



# An Event Calculus for Run-Time Reasoning

Periklis Mantenoglou  

National and Kapodistrian University of Athens, Greece  
NCSR “Demokritos”, Athens, Greece

---

## Abstract

In stream reasoning, the task is to derive high level abstractions of large data streams with minimal latency, as required by contemporary applications. This work presents an Event Calculus-based approach to stream reasoning, highlighting its core features and recent extensions.

**2012 ACM Subject Classification** Computing methodologies → Temporal reasoning

**Keywords and phrases** Event Calculus, temporal pattern matching, complex event recognition

**Digital Object Identifier** 10.4230/LIPIcs.TIME.2023.14

**Category** Extended Abstract

**Funding** This work was supported by the ENEXA project (No 101070305), and by the Hellenic Foundation for Research and Innovation (HFRI) under the 3rd Call for HFRI PhD Fellowships (Fellowship Number: 6011).

## Motivation

Modern applications require the processing of large, high-velocity data streams that are being generated continuously. A stream reasoning system derives instances of spatio-temporal pattern satisfaction, based on the actions/events reported in such data streams, with minimal latency. These spatio-temporal patterns may define a set of situations of interest in the target application domain. In maritime situational awareness, e.g., a stream reasoning system can be used to detect vessel activities that may be suspicious, illegal, dangerous or have a negative environmental impact, based the position and velocity signals that are continuously being emitted by sailing vessels [7]. For instance, spatio-temporal patterns may specify an illegal fishing activity in a prohibited area or an unexpected halt in signal transmissions.

There are several requirements for effective stream reasoning. First, a stream reasoning system should be based on a formal pattern specification language, in order to allow the user to express situations of interest without ambiguity. Second, this language has to be expressive enough to support all situations of interest that need to be detected in the target application. Third, the system should be equipped with highly-efficient algorithms for detecting these patterns, taking into consideration that input actions/events cannot be stored in memory en masse when operating in a streaming setting.

## The Event Calculus for Run-Time Reasoning

Towards addressing the requirements of stream reasoning, we proposed “The Event Calculus for Run-Time Reasoning” (RTEC), a logic-based, formal computational framework that is optimised for stream reasoning [2, 6, 5]. RTEC is based on a logic programming implementation of the Event Calculus [3], a temporal formalism for representing and reasoning about events and their effects. The Event Calculus dialect used by RTEC is many-sorted and includes events, fluents, i.e., properties that may have different values at different points in time, and a linear time model with integer time-points. The built-in Event Calculus predicates of RTEC are used to express event occurrences and changes in the values of fluents, and specify the time periods during which fluent-value pairs (FVPs) hold continuously.



© Periklis Mantenoglou;

licensed under Creative Commons License CC-BY 4.0

30th International Symposium on Temporal Representation and Reasoning (TIME 2023).

Editors: Alexander Artikis, Florian Bruse, and Luke Hunsberger; Article No. 14; pp. 14:1–14:3

Leibniz International Proceedings in Informatics



LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

$\text{happensAt}(E, T)$  denotes that event  $E$  takes place at time-point  $T$ , while  $\text{initiatedAt}(F = V, T)$  (resp.  $\text{terminatedAt}(F = V, T)$ ) expresses that a time period during which fluent  $F$  has the value  $V$  is initiated (terminated) at time-point  $T$ .  $\text{holdsFor}(F = V, I)$  states that fluent  $F$  has the value  $V$  continuously in the maximal intervals included in list  $I$ . Finally,  $\text{holdsAt}(F = V, T)$  states that fluent  $F$  has the value  $V$  at time-point  $T$ . Moreover, RTEC adopts the specification of the common-sense law of inertia used in the Event Calculus, expressing that FVPs persist through time, unless an event that affects the value of the fluent takes place.

RTEC features two types of fluents, called “simple” and “statically determined”. The conditions under which event occurrences may affect the values of simple fluents are expressed through domain-specific  $\text{initiatedAt}$  and  $\text{terminatedAt}$  rules. Given the “initiation points” and the “termination points” of some simple fluent  $F$  with value  $V$ , RTEC computes  $\text{holdsFor}(F = V, I)$ , i.e., the maximal intervals  $I$  in which  $F = V$  holds continuously. In the case of a statically determined fluent  $F$ , RTEC the maximal intervals of  $F = V$  directly, i.e., without computing the initiation and termination points of  $F = V$ , using an domain-specific  $\text{holdsFor}(F = V, I)$  rule, defining the maximal intervals  $I$  of  $F = V$  in terms of the maximal intervals of other FVPs via interval operations, such as union, intersection and relative complement.

RTEC supports stream reasoning applications by integrating the aforementioned representation and reasoning formalism with caching, indexing, windowing and a “forget” mechanism that removes redundant events and FVP intervals from its knowledge base. Moreover, RTEC is restricted to hierarchical knowledge bases that allow bottom-up processing, thus avoiding re-computations. The complexity analysis of RTEC is available in [2].

## Recent Extensions

The specifications of modern applications may include cyclic dependencies. In maritime situational awareness, e.g., a fishing trip consists of several stages, such as approaching a fishing area, fishing and returning to a port. These stages form a cycle, as each stage depends on the previous one. Moreover, situations of interest are often defined as temporal combinations of other situations, which are typically durative and take place within temporal intervals. The corresponding patterns can be expressed using Allen’s interval relations [1]. For instance, we may detect the suspicious situation where a vessel stops signal transmissions while being close to another vessel using the “during” Allen relation.

We extended RTEC for expressing patterns featuring cyclic dependencies and Allen relations [6, 5]. Our theoretical analysis of the resulting framework highlighted its semantics, correctness and complexity, while our empirical evaluation demonstrated its effectiveness when reasoning over large streams of benchmark and real data from modern applications. This extended version of RTEC is publicly available<sup>1</sup>.

## Further Work

Uncertainty is inherent in modern applications. In maritime situational awareness, e.g., malfunctions in signal transmitters may lead to data streams that include empty fields or erroneous field values. A recent work tackles uncertainty by associating input stream items with probability values, serving as a confidence estimate, and then employing probabilistic reasoning to derive the probabilities of pattern satisfaction instances based on such an input [4]. In the future, we would like to extend RTEC with probabilistic reasoning techniques.

---

<sup>1</sup> <https://github.com/aartikis/RTEC>

---

**References**

---

- 1 J. Allen. Maintaining knowledge about temporal intervals. *Comm. of the ACM*, 26(11):832–843, 1983.
- 2 Alexander Artikis, Marek Sergot, and Georgios Paliouras. An event calculus for event recognition. *IEEE Transactions on Knowledge and Data Engineering*, 27(4):895–908, 2015.
- 3 Robert Kowalski and Marek Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–96, 1986.
- 4 Periklis Mantenoglou, Alexander Artikis, and Georgios Paliouras. Online probabilistic interval-based event calculus. In *ECAI*, volume 325, pages 2624–2631, 2020.
- 5 Periklis Mantenoglou, Dimitrios Kelesis, and Alexander Artikis. Complex event recognition with Allen relations. In *KR*, 2023.
- 6 Periklis Mantenoglou, Manolis Pitsikalis, and Alexander Artikis. Stream reasoning with cycles. In *KR*, pages 544–553, 2022.
- 7 Manolis Pitsikalis, Alexander Artikis, Richard Dreo, Cyril Ray, Elena Camossi, and Anne-Laure Joussemme. Composite event recognition for maritime monitoring. In *DEBS*, pages 163–174. ACM, 2019.