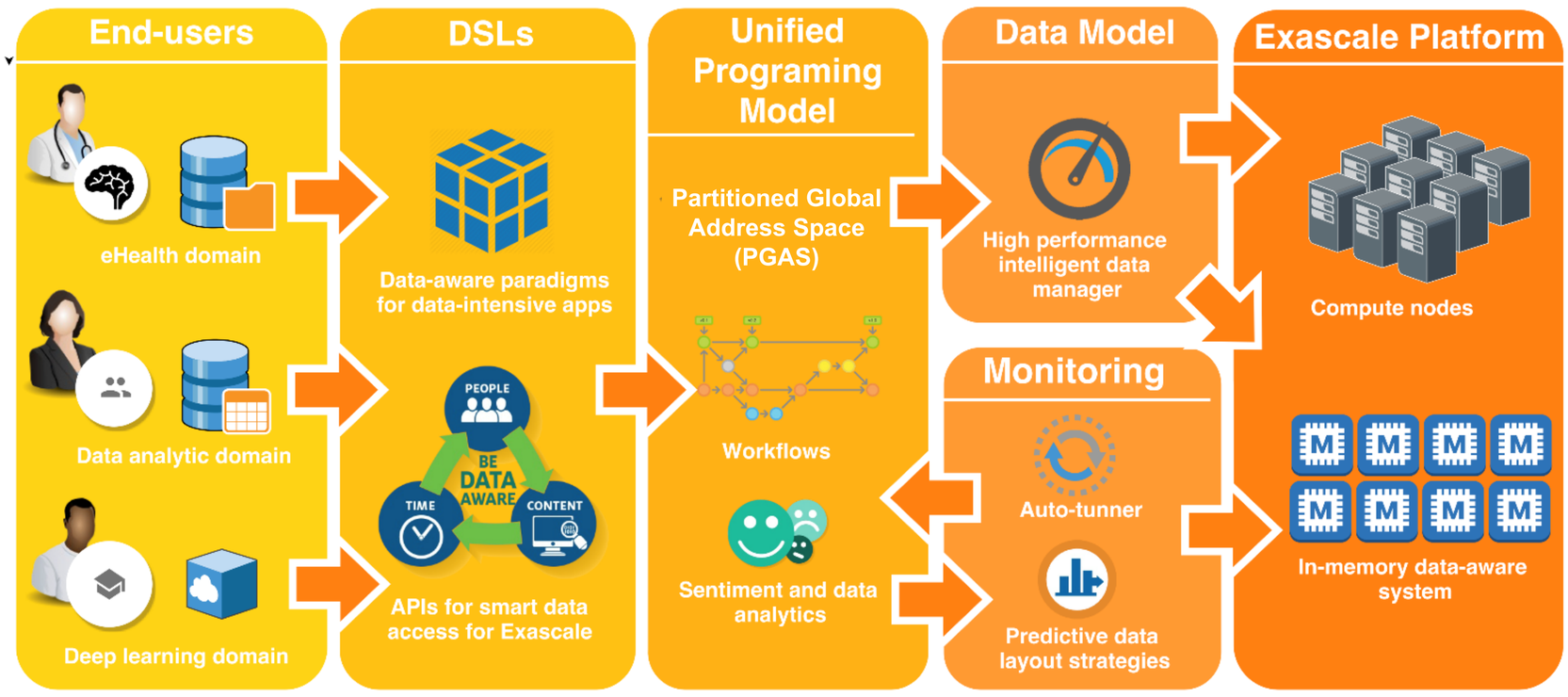
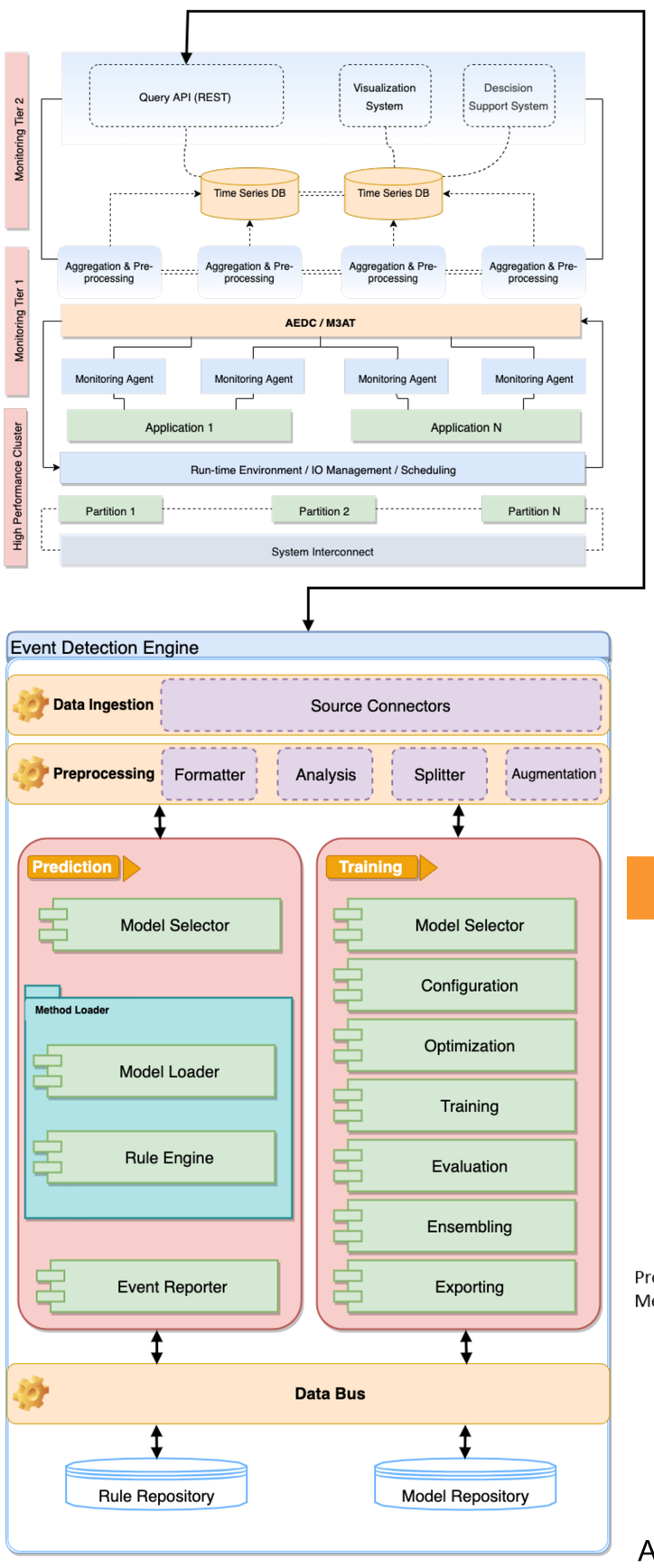
EXASCALE PROGRAMING MODELS FOR EXTREME DATA PROCESSING

https://www.aspide-project.eu/

The **ASPIDE** project will contribute with the definition of a new programming paradigms, APIs, runtime tools and methodologies for expressing data-intensive tasks on Exascale systems, which can pave the way for the exploitation of massive parallelism over a simplified model of the system architecture, promoting high performance and efficiency, and offering powerful operations and mechanisms for processing extreme data sources at high speed and/or real-time.



Monitoring + Event Detection Engine



The **ASPIDE** monitoring architecute exposes and associates a vast set of collected metrics with a main goal to expose potential application bottlenecks, which execute in Exascale systems. The key insight behind such an approach is that the source of a bottleneck in a data-intensive applications is often not place where it is detected (i.e. where the data is processed with a high communication or thrashing overhead), but where it is allocated.

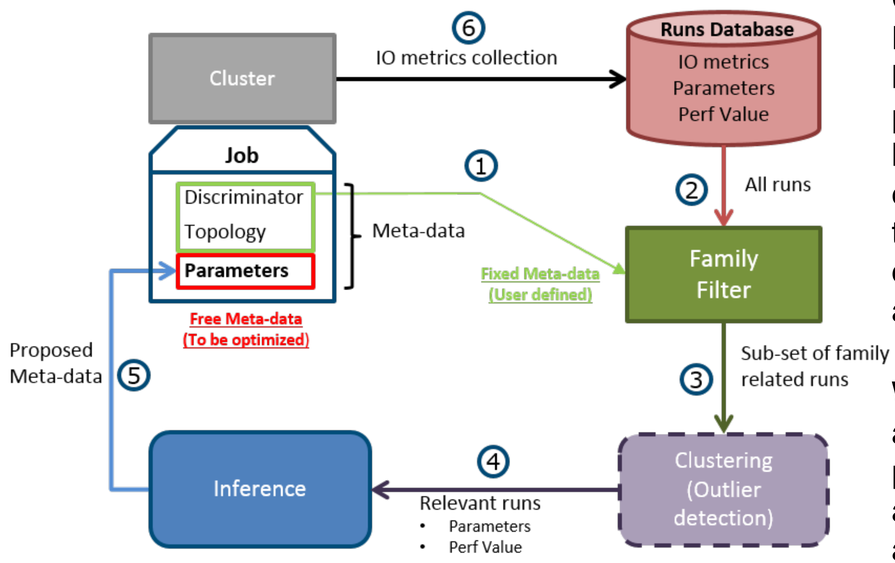
It is not enough to create a system where the monitoring data can be reliably stored and queried. In order for the monitoring data to have real value we have to analyse it and extract additional data and insights. The event detection sub system will handle the processing of raw monitoring data in order to identify events and anomalies during application execution.

Unified Programing Model

We have implemented a protopype versión (***DCEX***) that is able to generate the task corresponding to a pattern composition for distributed-memory platform (*tb*). Afterwards, tasks are processed by a set of worker entities using a FIFO scheduler. Currently, we are working on the new communication channels using ZeroMQ for communicating tasks in a distributed environments.

```
Carea.loadConf(fileConf); //Scheduling, monitorig setup
Carea ca(5000);  // 5000 compute nodes
dcex::Container_text in("in/", ca);
dcex::Container_text out("out/", ca);
dcex::pipeline(dcex::tb{ca}, // Defines a workflow
    in,
    farm([](auto x)->json {
       return json::parse(x);
    }),
    farm([](json x)->pair<Cell,unordered_map< string,int>>{
      static const int sc = 200;
      return make_pair(getCell(x,sc), getTagFreq(x));
    }),
    out
);
```

Data model for profiling and analyzing data-intensive applications



One of the existing challenges in the design of High Performance Computing (HPC) infrastructures is to efficiently balance the computational and storage I/O resources of the platform. This balance depends on the interaction between the hardware resources, the system software stack, and the executing applications. In general, this interaction is difficult to tackle because of the components complexity and the dynamic nature of a HPC platform that executes different applications with distinct characteristics.

We propose a methodology that aims to tackle the aforementioned problems. The proposed methodology provides configurations strategies of the I/O system that will allow to enhance the whole data life-cycle. Several approaches can be envisioned to tackle the problematic of configurations:

An *offline static predictive configuration*. The runtime instrumentation data is used offline to identify I/O acceleration opportunities and then trigger the configuration of I/O accelerators.

An *online dynamic configuration* approach will monitor and adapt online the configuration to provide an optimal performance at any moment of the execution of the jobs.

Call: H2020-FETHPC-2016-2017
Type of Action: Research and Innovation Action (RIA)
Acronym: ASPIDE
Duration: 36 months - **Start Date**: 2018-06-15
Project Costs: € 2,464,093.75
EU Contribution: € 2,464,093.75
PI: Dr. Javier Garcia Blas (fjblas@inf.uc3m.es)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 801091.