ProaSense: The Proactive Sensing Enterprise

Dominik Riemer, Ljiljana Stojanovic and Benedikt Kämpgen

FZI Research Center for Information Technology, Karlsruhe, Germany
firstname.lastname@fzi.de

Abstract. The ProaSense FP7 project deals with proactive detection of situations of interest in digital enterprises. We briefly present the project's objectives and outline our approach on using semantics in order to dynamically create processing pipelines for high-frequency stream processing applications.

1 Introduction

ProaSense addresses the "Digital Enterprise", a new form of enterprises with ad-hoc extensive connectivity of digital assets. ProaSense enhances business processes through integration of sensing capabilities. The goal of the project is to design and develop methodologies and tools that support proactivity in digital enterprises, making them able to anticipate problems and opportunities and support their realization and resolution. In order to achieve this vision, ProaSense implements a distributed, scalable real-time architecture that follows the Observe, Orient, Decide and Act (OODA) cycle. ProaSense showcases its results in two use cases, one of them coming from the automotive manufacturing domain, the other one from the oil drilling industry.

1.1 ProaSense Architecture

We defined an architecture (illustrated in figure 1) that consists of two layers, *design stage* and *real-time stage*. The real-time stage is implemented as an Apache Storm[1] topology facilitating distributed high-frequency data processing, where each component is represented as a bolt in the topology. The design stage interacts with the real-time layer by the means of *control spouts* which dynamically change the behaviour of the underlying topology at runtime. For instance, the CEP editor serves as a user interface to the definition of processing pipelines as detailed in the following section.

2 Semantics-Based Processing Pipelines

A major contribution of the project is StreamPipes (as part of the CEP component in the architecture), a methodology enabling users to create *processing*

[1] storm.apache.org

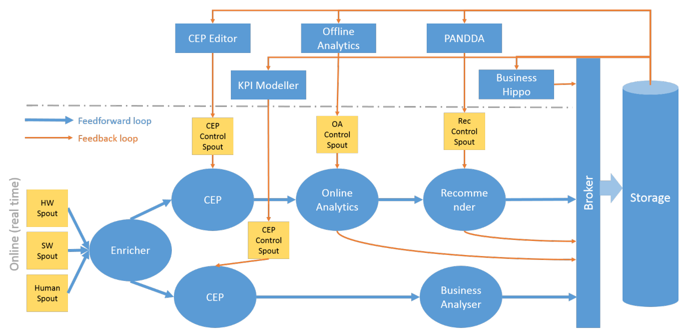

Fig. 1. ProaSense Architecture

pipelines as compositions of multiple, heterogeneous (i.e., language and vendor-independent) processing agents [1]. Pipelines consist of one or more event sources producing an event stream, an arbitrary number of processing elements transforming an incoming stream to an output stream and an event consumer that implements some after-processing logic (e.g., dashboarding or decision making algorithms). In order to support this integration, we defined an ontology to describe capabilities of event sources, streams, processing agents and event consumers (defined as *nodes*). This ontology is used by a management and execution framework to compute possible matchings between nodes. Afterwards, it delegates the execution of each node to a runtime implementation (e.g., a Storm topology or a CEP engine). Furthermore, StreamPipes consists of a web-based authoring tool to simplify the generation of processing pipelines for non-technical users.

3 Demo Setup and Relevance to the Track

We will demonstrate StreamPipes based on one ProaSense use case, proactive manufacturing. This includes a demonstration of how streams and processing agents can be properly described and deployed according to our semantics-based data model as well as a prototype of our pipeline authoring tool.
We believe our attendance in the networking session can be benefitial from two points of view: First, our approach can be used in many different application areas that depend on integration and transformation of real-time data, e.g., IoT applications. Second, exchange between the semantic web community and ProaSense's focus on real-time systems can clearly facilitate discussions on our semantics-based description and matchmaking framework.

References

1. Riemer, D., Stojanovic, L., Stojanovic, N.: Sepp: Semantics-based management of fast data streams. In: 7th IEEE International Conference onService-Oriented Computing and Applications (SOCA). pp. 113–118. IEEE (2014)