, two more conditions are necessary for computing the final spline, which were selected to be that the initial and final velocities should have fixed values selected by the user $\dot{s}(t_0) = v_0$ and $\dot{s}(t_n) = v_n$. Among other possible choices, one can assign the initial and final accelerations to fixed values, set the final velocity and acceleration equal to their initial values (cyclic conditions for periodic splines) or finally ensure the continuity of the jerk at every time instance t_k. Eventually, after gathering all conditions the following equation must be solved:

$$A(T) \cdot v = c(T, q, v_0, v_n) \quad (3)$$

where $T = [T_0, T_1, \ldots, T_{n-1}]^T$ are the time lengths of each segment, $q = [q_0, q_1, \ldots, q_n]^T$ are the points forming the selected route (x or y coordinates) and $v = [v_1, v_2, \ldots, v_{n-1}]^T$ are the velocities to be found. The $(n-1)\times(n-1)$ matrix $A(T)$ has a diagonal dominant structure and therefore it is always invertible if $T_k > 0$. Once, the inverse of A has been computed then velocities can be calculated from $v = A^{-1} \cdot c$ and therefore the problem is solved and an analytical form (position, velocity and acceleration in every desired time instance) of ship's trajectory can be found using spline polynomials (1), (2) and their derivatives.

It should be noted that since the initial selected ship route is defined using geographic coordinates, there is a transformation of each point (longitude, latitude) from the global coordinate system to the local tangential coordinate system. The interpolation process is repeated twice, one time for all x-coordinates and one time for all y-coordinates and in the end the output contains the two-dimensional positions (x, y) of the ship through time alongside with its velocity and acceleration in each position. Coordinates can be then transformed back to the global coordinate system. The output time is calculated using the following formula:

$$t_i = \begin{cases} i \cdot dt & vessel\_type = radar \\ t_{i-1} + dt_i & vessel\_type = ais \end{cases} \quad (4)$$

where dt is the refresh time in seconds. If the simulated vessel is a radar contact, then dt is constant and is set to 0.5 or 1 seconds. If the simulated vessel is received from an AIS station then dt_i is not constant, but it is approximated as a gaussian random variable with mean and standard deviation selected by the user.

3.2 System Architecture

Both services (TRITON and the simulator) are implemented as Java 8 standalone applications. They are using Jboss to access (read and write data) to the database and ActiveMQ for messaging purposes between them and also the graphical user interface. PostgreSQL is used as the main database, which is optimized for storing and querying data that represent objects defined using geospatial information.

The complete system is deployed inside a Virtual Machine (VM) running Debian as the operating system. The tightly isolated software container with an operating system and applications inside, ensures that development can be done in a controlled environment and then the whole system to be deployed in the final server without worrying about compatibility issues. Also, this way the computer resources assigned to TRITON VM can be modified according to the current needs. Furthermore, updates can be applied more easily without affecting the other applications running on the server, which may be VMs hosting other components of the system. The communication with other modules is accomplished using an adaptor, which translates data from the internal TRITON format to the one used from the overall external system. For example, TRITON has been deployed in various sites for the purpose of the Marisa Project [9]. In

this case, Kafka is the messaging server for communication purposes within the server, so an adaptor has been developed, which implements a bidirectional translation of information between Kafka and ActiveMQ. This way TRITON can be used in any system by spending minimal effort in developing the appropriate translation unit. In our case, for the purposes of this paper, data are generated by the simulator within the VM, but it is possible to provide both real data (using the appropriate adaptor) and simulated vessels at the same time, so it is possible to examine more complex scenarios in real conditions.

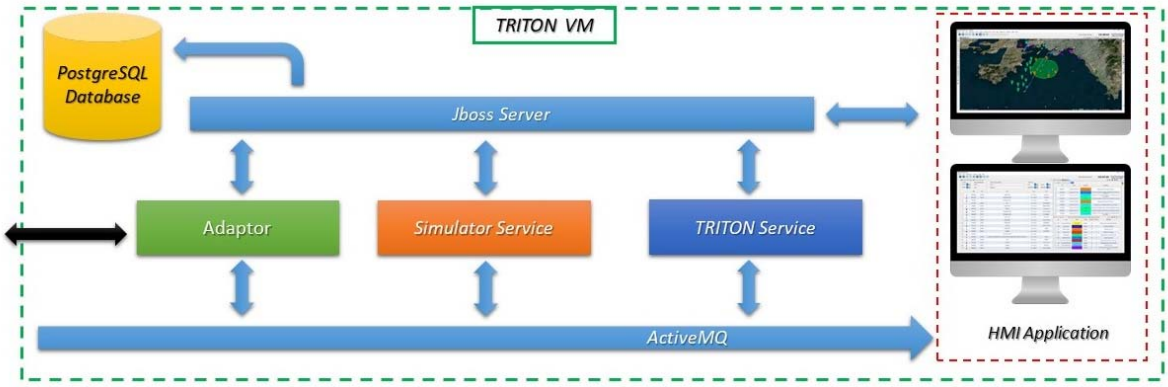

Figure 5: Internal architecture of TRITON Virtual Machine

3.3 Evaluation of service

In order to evaluate the service more accurately, the simulator was designed to generate vessels in random fashion according to the following rules. The total number of vessels that are created in every port, are uniformly distributed in the space $[0\ldots15]$. Their initial position (longitude, latitude) is selected using a Gaussian distribution with mean value the longitude and latitude of the port respectively and standard deviation equal to 0.025 degrees in both axes. The type of the vessel is selected randomly according to a uniform distribution of all the available vessel types as defined in [10]. The following types -radar, mobile offshore drilling unit and nuclear ship- have also been included. Their position, heading and velocity are considered as Gaussian random variables with mean and standard deviation set by the user and are estimated accordingly in each update of the simulator. As far as it concerns the ferry routes, the generated vessels can only be passengers or high-speed vessels (if the total travelled distance is big enough). The selection of the type is also done randomly. The initial position of the vessel is selected by considering the starting point of each segment of the route as uniformly distributed and selecting a random one, from which each vessel begins its trip. Initial and final velocities are selected randomly within the range $[3\ldots8]$ Knots. The AIS refresh rate is considered as a Gaussian random variable with mean and standard deviation set by the user. The same approach is also used for the routes occupied by pleasure crafts or sailing boats and the routes for the fishing vessels. As far as it concerns the radar tracks, the only change is that the refresh rate is constant (1 sec), their initial velocity is randomly selected in the range $[5\ldots8]$ Knots and their final velocity randomly selected in the range $[8\ldots12]$ Knots. Finally, in the four trading shipping lanes the generated vessels are either cargo ships or tankers and the generated vessels are 100 per direction. The rules applied are the same as in the passenger ships except for the initial velocity which is randomly selected in the range $[12\ldots14]$ Knots and the final velocity which is randomly selected in the range $[16\ldots19]$ Knots.

The total number of vessels generated by the simulator for each type of vessel is shown in figure 6. The total number is 2378 vessels of all types, while the real traffic is no more than 700 vessels around Greece. The passenger and high-speed vessels (294), cargo vessels and tankers (1130), pleasure crafts and sailing boats (80), fishing boats (50) and radar tracks (44) were the ships of interest (non-stationary) because most of the rules applied would track their behavior.

The evaluation of TRITON service took place using a laptop with 16GB of RAM and an i7-5500U CPU running at 2,4GHz with 4 logical dual core processors. Statistics about the CPU and Memory Heap utilization was performed using

the Java VisualVM. Also, the total number of threads running during the whole test, was recorded (64 threads allocated by the service).

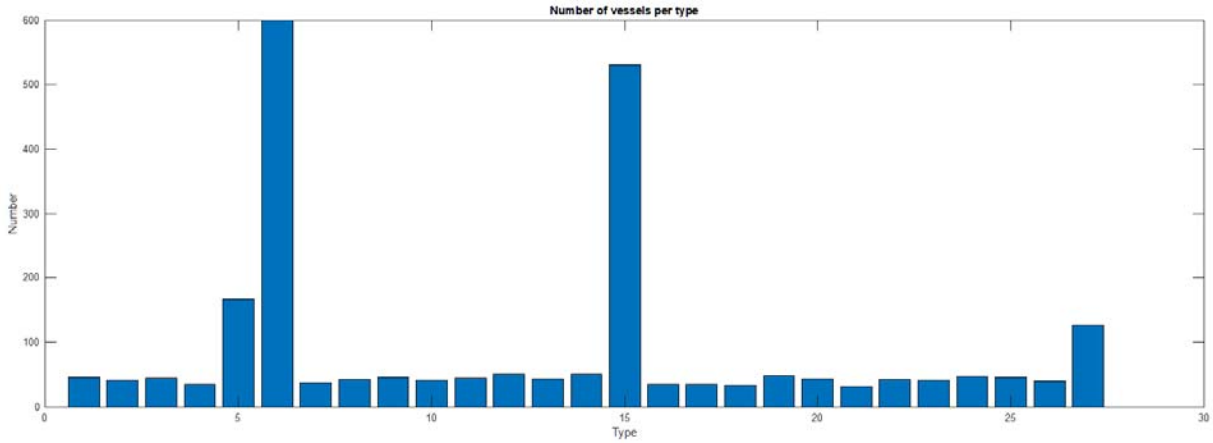

Figure 6: Total number of generated vessels per type. 1=pleasure craft, 2=not available, 3=radar, 4=vessel, 5=passenger, 6=tanker, 7=anti-pollution, 8=search & rescue, 9=tug, 10=port tender, 11=wig, 12=fishing, 13=ship No.18, 14=underwater operations, 15=cargo, 16=military, 17=sailing, 18=mobile offshore drilling, 19=pilot, 20=towing, 21=spare for assignment to local vessels, 22=law enforcement, 23=nuclear, 24=medical, 25=diving operations, 26=other, 27=high speed

The simulation lasted for 6 hours and 25 minutes. The CPU and memory utilization are shown in the next figures.

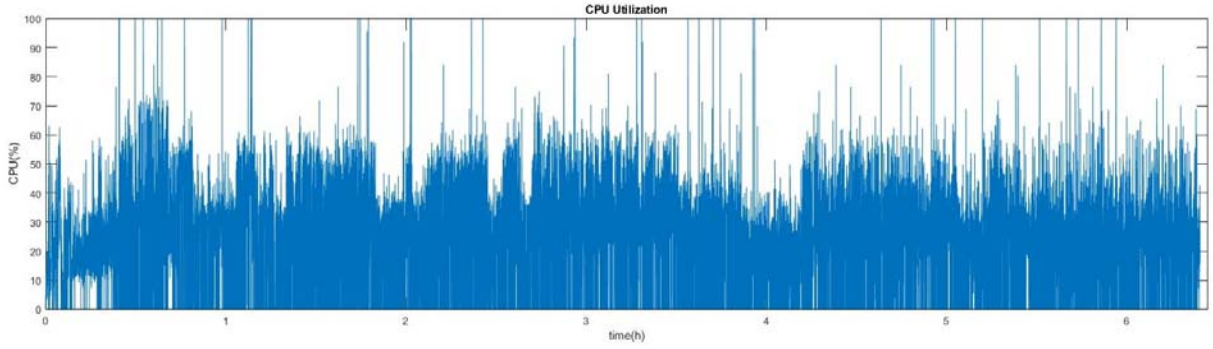

Figure 7: Usage of CPU

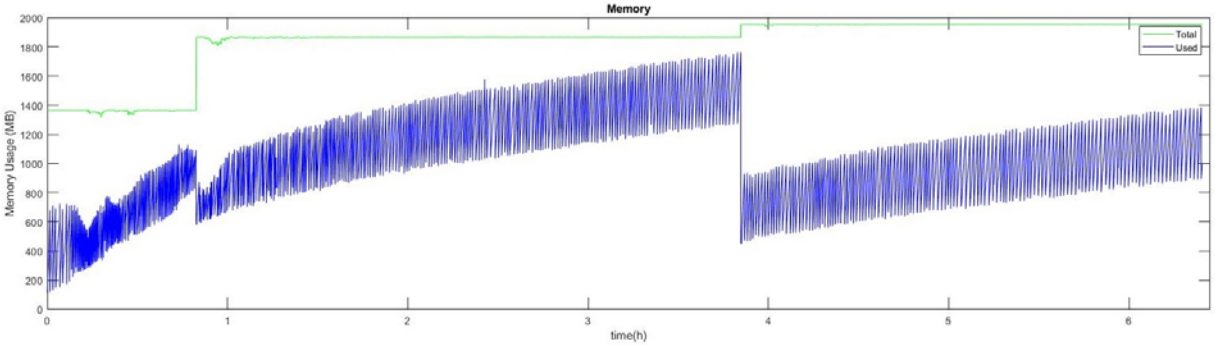

Figure 8: Total memory allocated for TRITON service (green) and memory used by the service (blue)

The average CPU usage was 22.4% and the standard deviation 18% during the whole time of the simulation. The average memory used by the service was 1GB and the maximum value was 1.7GB. When the peak value was reached, the garbage collector was called by the system and the memory usage dropped to 450MB and it continued to rise with the same rate as before, which means that the true need for memory is not so much big. From the figure is seen that if enough memory is pre-allocated at the beginning of the execution, then the rate of memory consumption is 1GB per 3 hours and then it resets when the garbage collector is called automatically.

Finally, in the next figure are displayed the generated events and alarms per minute. The total number of events created, was 1955548 and the total number of alarms 129768. The large number of alarms can be explained, because when configuring the rules, some of them were applied in the whole geographic area where the simulation took place and the snooze time (parameter for limiting alarm rate) was left to zero in order to stretch the limits of the engine.

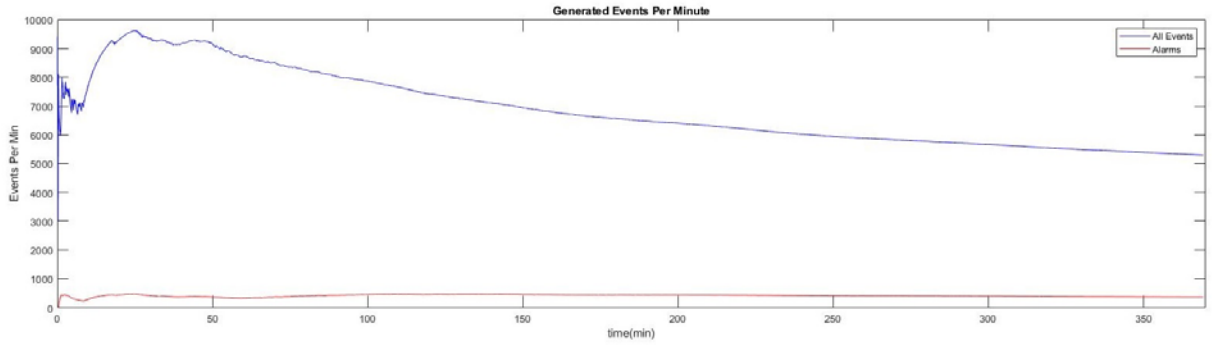

Figure 9: Rate of creation for all events (blue) and alarms (red)

In the above figure the maximum number of events per minute is 9639 and it happens at the beginning of the simulation when all vessels have been created. As time passes the rate is reduced, which is logical since a lot of vessels arrive at their destination and stop to transmit information. Knowing the total number of vessels, there is a 1:4 ratio of existing vessels to generated events.

TRITON service was able to handle the large amount of vessel traffic for the whole duration of the simulation. Even if the traffic (2700 vessels) was a lot higher than the average expected in Greece (~700 vessels), it was able to process events without any delays, keeping the CPU usage at low levels and efficiently handling the used memory.

4. CONCLUSIONS

The present work shows how the geospatial complex event processing paradigm can be used to build a robust, real time, with low latency, memory efficient and scalable service for abnormal vessel behavior detection. The service can recognize many patterns and it can be extended by adding new rules, when new anomalies should be detected. Thus, it is capable for supporting a wide range of maritime applications and to be a primary component of modern VTS platforms. TRITON has been tested with real and simulated data and has shown its efficiency in detecting anomalies. Future work includes the use of artificial intelligence for recognizing specific patterns using reinforcement and unsupervised learning.
The work presented in this paper was partially funded by the ongoing EC H2020 Marisa (GA 740698) project.

REFERENCES

[1] Andersson, M. and Johansson, R., “Multiple sensor fusion for effective abnormal behaviour detection in counter-piracy operations,” in Proceedings of International Waterside Security Conference, Carrara, IT (2010).
[2] Cugola, G., and Margara, A., “Processing flows of information: From data stream to complex event processing,” ACM Computing Surveys 44(3), 15:1–15:62 (2012).
[3] T. Lu, X. Zha, and X. Zhao, “Multi-stage monitoring of abnormal situation based on complex event processing,” Procedia - Procedia Comput. Sci. 96, 1361–1370 (2016).

[4] Tawsif, K., Hossen, J., Raja, J.E., Jesmeen, M. Z. and Arif, E., "A Review on Complex Event Processing Systems for Big Data," 2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP), 1-6 (2018).
[5] Terroso-Saenz, F., Valdes-Vela, M. and Skarmeta-Gomez, A. F., "A complex event processing approach to detect abnormal behaviours in the marine environment," Information Systems Frontiers 18(4), 765-780 (2016).
[6] EsperTech, "Esper Description," EsperTech, 2019, < http://www.espertech.com/esper/> (2019).
[7] Roy, J., Davenport, M., "Categorization of maritime anomalies for notification and alerting purpose," Nato Workshop on Data Fusion and Anomaly Detection for Maritime Situational Awareness, 15-17 (2009)
[8] Biagiotti, L. and Melchiorri, C., Trajectory Planning for Automatic Machines and Robots, Springer-Verlag, Berlin Heidelberg, 166-172 (2008)
[9] Marisa, "Maritime Integated SurveillanceAwareness," Marisa Project, < https://www.marisaproject.eu/>.
[10] Navigation Center, "Ais Class A Ship Static and Voyage Related Data," U.S. Coast Guard, < https://www.navcen.uscg.gov/?pageName=AISMessagesAStatic>.