FP7 Project ASAP
Adaptable Scalable Analytics Platform

D1.1 Early User Requirements

WP 1 – Architecture and Requirements Analysis

Nature: Report

Dissemination: Public

Version History

Version	Date	Author	Comments
0.1	19 Aug 2014	P. Pratikakis	Initial Version
0.2	28 Aug 2014	P. Pratikakis	First Revision
1.0	31 Aug 2014	P. Pratikakis	Final Version

Acknowledgement This project has received funding from the European Union's 7th Framework Programme for research, technological development and demonstration under grant agreement number 619706.

Executive Summary

This document outlines the current results of the ongoing requirements analysis for the ASAP project. It describes issues relating to the ASAP system design and required functionality for the analytics applications used by IMR and WIND.

The process for analyzing user requirements was based on interviews with engineers and experts from WIND and IMR, as well as face-to-face all-consortium discussions. The results were analyzed and discussed in two working groups, arriving at a set of the specification of five use cases that overlap all areas of the proposed research and also a set of functional and non-functional requirements for ASAP.

Contents

Chapter 1

Introduction

One of the objectives of project ASAP is to perform research and development towards solving existing problems in data analytics. A deep understanding of the users' real requirements is a crucial factor in the development of any high-quality software product. Similarly to software engineering, research objectives and results have greater impact when they are applicable to — or even driven by— real problems in IT. To that end, the project includes two industrial partners (WIND, IMR) that are facing such problems and limitations in their real analytics applications, and would stand to gain by the proposed research results.

As in most software development (or research) projects, user requirements may not be completely clear at first. This could be because

- Developers (researchers) do not have a clear idea of the user domain and specific needs of the application, even if they understand the problem at an abstract level.
- Users may not have a clear view of what they may gain by using new technology, how it is applicable to their problem, or even what is and is not possible to achieve compared to the existing solutions.

For these reasons, it is important to follow a process that elicits the requirements and specifications of the ASAP analytics applications in such a way that they form *well-defined* and *achievable* research objectives. However, as research projects usually include a higher level of uncertainty than traditional software development, it is not possible to follow a standard waterfall model for user requirements analysis and specification. Instead, we develop an initial set of user requirements and specifications, whose analysis, re-evaluation and re-definition is an ongoing process that interacts with the remaining work packages, adapting to new constraints as they are discovered.

1.1 Purpose of this Document

This document presents preliminary results of *Work Package 1: Architecture and Requirements Analysis* (WP1) regarding the user specifications and use cases developed within *Task 1.1: User*

Requirements of the work package. A goal of project ASAP is to design and develop an analytics platform for the two analytics applications of consortium partners WIND and IMR, regarding telecommunication analytics and web analytics, respectively. As such, this work package aims to elicit the specifications, requirements and constraints of these analytics applications; and define open problems to drive the research and development performed in the project.

This report summarizes the activities carried out between March 2014 and August 2014 towards the development of user requirements definitions and specifications for ASAP and its two driving analytics applications, in WP1. We describe the methodology, initial findings, and use cases developed during this phase of requirements analysis. The results of this analysis will influence work in other work packages and subsequent research.

1.2 General Description of ASAP

The ASAP project aims to develop a unified framework for data analytics. This section summarizes a high-level view of the project for convenience. We refer the reader of this document to the project Description of Work for a detailed discussion and analysis of the objectives and their motivation.

In short, ASAP aims to build a unified, open-source execution framework for scalable data analytics. It is based on the idea 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.

ASAP 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 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.

- Develop an intelligent management platform that models and manages multiple execution and storage engines to the submitted jobs. The modeling framework must 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 they have 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. ASAP 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 they have 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.

1.3 Referenced Platforms

This section presents background information on multiple existing software components used in the use case specifications.

1.3.1 Hadoop

Hadoop is a distributed execution engine for Map-Reduce computations [14]. Currently, both IMR and WIND use Hadoop to manage distributed analytics computations. Query systems built on top of Hadoop include Pig [11], Hive [13], Sawzall [12], Jaql [2] and Tenzing [4].

1.3.2 HDFS

The Hadoop File System [3] is a distributed filesystem, initially developed as part of the Hadoop execution engine for Map-Reduce computations. It is currently deployed in data centers of both IMR and WIND and used to store and distribute input data, data produced and consumed during the execution of a computation, and in some cases the results of analytics computations.

1.3.3 Spark

Spark is an analytics execution engine optimized for iterative Map-Reduce computations [15]. Spark uses an in-memory data representation to avoid unnecessary storage of intermediate data and can express complex workflows with multiple Map-Reduce steps. Moreover, Spark includes a large library of machine-learning tools and support for SQL-like queries.

1.3.4 GATE

GATE is a collection of open-source tools for Natural Language Processing (NLP) computations [5], used in the Web Analytics application of ASAP.

1.3.5 Lucene

Lucene [9] is an open-source text indexing and search engine library, used in the web-analytics use cases.

1.3.6 Elasticsearch

Elasticsearch is a distributed indexing and searching server [8]. Elasticsearch builds on top of Lucene, using multiple Lucene nodes to implement a distributed search system. It is also used in the web analytics use cases.

1.3.7 PostgreSQL

PostgreSQL is an open-source RDBMS used to store and query the named entity data used in the NLP computations for the web analytics use cases.

1.3.8 DBpedia Spotlight

DBpedia Spotlight is a tool for annotating DBpedia resource in text [10].

1.3.9 AIDA

AIDA is a tool for disambiguation of named entities in text and tables [7].

Chapter 2

Methodology

In traditional software engineering, the requirements analysis process includes surveys, interviews, and work-group meetings, in which domain experts and engineers co-author a set of use cases that best describe the desired behavior of the system. The system specifications can then be extracted from the use cases in the next step of the process, by software architects [1].

ASAP requirements analysis applies this process to derive a set of use cases from the two analytics applications, using interviews, face-to-face meetings and working group discussions with expert engineers from WIND and IMR. The goal of the requirements analysis is to develop a set of use cases that must satisfy both constraints, that is (i) have interesting properties that overlap the proposed research areas in Work Packages 2 through 6, and (ii) be realistic scenarios of the two analytics applications, so that the technology developments will immediately benefit these applications.

Of course, the waterfall model is rarely directly applicable in software development, and never in research projects, where solutions, designs, and constraints may not be obvious at first. Thus, we perform an initial requirements analysis to elicit use cases that can be used to drive the research and development process, and benchmark results, while at the same time allowing the specification of requirements to evolve for the first part of the project, in what would correspond to a more agile software engineering methodology [6]. This document presents the early results of the requirements analysis, as a set of use cases that together can be used to specify the desired behavior of the ASAP platform that would benefit the analytics applications of the ASAP industrial partners.

2.1 Teleconferences

During the first six months of the project, we conducted several teleconferences involving representatives from all partners. Among other agenda topics, the teleconferences involved brainstorming discussion on user requirements. Based on the material presented in kick-off meetings and posted on the project Wiki, these discussions resulted in a more detailed understanding of existing practices of the industrial partners on behalf of the research partners. Moreover, they resulted in a better understanding of the goals of the new technology proposed on behalf of the industrial partners.

2.2 One-on-one interviews

In addition to teleconferences involving the whole consortium, we conducted one-to-one teleconferences with representatives from WIND. We focused this technique on the WIND Telecommunication Analytics application, as it involves objectives and ideas that are currently not implemented. The objective for the Telecommunication Analytics application is to develop entirely new functionality using ASAP technology. In contrast, the IMR Web Analytics application is currently implemented to a large extent using an array of existing analytics technologies. It still stands to gain in performance, flexibility, and also novel functionality by using technology proposed in ASAP, although that refers to new components and functionality added to an existing application.

The objective of the one-to-one teleconferences was to arrive at representative scenaria for the Telecommunication application that clarify:

- The objective of the application.
- How the application creates value and why its development is desired by WIND.
- The nature, size, complexity, and availability of data involved.
- The kinds of computations required to implement parts of the application.
- The actors involved in operating the application.

Resulting from these interviews, we were able to arrive at two indicative application scenaria. These were then discussed in a focus group among the whole consortium.

2.3 Focus group

During the first face-to-face meeting of the ASAP project, we performed summary presentations of the two applications, involving scenarios and any specifications elicited during the teleconferences. Also, research partners presented specifications of the research work packages. The objective of the presentations was to communicate the constraints and problems attacked by the planned research to the industrial partners, to help define the properties of desirable use cases that, if found in an application, would fit well with the planned research.

2.4 Use cases

Finally, during the face-to-face meeting, we used focus groups to elicit a set of use cases that best fit the properties and objectives desirable to the research partners, while still being part of the analytics applications of the industrial partners.

Chapter 3

Web Analytics

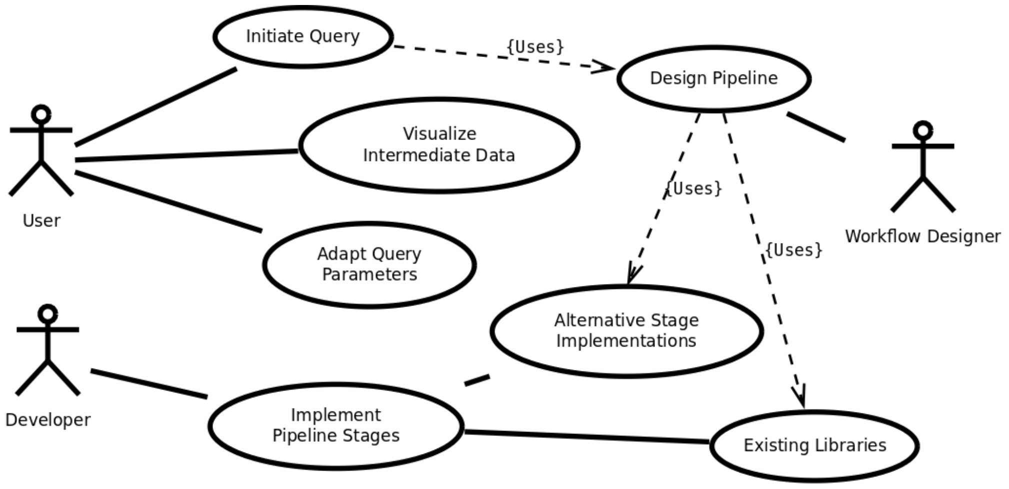

Figure 3.1: A summary of the web analytics use case.

The Web Analytics application includes several data flow processes and query workflows. Figure 3.1 presents a summary of the application actions. In short, developers implement stages of computation pipelines, which are then synthesized by workflow designers, and executed to answer user queries. This chapter presents three indicative use cases selected for relevance to the ASAP research Work Packages. The use cases are based both on (i) existing implementations redesigned to take advantage of the optimizations proposed in ASAP, and on (ii) development of new computations that were not previously possible with existing tools. Initially, the use cases are described and specified to target a small data set of almost 18k documents, averaging 200kb per document; in total 139GB of data. This is so that use cases are easy to deploy outside of the actual IMR data center, to facilitate research and development by the ASAP research partners. We plan to eventually deploy and test the ASAP platform on the larger, actual data set in the IMR data center.

3.1 Use case 1: NLP-classification

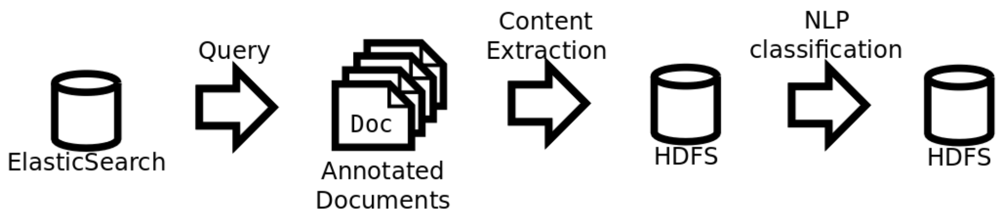

Figure 3.2: The data flow of the first use case: storage and computation phases.

Figure 3.2 presents an overview of the data flow in this use case. The first query in the pipeline uses Elasticsearch to select documents from the document store. The results are processed to annotate each document with its metadata, in JSON format.

The second stage extracts the textual content from the documents using one of 3 alternative algorithms, two of which are open-source and one is proprietary, developed by IMR, then appends the extracted plain texts to the annotated documents as additional metadata. The stage outputs the annotated documents with the extracted plain texts in JSON format, and sends them directly to the next stage.

In the final stage of this use case performs NLP classification on this data, using one of two alternative implementations. The stage also appends classification results to the annotated documents as additional metadata. The three stages are pipelined and deployed on multiple worker machines in a cluster. Intermediated data of each stage are directory propagated to the next stage deployed on the same machine. Finally, the resulting data of the final stage are stored in HDFS.

This use case captures a typical form of the "current" IMR Web analytics pipeline such that (i) a pipeline is a sequence of operators applied per document, and (ii) each operator is implemented as an exchangeable component which allows us to freely choose and connect to define a pipeline specialised for customer's requirements.

The use case fits well with several of the desired properties for the research work packages in ASAP:

- It contains several stages that compute intermediate data (WP3, WP5, WP6).
- It is an on-line computation, as the initial query is customized by the user (WP5).
- It contains more than one alternative implementations per state of computation (WP3).

Moreover, as an existing part of the Web Analytics application it clearly benefits from the results of that research.

3.1.1 Actors

User The actor that submits the initial query and starts the computation. Also reads the final computed data after all stages have finished.

Developer The developer implements logic of each stage on the top of an abstracted framework which hides details of distributed execution engines.

Workflow designer The workflow designer designs a concrete pipeline by choosing actual implementations of each stage and wiring them to define data exchange between stages.

3.1.2 Requirements

UR-1 Given an Elasticsearch query, the cost of the pipeline can be estimated for all possible alternative implementations of each stage.

UR-2 The user can visualize or query intermediate data between stages while the next stage is being computed.

UR-3 The user can choose to adapt a parameter of the query that is used in a later stage, after the initial query starts its computation, but before that stage has finished its computation.

UR-4 The developer can implement logic of each stage as an exchangeable component with enough metadata including schema of input data and output data.

UR-5 The workflow designer can define a pipeline by using a well-abstracted language without coding.

3.2 Use case 2: K-Means

In contrast to the first use case, the second use case is not currently implemented in the Web Analytics application and will be implemented in ASAP. Figure 3.3 shows the data flow of the second use case. As described above, the user submits a query to Elasticsearch, selecting a set of annotated documents. The second phase similarly extracts the relevant content from the annotated documents. In addition to these computations the second use cases includes a last stage of computation that reads the annotated document content as produced by the content extraction phase, the results (feature vectors, classification output, etc) of the NLP classification computation, and performs clustering using the k-means algorithm. Alternative implementations of k-means clustering are included in systems such as Mahout, WEKA, Spark MLlib.

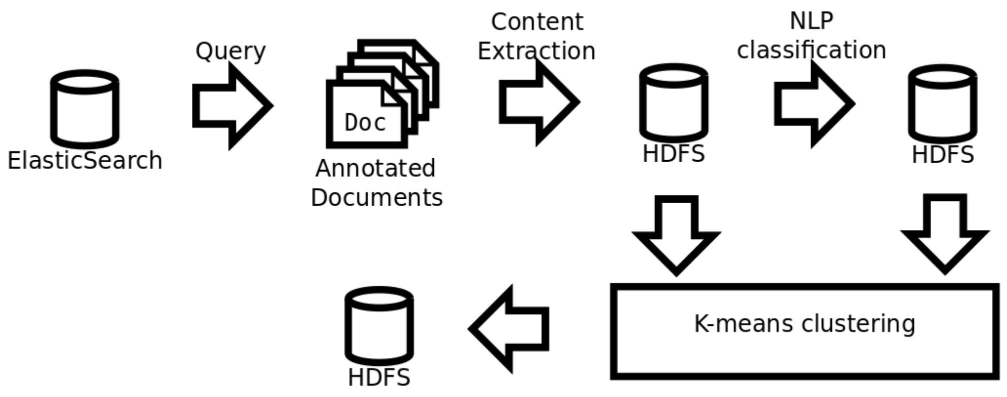

Figure 3.3: The data flow of the second use case: storage and computation phases.

This use case is an example of applications which IMR currently does not have but intends to implement in the near future for more advanced analysis. The framework which IMR is currently using does not have operators which abstract iterations in a pipeline, and decision making engines to determine the best engine and strategy for executing a pipeline.

Since it is very similar, this use case is also very relevant to many of the research objectives of ASAP. In addition, the k-means computation is an iterative algorithm, which performs sub-optimally when expressed in Map-Reduce. A better iterative computation engine (such as Spark) already outperforms Mahout because of this phenomenon. We expect that for the same reason, the k-means iterative computation is an interesting motivating example for ASAP research:

- The algorithm is iterative and hierarchical, includes data decomposition and redistribution in every step and so, would benefit from a programming model with such abstractions (WP2).
- It fits the characteristics of algorithms that will benefit from a dynamic dependence analysis in the presence of imbalance (WP4).
- The results of k-means clustering and NLP classification are visualizable in an intuitive way (WP6).

3.2.1 Actors

User The actor that submits the initial query and starts the computation. Also reads the final computed data after all stages have finished.

Developer The developer implements logic of each stage on the top of an abstracted framework which hides details of distributed execution engines.

Workflow designer The workflow designer designs a concrete pipeline by choosing actual implementations of each stage and wiring them to define data exchange between stages.

3.2.2 Requirements

UR-6 The k-means clustering algorithm should have multiple implementations.

UR-7 Some of the clustering alternative implementations should allow the user to adjust the parameters of computation.

UR-8 The k-means algorithm must be expressable in the ASAP programming model.

UR-9 The developer can implement (or can reuse an existing implementation of) k-means logic using iterations as an exchangeable component.

UR-10 The workflow designer can define a pipeline without coding.

3.3 Use case 3: Named Entity Disambiguation

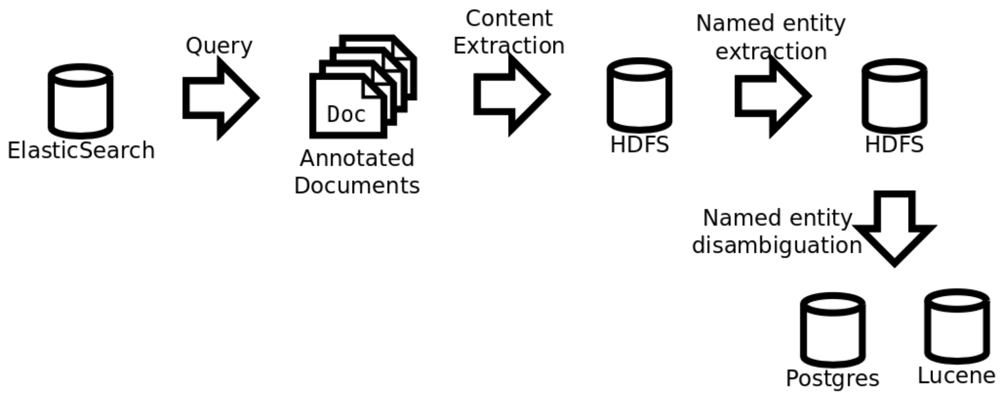

Figure 3.4: The data flow of the third use case: storage and computation phases.

Figure 3.4 presents the data flow among all phases of the third use case. As above, document selection and preprocessing remains the same. The post processing of selected documents has two phases. The first phase is named entity recognition (NER) which seeks occurrence of named entities in plain texts. It is performed by using one of existing libraries including GETA, Stanford NER, etc. The last phase is named-entity-disambiguation (NED). It determines identities of named entities in texts and link them to a knowledge base like DBpedia. There are several open source implementations of NED including DBpedia spotlight and AIDA. It is long-running computation using large (50-100GB) indices of target knowledge bases, installed in different software (PostgreSQL, Lucene, etc), sometimes by using specialised hardware (SSD, large main memory, etc.). The result of NED is stored in HDFS.

This use case is one of applications that IMR is experimentally doing, but has difficulty to put into production because it requires manual intervention in scheduling and deployment of a pipeline. We expect this use case to demonstrate improvement using ASAP research, by distributing the last phase of the computation as well as overcoming resource-bound performance constraints. This use case is different to the previous two in terms of interaction, as it is not triggered by user queries and is a long-running, off-line, computation.

3.3.1 Actors

User The actor that submits the initial query and starts the computation. Also reads the final computed data after all stages have finished.

Developer The developer implements logic of each stage on the top of an abstracted framework which hides details of distributed execution engines.

Workflow designer The workflow designer designs a concrete pipeline by choosing actual implementations of each stage and wiring them to define data exchange between stages.

3.3.2 Requirements

UR-11 The computation of NED should be distributed to multiple machines in a cluster by taking into account machine resources.

Chapter 4

Telecommunication Analytics

The Telecommunication Analytics application includes many possible query workflows, on-line queries, off-line long-running computations and ad-hoc analytics. This chapter presents two indicative use cases selected for relevance to the ASAP research Work Packages. As with the use cases presented in the previous chapter, the use cases are selected to allow for the optimizations and novel features that are being researched in ASAP. Initially, the use cases are described and specified to target a small data set of Call Detail Record (CDR) data from five regions of Italy for a duration of a few days, describing many millions of calls and averaging about 1GB per region. All user data are anonymized in this "development" small data set. This is so that use cases are easy to deploy outside of the WIND data centers, to facilitate research and development by the ASAP research partners. ASAP plans to perform experiment and test the developed technology on actual data and WIND infrastructure.

4.1 Use case 4: Peak detection

Figure 4.1 displays the actions involved in this use case. We assume an anonymization process occurs at regular times, e.g., weekly or monthly, refreshing the data available in the analytics database from real user data. The second phase involves the processing of the anonymized CDR data for clustering along time and space. The objective of this processing is to detect peaks in load, according to a set of criteria. Criteria may include the minimum size of a cluster, the cut-off distance, or other parameters of the clustering algorithm. These parameters should be adjustable by the analytics engineer, marketing expert, etc., who uses the peak analysis results. The results of this phase are added to a database (relational or graph DBMS) that contains peaks detected in previous data. The database of peaks can then be queried by a user, using queries such as:

- Discover clusters of calls that occur with regularity e.g., every week.
- Discover clusters of calls that occur without any regularity.

or similar ad-hoc queries based on the pre-computed peak data.

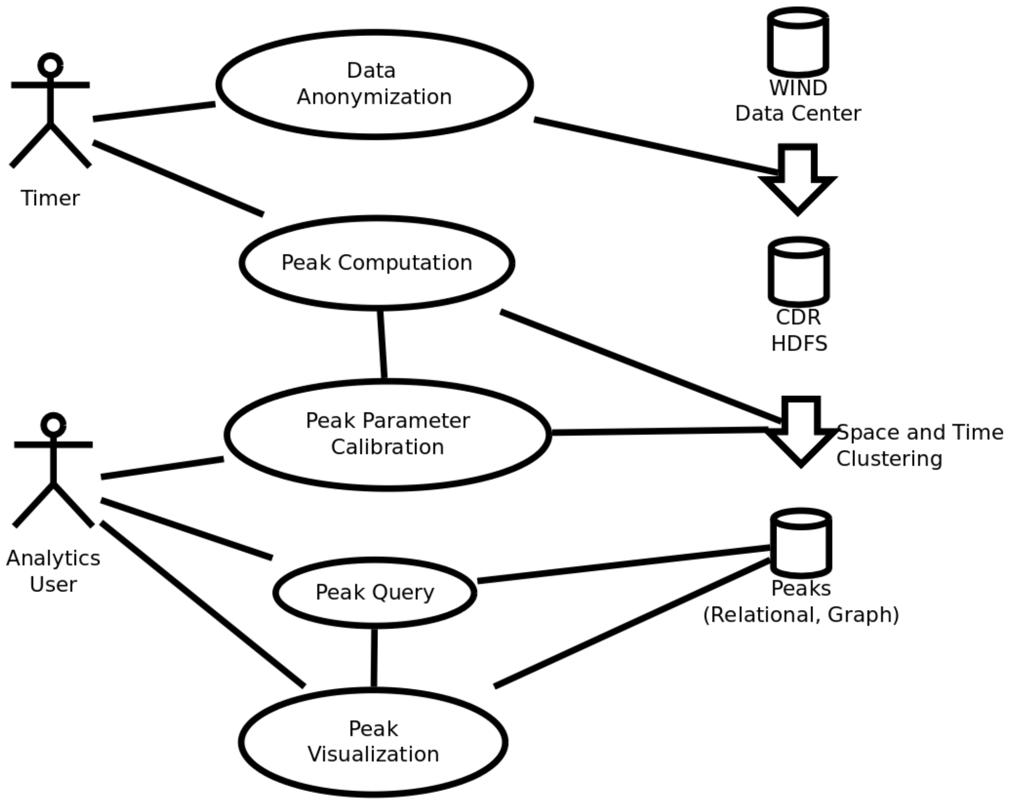

Figure 4.1: Use case diagram of the peak detection scenario.

4.1.1 Actors

Timer Some of the computations in this use case are expected to be triggered automatically at a regular basis.

Analytics User An actor that can query or visualize the results of the peak detection analysis. Based on this feedback, the Analytics User is able to readjust the parameters for peak detection.

4.1.2 Requirements

UR-9 The data coming from the data center must be anonymized.

UR-10 Every anonymization maintains user identity accross different calls by the same user but removes all other private user information.

UR-11 Clustering of calls into peaks should be highly parallel and distributable, using either existing parallel implementations in existing execution engines (e.g., Hadoop) or written in the ASAP programming model and runtime.

UR-12 The user should have a high-level way to visualize and query the database containing the detected peaks.

4.2 Use case 5: Graph join

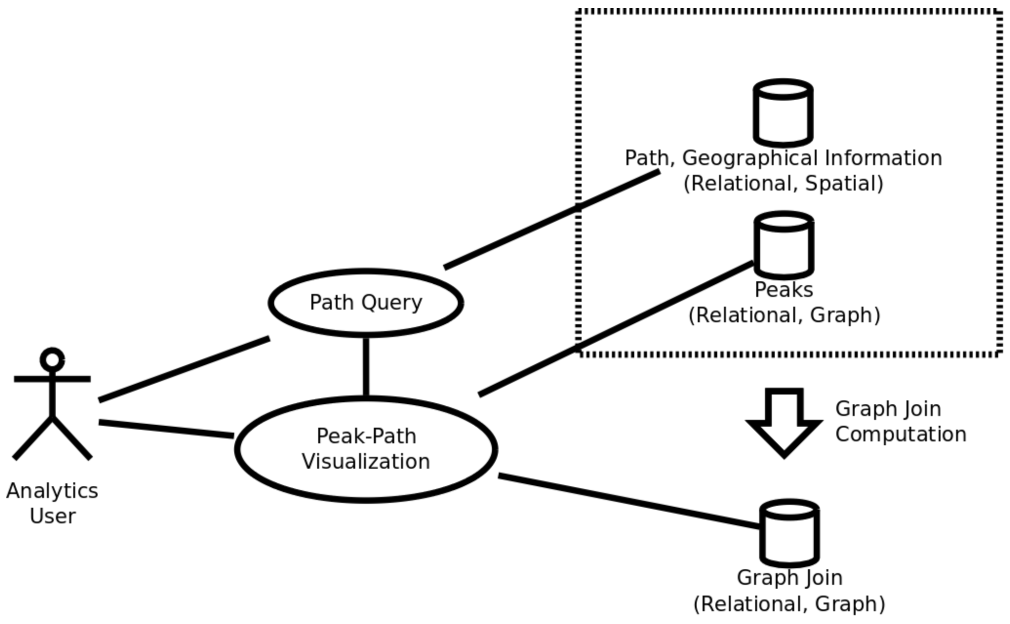

Figure 4.2: Diagram of the graph join scenario.

This use case describes a complex iterative computation with irregular structure (WP2, WP4). Such a computation can be used in the Telecommunication Analytics application to correlate graph-structured data. The scenario of such a correlation query to the Telecommunications Analytics application is displayed in Figure 4.2. This use case answers queries of the form:

- Discover peaks in calls that form a specific pattern, i.e., form a geographical path from point A to point B.

We expect this query to be useful in visualizing data correlated with geographical locations, paths, etc.

4.2.1 Actors

Analytics User The actor that queries the graph database of peaks described in the previous use case, correlates the outcome with geographical information and visualizes the results.

4.2.2 Requirements

UR-13 The implementation of the join operation between the peak information and geographical data should be implemented in the ASAP programming model.

UR-14 The join operation should compute all peaks that occur on a specific geographical path.

4.3 Use Cases and ASAP Objectives

Based on the overall objectives of the project, the objective of WP1 is to perform user requirements analysis and arrive at specifications and use cases that (i) are part of the ASAP analytics applications and (ii) overlap all research areas in ASAP and provide interesting benchmarks for the technology being developed. The above use cases indeed overlap with one or more components of new technology in the platform:

- a new analytics programming model that will incorporate a user's cost and performance requirements;
- an intelligent management platform that models and manages multiple execution and storage engines to the submitted jobs;
- an analytics execution engine that enables the user to amend queries at a later stage;
- a unique runtime monitoring methodology for retrieving the progress of analytics jobs in real time; and
- state of the art visualization tools and UIs that enable intuitive, real time access to the processed data and computations.

4.3.1 Irregular Parallelism and Graph Computations

Research proposed in *Work Package 2: A unified analytics programming model* (WP2) aims to design and develop a programming model in which it will be possible to express irregular parallel computations. Such computations cannot usually be expressed in Map-Reduce, or can be expressed only as expensive iterative Map-Reduce computations, because they include data-dependent flow of control or compute properties that are not data-parallel or results of simple reductions. Usual examples of irregular queries are graph computations. Some graph computations are easy to express as Bulk-Synchronous Parallel (BSP) computations, which require execution engines such as Giraph or Pregel. Other graph computations are not easy to express in BSP, either because they use multiple kinds of data stores, multiple kinds of graphs, etc.

ASAP aims to develop a programming model that is able to express arbitrary graph, tree, or hierarchical computations. To drive the development of this idea, we investigated whether the two analytics applications could take advantage of such functionality. The above use cases, especially use cases 3, 4 and 5, can take advantage of such technology.

In addition, *Work Package 4: A dependency-aware query execution engine* (WP4) proposes to design and implement a query execution system that will be able to efficiently execute such computations, without reducing them into iterative super-steps, as these executions often suffer from load imbalance. The execution of such queries will include detection of dependencies between units of computation, so that they are synchronized and scheduled for maximum efficiency (on the same node, if possible), without delaying the rest of the system. Graph computations with multiple steps would immediately benefit from such technology. Therefore, the above use cases in the applications including such irregular and data-dependent graph computations, can be used to motivate and drive the research in WP2 and WP4, as well as evaluate their results.

4.3.2 Operands with Alternative Implementations

Work Package 3: Intelligent, Multi-engine Resource Scheduling Platform (WP3) aims to develop a modeling, cost-estimation and high-level scheduling platform for optimizing queries and workflows with many alternative implementations of different cost and performance. The elicited use cases indeed showcase these characteristics: workflows of queries, where one or more components have multiple alternative implementations. Such examples can be difficult to schedule as exploring the space of all possible parameters is prohibitive. They would, therefore, benefit the most from the cost estimation and scheduling system developed in WP3.

4.3.3 Workflows, Long-running Queries, Adaptive Computations

The objectives of *Work Package 5: Adaptive Data Analytics* (WP5) include the development of methods for monitoring long-running queries and long workflows, allowing analytics experts to understand partial results or intermediate data and calibrate the parameters of the query without stopping it. Such functionality will best benefit computations that involve long workflows or dataflow graphs of queries, where monitoring the intermediate results may give insights much quicker than waiting for the whole computation. As both applications include scenarios that may result in such computations, the elicited use cases also demonstrate these desirable properties, and thus stand to gain from the developed adaptive query technology.

4.3.4 Resource-bound Performance Constraints

In many cases, even when there are alternative implementations for parts of a workflow, storage formats, execution engines, etc, it is not clear that the most time-efficient should always be selected, when optimizing for total cost. Research in WP2 aims to allow for constrained cost optimization in the scheduler and to benefit such use cases that include queries with constrained and resource-bound scheduling performance constraints.

4.3.5 Visualization of Intermediate Results

Finally, *Work Package 6: Information Visualization* (WP6) aims to develop visualization tools for monitoring query data and also visualizing intermediate information during query execution, in combination with work in WP5. Indeed, the selected use cases include computations that produce visualizable information, including as intermediate data in long-running workflows.

Chapter 5

Conclusion

This document describes the early results of the requirements analysis process in ASAP. The process has resulted in five use cases covering interesting aspects and computations from both analytics applications. Each selected use case motivates several research and optimization topics in the research work packages. In addition, the use cases will assist in designing mock-ups, early prototype implementations and evaluation benchmarks. Finally, as the requirement analysis of the ASAP applications is an evolving process, dependent on the results and design of the planned research, this User Requirement Specification is expected to evolve accordingly in the future.

Bibliography

[1] H. D. Benington. Production of large computer programs. In *Proceedings of the 9th International Conference on Software Engineering*, ICSE '87, pages 299–310, Los Alamitos, CA, USA, 1987. IEEE Computer Society Press.

[2] Kevin S Beyer, Vuk Ercegovac, Rainer Gemulla, Andrey Balmin, Mohamed Eltabakh, Carl-Christian Kanne, Fatma Ozcan, and Eugene J Shekita. Jaql: A scripting language for large scale semistructured data analysis. In *Proceedings of VLDB Conference*, 2011.

[3] Dhruba Borthakur. Hdfs architecture guide. *HADOOP Apache Project Documentation*, 2008.

[4] Biswapesh Chattopadhyay, Liang Lin, Weiran Liu, Sagar Mittal, Prathyusha Aragonda, Vera Lychagina, Younghee Kwon, and Michael Wong. Tenzing a sql implementation on the mapreduce framework. In *Proceedings of VLDB*, pages 1318–1327, 2011.

[5] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. *Text Processing with GATE (Version 6).* 2011.

[6] Michael Hicks and Jeffrey S. Foster. Score: Agile research group management. *Commun. ACM*, 53(10):30–31, October 2010.

[7] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 782–792, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[8] Rafal Kuc and Marek Rogozinski. *ElasticSearch server.* Packt Publishing Ltd, 2013.

[9] Michael McCandless, Erik Hatcher, and Otis Gospodnetic. *Lucene in Action, Second Edition: Covers Apache Lucene 3.0.* Manning Publications Co., Greenwich, CT, USA, 2010.

[10] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 1–8, New York, NY, USA, 2011. ACM.

[11] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. Pig latin: a not-so-foreign language for data processing. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 1099–1110, New York, NY, USA, 2008. ACM.

[12] Rob Pike, Sean Dorward, Robert Griesemer, and Sean Quinlan. Interpreting the data: Parallel analysis with sawzall. *Sci. Program.*, 13(4):277–298, October 2005.

[13] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu, and Raghotham Murthy. Hive-a petabyte scale data warehouse using hadoop. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 996–1005. IEEE, 2010.

[14] Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 2012.

[15] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 2–2, Berkeley, CA, USA, 2012. USENIX Association.

FP7 Project ASAP

Adaptable Scalable Analytics Platform

End of D1.1 Early User Requirements

WP 1 – Architecture and Requirements Analysis

Nature: Report

Dissemination: Public