	SocEDA	
	Cloud based platform for large scale social aware EDA	
	ANR-10-SEGI-013	

Document name:	Identification of requirements to support distributed CEP
Document version:	1.0
Task code:	
Deliverable code:	2.2.3
WP Leader (organisation):	ADAM
Deliverable Leader (organisation):	ADAM
Authors (organisations):	Nabil DJRALLAH, Lionel SEINTURIER, Fawaz PARAISO
Date of first version:	02/05/2012

Change control

Changes	Author / Entity	Code of version
Creation of the document	Nabil DJARALLAH	

Exclusive Summary

Throughout the last years, distributed event-based (or message-based) systems are subject to disseminate a huge volume of heterogeneous events. In such systems, *Complex Event Processing* (CEP) has emerged as a solution to detect, handle, and react (in real-time) to complex events, which are correlations of more primitive events. Moreover, CEPs have gained considerable importance as a means to extract information from those distributed event-based systems.

Although various CEP engines implement the support for dealing with the business heterogeneity of events, the technological integration of these events remains uncovered. Therefore, this deliverable aims at providing a holistic set of general and technical requirements to provide a framework platform capable to integrate and federate a distributed system of CEP engines, as targeted by the SocEDA project. We called this framework platform, *Distributed Complex Event Processing Engine* (DiCEPE), a platform which focuses on the integration of CEP engines in distributed systems.

Moreover, we present relevant challenges we are facing to provide this distributed framework platform. The identification of challenges would be of great help to understand the federation of heterogeneous CEP engines problem, in order to design a viable and a coherent solution that takes these challenges into account.

Table of Contents

1. Introduction

Nowadays, there is a huge amount of information sources everywhere around us, and heterogeneity is the rule among them. These events are used by distributed applications such as Air-Traffic Control Systems, Automated Banking Systems, etc., that require them to be filtered and correlated for complex pattern detection and transformed to new events that reach a semantic level appropriate for higher-level applications. The need for real-time processing of information is relevant for such systems. However, one major problem in such complex systems is the overwhelming amount of data that has to be considered at the same time, which may lead to information overload. So, the way data is handled is very important.

One key solution is to introduce the new paradigm for real-time event-based applications and systems [3]; Complex Event Processing (CEP). CEPs are considered as useful and important technology to realize an Event-Driven Architecture [7]. CEP is designed to match event correlations, called event patterns, from a constant flow of events, called an event stream. One key aspect of CEP is the ability to define reactive rules corresponding to event pattern definition and triggering a process in real-time when an event pattern is detected. CEP is used in a wide diversity of applications, which have to deal with voluminous streams of incoming data, the complexity of processing and time response [4], [5], and [6]. There are many domains in which CEP is actually used, like network management, traffic monitoring, or fraud detection, and it has drawn the attention of many research projects such as [1] and [2], and commercial products such as [4] and [5].

Unfortunately, processing events using only one CEP engine may be a dangerous practice [8], as it may miss some important information due to being overwhelmed by all the events. For these reasons, within the SocEDA project, we believe that an interconnected system of CEPs can overcome these issues. The CEPs that compose our interconnected system may of different vendors (heterogeneous CEPs) and support different interfaces (heterogeneous communication protocols).

In this deliverable we provide a holistic set of requirements to provide a federated framework platform capable to integrate and federate a distributed system of CEP engines, as targeted by the SocEDA project. We called this framework platform, *Distributed Complex Event Processing Engine* (DiCEPE). The major objective of the DiCEPE platform is to handle various communication protocols in order to federate CEP engines and ease the deployment of complex systems-of-systems.

The deliverable first presents challenges to manage interconnected CEP systems. Then we discuss the requirements, focusing on the inherent issues of CEPs' heterogeneity and communications interfaces diversity.

2. Challenges to Federate Complex Event Processing Systems

Several CEP engines (e.g. [9], [10], [11], etc.) have emerged from the growth of event processing in general. All those engines have a common goal of enabling event-driven processes and applications that react to single and to a composition of events, occurring at different times and within different contexts. However, the number of events can rise quickly depending to processes, applications, and their scenarios.

An example of such applications could be the nuclear management scenario (see D5.3.1 [12]), where the volume of heterogeneous information could grow rapidly and require urgent processing, taking into account the critical dependencies between actions. One key solution is to enable the cooperation of different CEP engines in order to share the management of the event-oriented system. These CEP engines could be vendor-specific with heterogynous interfaces. Therefore, the federation of these distributed CEP engines to enable cooperation and interoperability is essential and imposes specific challenges like the ones listed below:

- **Communication heterogeneity**: The communication through the federated CEP engines is essential and may need different types of communication protocols. For instance, when interacting in a private network, an asynchronous protocol, such as JMS, may be much faster and as reliable as a synchronous one. However, when leaving the private network, it may be blocked due to firewall restrictions, and its reliability may suffer as well. In that case, a synchronous approach like REST could become more suitable and useful.
- **Heterogeneous CEP**: The actors have different tasks which may generate multiple complex events. These events can then be processed by other CEP engines that may belong to the same or to different domains. Each group of actors can have different kinds of complex event processing engines. In this point, the challenge consists in supporting the interaction among heterogeneous complex event processing engines.
-
- **Scalability**: This distributed system could process a huge volume of events and should be able to handle increasing loads in real-time while maintaining other performance criteria such as latency, coherence between actions, etc.

- **Adaptability**: In an evolutionary system, constraints, contexts and actions may change over time. Therefore, the possibility to deploy new rules at run-time is essential to deal with the new situations.

All these issues are clearly within the scope of the SocEDA project, which addresses them in a holistic manner and encompasses technical aspects amongst others. This also motivates the definition of requirements, as well as the solution (called DiCEPE: Distributed Complex Event Processing Engine) that overcomes previous challenges. With the DiCEPE, we will allow the federation of distributed CEP engines that can deal with the heterogeneity of the event's sources and communication protocols, address scalability, provide the means to hierarchically control, and adapt the rules of the different CEP engines in the distributed environment. This approach will allow a domain-specific engine to be managed by the experts of the domain, while providing at the same time a way to create collaborations among different domains.

3. General and Technical Requirements

Events from various sources such as federated SOAs (Service Oriented Architectures), and other can become very complex and difficult to process with centralized CEP architecture, which requires greater bandwidth and computational capability and lacks of robustness and scalability because of single point failure or network break. For those reasons we thought of a solution more distributed where we spread centralized CEP tasks load over multiple communicating stations by a space-based communication paradigm over multiple CEP

In this section, a series of requirements are discussed resulting from objectives and challenges dimensions. Those requirements address the aspects that are relevant for assuring the processing of primitive and complex events within a distributed CEP-based system. The primary focus here is to enhance the cooperation between heterogeneous CEPs within a given distributed system, in order to efficiently manage the processing of the generated events inside the SocEDA platform.

Functional Requirements	
FR001	DiCEPE platform MUST take primitive event(s) and/or complex event(s), and rule(s) (described in Event Processing Language and called also “statements”) as inputs
FR002	DiCEPE platform MUST return complex event(s) as output(s)
FR003	DiCEPE platform MUST synchronize events that arrive from different sources in different time due to network delays
FR004	DiCEPE platform MUST support different kind of events
FR005	DiCEPE platform SHOULD know the address to which it will send (publish) the processing results (complex events)
FR006	Inter-DiCEPE interaction SHOULD be done over a subscription/publication process to exchange events and rules
FR007	DiCEPE platform MUST support the add of event processing rules without stopping the system
FR008	DiCEPE platform MUST support the removal of event processing rules without stopping the system
FR009	DiCEPE platform MUST support the update of event processing rules without stopping the system
FR010	DiCEPE platform MUST support a listing feature of all event processing rules loaded in the DiCEPE without stopping the system
FR011	DiCEPE platform MUST be able to return an event processing rules according to its identifier without stopping the system
FR012	DiCEPE platform SHOULD receive topic(s) to which it must subscribe AND must receive events as notifications when they occur

FR013	DiCEPE platform SHOULD be managed trough the CEPEditor AND/OR the SeaCloud which interact with exposed service interfaces as web services
Interoperability Requirements	
IR001	DiCEPE platform SHOULD be portable and interoperable
IR002	Within the DiCEPE platform, CEPs MAY be heterogeneous (vendor-specific)
IR003	DiCEPE platform must be an open solution that will allow the different CEPs to select their communication protocol, i.e. DiCEPE must support synchronous AND asynchronous communication protocols (e.g. REST, JMS, WS, etc.)
IR004	DiCEPE platform MUST handle an interoperable and common format for exchanging events
Performance Requirements	
PR001	DiCEPE platform MAY be designed as SCA components with FraSCAti
PR002	DiCEPE platform SHOULD be capable to fragment statement(s) between involved remote DiCEPEs
Scalability Requirements	
ScR01	DiCEPE platform MUST support a huge volume of events and rules in an optimal way
ScR02	DiCEPE platform SHOULD be able to distribute event-processing task over other DiCEPEs if it does not have all the capabilities to preform it OR to do a load balancing

Table 1. Requirements for the DiCEPE Framework Platform

4. Conclusion

In this deliverable challenges and requirements about the management of heterogeneous, interconnected, and distributed CEPs are discussed and have led us to consider a novel solution called DiCEPE. The DiCEPE platform will offer interoperability for distributed complex event processing engines, via federation.

This platform will focus on providing a very flexible component architecture, which supports the interaction of different complex event processing engines simultaneously, while enabling communication (synchronous and asynchronous communication protocols) among them with a distributed system and deployment. Thanks to its distributed nature, DiCEPE also will offer real scalability.

5. Bibliography

[1] Chakravarthy, S., and Jiang, Q., "Stream data processing: a quality of service perspective : modeling, scheduling, load shedding, and complex event processing". Advances in Database Systems, Vol.36, Springer, 2009.

[2] Kasi, M. K., and Hinze, A. M., "Cost analysis for complex in-network event processing in heterogeneous wireless sensor networks". *Proceedings of the 5th ACM international conference on Distributed event-based system, DEBS'11*, pp 385-386, New York, NY, USA, 2011.

[3] Luckham, D. C., "The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems". Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001.

[4] RTView-platform, "SL RTView platform". Retrieved from http://www.sl.com/solutions/cep.shtml

[5] SpatialRules. "SpatialRules". Retrieved from http://www.objectfx.com/geospatial-solutions-products/spatialrules

[6] WSO2, "WSO2 Complex Event Processing Server". Retrieved from http://wso2.com/products/complex-event-processing-server

[7] Hohpe, G., "Programming without a call stack – Event-driven Architecture". Retrieved from http://www.eaipatterns.com/docs/EDA.pdf, 2006.

[8] Luckham, D., "The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems". Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2008.

[9] StreamBase, "StreamBase CEP". Retrieved from http://www.streambase.com/products/streambasecep/#axzz1uSV6321A

[10] EsperTech, "Esper CEP". Retrived from http://esper.codehaus.org/

[11] Etalis, "Etalis CEP". Retrived from http://code.google.com/p/etalis

[12] SocEDA Project, "Deliverable D5.3.1: Use Case Description". Retrived from: www.soceda.org., February 2011.