Publishable Summary

ASAP Vision and Objectives

The field of data analytics includes techniques, algorithms and tools used to inspect collections of data to extract patterns, generalizations and other useful information. Big data analytics is very important in risk assessment, pharmaceutics, fraud detection, epidemiology, business process effectiveness, market analysis, anti-terrorism, etc. More importantly, large-scale analytical data processing has become a necessity in the majority of industries. Enabling engineers, analytics experts and scientists alike to tap the potential of vast amounts of business-critical data has grown increasingly important. Such data analysis demands a high degree of parallelism, in both storage and computation. Business datacenters host vast amounts of data, stored over large numbers of nodes with multiple storage devices, and process them using thousands or millions of cores.

To be useful and effective, analytics applications must produce near-real-time (or interactive) response times to the engineers' queries, a task that directly conflicts with the size of the data, their diversity of structure and representation, their location, and the ad-hoc nature of analytical queries. These issues have given rise to many diverse programming models, execution engines and data stores to assist with large-scale data management. Such systems target specific kinds of data or computations, where they greatly outperform general-purpose solutions like traditional relational databases. For instance, key-value stores drop the relational model in favor of much greater scalability and parallelism, reaching high IOPS (Input/Output Operations per Second) performance at a fraction of the cost of a relational system. For instance, the most widespread programming model for scaling applications to big data is Map-Reduce, which is very effective with computations expressed as data-parallel "mapper" tasks and "reducer" tasks that merge the results. Many operations, however, act on irregular data that may be heterogeneous, structured or unstructured, streaming or stored in various formats. Complex analytics applications may also perform computations with dynamic dependencies. For example, a graph computation may be data dependent, not knowing the next data to be processed before visiting the previous node. Similarly, online games or analytics of social graphs perform graph traversals or fixpoint computations that are difficult and inefficient in Map-Reduce, as each phase must store the intermediate graph state and redistribute it across machines before the next phase starts.

To alleviate these difficulties, specialized models and systems have been proposed, such as Pregel, Hama, Dremel, Powerdrill, implementing different computing models. While all these systems have had great success, they still showcase their advantages on a limited subset of applications and types of data: For instance, graph-processing engines limit the amount of freedom in the computation at each node (or part of a graph) and fail to fully exploit possible parallelism.

Many modern datacenter applications are more complex than this.

- First, dynamic data processing is often very heterogeneous in terms of complexity and time consumption. In graph-structured data, for instance, the amount of stored data and required computation may differ from node to node. Thus, computing graph-queries in lock-step (i.e., by repeating whole-graph processing steps as in Pregel) loses parallelism, as all computations of a phase depend on the computations of the previous phase. Moreover, this requires the storing of all intermediate results which may not all be necessary for the next phase. For example, a graph query may need to repeat reductions over the whole graph until the longest path or cycle converges.

- Second, modern datacenters often include many heterogeneous storage formats, each used in multiple contexts and by different applications. For instance, a datacenter running multiple applications may include data stores ranging from traditional row-stores and modern column-stores, unstructured data in raw files and semi-structured data in XML, RDF or similar formats stored either in files, adapted relational stores or specialized stores (e.g., RDF stores). Depending on the data format, the computation of a query may differ in complexity and performance. As it is not always optimal to convert data formats and storage, for legacy (existing working applications) or performance optimization (data format optimized for application queries) reasons, data analytics framework that scales the datacenter needs to support and adapt to multiple data storage formats.

- Lastly, datacenters often host multiple applications. For example, an application may involve the processing of a data stream, by fast querying and updating data in one or more formats and datastores, while other applications perform long-running queries with multiple phases over the same data. Taking this discussion one step further, it is evident that the ad-hoc manner of data analytics, together with the sheer size and complexity of data, call for increased human participation: Scientists and engineers often "experiment" by posing long-running queries on huge datasets, trying to identify trends and fuse data into new "signals". As most of these operations are particularly I/O- and time-consuming, an early evaluation and re-calibration of the submission parameters would greatly assist the process.

This project aims to fill this gap, delivering a *fully automated and highly customizable system* for the easy development and execution of arbitrary data analytics queries on large heterogeneous data stores. The *vision* of the **ASAP** (**A**daptive, highly **S**calable **A**nalytics **P**latform) project is to provide a complete software stack that efficiently computes complex analytics queries over large, heterogeneous, irregular or unstructured data. To achieve that, the consortium develops a programming model for writing analytics queries at a high level of abstraction and a set of execution engines that schedule queries over large data sets that may span various sources and stores, and have irregular dependencies. The outcome of the project is a *modular, completely open-source system* that offers a unified way for the rapid development and execution of analytics queries with arbitrary dependencies over heterogeneous, irregular or unstructured data.

This project aims to build a unified, open-source execution framework for scalable data analytics. The main idea behind ASAP is that (i) no single execution model is suitable for all types of tasks; (ii) no single indexing and data-store is suitable for all types of data; and (iii) an adaptive system that has correctly modeled analytics tasks, costs and is able to monitor its behavior during tasks is a more general, efficient way of tackling this problem.

The ASAP system aims to develop the technology to facilitate the development and execution of general-purpose analytics queries over irregular data. To achieve this goal, the project focuses on the following *scientific and technological objectives*:

- Develop a general-purpose task-parallel programming model, implemented by a task-parallel execution engine, making the development of complex, irregular datacenter queries and applications as easy as writing regular Map-Reduce computations. The task-parallel runtime will incorporate all the benefits of Map-Reduce systems and state-of-the-art task-parallel programming models, namely: (i) express irregular general-purpose computations, (ii) take advantage of resource elasticity to use resources only when required by the application, (iii) hide synchronization, data-transfer, locality and scheduling issues from the programmer, (iv) be able to handle large sets of irregular distributed data, and (v) be tolerant to node, system, or disk faults.

- An intelligent management platform that models and manages multiple execution and storage engines to the submitted jobs. This modeling framework will take into consideration the type, location and size of data, the type of computation and available resources in order to decide on the most advantageous store, indexing and execution pattern available. To that direction, our system will complement our execution model with existing open-source solutions (Map-Reduce) as well as with state-of-the-art distributed storage engines (NoSQL, column-stores, distributed file-systems, etc.) in order to have a broad applicability and increased performance gains.

- A unique adaptation methodology that will enable the analytics expert to amend the task she has submitted at an initial or later stage. This is a process often required for analytics tasks that fail to capture the users' intention due to erroneous parameter or dataset choices. Our system will be able to adapt the execution strategy according to the already created results and the changed parameters.

- A monitoring methodology that will enable the analytics expert to obtain accurate, intuitive and timely results of the analytics tasks she has initiated. Through a visualization engine, initial and intermediate results and meta-analytics will be shown in real-time, enabling the scientist to assess the usefulness of the method.

In addition to the development of the methods mentioned above, the ASAP consortium builds two applications for validating and showcasing the technology: one in the area of business analytics on telecommunication data, and one in the area of web analytics.

Year 2 Achievements

In the second year of the project, the consortium coordinated its efforts into implementing and refining the design of a functional ASAP system. The project focused on implementing and integrating the modules designed during the first year, while refining the requirements and use cases of both the overall ASAP system and each individual component separately. Towards that direction, in the first six months considerable effort was put into development of specific use-cases from the two User Partners (IMR and WIND), and building an early version of the ASAP platform. Using this outcome, in the second six months of the project effort was focused in integrating the individual components of the ASAP system, generating test cases from the application use cases. Overall, this effort has led to an integrated prototype of the ASAP System Architecture that includes all modules, even if they are at an early stage of development, and is able to execute application workflows. The outcome of this effort is thoroughly described in deliverables D1.3 (M24) and D7.2 (M24).

Alongside this effort, technical Working Groups initiated design and development activities in their respective Work Packages (WP). Specifically, WP2 implemented a parallel language for defining analytics operators and a compiler into a high-performance low level language, as described in deliverable D2.2 (M24). In WP3, the first version of the Intelligent Resource Scheduler (IReS) platform implemented, as described in D3.2 (M18). In WP4, the early prototype was extended with support for nested data decomposition and a first version of a distributed scheduler was developed, as described in deliverable D4.2 (M24). In WP5, a workflow management tool is designed and implemented as presented in deliverable D5.2 (M18). In WP6, the first version of InfoViz services was developed as described in deliverable D6.2 (M18). In WP7, integration of all existing modules was implemented, with automated testing for each module and for the integration of each pair of interacting modules, as described in deliverable D7.2 (M24). In WP8, the first prototype of continuous query workflow was implemented, presented in deliverable D8.3 (M24). Similarly, in WP9, the application specification was refined and finalized, and an early prototype was developed, as presented in deliverable D9.3 (M24) In WP10, the results of achieved dissemination for the second year of the project (including publications, workshop organization, media presence, etc) are reported in deliverable D10.5 (M24).

To summarize, Y2 main results achieved are the following:

- ASAP Use-cases, System Architecture and Workflows refinement and extension of design, as well as additional use cases and tests.
- Operator language implementation, high-level to low-level language compiler.
- Prototype of IReS platform, operator cost modeling.
- Prototype of irregular query execution engine with distributed scheduler.
- Prototype of workflow language and modeling tool.
- Visualization components for structured, clustered data.
- Integration framework prototype, deployment on two clusters, development of integration tests.

- ASAP Integrated System posted as open source, technical and user documentation, installation scripts.

- Dissemination and exploitation in workshop organization, publications, media presence, social media.

- Web analytics application use cases, workflows, scenarios partly implemented and functioning.

- Telecommunication analytics use cases, workflows, scenarios partly implemented and functioning.

ASAP Expected Outcome and Impact

The main expected outcome of the project is a complete set of methods materializing into open-source tools that will allow intelligent, efficient and real-time and user-customized execution and management of analytics tasks. Specifically: (i) a new analytics programming model that will incorporate a user's cost and performance requirements; (ii) an intelligent management platform that models and manages multiple execution and storage engines to the submitted jobs; (iii) an analytics execution engine that enables the user to amend queries at a later stage; (iv) a unique runtime monitoring methodology for retrieving the progress of analytics jobs in real time. Using state of the art visualization tools and UIs, ASAP will enable its users, both end-users and analytics engineers, with intuitive, real time access to the services it offers. Our modules will be both generic and open-source, in order to allow for maximum utilization and ease of adaptation with existing commercial, academic and community systems.

Figure **??** shows the main outcome of the project. Users of the system are now able to express complex analytics queries via the Query Description Tool over multiple and heterogeneous data. The UI also enables live monitoring of the query progress (Visualization Cockpit) and intermediate results (Query Cockpit) using intuitive visualizations, integrated with monitoring capabilities of any existing analytics engines used. Analytics jobs inserted through the UI are adaptively scheduled to the most beneficial runtime and datastore available by the ASAP Scheduler. The Decision Making Module takes the decision of which part(s) of the job are executed over which technology using analyses and comparing against already stored models (Modeling/Learning Engine). Running jobs are monitored in real time by the Compute and Storage Monitor modules, sending intermediate and final job results to the UI. The Runtime Deployment Inspector manages dynamic resources among the query execution engines and data stores depending on the executed job.

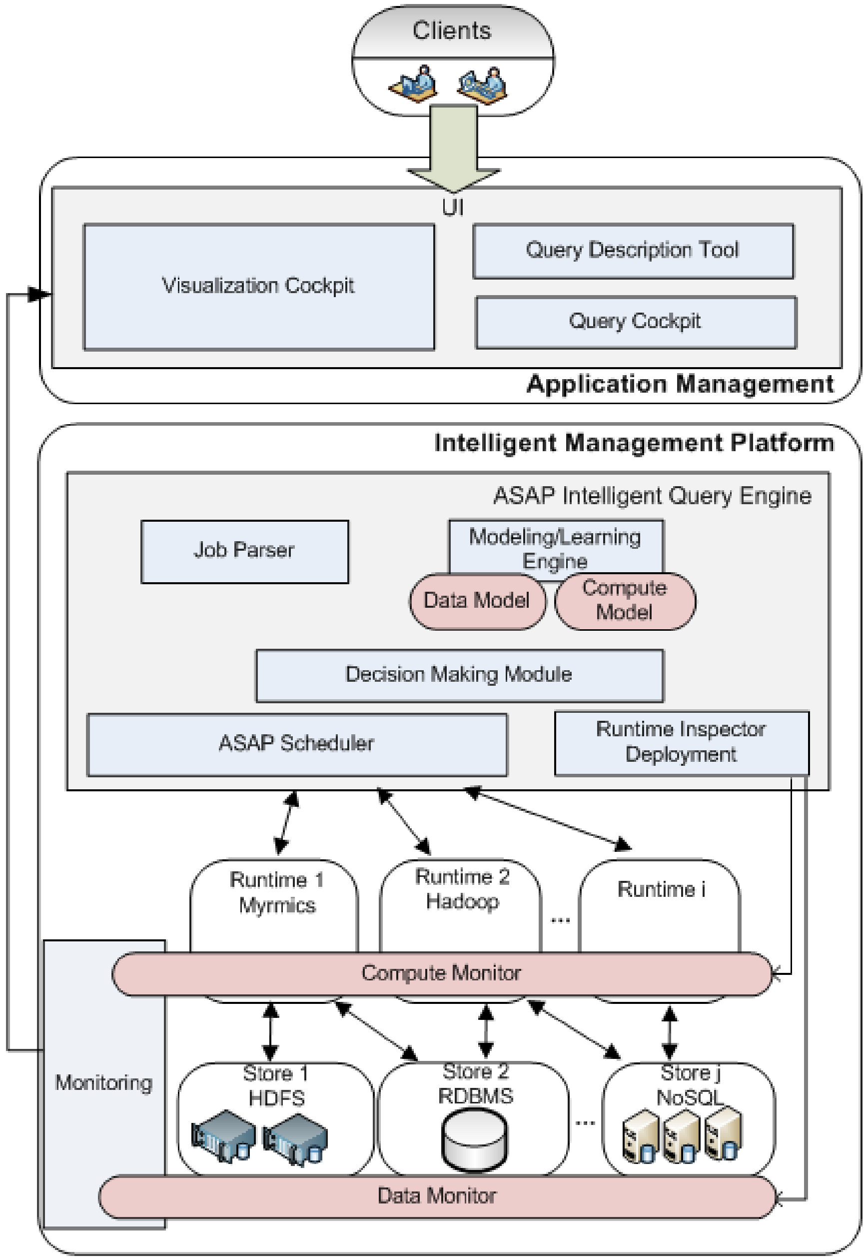

Figure 1: ASAP Technology Overview

ASAP Website

For further information and for keeping up-to-date with Project ASAP and its results, please visit our website at `http://www.asap-fp7.eu/` and follow us on Twitter `@ASAP_EU`.

ASAP Consortium and Contacts

Foundation for Research and Technology – Hellas
Coordinator
Contact person: Dr. Polyvios Pratikakis
E-mail: polyvios@ics.forth.gr

Institute of Communication and Computer Systems
Contact persons: Prof. Dimitrios Tsoumakos, Dr. Katerina Doka
E-mail: dtsouma,doka@cslab.ece.ntua.gr

Université de Genève
Contact person: Dr. Verena Kantere
E-mail: Verena.Kantere@unige.ch

Queen's University Belfast
Contact persons: Prof. Hans Vandierendonck, Prof. Dimitrios Nikolopoulos
E-mail: h.vandierendonck,d.nikolopoulos@qub.ac.uk

Internet Memory Research
Contact person: Prof. Philippe Rigaux
E-mail: philippe.rigaux@internetmemory.net

Wind Telecommunicazioni
Contact persons: Roberto Bertoldi, Maria Rita Spada
E-mail: roberto.bertoldi,MariaRita.Spada@wind.it

webLyzard technology
Contact persons: Arno Scharl
E-mail: scharl@weblyzard.com