University of Peloponnese

Department of Informatics and Telecommunications
Tripoli, Greece

ADVANCES IN METRICS AND E-COMMERCE EVALUATION TECHNIQUES

Author: Constantine J. Aivalis

PhD Thesis

May 2020

Supervisor:
Professor Anthony C. Boucouvalas

ACKNOWLEDGMENTS

I would like to I would like to express my sincere gratitude to my PhD advisor, Professor Anthony C. Boucouvalas for supporting me during the past years and sharing his knowledge, ideas and advice. Anthony is someone you will immediately like and never forget once you have met him. He has excellent humor, is patient, enthusiastic, energetic, is a brilliant speaker even to wide audiences and has a wide spectrum of interests.

I greatly appreciate his help during this task and am looking forward to working with him in the future.

CONTENTS

Acknowledgments 3
Contents 5
List of Figures and Pictures 7
0. Abstract 9
1. ΠΕΡΊΛΗΨΗ 17
2. Foreword 25
2.1. Key Performance Metrics *28*
2.2. Apdex index *30*
3. Introduction 33
3.1. Motivation *33*
3.2. Outline *36*
3.3. Analytics Status Quo *38*
3.4. Importance of Gaining Insite *41*
4. Literature Review 42
5. Web Analytics Systems 49
5.1. Analytics Goals *49*
5.2. The Access Log File *50*
5.3. How to Generate the Access Log File *53*
5.4. Four Types of Analytics Systems *56*
5.5. Logfile Analysis VS. Tagging *57*
5.6. Page Tagging Systems 60
5.7. Usage Example *61*
6. E-Commerce Application Characteristics 65
6.1. Importance of e-commerce systems *65*
6.2. Database Entity Relationship Diagram *70*
7. Specifications and Requirements for the Analyzer Application 72
7.1. Introduction *72*
7.2. Requirements *74*
7.3. On-Demand Analyzer *77*
7.4. Real-Time Analyzer *79*
7.5. Near Real Time With External Data *82*
7.6. Type 1: Standard Logfile Analyzer *91*
7.7. Type 2: Extended Log File Analyzer *96*
7.8. Type 3: Hybrid Logfile Analyzer *98*
7.9. Type 4: Social Media Aware Analyzer *100*
8. Analyzer Application Details 102
8.1. Operating Environment Specifications *102*
8.2. Application Perspective *103*
8.3. User Interface *104*
8.4. Functionality *107*
8.5. Main Frame and Main Menu *107*
8.6. Logfile Loading *108*
8.7. Single Logfile Loader *109*
8.8. Rotatable Multi Logfile Load *110*
8.9. Logfile Loading Process *111*
8.10. Batch Processing *116*
8.11. Session Processing *121*

8.12. Streaming and Daemons *132*
8.13. Visualization *135*
8.14. Web Application *144*
8.15. Database *152*
9. Big Data Technologies 157
9.1. Introduction To Big Data *158*
9.2. Definition of Big Data *159*
*9.3. Hadoop and MapReduce Expedite **Volume*** *160*
*9.4. NoSQL Databases Promote **Variety*** *162*
*9.5. Hadoop and **Velocity*** *164*
9.6. The Hadoop Ecosystem *168*
9.7. Big Data Infrastructure Vendors *171*
10. Extending Use of the Analyzer for Physical Retail Stores 173
10.1. Combining E-commerce Sites and Physical Stores *173*
10.2. Mobile Technologies *175*
10.3. The Physical Store *176*
10.4. The Electronic Commerce Site *177*
10.5. The Log File Analyzer *179*
10.6. The Combined Analyzer *181*
10.7. Enhancing Input *183*
10.8. Benefits *188*
11. Conclusions 190
12. Future Work 193
13. Literature 198

LIST OF FIGURES AND PICTURES

Figure 1: On-Demand Analyzer Architecture 16
Figure 2: Six Number Pair Sets with Same Average 31
Figure 3: Amazon.de categories 34
Figure 4: Amazon Product Category Combo box 35
Figure 5: Web Analytics Considered Integral Component of E-Commerce........ 39
Figure 6: AWStats Summary 40
Figure 7: A Bug's Life 52
Figure 8: Brian Clifton's Page Tagging VS. Logfile Analysis Table 57
Figure 9: Google Analytics Access Results 61
Figure 10 Request action with productId and SessionId 63
Figure 11 Monthly Traffic Diagram........ 63
Figure 12: Components of an E-Commerce System........ 69
Figure 13: ERD of a Typical ECommerce Application........ 70
Figure 14: Use-Case Diagram of the LFA 73
Figure 15: Enhanced Analyzer Architecture........ 77
Figure 16: On Demand Analyzer Modules........ 78
Figure 17: Real Time Analyzer Modules........ 80
Figure 18: Near Real Time Big Data Analyzer........ 81
Figure 19: External Data Extensions........ 86
Figure 20: NRT Order Processing UML........ 88
Figure 21: Database Trigger Sample........ 90
Figure 22: Database Table for NRT Information Transmission........ 90
Figure 23: Type 1, Traditional Log File Analyzer 91
Figure 24: Type 2 LFA Dataflow Diagram........ 97
Figure 25: Hybrid LFA........ 99
Figure 26: Social Media Processor Integration........100
Figure 27: Class Diagram for Dialogs 104
Figure 28: Sites Dialog........106
Figure 29: Main Frame of The LFA........107
Figure 30: Logfile Loaders........108
Figure 31: Single Log File Loader 110
Figure 32: Rotatable Logfile Loader 111
Figure 33: Batch Processing 116
Figure 34: Extract From The Bot And Spider Database 117
Figure 35: Selecting A Batch 118
Figure 36: Searching For Session-IDs........119
Figure 37: Hostname of Remote Address........120
Figure 38: Finding The Session 121
Figure 39: Session Found 122
Figure 40: CBMG For a Session........123

Figure 41: Manipulating Nodes....................124
Figure 42: Session 6734....................124
Figure 43: Compound CBMG....................125
Figure 44: Manipulating nodes....................126
Figure 45: Struts Actions....................127
Figure 46: RabbitMQ Message Flow....................133
Figure 47: Logfile Deltas Poling....................134
Figure 48: Database and Query Dialog....................136
Figure 49: The Evaluation Menu and its Menu items....................137
Figure 50: Statistics Dialog....................138
Figure 51: Access Per Product Date Selection Dialog....................139
Figure 52: Access Per Product Form Filled....................139
Figure 53: Access Per Product Report Dialog....................140
Figure 54: Export Options For the Report....................141
Figure 55: JasperSoft Studio Editor....................142
Figure 56: Chart Data Configuration Dialog....................143
Figure 57: Web Server Applications Topology....................144
Figure 58: User Interface of the web Application....................146
Figure 59: Demographic Information aboout visitor location....................147
Figure 60: System Information....................148
Figure 61: Traffic Sources....................149
Figure 62: Real Time Session Graph....................150
Figure 63: Average Speed Gauge....................151
Figure 64: Analyzer Database ERD....................154
Figure 65: Actions per Product Pie Chart....................175
Figure 66: The Typical Store MIS....................177
Figure 67: The Typical e-Commerce Application....................179
Figure 68: The Log File Analyzer....................180
Figure 69: The Log File Analyzer Combining e-Shop and MIS....................182
Figure 70: Mobile Technologies and LFA....................184
Figure 71: Mobile Data Capturing Mechanism....................185
Figure 72: Order Processing....................187

0. ABSTRACT

The "Abstract" chapter starts with an attempt to define Analytics and Analytics application characteristics and scope and present the overlap of Analytics with existing sections of Informatics. It continues with the definition of the goals of this work, a short mention of the evolution steps the developed software underwent, and the way Big Data and Social Media can enhance information. It explains methods and techniques that achieve these goals. The workflow and the necessary functionalities are listed. Then it summarizes the problem of why accurate performance measuring is necessary for a web or e-commerce application, the steps taken in order to achieve creating an innovative system as a basis for evaluating the web site and providing a usable tool to the administrators to produce visuals and statistics. Shortly describes the needs of an e-commerce site in terms of operational insight and describes the initial approach of the LFA produced.

Web Application Evaluation and Analytics Systems is intrinsically a multidisciplinary topic of Informatics, since its primary goal is to collect data from every possible facet of the operational environment of a web application system, process it and provide insight to the management. Analytics applications are primarily based on Network Programming techniques, Data Structure Design and Implementation, Database Management Systems, Human Factors and Interfaces, Web Development Frameworks, Data and Information Visualization and Data Mining. Social Media with their broad content and their Application Programming Interfaces, as well as mobile devices which enhance the experience by extending the geographical boundaries of their accessibility.

This research aims at analyzing the status quo, studying and proposing advances of Web Analytics applications and providing a software prototype, including innovative Analytics techniques. Problems and arising issues are pointed out, and appropriate solutions and techniques are presented, that provide software specifications and implementations of systems that offer full insight to the way web and E-Commerce applications operate, as well as to the mode they are visited and used by customers and the public using the Internet. Metrics concerning performance and customer habits and visitor behavior are examined and various algorithms and environments that have been developed to provide them are described. Hardware and Software innovations resulting to a perpetual evolution of platforms and foundations used to develop and operate Web Applications are taken into consideration.

How the evolution of platforms is dealt with, is an interesting topic that has been analyzed here as well. This perpetual evolution triggers the development of appropriate techniques to accommodate and support adaptive measurement technologies and Analytics applications. Several solutions that have been implemented are presented in this work. Big Data techniques, that allow horizontal scaling of the volume of data the application can support, as well as enrichment of the variety of data sources provide a more accurate, higher velocity, saving time and give a more exact picture of the operation and finally remote accessibility to the e-commerce application. In addition to data collected internally by the web server, running the web application and enriched by Big Data techniques, data sources such as various Social Media Applications are also used to enhance the information collection ever further. Social Media offer large possibilities of combining personal information from external sources and allow the analyzer to complete the insight.

There are various forms of Analytics applications and add-ons. In general Analytics applications can be viewed as compound systems or often as mash-up applications that operate as web-based applications, desktop applications or even both. The workflow consists of four general groups of functionalities that can be conceptually viewed and implemented in a variety of ways:

- Data Collection
- Data Preparation & Storage
- Computational and Mining
- Result Presentation and Data Visualization

The data collection procedure traditionally is based on extracting data from various sources and log files that are being generated by the web server, hosting the web application. Additional sources are used in parallel that enhance the information and provide a more consistent and global view of the users' profile.

The collected data need preparation before storage. This preparation involves locating the data sources, cleaning up redundant and unnecessary data, separating data fields, grouping and indexing. Storage takes place mainly in relational database systems, to provide flexible support for complicated queries, data correlations and searches at a later phase.

Computational algorithms, data mining techniques and applications form the kernel-layer of the Analyzer. They generate metrics based on the collected data sets and enhance the existing data sets by importing additional information from external sources.

Modular design and integration with contemporary report generation and visualization systems allow feeding the visualization subsystems with data. Interactive data visualization systems establish two-way communication with the computational subsystems and provide excellent interfaces for filtering and extracting results and metrics.

This research focuses on ways of extending the input data of an analyzer to support retail businesses transactions as well. The key clicks of the web application users are being substituted by sensor input, automatically generated by mobile applications that are activated by retail customers transaction data while the customer is visiting the shopping floor and searches through the aisles of the shopping floor for products. A sensor mobile device environment is particularly cheap, easy to integrate and useful, especially in situations where the retailer operates electronic shopping applications in parallel, selling online the same products that are being offered on the shelves.

Like most web applications and services, e-commerce applications are often implemented without significant built-in subsystems and modules acting as internal performance measuring mechanisms. Developers constantly strive to program as efficiently as they can, using the best and fastest possible run-time optimization techniques and tools and optimize their code as well as they can. Other than that, overall performance measurement is not a primary responsibility of the programmers. The reason lies mainly in the fact that performance of web applications happens to be heavily dependent on the environment characteristics they operate in. These environments are usually complex, and consist of servers and various network connections, as well as services that happen to be distributed externally, often in other countries, like payment portals and security certificates. Thus, measurements need to be designed in a way that extends beyond the core software application implemented by the developer team whose main goal is to fulfill the functional requirements of the designers. Because responsiveness is always a major factor, since high revenue is sought for in all e-commerce applications, it is important to perform these measurements well and accurately.

Occasionally overall response times increase and the administrators responsible often lack the means to even notice that. To remedy such conditions, it is crucial to have a precise performance, user-action and behavior measuring and visualizing system on hand, that will make all bottlenecks and problems visible and will even predict shortage of resources. Techniques used for obtaining operational and transactional data and presenting them on time to monitor the safe operation of any web site, and particularly an e-commerce site are presented in this work.

Innovative ideas range from log file data enhancement techniques to real time visualization and customer behavioral pattern analysis. A customizable and extendable log file analyzer has been developed. During the development phase, four distinctive versions of the Analyzer Application have been produced. They can be used interchangeably according to the operational environment in scrutiny. These versions are being presented as steps of development in detail in the current document.

They can be viewed as complementary functionalities. Important requirements for the software developer are ease of installation, integration, configuration and tuning. Portability across operating environments and possibility of combining cross platform installations lead to using 100% Java as a developed platform. This way, evaluating any e-commerce application becomes easy for the administrators.

Issues involved in creating a prototype customizable e-commerce log file analyzer for measuring customer access to E-shops are pointed out and solutions provided. The analyzer is basically a toolbox, consisting of a set of the necessary tools to load the necessary data into its database and provide exact insight for customer access and system response of e-shops. Insight can be obtained through generation of extensive reporting, graphical reports and various visuals and statistics.

The analyzer provides answers to standard questions such as: How many times has a specific product been added to the cart over a period? What is the average, maximum or minimum duration of visits? How many bytes were sent from the web server per day? What is the average duration of a complete payment cycle? How many customers have visited the site in a specified time span? How much revenue do we make per day or per hour? However, it provides answers to more interesting questions on session analysis such as what the customer route within the web application was, which web pages or selections are only seldom visited and the profile characteristics of the visitors. Additionally, comparisons between different e-shops and comparisons with previous years, months and days are feasible and easy, as is error reporting and detection as well.

Our first approach of this research is based on the first four steps of the Quantitative Analysis Cycle of an E-business Site [1]:

1. Insight of e-business site architecture
2. Measuring system performance from different reference points
3. Understanding customer behavior by generating a Customer Behavior Model Graph [2]
4. Workload and Session analysis

The final approach is providing a platform that will promote:

5. Performance model development
6. Performance parameter definition
7. Workload forecasting
8. Prediction of site performance

The e-shop log file analysis tool can transparently display user actions and allow management to locate weak spots of the e-shop design, since it provides information about all user selected paths before a successful purchase or even an unsuccessful purchase attempt. It allows measurement of the times between all steps involved. Most open-source e-commerce solutions offer some statistical tools, like reports displaying orders per day or highest selling items. These reports inform the staff about daily procedures and only successful purchases. The log file analysis application, on the other hand, is more comprehensive and powerful informing about performance, user behavior and user preferences.

The goal of a business to customer (B2C) e-shop application is to promote retail sales and create profit. A virtual store allows buying products or services through a website, in analogy to a bricks-and-mortar retailer or a shopping mall. The Internet is no longer a niche technology – it is mass media and an utterly integral part of modern life. Over 85 per cent of the world's online population has used the Internet to make a purchase. Intention to shop online in Europe is high.79 percent of online European consumers plan to purchase products or services via the Internet in the next six months. Online consumers in Norway and Great Britain show the greatest propensity with almost 90 percent planning a web purchase soon [3]. The e-shop must have a minimal interface, consisting of search engines and product presentation mechanisms. They must also be able to support easy and quick adding of items to the cart and finally allow secure payments and possibly offer one-page checkout.

Deficient performance of an e-shop will lead to lost revenue. According to the so-called 8-second rule, a user will not tolerate delays longer than 8 seconds per page-refresh of a website, not even if the user is equipped with a low-speed dial-up connection. This forces the e-shop designer, to design and implement every page as efficiently as possible. In 2001 Zona Research report more than $25 billion in potential lost business due to Web performance issues [4]. Today, not only overall bandwidth has increased dramatically, but also the number of users, the demand for multimedia and the overall traffic. The 8-second rule still applies and the need to measure performance is still valid but under very different conditions.

The web server can be configured in such a way that any access to the e-shop can be registered into an access log file. The remote IP address, time stamp of access, requested or sent object, size in bytes, duration etc. are registered here. The log file analyzer mainly operates on this file.

Many issues involved with e-shop log files must be considered. Successful sale sessions are always fewer than the total number of sessions. Additionally, to human users we have sessions created by crawlers and robots. These sessions alienate the measurements.

Web Robots (also known as Web Wanderers, Crawlers, or Spiders), are programs that traverse the Web automatically. Search engines, agents and research applications use them to index the web content, spammers use them to scan for email addresses, and they have many other uses [5]. Robots can be identified through their behavior [6] but it may not always be feasible to detect them. The assumption is made that robots never enter the pay section of the e-shop. Still, theoretically a robot could be used to buy products. [6] [7]

The log file analyzer we have developed contains a toolbox with specialized tools necessary for measuring performance of e-shops as well as customer behavioral patterns. Standard general-purpose log file analyzers usually process solely log files, to evaluate access hits, calculate bandwidth and report visited pages on hourly and daily basis, as well as visitor countries and browser-agent statistics [7] [8]. This information is very useful for a content management system, a portal, or even a static website administrator, because the pages visited, and the visit durations are enough to measure the success of the site. An e-commerce site administrator, on the other hand, needs more specific information about the performed actions and transactions, which must be combined with the log file data. E-shop specific data about products, product-categories, orders and customers are used in our toolbox to gain more precise information of the access events. This way, the end user deals with more business-specific objects, since familiar terms and items are used. That makes the application easier to adopt by the administrator.

The initial approach, as seen in Figure 1, has been built as a standalone application with a simple, easy to use and intuitive graphical user interface, it maintains its own database and includes options that allow its user to easily adapt and load data from both log files (Connection ❶) and bidirectionally with the e-shop data base (Connection ❷). This model has been used as the base application upon which various extensions and addon features were implemented. This application can run anywhere, not necessarily on the machine where the web server resides. It can accommodate multiple e-shops, running on multiple web server architectures and provides a basis for the later approaches. This is a typical compound menu-driven ETL[1] application which includes integrated visualization components and mechanisms that integrate data and information of the e-shop into the analyzer.

The second approach integrates external metrics, collected from a tagging Analytics provider, like Google Analytics. The Java API allows access to registered users. This addition turns the entire system to hybrid. Hybrid Analytics applications alleviate all limitations of pure log file analyzers and tagging systems.

These two initial approaches are further extended and Log File Analyzers with additional capabilities were designed and are described and presented in this document.

[1] Extract Transform Load

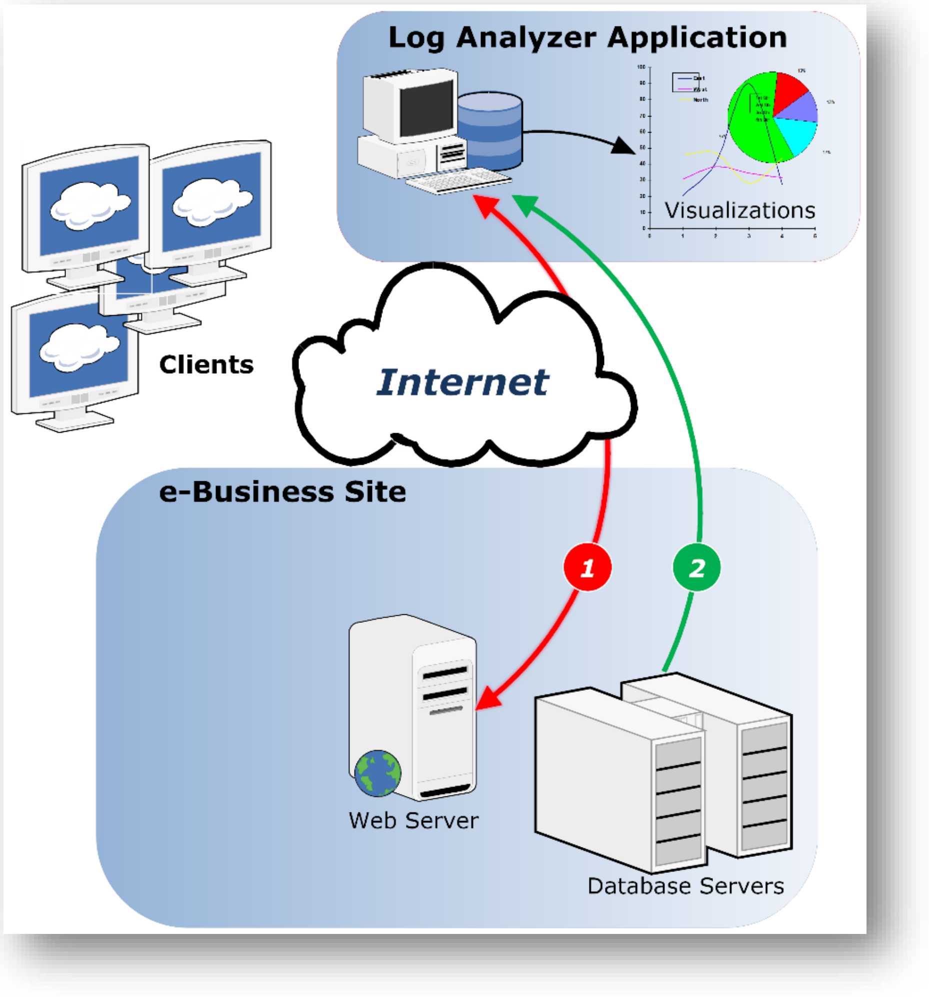


FIGURE 1: ON-DEMAND ANALYZER ARCHITECTURE

The following section contains an introduction to basic performance testing techniques and adapts the application key performance metrics from Stackify to web applications. It further explains how the application Performance index is calculated.

1. ΠΕΡΊΛΗΨΗ

Το κεφάλαιο "Περίληψη" ξεκινά με μια προσπάθεια ορισμού της έννοιας Analytics και των χαρακτηριστικών και του πεδίου εφαρμογής τους και παρουσιάζει την επικάλυψη με άλλους τομείς της Πληροφορικής. Περιέχει καθορισμό των στόχων αυτής της εργασίας, μια σύντομη αναφορά στην εξέλιξη του συστήματος λογισμικού Analytics, του τρόπου με τον οποίο τα Μεγάλα Δεδομένα και τα κοινωνικά μέσα ενισχύουν τις πληροφορίες και εξηγεί μεθόδους και τεχνικές που τις επιτυγχάνουν. Περιγράφονται η ροή εργασιών και τα απαραίτητα λειτουργικά χαρακτηριστικά ενός συστήματος Analytics. Στη συνέχεια, αιτιολογεί γιατί είναι αναγκαία η ακριβής μέτρηση των επιδόσεων για μια εφαρμογή ιστού ή ηλεκτρονικού εμπορίου, περιγράφει τα βήματα που έχουν ληφθεί για να επιτευχθεί η δημιουργία ενός καινοτόμου συστήματος που χρησιμεύει ως βάση για την αξιολόγηση ενός ιστοτόπου και η παροχή ενός χρήσιμου εργαλείου στους διαχειριστές να παράγουν οπτικά και στατιστικά στοιχεία. Περιγράφει σύντομα τις ανάγκες ενός ιστοτόπου ηλεκτρονικού εμπορίου όσον αφορά την επιχειρησιακή γνώση και περιγράφει την αρχική προσέγγιση υλοποίησης του LFA.

Η αξιολόγηση διαδικτυακών εφαρμογών και τα συστήματα Analytics είναι εγγενώς ένας διεπιστημονικός τομέας της πληροφορικής, με το δεδομένο ότι ο πρωταρχικός στόχος της είναι η συλλογή δεδομένων από κάθε πτυχή του λειτουργικού περιβάλλοντος μίας διαδικτυακής εφαρμογής, η επεξεργασία του και η παροχή της πληρέστερης δυνατής εικόνας στη διοίκηση. Οι εφαρμογές των συστημάτων Analytics βασίζονται κατά κύριο λόγο σε τεχνικές διαδικτυακού προγραμματισμού, σχεδίαση και υλοποίηση δομών δεδομένων, σε συστήματα διαχείρισης βάσεων δεδομένων, συστήματα ανάλυσης ανθρώπινου παράγοντα και διεπαφών, σε frameworks κατασκευής διαδικτυακών συστημάτων, σε συστήματα οπτικοποίησης δεδομένων και πληροφοριών και σε τεχνικές εξόρυξης δεδομένων. Τα κοινωνικά δίκτυα με το ευρύ περιεχόμενο και τις προγραμματιστικές τους διεπαφές καθώς και τις κινητές συσκευές που ενισχύουν την εμπειρία, επεκτείνοντας τα γεωγραφικά όρια της προσβασιμότητας τους.

Αυτή η έρευνα στοχεύει στην ανάλυση της παρούσας κατάστασης, τη μελέτη και την πρόοδο των εφαρμογών Web Analytics και την παροχή πρωτοτύπου λογισμικού, συμπεριλαμβανομένων καινοτόμων τεχνικών Analytics. Παρουσιάζονται προβλήματα και λεπτομέρειες υλοποίησης και παρουσιάζονται κατάλληλες λύσεις και τεχνικές που παρέχουν προδιαγραφές λογισμικού και εφαρμογές συστημάτων που προσφέρουν όσο το δυνατόν πιο πλήρη γνώση του τρόπου λειτουργίας των διαδικτυακών εφαρμογών και των εφαρμογών Ηλεκτρονικού Εμπορίου καθώς και του τρόπου με τον οποίο δέχονται επισκέψεις και χρησιμοποιούνται από τους πελάτες και το κοινό που επικοινωνεί μαζί τους χρησιμοποιώντας το Διαδίκτυο. Υπολογίζονται μετρήσεις που αφορούν τις επιδόσεις και τις συνήθειες των πελατών και τη συμπεριφορά των επισκεπτών και περιγράφονται διάφοροι αλγόριθμοι και περιβάλλοντα που έχουν αναπτυχθεί για την παροχή τους. Οι καινοτομίες στο το υλικό και το λογισμικό, οι οποίες έχουν ως αποτέλεσμα την συνεχή εξέλιξη των συστημάτων και των υποδομών που χρησιμοποιούνται για την ανάπτυξη και τη λειτουργία εφαρμογών ιστού, λαμβάνονται υπόψη.

Πώς αντιμετωπίζεται η εξέλιξη των frameworks, είναι ένα ενδιαφέρον θέμα που έχει αναλυθεί και εδώ. Αυτή η αέναη εξέλιξη ενεργοποιεί την ανάπτυξη κατάλληλων τεχνικών για την υποδοχή και υποστήριξη προσαρμοστικών τεχνολογιών μέτρησης και εφαρμογών Analytics. Στην εργασία αυτή παρουσιάζονται οι διάφορες λύσεις που έχουν εφαρμοστεί. Οι τεχνικές μεγάλων δεδομένων, που επιτρέπουν το οριζόντιο scaling του όγκου των δεδομένων που μπορεί να υποστηρίξει η εφαρμογή, καθώς και ο εμπλουτισμός της ποικιλίας των πηγών δεδομένων παρέχουν πιο ακριβή, υψηλότερη ταχύτητα, εξοικονομώντας χρόνο και δίνουν μια πιο ακριβή εικόνα της λειτουργίας και τέλος μεγαλύτερη προσβασιμότητα στην εφαρμογή ηλεκτρονικού εμπορίου. Εκτός από τα εσωτερικά δεδομένα που συλλέγονται από τον εξυπηρετητή διαδικτύου της εφαρμογής, με την εκτέλεση της διαδικτυακής εφαρμογής και εμπλουτισμένα με τεχνικές μεγάλων δεδομένων, χρησιμοποιούνται επίσης εξωτερικές πηγές δεδομένων, όπως διάφορες εφαρμογές Κοινωνικών Μέσων για την περαιτέρω βελτίωση της συλλογής πληροφοριών. Τα Κοινωνικά Μέσα προσφέρουν μεγάλες δυνατότητες συνδυασμού προσωπικών πληροφοριών από εξωτερικές πηγές και επιτρέπουν στον αναλυτή να εμπλουτίσει τις μετρήσεις του και να ολοκληρώσει την εικόνα καλύτερα.

Υπάρχουν διάφοροι τύποι εφαρμογών Analytics, καθώς και εργαλεία λογισμικού. Σε γενικές γραμμές, οι εφαρμογές Analytics μπορούν να θεωρηθούν ως σύνθετα συστήματα, που εμφανίζονται συχνά ως διαδικτυακές εφαρμογές, εφαρμογές γραφείου ή και οι δύο. Η ροή εργασίας αποτελείται από τέσσερις γενικές ομάδες λειτουργειών που μπορούν να θεωρηθούν και να υλοποιηθούν με πολλούς τρόπους:

- Συλλογή δεδομένων
- Προετοιμασία και αποθήκευση δεδομένων
- Υπολογισμοί και Εξόρυξη Δεδομένων
- Παρουσίαση αποτελεσμάτων και οπτικοποίηση δεδομένων

Παραδοσιακά η διαδικασία συλλογής δεδομένων βασίζεται στην εξαγωγή δεδομένων από διάφορες πηγές και αρχεία καταγραφής συμβάντων (log files) που παράγονται από τον εξυπηρετητή διαδικτύου που φιλοξενεί την διαδικτυακή εφαρμογή. Πρόσθετες πηγές χρησιμοποιούνται παράλληλα, για να βελτιώσουν, να «ενισχύσουν» τις πληροφορίες και να δώσουν μια πιο συνεκτική και σφαιρική άποψη του προφίλ των χρηστών.

Τα δεδομένα που συλλέγονται χρειάζονται προετοιμασία πριν από την αποθήκευση. Αυτή η προετοιμασία περιλαμβάνει τον εντοπισμό των πηγών δεδομένων, τον καθαρισμό επαναλαμβανόμενων και περιττών δεδομένων, τον διαχωρισμό των πεδίων δεδομένων, την ομαδοποίηση και το indexing. Η αποθήκευση πραγματοποιείται κυρίως σε συστήματα σχεσιακών βάσεων δεδομένων, για να παρέχει ευέλικτη υποστήριξη για σύνθετα ερωτήματα και αναζητήσεις συσχετισμών δεδομένων σε μεταγενέστερη φάση.

Οι υπολογιστικοί αλγόριθμοι, οι τεχνικές εξόρυξης δεδομένων και οι εφαρμογές αποτελούν τον πυρήνα του Αναλυτή. Δημιουργούν μετρήσεις με βάση τα συλλεχθέντα σύνολα δεδομένων και βελτιώνουν τα υπάρχοντα σύνολα δεδομένων εισάγοντας πρόσθετες πληροφορίες από εξωτερικές πηγές.

Ο αρθρωτός σχεδιασμός και η ενσωμάτωση με σύγχρονα συστήματα δημιουργίας και απεικόνισης αναφορών επιτρέπουν την τροφοδοσία των υποσυστημάτων οπτικοποίησης με δεδομένα. Τα συστήματα αλληλεπιδραστικής απεικόνισης δεδομένων δημιουργούν αμφίδρομη επικοινωνία με τα υπολογιστικά υποσυστήματα και παρέχουν εξαιρετικές διεπαφές για το φιλτράρισμα και την εξαγωγή αποτελεσμάτων και μετρήσεων.

Αυτή η έρευνα επικεντρώνεται και στους τρόπους επέκτασης των δεδομένων εισόδου ενός αναλυτή για την υποστήριξη και συναλλαγών φυσικών επιχειρήσεων λιανικού εμπορίου. Οι βασικές λειτουργίες των χρηστών εφαρμογών διαδικτύου αντικαθίστανται από καταγραφές δεδομένων που καταγράφονται από αισθητήρες και παράγονται αυτόματα από εφαρμογές φορητών συσκευών που ενεργοποιούνται από δεδομένα συναλλαγών πελατών λιανικής ενώ ο πελάτης επισκέπτεται το εμπορικό πάτωμα και επιλέγει αναζητώντας μέσω των διαδρόμων του εμπορικού ορόφου για προϊόντα. Ένα περιβάλλον κινητής συσκευής αισθητήρα είναι ιδιαίτερα φθηνό, ενσωματώνεται εύκολα στην επιχείρηση και είναι χρήσιμο, ειδικά σε περιπτώσεις όπου ο λιανοπωλητής διαχειρίζεται παράλληλα και ηλεκτρονικό κατάστημα, πωλώντας διαδικτυακά τα ίδια προϊόντα που προσφέρονται στα ράφια.

Όπως και οι περισσότερες εφαρμογές και διαδικτυακές υπηρεσίες, οι εφαρμογές ηλεκτρονικού εμπορίου υλοποιούνται συχνά χωρίς σημαντικά ενσωματωμένα υποσυστήματα και ενότητες που να λειτουργούν ως εσωτερικοί μηχανισμοί μέτρησης απόδοσης. Οι προγραμματιστές προσπαθούν συνεχώς να προγραμματίζουν όσο πιο αποτελεσματικά μπορούν, χρησιμοποιώντας τις καλύτερες και ταχύτερες τεχνικές και εργαλεία βελτιστοποίησης χρόνου εκτέλεσης και βελτιστοποιώντας τον κώδικα τους όσο μπορούν. Εκτός αυτού, η μέτρηση της συνολικής απόδοσης δεν αποτελεί πρωταρχική ευθύνη των προγραμματιστών. Ο λόγος έγκειται κυρίως στο γεγονός ότι η απόδοση των εφαρμογών ιστού εξαρτάται σε μεγάλο βαθμό από τα χαρακτηριστικά περιβάλλοντος στα οποία λειτουργούν. Τα περιβάλλοντα αυτά είναι συνήθως πολύπλοκα και αποτελούνται από διακομιστές και διάφορες συνδέσεις δικτύου, καθώς και από υπηρεσίες που διανέμονται εξωτερικά , συχνά σε άλλες χώρες, όπως οι πύλες πληρωμών και τα πιστοποιητικά ασφαλείας. Έτσι, οι μετρήσεις πρέπει να σχεδιάζονται με τρόπο που εκτείνεται πέρα από την βασική εφαρμογή λογισμικού που εφαρμόζει η ομάδα προγραμματιστών, ο κύριος στόχος της οποίας είναι να εκπληρώσει τις λειτουργικές απαιτήσεις των σχεδιαστών. Επειδή η χρονική απόκριση είναι πάντα ένας σημαντικός παράγοντας, καθώς αναζητούνται υψηλά έσοδα σε όλες τις εφαρμογές ηλεκτρονικού εμπορίου, είναι σημαντικό να πραγματοποιούνται αυτές οι μετρήσεις ολοκληρωμένα και με ακρίβεια.

Περιστασιακά, οι συνολικοί χρόνοι απόκρισης αυξάνονται και οι αρμόδιοι διαχειριστές συχνά δεν έχουν τα μέσα για να το παρατηρήσουν. Για να αποκατασταθούν αυτές οι συνθήκες, είναι ζωτικής σημασίας να υπάρχουν συστήματα που να δίνουν ακριβείς πληροφορίες για την απόδοση και την δράση των χρηστών και μετρήσεις συμπεριφοράς και οπτικοποίησης της κατάστασης συστήματος, που θα καταστήσουν ορατά τα σημεία συμφόρησης και τα προβλήματα και θα προβλέψουν ακόμη και την έλλειψη πόρων. Οι τεχνικές που χρησιμοποιούνται για την απόκτηση επιχειρησιακών και συναλλακτικών δεδομένων και την έγκαιρη παρουσίασή τους για την παρακολούθηση της ασφαλούς λειτουργίας οποιουδήποτε ιστοτόπου, και ιδιαίτερα ενός δικτυακού τόπου ηλεκτρονικού εμπορίου παρουσιάζονται σε αυτό το έργο.

Οι καινοτομίες που παρουσιάζονται είναι οι τεχνικές εμπλουτισμού των δεδομένων του αρχείου καταγραφής συμβάντων και η οπτική απεικόνιση σε πραγματικό χρόνο και ανάλυση συμπεριφοράς πελατών. Έχει αναπτυχθεί ένας προσαρμόσιμος σε πολλά περιβάλλοντα και επεκτάσιμος αναλυτής αρχείων καταγραφής. Κατά τη διάρκεια της φάσης ανάπτυξης, κατασκευάστηκαν τέσσερις ξεχωριστές εκδόσεις της εφαρμογής. Μπορούν να χρησιμοποιηθούν εναλλακτικά ανάλογα με το λειτουργικό περιβάλλον που ελέγχουν. Αυτές οι εκδοχές παρουσιάζονται με λεπτομέρεια ως βήματα ανάπτυξης στο παρών έγγραφο.

Μπορούν να θεωρηθούν ως συμπληρωματικές λειτουργίες. Σημαντικές απαιτήσεις για τον προγραμματιστή λογισμικού είναι η ευκολία εγκατάστασης, ενσωμάτωσης, διαμόρφωσης και συντονισμού. Η φορητότητα σε λειτουργικά περιβάλλοντα και η δυνατότητα συνδυασμού εγκαταστάσεων πολλαπλών πλατφορμών οδήγησαν στη χρήση 100% Java ως περιβάλλον υλοποίησης. Με αυτόν τον τρόπο, η αξιολόγηση της εφαρμογής ηλεκτρονικού εμπορίου καθίσταται εύκολη για τους διαχειριστές.

Εξετάζονται προβλήματα και θέματα που σχετίζονται με τη δημιουργία ενός πρωτοτύπου προσαρμόσιμου αναλυτή αρχείων καταγραφής ηλεκτρονικού εμπορίου για τη μέτρηση της πρόσβασης των πελατών σε ηλεκτρονικά καταστήματα και παρουσιάζονται λύσεις τους. Ο αναλυτής είναι βασικά μία εργαλειοθήκη που αποτελείται από ένα σύνολο από εργαλεία λογισμικού για τη φόρτωση των απαραίτητων δεδομένων στη βάση δεδομένων του και παρέχει ακριβή εικόνα για την πρόσβαση των πελατών και την ανταπόκριση του συστήματος στα ηλεκτρονικά καταστήματα. Η πληροφόρηση επιτυγχάνεται μέσω της δημιουργίας εκτεταμένων αναφορών, γραφικών αναφορών, οπτικοποιήσεων και παροχής στατιστικών στοιχείων.

Ο αναλυτής παρέχει απαντήσεις σε τυποποιημένες ερωτήσεις όπως: Πόσες φορές έχει προστεθεί ένα συγκεκριμένο προϊόν στο καλάθι σε μια περίοδο; Ποια είναι η μέση, μέγιστη ή ελάχιστη διάρκεια επισκέψεων; Πόσες απομακρυσμένες αποστολές στέλνονταν από τον διακομιστή ανά ημέρα; Ποια είναι η μέση διάρκεια ενός πλήρους κύκλου πληρωμής; Πόσοι πελάτες έχουν επισκεφτεί τον ιστότοπο σε συγκεκριμένο χρονικό διάστημα; Πόσα έσοδα έχουμε ανά ημέρα ή ανά ώρα; Παρέχει όμως απαντήσεις σε πιο ενδιαφέρουσες ερωτήσεις σχετικά με την ανάλυση της συνόδου (session), όπως είναι η διαδρομή του πελάτη στις επιλογές της ιστοσελίδας, ποιες ιστοσελίδες ή επιλογές επισκέπτονται πελάτες σπάνια και παρουσιάζει τα χαρακτηριστικά του επισκέπτη. Επιπλέον, οι συγκρίσεις μεταξύ διαφόρων ηλεκτρονικών καταστημάτων και οι συγκρίσεις με προηγούμενα έτη, μήνες και ημέρες είναι εφικτές και γίνεται εύκολη η αναφορά και η ανίχνευση σφαλμάτων.

Η πρώτη μας προσέγγιση αυτής της έρευνας βασίζεται στα πρώτα τέσσερα βήματα του κύκλου ποσοτικής ανάλυσης ενός ιστοτόπου για ηλεκτρονικού επιχειρείν [1]:

1. Διερεύνηση της αρχιτεκτονικής του ιστοτόπου του ηλεκτρονικού επιχειρείν
2. Απόδοση του συστήματος μέτρησης από διαφορετικά σημεία αναφοράς
3. Κατανόηση της συμπεριφοράς των πελατών δημιουργώντας ένα γράφο μοντέλου συμπεριφοράς πελατών [2]
4. Ανάλυση φόρτου εργασίας και συνόδων

Η τελική προσέγγιση αποτελεί μια πλατφόρμα που θα προωθήσει:

5. Ανάπτυξη μοντέλου απόδοσης
6. Ορισμός παραμέτρων απόδοσης
7. Προβλέψεις φόρτου εργασίας
8. Πρόβλεψη της απόδοσης του χώρου

Το εργαλείο ανάλυσης αρχείων καταγραφής ηλεκτρονικού καταστήματος μπορεί να προβάλει με διαφάνεια τις ενέργειες των χρηστών και να επιτρέψει στη διεύθυνση να εντοπίσει αδύναμα σημεία του σχεδιασμού του ηλεκτρονικού καταστήματος, καθώς παρέχει πληροφορίες για όλες τις επιλεγμένες διαδρομές πριν από μια επιτυχημένη αγορά ή ακόμα και μια ανεπιτυχή προσπάθεια αγοράς. Επιτρέπει τη μέτρηση των χρόνων μεταξύ όλων των σχετικών βημάτων. Οι περισσότερες λύσεις ηλεκτρονικού εμπορίου ανοιχτού κώδικα προσφέρουν ορισμένα στατιστικά εργαλεία, όπως αναφορές που εμφανίζουν παραγγελίες ανά ημέρα ή προϊόντα με υψηλότερες πωλήσεις. Αυτές οι αναφορές ενημερώνουν το προσωπικό για τις καθημερινές διαδικασίες και μόνο όσον αφορά επιτυχημένες αγορές. Η εφαρμογή ανάλυσης αρχείων καταγραφής, από την άλλη πλευρά, είναι πιο ολοκληρωμένη και ισχυρή πληροφόρηση σχετικά με την απόδοση, τη συμπεριφορά των χρηστών και τις προτιμήσεις των χρηστών.

Ο στόχος της εφαρμογής ηλεκτρονικού καταστήματος B2C (business to customer) είναι να προωθήσει τις λιανικές πωλήσεις και να δημιουργήσει κέρδη. Ένα εικονικό κατάστημα επιτρέπει την αγορά προϊόντων ή υπηρεσιών μέσω μιας ιστοσελίδας, κατ' αναλογία με ένα φυσικό κατάστημα λιανικής πώλησης ή ένα εμπορικό κέντρο. Το Διαδίκτυο δεν είναι πλέον μια εξειδικευμένη τεχνολογία - είναι μέσο μαζικής ενημέρωσης και ένα αναπόσπαστο μέρος της σύγχρονης ζωής. Πάνω από το 85% του παγκόσμιου πληθυσμού στο διαδίκτυο έχει χρησιμοποιήσει το Διαδίκτυο για να κάνει μια αγορά. Περισσότεροι από τους μισούς χρήστες του Διαδικτύου είναι τακτικοί ηλεκτρονικοί αγοραστές που πραγματοποιούν ηλεκτρονικές αγορές τουλάχιστον μία φορά το μήνα [3]. Το ηλεκτρονικό κατάστημα πρέπει να διαθέτει μια ελάχιστη διεπαφή, αποτελούμενη από μηχανές αναζήτησης και μηχανισμούς παρουσίασης προϊόντων. Πρέπει επίσης να είναι σε θέση να προσθέτουν γρήγορα προϊόντα στο καλάθι και τελικά να επιτρέπουν ασφαλείς πληρωμές και ενδεχομένως να διαθέτουν γρήγορο checkout μονής σελίδας.

Η ανεπαρκής απόδοση ενός ηλεκτρονικού καταστήματος οδηγεί σε απώλεια εσόδων. Σύμφωνα με τον λεγόμενο «κανόνα των 8 δευτερολέπτων», ο χρήστης δεν θα ανεχτεί καθυστερήσεις μεγαλύτερες από 8 δευτερόλεπτα ανά ανανέωση σελίδας ενός ιστοτόπου, ακόμη και αν ο χρήστης είναι εξοπλισμένος με σύνδεση χαμηλής ταχύτητας μέσω τηλεφώνου. Αυτό αναγκάζει τον σχεδιαστή του ηλεκτρονικού καταστήματος να σχεδιάσει και να υλοποιήσει κάθε σελίδα όσο πιο αποτελεσματικά γίνεται. Η Zona Research, το 2001, αναφέρει περισσότερα από 25 δισεκατομμύρια δολάρια σε πιθανές απώλειες επιχειρήσεων λόγω προβλημάτων επιδόσεων ιστού [4]. Από τότε, όχι μόνο το συνολικό εύρος ζώνης έχει αυξηθεί δραματικά, αλλά και ο αριθμός των χρηστών, η ζήτηση για πολυμέσα και η συνολική κίνηση. Ο κανόνας των 8 δευτερολέπτων εξακολουθεί να ισχύει και η ανάγκη μέτρησης της απόδοσης εξακολουθεί να ισχύει αλλά υπό πολύ διαφορετικές συνθήκες.

Ο διακομιστής μπορεί να ρυθμιστεί κατά τέτοιο τρόπο ώστε οποιαδήποτε πρόσβαση στο ηλεκτρονικό κατάστημα να καταχωρείται σε ένα αρχείο καταγραφής πρόσβασης. Η διεύθυνση απομακρυσμένης διεύθυνσης IP, η χρονική σφραγίδα πρόσβασης, το αίτημα ή το αντικείμενο που στάλθηκε, το μέγεθος σε bytes, η διάρκεια φόρτωσης κ.λπ. καταχωρούνται εδώ. Ο αναλυτής αρχείου καταγραφής λειτουργεί κυρίως σε αυτό το αρχείο.

Πολλά ζητήματα που σχετίζονται με τα αρχεία καταγραφής ηλεκτρονικών καταστημάτων πρέπει να ληφθούν υπόψη. Οι επιτυχείς συνεδρίες πώλησης είναι πάντα λιγότερες από τον συνολικό αριθμό των περιόδων σύνδεσης. Επιπλέον, εκτός από τους πραγματικούς επισκέπτες, έχουμε συνεδρίες από ηλεκτρονικούς επισκέπτες που δημιουργούνται από προγράμματα ανίχνευσης και συλλογής περιεχομένου (crawlers) και ρομπότ. Αυτές οι συνεδρίες αλλοιώνουν τις μετρήσεις.

Τα διαδικτυακά ρομπότ (γνωστά και ως Web Wanderers, Crawlers ή Spiders) είναι προγράμματα που επισκέπτονται αυτόματα σελίδες στον Ιστό. Οι μηχανές αναζήτησης, οι πράκτορες και οι ερευνητικές εφαρμογές τις χρησιμοποιούν για την ευρετηρίαση του περιεχομένου ιστού, οι spammers τις χρησιμοποιούν για τη σάρωση διευθύνσεων ηλεκτρονικού ταχυδρομείου και έχουν πολλές χρήσεις [5]. Τα ρομπότ μπορούν να εντοπιστούν μέσω της συμπεριφοράς τους [6], αλλά μπορεί να μην είναι πάντα εφικτό να τα ανιχνεύσουμε. Μια υπόθεση είναι ότι τα ρομπότ δεν μπαίνουν ποτέ στο τμήμα πληρωμών του ηλεκτρονικού καταστήματος. Βέβαια, ένα ρομπότ θα μπορούσε να χρησιμοποιηθεί για την αγορά προϊόντων.

Ο ανιχνευτής αρχείων καταγραφής που έχουμε αναπτύξει περιέχει εργαλειοθήκη με εξειδικευμένα εργαλεία που είναι απαραίτητα για τη μέτρηση της απόδοσης των ηλεκτρονικών καταστημάτων καθώς και των προτύπων συμπεριφοράς των πελατών. Οι τυπικοί αναλυτές αρχείων καταγραφής γενικής χρήσης συνήθως επεξεργάζονται αποκλειστικά αρχεία καταγραφής, αξιολογούν τα αποτελέσματα πρόσβασης, υπολογίζουν το εύρος ζώνης και αναφέρουν τις σελίδες επισκέψεων σε ωριαία και καθημερινή βάση, καθώς και τις χώρες επισκεπτών και τα στατιστικά στοιχεία των browsers [7] [8]. Αυτές οι πληροφορίες είναι πολύ χρήσιμες για ένα σύστημα διαχείρισης περιεχομένου, μια πύλη ή ακόμα και ένα στατικό διαχειριστή ιστοτόπου, επειδή οι σελίδες που επισκέφτηκαν και οι διάρκειες επίσκεψης είναι αρκετές για να μετρήσουν την επιτυχία του ιστοτόπου. Ένας διαχειριστής ιστοτόπου ηλεκτρονικού εμπορίου, από την άλλη πλευρά, χρειάζεται πιο συγκεκριμένες πληροφορίες σχετικά με τις εκτελούμενες ενέργειες και συναλλαγές, οι οποίες πρέπει να συνδυαστούν με τα δεδομένα του αρχείου καταγραφής. Τα ειδικά στοιχεία του ηλεκτρονικού καταστήματος σχετικά με τα προϊόντα, τις κατηγορίες προϊόντων, τις παραγγελίες και τους πελάτες χρησιμοποιούνται στην εργαλειοθήκη μας για να αποκτήσουν ακριβέστερες πληροφορίες σχετικά με τα γεγονότα πρόσβασης. Με αυτόν τον τρόπο, ο τελικός χρήστης ασχολείται με πιο εξειδικευμένα στοιχεία για τη συγκεκριμένη επιχείρηση, καθώς χρησιμοποιούνται οικείοι όροι και στοιχεία. Αυτό καθιστά την εφαρμογή πιο εύκολη στη χρήση για τον διαχειριστή.

Η αρχική προσέγγιση, όπως φαίνεται στο Σχήμα 1, έχει κατασκευαστεί ως μια αυτόνομη εφαρμογή με ένα απλό, εύχρηστο και διαισθητικό γραφικό περιβάλλον χρήστη, διατηρεί τη δική της βάση δεδομένων και περιλαμβάνει επιλογές που επιτρέπουν στον χρήστη να προσαρμόζει εύκολα και να φορτώνει δεδομένα από αμφότερα τα αρχεία καταγραφής συμβάντων πρόσβασης (access logfile) (Σύνδεση ❶), αλλά και τη βάση δεδομένων του ηλεκτρονικού καταστήματος (Σύνδεση ❷). Αυτό το μοντέλο χρησιμοποιήθηκε σαν θεμέλιο επί του οποίου βασίζονται οι διάφορες επεκτάσεις. Αυτή η εφαρμογή μπορεί να εκτελεστεί οπουδήποτε, όχι απαραίτητα στο μηχάνημα όπου βρίσκεται ο διακομιστής ιστού. Μπορεί να φιλοξενήσει πολλαπλά ηλεκτρονικά καταστήματα, που εκτελούνται σε πολλές αρχιτεκτονικές διακομιστών Διαδικτύου και λειτουργεί σαν βάση για τις μεταγενέστερες προσεγγίσεις που έχουν υλοποιηθεί. Πρόκειται για μια τυπική ETL[2] εφαρμογή που βασίζεται σε μενού, η οποία περιλαμβάνει ενσωματωμένα στοιχεία και μηχανισμούς απεικόνισης που ενσωματώνουν στοιχεία και πληροφορίες του ηλεκτρονικού καταστήματος στον αναλυτή.

Η δεύτερη προσέγγιση ενσωματώνει και εξωτερικές μετρήσεις, που συλλέγονται από έναν πάροχο Analytics με tagging, όπως το Google Analytics. Το Java API επιτρέπει πρόσβαση σε εγγεγραμμένους χρήστες. Αυτή η προσθήκη μετατρέπει ολόκληρο το σύστημα σε υβριδικό. Οι υβριδικές εφαρμογές Analytics ξεπερνούν όλους τους περιορισμούς των αναλυτών αρχείων καταγραφής και των συστημάτων tagging.

Αυτές οι δύο προσεγγίσεις διευρύνθηκαν και σχεδιάστηκαν και κατασκευάστηκαν αναλυτές αρχείων καταγραφής με πρόσθετες δυνατότητες και περιγράφονται σε αυτό το έγγραφο.

[2] ETL – Extract Transform Load

2. FOREWORD

This section briefly defines basic testing techniques and adapts the key application performance metrics from Stackify to web applications and explains how the application Performance index is calculated.

Today the Internet is used as an important corporate platform, as a medium for advertisement, a gateway for promoting sales, as a tool for offering services and as a communication platform.

Web Analytics Systems enable the calculation of the cost-effectiveness for the operation of online systems. The operation of web applications and actions is measurable. Their successful implementation can be programmed. Evaluation of measurable performance and behaviors help achieve goals. An important first step is to search and define the appropriate Key Performance Indicators for every business model.

Web sites constantly evolve. While Web 1.0, the primary Internet format consists of simple static websites with little interactivity, communication via e-mail and is measured with simple techniques like visit counters, Web 2.0 is a collaborative Internet. It makes use of asynchronous updates and communications based on AJAX. The applications communicate extensively with Database Management Systems. Since these are more complex systems the measurements are more demanding. Social Networks play also a vital role in Web 3.0, also called Semantic Web or Social Web. The profile is composed by poling many media: Blogs, Twitter, Facebook etc. Web analytics applications must adapt to the evolution of the Web and provide a dynamic base platform that can provide metrics for all novelties and new configurations.

The Web is the most interactive and measurable medium ever created, but realizing its full value requires significant effort and clear strategy. Building a website is the easy part, measuring your success is a lot more difficult, and taking action based on what you measure is even harder [9].

Fortunately, the websites conform to certain standards. E-Marketing, E-Business and E-Commerce sites usually operate under a restricted number of web servers, they all use database management systems and their substructure is often based on freeware or open source code. This makes customizing general solutions easier feasible.

Testing performance does not only involve measuring speed. Important factors are capacity, stability, accuracy, stamina and economy. Compliance to standards and norms, security and usability must be taken also into consideration, but still speed is the main key factor.

Meeting performance standards and passing speed tests guarantees better customer and user acceptance for the developers. Performance requirements often are among contractual obligations for the developer. Customers expect better benchmark results when they upgrade their system and performance measurements and standards can be used as tools for the comparison with previous versions. Better performance allows building supremacy over competition, because speed is an asset for any software application.

In order to get meaningful measurements, typical often occurring transactions must be selected. Basic workflows for example, of common day to day interaction: when a shopper searches, selects, adds a product to cart, logs in and pays. User types can be explored according to their visit profiles. Certain actions can be isolated, for example, buyer check out, searches, accounting procedures etc. An excellent tool that can be applied in order to measure overall performance of an application is JMeter. JMeter[3] is a mature open source Java Apache project that allows defining virtual users and assigning tasks to them in order to measure performance and find the limits of any system. In case of web site evaluation, the tester can define the virtual users' typical transactions and orchestrate their concurrent operation. If a transaction sequence is typical or not, can be deducted by analyzing the log file of the web server. JMeter allows defining any number of users that perform these typical transactions against the tested application. This way the performance of a system can be analyzed by changing the load by extending the concurrent virtual user numbers and types. The application emulates physical people and automatically generates controlled requests acting as groups of users. If the usage scenarios are focused on realistic types of users, the response measurements will be also realistic. We can generate scripts for example where 100 users execute simultaneously requests, search for products, add products to the cart and even log into the system and checkout. This application gives independence to the tester by emulating huge groups of visitors that perform requests simultaneously and concurrently. **Load testing** is the process of putting demand on a system and measuring its response; that is, determining how much volume the system can handle. **Stress testing** is the process of subjecting the system to unusually high loads far beyond its normal usage pattern to determine its responsiveness. These are different from performance testing whose sole purpose is to determine the response and effectiveness of a system; that is, how fast is the system. Since load ultimately affects how a system responds, performance testing is almost always done in conjunction with stress testing. [10].

During JMeter measurements, the target system can often be so heavily challenged, that normal operation for customers

[3] http://jmeter.apache.org/

is not possible. Time measurements are described in modeled scripts. A test usually includes one or more such scripts, with multiple configuration data, that are repeated for one or more Virtual Users. Each test can be scheduled to run at specific times. Key Performance Indicators are: 1. Response Time - The time it takes to send a request, the time it takes to process and send a response, 2. Latency - The time it takes for the request to be sent and processed by the server. It stops when the response begins. 3. Render Time - The time it takes for the results to appear. 4. Maximum simultaneous virtual user number and 5. Throughput - responses per second or minute.

Performance test types are usually: load test, stress test and endurance test. Load test, also called performance test is performed with a fixed number of concurrent virtual users, has a fixed duration or a specific number of transactions and shows the reliability of the system and the volume of load it can deal with. In order to stress-test or torture-test the system, extreme load of virtual users is applied until the system stops operating. Throughput is measured during continuous iterations of transactions and is used for detecting possible bottlenecks and benchmarking. Endurance is tested by applying load for extended periods of time with fixed virtual user numbers and by letting the system serve random requests in order to avoid artificial speed ups due to caching that make the system appear faster than it really is.

Apache JMeter is a very useful tool in order to perform measurements and tests that allow assessments of performance and stress tests. For e-commerce sites though, these tests should take place when the system is not in operation and accessible by real visitors and prospective customers, because the tests would impact overall performance and thus risk losing customers. The application is used before the inauguration of the application in order to test overall performance and make sure that the configuration of the application is powerful enough to meet the needs.

In this work, the proposed metrics do not impact the overall performance significantly and are designed to run constantly.

2.1. Key Performance Metrics

Mainstream commercial computer software applications are increasingly moving from the desktop to the browser and the World Wide Web. The Web is a more complex and demanding foundation that forces different development approaches and tools than corporate Local Area Networks.

During the recent past few years the increasing technological advances and the ever-expanding business competition have increased the requirement complexity of software applications and have generated a demand for faster development cycles, verification, better performance and lower response times as well.

Complex systems have also higher and increased needs for perpetual evaluation of their operational characteristics, scrutinizing capabilities of application performance measurements under various loads and conditions, so that the response time is always below some reasonable threshold boundaries in order to consider every system of applications usable.

Knowing the limits of any software application system helps prevent slow responses, down times and allows defining precise operational specifications, providing awareness. Applications often are designed so that they include their own internal measuring and evaluation modules, integrated in the business logic layer. The advantage of this approach is that certain predefined measurements and their functionality are placed inherently by installing the application and can be run directly through the standard interface of the application. The disadvantage lies simply in the fact that these modules add to the complexity of the application and since the measurements needed may vary in a very dynamic environment, covering the new requirements would force modifications to the entire application. To prevent interfering with the application, external analytics applications are of advantage.

While desktop compiled applications executables run either straight by the operating system or through a virtual machine, like the Java Virtual Machine for example, and interpreted applications by the interpreter, which also run straight by the operating system, web applications run by a web server, a daemon process. This adds one more level of software to the stack.

The web server software interfaces with the Web and accepts and supervises HTTP requests. It produces and delivers static or dynamic HTML pages to user agents like web browsers and receives and manipulates form input data and supports file uploads. In other words, web applications run on web servers and their desktop is the browser, both being platform and location independent.

The huge dissemination of the World Wide Web renders any possible browser user to become a potential customer for any e-Commerce application. While clients of corporate applications so far could only be users on local or remote nodes with access permissions to the corporate network, a web-based application allows literally anybody to use it. The potential customer is always just a click away from the e-store. Security mechanisms and accessibility can be arranged and controlled at the server-side affecting all visitors immediately and deploying new versions of software is a one-step procedure.

The importance of operating well-functioning and responsive web applications is especially crucial when they are used as electronic commerce platforms. Delays and erratic behavior may scare the visitor away to the competition.

The purpose of this work is to specify and implement a customizable toolkit that will enable the operators of any e-commerce website to evaluate the performance and scrutinize the behavior of the visitors in a timely and simple manner.

Metrics provide the ability to study and understand previous operational behavior of the application, document specific events and shed light into details and exception situations. Finally, they allow focusing on improvements and measure the impact of their achievement during their implementation.

According to Matt Watson from Stackify[4], key performance metrics that need to be presented for any application are:

1. User Satisfaction – Apdex scores
2. Average Response Time scores
3. Error Rates
4. Application Instances Count
5. Request Rate

[4] https://stackify.com/application-performance-metrics/ (Matt Watson July 3, 2017)

6. Application and Server CPU
7. Application Availability
8. Garbage Collection

The goal of the Analyzer we have designed is to provide key performance metrics for web applications, which leaves out points 6 and 8, since we stand at a different level. CPU times and garbage collection are reflected in the overall response times of the web server serving the web application.

2.2. Apdex index

The **Apdex** user satisfaction index is based on a fixed presumed goal for what may be considered as the acceptable time needed to perform a transaction. This assumed value is used as the criterion for evaluating each transaction timed as satisfactory, tolerable or unsatisfactory. The Apdex (**A**pplication **P**erformance In**dex**) is calculated according to the following formula:

$$\boxed{Apdex_t = \frac{SatisfiedCount + \frac{ToleratingCount}{2}}{TotalSamples}}$$

Apdex score calculation formula

This satisfaction index is an easily obtainable, but useful measurement, with every result being a decimal number within the range of 0 and 1. Certainly, like any other index, it only tells part of the story, especially since the threshold, being a static number is picked arbitrarily, based upon experience. As transaction time we can define the time from the first visit until the next request, or the time from one request to another, or as the time it takes to upload the entire page to the visitor.

Averages are standard and straight forward obtainable metrics that describe performance in general quite well. Average response times, load and usage factors are used in every analytics application. Still, averages can be misleading, especially when they lack weighing factors. Important sessions may be slow, while less important ones may be fast and lead to “acceptable” average values. All six Apdex sets of the graph below (Figure 2) have the same average value, which happens to be 0.5. Yet the results are very different and certainly many of these users will not be satisfied by them.

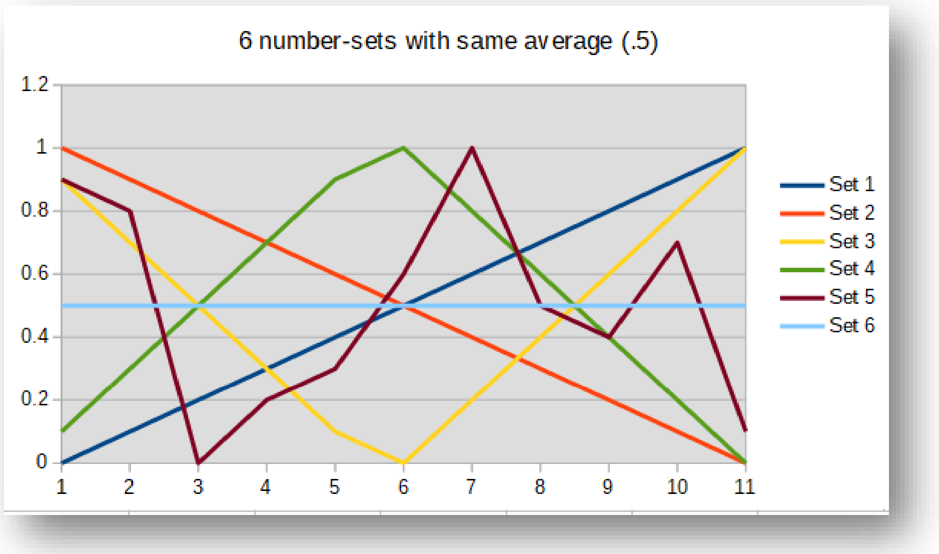


FIGURE 2: SIX NUMBER PAIR SETS WITH SAME AVERAGE

Error rates should be kept at a minimum. Successful operation requires error free minimum. The logfile analyzer can easily locate and reveal all http error codes, measure their frequency and their context since they are part of the information stored in the log file. The access log file is usually configured so that every line of the log file may include the status code of the response. Additional errors and warnings are in the additional log files of the web server.

Application Instances depend on the configuration of the web applications. Often a web application or e-commerce site is bundled with its own webserver, having its own settings, database and configuration. The access log file is also separate. To host more distinct e-commerce applications on the same server they need to be configured so that their ports do not collide, which applies to HTTP and HTTPS ports. Since the Log File Analyzer is designed so that it can accommodate multiple e-commerce sites, the number of instances is irrelevant for its operations. Certainly, every instance has impact on the performance of every other instance on the same server. A different issue arises when the e-commerce application is hosted on a cloud.

Request and **Traffic Rates** are two of the most primary reasons for deployment of Analytics Applications. Specialized techniques for extracting this information from log files and tagging services are described and implemented here. Concurrent users and load are visualized in various places in the Analyzer. Also available in real time.

Application and **Server CPU** times are not considered to be target components for measuring by any LFA. They are crucial metrics though, provided by specialized low-level applications that measure often anything available by the hardware, like memory, loads and even operational temperatures of the hardware of any server and it is the System Administrations' responsibility to provide physical server monitoring. Measuring and checking limitations of the hardware at the operating system level allows preventing downtimes. Web application metrics can also spot and report low responses on a different level. It is possible to provide insight by comparing the metrics of the web applications to the metrics of the server.

Application Availability becomes visible through the logfile contents. Only when considerable time gaps in the logfile exist it is possible that they are caused by down times. Another reason for these gaps may be extended periods of time without visitor requests. A remedy to this, is a throughput measuring application that generates low quantities of artificial load to the web application every time there is no traffic to keep the speed gauge of this service alive. This way there is no need to ping the server.

Finally, **Garbage Collection**, pretty much like CPU times is a matter concerning the web server administration and not the Analytics Application.

The following section contains a short description of the motivation that lead to this work, an outline of the development environment and the application extensions designed and developed and finally provides a short introduction into Analytics history and presents the leading analytics companies.

3. INTRODUCTION

This section contains a short description of the motivation that lead to this work, an outline of the development environment and the infrastructure used, as well as the application extensions designed and developed and finally provides a short introduction in Analytics history, pointing out why the subject is important and listing the main companies involved.

3.1. MOTIVATION

New desktop applications, web applications and mobile apps are being developed with increasing frequency, driven by the need to improve existing solutions and to exploit all the latest technological advances. Older applications, if not regularly updated, stop being compliant with latest needs and ever evolving requirements and environmental changes and finally end their lifecycles, remaining unused in software repositories.

The basic motivation behind this thesis was the wish to create a modular environment that allows studying the operational characteristics of web sites and especially sites used for electronic commerce. Ecommerce sites are a special category of commercial applications with web interfaces for the public. The basic difference between an ecommerce site and any other general content web site, lies in the fact that electronic shops tend to behave like huge electronic product catalogs that can support sales. The hierarchical tree-structure that represents the contents of an ecommerce site is usually very shallow in comparison with the site structure of a news agency web site for example. It usually contains just very few levels of categories and subcategories. Even Amazon, which is one of the oldest and largest ecommerce sites in the world and has been the source and the testbed of many innovative technologies that deal with data volume and parallel processing because of its size, despite the huge range and number of products they sell is designed with just very few levels of categories.

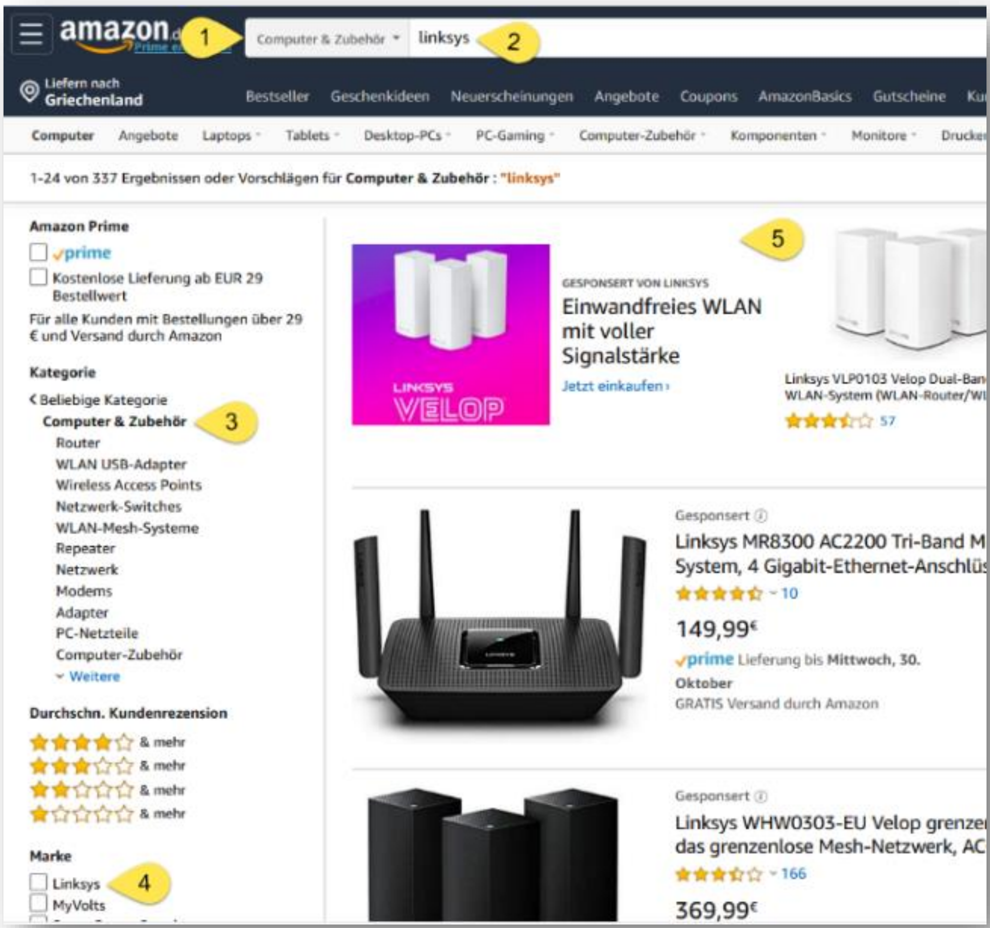

amazon.de
1
Computer & Zubehör
linksys
2
Liefern nach Griechenland
Bestseller
Geschenkideen
Neuerscheinungen
Angebote
Coupons
AmazonBasics
Gutscheine
Computer
Angebote
Laptops
Tablets
Desktop-PCs
PC-Gaming
Computer-Zubehör
Komponenten
Monitore
1-24 von 337 Ergebnissen oder Vorschlägen für Computer & Zubehör : "linksys"
Amazon Prime
prime
Kostenlose Lieferung ab EUR 29 Bestellwert
Für alle Kunden mit Bestellungen über 29 € und Versand durch Amazon
Kategorie
‹ Beliebige Kategorie
Computer & Zubehör
3
Router
WLAN USB-Adapter
Wireless Access Points
Netzwerk-Switches
WLAN-Mesh-Systeme
Repeater
Netzwerk
Modems
Adapter
PC-Netzteile
Computer-Zubehör
Weitere
Durchschn. Kundenrezension
& mehr
Marke
Linksys
4
MyVolts
5
LINKSYS VELOP
GESPONSERT VON LINKSYS
Einwandfreies WLAN mit voller Signalstärke
Jetzt einkaufen ›
Linksys VLP0103 Velop Dual-Ban WLAN-System (WLAN-Router/W
57
Gesponsert
Linksys MR8300 AC2200 Tri-Band M System, 4 Gigabit-Ethernet-Anschlü
10
149,99€
prime Lieferung bis Mittwoch, 30. Oktober
GRATIS Versand durch Amazon
Gesponsert
Linksys WHW0303-EU Velop grenze das grenzenlose Mesh-Netzwerk, AC
166
369,99€

FIGURE 3: AMAZON.DE CATEGORIES

Figure 3 shows the direct route that leads to specific product pages in Amazon.de. In step 1 the visitor can choose among all categories of Amazon through a convenient combo box. Step 2 can restrict the selection to only a specific brand name or a specification. Steps 3 and 4 also assist the potential customer to reach step 5, in which the final product can be added to the shopping cart.

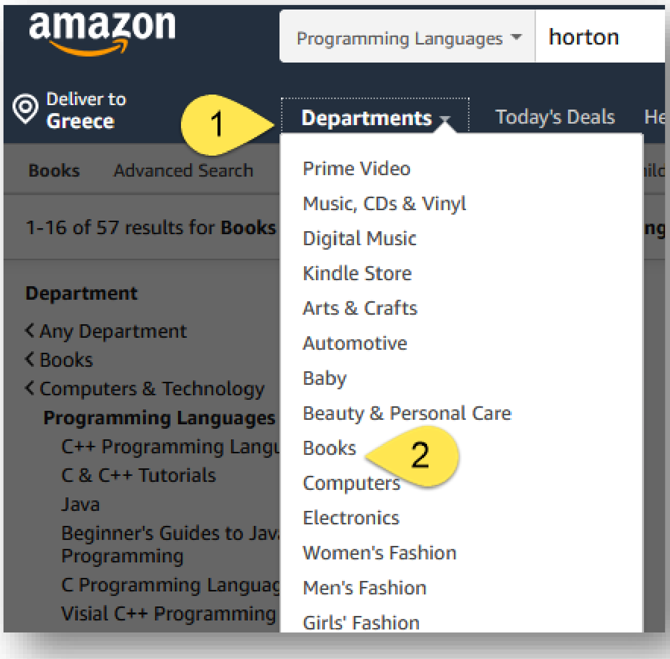


FIGURE 4: AMAZON PRODUCT CATEGORY COMBO BOX

In Figure 4, pointer 1 shows the product category combo box, called by Amazon "Departments", and a portion of the displayed options which are the categories or "Departments" in pointer 2.

The initial inspiration to proceed with this topic originated from Professor Boucouvalas, whom I would like to thank again for his support and for assisting this work. Daniel Menasce's and Virgilio Almeida's book "Scaling for E-Business" [1] triggered the idea. Menasce proposes the generation of a Customer Behavioral Model Graph (CBMG) for e-commerce sites to support analyzing and measuring visitor sessions which shows the character of the visit.

In order to deal with the "shallow tree" design characteristics of e-commerce sites and be able to analyze the important milestones of a single session we follow what we consider as the relevant activity steps. Steps like: "select category", "select product", "add to cart", "checkout", etc. The application allows selecting these relevant activities that should be displayed in the CBMG. Depending on the technology and the framework used for implementing the e-commerce site, activities can be registered in the HTTP GET request text, along with other parameters defining product category, product codes or other codes. Product and product category codes are usually difficult to memorize and can be numeric, so in order be able to make better sense of the GET request and to present results and visualizations to the operator of the system in a user friendly manner, the analyzer needs to gain read permission to the few tables of the e-commerce site database that store product categories, product codes, descriptions and retail price. This step will allow enriching category, product keys with descriptions and additional information that can customize the output of the LFA with the appropriate left-joins.

3.2. Outline

This work revolves around defining the necessary data and designing and implementing a modular log file parsing and analyzing mechanism, that should easily cooperate and adapt to multiple web servers. The analyzer application is written in 100% Java, since Java is a powerful, portable and convenient object-oriented programming language. The Java development environment is available from Oracle under an open source license and is free of charge for general purpose desktop and server use[5]. A huge number of various application programming interfaces (APIs) for almost every need, ranging from computational problems like matrix computations, interfaces, data access and visualization supports versatility and programming efficiency. Java has a native support for the Document Object Model and offers native extended markup language (XML) support.

The design specifications of the Analyzer are described in detail. External and internal libraries, as well as microservices are used and configured in such way that they can support expanding and adapting to various environments, receive and send data, integrated, and acting as one compound but modular system.

The development itself was performed interchangeably on Linux and Windows machines, in order to safely check the portability of all features during the development time and avoid environment-specific assumptions, code and settings.

The form, type and specifications of the data that need and can be collected depend heavily on the architecture and foundations upon which the e-commerce site was built.

MySQL is being currently used as the central data repository of the application. MySQL offers several useful development tools, like MySQL Workbench and a reliable Java Database Connectivity driver. It supports triggers well

[5] https://www.oracle.com/technetwork/java/javase/overview/faqs-jsp-136696.html

and the overall performance on a single machine is adequate. MySQL supports standard SQL and transactions with the InnoDB engine. On top of that, the Community edition is free and available to everybody.

For higher availability and throughput, the InnoDB engine can transparently to the application be substituted by the multi-node MySQL Cluster. The MySQL cluster adds redundant components on separate nodes. A simple setup could run on three nodes: one MySQL server and cluster manager and two cluster nodes. The number of cluster nodes can easily be extended if necessary, by adding nodes to the local area network. This can be done with simple setup steps in the */etc* area. There is no need to modify the application software when this upgrade takes place, since the MySQL Cluster port is fully transparent to the application.

The open-source commercial ecommerce software application of Zucchetti Group Konakart[6] has been used as a main paradigm of an e-commerce site. Konakart Community edition is free. Zucchetti sell mainly the Enterprise edition that offers several additional capabilities, like multi-store and multi-vendor e-shops. It has better search engines, provides easier ERP integration, has more sophisticated billing. These functionalities do not exist in the community edition. Still the overall functionality of the community edition is enough for its role as a sample application used for our Analyzer System needs. Both the front-end and the backend of Konakart are Java applications. The former is based on the Apache Struts2[7] MVC framework and the latter on Google Web Toolkit[8]. Konakart runs on the Apache Tomcat web server and servlet engine and produces a typical detailed access log file that we can extracted, loaded and processed.

The basic Analyzer application, briefly described in Figure 1, acts as the foundation for the more advanced versions presented in this study. The LFA is composed of several components that will be presented one by one. The APIs used and the visualization tools that helped integrate result prints, reports, graphs and results will be presented.

We will also describe the functionality requirements and present the necessities that lead to adding features and finally generating a version that is collecting information through microservices and streaming, as well as cooperating with social media applications and operating in near real time [11].

[6] https://www.konakart.com/

[7] https://struts.apache.org/

[8] http://www.gwtproject.org/

A summary of the Big Data technologies available will be summarized, since they alleviate any storage restrictions that may appear whenever data collected from the LFA are growing beyond the capabilities of single servers and database clusters. File systems like Apache Hadoop and its ecosystem consisting of clusters of nodes that process parallelly and MapReduce or Apache Spark in conjunction with NoSQL databases can store and process petabytes of data and allow timely retrieval, generation of information as well as visualizations with practically no limits.

The final extension proposed in this work is for making the LFA capable of processing also data generated by sensors like Near Field Communications and Blue Tooth Low Emission as used for detecting NFC Tags and iBeakons. These devices may be spread in physical stores and retail. If the management convinces the visitor to run the appropriate application on her mobile device that allows online access to the details of the products from the database of the e-shop, the streams of data generated could easily be merged with the keyclicks and the requests for the e-shop. The customer will experience better service and be able to read and study any details and all characteristics of the items of interest on the fly, while shopping in the physical store aisles.

3.3. Analytics Status Quo

Analytics systems are gaining in importance. Often, they are being considered as integral components of e-commerce applications (Figure 5). The requirements complexity and the mix of applications, services and microservices that are being deployed in a web application lead to the need of a stable and precise external monitoring mechanism. Web Analytics systems can play this role. All interaction and operating information are collected from their sources, mainly log files or artificially generated secondary information, are registered in some form of repository and become available for data mining or direct processing, visualization or messaging.

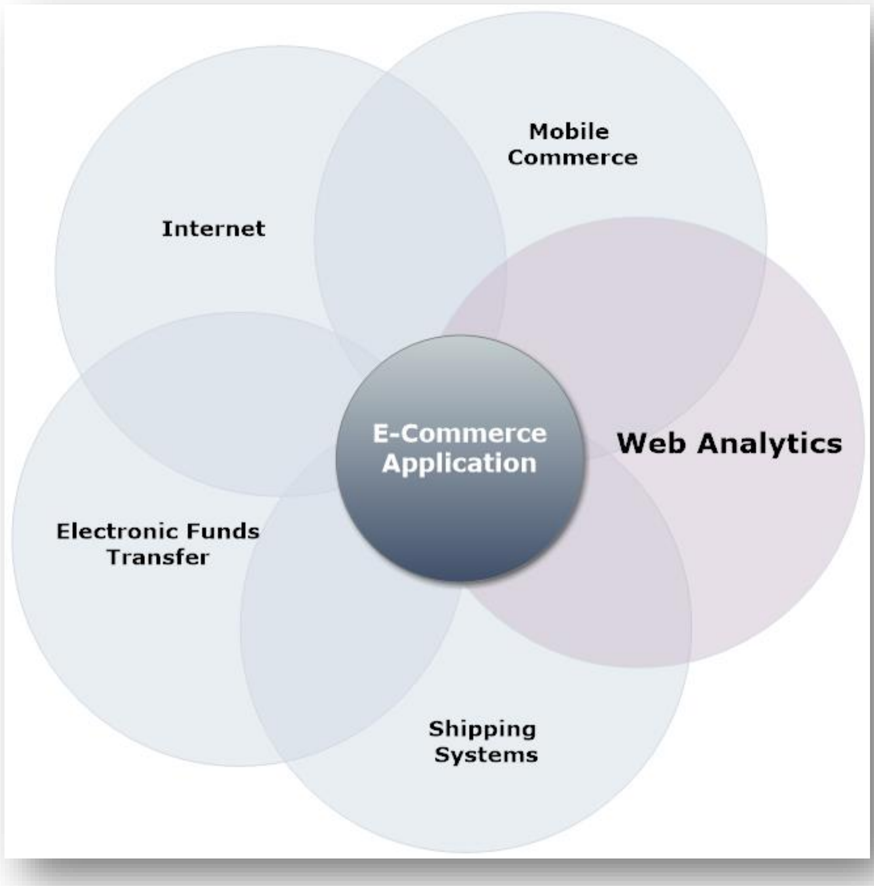


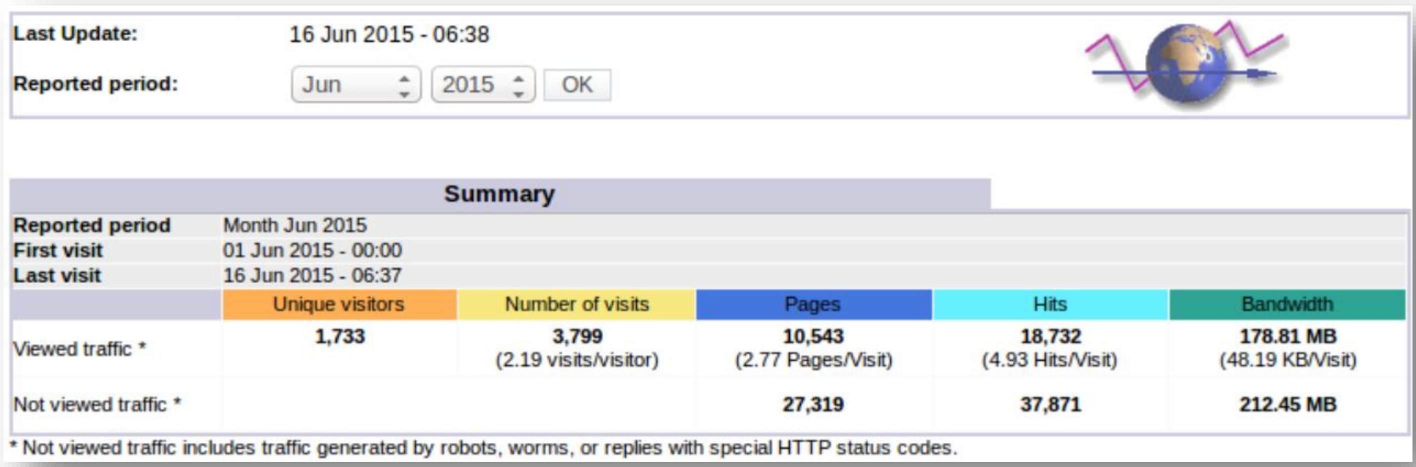
FIGURE 5: WEB ANALYTICS CONSIDERED INTEGRAL COMPONENT OF E-COMMERCE

Analytics has been is a growing and vivid market. Companies, technologies and products change hands and large corporations buy out small companies. The basic players in the Analytics market today are:

- Google Analytics (Urchin)
- Microsoft Azure
- Heap Analytics
- Yahoo Web Analytics (indexTools)
- Adobe Web Analytics (Omniture)
- IBM Unica NetInsight
- Hitmatic
- Open Web Analytics (OWA)

- Parse.ly
- Woopra
- MixPanel
- Clicky
- The Webalizer

One of the first applications, released in year 2000, for web analytics reporting, is AWStats[9]. AWStats is written in Perl, licensed under the GNU General Public License and runs on the web server of the web application. AWStats generates web pages with access and visit information (Figure 6). It works as a Common Gateway Interface and has a very structured month, day or hour-based reporting format.



Summary					
Reported period	Month Jun 2015				
First visit	01 Jun 2015 - 00:00				
Last visit	16 Jun 2015 - 06:37				
	Unique visitors	Number of visits	Pages	Hits	Bandwidth
Viewed traffic *	1,733	3,799 (2.19 visits/visitor)	10,543 (2.77 Pages/Visit)	18,732 (4.93 Hits/Visit)	178.81 MB (48.19 KB/Visit)
Not viewed traffic *			27,319	37,871	212.45 MB

* Not viewed traffic includes traffic generated by robots, worms, or replies with special HTTP status codes.

FIGURE 6: AWSTATS SUMMARY

[9] https://awstats.sourceforge.io/

3.4. IMPORTANCE OF GAINING INSITE

Analytics systems provide information about a wide spectrum of topics dealing with performance, metrics, times, volume of traffic, profitability. They should have a modular architecture and should be expandable and capable of dealing with the inevitable evolution of the market and the ever-changing needs of the sites they support.

Analytics applications should be able to deal with:

- Order Values/Numbers
- Visits
- Time spent per product or service
- Accesses per product or service
- Orders per Product or service
- Bots visited
- Visitors
- Uncompleted ordering sessions
- Profitable customer groups
- Profitable products or services
- Overall profits
- Promotion impact
- Analysis of bots and their search engine behavior concerning e-shops.
- Recognition of anonymous bots and spiders through their access patterns.
- Customer rating and evaluation application based on non-purchase behavior.
- Agent implementation in order to automatically promote the rank of less sought for products.

Important literature on analytics and the supporting infrastructure technologies involved in order to generate applications is included in the following chapter.

4. LITERATURE REVIEW

Important literature on analytics and the supporting infrastructure technologies involved in order to generate applications.

Web Analytics is a relatively young and multidiscipline topic. Solutions can be achieved through complex systems, involving combinations and cooperation of software applications at different levels. Systems that combine data and content from various sources to a single interface are informally called mashup applications. The number of different platforms and application choices becomes even larger in the big data arena, where the lists of software options are huge. This fact makes also the specialized literature very diverse. Many books that describe specialized techniques and technologies become quickly obsolete, since they describe software application versions that are substituted by new ones, since the evolution is rapid. This review takes important books into consideration, starting with the beginning generations of analytics systems, mainly developed for local area networks, called Extract-Transform-Load (ETL) Systems and move on to contemporary systems.

Ralph Kimball has done serious work on ETL applications and has written three books over the years, together with various coauthors, that define, describe and analyze ETL systems, data warehouses and Webhouses. In "The Data Webhouse Toolkit: Building the Web-Enabled Data Warehouse" [12] Kimball and Merz define what data Webhouses are and how they work and explain how to extract web data and insert it into them. The book proposes ways to track user actions. Describes how to use clickstreams to make decisions. Explains how to design a website to support warehousing. Focuses on building clickstream data marts. Defines and provides examples of value chains. Shows how to implement a clickstream postprocessor. Deals with issues of internationalizing the Webhouse and security and scaling issues. "The data warehouse ETL toolkit: practical techniques for extracting, cleaning, conforming, and delivering data" Kimball and Caserta. [13] describe the architecture of ETL systems that prepare data for feeding data to warehouses. Show ways to organize, fetch and prepare data for storage in data warehouses and offer a roadmap for designing, building and running the necessary ETL software system. The book is focused on extracting, cleaning, conforming and delivering. Extracting includes methods for choosing data sources and preparing the transformations. Cleaning describes auditing techniques for the data. Conforming sets definitions for conformed dimensions and facts and defines the publication strategy. Preparing leads to visible results. The book introduces Dimensional Modeling and Real-Time ETL and reviews different architecture options for their implementation. In 2013, Ralph Kimball with Margy Ross publish "The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling", [14] and introduce techniques for defining dimensional modeling, which appears in the first edition of The Data Warehouse Toolkit series. Dimensional modeling becomes a widely accepted approach for presenting information in data warehouses and business intelligence (DW/BI) systems. This book presents a comprehensive library of dimensional modeling techniques. It offers guidelines for designing dimensional models based on case studies. It covers: Practical design techniques for dimension and fact tables. Includes case studies for retail sales, electronic commerce, customer relationship management, procurement, inventory, order management, accounting, human resources and other branches. Explains how to avoid dimensional modeling mistakes. Defines best practices for Big Data analytics and offers guidelines for collaborative design sessions with business owners. Includes an overview of a DW/BI project lifecycle methodology and a comprehensive review of extract, transform, and load (ETL) systems and their design considerations.

"Business Analytics: A Practitioner's Guide" [15], by Rahul Saxena and Anand Srinivasan focuses on the lifecycle and all stages and steps involved for the corporate Analytics procedures based on information technology.

Many books about Google Analytics have been published by several writers. Among the first ones is "Google Analytics" by Mary Tyler and Jerri Ledford [16]. The book describes the way Google Analytics was designed and how to set it up as of year 2006. One year later the advances of the GA interface and capabilities rendered the book obsolete and forced the writers to revise it. The new book is titled "Google Analytics 2.0" [17]. A final version of this series is titled "Google Analytics, 3rd Edition" [18]. All three books serve the same goal, which is explaining the concepts of GA and teaching how to setup and how to make best use of the various versions of this application.

Brian Clifton, published in 2008 [19], 2010 [20] and 2012 three editions of his excellent book, that also deals with Google Analytics, titled "Advanced Web Metrics with Google Analytics" [21]. Although the book focuses on the Google Analytics tagging system application, its features and capabilities, it starts with an accurate general introduction to web measurements and explains the goals of Analytics for the business. Clarifies what can be measured in Analytics in general and in Google Analytics, discusses the methodologies available and compares them in terms of accuracy. It lists features, but also limitations of Google Analytics, provides a detailed description of the user interface of the application as well as the main reports that it generates. Shows how to activate an account and explains how to define Key Performance Indicators and how to identify poor performing pages and even how to integrate third party applications through the Google Analytics export API. In 2015 Brian Clifton publishes "Successful Analytics: Gain Business Insights by Managing Google Analytics" [22]. This is no more a tool specific book like the previous ones, describing how details of the Google Analytics application work, but focuses on how to set the necessary key performance indicators in order to achieve user acceptance and successful operation of a web site through Analytics.

Menascé and Almeida have published interesting and important books and papers describing the four-level model for Analytics and the application of the Customer Behavior Model Graph to visualize visitors' steps. The book "Scaling for e-business: technologies, models, performance, and capacity planning" by Daniel A. Menascé and Virgilio A. F. Almeida [1] is important since it presents a quantitative approach to understanding and analyzing e-business scalability based on a four-level reference model. The four-level model is composed of a business model, a functional model, a customer behavior model, and an IT resource model. This framework is used throughout the book to explain how e-business technologies work and how they impact performance, to characterize and forecast the workload of e-business sites, and to plan their capacity with the use of performance models. New performance metrics for e-business are presented in the book. Models at the various levels of the reference-model are generalized in order to represent and support understanding of problems in e-business. These models include customer behavior models (e.g., Customer Behavior Model Graphs and Customer Visit Models), Client/Server Interaction Diagrams, and analytic (e.g., state transition diagrams and queuing networks) and simulation performance models. The combined use of these models provides a framework for assessing and evaluating the scalability of e-business sites. Many examples derived from real e-business situations are used to illustrate the concepts presented in the book.

In 2004, Menasce and Almeida publish together with Lawrence Dowdy "Performance by Design: Computer Capacity Planning by Example" [23]. This book describes how to map real-life systems (e.g., databases, data centers, e-commerce applications) into analytic performance models. The authors elaborate upon these models and use them to help the reader thoroughly analyze and better understand potential performance issues.

An interesting book from Rob Sanders defines rules for Google Analytics. "42 Rules for Applying Google Analytics. A Practical Guide for Understanding Web Traffic, Visitors and Analytics" [24].

Another important book covering similar topics, with more emphasis on mathematical models is "Modeling the Internet and the Web: Probabilistic Methods and Algorithms" [25], also a relatively early book, that discusses mathematical, graphical and probabilistic models for crawling techniques and behavior measurements.

Messaging solutions that allow integration of systems are Data streaming from various points allows sending data to the Analyzer using queues. "Mastering RabbitMQ" [26], a book from 2015 by Emrah Ayanoglu et. al. is a book that covers many issues of setting up, programming, and operating queueing software streaming based on message broker systems. RabbitMQ[10] is a wrapper based on AMQP which allows solving scalability and data serialization and sending over networks. The book describes how brokers are designed and work. Security, reliability, interoperability and

[10] https://www.rabbitmq.com/

compliance to standards as well as the architecture of the underlying AMQP[11] application layer protocol for message-oriented middleware. Installation of the Erlang programming environment and RabbitMQ and setup instructions are presented for GNU-Linux, Windows and OS X. The book explains how to use Federation in RabbitMQ, in order to handle issues like having to transfer lots of messages among multiple RabbitMQ servers. The book introduces the Erlang[12] general purpose, concurrent, garbage-collected programming language and runtime system. A special section describes all RabbitMQ managing options, command and web interfaces and configuration files. The book includes Java, Ruby and Python code examples and describes how RabbitMQ can be used with small, medium and big data. Beside Ayanoglu, also Stephen Haunts introduces RabbitMQ in "Message Queuing with RabbitMQ Succinctly" [27]. The code samples are written for Microsoft Visual Studio but are easily convertible to Java. "RabbitMQ in Depth" [28] by Gavin M. Roy explains details of the RabbitMQ software system. Another book that introduces queuing models of enterprise applications and analyzes performance issues is Leonid Grinshpan's "Solving Enterprise Applications Performance Puzzles: Queuing Models to the Rescue" [29]. This publication deals with complicated connections and correlations using parameters involving workloads, hardware and software.

"Introducing Erlang: Getting Started in Functional Programming" [30] written by Simon St. Laurent in 2017 provides a good introduction to Erlangs' functional programming style.

An important book for streaming in Big Data is "Real-Time Analytics: Techniques to Analyze and Visualize Streaming Data" by Byron Ellis [31]. Byron Ellis describes the architecture concepts of streaming analytics. The book describes how streaming allows data to become available to remote applications just a few seconds or minutes after it has been generated and shows tool combinations that will help achieving this goal. There is a broad introduction on how data collection can be optimized and the data formats that are in common use. An introduction into Hadoop and Map-Reduce processing informs the reader about big data in the begin of the first chapters. Beside Java, Scala and Clojure are presented as options for the development, as well as JavaScript. An entire chapter is dedicated to describing Apache ZooKeeper[13], because of its importance for maintaining horizontal scalability. Dataflow management is handled with Kafka[14] and Flume[15]. Both applications are described. Another chapter is dedicated to processing streaming data with

[11] http://www.amqp.org/

[12] https://www.erlang.org/

[13] https://zookeeper.apache.org/

[14] https://kafka.apache.org/

[15] https://flume.apache.org/

the Apache projects Storm[16] and Samza[17]. The steps of storing the streaming data to MongoDB are described and finally the processing in order to provide streaming metrics. The visualizations are done with HTML5, JavaScript and D3[18] or NVD3[19].

"Social Media Data Mining and Analytics" by Gabor Szabo, Gungor Polatkan, Oscar Boykin and Antonios Chalkiopoulos [32] is analyzing data extracted from social media applications. The authors succeed in answering the "who, how, when and what" questions. They approach many "Social media" systems, download data and process it with various languages. In order to answer the 'who' question Wikipedia and Tweeter are addressed, and the data extracted is processed with Python and R. They compare the degree of diversity between the users. The 'how' question checks the connections between users of a chat application and the generation of a graph of social connections is presented with the use of Python and matplotlib, along with the metrics. The distribution of degrees (numbers of neighbors) that the users have. The 'when' question is answer with the temporal characteristics of the users' actions. Timestamps of activities are collected, and Poisson process model applied. The autoregressive integrated moving average (ARIMA) gives more precise predictions since it compares previous values. The 'what' answer is approached as textual content analysis. The goal is to be able to analyze text and unstructured data. The authors download an XML file from a Stack Exchange sub site and analyze 46,000 questions and 88,000 answers. Term occurrences are calculated and graphed, and dissimilarities are computed with a distance matrix of terms. Word clouds show the identified topics. Popularity of topics and clustering of individual documents are measured. After the four questions have been asked and answered, the book proceeds to apply the classic MapReduce algorithm in order be able to solve counting words and processing huge amounts of data with big data techniques on Hadoop clusters instead of single computer file systems. Finally, the book describes a learning methodology that can be used in order to make predictions about what users like and feed a recommender system with options.

Beside the standard manuals that are available to the programmer and can be downloaded from the respective websites of the used software subsystems, like MySQL server, MySQL Workbench, MySQL Connector/J, Java SDK, Apache Commons, Jaspersoft Studio, Maven, Apache2 and Apache Tomcat Webserver, the following books offer valuable support: for the Java programming language Ivor Horton's "Beginning Java – Java 7 Edition" [33], Tanuj Khare's "Apache Tomcat 7 Essentials" [34] and Vukotic – Goodwill "Apache Tomcat 7" [35] describe many administration details and advanced features of the Apache Tomcat web server. "MySQL 8 Administrator's Guide" [36] explains

[16] https://storm.apache.org/

[17] http://samza.apache.org/

[18] https://d3js.org/

[19] http://nvd3.org/

MySQL administration concepts and configuration. “Murach’s MySQL 2nd Edition” [37] is a helpful book for SQL details and query support. “Apache Maven Cookbook” [38] explains how to setup complex Java projects involving multiple libraries easily using Maven. “Ubuntu Unleashed 2019 Edition: Covering 18.04, 18.10, 19.04, 13th Ed” [39] and “Mastering Ubuntu Server Second Edition” [40] by Jay LaCroix are a useful publication describing and supporting advanced settings of the Ubuntu operating system.

“Beginning NFC Near Field Communication with Arduino, Android and PhoneGap” [41] by Tome Igoe, Don Coleman and Brian Jepson covers details about NFC close proximity data exchange. Two books that describe iBeacon physical details and programmatic specifications are “Learning iBeacon” [42] by Craig Gilchrist and “Building Applications” [43] with iBeacon by Matthew S. Gast.

Regular expressions are a very convenient method for searching texts and extracting complex information out of web access logfiles. “Regular Expressions: Pocket Primer” [44] by Oswald Campesato, “Regex Quick Syntax Reference: Understanding and Using Regular Expressions” [45] by Zsolt Nagy and the Ben Fortas’ E-book “Learning Regular Expressions” [46] as well as the Regular Expressions chapter 14 of “Beginning Java 8 Fundamentals: Language Syntax, Arrays, Data Types, Objects, Regular Expressions” [47] written by Kishori Sharan and “Introducing Regular Expressions” [48] by Michael Fitzgerald are useful references with examples of typical regular expressions and their handling in various programming languages.

“Performance Testing with JMeter” [10] and “JMeter Cookbook” [49] by Bayo Erinle and also “Apache JMeter” [50] by Emily H. Halili can provide adequate information for accurate external performance measuring.

Finally, literature for development for the Google Web Toolkit framework that was used, includes the standard Manning book “GWT in Action” [51], by Adam Tacy et. al., the Packt publication “Google App Engine Java and GWT Application Development” [52], by Guermeur and Unruh and the Tutorial Point’s “GWT Tutorial” [53].

“Cloud Computing Bible” [54], by Barrie Sosinsky focuses on the platforms and technologies needed for shifting applications to the Cloud for lowering computing costs with a pay-as-you-go contracts that offer infinitely scalable, and universally available systems that allow very fast expansion.

The book “Webbots, Spiders, and Screen Scrapers: A Guide to Developing Internet Agents with PHP/CURL” [55] allows understanding the functionality of spiders and webbots.

The next section elaborates on the general goals of Analytics, further explains the access log file affinity with the application it stems from and explains how to define its contents and configure the generation procedure details. The four types of analytics systems are described and compared. The impact of the shift to Rich Internet Applications with heavy AJAX use to the access log file and the way a remote web server can be configured are explained.

5. WEB ANALYTICS SYSTEMS

The “Web Analytics Systems” chapter elaborates on the general goals of Analytics. It further explains the access log file affinity with the application it stems from and explains how to define its contents and configure the generation procedure details. The four types of analytics systems are presented, described and compared. The impact of the shift to Rich Internet Applications with heavy AJAX use to the access log file and the way a remote web server can be configured are explained.

5.1. ANALYTICS GOALS

To extract knowledge and information from the operation of web applications, algorithms are needed that trace the web page usage and measure key-clicks and actions of all visitors, to gain the needed user behavioral insight.

Basic information provided by an Analytics system is:

a. Overview of the details of the visitors’ requests. Accurate time and exact steps of the visitor are registered, loaded and can be analyzed.
b. Information about the traffic load and the response times of a web site.
c. Approximation of the geographical location of the visitor
d. Details about the user’s browser version and operating environment.
e. Measurements of campaign results.
f. Referrer recognition
g. Possibility of grouping and clustering visitors according to the style of their visit.
h. Measurements of the influence of social media applications to the traffic of a web site.
i. Finding errors of the web site.
j. Capability of displaying measurements in time periods: per hour, day, month, year and comparing time units.
k. Ability of presenting measurements and statistics in near real time (NRT).

As the Internet evolves towards to what is known as Web 3.0, analytics application must also change in order to adapt to this technology shift so they can keep providing results. Standard typical log file analyzers cannot provide results when the web application is a rich Internet application (RIA). This is since they are more web browser bound and make heavier use of Ajax.

5.2. THE ACCESS LOG FILE

Logfiles are automatically generated text files that usually carry very detailed information about the interaction of visitors with a web application. Although both the size and level of detail of the access log file depend on the application, their size makes manual processing difficult and inconvenient. Log File Analyzers have facilities that will read log file lines, preprocess them and load the information into relational databases.

Programmers can always implement their own logging mechanisms. It is though usually more convenient to avoid reinventing the wheel and stick to existing logging systems that have been used by professionals for years, have become de facto standards, are popular and are guaranteed to perform well. For the Java developer for example, the site java-source.net[20] includes a list of logging libraries and front ends, many of which belong to the open source community and can easily be used with any application.

Apache Software Foundation offer two basic types of logs for their web servers: internal and external.

Internal logs or error logs, are used for reporting about the status of the web server operation, and register information and errors:

```
25-Apr-2019 10:03:14.734 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler
["ajp-nio-8788"]
25-Apr-2019 10:03:14.754 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 16687 ms
```

TEXTBOX 1: INTERNAL LOG FILE SAMPLE

External logs are used for logging access to the web server. They register all external requests to the web server and details about each transaction.

[20] http://java-source.net/open-source/logging

```
…

62.1.183.170 127.0.1.1 - 0 62.1.183.170 - GET 8780 ?prodId=8 - [05/Apr/2019:00:00:03 +0300] "GET
/konakart/SelectProd.do?prodId=8 HTTP/1.1" 302 /konakart/SelectProd.do 19 6924E3EAC9BAE3649D0DCF6B4EFFFFB9
"Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101 Firefox/66.0"
"http://aivalisco.ddns.net:8780/konakart/Welcome.do"

62.1.183.170 127.0.1.1 25906 25906 62.1.183.170 - GET 8780
?prodId=8&manufacturer=Warner&category=Cartoons&name=A+Bug%27s+Life&model=DVD-ABUG - [05/Apr/2019:00:00:03
+0300] "GET
/konakart/SelectProd.do?prodId=8&manufacturer=Warner&category=Cartoons&name=A+Bug%27s+Life&model=DVD-ABUG
HTTP/1.1" 200 /konakart/SelectProd.do 29 6924E3EAC9BAE3649D0DCF6B4EFFFFB9 "Mozilla/5.0 (X11; Ubuntu; Linux
x86_64; rv:66.0) Gecko/20100101 Firefox/66.0" "http://aivalisco.ddns.net:8780/konakart/Welcome.do"

62.1.183.170 127.0.1.1 2885 2885 62.1.183.170 - GET 8780  - [05/Apr/2019:00:00:03 +0300] "GET
/konakart/images/dvd/a_bugs_life_1.jpg HTTP/1.1" 200 /konakart/images/dvd/a_bugs_life_1.jpg 0
6924E3EAC9BAE3649D0DCF6B4EFFFFB9 "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101
Firefox/66.0"
"http://aivalisco.ddns.net:8780/konakart/SelectProd.do?prodId=8&manufacturer=Warner&category=Cartoons&name=A
+Bug%27s+Life&model=DVD-ABUG"

62.1.183.170 127.0.1.1 4558 4558 62.1.183.170 - GET 8780  - [05/Apr/2019:00:00:03 +0300] "GET
/konakart/images/dvd/a_bugs_life_3.jpg HTTP/1.1" 200 /konakart/images/dvd/a_bugs_life_3.jpg 0
6924E3EAC9BAE3649D0DCF6B4EFFFFB9 "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101
Firefox/66.0"
"http://aivalisco.ddns.net:8780/konakart/SelectProd.do?prodId=8&manufacturer=Warner&category=Cartoons&name=A
+Bug%27s+Life&model=DVD-ABUG"

62.1.183.170 127.0.1.1 29381 29381 62.1.183.170 - GET 8780  - [05/Apr/2019:00:00:03 +0300] "GET
/konakart/images/dvd/a_bugs_life_3_big.jpg HTTP/1.1" 200 /konakart/images/dvd/a_bugs_life_3_big.jpg 0
6924E3EAC9BAE3649D0DCF6B4EFFFFB9 "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101
Firefox/66.0"
"http://aivalisco.ddns.net:8780/konakart/SelectProd.do?prodId=8&manufacturer=Warner&category=Cartoons&name=A
+Bug%27s+Life&model=DVD-ABUG"

62.1.183.170 127.0.1.1 3377 3377 62.1.183.170 - GET 8780  - [05/Apr/2019:00:00:03 +0300] "GET
/konakart/images/dvd/a_bugs_life_4.jpg HTTP/1.1" 200 /konakart/images/dvd/a_bugs_life_4.jpg 0
6924E3EAC9BAE3649D0DCF6B4EFFFFB9 "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101
Firefox/66.0"
"http://aivalisco.ddns.net:8780/konakart/SelectProd.do?prodId=8&manufacturer=Warner&category=Cartoons&name=A
+Bug%27s+Life&model=DVD-ABUG"

62.1.183.170 127.0.1.1 24667 24667 62.1.183.170 - GET 8780  - [05/Apr/2019:00:00:03 +0300] "GET
/konakart/images/dvd/a_bugs_life_1_big.jpg HTTP/1.1" 200 /konakart/images/dvd/a_bugs_life_1_big.jpg 0
6924E3EAC9BAE3649D0DCF6B4EFFFFB9 "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101
Firefox/66.0"
"http://aivalisco.ddns.net:8780/konakart/SelectProd.do?prodId=8&manufacturer=Warner&category=Cartoons&name=A
+Bug%27s+Life&model=DVD-ABUG"

…
```

TEXTBOX 2: EXTERNAL LOG FILE SNIPPET

JULI or java.util.logging, a renamed fork of Apache Commons Logging, Apache Log4J and Log4j2, the LOGBack Project are popular logging systems. JULI is the logging mechanism for the Apache Tomcat web server internal logging. Apache Catalina allows the insertion of additional components into the request processing pipeline of each associated container[21] for external logging. The characteristics, the format and contents of the generated logfile are defined in a configuration file, defined in the "Access Log Valve". When the access log files generated during the operation of the web server are processed, they provide information and insight into each visit request and response.

Textbox 2 shows a small portion of an access log file of an e-commerce application consisting of just 7 such lines. One can immediately recognize the IP addresses of the server, the number of bytes sent, the fact that these are all get-commands, the parameter prodID=8, which is the code of the requested product, the timestamp, the status of the request, being 302 and 200's, followed by the session-ID, the agent and the referrer. If multiple visitors are being served at the same time their lines stamped by the session-ID's are intermixed. Although the timestamp is the same for all lines in this snippet, `[05/Apr/2019:00:00:03 +0300]`, the sequence of logged events is always in the correct order. The accuracy of one second is enough, since the log file registers manual activities that are activated by the visitor's mouse. One can see from this example that all lines dealing with the click of the user are aimed to loading the appropriate product and its pictures in the visitor's browser. In our case, a DVD Movie from the Cartoons category called "A Bug's Life" (Figure 7). Loading all the pictures and the appropriate information for this product is a sub second activity.

FIGURE 7: A BUG'S LIFE

[21] https://tomcat.apache.org/tomcat-8.5-doc/config/valve.html#Access_Logging

In Apache Tomcat, the enhanced SimpleDateFormat pattern "%{xxx}t xxx", for example using "%{yyyy-MM-dd HH:mm:ss.SSS}t" as instead of %t, would allow millisecond accuracy of the timestamp if necessary. This feature works for Tomcat version 7 and all later versions. The latest version being today version 10.0.0-M1[22].

The timestamp and the session-ID are valuable information as they reveal the steps and the detailed path of each visitor in the e-commerce site.

5.3. How to Generate the Access Log File

A few types of webservers are being used today. Each one offers its own way of reporting and logging events into log files. The format of the log file line output is always, to a large extent, configurable. The default settings usually give only reduced access information and but report exceptions and error conditions.

A reconfiguration of the web server settings is usually required in order to allow the server to report more detailed user access information in the access log file. This is done by setting appropriate values in a specific XML file. With Tomcat for example, it is necessary to reconfigure the *server.xml* file, in order to get access-log files and information about visiting users in a precise way, which allows timing- and response- measurements, as well as access-attitude information of prospective or existing customers.

Apache Tomcat provides the configuration directory conf, that contains the file *server.xml*. At the bottom of this file, a line that includes the adjustments valve parameter, called *AccessLogValve* must be un-commented if it is and changed. This is a one-time setting that can be performed either by the Log Analyzer if it supports this feature, or manually. Tomcat needs to be restarted, so that the new settings take effect. If there are errors in the server.xml file, the web server will not start, and the error messages will be registered in the internal log file.

The default valve parameters and patterns can easily be modified, either through an editor, or by the application. A reasonably detailed log file that can be used for access metrics may look like in Textbox 3:

[22] https://tomcat.apache.org/download-10.cgi

```
<Valve className="org.apache.catalina.valves.AccessLogValve"  directory="logs"
      prefix="kk8_access." suffix=".log"
      pattern='%a %A %b %B %h %l %m %p %q %u %t "%r" %s %U %D %S "%{User-Agent}i" "%{Referer}i"'
      resolveHosts="false" rotatable="true" fileDateFormat="yyyy-MM-dd" />
```

TEXTBOX 3: AN ACCESS LOG VALVE

Almost all necessary options are activated in the example above, in order to obtain as much information as possible in the access log file. In general, the more pattern codes are used, the larger each line of the access log file will be. The included codes are a matter of the company policy and define the contents and the sequence of the information stored in each line of the produced log file. The LFA will parse each log file and load the data to a database. This phase offers a second chance to leave out specific information from the log file, generated by any code that seems excessive. If the pattern is modified often, then the web server generates log files with different formats and are a little more difficult to process together. Thus, it is advisable not to change the pattern all too often. Changes to the pattern automatically imply also changes to the software that is used in order to parse the log file for loading to a database for further processing.

The pattern variables are usually enhanced with every new version of the web server. The pattern variables of Apache Tomcat 8.5 can be seen in Table 1:

TABLE 1: PATTERN VARIABLE OPTIONS FOR ACCESS VALVE[23]

%a	Remote IP address
%A	Local IP address
%b	Bytes sent, excluding HTTP headers, or '-' if zero
%B	Bytes sent, excluding HTTP headers
%h	Remote host name (or IP address if `enableLookups` for the connector is false)
%H	Request protocol
%l	Remote logical username from identd (always returns '-')
%m	Request method (GET, POST, etc.)
%p	Local port on which this request was received. See also `%{xxx}p` below.
%q	Query string (prepended with a '?' if it exists)
%r	First line of the request (method and request URI)

[23] https://tomcat.apache.org/tomcat-8.5-doc/config/valve.html#Access_Logging

%s	HTTP status code of the response
%S	User session ID
%t	Date and time, in Common Log Format
%u	Remote user that was authenticated (if any), else '-')
%U	Requested URL path
%v	Local server name
%D	Time taken to process the request in milliseconds. Note: In httpd %D is microseconds. Behavior will be aligned to httpd in Tomcat 10 onwards.
%T	Time taken to process the request, in seconds. Note: This value has millisecond resolution whereas in httpd it has second resolution. Behavior will be aligned to httpd in Tomcat 10 onwards.
%F	Time taken to commit the response, in milliseconds
%I	Current request thread name (can compare later with stack traces)

Beside the contents of the log file, its location (directory path relative to default path), name and characteristics can be configured. The directory attribute defines the location of the resulting file relative to the installation directory. The prefix of the file name, as well as the suffix can be customized here. When the rotatable parameter is set to “false” the system creates one single log file, that grows as event lines are appended. The basic problem of this setting is that the log file of a site with heavy use tends to become large very quickly, and hence difficult to manage. When the rotatable attribute is set to true, a new log file is recreated every time there is a change of date or hour, depending on whatever matches the content of the *fileDateFormat* parameter. When set to “yyyy-MM-dd”, the date format is used as part of the file name and we have a new log file for every day. Combined with *rotatable*=”true” it triggers the log file to close and a new one to open as soon as the day changes. If there would be an “.HH” addition to the file date format, the system would produce a new log file with the new file name every hour.

As stated above, the AccessLogValve allows Apache-Tomcat to produce the details described by the pattern variable. If the AccessLogValve tag is commented out, then no access log file is generated. By default limited information is generated and common pattern is used. The variable *pattern* = ”common” defaults to '%h %l %u %t "%r" %s %b', while pattern = "combined" appends the values of the Referrer and User-Agent headers, each enclosed in double quotes, to the common pattern.

When the modified valve pattern of Textbox 3 is used, seven lines of the generated logfile are shown in the ASCII file in the External Log File Sample Textbox 2. This file is relatively difficult to read manually, since it may contain hundreds of thousands of lines. Any request for information and statistical analysis is very difficult to calculate without extracting every line, transforming it and loading to a database table. Sometimes it may be necessary to check the log file, as generated, with the use of a text editor for finding specific information on exceptionally occurring events, such as cracker attacks. This evaluation work becomes a lot easier when a Log File Analyzer (LFA) is available.

5.4. Four Types of Analytics Systems

There are four conceptual types of analytics systems:

1. Pure Logfile Analyzers

   Log files are semi-structured files containing detailed data about each web transactions and request. The data must be carefully selected and processed in order to provide information. This processing extracts and transforms the data generated by the web serve to the application framework that produces metrics.

2. Page Tagging

   Page Tagging is done through JavaScript code, embedded in the web page, that sends the visitor's browser to a dedicated external web server, along with the page URL or name and a preassigned code to the applications account number automatically, after visiting each page.

   The dedicated server logs the visit data, stores it to a special data base for each site, based on its account number and offers access to the metrics it produces.

3. Network Data Collection Devices

   These are hardware devices, or specialized software that intercept IP packages from the network and allow logging and analyzing traffic. These devices are called packet sniffers, since they operate on packet level.

4. Hybrid Methods

   We call hybrid systems that combine usually two approaches, usually Logfile Analyzers and Page Tagging. The disadvantages of each method are alleviated.

Types 1, 2 and 4 are taken into consideration in this work. Sniffers are primarily used by network specialists for checking for network flaws and bottlenecks.

Advantages and disadvantages of log file analysis versus tagging will be analyzed in the next paragraph. We will see that type 1's disadvantages are type 2's advantages and vice-versa. Therefore, using type 4, which is hybrid method, eliminates the disadvantages and allows exploiting all advantages of both techniques.

5.5. LOGFILE ANALYSIS VS. TAGGING

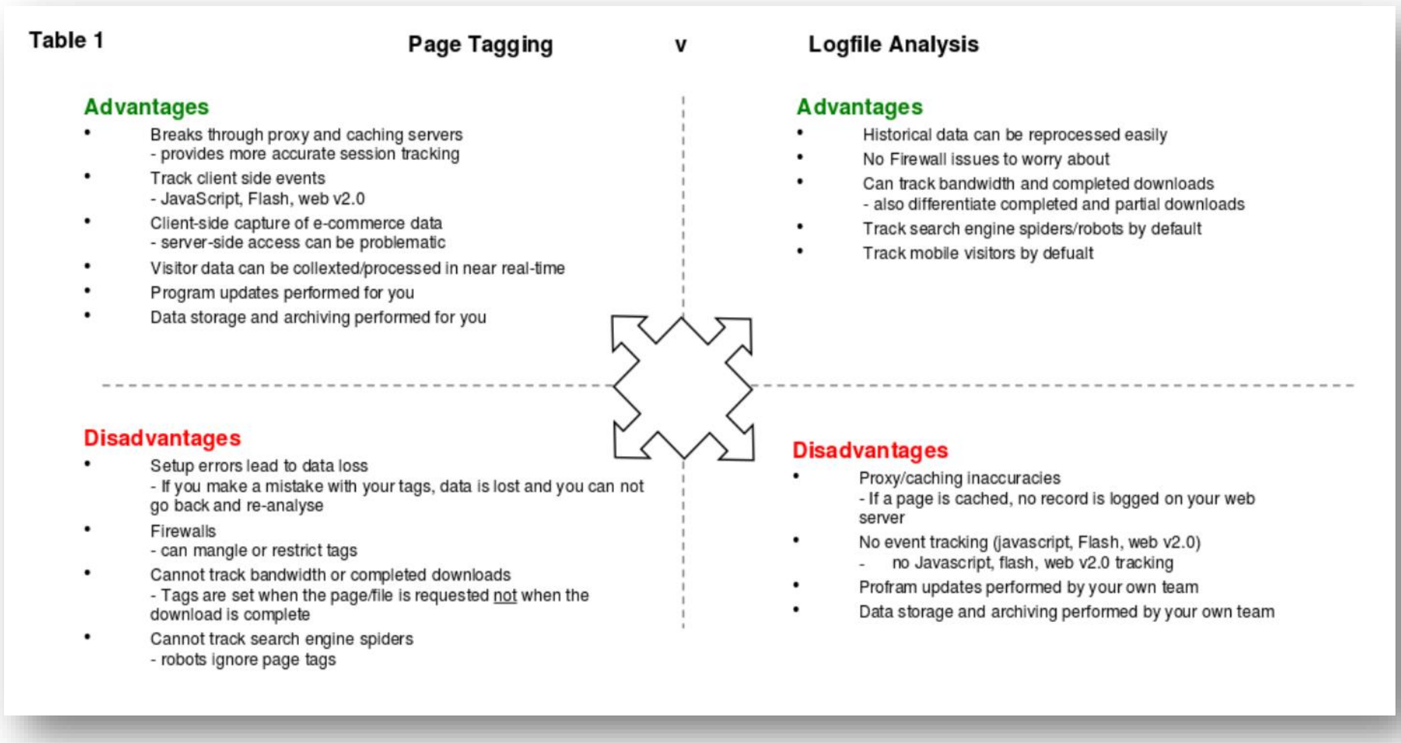


Table 1 **Page Tagging** **v** **Logfile Analysis**

Page Tagging – Advantages

- Breaks through proxy and caching servers
  - provides more accurate session tracking
- Track client side events
  - JavaScript, Flash, web v2.0
- Client-side capture of e-commerce data
  - server-side access can be problematic
- Visitor data can be collexted/processed in near real-time
- Program updates performed for you
- Data storage and archiving performed for you

Page Tagging – Disadvantages

- Setup errors lead to data loss
  - If you make a mistake with your tags, data is lost and you can not go back and re-analyse
- Firewalls
  - can mangle or restrict tags
- Cannot track bandwidth or completed downloads
  - Tags are set when the page/file is requested not when the download is complete
- Cannot track search engine spiders
  - robots ignore page tags

Logfile Analysis – Advantages

- Historical data can be reprocessed easily
- No Firewall issues to worry about
- Can track bandwidth and completed downloads
  - also differentiate completed and partial downloads
- Track search engine spiders/robots by default
- Track mobile visitors by defualt

Logfile Analysis – Disadvantages

- Proxy/caching inaccuracies
  - If a page is cached, no record is logged on your web server
- No event tracking (javascript, Flash, web v2.0)
  - no Javascript, flash, web v2.0 tracking
- Profram updates performed by your own team
- Data storage and archiving performed by your own team

FIGURE 8: BRIAN CLIFTON'S PAGE TAGGING VS. LOGFILE ANALYSIS TABLE

As stated above, we consider the three common approaches as means for measuring Web Traffic: Logfile Analysis, Tagging systems and Hybrid solutions.

The very detailed semi-structured information about each tiny request by any client over the operation period that need to be extracted from the log file and parsed into a database and querying against the database in order to obtain relevant information procedure needs to be orchestrated by the owner of the e-commerce application.

Page Tagging systems, on the other hand, use an external commercial web server, to whom the visitor's browser is automatically rerouted every time any web page is requested. This server collects log data generated by the visitor's browser to a log file and provides analysis techniques as a service that are used to extract hits of the specific web sites and makes the information available, for all customers based on their account number. Tagging system operators are responsible for dealing with all details and providing results.

Hybrid systems combine both technologies. They offer the advantages of both methods and this way they alleviate the disadvantages of each method.

Brian Clifton's comparative diagram in Figure 8 shows the advantages and disadvantages of both techniques.

LOGFILE ANALYSIS

ADVANTAGES

- Historical data can be reprocessed easily
- No Firewall issues to worry about
- Can track bandwidth and completed downloads

  -also differentiate completed and partial

- Track search engine spiders/bots by default
- Track mobile visitors by default

DISADVANTAGES

- Proxy/Caching inaccuracies

  - If a page is cached, no record is logged on your web server

- No event tracking (JavaScript, Flash, Web 2.0)

  - no JavaScript, Flash, Web 2.0 tracking

- Program updates performed by your own team
- Data storage and archiving performed by your own team

PAGE TAGGING

ADVANTAGES

- Breaks through proxy and caching servers

  - provides more accurate session tracking

- Track client-side events

  - JavaScript, Flash, web 2.0

- Client-side capture of e-commerce data

  - server-side access can be problematic

- Visitor data can be collected/processed in near real-time
- Program updates performed for you by the vendor
- Data storage and archiving performed for you by the vendor

DISADVANTAGES

- Setup errors lead to data loss

  - If you make a mistake with your tags, data is lost, and you cannot go back and re-analyze

- Firewalls

  - can mangle or restrict tags

- Cannot track bandwidth or completed downloads

  - Tags are set when page/file is requested not when the download is complete

- Cannot track search engine spiders

  - robots ignore page tags

It is obvious that only a hybrid solution can provide a complete analysis of the web site visitor behavior. Because of their complexities, only a small number of vendors can offer a hosted hybrid solution [56]. We approach this complexity by setting up a log file analyzer with tagging extensions.

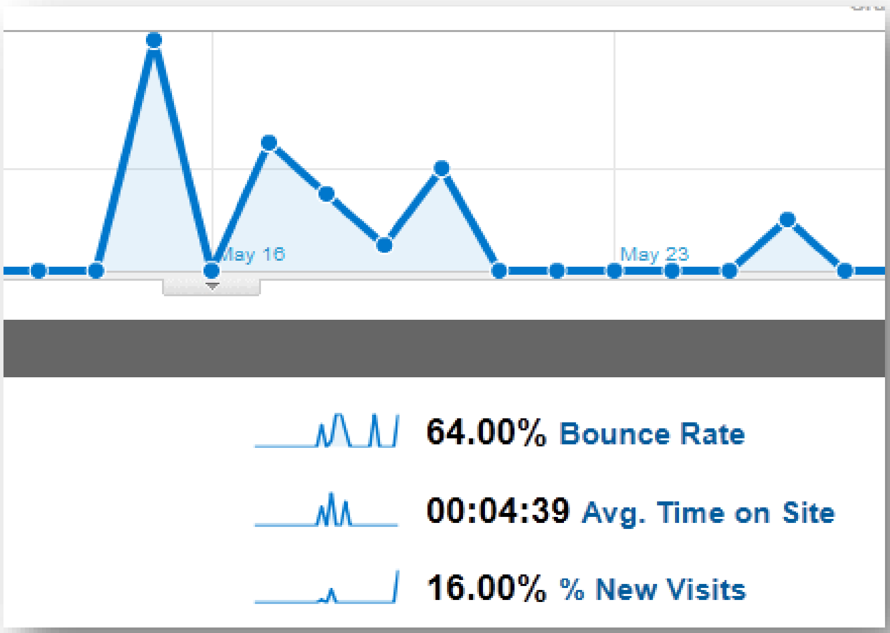
5.6. PAGE TAGGING SYSTEMS

Google Analytics page tagging system, as an example, requires the following JavaScript code to be inserted in each web page that needs to be tracked and made part of the analytics procedure:

```
<meta name="google-site-verification" content="XXXXXXXXXXXX" />
<script type="text/javascript">
var _gaq = _gaq || [];
_gaq.push(['_setAccount', 'UA-XXXXXXXX-Y']);
_gaq.push(['_trackPageview']);
(function() {
   var ga = document.createElement('script');
   ga.type = 'text/javascript';
   ga.async = true;
   ga.src = ('https:' == document.location.protocol ?
      'https://ssl': 'http://www') + '.google-analytics.com/ga.js';
   var s = document.getElementsByTagName('script')[0];
   s.parentNode.insertBefore(ga, s);
}) ();
</script>
```

GOOGLE ANALYTICS JAVASCRIPT SNIPPET

This overhead allows Google Analytics to provide detailed information about each page of the site registered. The innovation of the approach is that the owner of the site does not need to generate or analyze a local log file, since Google take the responsibility of doing the analysis for them, providing a web interface and even an API for this service (Figure 9).



64.00% Bounce Rate

00:04:39 Avg. Time on Site

16.00% % New Visits

FIGURE 9: GOOGLE ANALYTICS ACCESS RESULTS

5.7. USAGE EXAMPLE

This is a typical visitor access scenario, including the following access steps, which may occur in a simple session. An already registered customer returns to the e-shop to place an order. The steps followed by the customer shown in this textbox may be:

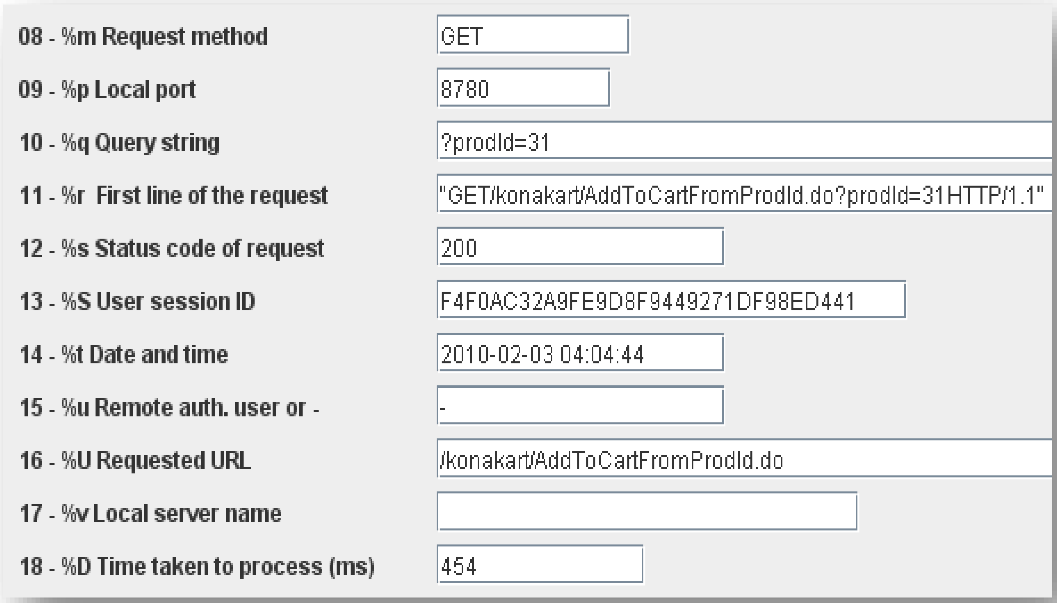
- **connection with the shop and display of the first page,**
- **selection of the required language if not already selected,**
- **product search through a keyword or category**
- **display of proposed items matching the key or category,**
- **adding to product to cart**
- **completion of the transaction after the customer enters an e-mail and the appropriate password**
- **selection of pay- and shipment- method**
- **confirmation and disconnection**

This is a very common customer visit for a re-purchase. The eight steps above may generate for example at least 250 to 400 lines entries in the access log file of the e-shop, depending on the implementation framework of the site. These lines include “GET” request lines for every single tiny image and component needed to render the page. These lines must be separated, from the actions. Since the size of the image and the time is included, these details can be used for calculating accurately the overall throughput.

The resulting data is ordered by the timestamp and the session-ID. The timestamp defines the exact date and time of every step and can be used to measure the sequence of selections, and the start and end-time of each visit. Since an e-shop is simultaneously accessible by many users, the session data lines are scattered between different user-sessions and ordered by date and time in the log file. It is very simple to separate or join the information at any time for measuring performance since it is transferred into the transactions database table of the log analyzer toolbox.

The statistics results of an e-shop may also be influenced, besides robot and crawler visits, through frequent administrator accesses through the back-office application to the database. These are necessary steps, to maintain the shop data, pricing, check and service orders etc. All these entries must be identified and filtered out of the customer behavior statistics results.

The action “*AddToCartFromProdId.do”*, as expected, adds a product to the cart. This is an example of an important action, stored in the actions database table of the log analyzer toolbox and has one parameter, the product ID. It is part of the requested URL and it appears through the %r and %U parameters of the Tomcat valve, combined with the *prodId=31* parameter, visible at the end of the URL also in the log file. This information indicates that some user, from the IP address that is also visible and maybe even known, as well as the timestamp at the given time added this product to the cart. We also have the session-Id it belongs to. Details from the log file, as it finally was transferred to the transactions database table are viewed in Figure 10.



08 - %m Request method	GET
09 - %p Local port	8780
10 - %q Query string	?prodId=31
11 - %r First line of the request	"GET/konakart/AddToCartFromProdId.do?prodId=31HTTP/1.1"
12 - %s Status code of request	200
13 - %S User session ID	F4F0AC32A9FE9D8F9449271DF98ED441
14 - %t Date and time	2010-02-03 04:04:44
15 - %u Remote auth. user or -	-
16 - %U Requested URL	/konakart/AddToCartFromProdId.do
17 - %v Local server name	
18 - %D Time taken to process (ms)	454

FIGURE 10 REQUEST ACTION WITH PRODUCTID AND SESSIONID

The analyzer generates various reports. In Figure 11 the monthly traffic diagram combines (scaled) accesses with time and bytes transferred.

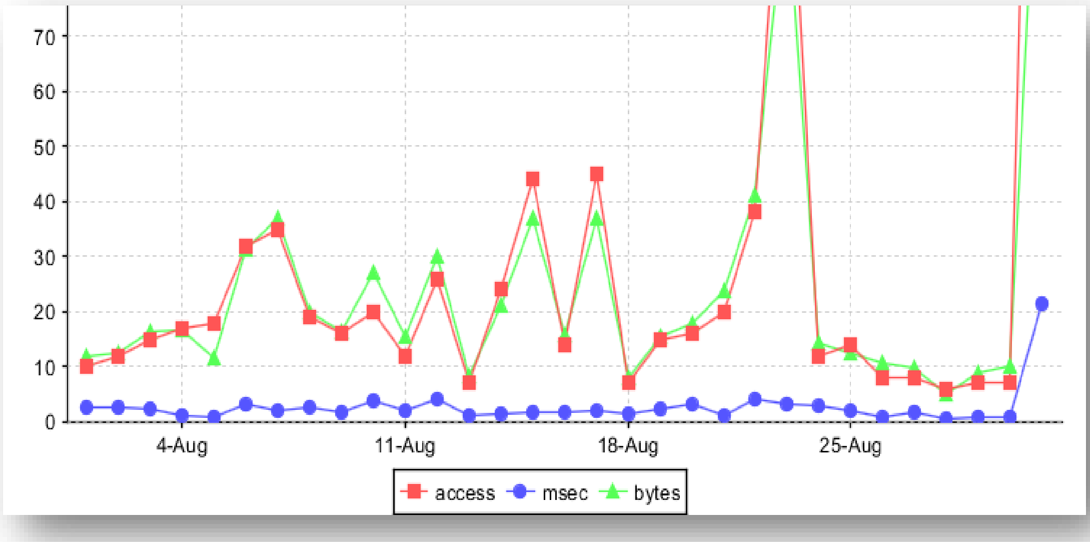


FIGURE 11 MONTHLY TRAFFIC DIAGRAM

Since log file data is transferred to the database, it can be analyzed with SQL-queries via JDBC. This offers great flexibility and speed. Complex queries are formed as select statements with their appropriate joins.

The "E-Commerce Application Characteristics" chapter that follows points out the importance of E-Commerce, lists systems of different types and specific e-commerce applications available, discusses component mix and design and architecture issues.

6. E-COMMERCE APPLICATION CHARACTERISTICS

The “E-Commerce Application Characteristics” chapter points out the importance of E-Commerce, lists systems of different types and specific e-commerce applications available, discusses component mix and design and architecture issues.

6.1. IMPORTANCE OF E-COMMERCE SYSTEMS

The Internet revolution, the speed improvements of the connections and its vast expansion via mobile devices during the last few years, as well as the global acceptance of social media that boosted promotion, allowed e-commerce to become a mainstream activity for a very large number of businesses. E-commerce improves all business activities by speeding up buying and selling procedures. The Web alleviates geographical limitations and provides low cost 24/7 accessibility to the store. It also allows to extend the customer base beyond city limits or geographical boundaries. The 24/7 availability extends the business operation hours and days without high additional costs and saves money by automating the entire purchase procedure, reducing communications and practically eliminating manual intervention during customer-business interaction[24].

Contemporary, full-fledged e-commerce applications are usually complicated web-based software systems, consisting of large numbers of components, services, interfaces and Application Programming Interfaces (API s), in various levels of integration. Typically, they consist of inventory management components, customer management software, order and invoicing applications, payment modules and shipping modules as well as a number of visualization and reporting components, back-end management and a store front interface.

E-commerce sites are supersets of classic inventory applications that deal with sales in a traditional way but offer a web interface that supports sales and must substitute the salesperson. The information of the products offered contains beside the standard data, detailed product description, links to manufacturer or manuals, various images, product type category, pricing details, discount conditions and periods and evaluations of other customers, going a little beyond the scope of what typical inventory application systems offer.

[24] https://go.forrester.com/

Since communication with the customer is done through the site, personal details from every customer must include password and email address.

Customer management is necessarily involved since pricing policies, discounts and orders must be personalized. Often e-commerce site operators use in parallel dedicated Customer Relationship Management applications.

At a very simplistic abstraction level, an e-commerce application can be viewed as a web-based product catalog with integrated payment services.

Beside commercial e-commerce applications, a substantial number of public-domain, free source and open applications are available for many operating system platforms and complexity levels, that are customizable and capable of dealing with almost any type of e-shop.

Examples of very successful, well known and widely used free, open or moderately priced e-commerce applications[25] include the popular free **OsCommerce** (https://www.oscommerce.com/), a traditional application that was released in year 2000, with a developer community that offers a vast number of plugins and add-ons. **Magento** (https://magento.com/products/open-source), is a very complete open-source system. It has a fast-growing installation base and is offered in both Community or Enterprise releases, with the former being free and the latter having a better support and featuring higher functionality. **OpenCart** (https://www.opencart.com/) was developed in 1998, **PrestaShop** (https://www.prestashop.com/en) and **ZenCart**, all of which are PHP and MySQL or MariaDB based applications. Then there is **Spree Commerce** (https://spreecommerce.org/), a free e-commerce application written in Ruby on Rails, that uses either MySQL or PostgreSQL. **Shopify** (https://www.shopify.com/), is hosted by its vendor. **KonaKart** (https://konakart.com/), is a Java Application based on the Struts2 framework and using a sophisticated persistence layer that supports the use of any Database Management System. Initially Konakart was introduced as an OSCommerce workalike, fully compatible with the OSCommerce database, probably with the goal to tempt OSCommerce customers to substitute their OSCommerce application with a faster workalike. **BigCommerce** is different type of system. It is a self-hosted application (https://www.bigcommerce.com/). Self-hosted applications, like Shopify and BigCommerce provide extensive Application Programming Interfaces that allow integration with Enterprise Resource Planning Systems and access to data, that is stored at the remote host.

25 https://blog.capterra.com/best-free-open-source-ecommerce-software-solutions/

Another important type of e-commerce application are e-commerce plugins and add-ons, that can be integrated into specific general-purpose Content Management Systems, like **Magnolia** CMS and **Hybris**, **OpenCms** and **Konakart**, **Joomla** and **VirtueMart** or **HikaShop**, **JooCommerce** or **eShop**.

For some reason, the majority the most representative free e-commerce applications have been developed in PHP and run under Apache2 httpd server.

Typical e-commerce applications are designed to look and feel like well-designed virtual interactive product and services price catalogs. They exhibit and present products, in a fancy but simple manner through any browser and use multiple photographs and descriptions of their characteristics, details and features and implement a metaphor of a shopping cart. They usually support a front and a back-end application interface, like typical content management systems. The front end is used by customers and the backend by administrators.

The backend has functionality comparable to that of desktop inventory applications and ERPs and is used to maintain the data of the e-shop and operations. Products, payment schedules and policies, shipping modules and processing of the orders and customers are handled here.

E-commerce applications appear in a multitude of forms:

1. As standalone web applications for retail applications.
2. As extensions or part of classical web sites or content management systems.
3. Often, when a company offers a very limited number of products or services, as simple PayPal buttons, appearing in general purpose static web pages, next to these service or product descriptions.
4. Airline and hotel reservation systems are also specific forms of e-shops, since they although they are more often characterized as reservation systems, they do offer e-shop functionality.

Typical retail e-shops are the most sophisticated and extensive applications of all forms. They share many standard architectural characteristics, design and features:

Commonly used front-end filtering components of all typical e-shop implementations:

1. Category control.
2. Manufacturer selector.
3. Text Search which allows misspelling and combining words.
4. Price range filters.
5. Sorting mechanism, by product code, price and popularity.

In addition, e-commerce applications can accept and providing product reviews and evaluations by customers and visitors, which prove to be a very useful tool for the entire user community. Figure 12 shows the components of an e-commerce application.

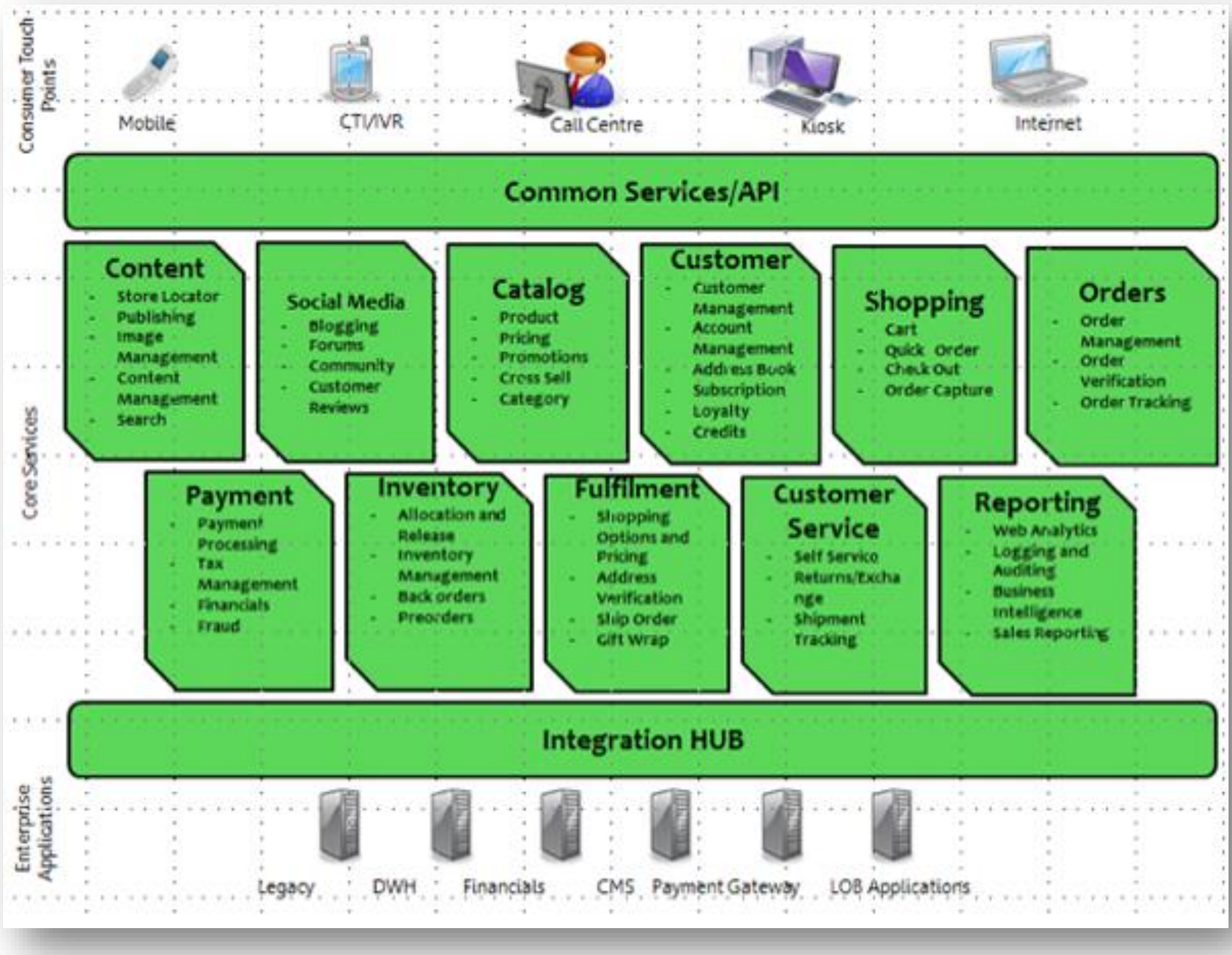


FIGURE 12: COMPONENTS OF AN E-COMMERCE SYSTEM[26]

[26] https://www.bptrends.com/ecommerce-in-the-customer-empowerment-era/

6.2. Database Entity Relationship Diagram

The Entity Relational Diagram of some typical E-Commerce site is shown in Figure 13. The “crows’ foot” notation shows the exact relations between the database tables. The great similarity to an Inventory, Invoice, and Customer-Supplier application is immediately evident.

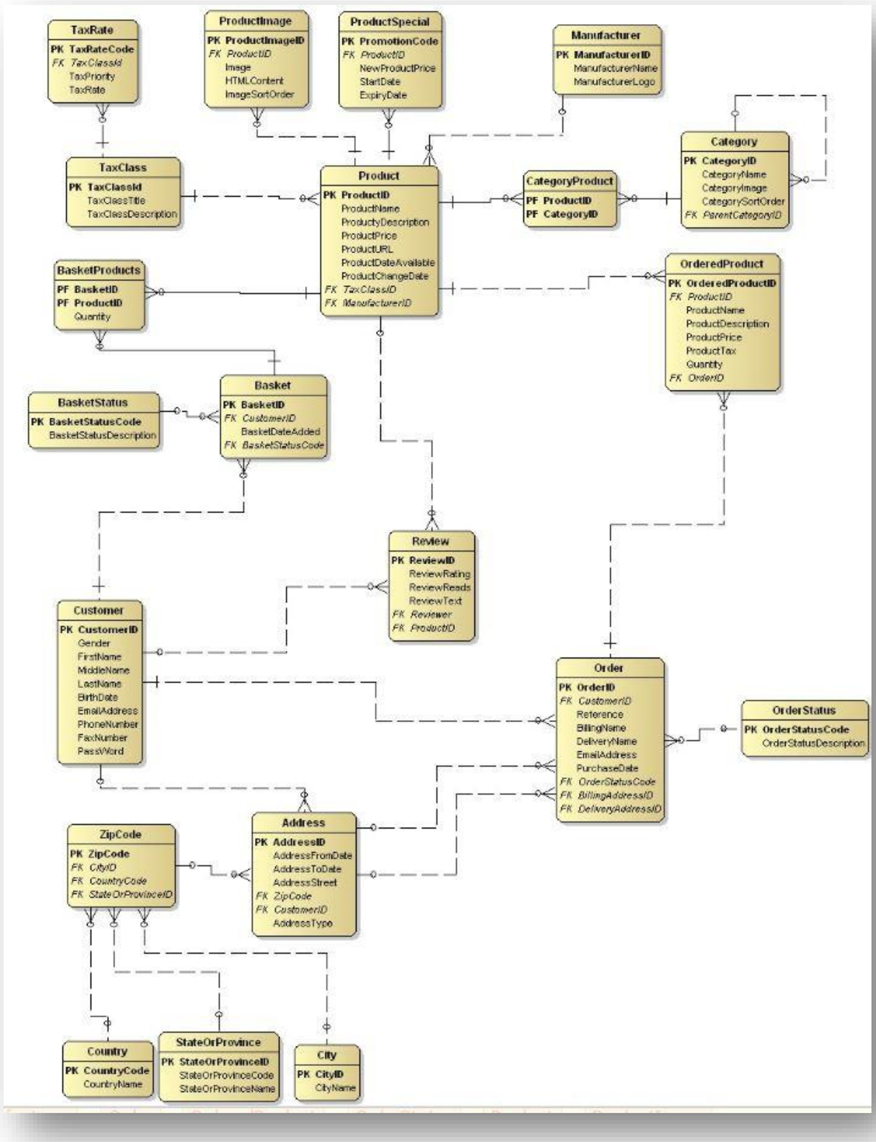


Figure 13: ERD of a Typical ECommerce Application[27]

27 http://www.erdiagrams.com/datamodel-online-shop-crowsfoot.html

The following chapter covers issues and characteristics of the e-commerce environment and the nature of the problem have been defined in the previous sections, the "Specifications and Requirements for the Analyzer Application" chapter elaborates on the nature, the characteristics and features of the LFA system that provide analytics in this environment. The chapter starts with a use-case diagram, showing the basic components of this system and continues with functional and non-functional requirements. The two "natures" of the LFA, "on demand" and the "real time" are presented along with an example showing how customer orders could feed the analyzer automatically in near real time and by adding appropriate triggers to the database of the e-commerce site. Four types of LFAs and their capabilities and extent are presented, all of which can be based on the same software backbone.

7. SPECIFICATIONS AND REQUIREMENTS FOR THE ANALYZER APPLICATION

The issues and characteristics of the e-commerce environment and the nature of the problem have been defined in the previous sections, the "Specifications and Requirements for the Analyzer Application" chapter elaborates on the nature, the characteristics and features of the LFA system that provide analytics in this environment. The chapter starts with a use-case diagram, showing the basic components of this system and continues with functional and non-functional requirements. The two "natures" of the LFA, "on demand" and the "real time" are presented along with an example showing how customer orders could feed the analyzer automatically in near real time and by adding appropriate triggers to the database of the e-commerce site. Four types of LFAs and their capabilities and extent are presented, all of which can be based on the same software backbone.

7.1. Introduction

A log file analyzer can have various forms. It can function as a web application, as well as a standard desktop application, built with a graphical user interface or even as a desktop application with additional web interface and share both natures. A menu driven application designed around the toolboxes is a very familiar setup to almost anybody and is simple and straight to operate, since it is a common well-known setup, familiar to all and common to many applications, like editors, word processing systems etc.

The e-shop administrator should be given a user-friendly and reliable application to work with, with crisp and responsive user interface. This application should be able to quickly track the transaction history of any e-shop, allow setting up required parameters, find measurements and sequences of actions, and compute performance indexes of actions or of the entire application, reveal the results of measurements under various load conditions and compare with previous operation periods.

The architecture must be easily expandable, in order to allow easy inheritance of its base classes, which handle data imports and make simple negotiations with the various types of web servers, so that whenever web sites that run on new web server architectures must be analyzed, their respective log files and any e-shop application databases could be adapted and evaluated relatively fast. In order to be usable, the toolbox must contain a tool with the ability to load entire log files or log file deltas as easy as possible and display the e-shops latest status for evaluation at any time.

The use-case diagram in Figure 14 describes visually the system requirements for the LFA. The goal of such a system is simple: support the easy loading of log files to an environment that allows the generation of statistics, reports and visualizations.

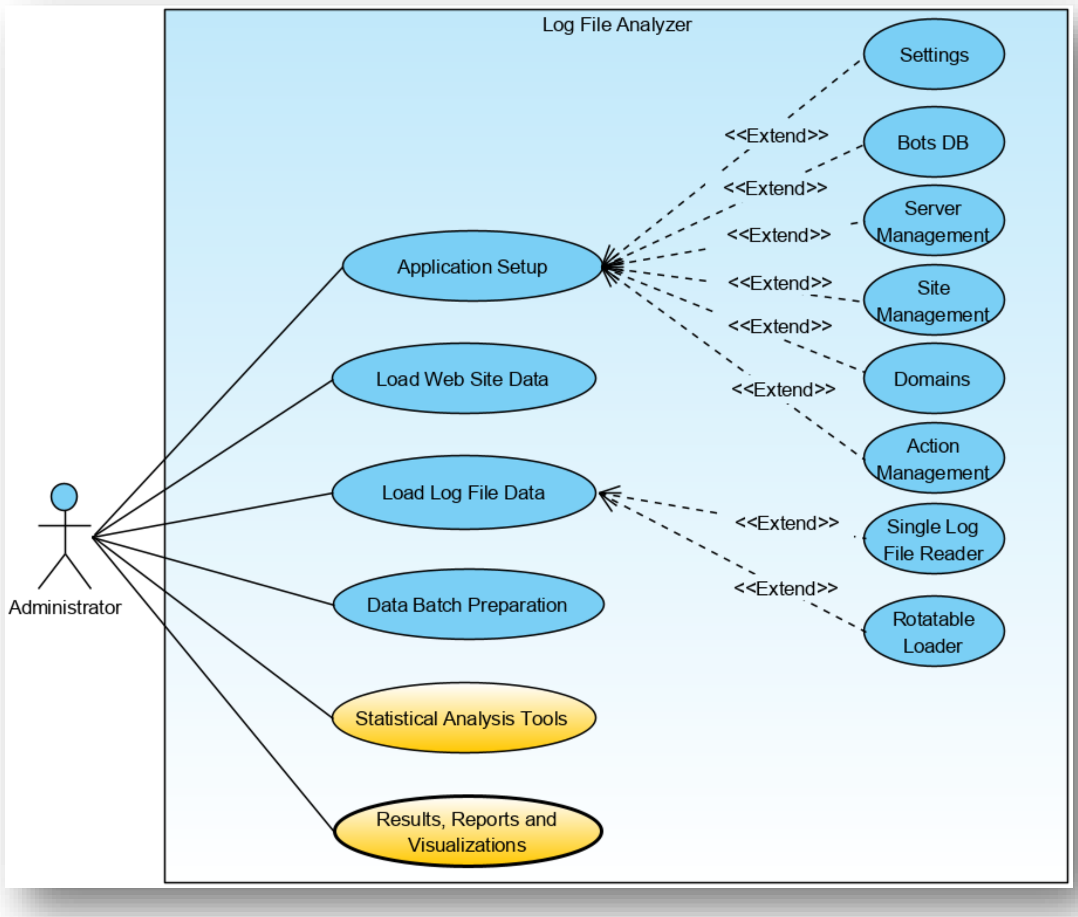


FIGURE 14: USE-CASE DIAGRAM OF THE LFA

The administrator, or the administrators, of the system, must be able to:

- setup the application environment. This includes activities like:
  1. defining and maintain access to the database of the LFA,
  2. the types of the web servers where the ecommerce sites operate,
  3. maintaining known bot applications and spider engines,

4. configuring the sites that will be examined by specifying their URL, HTTP port, the logfile specifics,
5. configuring the relevant actions for the automated Customer Behavioral Graph generation,
6. simple domain management in order to recognize the geographical location of the visitors.

- load database details and by copying category and product details,
- load the access log file rotatable or non-rotatable,
- prepare data batches and
- run visualizations reporting and statistics.

7.2. Requirements

Non-Functional Requirements

Non-Functional Requirement Number	***Non-Functional Requirement Description***
NFR 1	The LFA system must respond quickly, without letting the user wait long. This point is where all tricks and techniques need to be applied, since log files are usually very large ASCII files containing lots of details. They are often located in remote web servers and they must be extracted and loaded through DSL connections that might be slow.
NFR 2	The LFA should scale well, because data volume grows fast as time passes and years of historical data should be available for comparing and mining purposes.
NFR 3	The storage capacity should be unrestricted. Shifting from a local file system to a Hadoop file system cluster could become necessary and should be pre-planned in order to be easily feasible.
NFR 4	The LFA must be reliable and the data should be safe.
NFR 5	The software should easily adapt to new web server types and new versions since any platform may be used with web applications.
NFR 6	The LFA system should be able to run on multiple operating environments and its portability should be guaranteed.
NFR 7	The system must be easy to setup, manage and use in order to be effective.

FUNCTIONAL REQUIREMENTS

Functional Requirement Number	***Functional Requirement Description***
FR 1	The main user of the LFA is the web application administrator.
FR 2	Each log file processed should be reloadable and easy to delete from the database of the Analyzer.
FR 3	The parsing should be done using a regular expression which must be analogous to the pattern of *AccessLogValve* entry.
FR 4	Authentication is required, in order to maintain secure access to any remote system involved, that hosts the web application, the configuration files and the log file under scrutiny, as well as access to read and download data from its operational database.
FR 5	The LFA must adapt the standards of the web servers it is checking.
FR 6	Graphics and Reports should be generated on demand. Visualization and reporting are more efficient if it based on some well-designed standard commercial Report Generator Application. This simplifies the ad-hoc and routine reporting procedure.
FR 7	The data base should accommodate historical data in order to support year, month and day value comparisons and data mining.

The application must be easily expandable, since web servers evolve. Easy adaptation to novel features and new types of web servers would be an asset, because new versions appear frequently. Even if the LFA is used only for accessing one web application, it should support newer versions of the web site with minor software modifications. Figure 16 shows the main software components of the application for log file analysis on demand. The administrator runs the system, asks for log files, the system loads and parses the selected log file data, loads it to the database and is ready to produce visualizations.

Main goal of Analyzer Application is to provide insight and information about the way a web application is visited, the behavior of the users and performance of the system. The generation and visualization of metrics in various forms. The application must be able to adapt easily to various operational scenarios and act as a tool, or as a mix of tools, capable of dealing with the generated data.

We classify the Analyzer Application based on input acquisition mode in the following 2 types (Figure 15):

i. **On Demand**
ii. **Near Real Time** with Streaming and Big Data
iii. **Enhanced** or Combined

Further the Analyzer Application is classified based on scope of input in the following 4 types:

iv. **Standard** Logfile Analyzer
v. **Extended** Logfile Analyzer with adapted operational data from the e-shop
vi. **Hybrid LFA**, capable of working with API's for tagging systems as well
vii. **Social Media Aware** with Open Communication Channels.

Any Analyzer of any of the above 4 input-types can acquire data by any of the 3 input acquisition techniques.

The proposed ideal approach must combine both log file analysis as well as tagging, be extended and Social Aware with combined acquisition capabilities to be more versatile. So, we propose a hybrid system. Still several additional innovations must extend its functionality.

The measurements that can will be presented and visualized are of both operational and business nature.

Operational measurements deal mainly with the operating environment and are mainly performance related. They provide the administrator with response times, speed, response behavior and load of the system.

Business measurements provide information about sales, profitability and turnover measurements and data about origin and behavior of customers, as well as overall general business information and customer behavioral information.

The ideal Analyzer, is a system consisting of two applications that may operate independently:

1. An On-Demand Analyzer application (ODA), which is a standard analyzer tool, customized and extended in such a way that it is capable of directly including operational data from the e-shop, allowing the operator to receive operational metrics based on a familiar environment as if it were an extension of the e-shop, using known product codes and procedures.

2. A Near Real Time Analyzer web application (NRTA), which makes operational details available and eventually visible on browsers and can display immediately the status of operations in near real time.

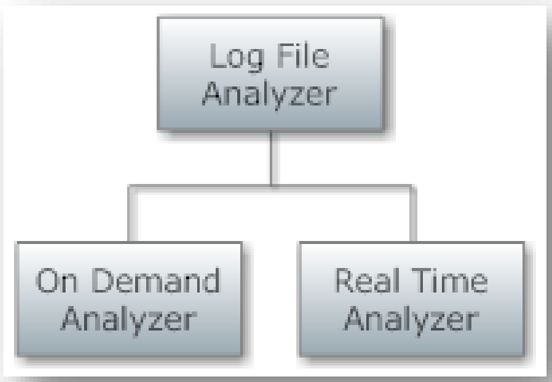


FIGURE 15: ENHANCED ANALYZER ARCHITECTURE

Although each of these two applications can exist on its own, if used together, they can act as complementary components. They can share a common database as well as a common application programming interface. The first application is a static e-shop scrutinizing and user action analysis tool, while the second is a generator and presenter of live measurements and performance metrics in near real time.

7.3. ON-DEMAND ANALYZER

The On-Demand Analyzer application consists of the components shown in Figure 16. The general architecture diagram can be found in Figure 1. The components used for input and adjustments of the application include:

- Settings about servers, and e-shops that will be processed
- Rotatable and non-rotatable log file delta data readers and loaders
- Import of e-shop specific data like items, categories, customers, statuses and orders from the e-shop databases

The processing that is done by the analyzer is:

- Preprocessing of the log file and import into the transactions table, in batches.

- Log file Analysis
- Statistical Information Report generation
- Chart and graph computational algorithms

Output produced:

- Graphical reports
- PDF output
- Various Printouts and reports
- Customer Behavior Business Graph

The basic differentiation of this LFA from standard plain log file analyzers lays in the fact that the architecture proposed in Figure 1 can configure the web server of the e-commerce site to produce an access log file with specific characteristics and contents and can also adapt and embed product and category information from the e-shop that are used in order to customize the reports and the visualizations produced.

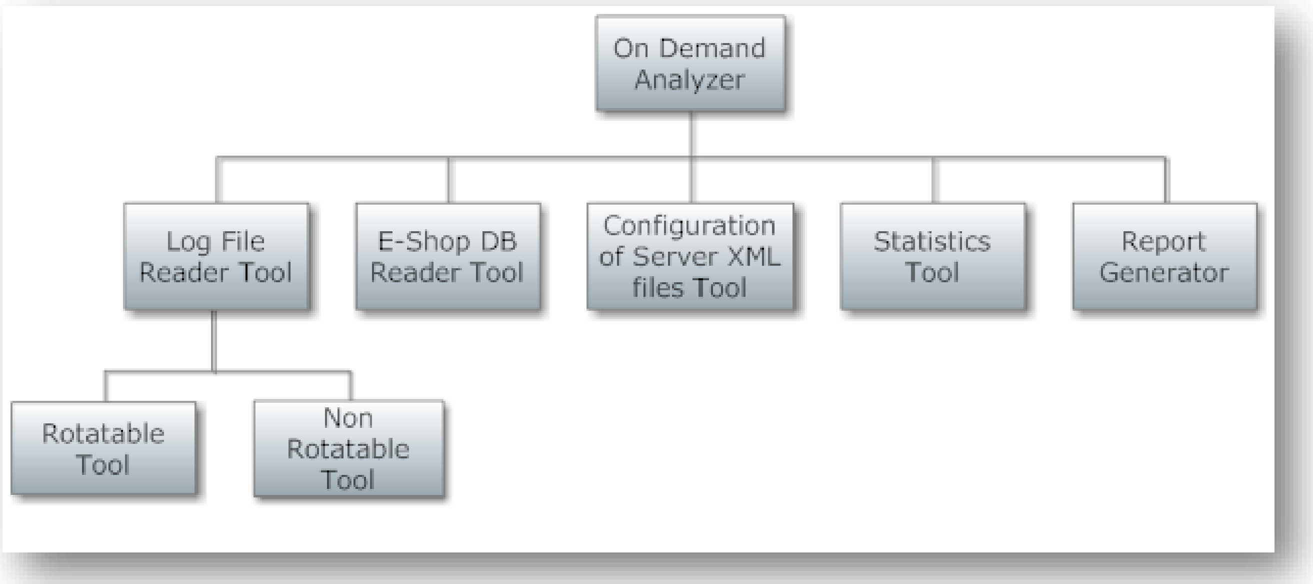


FIGURE 16: ON DEMAND ANALYZER MODULES

7.4. Real-Time Analyzer

Figure 17, shows the basic components of the Near Real Time Analyzer Web Application. They provide an alternative input source for the log file entries, plus a technique to receive business data and any additional event information, right after they are generated.

The log file entries are permanently queued as they are produced by the web server, by a daemon application, residing on the web server that hosts the e-shop. They are dequeued and accessed from the Real Time Analyzer Web Application. This application may be located anywhere. It is not necessary to run these two applications on the same database, but it would eliminate data traffic and would also provide fresh data to the On-Demand Analyzer (ODA).

Thus, the components used for input are:

- Data extractor from the operational e-shop database
- A log file queue processing tool, which dequeues the log file data received and parses it using the same algorithms as the ODA. The data ends up in the transaction database and becomes immediately available for the analyzer.
- Every metrics presentation tool of the RTA and every instrument selected to display measurements requires another data queue to operate. The information source originates from the e-shop database. Every time a sale is completed, or a user has logged into the system the generated information is pushed into the queue and the appropriate presentation tool is refreshed in near real time.

The processing is targeted on producing quick graphical information and results to the user. In addition to that, action is taken whenever a user logs in to the system for identifying the session statistics and alleviate anonymity. Demographic information of the user and her shopping habits and behavior before logging in become known, since the entire session corresponds to a known customer.

Output produced:

- Number of current users

- Real time turnover meter
- Throughput meter
- Statistical Information about origin of visitor, browser, operating system etc.

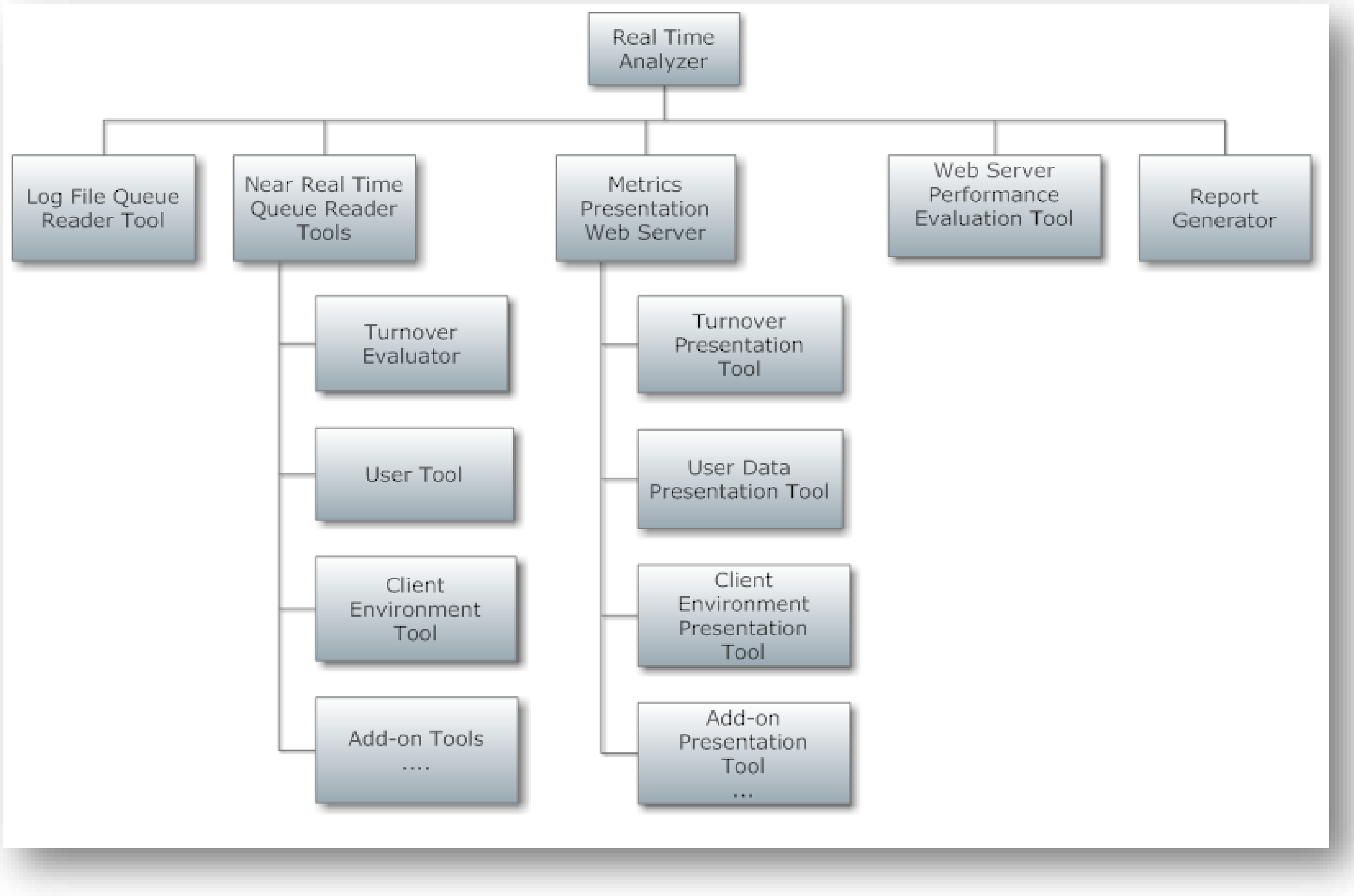

Real Time Analyzer
Log File Queue Reader Tool
Near Real Time Queue Reader Tools
Metrics Presentation Web Server
Web Server Performance Evaluation Tool
Report Generator
Turnover Evaluator
User Tool
Client Environment Tool
Add-on Tools ....
Turnover Presentation Tool
User Data Presentation Tool
Client Environment Presentation Tool
Add-on Presentation Tool ...

FIGURE 17: REAL TIME ANALYZER MODULES

Every presentation tool of the web server needs its queue sibling. In example, the turnover presentation is a tool that displays the turnover of the e-shop. Whenever orders are placed by customers the turnover presentation tool displays the total amount. To feed the graph with accurate data, two daemon processes are needed. A producer, running on the e-shop web server and its consumer running at the web server where the RTA is located.

Thus, the source of the incoming information comes from the log file, generated by the e-shop's web server and the source of the queue tools comes directly from the e-shop database. The output consists of a large set of graphical representations and operational and business statistics.

The application is easily extensible this way. To support any new metric, a new queue tool consisting of a producer and a new consumer provide data to the new presentation tool, which needs to be placed to the right place in the web application.

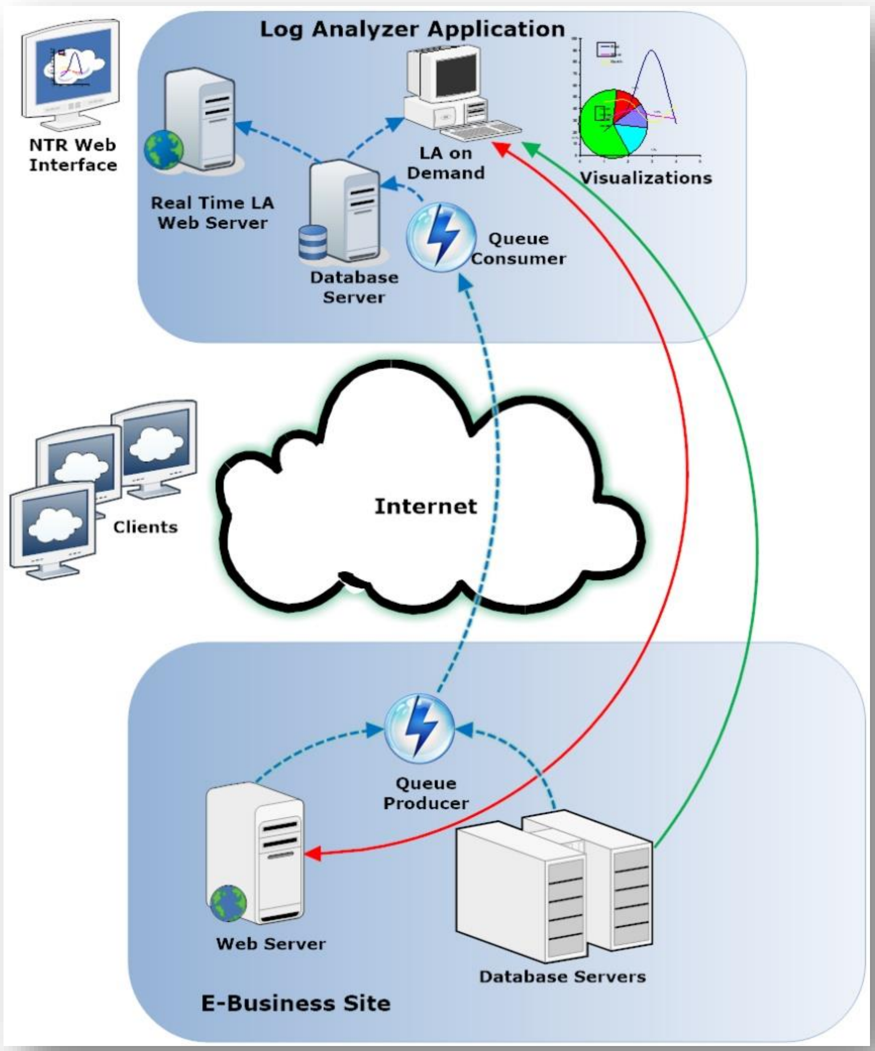


FIGURE 18: NEAR REAL TIME BIG DATA ANALYZER

The application to be scrutinized defines which type is best suited for this goal. To integrate, enhance and adapt the results and visualizations to every web application, the type 2 Analyzer extracts specific portions of their operational databases and embeds them into its own database. To cover the sparse log file of Rich Internet Applications, types 3 and 4 include streaming mechanisms that collect generated data in near real time. Additionally, collections of external sources of information, like in example social media data or even data from information systems running in parallel enhance the collected data and provide broader information and results, offering more versatile visualizations. Type 5 facilitates the appropriate mechanisms that support extraction and loading from various external sources.

The Analyzer Application, no matter what type, must be user friendly, simple and direct to operate, have a crisp unambiguous user interface and allow simple installation and adaptation to all necessary environments and platforms involved. Easily configurable ETL subsystems should collect data from various sources and built-in parametrization and configuration must make this adaptation simple and easy to apply.

7.5. Near Real Time With External Data

Modern developments often need to introduce Rich Internet Applications (RIA), in order to allow high quality desktop-like user interfaces to run on browsers and act as frontends to web applications. Since the main part of the interaction happens at the browser of the client, they produce reduced and less detailed log files. This renders all pure log file-based analyzer applications useless. This is because we have here higher amortization of the client browser's JavaScript interpreter. The communication between the client and the server becomes different and the interface of the application becomes very similar to what the user is used to deal with, in modern desktop application environments.

The following snippet is an extract from a Tomcat access log file, which shows the very sparse nature of the log file resulting from using rich Internet Applications:

```
…
… "GET /LogDB/ HTTP/1.1" 200 /LogDB/ 117 -
… "GET /LogDB/LogDB.css HTTP/1.1" 200 /LogDB/LogDB.css 19 -
… "GET /LogDB/com.art.logdb.LogDB/com.art.logdb.LogDB.nocache.js HTTP/1.1" 200 44 -
… "GET /LogDB/com.art.logdb.LogDB/1D630481E14BBD07DE7ED3D963A012CE.cache.html HTTP/1.1" 200 23 -
… "POST /LogDB/com.art.logdb.LogDB/DBActions HTTP/1.1" 200 450 -
… "POST /LogDB/com.art.logdb.LogDB/DBActions HTTP/1.1" 200 371 -
… "POST /LogDB/com.art.logdb.LogDB/DBActions HTTP/1.1" 200 449 -
… "POST /LogDB/com.art.logdb.LogDB/DBActions HTTP/1.1" 200 378 -
… "POST /LogDB/com.art.logdb.LogDB/DBActions HTTP/1.1" 200 885 -
…
```

EXTRACT FROM A GWT APPLICATIONS ACCESS LOG

The extract above shows the contents of a logfile, generated when the user logged into an RIA system, using an email and a password, after running a few search queries and transactions, none of which are visible in the logfile. The detailed step description, the familiar contents of the log file, is not generated anymore.

This log file lacks the entire detailed user interaction information and all the key-click activity, because they are handled by the JavaScript interpreter of the client browser.

The problem of filling the missing gaps of the sparse log file is not an issue with the tagging portion of the Analyzer hybrid system, since tags do not depend on the nature of the user interface of the e-shop to register detailed actions. The registration of the actions takes place by sending and collecting the tag for every web page request and refresh. Each tag sent is logged by the operator server into a traditional log file and log file-based analyzers used by the tag-based operator provide the metrics.

To keep the hybrid character of the Analyzer and make it capable of operating with RIA web applications, Near Real Time extensions (NRTE) were introduced.

Near real time extensions, solve not only the problems that are expected to occur whenever Rich Internet Applications substitute traditional ones, but at the same time, they provide metrics and information to the e-commerce site operating personnel in near real time. So, the benefit is double.

Near real time extensions can be applied both to traditionally designed web applications as well as RIA. Although traditional frameworks generate full logfiles, NRTE's can produce useful real time metrics. For RIA web applications they also fill the missing gaps among the log file entries.

Near real time extensions (NRTE) are used to produce "live" metrics of the operational aspects of the site under scrutiny. They rely on automated extraction of data from the site and automated loading into the database of the LFA. The data is processed upon arrival and the metrics are made available to the operating personnel of the site via a separate special web application. Metrics can include any aspect of the operation including the turnover of the e-shop, number of visitors analyzed by category, speed, geolocation etc. as well as the load of the system, throughput and responsiveness. The implementation is based on queuing systems and the architecture and distribution is completely flexible in terms of location of sensors (data producers) and location of the consumers. The overall system is designed in a way that is light to operate and does not require a lot of resources of the server-side [57].

The principle behind the NRTE implementation is based on the introduction of a set of daemon processes, each of which track occurring events of the e-shop operation, like for example a concluded purchase, or when some prospective customer adds an item to the cart, the appropriate data is collected by the producer daemon, enqueued in a named queue as a data package, and marked as sent. If, or whenever the client runs the corresponding consumer daemon, the data is dequeued, moved to the Analyzer database and finally visualized whenever needed, or used in any manner according to its content.

There may be a short delay of a few seconds or minutes if the consumer has not been running for a while because the enqueued datasets may be large, but the packages are designed to be small and this delay may seem irrelevant. The data sources operate fast and adding metrics for any aspect necessary becomes a simple task.

Implemented near real times metrics currently include:

- Overall speed bytes per second
- Number of active visitors
- Requested items per visitor and overall
- Orders completed
- Customer behavioral graph
- Number of logged in customers
- Turnover or profit per hour, day
- Internet bot counter

Whether the application under scrutiny is generating a traditional detailed full log file, or a sparse one, a specific daemon is used, that places every single line of this log file to a named queue, in addition to the metrics mentioned above, and so the information is sent to a consumer that parses it straight to the Analyzer transaction database. This can be considered as a third tool to load the parsed contents of the access log file to the analyzer. So far we had the rotatable and the non-rotatable access log file readers, presented in Figure 16, which operate on demand. The queued log file daemon transfers automatically the contents of the log file in near real time and stores the parsed data in the transactions database table.

Any arbitrary number of queues and their corresponding producer and consumer daemon application can be implemented to support the necessities and the requirements of every e-shop operator.

The latest evolution of the Analyzer Application, with most of its add-ons, is presented in Figure 19 and its Data Flow Diagram summary in Figure 26. The e-shop services accept requests from the customers clients. While the requests are registered to the log file the file is being constantly polled by the producer daemon. The data is queued and sent to the consumer daemon that parses and inserts it into the analyzer data base. The web server of the Analyzer application makes the data available to any authorized client through the World Wide Web.

The Turnover feed, for example, shown in Figure 19, is activated whenever an order is placed by a customer. An after-insert trigger is placed in the orders data base table. This trigger copies the order-id auto-number unique key to a new database table, which was added to the e-shop database. The table is polled periodically for any pending records by the queue producer.

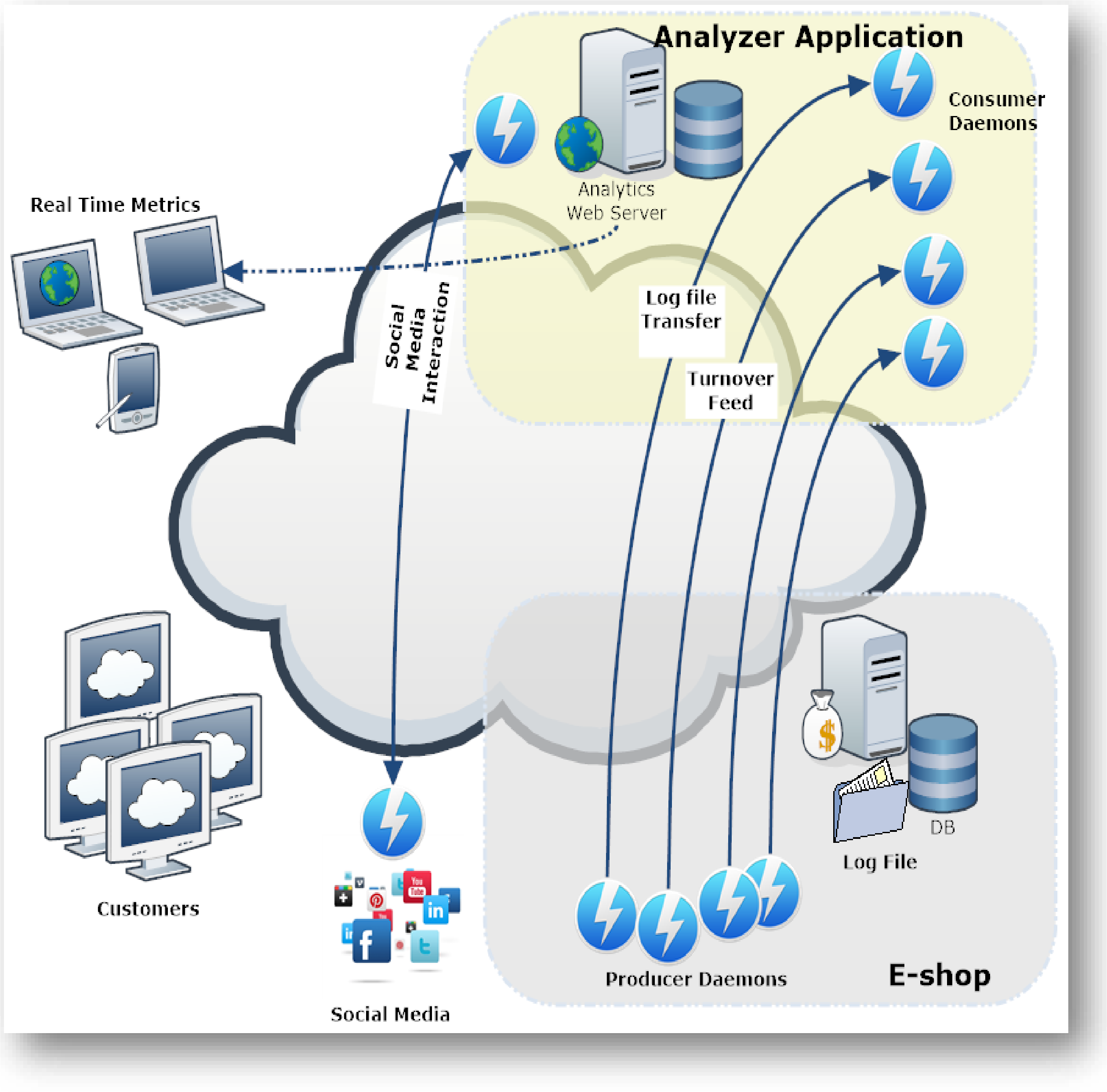


FIGURE 19: EXTERNAL DATA EXTENSIONS

If pending entries are located, based on the order number, all required additional information is gathered with the appropriate SQL left join commands with other e-shop database tables and appropriate log file data and the resulting data structure is enqueued also. The queue consumer at the LFA server-side dequeues the information and transforms and loads the data to the LFA database. From there the order data is made available to the web application of the Analytics server and is accessible to the Analyzer [58] .

The web application, running on the web server of the Analyzer, is GWT based and contains several visualization tools for every producer daemon, acting as a specialized sensor, transmitting information in near real time, but also general statistics and graphs generated by comparing historical data.

Near real time extensions (NRTE) are used to produce live metrics of the operational aspects of the site under scrutiny. They rely on automated extraction of data from the site and automated loading into the database of the LFA. The data is ready to be processed upon arrival and the metrics can be made available to the operating personnel of the site via a web application. Metrics include turnover of the e-shop, number of visitors analyzed by category, speed and load of the system etc. The implementation should be designed in a way that is light to operate and does not require a lot of resources of the server-side [57].

Additionally, a notification mechanism, can automatically inform the operating personnel of the e-shop with a visual message on a status panel, and/or via SMS, and/or email, when specific conditions, events or situations occur. If for example stock quantities drop or tend to drop below critical levels, if increased demand, or even if the throughput of the e-shop or web site starts to suffer from degradation due to high visitor demand or because the load of the system has passed a specified threshold.

Ease-of-installation and the need of a minimum of changes to the setup of the web application under observation are very critical factors to gain acceptance to install any general-purpose performance and timing measuring system or analyzer extension application. A log file analyzer, with near real time extensions system should be easy to embed into any web application or e-shop, without needing of extra source code modifications for its adaptation to the system and in order to operate smoothly. Source code changes have need to be maintained always when future releases and versions replace the current system.

As mentioned above, NRTE systems should be multifaceted, to can collect data on demand, for multiple performance metrics, as well as handle multiple types of events and situations. One aspect of the proposed system is presented in Figure 19.

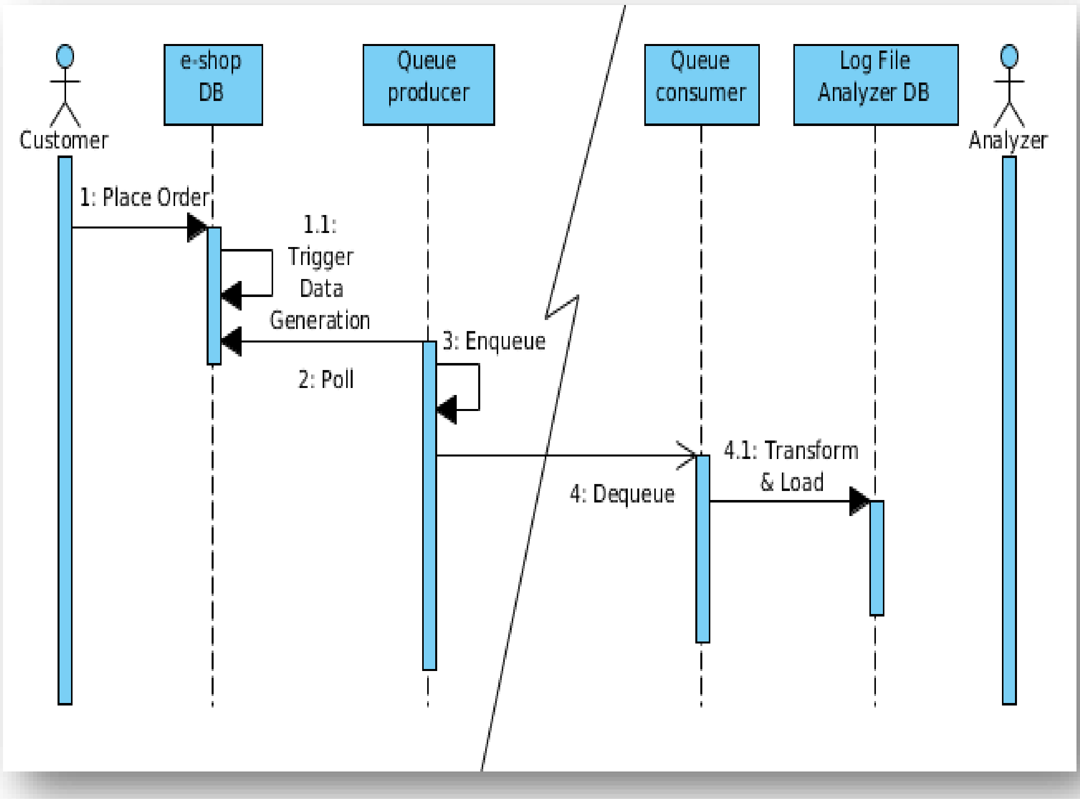


FIGURE 20: NRT ORDER PROCESSING UML

This following scenario takes place when the user places an order:

1. The placement of the order by the customer starts this cycle and is the event that triggers this process.

1.1. To automatically intercept the order data, an after-insert trigger is placed in the orders database table of the e-commerce site. This trigger copies the order-id auto number unique key to the new database table named “transmitted”, which has been added to the e-shop database. This new table could also reside in a separate database, in case the database administrators want to keep the e-shop database changes and additions to a minimum. The contents of the transmitted table are kept simple. They just include:

- An auto number as a primary key,
- The foreign key order number from the orders table and
- One bit, indicating that the transmission to the LFA is pending.

2. This table is polled periodically for any pending records by the queue producer daemon.

3. If pending entries are located, based on the order number, all required additional information is gathered with the appropriate left joins with other e-shop database tables and appropriate log file data and the resulting data structure is enqueued.

4. The queue consumer at the LFA server-side dequeues the information and

4.1. Transforms and loads the data to the LFA database. From this point on the order data is made available to the web application for presentation to the user.

The proposed approach must combine both log file analysis as well as tagging, with emphasis on the first technique, which makes it a hybrid system. Still several additional innovations must extend its functionality.

The measurements that will be presented and visualized are of both operational and business nature.

Operational measurements deal mainly with the operating environment and are mainly performance related. They provide the administrator with response times, speed, response behavior and load of the system.

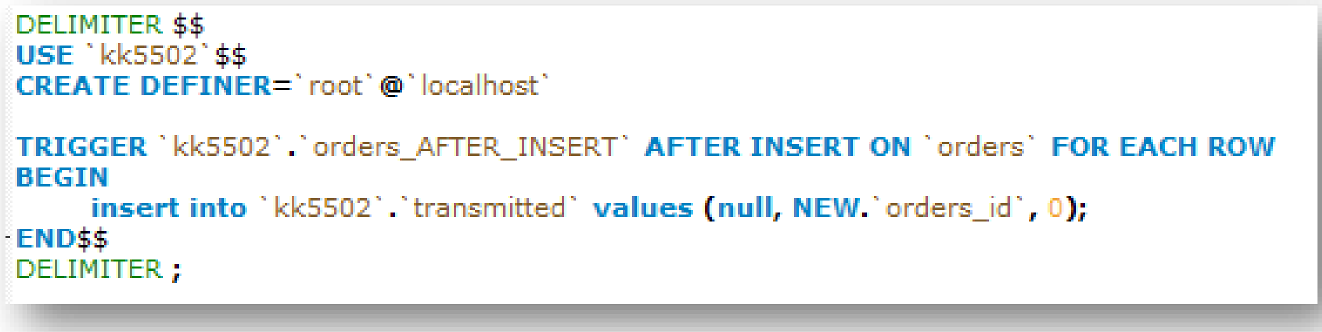
Business measurements provide information about sales, profitability and turnover measurements and data about origin and behavior of customers, as well as overall general business information and customer behavioral information.

The technology shift towards extensive Ajax and RIAs, which reduces the access log file size, is remedied by the NRTA. The missing log file user interaction information is substituted by customized data generated by queries that are triggered by the database, queued as messages and sent to the analyzer. The reduced log file although it still includes session id, IP-address and crucial details about the client does not contain all the clicks anymore. The database triggers can send appropriate details in order to substitute the click. The generated information is reduced, so filtering of unneeded data is no more necessary.

The data feeding mechanism that will provide the Analyzer with data generated from the e-commerce site in near real time can be based on messaging. According to Hohpe and Woolf [59], *Messaging* is a technology that enables high-speed, asynchronous, program-to-program communication with reliable delivery. Programs communicate by sending packets of data called *message*s to each other. *Channel*s, also known as queues, are logical pathways that connect the programs and convey messages. A channel behaves like a collection or array of messages, but one that is magically shared across multiple computers and can be used concurrently by multiple applications. A *sender* or *producer* is a program that sends a message by writing the message to a channel. A *receiver* or *consumer* is a program that receives a message by reading (and deleting) it from a channel.

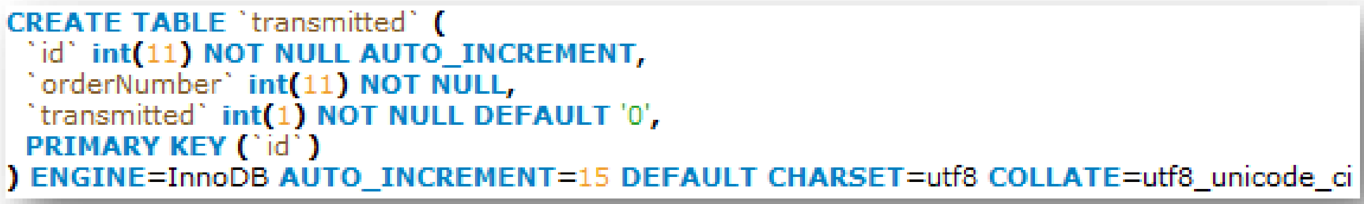
There are commercial and open source message queueing systems. Open-source systems are RabbitMQ, JBoss Messaging, and Apache ActiveMQ [27]. RabbitMQ is based on the Advanced Message Queuing Protocol (AMQP) which provides a low level, or "wire level" mechanism to communicate [26].

A simple MySQL example of a database trigger dealing with order processing is shown in Figure 21. This trigger runs automatically after every insert command on the orders table. The trigger generates the insertion of a row to the table transmitted, seen in Figure 22 below.



```
DELIMITER $$
USE `kk5502`$$
CREATE DEFINER=`root`@`localhost`

TRIGGER `kk5502`.`orders_AFTER_INSERT` AFTER INSERT ON `orders` FOR EACH ROW
BEGIN
	insert into `kk5502`.`transmitted` values (null, NEW.`orders_id`, 0);
END$$
DELIMITER ;
```

FIGURE 21: DATABASE TRIGGER SAMPLE



```
CREATE TABLE `transmitted` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `orderNumber` int(11) NOT NULL,
  `transmitted` int(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=15 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
```

FIGURE 22: DATABASE TABLE FOR NRT INFORMATION TRANSMISSION

A daemon process checks periodically the transmitted table for new entries. Once a new row is found, it is sent by the producer to the queue and their transmitted field is set to 1.

7.6. TYPE 1: STANDARD LOGFILE ANALYZER

The dataflow diagram of type 1 or "Standard" Logfile Analyzers, is shown in Figure 23. These applications are designed like typical ETL (extract-transform-load) systems. Their input is mainly only the access logfile of the web application. This logfile is a growing text file. Its contents are being extracted, line by line, parsed, and the extracted fields are stored into a Relational Database Management System. Data visualization applications generate graphics representations and provide the necessary metrics and statistical calculations that give insight to the administration and the operating personnel based on SQL queries.

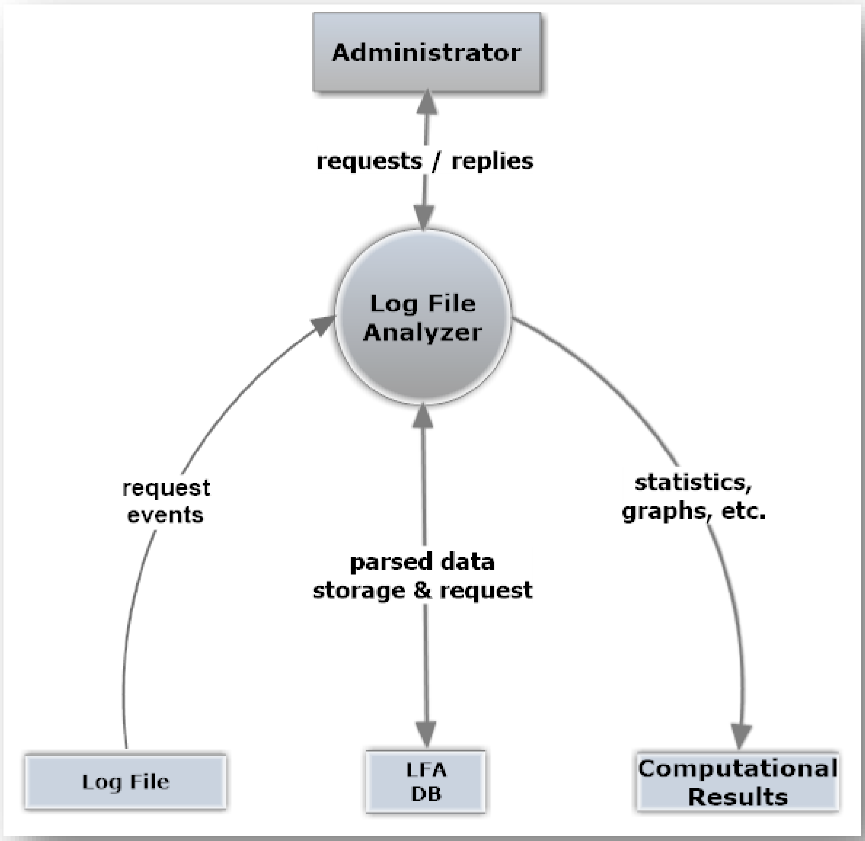


FIGURE 23: TYPE 1, TRADITIONAL LOG FILE ANALYZER

This architecture is simple, easy to apply to any web site and e-commerce site. These systems are straight forward to use, but their functionality is relatively limited because their input comes solely from the access logfile. The major shortcoming is the lack of the ability of cross correlating product and action descriptions. Hence, they force the user to work with the raw mode of data included in the logfile. The users must do their own decoding of the data and receive them in the same form as they appear in the log file of the e-commerce application and this is the way they are shown by the visualization system as well.

The main advantage of type 1 systems is the simplicity of their installation. They can directly be applied with very little preparation and can directly produce interesting metrics and information.

When a visitor for example interacts with the e-commerce site and performs the following transaction sequence:

(1) Chooses a category, (2) Chooses a subcategory and finally (3) Chooses a specific product of this subcategory, which is a very common customer activity sequence, the following entries are generated in the access-logfile. For readability purposes, only timestamp, HTTP request string and the returned status code are shown here.

The first click to the category refers to a category selection with category ID = 1:

```
[10/Oct/2018:23:11:02 +0300] "GET /konakart/SelectCat.do?catId=1 HTTP/1.1" 302
```

The demo e-commerce site is developed based on the Struts2 framework and when a selection is made, the status is set to 302 and a redirection enhances this requested data to include the manufacturer id, as well as the category title of the selected category. Here 'Hardware':

```
[10/Oct/2018:23:11:02 +0300] "GET /konakart/SelectCat.do?catId=1&prodsFound=-1&category=Hardware
HTTP/1.1" 200
```

We can depict from the redirected GET-string that the id 1 is used for Hardware. No products have been found, so the prodsFound parameter is -1, the reason for that is that we have not a specific product yet, but a list of products instead that are under category 1.

Next, the customer chooses subcategory 8, of the category Hardware.

```
[10/Oct/2018:23:11:09 +0300] "GET /konakart/SelectCat.do?catId=8 HTTP/1.1" 302
```

The redirect enhances the GET string to provide the category name:

```
[10/Oct/2018:23:11:09 +0300] "GET /konakart/SelectCat.do?catId=8&prodsFound=2&category=Keyboards
HTTP/1.1" 200
```

Now we see that category-Id 8 is "Keyboards" and the Keyboard query successfully retrieved 2 keyboard products.

The customer now sees two products, chooses product id = 25.

```
[10/Oct/2018:23:11:21 +0300] "GET /konakart/SelectProd.do?prodId=25 HTTP/1.1" 302
```

The final redirection now shows:

```
[10/Oct/2018:23:11:21 +0300] "GET
/konakart/SelectProd.do?prodId=25&manufacturer=Microsoft&category=Keyboards&name=Microsoft+Internet+Ke
yboard+PS%2F2&model=MSINTKB HTTP/1.1" 200
```

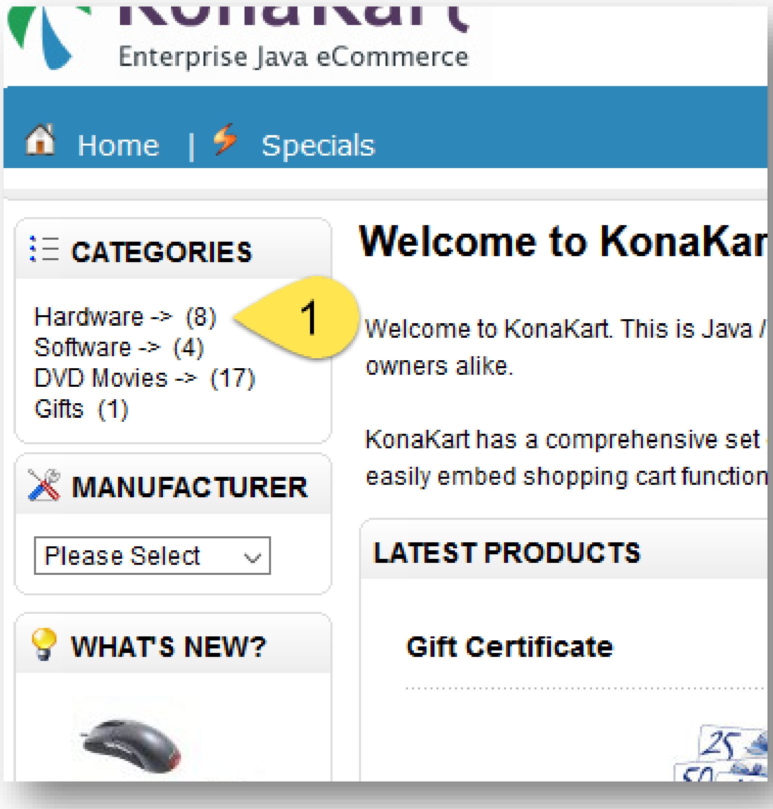
This redirection sends us to the product page. We depict the following additional information from the access logfile:

a. product ID 25 is the internal product code of the Microsoft Internet Keyboard PS/2

b. The model name is MSINTKB

c. The manufacturer of this product is Microsoft

These above steps look like this on the user's browser:



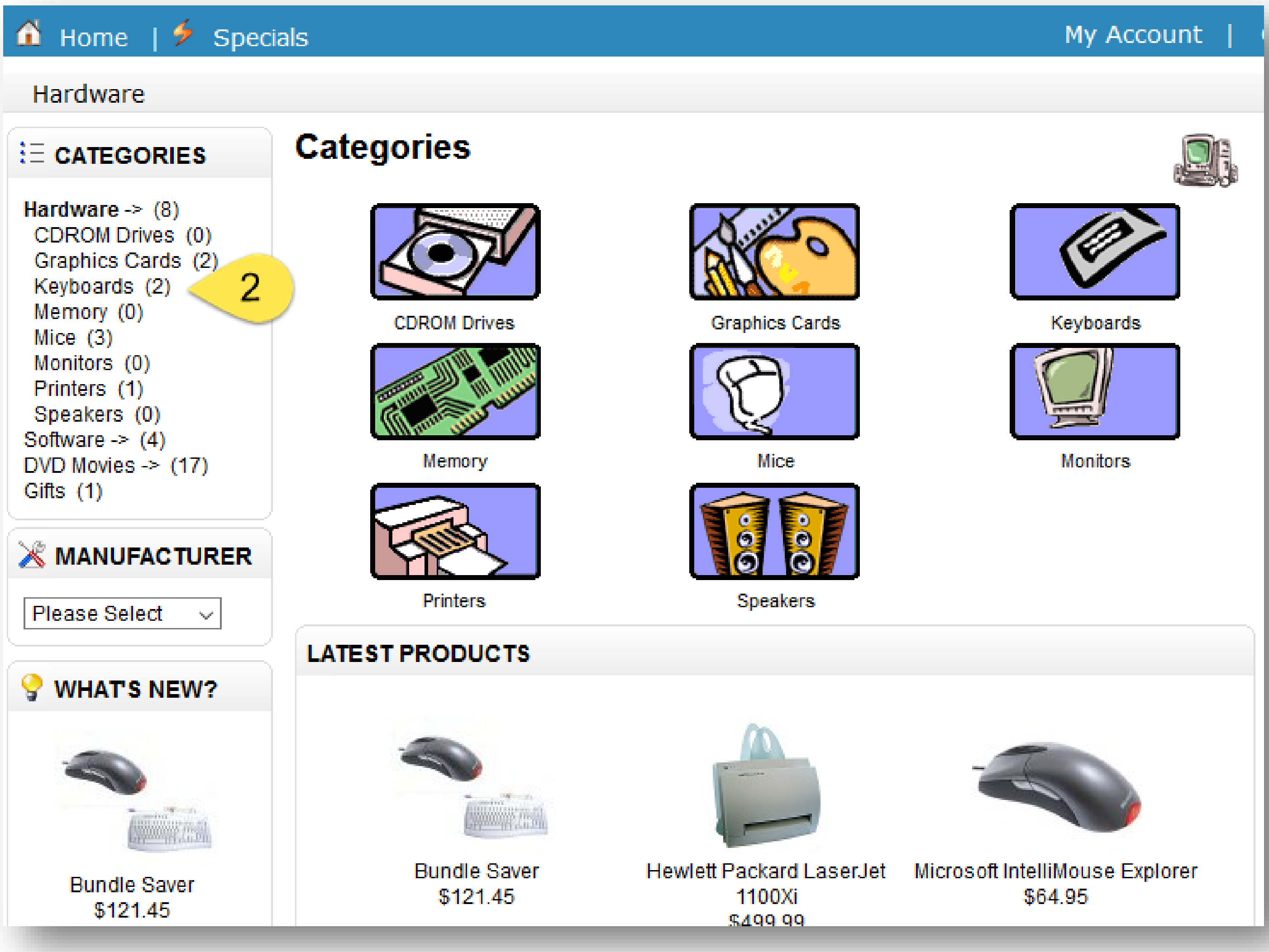
Welcome to KonaKar

Welcome to KonaKart. This is Java /
owners alike.

KonaKart has a comprehensive set
easily embed shopping cart function

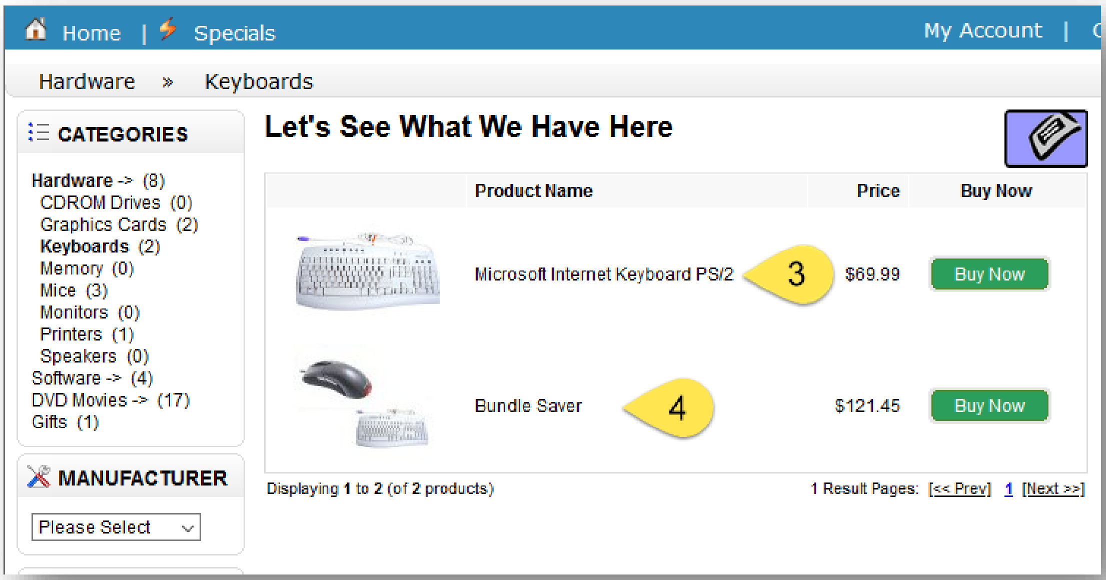
SCREEN 1: PRODUCT CATEGORIES

After selecting "Hardware":



SCREEN 2: HARDWARE CATEGORY

After selecting “Keyboards” the list with the two available keyboard models appears:



Hardware -> (8)
CDROM Drives (0)
Graphics Cards (2)
Keyboards (2)
Memory (0)
Mice (3)
Monitors (0)
Printers (1)
Speakers (0)
Software -> (4)
DVD Movies -> (17)
Gifts (1)

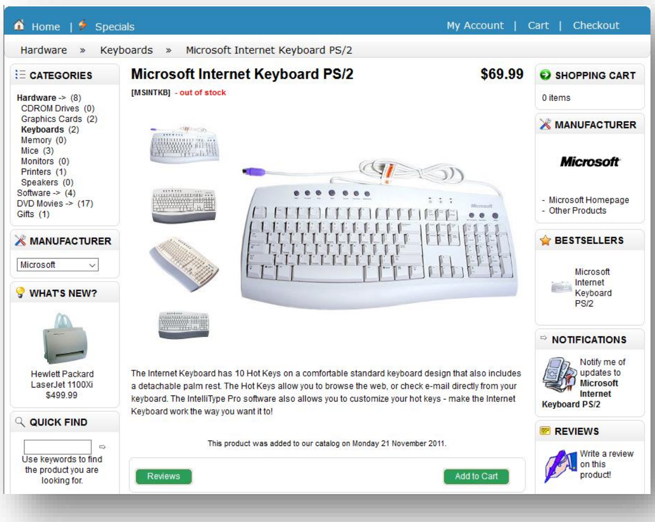
Let's See What We Have Here

	Product Name	Price	Buy Now
	Microsoft Internet Keyboard PS/2	$69.99	Buy Now
	Bundle Saver	$121.45	Buy Now

Displaying **1** to **2** (of **2** products) 1 Result Pages: [<< Prev] 1 [Next >>]

SCREEN 3: AFTER CHOOSING KEYBOARDS

And finally, after selecting Microsoft Internet Keyboard PS/2, the product page is loaded.



Hardware -> (8)
CDROM Drives (0)
Graphics Cards (2)
Keyboards (2)
Memory (0)
Mice (3)
Monitors (0)
Printers (1)
Speakers (0)
Software -> (4)
DVD Movies -> (17)
Gifts (1)

Hewlett Packard LaserJet 1100Xi $499.99

Use keywords to find the product you are looking for.

Microsoft Internet Keyboard PS/2 **$69.99**

[MSINTKB] - out of stock

The Internet Keyboard has 10 Hot Keys on a comfortable standard keyboard design that also includes a detachable palm rest. The Hot Keys allow you to browse the web, or check e-mail directly from your keyboard. The IntelliType Pro software also allows you to customize your hot keys - make the Internet Keyboard work the way you want it to!

This product was added to our catalog on Monday 21 November 2011.

0 items

Microsoft

- Microsoft Homepage
- Other Products

Microsoft Internet Keyboard PS/2

Notify me of updates to **Microsoft Internet Keyboard PS/2**

Write a review on this product!

SCREEN 4: AFTER CHOOSING 1ST OF THE 2 PRODUCTS

The above example shows that using the HTTP verbs GET, POST provides under circumstances the necessary information to reach a level of integration between the Analyzer and the E-commerce site.

7.7. Type 2: Extended Log File Analyzer

The Type 2 analyzer is tighter integrated with the Web Application or the E-commerce site it is operating on than type 1. Here it is not necessary to trace the encoding through the redirected HTTP GET request strings. The integration of operational data from the e-commerce site is direct. Data from the e-commerce site database is read into and transferred into the database of the Analyzer. Type 2 offers the following extra capabilities:

- It can provide direct **cross correlation** of the encoded information, collected from the requests registered in the log file, with product-item and group details, like descriptions, prices, customer details and so on. This feature makes all produced visualizations, reports and graphs more comprehensible for the store operators, without any need to refer to external references and code catalogs or redirections of the web application.
- **Data are loaded from the E-shop** data base straight into the File Analyzer data base, either on demand, or automatically, whenever unknown product groups and items appear in the log file. To avoid frequent massive inventory loads during daily operations of the e-shop, whenever new products or categories are inserted to the e-shop database, or price changes, each such insertion, update or deletion triggers also an update to the analyzer database as well, resulting to a self-updating system.
- Additionally, the LFA can enforce the **customization of the web application configuration** at the web server hosting it, to adapt the access log file information to the specific needs of every e-shop. The log file parsing mechanism also makes use of the configuration to extract information knowing the details of the access log valve.
- Analyzers of type 1 can only deal with items and categories that have been selected by visitors and customers, because these are the only ones whose code and description data appear between the lines of the access logfile. Type 2 systems know the entire category and product range.

The dataflow diagram of the type 2, extended LFA is shown in Figure 24. The system interacts with the E-commerce site database additionally to extracting the logfile and can even allow, if permissions are granted, to modify the web server XML configuration settings and stop and restart the web server itself to make these changes functional.

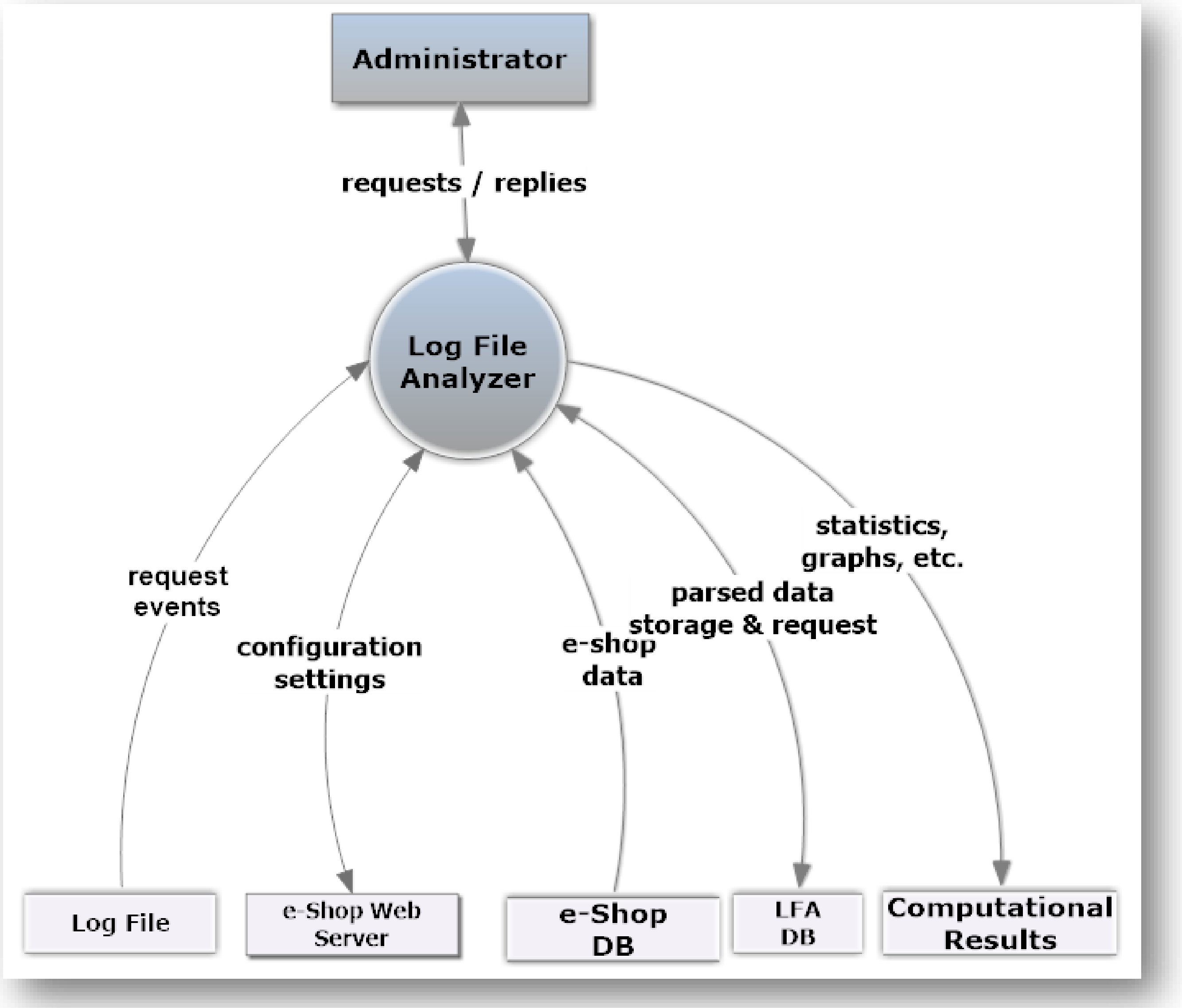


FIGURE 24: TYPE 2 LFA DATAFLOW DIAGRAM

7.8. Type 3: Hybrid Logfile Analyzer

Web traffic and visitor interaction is measured by:

- **Analysis of log files**
  Logfiles contain very detailed information about each request. The data must be carefully selected and
- **Page Tagging**
  Page Tagging requires an extra web server, to whom the visitor's browser is automatically sent. This server collects the log data generated by this visit and stores it to a specific data base for each site, based on an account number.

Logfile analysis is a precise methodology, but still has the following two disadvantages:

1. Proxy/Caching inaccuracies: If a page is cached, no record is logged on your web server.
2. No event tracking (JavaScript, Flash, Web 2.0): no JavaScript, Flash, Web 2.0 tracking

Tagging systems disadvantages are the advantages of Log File Analysis:

1. Firewalls can mangle or restrict tags
2. Cannot track bandwidth or completed downloads because tags are set when page/file is requested not when the download is complete.
3. Cannot track search engine spiders, since robots have the smarts to ignore page tags.

According to Brian Clifton [56], only a hybrid solution can provide a complete analysis of the web site visitor behavior. Because of their complexities, only a small number of vendors can offer a hosted hybrid solution.

Still at the end of the road, it is worth to go the extra mile and create a Hybrid System, since all disadvantages of the two methods can be eliminated and one system can back the other up in case of a technical problem.

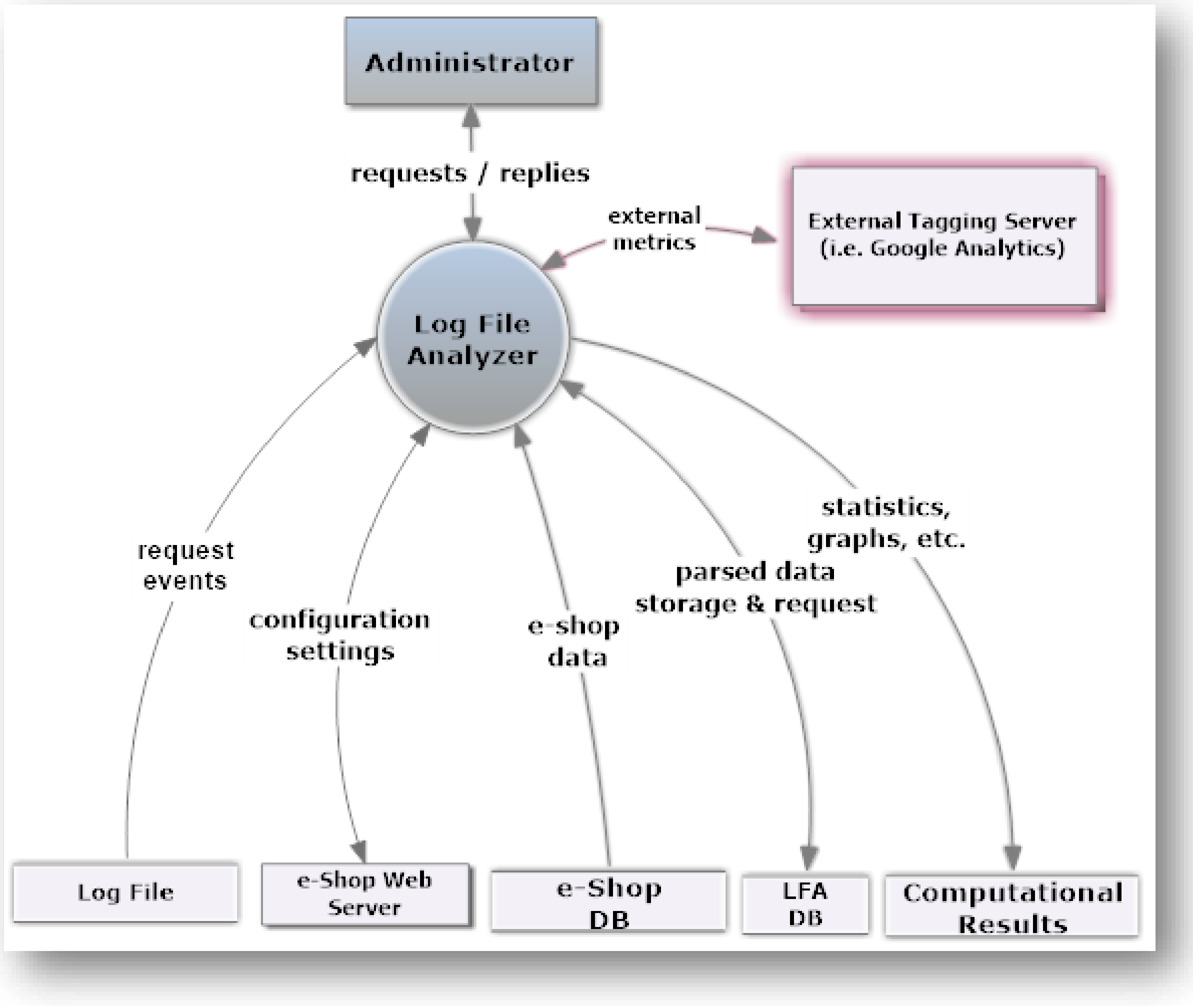


FIGURE 25: HYBRID LFA

Most E-commerce systems can easily be configured to use an external Page Tagging Analytics Application, like Google Analytics for example. A registration to the service and the addition of a small tagging JavaScript function snippet to all relevant HTML pages of the site to be measured, must be inserted and called by each web page that must be traced. These pages send a specific code-id to the Tagging System Operator. The Tagging System Operator uses logfile analysis to provide the measurements back to the site.

To support tagging our LFA makes use of the Google API (Figure 25) to get the information collected there, enhance the data of the log file and substitute to some extend data, that went missing through Proxies or Caching.

The log file and tagging back each other up and a comparison software mechanism is provided.

7.9. Type 4: Social Media Aware Analyzer

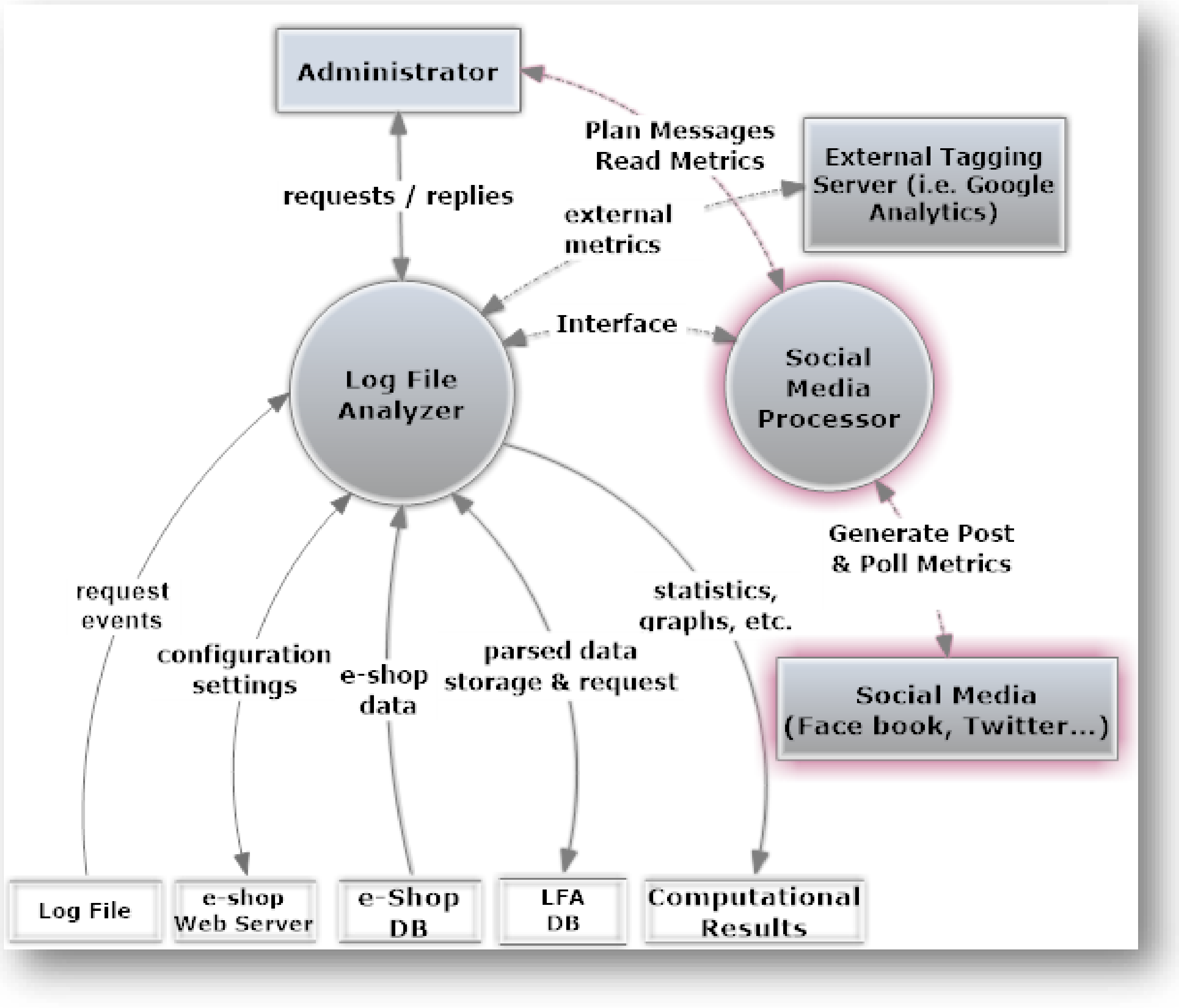


Figure 26: Social Media Processor Integration

The important "Social Media awareness" of the Analyzer is achieved through the Social Media Processor (SMP), which is a publisher application, embedded in the Analyzer (Figure 26), and automatically posts prewritten messages to selected social media applications (Facebook, Twitter, etc.), at specified times, either randomly, or following the list created by the administrators.

The Administrator can define what and where its content is posted. The SMP generates the postings, places them and regularly uses the Application Programming Interfaces to measure social feedback directly from the used applications.

The selected media are also regularly measured to evaluate their impact in terms of visits or sales through the access log file, as referrers.

Next chapter covers details and design characteristics of the Analyzer Application. It focuses on operational specifications, user interface design characteristics and details, functionality and implementation details of the on-demand log file loading, session processing and visualization of results. The streaming daemon functionality is presented and a description of the NRT log file loading process, that updates the database of the LFA in near real time. The Web Application Server and the database of the LFA are also presented.

8. ANALYZER APPLICATION DETAILS

Based on requirements defined in the previous chapter, the “Analyzer Application Details” section focuses on operational specs and details, the user interface design characteristics and details, functionality and implementation details of the on-demand log file loading, session processing and visualization of results. Streaming daemon functionality is presented and a description of the NRT log file loading process, that updates the database of the LFA in near real time. The Web Application Server and the database of the LFA are also presented here.

8.1. OPERATING ENVIRONMENT SPECIFICATIONS

A primary requirement for a software system is capability for simple installation, configuration and usage. Flexible support and quick modifications and customizing are also very important. All these goals can be achieved by developing in a widely accepted and portable multi-platform language. Porting parts or the entire application to any operational environment becomes easier without the need of maintaining multiple versions of the source code or owing and buying different hardware and operating platforms. Although the number of existing operating systems is in the range of a few hundred, standard and widely used operational environment platforms that dominate the computing systems today are reduced to mainly Unix and GNU-Linux flavors, mac-OS and Microsoft Windows. Of course, several additional Unix derivatives, like Raspbian for the Raspberry PI, Android and iOS operating systems used on mobile devices are being widely used as well and they happen to be very useful for supporting the necessary components of any modern analyzer. Since all the above environments support their individual Java Virtual Machines, it pays off to base the development on the Java programming language environment.

The analyzer application is written in 100% Java. The Java Development Kit is free, and so is the Java Runtime Environment. In addition to that, the language is object oriented, extremely versatile, very feature reach, well documented and supported. The fact that new releases of the language are under development by adding state of the art features offers the advantage of keeping the language alive for the foreseeable feature. The graphical development environment not only includes all necessary widgets and components but allows the programmer to easily customize and extend them as necessary.

The Analyzer is a complex software system, comprised of a desktop application for general management and visualizations, a variable number of headless processes running on local and remote servers and web-based interfaces for remote access and near real time information.

The complete On-Demand and Near Real Time Extended Hybrid and Social Media application is developed with NetBeans in Java 1.8, using Swing for the desktop graphical user interface. The visualization of the metrics and results is based on Jasper Reports. The reports were designed with the TIBCO JasperSoft Studio. The DBMS used is MySQL Community Server version 5.7.23.

The Web Application components were developed with GWT and the Eclipse Helios integrated development environment.

The producer and consumer daemons were also implemented in Java, based on RabbitMQ, which is written in Erlang and sits on top of an AMQP implementation.

8.2. Application Perspective

The basic characteristic that will assure perspective of the life and acceptance of any application running in a fuzzy, permanently changing and challenging environment like the one we face in the Analytics arena, is design and architecture that facilitates adaptability. Modular design allows ease of modifications and functionality additions. As the capabilities, functions and scope of the underlying software substructure used, including Operating Systems and Web Servers improve the Analytics Application must be able to adapt to these improvements, so that the application can face the constant evolution and collect and process data from all new popular data sources and applications, leading to better results, conclusions and information.

It is not only the constellation of popular applications and their usage mix that changes constantly. Also, the details of the underlying application foundations evolve as well. A simple example is the access log file valve of the web server, which evolves as well. The evolution is slight but decisive and every new release of the Apache Tomcat Web Server provides a richer set of details. To profit from the additional information, it is always of benefit to upgrade to the latest version and use the newest features by extending its pattern so that it generates the new tokens to the log file. This change leads to a modified regular expression that will be needed for the parsing of the new data and additional attributes of the database table that will store the additional information. Because of this change, the GUI and data handling programs and the visualization applications will need to be adapted. In a sanely designed modular Analyzer application these changes can be performed within a few hours and without risking data loss and erratic behavior. This is achieved partly because of the simplicity of the software architecture and partly due to well selected development tools that make the entire work-flow frictionless.

8.3. User Interface

The user interface of the application is as intuitive as can be, designed and implemented according to contemporary standards. The Swing package is the base foundation. The UML class diagram in Figure 27 shows the basic component classes and interfaces that are inherited in each dialog [60].

The dialog Sites implements the interface Record. It has one SitesToolBar which is a subclass of the ToolBarPanel, a subclass of the JPanel containing a JToolBar and having 13 JButtons that allow insertion, deletion, updating and navigation and search. This is the design of almost all Dialogs used to process data.

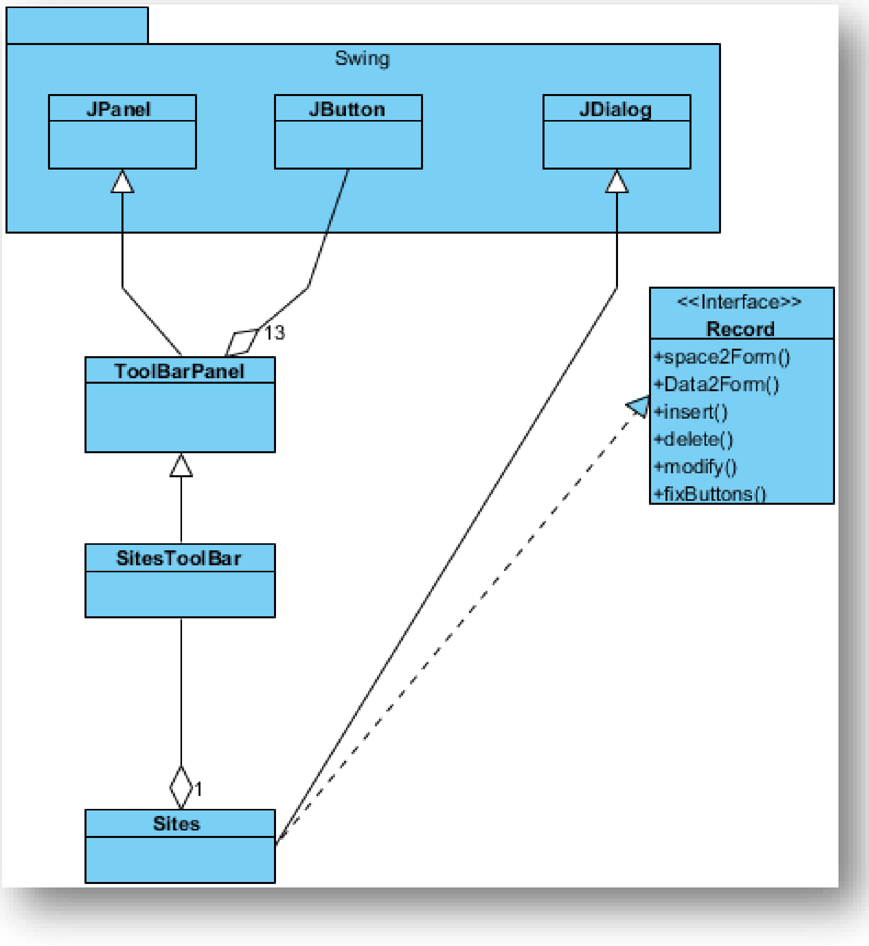
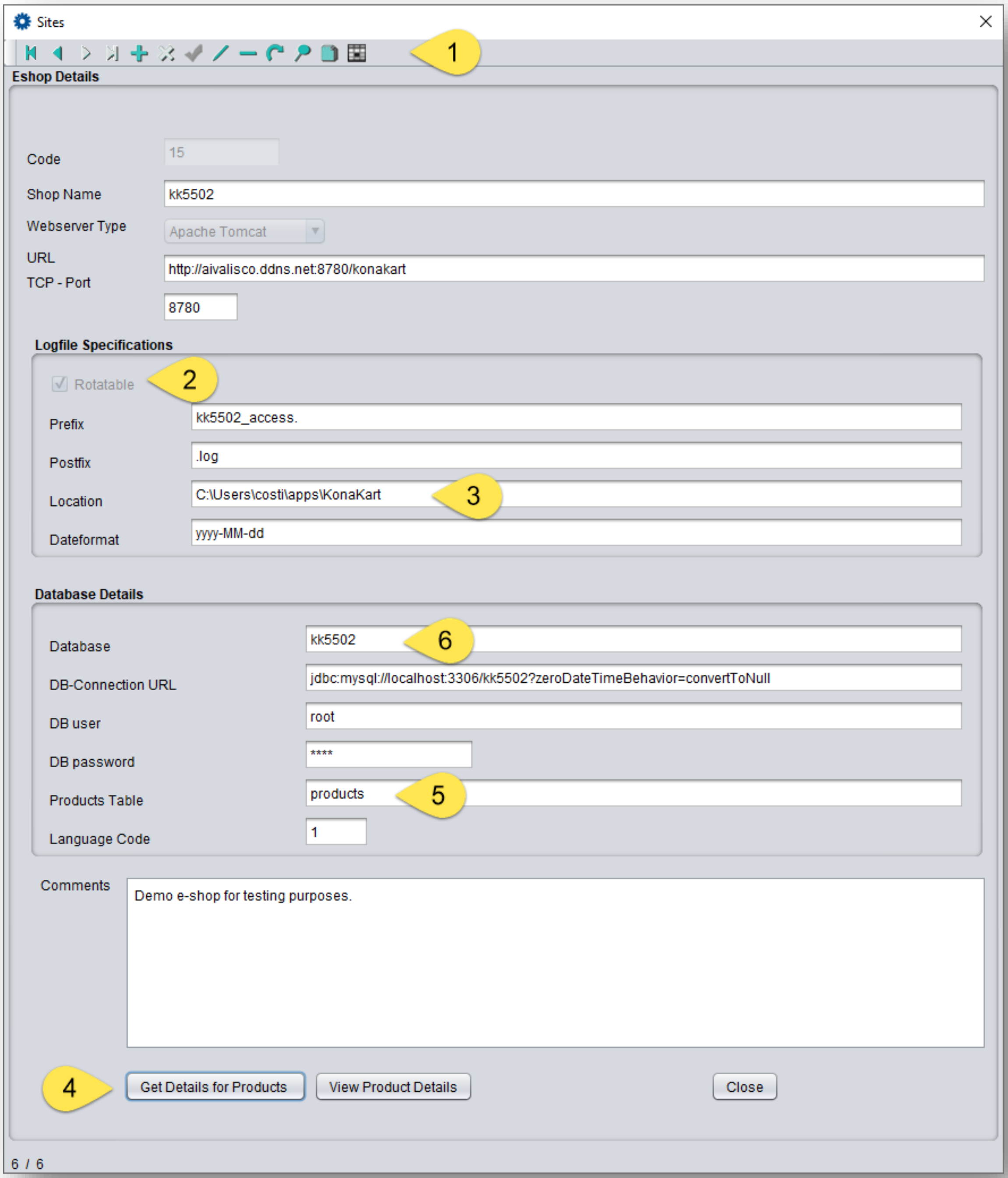


FIGURE 27: CLASS DIAGRAM FOR DIALOGS

The components of the class diagram above are used in most dialog windows. A sample dialog and the main components can be seen in Figure 28:

1. The Toolbar with navigational and processing buttons: First, Previous, Next, Last, Add, Cancel, OK, Modify, Reload, Find, Duplicate, Show all in a table.
2. Checkbox for defining whether the site has a rotatable logfile type or not.
3. Location of the applications, necessary for finding the configurations XML files and the log files.
4. Selection of products basic information for Type 2 Analyzers, used for loading the products of the e-shop to the analyzer database.
5. The database table name where the products are.
6. Name of the e-shop database
7. Connection String



Eshop Details

Code	15
Shop Name	kk5502
Webserver Type	Apache Tomcat
URL	http://aivalisco.ddns.net:8780/konakart
TCP - Port	8780

Logfile Specifications

Rotatable	✓
Prefix	kk5502_access.
Postfix	.log
Location	C:\Users\costi\apps\KonaKart
Dateformat	yyyy-MM-dd

Database Details

Database	kk5502
DB-Connection URL	jdbc:mysql://localhost:3306/kk5502?zeroDateTimeBehavior=convertToNull
DB user	root
DB password	****
Products Table	products
Language Code	1

Comments: Demo e-shop for testing purposes.

FIGURE 28: SITES DIALOG

8.4. Functionality

The Analyzer application is completely independent of the location of any e-commerce web site it scrutinizes. It must be able to process data and log files from multiple web sites, running on multiple servers and often having remote database servers running on different machines. The Analyzer must facilitate all these various locations and possess the ability to download the needed log files from the server to the local disk to process them and extract and load the data into the LFA DBMS.

Either the web server and application operational characteristics file, located for example if the server is Apache Tomcat, in the **conf/server.xml** file must be accessible to the web server, or the contents of this file should be known to LFA and stored in the appropriate database. On top of that, if the operators of the e-commerce site to be evaluated allow by giving the Analyzer privileges of modifying the server.xml file and the ability of restarting the web server, this would ease the entire procedure of setting up and configuring the information generation and extraction by eliminating the need of communication among administrators and by automating these modifications.

8.5. Main Frame and Main Menu

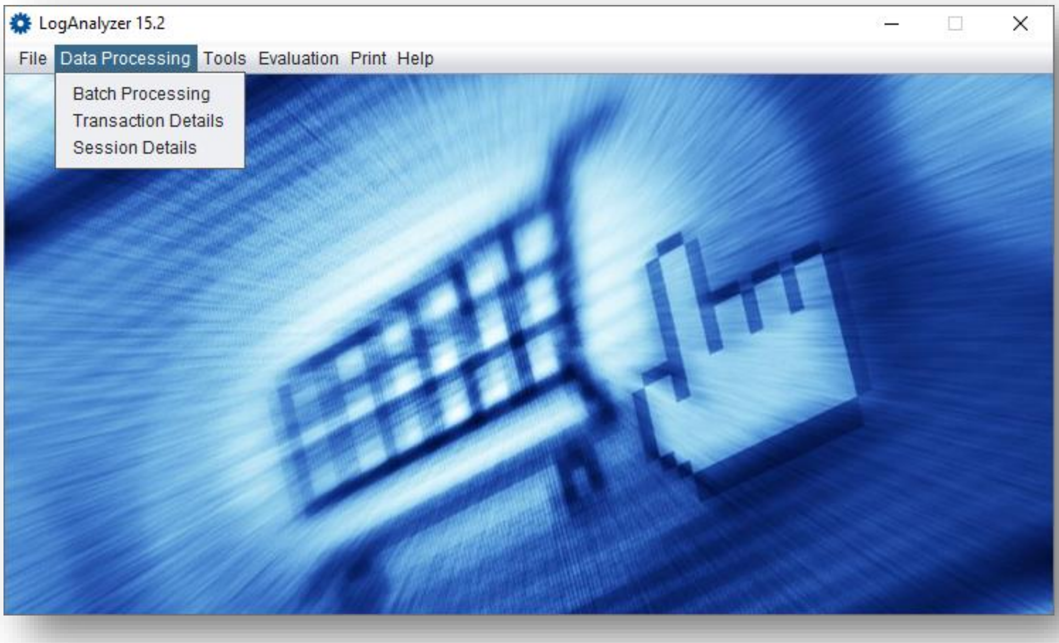


Figure 29: Main Frame of The LFA

The main menu is used for navigating to the components described above. The menu-bar contains a File menu, a Data Processing Menu, a Tools Menu that is used for setting up the environment of the applications, the Evaluation menu is used for data visualizations and statistics. The Print selection generates reports.

The basic components are:

1. A Log file Reader for loading log files:
    - Non-Rotatable Tool and
    - Rotatable Tool
2. E-Shop data Reader
3. Visualization Tools

8.6. Logfile Loading

The primary function for any Analyzer application is loading data from logfiles. The first node of the hierarchical functional diagram of Figure 16, is dedicated to logfile loading. Logfile analyzers and visualization systems need data.

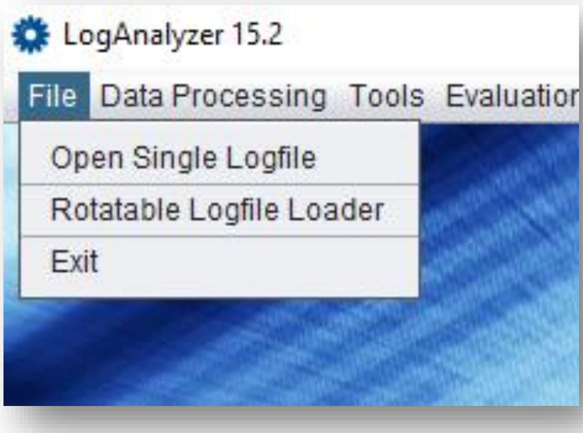


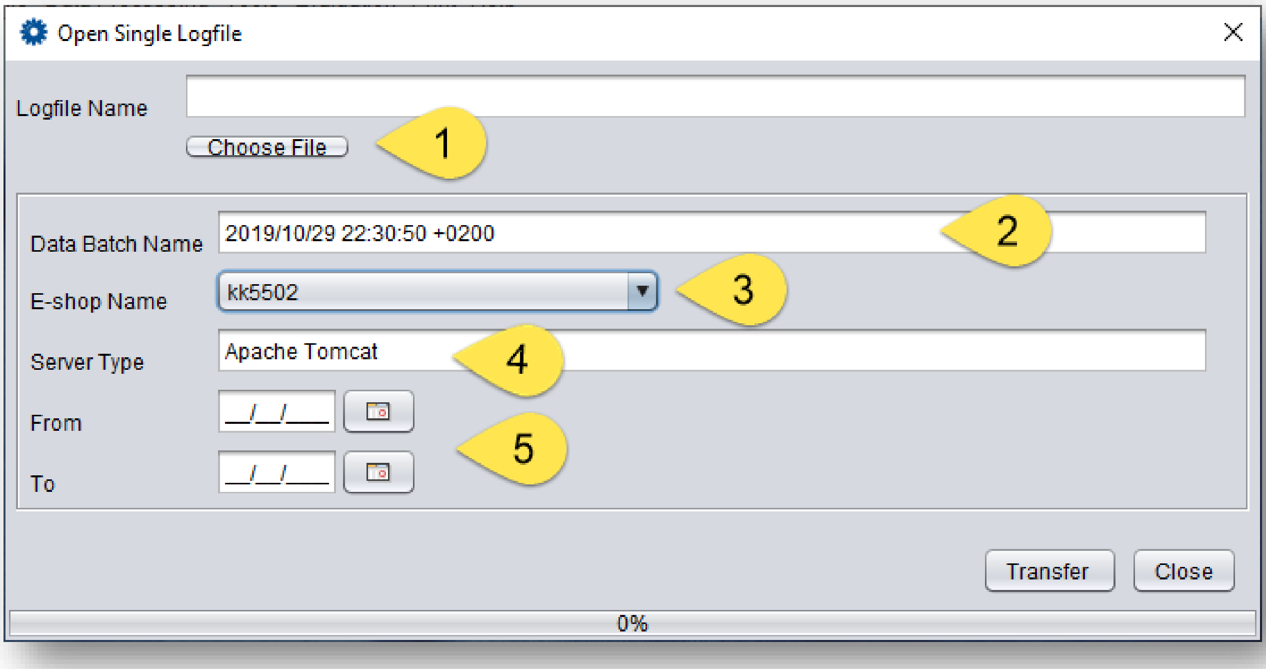
FIGURE 30: LOGFILE LOADERS

Beside the two loaders described in the next two sections, we will use a more convenient automated method of dealing with data, which is using data streaming. Data streaming allows data to arrive to the LFA within fractions of seconds after their generation.

8.7. Single Logfile Loader

For opening and processing a single logfile, a file chooser window widget is opened that allows defining the location of the log file in the file system (Figure 31):

1. Open file-chooser dialog, so the user can select the single file.
2. The data batch name can be entered here. The batch name is used in order to group all the lines of this run in the analyzer database together. The default value generated by the system is the timestamp during selection. This value can be modified or enhanced to something more characteristic.
3. This combo box relates the read data to the e-commerce site. The logfile must be correlated to a specific pre-configured e-shop, since the application can deal with multiple e-shops and sites at the same size.
4. The web server type.
5. It is possible to define date ranges by choosing from- and/or to-dates. This selection allows reducing the size of the batch to be loaded. The feature is necessary for reducing lines in huge multi-day non-rotatable logfiles.


Open Single Logfile

Logfile Name

Data Batch Name: 2019/10/29 22:30:50 +0200

E-shop Name: kk5502

Server Type: Apache Tomcat

From: __/__/____

To: __/__/____

0%

FIGURE 31: SINGLE LOG FILE LOADER

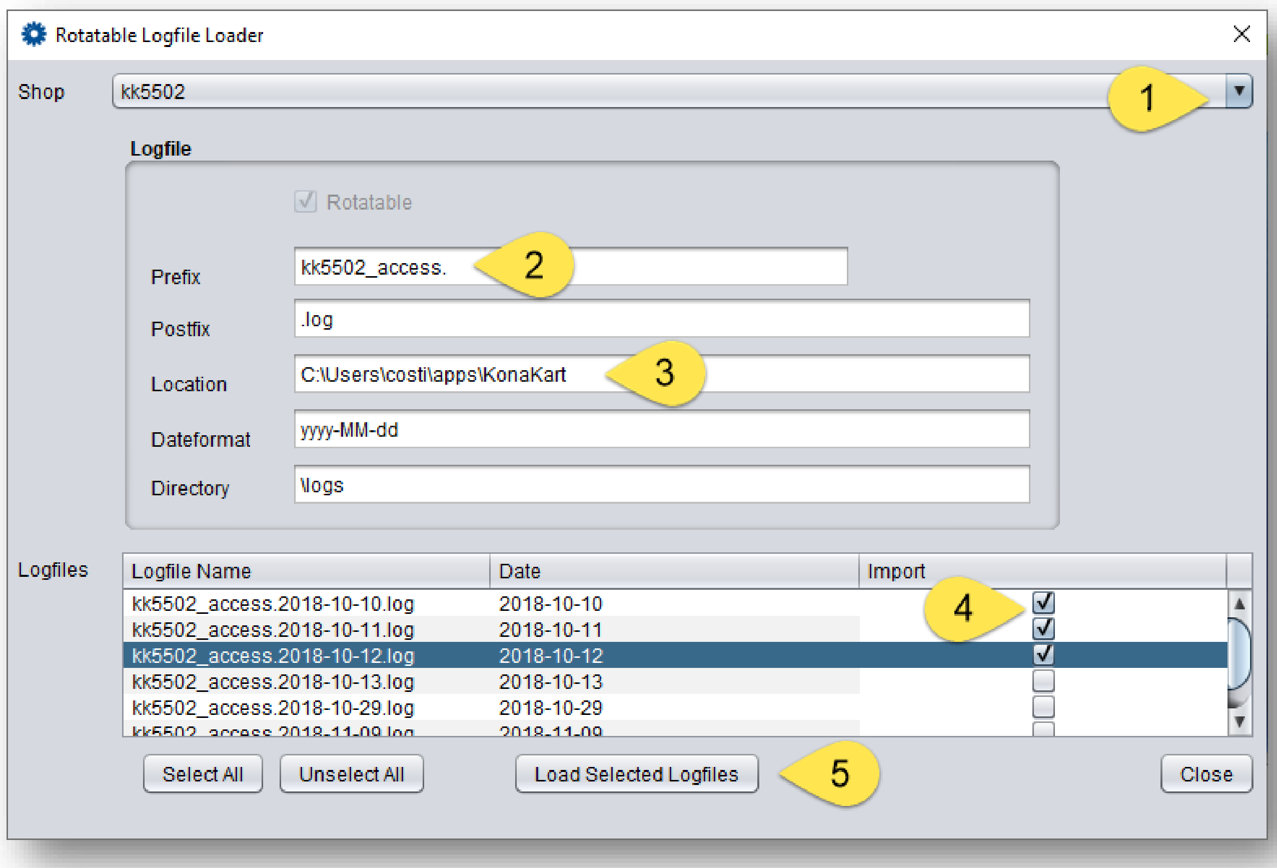
Once the transfer button is pressed the logfile lines are parsed and loaded to the database. The set of lines is at a later phase traceable by the batch name, either the timestamp which is the default name generated by the system or some string.

Any batch can be further processed by the Batch Analyzer in Figure 33.

1. Batch chooser
2. Number of logfile lines
3. Bots may be processed

8.8. ROTATABLE MULTI LOGFILE LOAD

The system allows multiple logfiles to load in one session. Since this procedure is started on demand the data drives the algorithm automatically to the right directory for each e-commerce site, displays a list of all unloaded logfiles and allows to import the ones that must be loaded (Figure 32).



Shop: kk5502

Logfile

- [✓] Rotatable
- Prefix: kk5502_access.
- Postfix: .log
- Location: C:\Users\costi\apps\KonaKart
- Dateformat: yyyy-MM-dd
- Directory: \logs

Logfiles

Logfile Name	Date	Import
kk5502_access.2018-10-10.log	2018-10-10	✓
kk5502_access.2018-10-11.log	2018-10-11	✓
kk5502_access.2018-10-12.log	2018-10-12	✓
kk5502_access.2018-10-13.log	2018-10-13	☐
kk5502_access.2018-10-29.log	2018-10-29	☐
kk5502_access.2018-11-09.log	2018-11-09	☐

FIGURE 32: ROTATABLE LOGFILE LOADER

1. The combo box allows selecting the appropriate e-shop among the sites supported.
2. Displays the prefix, the date and the postfix of the logfile is known.
3. The location is defined in the site-settings.
4. The so far unloaded log files are displayed in the table. The user can check which ones to load.
5. Loading begins after pressing the “Load Selected Logfiles” button.

8.9. LOGFILE LOADING PROCESS

As seen above so far, the logfile can be either non-rotatable or rotatable. Non-rotatable files can become very large since they contain data from many days of operation. Rotatable files are usually smaller but many. Loading of log file content is activated only on demand. The “Open Single Logfile” loader selection of the “File” menu can as well be used in order to generate batches of rotatable file contents if necessary, store them to the applications database and make them available for inspection and visualizations.

The contents of the access logfile depend on the settings used in the configuration file. In case of an Apache Tomcat web server this is the “config/server.xml” file. Our sample application is served by Tomcat, using the following pattern:

```
'%a %A %b %B %h %l %m %p %q %u %t "%r" %s %U %D %S "%{User-Agent}i" "%{Referer}i"'
```

Using the pattern above leads to the form of the logfile, as the one seen in Text Box 2. It is generally better to avoid frequent changes of the configuration pattern, especially after the web application system is in production. Changes to the pattern will subsequently generate changes to logfile and will need alterations of the database and the software in order to allow maintaining consistency with previous data. Another good practice would be to add columns for all possible options to the database table, even for ones that are not used and allow Null values for them. This way they can be added at a later phase, leaving the schema of this large database table unmodified.

The loading procedure is basically designed around a while not-end-of-file statement, reading the regular semi structured log file, which a text file, from its specific location, that has a specified name and a format defined by the pattern. Each line must be parsed individually, and all the collected fields must be prepared for storing to a database row one after the other.

There are two ways of parsing the details:

1. Using String methods
2. Using Regular Expressions

Both class libraries are included in the standard Java development kit. Using regular expressions is a more elegant method, because the whole complexity is resolved by applying just a single expression. The data components are of variable size and some of them include an undefined number of white spaces. The following regular expression pattern allows parsing the log file with pattern applied for our paradigm:

```
^([\d.]+|[\d:]+) (\S+) (\S+) (\S+) (\S+) (\S+) (\S+) ([\d]+) [a-zA-Z0-9_
]*(\S+) [-]?[ ]?\[([\w:/]+\s[+\-]\d{4})\] \"(.+?)\" (\d{3}) (\S+) ([\d]+)
(\S+) "(.+?)\" "(.+?)\"
```

Each pattern generates a different type of information in each line and thus needs a specific regular expression in order to match and extract the groups of data.

As an example, the following line is parsed into the following groups:

```
94.66.56.170 127.0.1.1 23730 23730 94.66.56.170 - GET 8780
?catId=2&prodsFound=-1&category=Software - [24/Oct/2019:19:16:54 +0300] "GET
/konakart/SelectCat.do?catId=2&prodsFound=-1&category=Software HTTP/1.1" 200
/konakart/SelectCat.do 35 CE9496EB17C260F5AD552FBF13CB2771 "Mozilla/5.0 (X11;
Ubuntu; Linux x86_64; rv:69.0) Gecko/20100101 Firefox/69.0"
"http://datalab.stef.teicrete.gr:8780/konakart/SelectCat.do?catId=1&prodsFound
=-1&category=Hardware"
```

This line starts at byte 36628 in the logfile and is 451 bytes long.

The parser splits the line into the following 17 groups:

Group 1:	**94.66.56.170**	(12 characters)
Group 2:	**127.0.1.1**	(9 characters)
Group 3:	**23730**	(5 characters)
Group 4:	**23730**	(5 characters)
Group 5:	**94.66.56.170**	(12 characters)
Group 6:	**-**	(1 character)
Group 7:	**GET**	(3 characters)
Group 8:	**8780**	(4 characters)
Group 9:	**?catId=2&prodsFound=-1&category=Software**	(40 characters)
Group 10:	**24/Oct/2019:19:16:54 +0300**	(26 characters)
Group 11:	**GET /konakart/SelectCat.do?catId=2&prodsFound=-1&category=Software HTTP/1.1**	(75 characters)

Group 12:	**200**	(3 characters)
Group 13:	**/konakart/SelectCat.do**	(22 characters)
Group 14:	**35**	(2 characters)
Group 15:	**CE9496EB17C260F5AD552FBF13CB2771**	(32 characters)
Group 16:	**Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:69.0) Gecko/20100101 Firefox/69.0**	(76 characters)
Group 17:	**http://datalab.stef.teicrete.gr:8780/konakart/SelectCat.do?catId=1&prodsFound=-1&category=Hardware**	(98 characters)

The extracted data, except the date in group 10, is completely ready for storage to the database.

The first group, **`^([\d.]+|[\d:]+)`** deals with both IPv4 and IPv6 addresses.

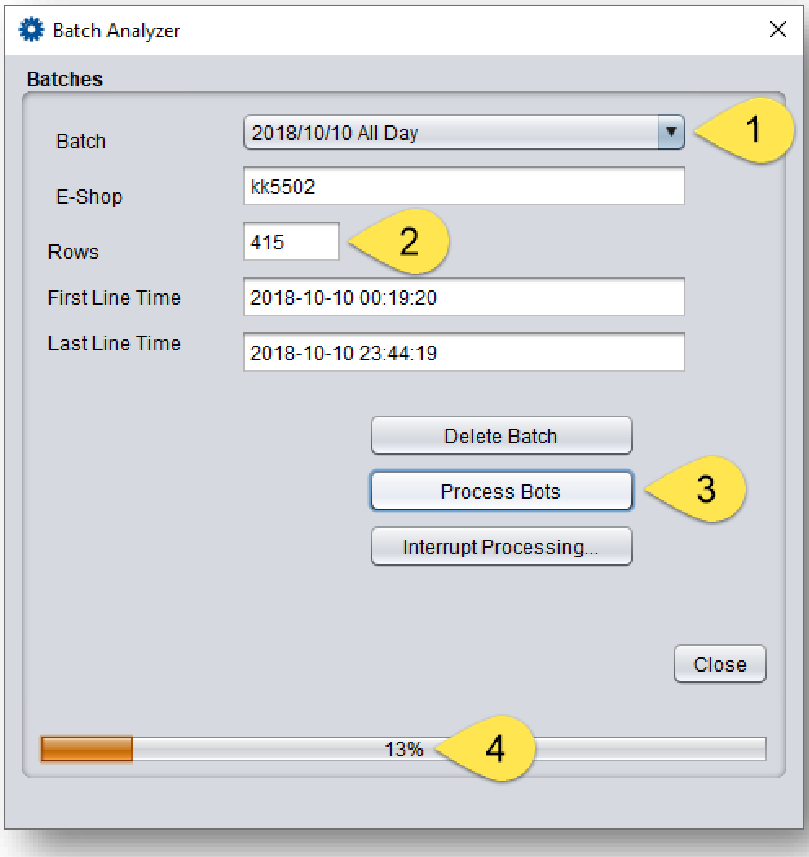
Following code snippet shows how regular Expressions are used in Java:

```
        String logEntryPattern = "^([\\d.]+|[\\d:]+) (\\S+) (\\S+) (\\S+) (\\S+)
(\\S+) (\\S+) ([\\d]+) [a-zA-Z0-9_ ]*(\\S+) [-]?[ ]?\\[([\\w:/]+\\s[+\\-]\\d{4})\\]
\\\"(.+?)\\\" (\\d{3}) (\\S+) ([\\d]+) (\\S+) \"(.+?)\\\" \"(.+?)\\\"";

        Pattern p = Pattern.compile(logEntryPattern);

        try {

            myFile = new FileReader(LOGFILE_NAME);
            buff = new BufferedReader(myFile);

            Matcher matcher;

            while (true) {

                String line = buff.readLine();

                if (line == null)
                    break;

                matcher = p.matcher(line);

                if (!matcher.matches()) {
                    System.err.println(line + matcher.toString());
                    return;
                }

                try {
                    rs.moveToInsertRow();

                    rs.updateString("aRemoteIPAddress", matcher.group(1));
                    rs.updateString("ALocalIPAddress", matcher.group(2));
                    …
                    rs.updateString("SUserSessionID", matcher.group(15));
                    …
                    rs.insertRow();

                } catch (SQLException e) {
                    …
                }
```

The regular expression is compiled only once, before use. After the compile we have a Pattern object p. Then the pattern is checked for matches for each line. If the pattern matches, it returns 18 groups of data. Group (0) includes the entire unparsed line. In this example, group(15) returns the user session-ID number, returned by the %S code of the pattern.

8.10. Batch Processing

The uploaded lines loaded to the LFA tables are stored in “batches”. Each batch has a name. The default batch name is a date-timestamp proposed by the system at load time. The user is free to rename the batch. Batches are used in order to group log file entries together and in order to split the relatively high number of rows into more manageable chunks of data. Figure 33 shows such a batch.


Batch Analyzer

Batches

Batch	2018/10/10 All Day
E-Shop	kk5502
Rows	415
First Line Time	2018-10-10 00:19:20
Last Line Time	2018-10-10 23:44:19

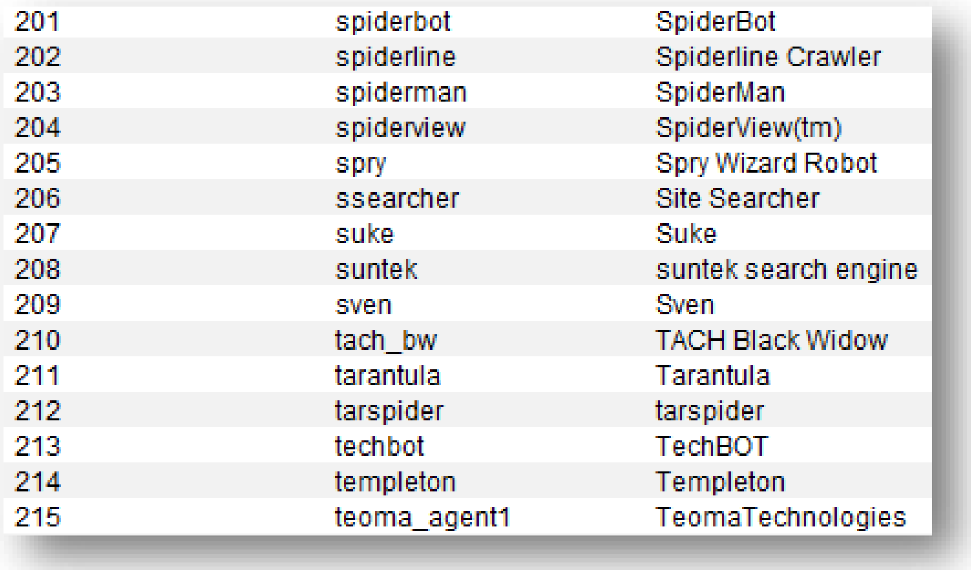
FIGURE 33: BATCH PROCESSING

1. The batch name combo box for selecting batches. E-Shop is the name of the site which owns the batch.
2. The raw number of rows of the batch.
3. Pressing Process Bots, searches the line for known bots. Known bots are recognized through their bot string. The bot strings are stored in the LFA database and the search is relatively slow, since it is done by checking

the visitor IP-addresses for with use for known spiders and bots from the maintained database table (Figure 34).

4. The progress bar shows the progress of loading.

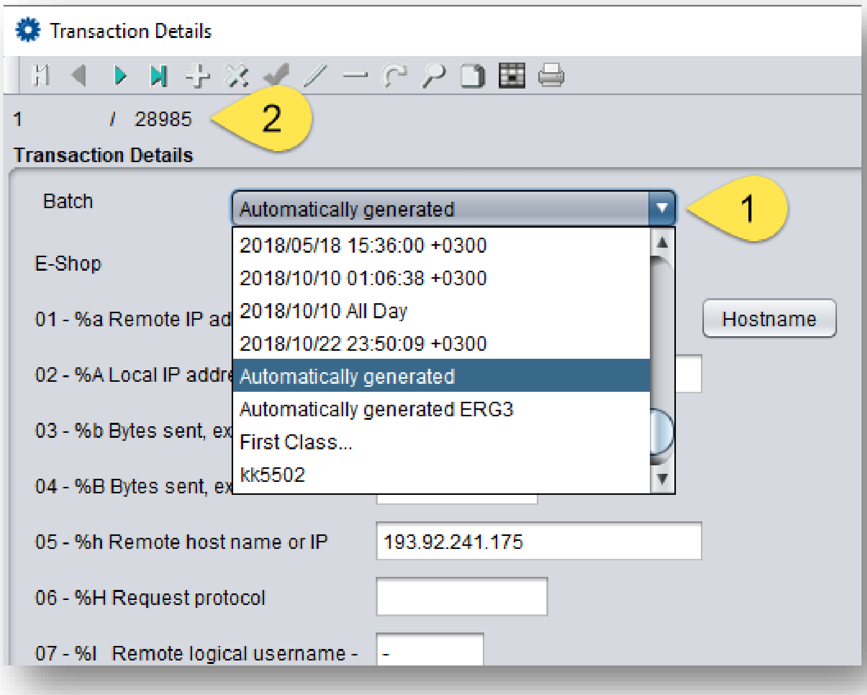
Each batch can also be used in order to delete entire named groups of logfile entries that are not need any further from the database. Batches that are used for testing purposes for example can easily be eliminated after the test they are loaded for is concluded.



201	spiderbot	SpiderBot
202	spiderline	Spiderline Crawler
203	spiderman	SpiderMan
204	spiderview	SpiderView(tm)
205	spry	Spry Wizard Robot
206	ssearcher	Site Searcher
207	suke	Suke
208	suntek	suntek search engine
209	sven	Sven
210	tach_bw	TACH Black Widow
211	tarantula	Tarantula
212	tarspider	tarspider
213	techbot	TechBOT
214	templeton	Templeton
215	teoma_agent1	TeomaTechnologies

FIGURE 34: EXTRACT FROM THE BOT AND SPIDER DATABASE

The Transaction Details Dialog allows viewing every individual detail line that belongs to every batch. The sorting sequence is defined by their physical sequence, as they are generated by the web server. The navigation buttons of the toolbox allow moving from one row entry to the next of previous within this batch and jumping to the first or last row. All line details parsed from the original logfile are displayed in the fields depicted by the selected valve pattern. This application does not allow deletion or modification of the data, since the database must reflect the true contents of the logfile.


Transaction Details

1 / 28985

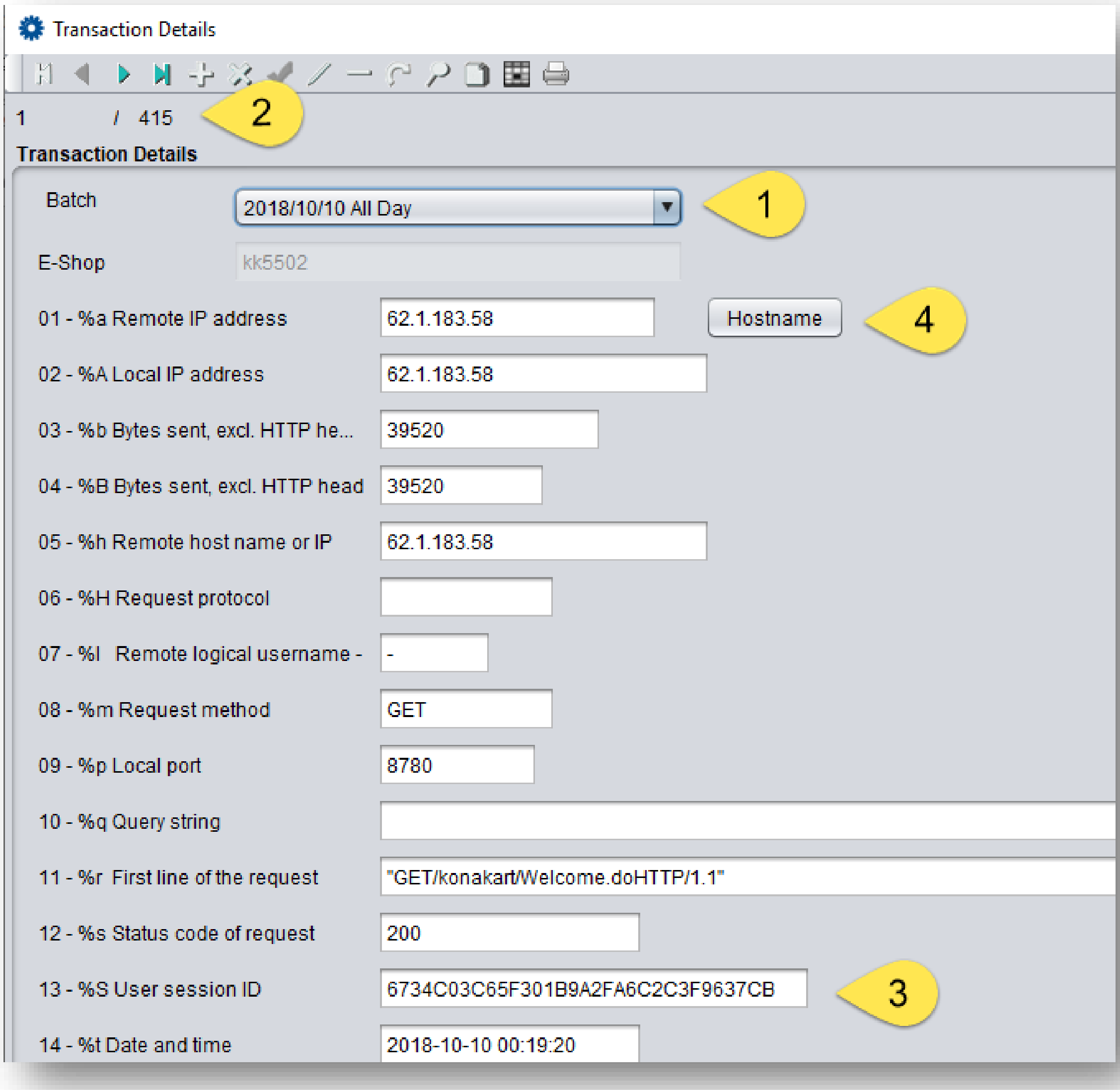
Transaction Details

Batch	Automatically generated
E-Shop	
01 - %a Remote IP ad	
02 - %A Local IP addre	
03 - %b Bytes sent, ex	
04 - %B Bytes sent, ex	
05 - %h Remote host name or IP	193.92.241.175
06 - %H Request protocol	
07 - %l Remote logical username -	-

FIGURE 35: SELECTING A BATCH

The Transactions Dialog (Figure 35) starts with a combo box (1), that allows choosing among the named batches. The total number of rows and the current row are displayed (2) here. After the selection the total row number of the batch is updated. Every new selection sets always at start row 1 as current.

The important information here, is the %S entry seen in Figure 36 at pointer 3. The session-ID is generated automatically by the web server and can be used for uniquely identifying the user session. The session-ID will be used to allow selecting and grouping all the lines and actions that were generated during the entire specific user session. The line is time stamped and the "Hostname" button (4) can be activated in order to check the origin of the visitor.



Transaction Details

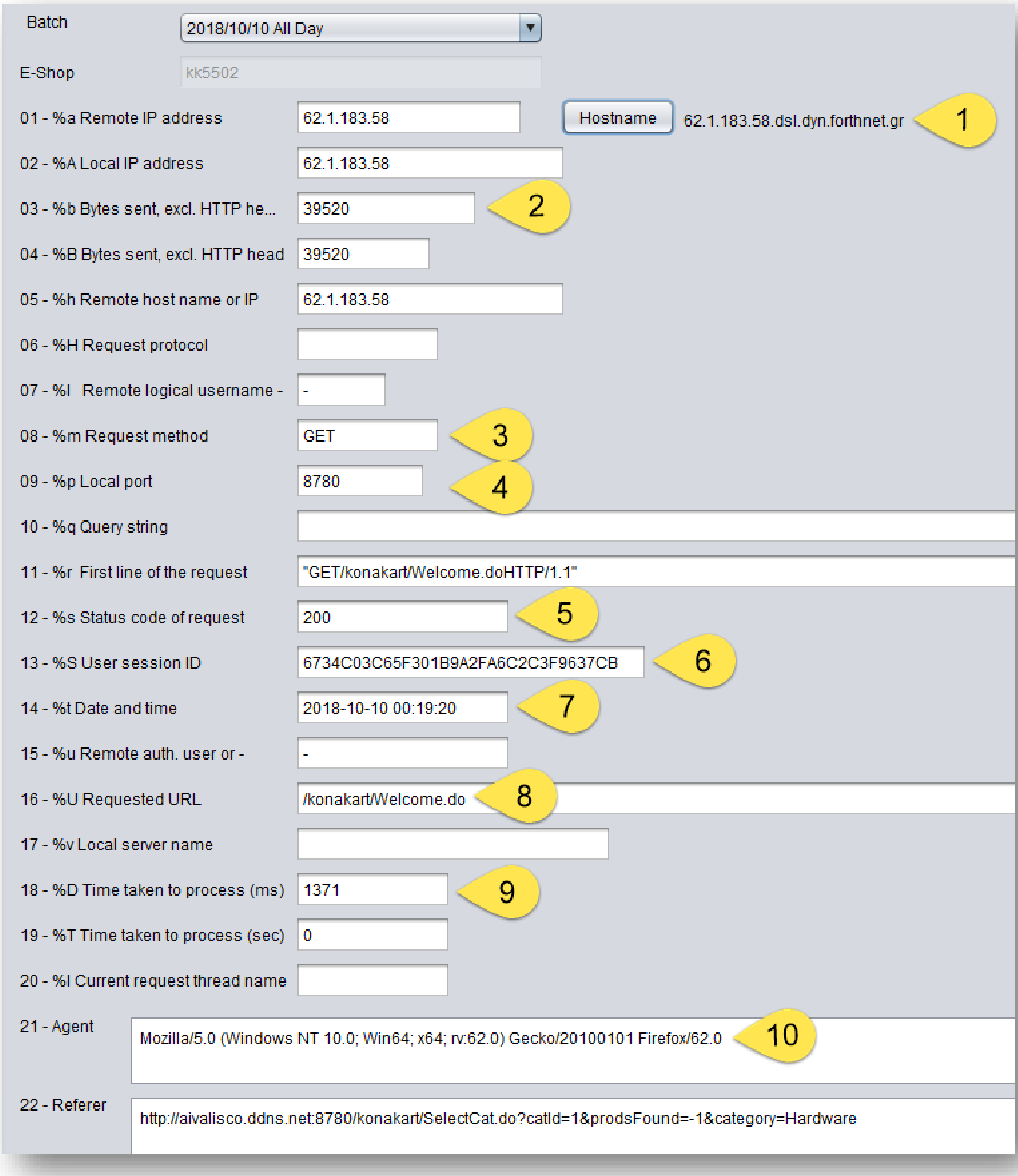
1 / 415

Transaction Details

Batch	2018/10/10 All Day
E-Shop	kk5502
01 - %a Remote IP address	62.1.183.58
02 - %A Local IP address	62.1.183.58
03 - %b Bytes sent, excl. HTTP he...	39520
04 - %B Bytes sent, excl. HTTP head	39520
05 - %h Remote host name or IP	62.1.183.58
06 - %H Request protocol	
07 - %l Remote logical username -	-
08 - %m Request method	GET
09 - %p Local port	8780
10 - %q Query string	
11 - %r First line of the request	"GET/konakart/Welcome.doHTTP/1.1"
12 - %s Status code of request	200
13 - %S User session ID	6734C03C65F301B9A2FA6C2C3F9637CB
14 - %t Date and time	2018-10-10 00:19:20

FIGURE 36: SEARCHING FOR SESSION-IDS

In line 13 of the Transaction Details dialog is the “%S User Session ID” (3) displayed. For this example, we see the 6734C03C65F301B9A2FA6C2C3F9637CB Session ID in the middle of the Details Dialog. Usually remembering the 3 or 4 first digits of this 32-digit hexadecimal number is enough to accurately identify the session and consequently analyze the entirety of the steps and the behavior of this visitor.



Batch	2018/10/10 All Day
E-Shop	kk5502
01 - %a Remote IP address	62.1.183.58
02 - %A Local IP address	62.1.183.58
03 - %b Bytes sent, excl. HTTP he...	39520
04 - %B Bytes sent, excl. HTTP head	39520
05 - %h Remote host name or IP	62.1.183.58
06 - %H Request protocol	
07 - %l Remote logical username -	-
08 - %m Request method	GET
09 - %p Local port	8780
10 - %q Query string	
11 - %r First line of the request	"GET/konakart/Welcome.doHTTP/1.1"
12 - %s Status code of request	200
13 - %S User session ID	6734C03C65F301B9A2FA6C2C3F9637CB
14 - %t Date and time	2018-10-10 00:19:20
15 - %u Remote auth. user or -	-
16 - %U Requested URL	/konakart/Welcome.do
17 - %v Local server name	
18 - %D Time taken to process (ms)	1371
19 - %T Time taken to process (sec)	0
20 - %I Current request thread name	
21 - Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:62.0) Gecko/20100101 Firefox/62.0
22 - Referer	http://aivalisco.ddns.net:8780/konakart/SelectCat.do?catId=1&prodsFound=-1&category=Hardware

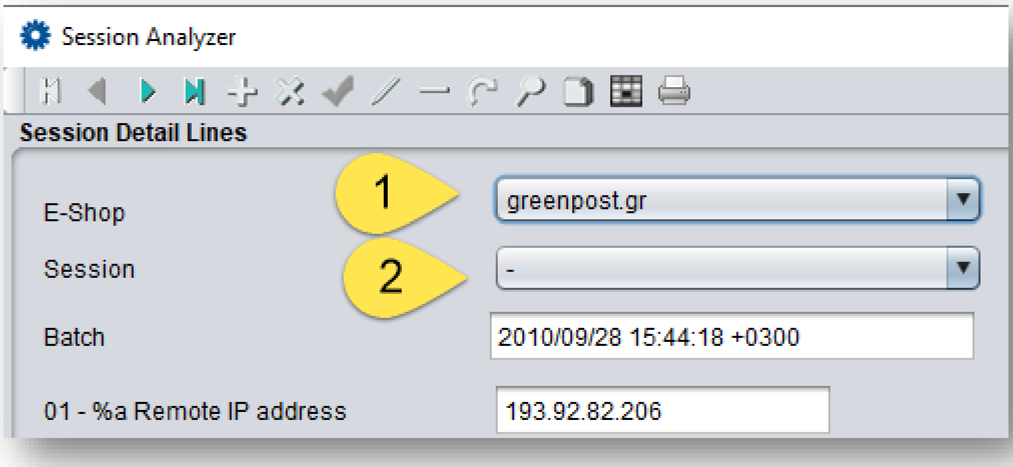
FIGURE 37: HOSTNAME OF REMOTE ADDRESS

We can see immediately in Figure 37:

1. That this session was started from a computer hosted by a dynamic IP from Forthnet S.A. in Greece.
2. There were 39529 bytes sent.
3. It was a GET request.
4. The IP-port where the application runs is 8780.
5. The status of the request was successful (200).
6. The specific session-ID.
7. The timestamp that shows that this request was made at 00:19:20 on October 10th, 2018
8. "Welcome.do" is the first page of the e-shop
9. It took 1371 milliseconds to process the request.
10. The visitor uses a Windows 10 machine and visited with a Firefox 62.0 browser.

8.11. Session Processing

The access logfile is read sequentially and each parsed line of the batch is stored in the data base table. The physical sequence of every line is defined according to its timestamp. This means we end up having lines of multiple sessions mixed that are being transferred one after the other to the database of the analyzer. Sessions need to be identified as sequences of steps. As stated above session-IDs are random 32-digit hexadecimal numbers. Remembering a few of the first digits, for example if we remember 6734 and the e-shop code that served this session, which is 'kk5502' we can isolate all steps of the session in the Session Details application of the Analyzer. By default, the Session Details application displays the first line of the first e-shop session. The two combo boxes are used to locate the session we want in Figure 38, pointer 1 and 2:

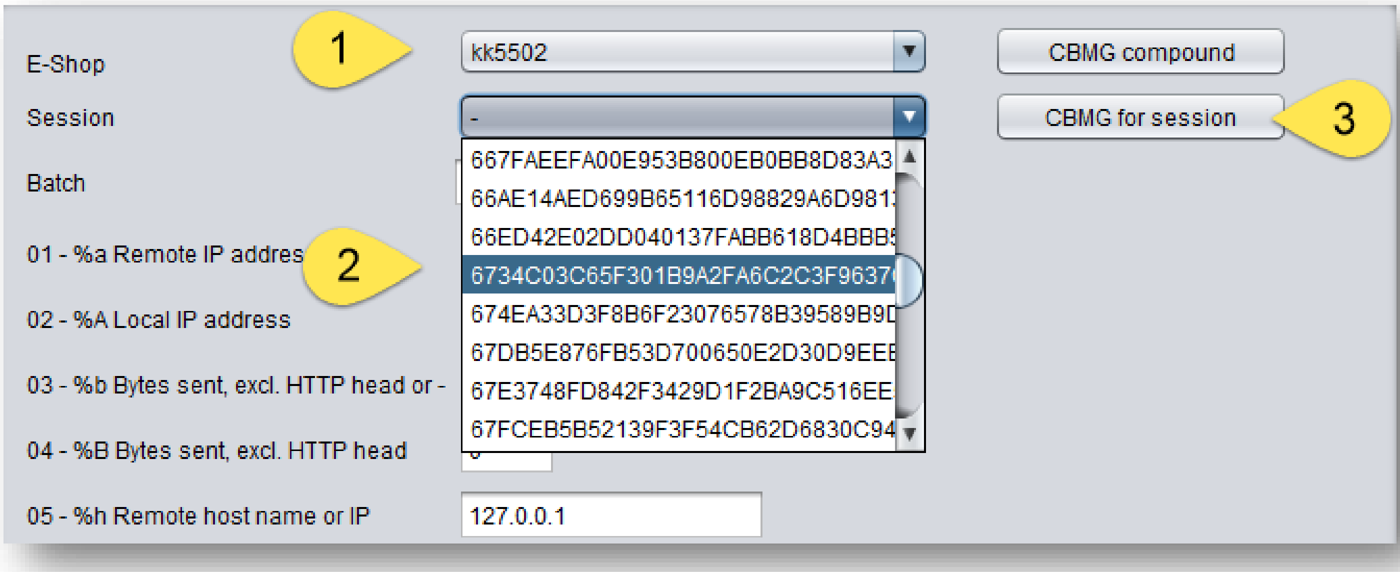


E-Shop	greenpost.gr
Session	-
Batch	2010/09/28 15:44:18 +0300
01 - %a Remote IP address	193.92.82.206

Figure 38: Finding The Session

We choose “kk5502” for the e-commerce site we are interested in at the first combo box (Figure 39) (pointer 1) and the session starting with “6734” in the session selection combo box (pointer 2). This simple procedure restricts the application selection to isolate just the lines of the wanted session. The 6743… session of this example consists of only 23 lines. The navigation buttons allow access only to these steps, from the first one to the last that belong to the same session.

The CBMG button (step 3) generates a Customer Behavior Model Graph for this short session.



E-Shop kk5502

Session -

667FAEEFA00E953B800EB0BB8D83A3
66AE14AED699B65116D98829A6D981
66ED42E02DD040137FABB618D4BBB
6734C03C65F301B9A2FA6C2C3F9637
674EA33D3F8B6F23076578B39589B9D
67DB5E876FB53D700650E2D30D9EEE
67E3748FD842F3429D1F2BA9C516EE
67FCEB5B52139F3F54CB62D6830C94

Batch

01 - %a Remote IP addres

02 - %A Local IP address

03 - %b Bytes sent, excl. HTTP head or -

04 - %B Bytes sent, excl. HTTP head

05 - %h Remote host name or IP 127.0.0.1

FIGURE 39: SESSION FOUND

We can navigate among the log file lines and see that the whole session stops after selecting a category. The session CBMG generated by the analyzer is connecting only two nodes (Figure 40).

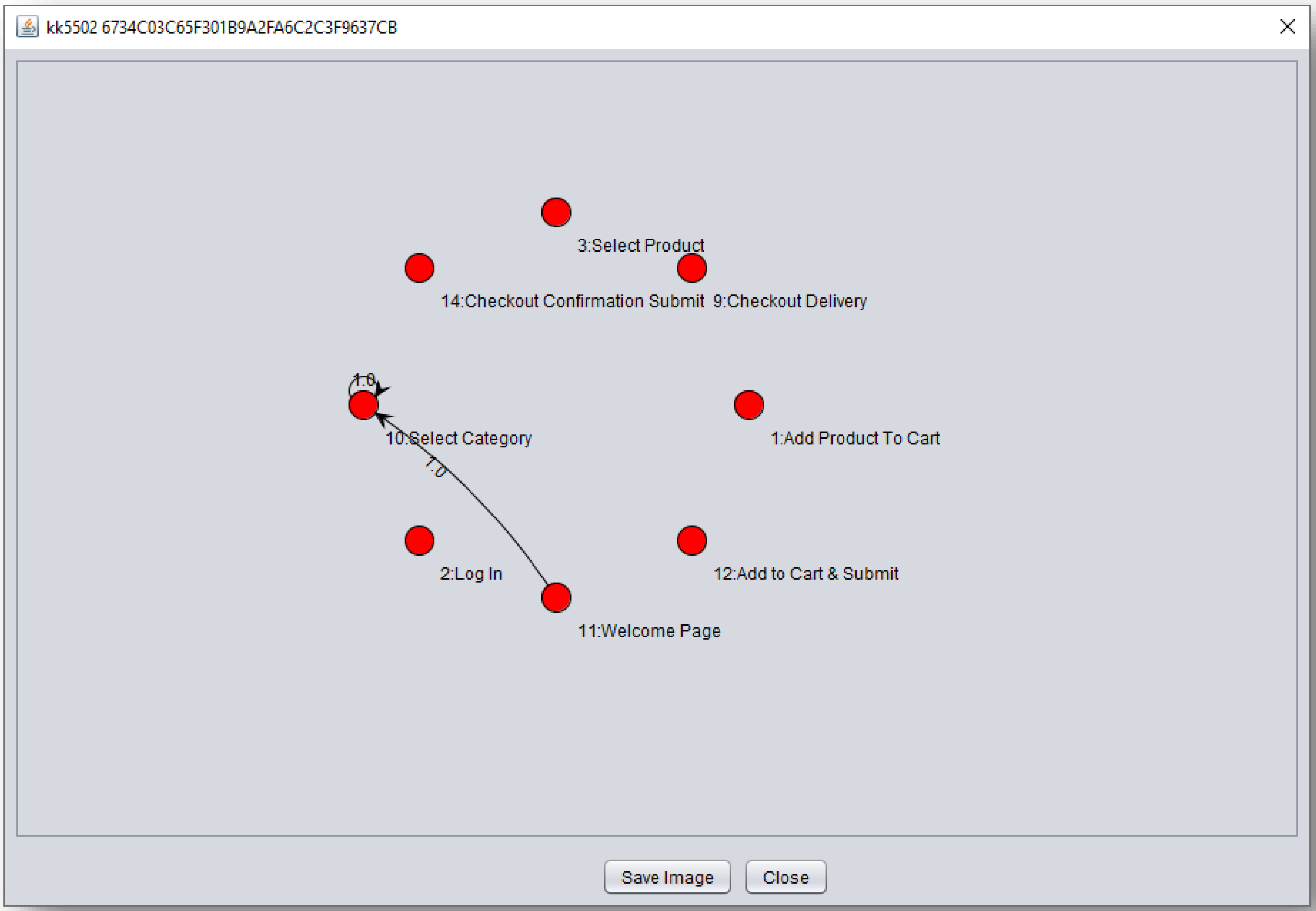


FIGURE 40: CBMG FOR A SESSION

We can see here that the visitor in this session did only select a category and left without even choosing a product. The CBMG application allows manipulation to provide better readability. Nodes can be selected and rearranged so that the final image can be viewed better and saved for further study. The title of the window includes the e-shop name and the specific session it displays.

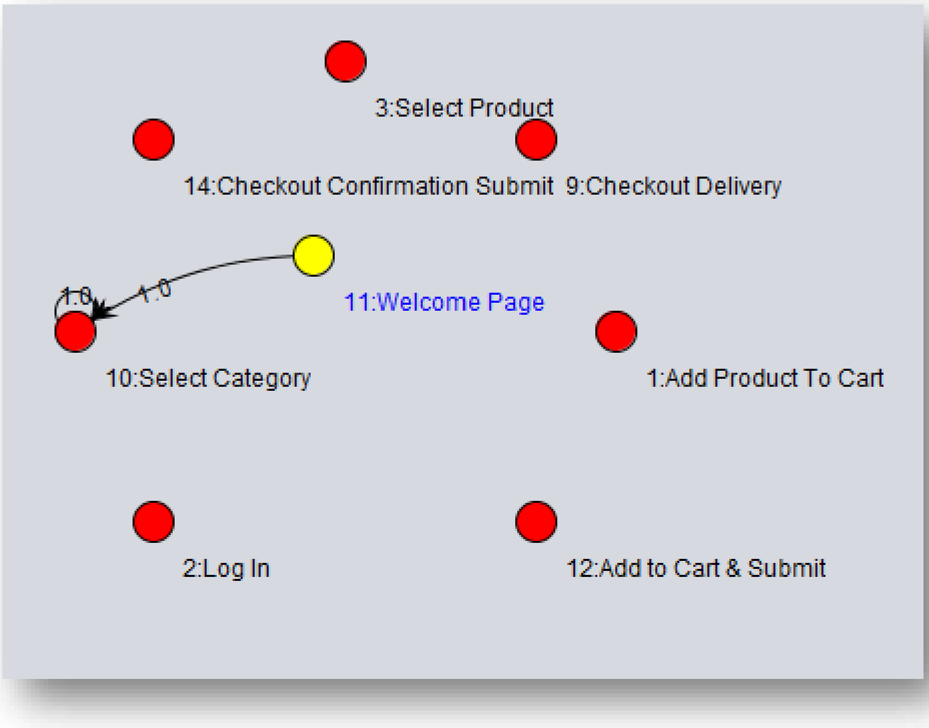


FIGURE 41: MANIPULATING NODES

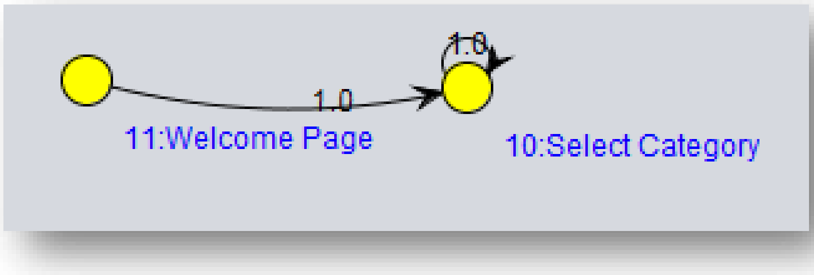


FIGURE 42: SESSION 6734...

These three lines, out of the 23 of this session, from logfile of this batch that generated this CBMG:

```
[10/Oct/2018:00:19:20 +0300] "GET /konakart/Welcome.do HTTP/1.1" 200

[10/Oct/2018:00:19:26 +0300] "GET /konakart/SelectCat.do?catId=1 HTTP/1.1" 302

[10/Oct/2018:00:19:26 +0300] "GET /konakart/SelectCat.do?catId=1&prodsFound=-
1&category=Hardware HTTP/1.1" 200
```

The transition matrix upon which the graph of session 6734 was based is a standard two state transition matrix:

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1.0	0	0	0
0	0	0	0	1.0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

The CBMG Compound button shown, generates a compound CBMG graph (Figure 43), visualizing the graph for all sessions of the e-shop.

We can manipulate the necessary nodes to enhance readability of this cluttered graph (Figure 44). Still JUNG2, the graphics library upon which the graph is based is fast but cannot deal very well with huge numbers of nodes and edges.

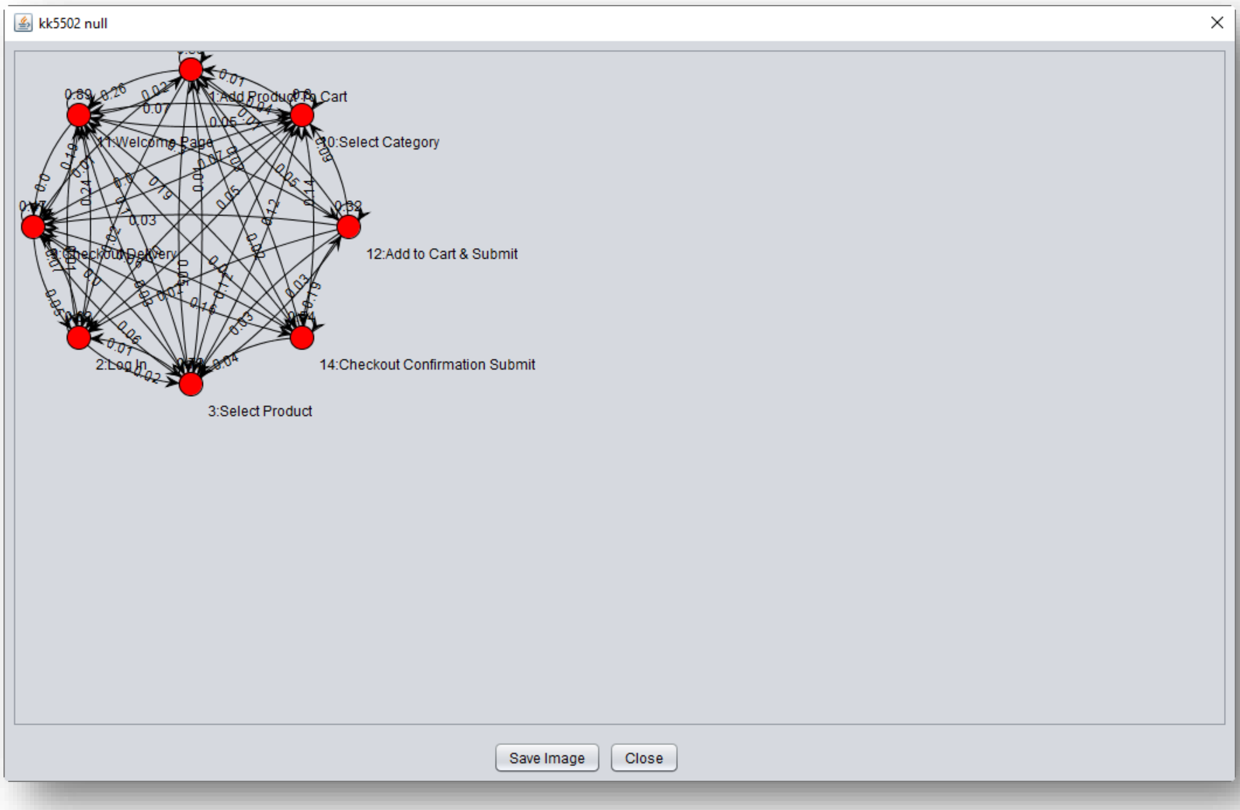


FIGURE 43: COMPOUND CBMG

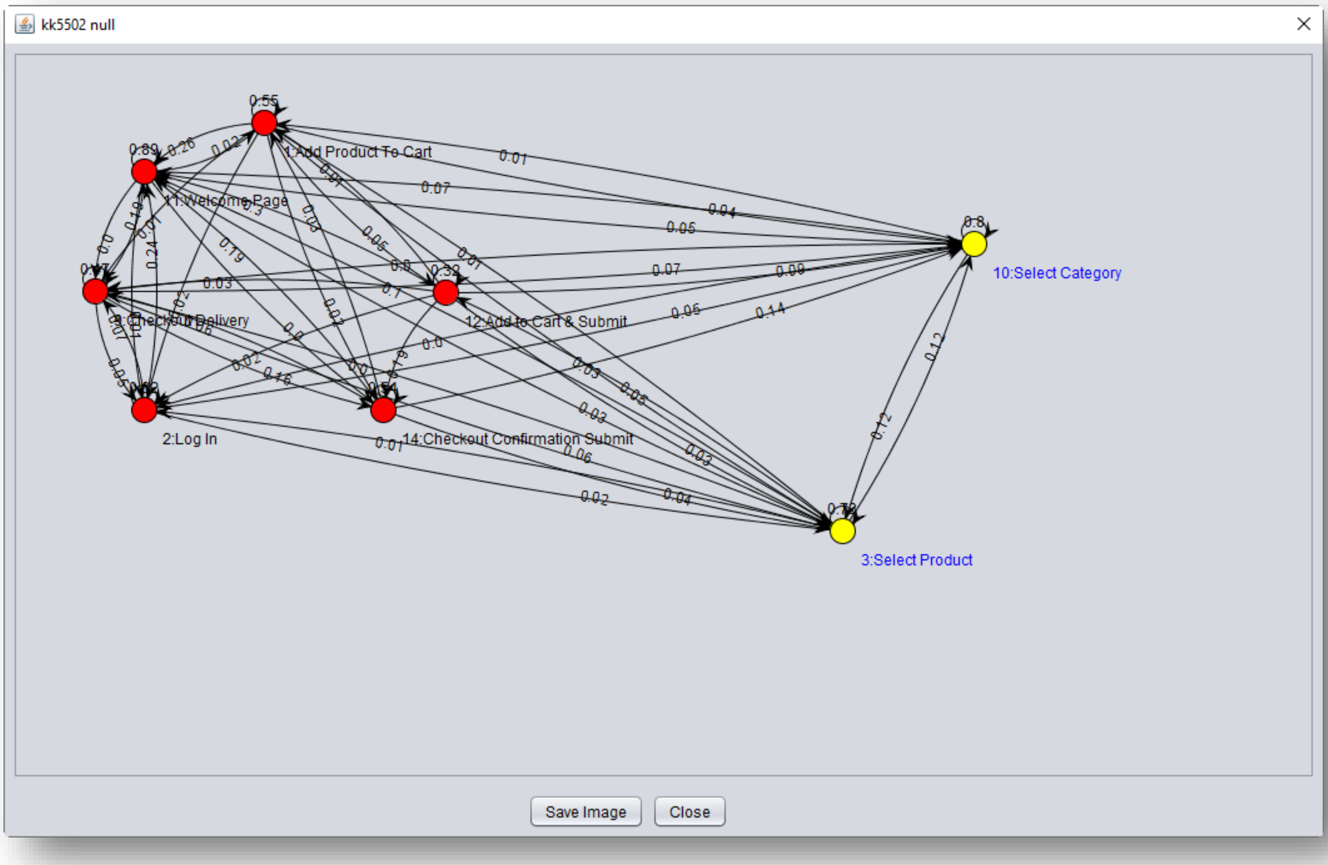
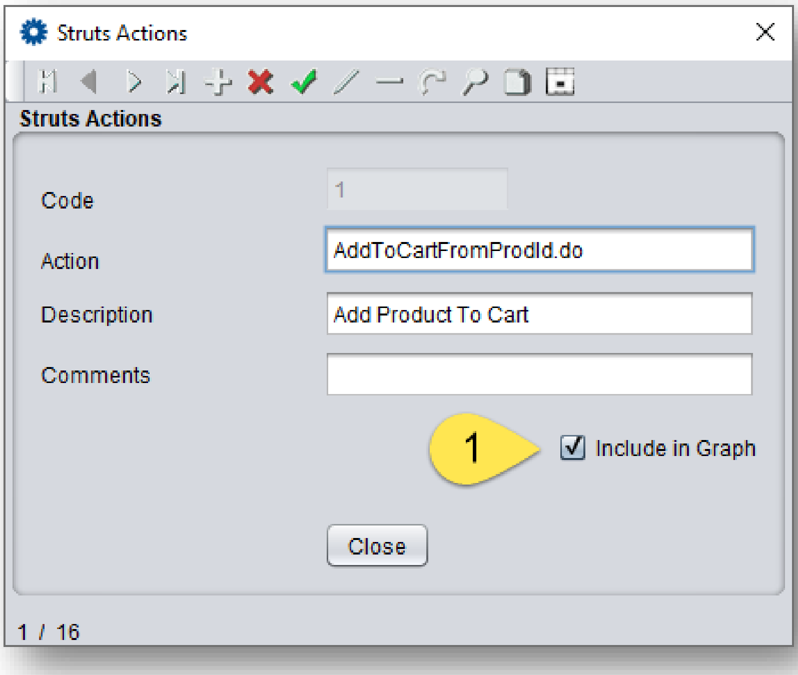


FIGURE 44: MANIPULATING NODES

The nodes represent actions like Category Selection, Logging Into the system, Selecting a Product etc. The user can select which nodes to include to the graph at any time. This is done by a node manipulation application in the Tools Menu. There are multiple actions that need to be analyzed. The sample e-commerce application is developed under the Struts2 framework. The actions are stored in an actions table in the database of the analyzer. Whether they must appear as nodes of the CBMG or not is user defined. If the checkbox (1) in Figure 45 is checked, then the action generates a node in the graph and participates in the computations that form the transition matrix.


Struts Actions

Struts Actions

Code 1

Action AddToCartFromProdId.do

Description Add Product To Cart

Comments

[✓ Include in Graph]

1 / 16

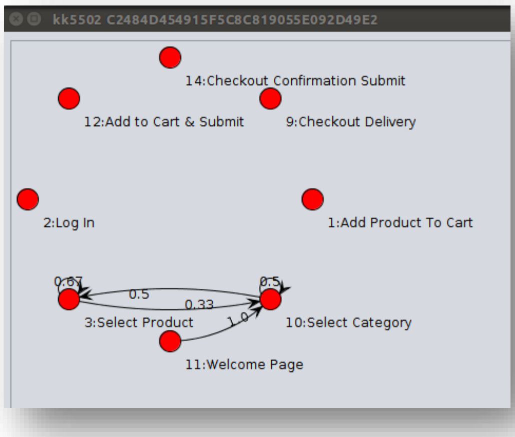
FIGURE 45: STRUTS ACTIONS

Four session examples, with their detailed activities listed, the CBMG as generated by the LFA, the details manipulated in order to make the figures clearer visible and finally the transition matrices that produce are following:

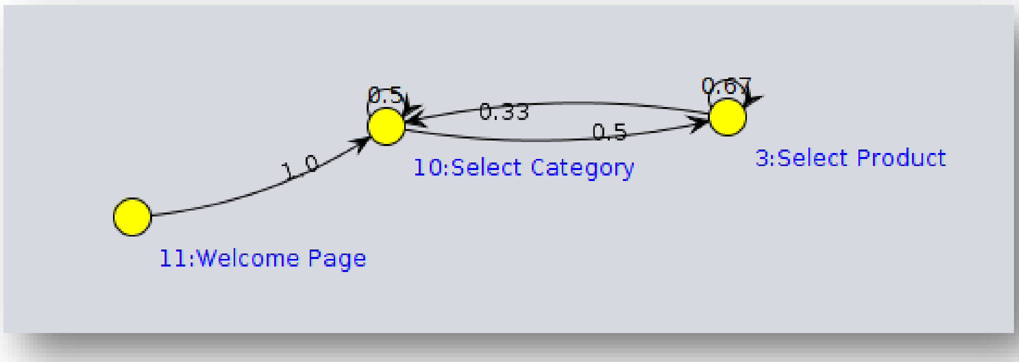
<u>Session Example 1</u>

Activity Chain: visit homepage ⟶ select category ⟶ choose product ⟶ select category ⟶ choose product

CBMG:



CBMG Detail:



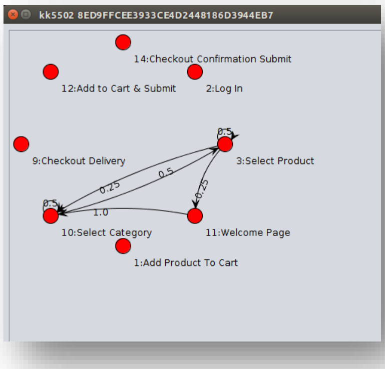
Transition Matrix:

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0.67	0	0.33	0	0	0
0	0	0	0	0	0	0	0
0	0	0.5	0	0.5	0	0	0
0	0	0	0	1.0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

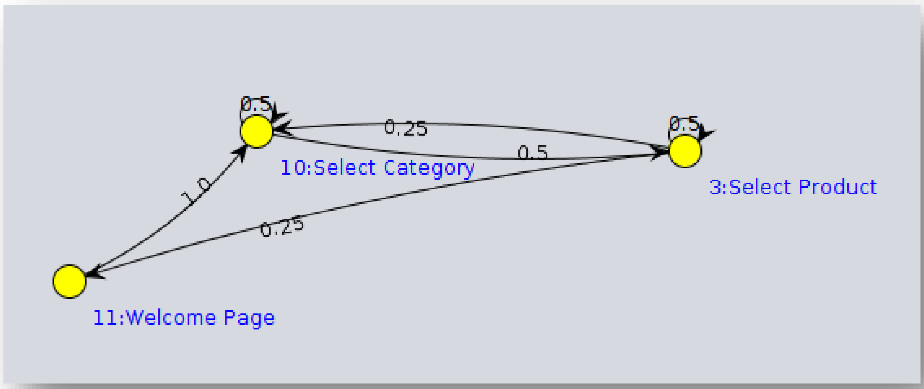
Session Example 2

Activity Chain: visit homepage ⟶ select category ⟶ choose product ⟶ select category ⟶choose product ⟶ visit homepage

CBMG:



CBMG Detail:



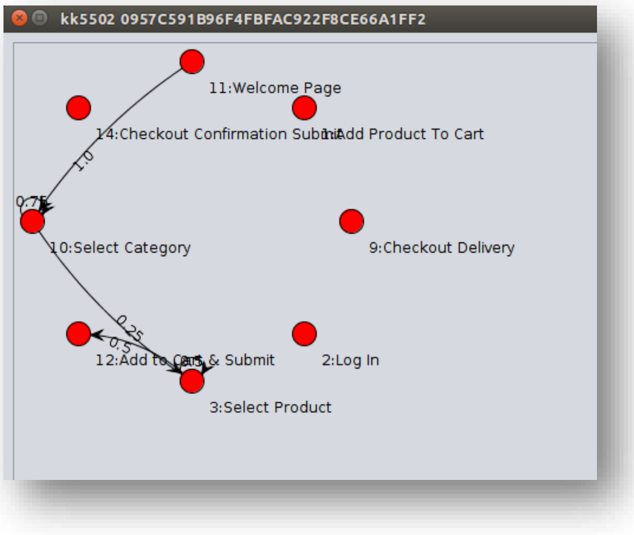
Transition Matrix:

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0.5	0	0.25	0.25	0	0
0	0	0	0	0	0	0	0
0	0	0.5	0	0.5	0	0	0
0	0	0	0	1.0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

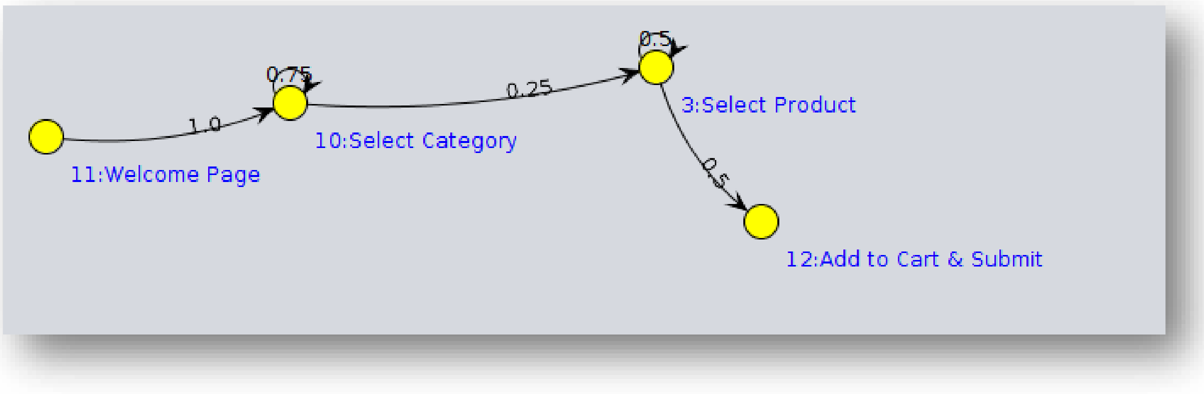
Session Example 3

Activity Chain: visit homepage → select category → select category → choose product → add to cart

CBMG:



CBMG Detail:



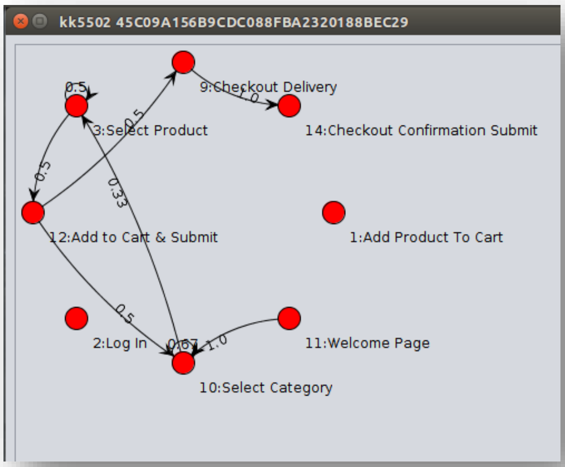
Transition Matrix:

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0.5	0	0	0	0.5	0
0	0	0	0	0	0	0	0
0	0	0.25	0	0.75	0	0	0
0	0	0	0	1.0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

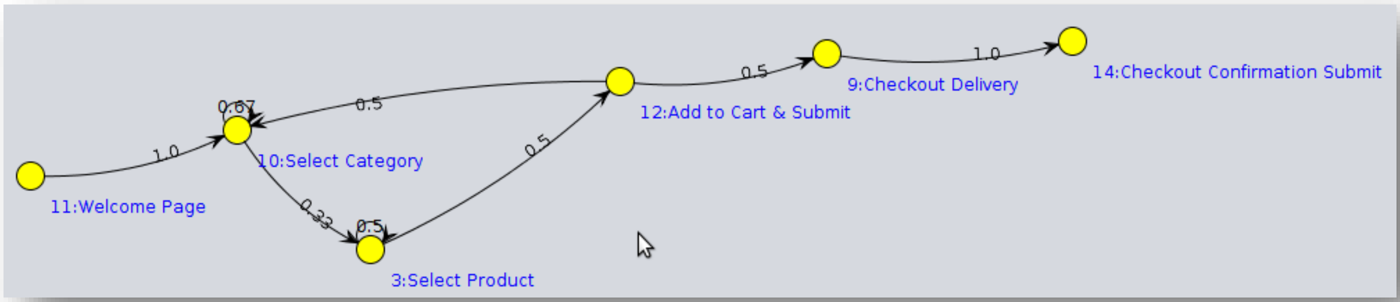
Session Example 4

Activity Chain: visit homepage ⟶ select category ⟶ select category ⟶ choose product ⟶ add to cart ⟶ select category ⟶ choose product ⟶ add to cart ⟶ checkout ⟶ confirm purchase ⟶ logout

CBMG:



CBMG Detail:



Transition Matrix:

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0.5	0	0	0	0.5	0
0	0	0	0	0	0	0	1.0
0	0	0.33	0	0.67	0	0	0
0	0	0	0	1.0	0	0	0
0	0	0	0.5	0.5	0	0	0
0	0	0	0	0	0	0	0

8.12. Streaming and Daemons

Streaming supports the information transportation from source to consumer. The data source is usually the web server, or the database server and consumer is the Analyzer application. Reason for streaming is the need to transfer either the contents of the logfile or specific information, generated by database triggers, needed by the analyzer and the presentation tools. Streaming is achieved through queue producer and queue consumer pairs, running as daemon processes on both systems.

Logfile loading for the on-demand analyzer takes place only when the user decides to run the loader applications. The two logfile loading tools, discussed in two previous sections of this chapter, can handle both single and multiple rotatable logfiles. The single logfile loader can either open a specific logfile or support the multiple selections from a rotatable file-chooser widget. Either way, the user is responsible for activating the loading procedure before new data can be used. Also, the amount of data loaded every time the loaders are activated can be large, since the loading periods may not be very frequent.

It is more convenient to automate the loading procedure to take place in near real time. Each time we have a logfile append, the delta should be collected, serialized and sent to the analyzer. This can be achieved by installing the appropriate queuing daemon processes on the sending and the receiving nodes. Daemon processes are programs running in the background that respond to specific events or run at predefined times. All operating systems have daemon processes for serving various needs, like printer-spooler daemons, mail daemons, web server daemons etc. Daemons are also called services in Microsoft Windows. Some daemons can be started at boot time, other via scripts or manually.

The sending daemon is the “producer” and the receiving daemon the “consumer”. Producer and consumer communicate via at least one specified TCP/IP port. When the system where the analyzer resides starts running the consumer, a connection with the producer is established and a dequeuing procedure starts. If the producer is empty the two systems are synchronized and waiting. If the producer has queued data before the consumer connected the packaged data is dequeued and loaded. If the queue is emptied the systems wait. The received information is parsed and stored to the database of the LFA as soon as the consumer receives it.

We use a message broker for the implementation of the queueing mechanism. The wire-level mechanism that is basically based upon sockets and ‘sits’ on top of the Advanced Message Queuing Protocol, or AMQP is RabbitMQ.

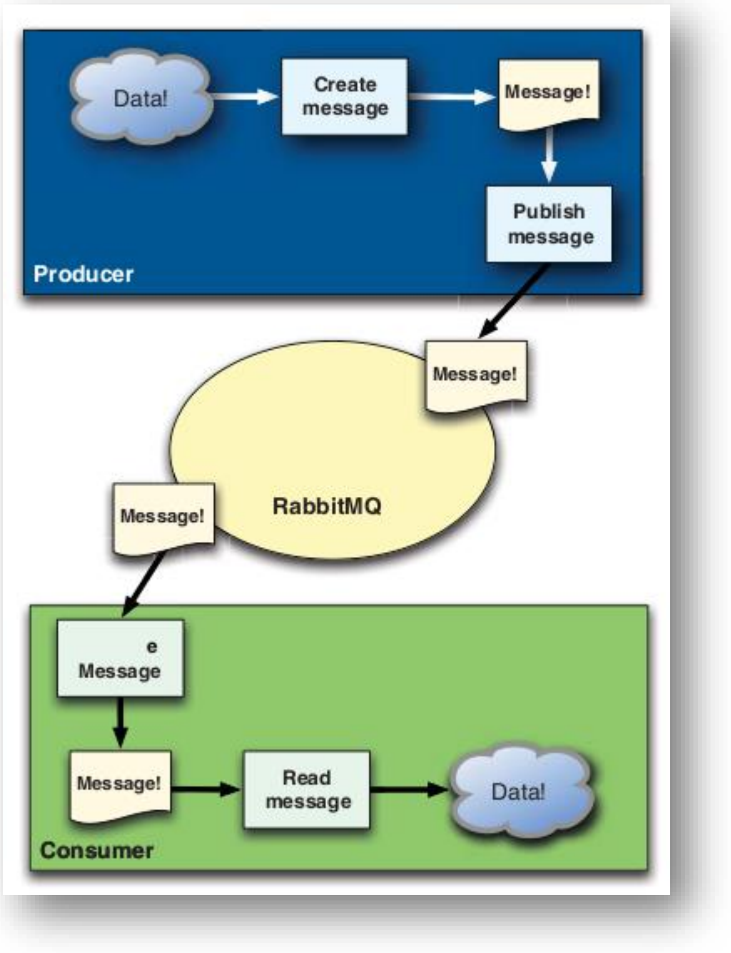


FIGURE 46: RABBITMQ MESSAGE FLOW

Figure 46[28] shows how messages flow in RabbitMQ. The producer 'knows' where the logfile is and opens it for reading as a RandomAccessFile. It constantly reads a line, converts the line to bytes, queues the bytes and publishes them to the open channel as a message and waits for new data.

If the access logfile is rotatable every day, the producer checks for the new generated logfile right after the day changes.

[28] Manning RabbitMQ in Action Distributed Messaging for Everyone Alvaro Videla, Jason J.W. Williams May 2012

The consumer opens a database connection to the database of the LFA, consumes the data from the body of the delivery, converts the bytes back to String, parses the information, stores it to the database and waits for the next delivery.

This way, even when the LFA is not running, if the consumer is up, the database is getting permanently updated. No data is lost, and it is no longer necessary to ever load logfiles manually.

While logfile appends trigger the producer to send data via the AMQP queuing mechanism, other sources of data send information through the queuing system in parallel. The source data is generated by triggers at the database level. We saw above that Rich Internet Application with heavy Ajax support do not generate detailed access logfiles that can be analyzed the way traditional access logfiles can. Tagging analyzers can remedy the situation, but in order to be able to avoid the shortcomings of tagging systems and have in house metrics as well, we generate the necessary data and stream it to the LFA (Figure 20), (Figure 21), (Figure 22).

The NRT logfile feed procedure UML activity diagram of Figure 47 shows the steps involved for an automated perpetual logfile feed. 1. While the user interacts with the e-shop (1) the web server registers (1.1) the interaction data to the logfile. The Producer polls for logfile deltas (2), fetches and enqueues them (3). The consumer polls the queue and dequeues the data (4), transforms it and loads to the Database of the LFA (5).

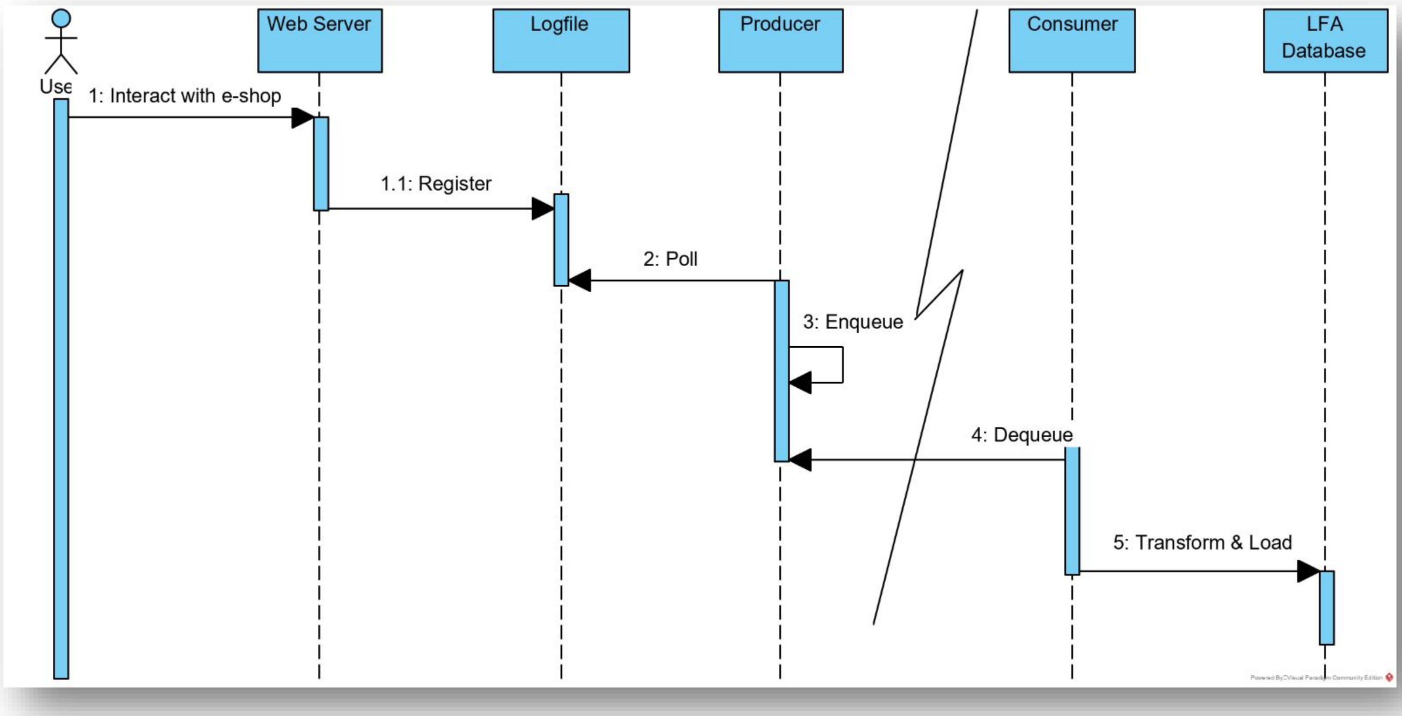


FIGURE 47: LOGFILE DELTAS POLING

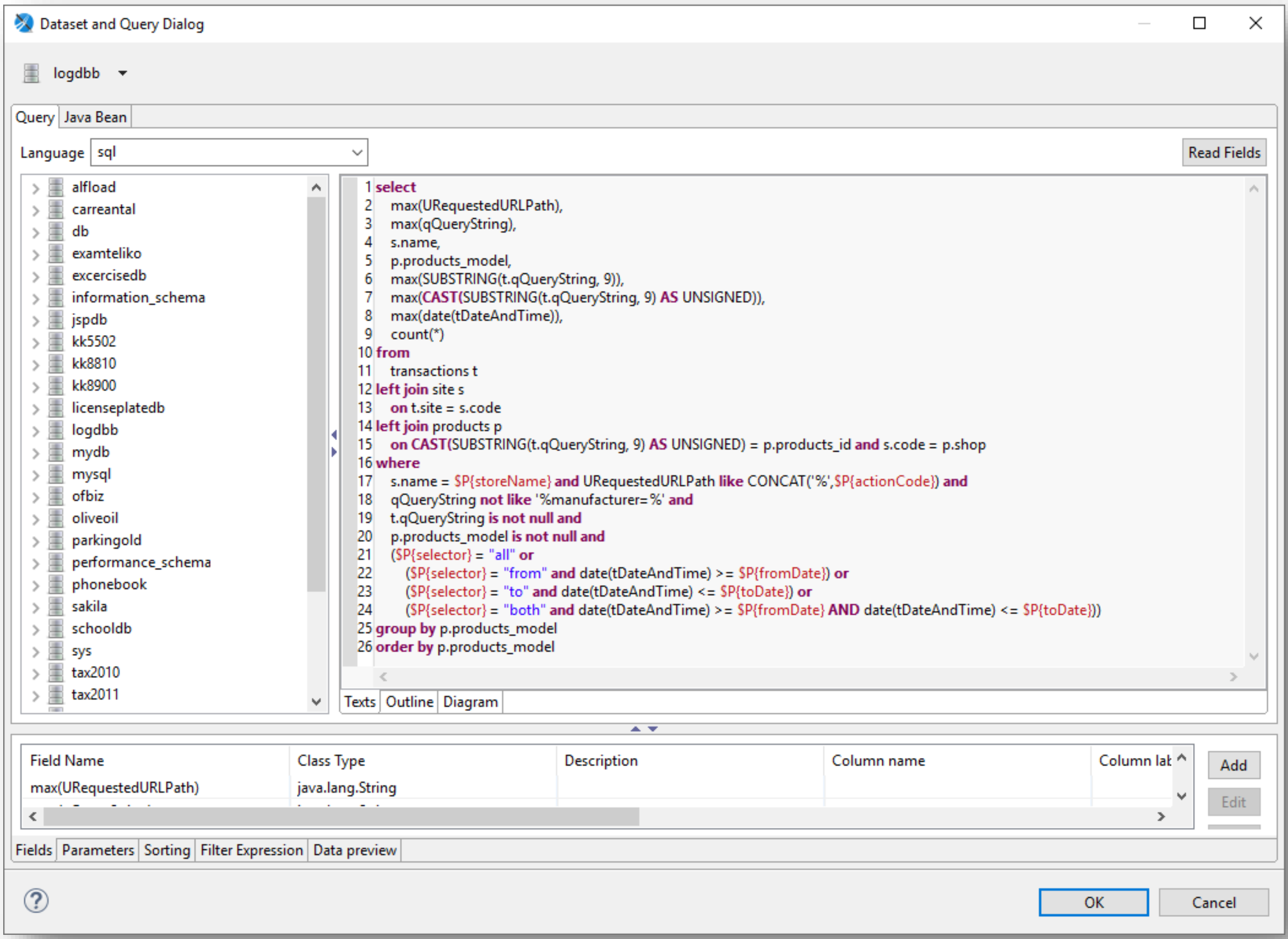
The Database of the LFA is almost immediately updated with fresh access data, ready for processing and visualizing.

8.13. Visualization

The visualization systems of the application generate reports, graphics, interactive graphics, the customer behavior model graph, and GWT graphics.

The generation of customizable reports and export of formatted results and information from the LFA is based on JasperSoft Studio. JasperSoft Studio is the evolution of the JasperReports library, written by Teodor Danciu in 2001 and the iReport Designer visual editor, written by Giulio Toffoli in 2004[29] [61]. The interface of the Integrated Development Environment of the system is based on Eclipse and allows interactive generation of reports. All necessary design details of each report are contained in a single XML file, produced by the system. These XML files could also be written by hand, with just the use of a simple text editor, but JasperSoft Studio generates them interactively and automatically while the user manipulates visually the document and their structure is quite complicated. JasperReports has defined its own XML-based markup language called JasperReports XML [62]. The generated XML files have the “JRXML” extension and they include all formatting instructions, parameters, including the SQL query in order to fetch the necessary data from the database, the fields used and their visual characteristics. A wizard interface allows the user to use the ‘dataset and query’ dialog (Figure 48), for writing the SQL select statement in order to fetch the necessary data for each report.

[29] https://www.jaspersoft.com/company



```sql
select
   max(URequestedURLPath),
   max(qQueryString),
   s.name,
   p.products_model,
   max(SUBSTRING(t.qQueryString, 9)),
   max(CAST(SUBSTRING(t.qQueryString, 9) AS UNSIGNED)),
   max(date(tDateAndTime)),
   count(*)
from
   transactions t
left join site s
   on t.site = s.code
left join products p
   on CAST(SUBSTRING(t.qQueryString, 9) AS UNSIGNED) = p.products_id and s.code = p.shop
where
   s.name = $P{storeName} and URequestedURLPath like CONCAT('%',$P{actionCode}) and
   qQueryString not like '%manufacturer=%' and
   t.qQueryString is not null and
   p.products_model is not null and
   ($P{selector} = "all" or
      ($P{selector} = "from" and date(tDateAndTime) >= $P{fromDate}) or
      ($P{selector} = "to" and date(tDateAndTime) <= $P{toDate}) or
      ($P{selector} = "both" and date(tDateAndTime) >= $P{fromDate} AND date(tDateAndTime) <= $P{toDate}))
group by p.products_model
order by p.products_model
```

Field Name	Class Type	Description	Column name	Column lat
max(URequestedURLPath)	java.lang.String			

FIGURE 48: DATABASE AND QUERY DIALOG

The fields involved in the query can be read, parameters can be defined, and data can be previewed in order to check the results that will be transferred to the report.

JasperSoft Studio offers a graphical user interface with drag and drop widget-capabilities for designing reports. It is a multipurpose tool that can deal with many types of input, including XML and JSON files, Excel files and has database support for most database management systems, including MySQL and Hive for Big Data applications. In terms of output, it can generate HTML, PDF, XML, ODT, DOCX, as well as many other forms of output, just by choosing the right option from a combo box.

TIBCO Software Inc., the company who own the software, offer also an open source community edition of the application, with slightly reduced functionality compared to the commercial version. This is the version that was used for the LFA and overs sufficiently all basic reporting needs of the Log File Analyzer.

Basic reports are pre-built into the application and are accessible through the menu bar. The JasperSoft Studio can be used in order to generate new reports at any time. It is simple to add the corresponding menu items and place them to the 'Evaluation' menu (Figure 49, point 2).

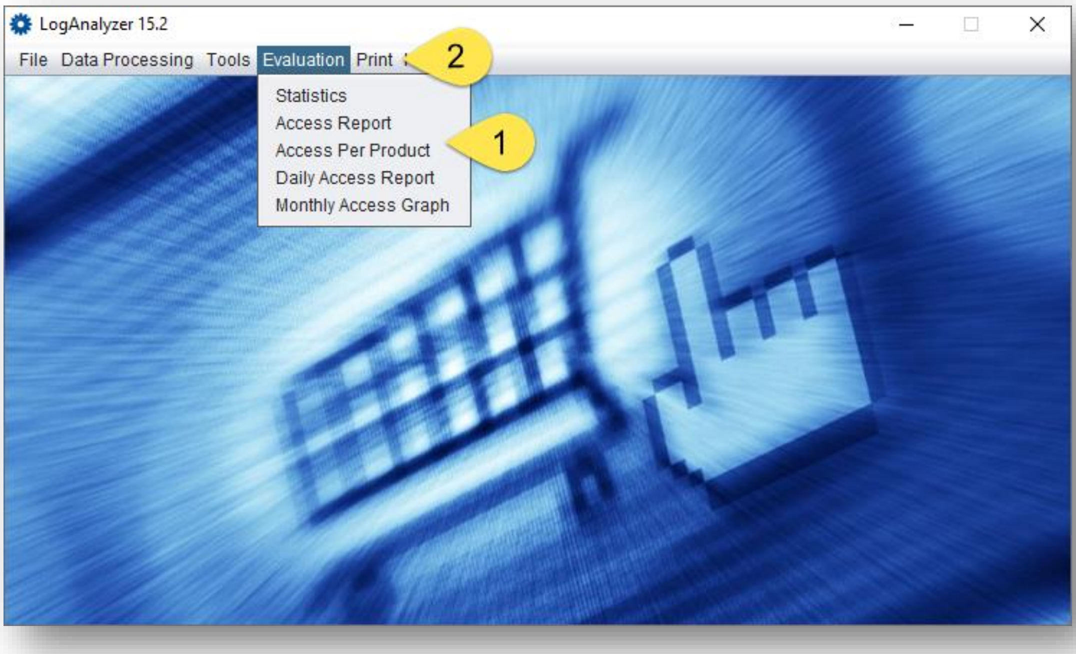
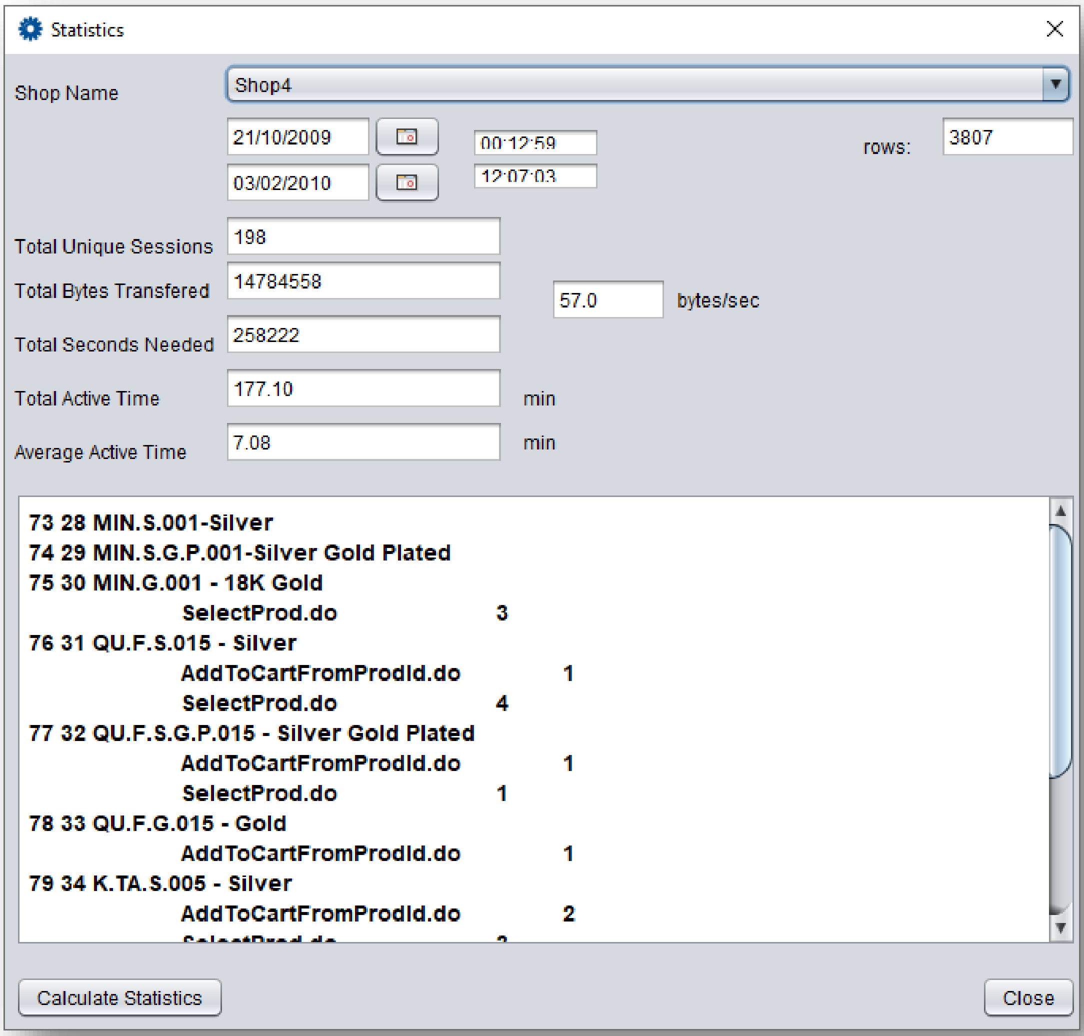


FIGURE 49: THE EVALUATION MENU AND ITS MENU ITEMS

The Statistics dialog of Figure 50 computes and displays number of unique sessions, total bytes transferred, bytes per second and user's activity times over periods of time for each e-shop. The text area, at the lower section shows the activity of each product of the e-shop during the selected time period.



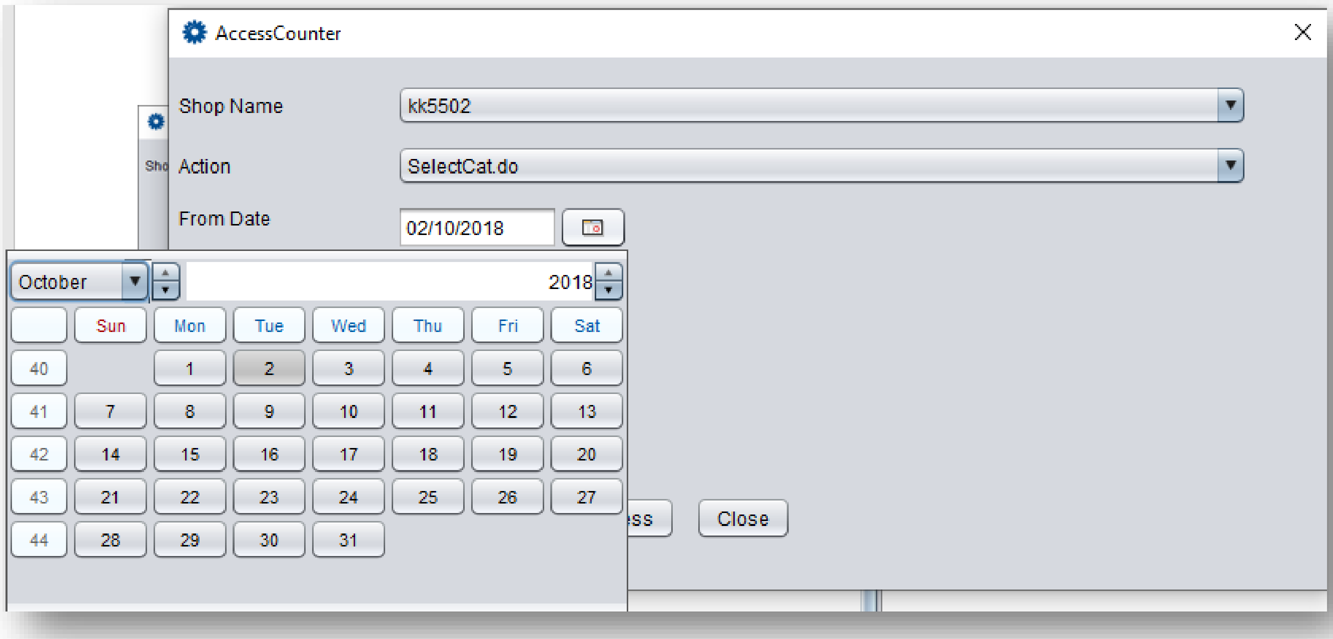
Shop Name: Shop4
21/10/2009 00:12:59 rows: 3807
03/02/2010 12:07:03

Total Unique Sessions	198	
Total Bytes Transfered	14784558	57.0 bytes/sec
Total Seconds Needed	258222	
Total Active Time	177.10	min
Average Active Time	7.08	min

```
73 28 MIN.S.001-Silver
74 29 MIN.S.G.P.001-Silver Gold Plated
75 30 MIN.G.001 - 18K Gold
                SelectProd.do                3
76 31 QU.F.S.015 - Silver
                AddToCartFromProdId.do          1
                SelectProd.do                4
77 32 QU.F.S.G.P.015 - Silver Gold Plated
                AddToCartFromProdId.do          1
                SelectProd.do                1
78 33 QU.F.G.015 - Gold
                AddToCartFromProdId.do          1
79 34 K.TA.S.005 - Silver
                AddToCartFromProdId.do          2
```

FIGURE 50: STATISTICS DIALOG

The menu items of Figure 49 lead to dialogs like the one shown in Figure 51. This dialog is used in order to read the parameter values that will be passed to the Jasper report that will collect, format and visualize the information. Here the shop name, action and the from- and to- dates are collected.


AccessCounter

Shop Name: kk5502

Action: SelectCat.do

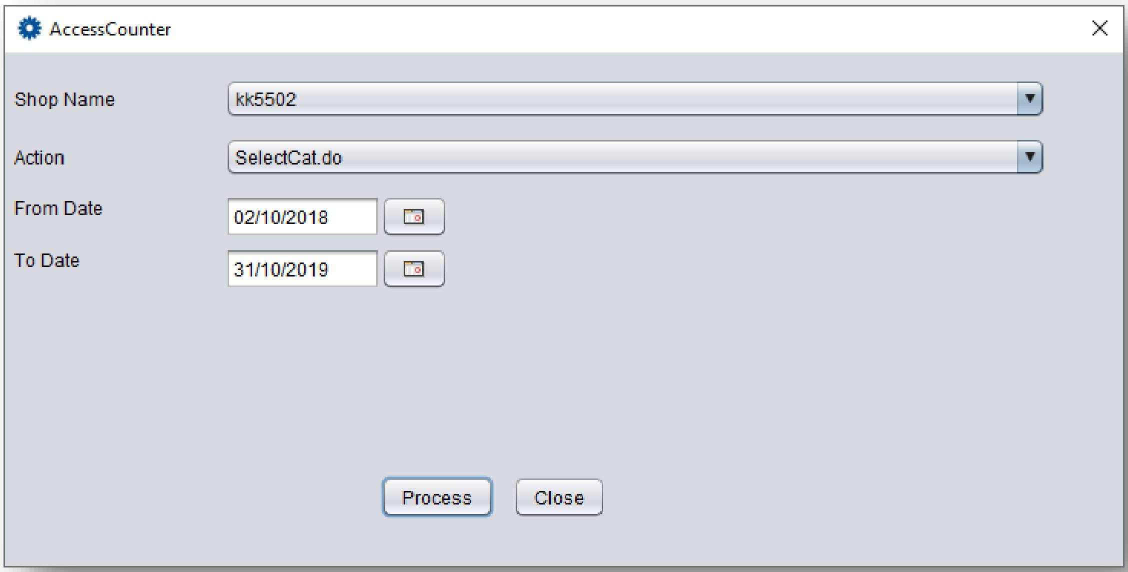
From Date: 02/10/2018

October 2018

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
40		1	2	3	4	5	6
41	7	8	9	10	11	12	13
42	14	15	16	17	18	19	20
43	21	22	23	24	25	26	27
44	28	29	30	31			

FIGURE 51: ACCESS PER PRODUCT DATE SELECTION DIALOG

Figure 52 shows the form with all fields filled in, ready to process the data and generate the report.


AccessCounter

Shop Name: kk5502

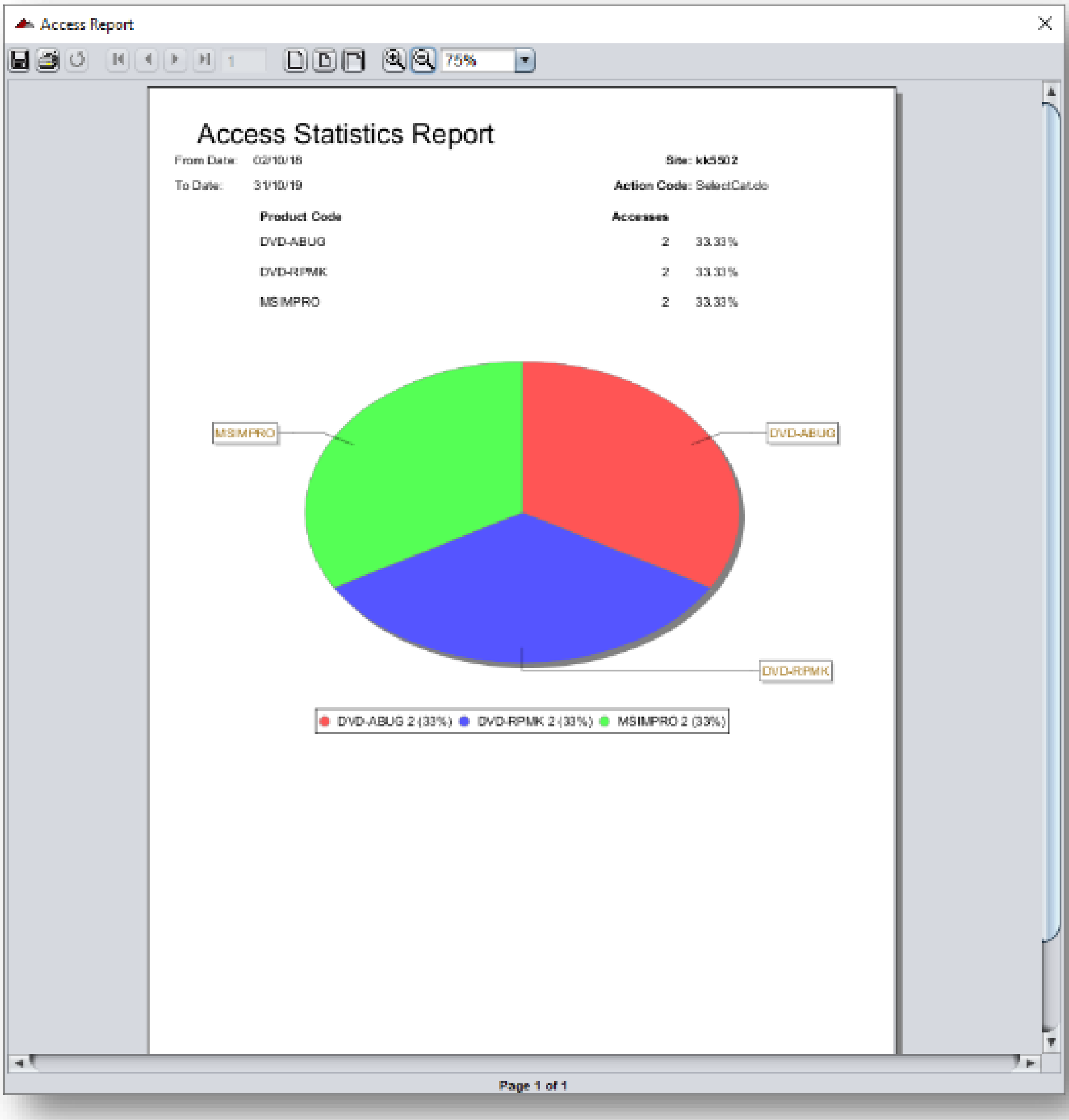
Action: SelectCat.do

From Date: 02/10/2018

To Date: 31/10/2019

FIGURE 52: ACCESS PER PRODUCT FORM FILLED

The dialog of Figure 53 shows the standard form of visualization results presentation. The generated report can be



Access Statistics Report

From Date: 02/10/18 Site: kk5502

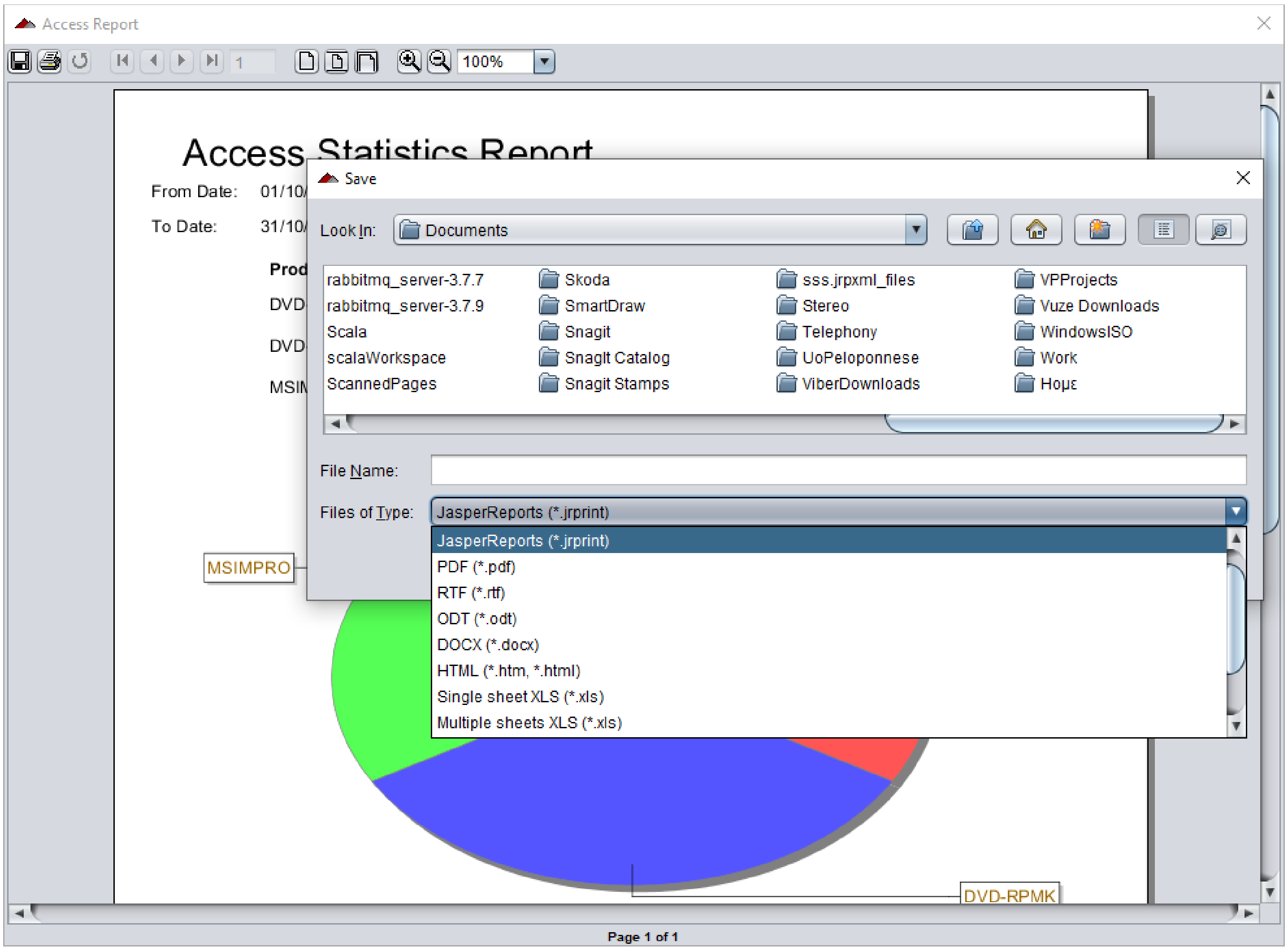
To Date: 31/10/19 Action Code: SelectCat.do

Product Code	Accesses	
DVD-ABUG	2	33.33%
DVD-RPMK	2	33.33%
MSIMPRO	2	33.33%

🟥 DVD-ABUG 2 (33%) 🟦 DVD-RPMK 2 (33%) 🟩 MSIMPRO 2 (33%)

FIGURE 53: ACCESS PER PRODUCT REPORT DIALOG

magnified, viewed and stored or exported in various formats, a few of which can be chosen by the “Files of Type” combo box of the Save file chooser dialog in Figure 54.


Access Report

Access Statistics Report

From Date: 01/10/

To Date: 31/10/

Save

Look In: Documents

rabbitmq_server-3.7.7	Skoda	sss.jrpxml_files	VPProjects
rabbitmq_server-3.7.9	SmartDraw	Stereo	Vuze Downloads
Scala	Snagit	Telephony	WindowsISO
scalaWorkspace	SnagIt Catalog	UoPeloponnese	Work
ScannedPages	SnagIt Stamps	ViberDownloads	Ηομε

File Name:

Files of Type: JasperReports (*.jrprint)

- JasperReports (*.jrprint)
- PDF (*.pdf)
- RTF (*.rtf)
- ODT (*.odt)
- DOCX (*.docx)
- HTML (*.htm, *.html)
- Single sheet XLS (*.xls)
- Multiple sheets XLS (*.xls)

Figure 54: Export Options For the Report

The designer editor of JasperSoft Studio, in Figure 55, shows the manipulation capabilities of the software package, along with the Eclipse based WYSIWYG interface. The lower left window consisting of a tree view of all data that will be inserted into the JRXML file, allows picking variables and fields and dropping them to the appropriate zone in the top center visualization of the final report. A visual form of the zones is available here.

The pie chart is positioned at the summary zone of the report. The summary zone includes data that appear only one time, at the end of the report.

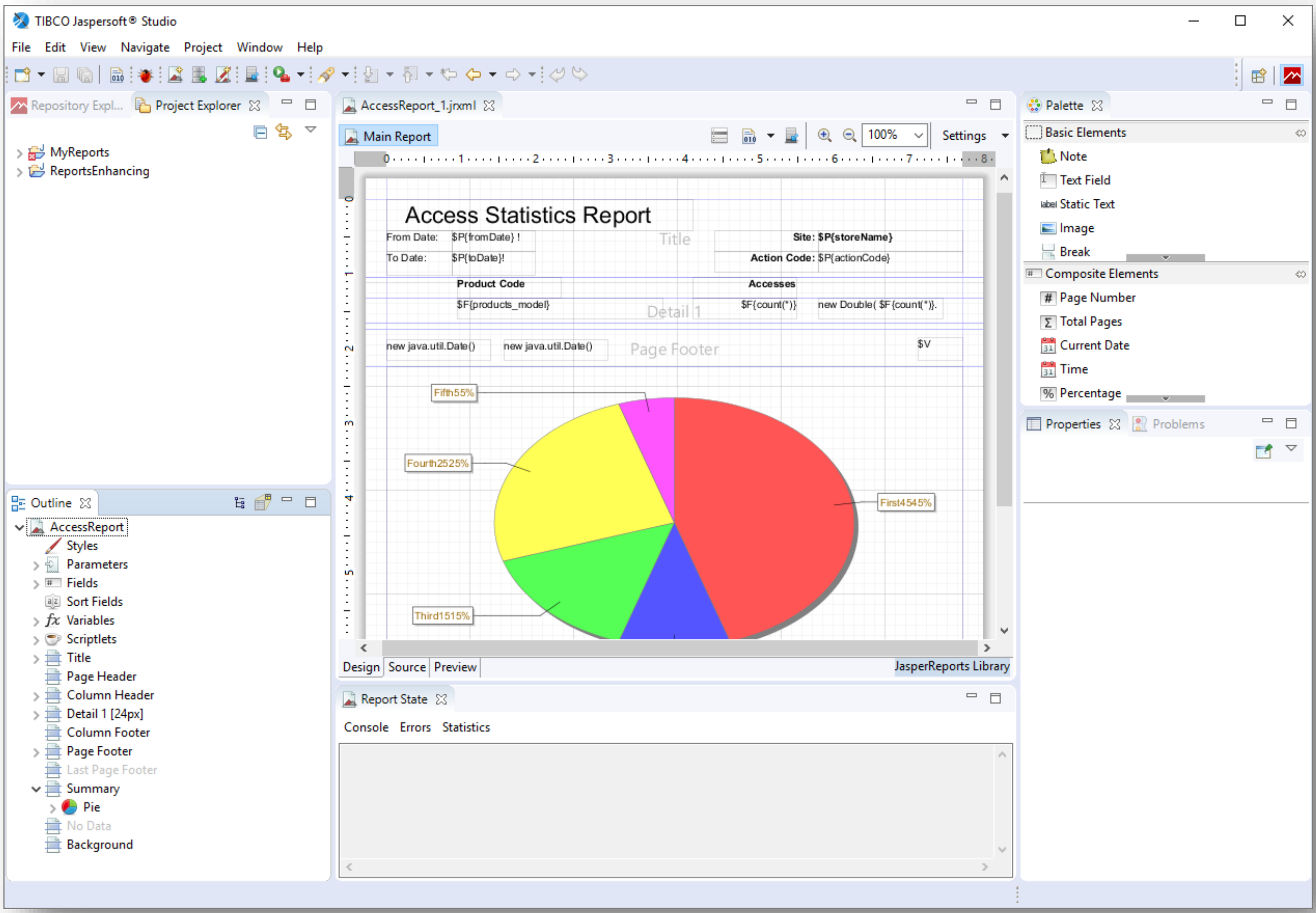


FIGURE 55: JASPERSOFT STUDIO EDITOR

Clicking on the pie chart image of the center window opens the Chart Data Configuration dialog (Figure 56) that allows defining the characteristics of the pie chart.

A further form of graphical results generated by the LFA is the Customer Behavior Model Graph, like in the one in Figure 40. The CBMG, as seen above offers some level of interactivity, in the sense that every node can be moved. The generation of GBMG is based on the JUNG2[30] library. JUNG is the abbreviation for Java Universal Network Graph library, which is an open source library facilitating network graph creation.

30 http://jung.sourceforge.net/

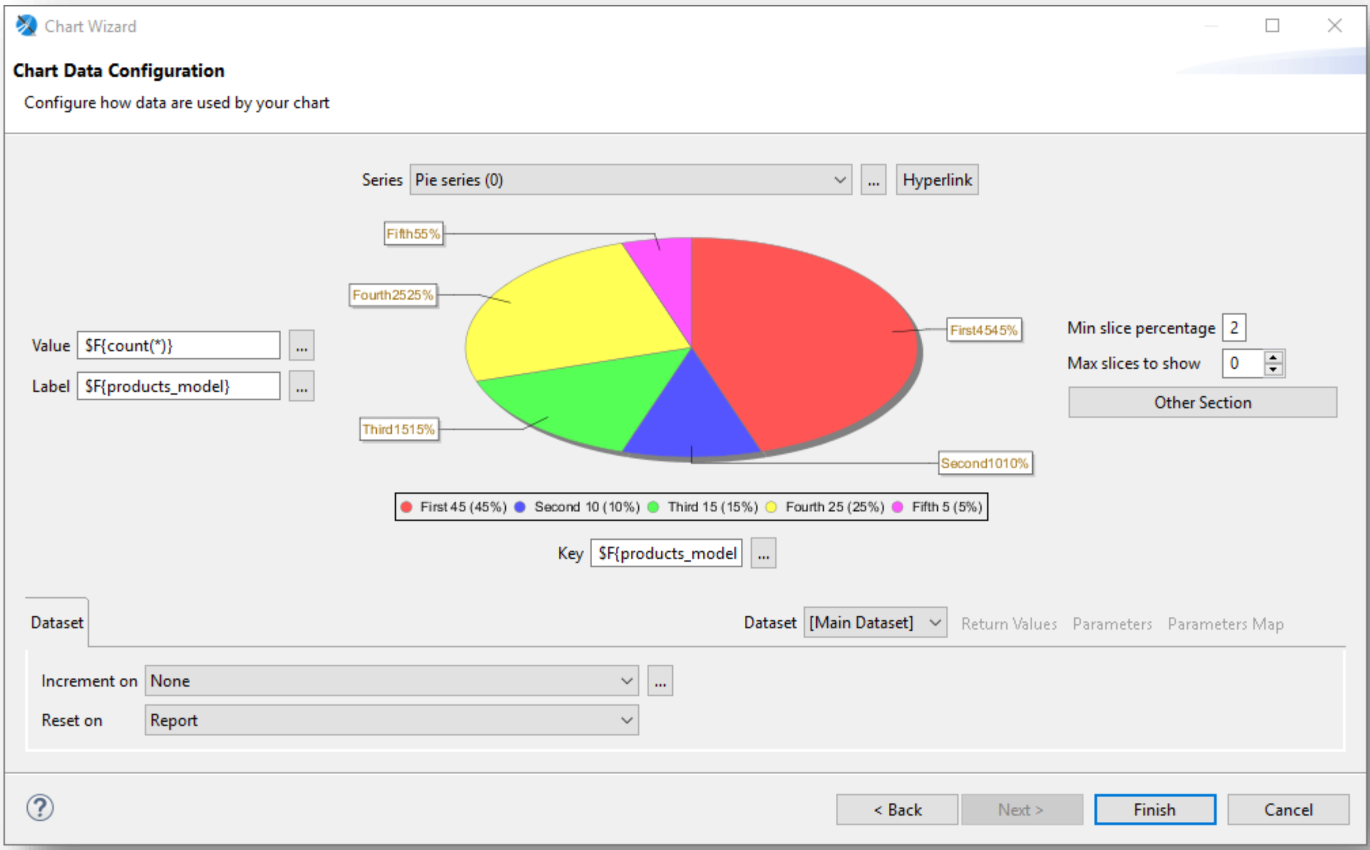


Chart Data Configuration

Configure how data are used by your chart

Series: Pie series (0)

Value: $F{count(*)}

Label: $F{products_model}

Min slice percentage: 2

Max slices to show: 0

Key: $F{products_model

Dataset

Dataset: [Main Dataset]

Increment on: None

Reset on: Report

FIGURE 56: CHART DATA CONFIGURATION DIALOG

The simplicity of the database design and the dataflows facilitate easy accommodation of any visualization library, like D3 (Data Driven Documents)[31], written by Michael Bostock and Gephi[32] for graphs with almost unlimited number of nodes and arcs.

[31] https://d3js.org/

[32] https://gephi.org/

8.14. Web Application

The web interface application for the analyzer is addon software, acting as an interface for external access, informing about the metrics and status of the hosted ecommerce sites. While the LFA is designed to support multiple ecommerce sites, the web application supports just one e-shop per instance. It is an autonomous web application, written in Java, based on the Google Web Toolkit (GWT) that uses the LFA API in order to import data. Web applications only need read permissions to the database of the LFA and can be physically located anywhere. As shown in Figure 57, the LFA can deal with more than one ecommerce sites. Several producers and consumers can do the feeding of interaction data in real time.

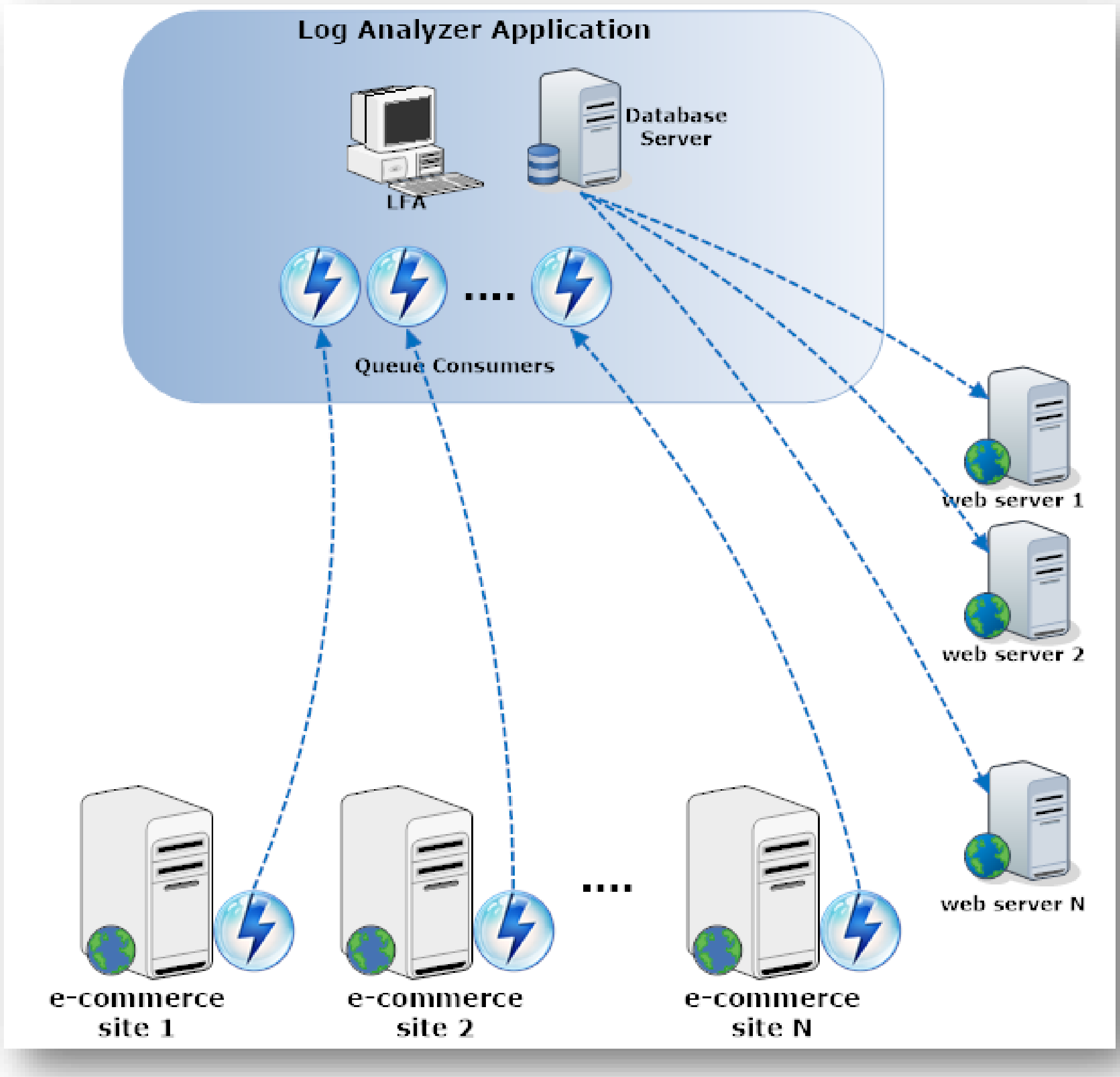


Figure 57: Web Server Applications Topology

Figure 57 shows the topology of the Analytics Web Application. While the LFA collects operational data from multiple e-commerce sites, each Web Application provides measurements and metrics just for one e-shop. Multiple web applications could also run on the same web server, provided they reserve different ports in order to avoid conflicts.

The web application is a rich Internet application based on Google Web Toolkit (GWT). GWT is a toolbox that allows creating responsive rich Internet web applications with emphasis on the client-side and full integration with the server.

Any GWT application is split in three parts:

- Server part
- Client part and
- Public or Shared.

The entire application is written in Java. While the server part is compiled to bytecode and runs on the local servers Java Runtime Engine (JRE), the client part of the software, which consists of the user interface, is compiled from Java into JavaScript. The client's machine needs no JRE to run it, since JavaScript runs directly on all browsers. The public part of the software is already in a form that can be deployed to the client upon request. The communication between server and client is established with asynchronous Remote Procedure Calls (RPCs).

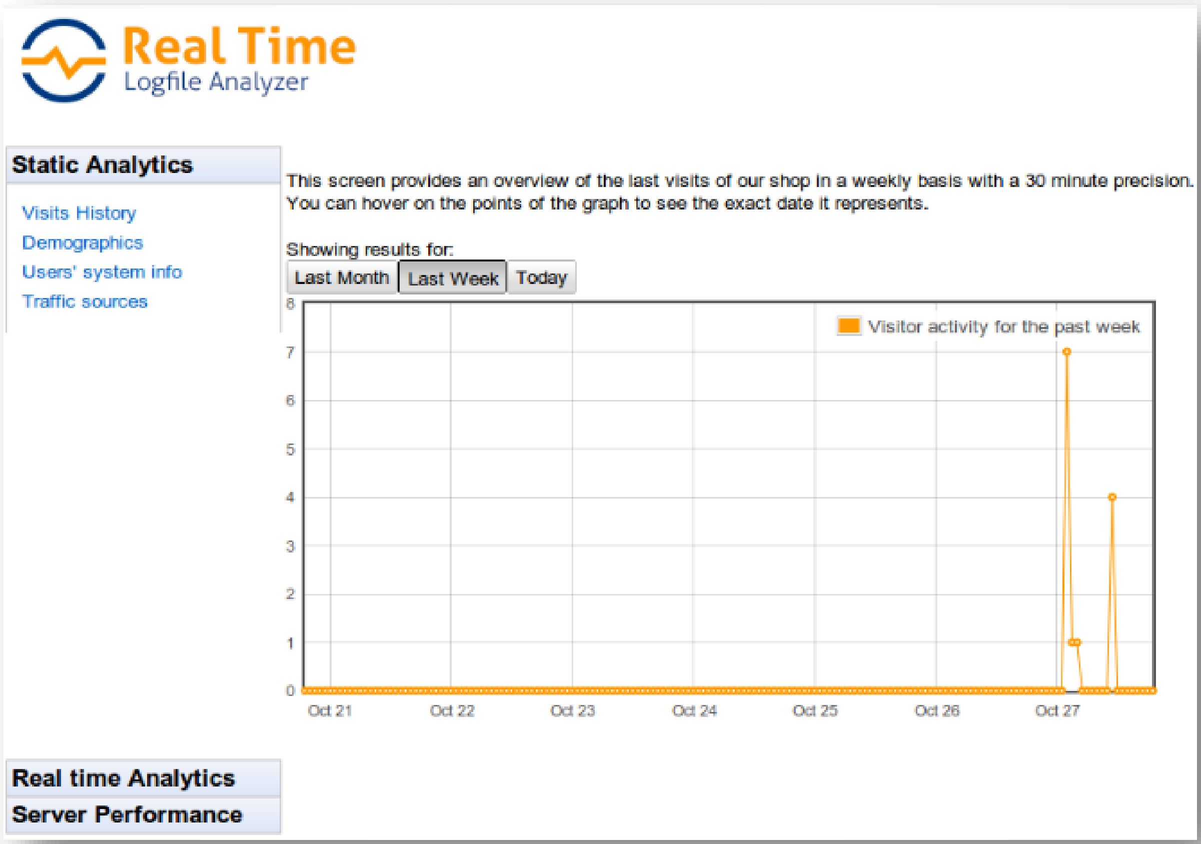
Since every web application of Figure 57 refers to one specific e-commerce site, the URL for the application leads directly to the metrics for this specific site. The interface of the application is based on a StackLayoutPanel[33] with 3 basic Menus: Static, Real Time Analytics and Server Performance. Static Analytics' MenuItems are: Visits History referring to visit activity graphs for the current day, last week and last month (Figure 58). The graph is generated with gflot[34]. Gflot is a GWT wrapper for the JavaScript library Flot[35]. This web application version was implemented by

[33] http://www.gwtproject.org/javadoc/latest/com/google/gwt/user/client/ui/StackLayoutPanel.html

[34] https://github.com/nmorel/gflot

[35] http://www.flotcharts.org/

John C. Aivalis[36] as part of the requirements for his Computer Science Bachelors' degree thesis for the University of Crete. It is based on the LFA API for fetching data.

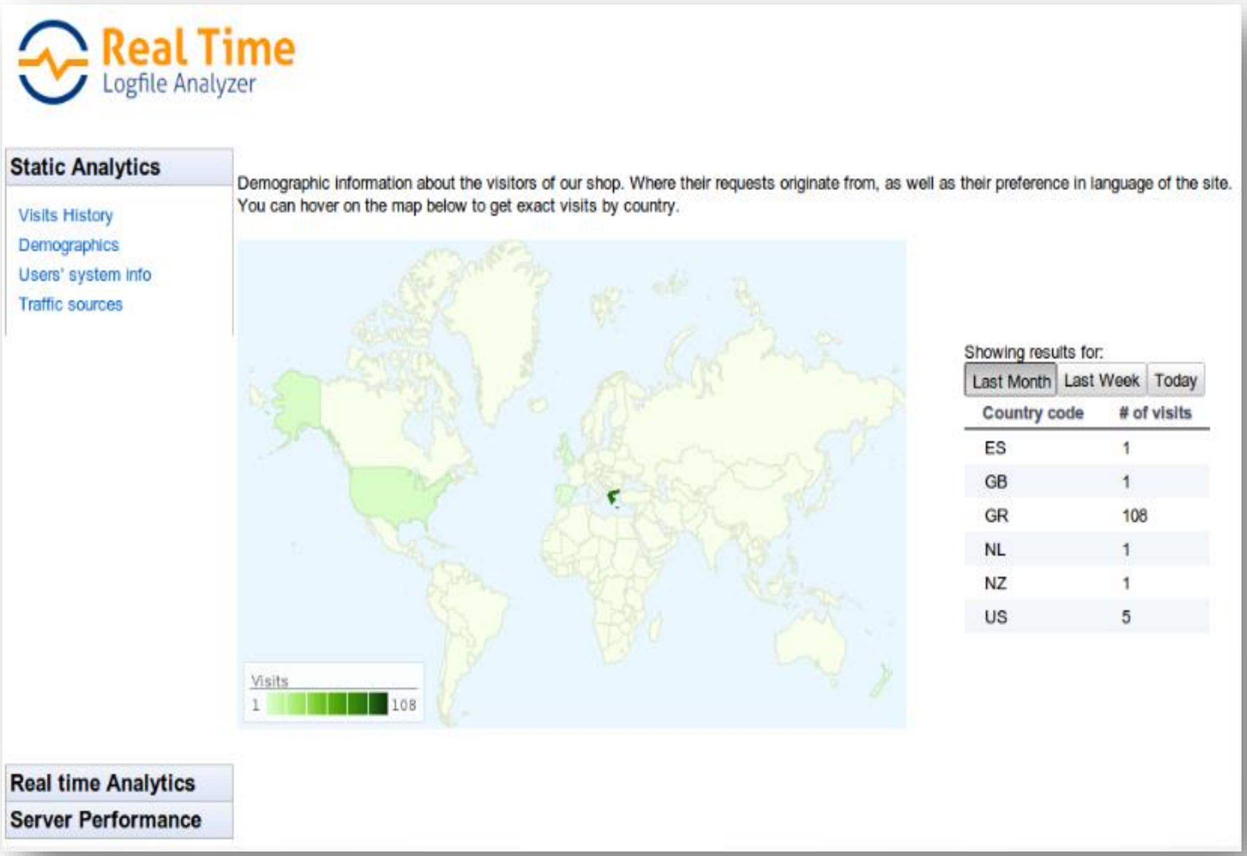


This screen provides an overview of the last visits of our shop in a weekly basis with a 30 minute precision. You can hover on the points of the graph to see the exact date it represents.

FIGURE 58: USER INTERFACE OF THE WEB APPLICATION

Figure 59 shows the geographical location of the visitors of the e-commerce site. The country approximation is found by the getting the canonical hostname of the IP address (`java.net.InetAddress`). The map is generated with the use of the GWT Visualization and GeoMap.

[36] johnaivalis@gmail.com

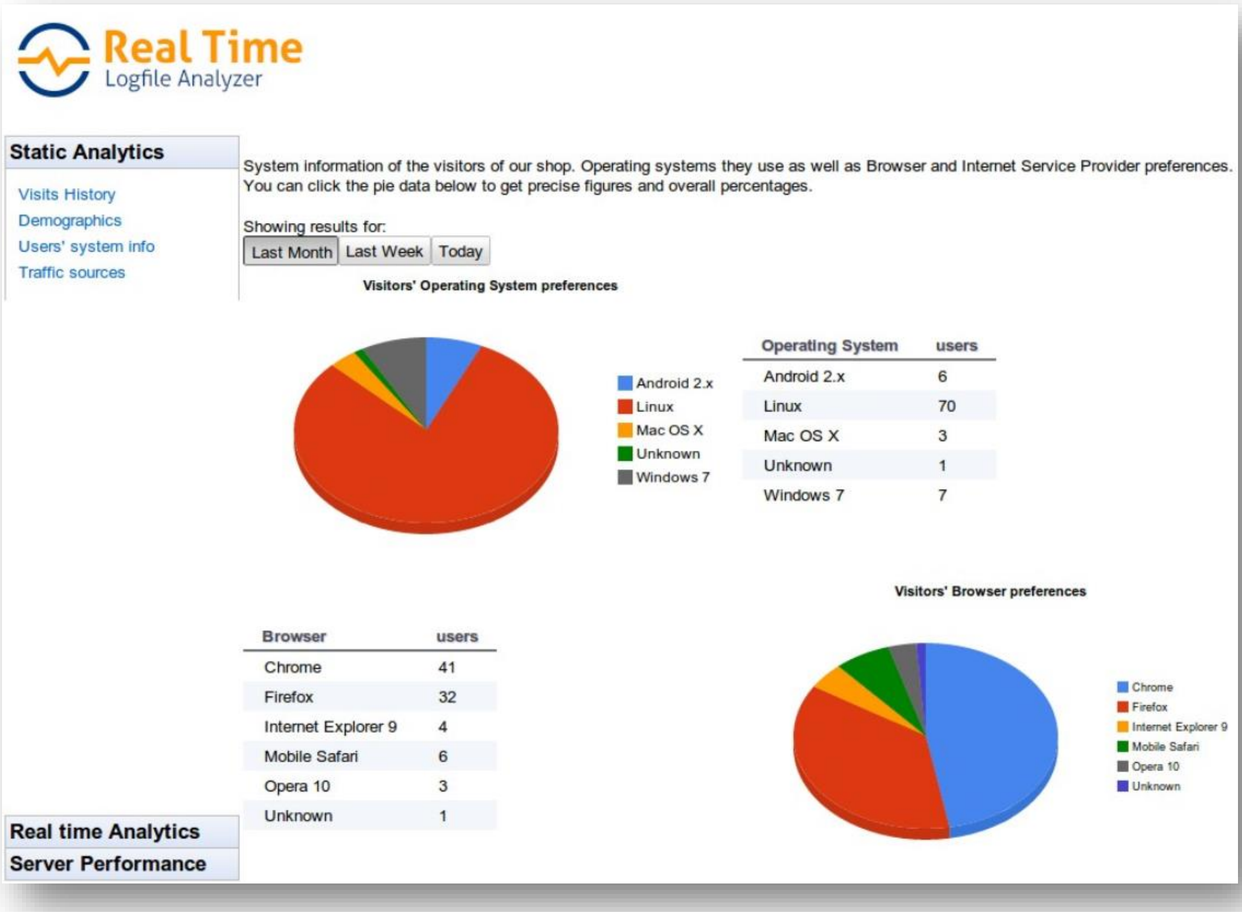


Demographic information about the visitors of our shop. Where their requests originate from, as well as their preference in language of the site. You can hover on the map below to get exact visits by country.

Country code	# of visits
ES	1
GB	1
GR	108
NL	1
NZ	1
US	5

FIGURE 59: DEMOGRAPHIC INFORMATION ABOOUT VISITOR LOCATION

Figure 60 shows the users system information displaying client environment information generated by activating the agent string of the access log file. We get Monthly, Weekly and Daily statistical measurements. The implementation is done with GWT Visualization and pie charts.



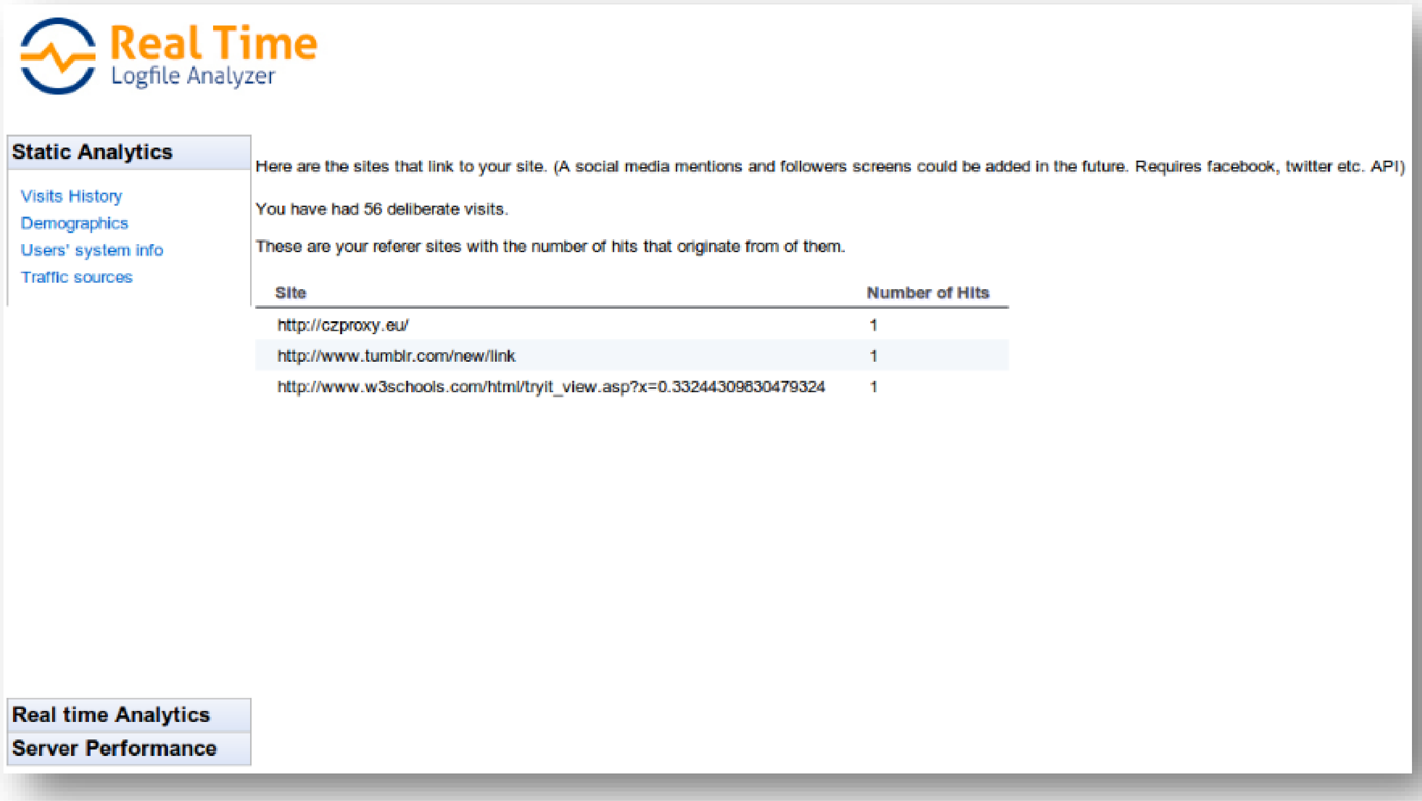
System information of the visitors of our shop. Operating systems they use as well as Browser and Internet Service Provider preferences. You can click the pie data below to get precise figures and overall percentages.

Operating System	users
Android 2.x	6
Linux	70
Mac OS X	3
Unknown	1
Windows 7	7

Browser	users
Chrome	41
Firefox	32
Internet Explorer 9	4
Mobile Safari	6
Opera 10	3
Unknown	1

FIGURE 60: SYSTEM INFORMATION

Next selection, 'Traffic Sources' shows referrers and number of visitors (Figure 61).



Here are the sites that link to your site. (A social media mentions and followers screens could be added in the future. Requires facebook, twitter etc. API)

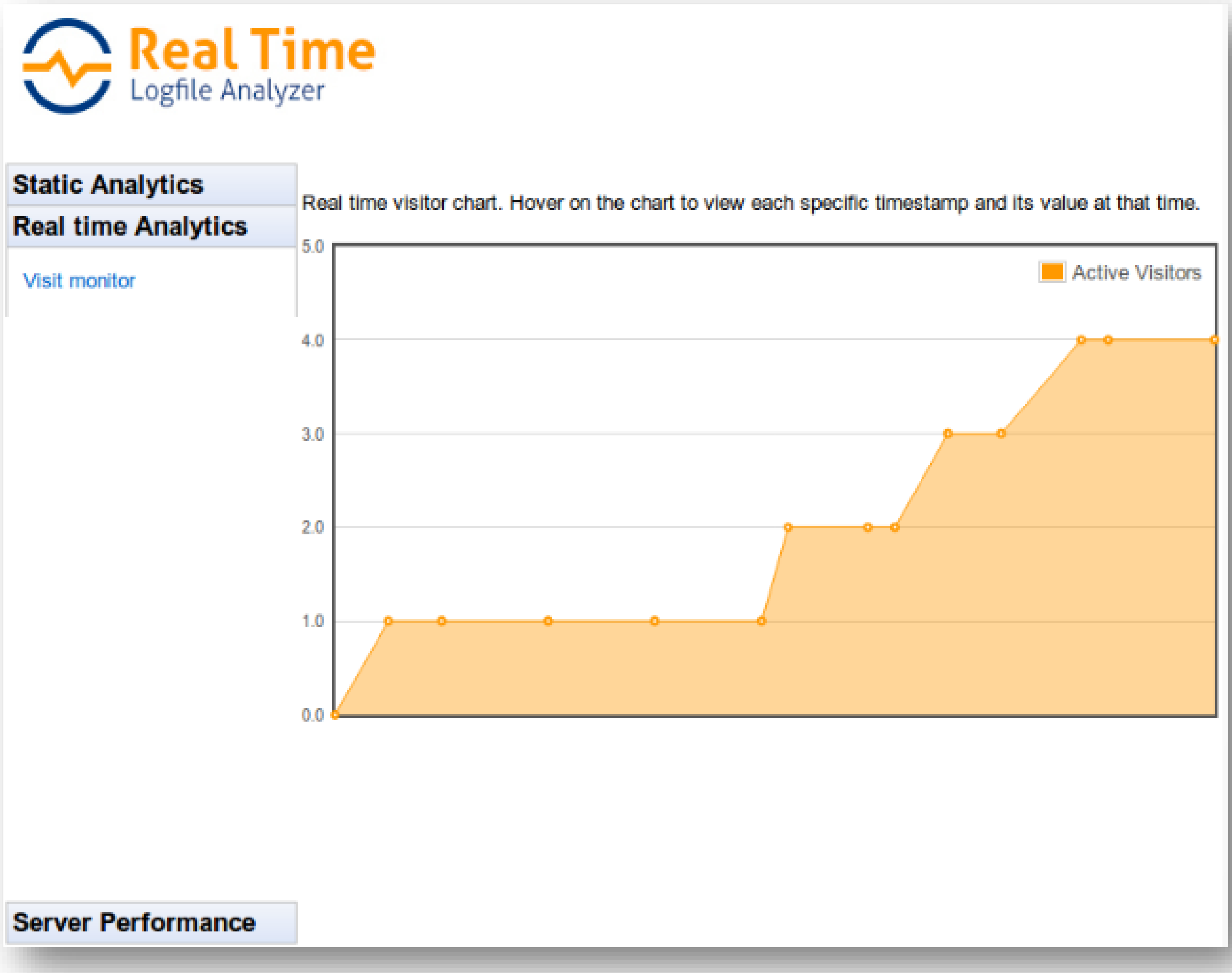
You have had 56 deliberate visits.

These are your referer sites with the number of hits that originate from of them.

Site	Number of Hits
http://czproxy.eu/	1
http://www.tumblr.com/new/link	1
http://www.w3schools.com/html/tryit_view.asp?x=0.33244309830479324	1

FIGURE 61: TRAFFIC SOURCES

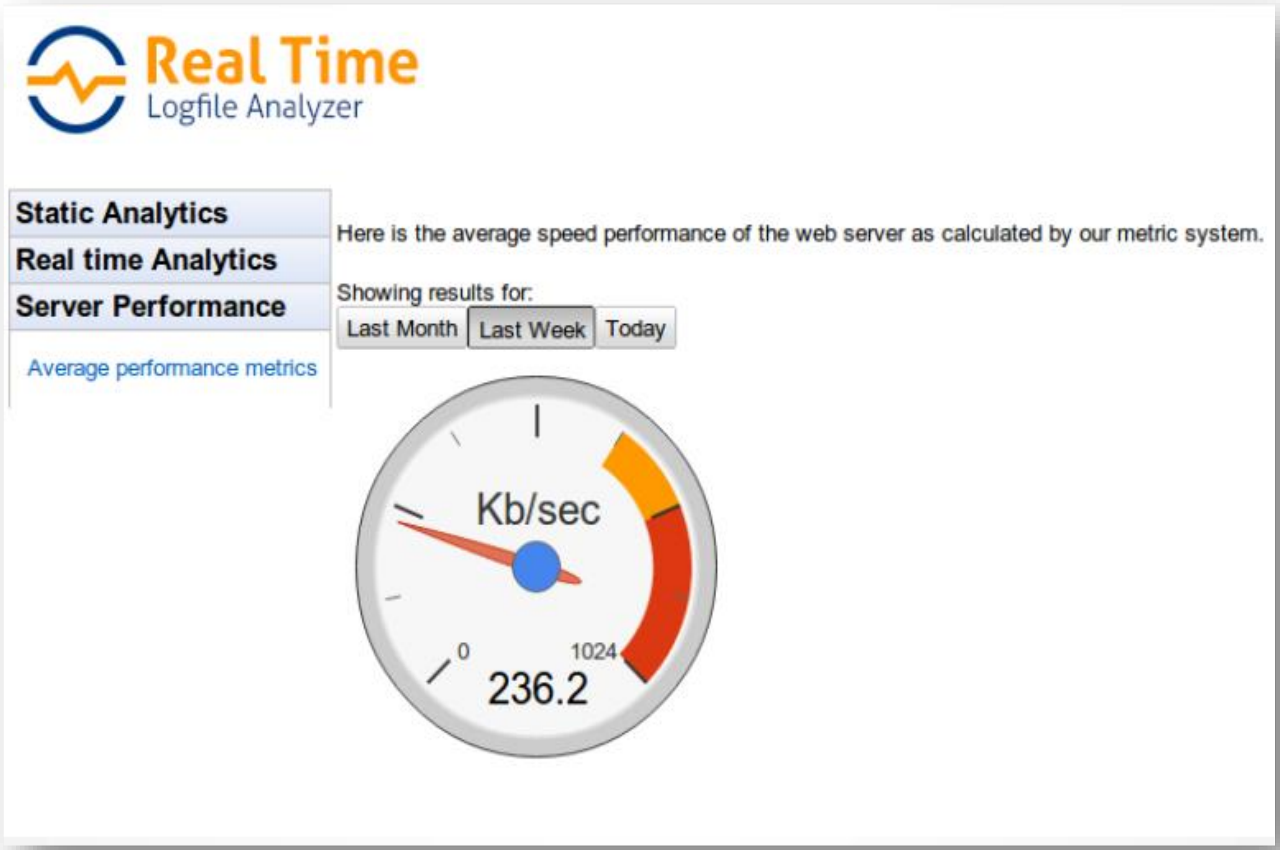
The Real Time Analytics menu displays a Gflot SlidingSimplePlot graph (Figure 62), that shows the number of active sessions. The graph is refreshed every 4 seconds and is fed by an AMQP queue implemented in RabbitMQ that measures the session ids.



Real time visitor chart. Hover on the chart to view each specific timestamp and its value at that time.

FIGURE 62: REAL TIME SESSION GRAPH

The Server Performance gauge (Figure 63) is implemented with the GWT Visualization Gauge widget. The calculation is done for a month, week or day and the metric is an approximation generated by dividing total bytes loaded by total time needed.



Here is the average speed performance of the web server as calculated by our metric system.

Showing results for:

FIGURE 63: AVERAGE SPEED GAUGE

The LFA application can easily be used as a data pool and a base for collecting data for any type of web applications that can promote it and display metrics.

8.15. Database

The main database tables maintained by the Analyzer are the following [63]:

- *Servers*: Stores information about the standard locations of different web servers supported. (log file directory, configuration file name and location etc.)
- *Site*: Table for the specific details of the various e-shops. Includes type of the server and details of its log file (like the example described in section III) must be stored here. It also contains the URL and IP-port, data base type and a username and password to access the items, categories and orders from the e-shop database.
- *Product Categories* for every e-shop (*)
- *Products* for every e-shop (*)
- *Orders* for every e-shop (*)
- *Order Statuses* for every e-shop (*)
- *Customers* of every e-shop (*)
- *Business-Actions:* (like "add To Cart" or "Pay" etc.) for every e-shop implementation. They are matched with the transactions. (*)
- Transactions: This is the table where every line of information from the log file is parsed into.

(*) imported from the e-shops.

The basic repository for logfile entries is the transactions table. The Data Definition Language command to generate it is:

```
CREATE TABLE `transactions` (
  `batch` char(50) NOT NULL DEFAULT '',
  `filename` varchar(256) DEFAULT NULL,
  `code` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `site` int(11) NOT NULL,
  `aRemoteIPAddress` char(50) DEFAULT NULL,
  `ALocalIPAddress` char(50) DEFAULT NULL,
  `bBytesSentOrDash` int(11) DEFAULT NULL,
  `BBytesSent` int(11) DEFAULT NULL,
  `hRemoteHostName` varchar(50) DEFAULT NULL,
  `HRequestProtocol` char(20) DEFAULT NULL,
  `lRemoteLogicalUsername` char(20) DEFAULT NULL,
  `mRequestMethod` char(20) DEFAULT NULL,
  `pLocalPort` int(11) DEFAULT NULL,
  `qQueryString` varchar(200) DEFAULT NULL,
  `rFirstLineOfRequest` varchar(300) DEFAULT NULL,
  `sHTTPStatusCode` varchar(50) DEFAULT NULL,
  `SUserSessionID` char(32) NOT NULL,
  `tDateAndTime` datetime NOT NULL,
  `uRemoteUserAuthenticated` char(60) DEFAULT NULL,
  `URequestedURLPath` varchar(120) DEFAULT NULL,
  `vLocalServerName` varchar(60) DEFAULT NULL,
  `DMillisToProcess` int(11) DEFAULT NULL,
  `TSecondsToProcess` int(11) DEFAULT NULL,
  `ICurrentReqThreadName` varchar(100) DEFAULT NULL,
  `isbot` tinyint(1) DEFAULT NULL,
  `bot_string` varchar(50) DEFAULT NULL,
  `agent` varchar(255) DEFAULT NULL,
  `referer` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`code`),
  UNIQUE KEY `code` (`code`),
  KEY `lineEntry` (`site`,`SUserSessionID`,`tDateAndTime`),
  KEY `batchKey` (`batch`,`site`,`tDateAndTime`),
  KEY `siteSession` (`site`,`tDateAndTime`,`SUserSessionID`),
  CONSTRAINT `transactions_fk` FOREIGN KEY (`site`) REFERENCES `site` (`code`)
) ENGINE=InnoDB AUTO_INCREMENT=167277 DEFAULT CHARSET=utf8
```

We have tried to keep the database design as simple as possible. The idea behind this is that we will eventually end up with a transaction table that needs fewer joins for processing.

The entity relationship diagram is shown in Figure 64. This is a MySQL database based on the InnoDB storage engine.

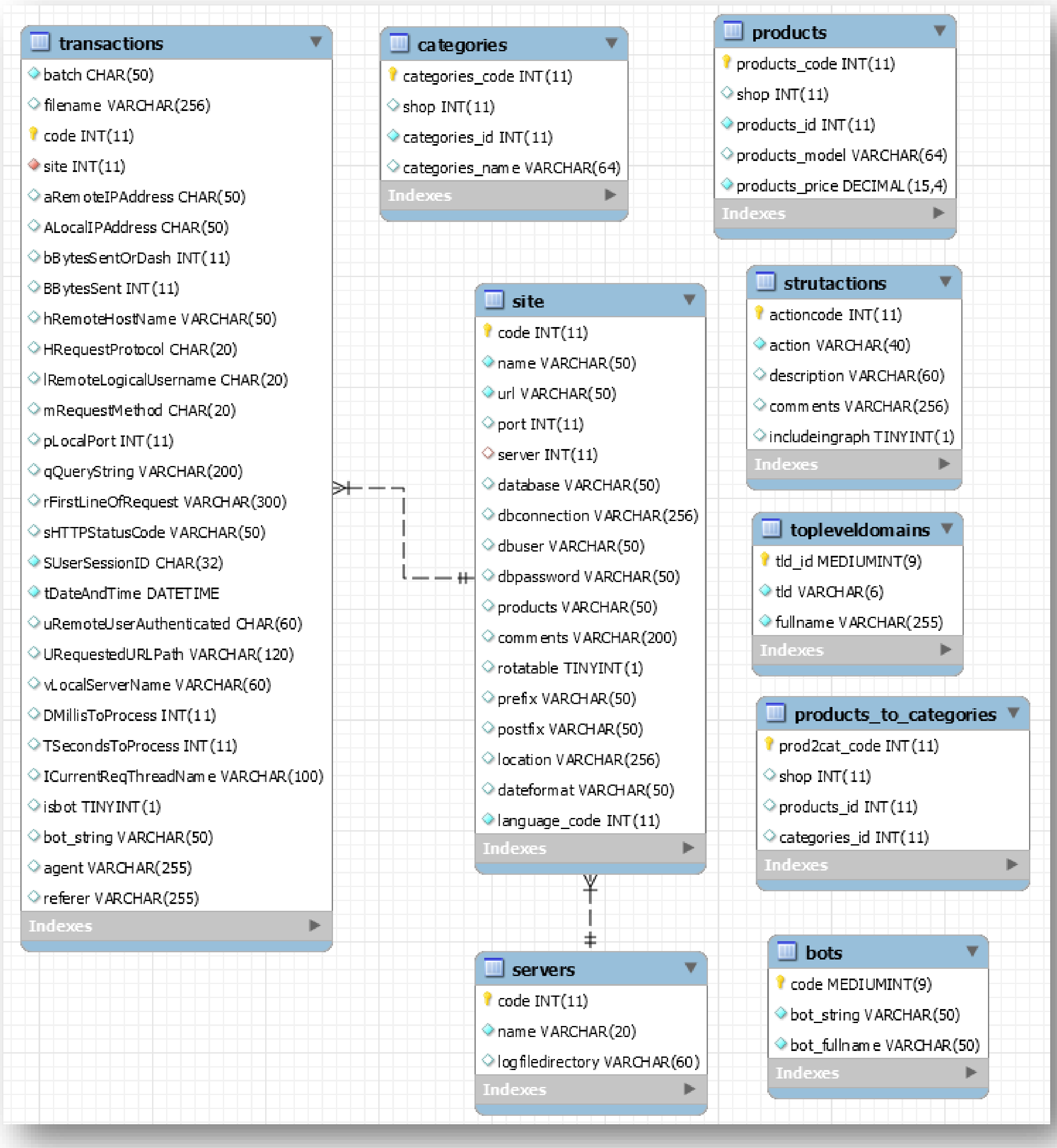


FIGURE 64: ANALYZER DATABASE ERD

The attribute names of the columns that are used for storing data from the logfile start with the letter of the access log valve code that generated it. For example, mRequestMethod for the %m output, tDateAndTime for %t etc.

The batch attribute allows grouping batches of logfile lines by name. These names enable quick selection and deletion when there is no more need for them.

The structure of the database schema allows simple queries for easy retrieval of information. Sample SQL queries return:

1. The number of entries for a specific site:

```
SELECT count(*) FROM transactions WHERE site=15;
```

2. The number of sessions for an e-shop and a date span:

```
SELECT count(distinct SUserSessionID) FROM transactions
WHERE site=15 AND
tDateAndTime >= '2018-01-01' AND tDateAndTime <= '2018-12-31';
```

3. The number of bytes transferred for an e-shop and a date span:

```
SELECT sum(BBytesSent) FROM transactions WHERE site=15 AND
tDateAndTime >= '2019-10-24 19:00' AND tDateAndTime <= '2019-10-24 19:59';
```

4. The different HTTP status codes for a period:

```
SELECT distinct sHTTPStatusCode  FROM transactions WHERE site=15 AND
tDateAndTime >= '2019-10-24 19:00' AND tDateAndTime <= '2019-10-24 19:59';
```

5. The total number of CSS files loaded over a period:

```
SELECT count(*) FROM transactions WHERE site=15 AND rFirstLineOfRequest LIKE
'%.css%'
AND tDateAndTime >= '2019-10-24 19:00' AND tDateAndTime <= '2019-10-24 19:59';
```

6. The total size of CSS loaded files:

```
SELECT sum(BBytesSent)  FROM transactions WHERE site=15 AND
rFirstLineOfRequest LIKE '%.css%' AND tDateAndTime >= '2019-10-24 19:00' AND
tDateAndTime <= '2019-10-24 19:59';
```

7. How many 'AddToCartFromProdId.do' requests were made for a specific site within a time range:

```
SELECT count(*)  FROM transactions WHERE site=15 AND rFirstLineOfRequest LIKE
'%AddToCartFromProdId.do%' AND tDateAndTime >= '2018-01-04 00:00' AND
tDateAndTime <= '2018-05-12 20:59';
```

8. The ids of the four most frequently added to cart products of one e-shop in a date range:

```
SELECT count(rFirstLineOfRequest) AS cnt, (rFirstLineOfRequest) FROM
transactions
WHERE site=15 AND rFirstLineOfRequest LIKE "%AddToCartFromProdId.do?prodId%"
AND
tDateAndTime >= '2018-01-04 00:00' AND tDateAndTime <= '2019-05-12 20:59'
GROUP BY rFirstLineOfRequest ORDER BY cnt DESC LIMIT 4;
```

9. Bot entries. The value of the TinyInt attribute named “isBot”, is 1 when the log file line was produced by a known bot. Next query measures the number of such lines:

```
SELECT count(*) FROM transactions WHERE site=15 AND isBot=1 AND
tDateAndTime >= '2019-01-04 00:00' AND tDateAndTime <= '2019-01-31 23:59';
```

The system can extract details and information using SQL queries. These queries can be embedded into the report generation system and extend the LFA.

The next section explains Big Data Technologies, lists the important players and technologies of the big data ecosystem and explains how they eliminate the data congestion problem that lingers behind the long-term operation of any application collecting data. A brief history of the conditions that lead to BD, a description of the basic problem and definitions. Hadoop and MapReduce, used dealing with volume of data, are introduced along with the overall Hadoop Ecosystem composed of applications involved for solving various tasks. The NoSQL and NewSQL database management systems are explained. Modern technologies used in order to remedy data velocity and the Spark ecosystem and available vendors are presented.

9. BIG DATA TECHNOLOGIES

The "Big Data Technologies" section lists the important players and technologies of the big data ecosystem and explains how they eliminate the data congestion problem that lingers behind the long-term operation of any application collecting data. A brief history of the conditions that lead to BD, a description of the basic problem and definitions. Hadoop and MapReduce, used dealing with volume of data, are introduced along with the overall Hadoop Ecosystem composed of applications involved for solving various tasks. The NoSQL and NewSQL database management systems are explained. Modern technologies used in order to remedy data velocity and the Spark ecosystem and available vendors are presented here.

Big Data has emerged as a new technological paradigm during the last few years, because of the need to master the occurring exponential growth of data. Big Data technologies offer toolboxes in form of frameworks that deal with the data explosion created by the ever-growing number of applications, mobile devices, sensors and the Internet of Things (IoT) in conjunction with the wish to have a better overview, receive answers to questions and measure behavior and operational complexity of today's systems.

Big Data refers to large datasets and dataflows whose processing lays beyond the capabilities of traditional information systems and databases. Information like log files, images, messages, transaction records from remote or local application databases, composite distributed data structures, sensor data from remote devices, data from public databases and IoT devices can be used selectively to enrich existing data to provide clear operational insight and support the recognition of trends and tendencies.

Very often, data generated in social media applications are used to measure or extend the impact of specific Internet campaigns for products and services. Big Data is the toolkit that targets the infamous "three V's" of data, which comprise the three basic characteristics: Volume, Velocity and Variety.

This chapter will explain the basic definitions of Big Data components, provide a list of the technologies used and vendors involved and show how the 3 Vs can be applied on hand of application examples.

9.1. Introduction To Big Data

Big Data issues have been around since the very early years of the Informatics era, although programmers have preferred to avoid dealing with them systematically, because the technologies that allow processing them conveniently have only become available during the last ten years. As a first example, the first Universal Product Code (UPC code) was read by a retail cash register in 1974. Since then, supply chains have been also using Radio Frequency Identification (RFID) to automate the identification and tracking of objects by attaching tags to them. Scanners have since then produced huge quantities of data, that have only been exploited partially and always in a structured manner, according to the standards and rules of traditional Information Systems, designed in the 70's era, basically aiming to generate typical invoices and printed statistical batch reports.

Contemporary software applications manage and generate very large quantities of data. Web based components and applications support online transactions, generate logfiles and feed analytics applications. Social Media applications also produce daily petabytes of Big Data. Portions of this data are publicly accessible, concerning companies and organizations and can be analyzed and used in order to target specific groups of customers, automatically generate customized advertisements for products and services and measure their impact by evaluating the feedback and acceptance they receive. Retailers can make smart offers and recommend specific products to target customers better. This way any data, structured or unstructured concerning any company or organization can be evaluated and this can greatly enhance the insight of the clientele base and reveal how they shop and how they react when they see ads of specific products.

The vast storage capacity of Big Data file systems allows also historical operational data, backups and old log files, generated by Information Systems and web servers over periods spanning many years to become directly accessible online instead of being stored in external magnetic or optical media. The drastic cost reduction of storage and commodity hardware, in conjunction with the free software revolution and the wide acceptance of open source software applications and operating systems, have allowed establishing reliable and simple to maintain low cost Big Data platforms.

The amazing penetration of the Internet in all branches and the geographical dissemination due to today's networking advances, in conjunction with mobile telephony, smart phones, mobile devices and sensors form the foundation for Big Data. As Sun Microsystem's Scott McNealy pointed out in the mid 80's: “The network is the computer”.

Analytics is based upon growing logfiles and large data collections. Since big data can break size and performance barriers and limitations of data processing it can be very useful for dealing with very large e-commerce sites with very large numbers of users.

9.2. Definition of Big Data

There is no unique definition for Big Data. The term Big Data means different things to different people. Everybody agrees though that Big Data starts where traditional On-Line Transaction Processing (OLTP) or On-Line Analytical Processing (OLAP) systems become too slow, or even start to fail due to data volume, data velocity and data variety. The simple substitution of a slow computer system with a more powerful one, known as vertical scaling, most often will not solve the problem, and if it does, then only temporarily, until the next bottleneck appears, that would call again for a new more powerful system.

Grace Hopper, the legendary United States Navy rear admiral and designer of the COBOL language, described scaling, many years ago, like this: "If one ox could not do the job, they did not try to grow a bigger ox, but used two oxen. When we need greater computer power, the answer is not to get a bigger computer, but… to build systems of computers and operate them in parallel" [64]. This approach is known as horizontal scaling. Big Data is based on parallelization and distribution of processing and file storage.

In parallel computing several processors are sharing the same memory system, while in distributed computing we have groups of computers sharing the same goal, where each processor is operating on its own memory. Information must be exchanged through message traffic between the nodes. The concept of distributed computing has been around since the introduction of the Ethernet in the 70's, when multiple network architectures, like peer-to-peer, n-tier and client-server and coupling models have been implemented. Algorithms have been developed, improved and established over the years that guarantee fault tolerant and efficient cooperation, simple installation, easy setup and expansion. The concept of using distributed and parallel systems is based on splitting a large problem into smaller ones and letting each of the processors individually carry out its part of the job. We must divide a big dataset into small partitions, run the appropriate algorithms on the individual machines and finally combine the output into one. This is a complicated procedure and the developer must organize the partitioning scheme of the data manually, as well as taking care of the appropriate communications that are required. Hadoop, for example, can deal with all non-trivial logistics involving interprocess communications, making a complicated Distributed File System, expanded over large clusters of cooperating computers, seem like a huge local file system to the developer.

While **Volume** deals with data size, **Velocity** deals with speed of generation, streaming and data collection techniques that allow processing and visualization in near real time and sometimes even in real time. **Variety** is the component that deals with the mapping of different classes and structures of data into concrete objects, so they can function together [65].

Four additional V-buzzwords complete the Big Data component landscape. These stand for: **Veracity**, **Variability**, **Visualization** and **Value**. Veracity is established by creating mechanisms that check the accuracy of data and ensure the overall correctness of the data collection. Variability refers to possible inconsistencies of data that must be sorted out by interpreting their exact context. Visualization tools allow presenting and summarizing selected portions of data in either static or interactive mode, leading to insights and views that allow understanding the nature of problems and the status of the operations generating Value. Although the last four V words are often used for big data, along with even more V-words like **Viability**, according to Doug Laney from Gartner Inc., they are not definitional specifically for big data and they apply on all types of data and not just on big data [66].

9.3. Hadoop and MapReduce Expedite *Volume*

Big Data technologies rely on distributed computing. The tool used as the foundation for most Big Data applications today is Apache Hadoop. Hadoop is not a single tool, but an entire ecosystem of applications. The base of this ecosystem is the Hadoop Distributed File System (HDFS), which allows clusters of commodity computers to act as a single storage component. Hadoop was designed in 2006 by Douglas Cutting for Yahoo!. Hadoop became an Apache project in 2008. HDFS is scalable and can grow on demand to hundreds of petabytes, simply by adding new DataNodes to our network. Very large companies own local area networks supporting HDFS, consisting of more than a thousand computers as DataNodes. Data in HDFS is organized in large sized blocks. The typical block size of the HDFS is 64MB. This size is huge when compared to the block size of 512 bytes, 4K or maximum 32K, usually used for the hard drives of regular local file systems.

Whatever data we store to the HDFS is automatically distributed to the DataNodes. The NameNode knows where the data blocks are located physically. Additional, replication of the data blocks takes place transparently and data blocks are spread among the physical DataNodes. This way, not only huge capacity, but also high levels of data integrity are guaranteed.

The fact that all complexities of distributed processing are automatically taken care of by the file system, leading to a simple programming model, made HDFS popular.

MapReduce (MR) is an algorithm that has traditionally been used for dealing with data stored in HDFS. MapReduce, originally generated in 2004 by Dean and Ghemawat in Google, was originally used for the Google File System (GFS). It is implemented in Java, and became an Apache project as well, along with Hadoop.

MapReduce is an algorithm design pattern that originated in the functional programming world. It consists of three steps. First, you write a *mapper function* or script that goes through your input data and outputs a series of keys and values to use in calculating the results. The keys are used to cluster together bits of data that will be needed to calculate a single output result. The unordered list of keys and values is then put through a sort step that ensures that all the fragments that have the same key are next to one another in the file. The reducer stage then goes through the sorted output and receives all values that have the same key in a contiguous block [67]. Programs written in this functional style are automatically parallelized and executed by a cluster of commodity machines.

These steps can be easily implemented and adapted to the needs of any search application. MR was designed by Google to count words, URL access frequencies and user counts [68]. The advantage of MR running on HDFS DataNodes lays in the fact that parallelism extremely enhances the speed of computations in comparison with monolithic architectures when searching large data sets.

As mentioned above, HDFS is a scalable file system, consisting of clusters of commodity computer systems (nodes) that operate as a single unit. This file system automatically supports replication for integrity reasons, making any use of Redundant Arrays of Independent Disks **(**RAID systems) completely unnecessary for all DataNodes. Still RAID is necessary for the NameNode [69].

The reason why HDFS uses such a very large block size (64 or 128 MB), is in order to keep the workload of the NameNode low. Hadoop is designed mainly for MapReduce applications. It is not particularly suited for supporting neither small files nor frequent updates of data.

Hadoop is constantly enriched with various projects and is always undergoing improvements. It supports distributed file storage and provides mechanisms that can easily be engaged with no need for manual intervention for segmentation, partitioning and collection of results. The Hadoop 2.0 Ecosystem was enhanced with Apache YARN (Yet Another Resource Negotiator). YARN allows better job scheduling and performance. YARN allows horizontal scaling to thousands of nodes and enhances security, by enabling auditing. YARN allows HDFS to support programming models beyond MapReduce and eliminate restrictions. [70]

YARN provides one resource manager RM per cluster and a standby RM, that consists of a scheduler and an application manager. The resource manager knows the location of the data nodes and their resources and manages the node managers of the cluster, that offer resources to the RM.

Hadoop is designed for batch processing and MapReduce provides a simple approach for the user, to generate metrics and do research among huge amounts of data, without the need to care much about the exact physical location of the data. The HDFS acts as a single file system and takes care of all the complexity involved in a distributed and parallel system. Traditionally Hadoop has been used for mining large quantities of historical data or for log file analysis in batch mode. In batch mode the users start a MR program and receive its results a few minutes later.

9.4. NoSQL Databases Promote *Variety*

The traditional model for database management systems that has been used for many decades has been the relational model. The S in the abbreviation SQL stands for structured. Relational databases are called structured databases today, because they deal with structured data. In the relational model, data is stored in tables. Tables have predefined columns, known as attributes and records are collections of attributes, stored in each row, always containing predefined columns according to the data definition. Null values can substitute non existing values, but every row always consists of all attributes predefined by the designer during the creation of the table. The relational model is very close to the way people tend to think about data. Using tables has been a very familiar way of organizing information. Relational database management systems offer indexing capabilities, allow customized views of the database, in concordance with the user, they support normalization in order to reduce redundancy and offer ways of defining constraints and attributes referencing to additional tables.

Relational database systems offer Atomicity, Consistency, Isolation and Durability and are thus called ACID compliant. This means that they allow valid transactional processing, even in the event of failures and errors. Atomicity means that any transaction consisting of multiple steps is considered successful only if all steps succeed. Only data that respect all predefined rules are considered Consistent. Isolation is achieved when transactions do not interfere with each other's data. Durability means that no data is lost even if a power failure occurs during processing. Very fast data insertions, selections, deletions and updates are guaranteed when operating within the limitations defined by each RDBMS vendor. Size can be a serious problem, especially when we try to use relational databases with Big Data, since structured databases can mainly scale vertically. There are even relational database engines, that allow storing database data straight into dedicated disk raw partitions for better performance, like Sybase for example, which later became a part of the SAP Corporation. This approach first appeared in the early 80's, and is very fast, since it avoids the overhead of the file system. Still the issue with RDBM is volume.

Hadoop has alleviated the restriction of vertical scalability for a file system. In order to lift this restriction also for databases, schema less NoSQL database systems have been developed. NoSQL stands for "not only SQL". NoSQL databases do not have table structure, nor fields and are not relational. They scale horizontally and can operate well on HDFS. We have seen so far that by scaling horizontally, NoSQL databases support Volume. The reason they solve the Variety issue, lies in their nature. Since they are not restricted to supporting predefined structures and data format, they are excellent tools for storing and retrieving semi-structured and unstructured data.

In MongoDB, for example, the Document substitutes the row of a structured database table. The Document has a JSON-like (JavaScript Object Notation) key-value format. Documents are included in collections. Each Document inside a Collection can have its own key-value set and the format can become very complex if necessary.

Different types of NoSQL databases exist:

- Wide Column Store Databases (HBase, Cassandra)
- Cache Systems (MemcachedDB, Redis)
- Key-Value Stores (Couchbase, Redis)
- Document Store Databases (CouchDB, MongoDB)
- Graph Databases (Neo4J)

NoSQL databases are highly scalable, fault tolerant and distributed, but offer no ACID compliance. A newer category of database management systems, known as NewSQL act as hybrid systems, since they fully support SQL and the relational data model and at the same time, they can run on HDFS, offering the speed, flexibility and scalability of NoSQL. They even support online transaction processing (OLTP). NewSQL databases often use Sharding, based on a middleware layer that splits huge relational database tables into manageable sized pieces (shards) that are stored on different nodes of HDFS, making horizontal scaling automatically feasible without the need to do any additional programming and form a transparent consistent distributed system for the database programmer and user. A Shard Map Manager (SMM) keeps track of the location of the shards and is responsible of coordinating all communications with the clients.

NoSQL database management systems are gaining in importance. There are many systems to choose from[37].

Database management systems are constantly evolving. The latest very promising type are the in-memory database management systems (IMDBMS). These database management systems take advantage of the latest advances of random-access memory, where capacity, speed and moderate to low pricing allow cheap commodity machines to often have 64, or even 128 Gigabytes of RAM or more. By keeping large portions of their data in memory there is a significant latency reduction and performance boost. They also cooperate very well with Apache Spark, which is also heavily memory bound. Spark will be described later in this chapter.

9.5. Hadoop and *Velocity*

A Hadoop File System can serve petabytes of data. According to Forbes[38], “the data we produce at the current pace every day are 2.5 quintillion bytes. This number is growing exponentially. 90% of the existing data was generated during the past two years”.

[37] http://nosql-database.org/

[38] https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read

Mobile phone apps use the various sensors of the device and generate useful data that is being streamed perpetually to data centers hosting specialized applications, for storing, processing, summarizing, analyzing and visualizing. Back in 2013, for example, a Danish app called "Endomondo" managed to build a global sports community with around 20 million users, growing to over 25 million users by 2015[39], that engage in sports activities. Every route of a biker or a jogger that provides GPS access to the app is registered in near real time and the central application provides statistics about speed, distance, slope and registers the activity on the map. The application gives calorie advice to each athlete individually acting as a personal trainer. Historical data, statistics and details can be retrieved for every registered activity. The activities can be shared on Facebook along with camera pictures and details of the route. Endomondo provides rich data of various formats and performs well even with low speed mobile data connections. Velocity is here one of the primary key factors for the success of the application.

A huge fast-growing source of streaming data is also the Internet of Things (IoT), where large numbers of sensors send and receive data. With social media, some 2.7 billion active Facebook users upload enormous amounts of pictures, text and messages every day. Other social media applications, like WhatsApp, Tumblr, Twitter, Quora, LinkedIn operate applications that support heavy communications of vast numbers of members, use fast peer to peer messaging, allow storing and streaming of music and videos and offer advanced application programming interfaces that are often embedded in many websites and are used in conjunction with various contexts.

Velocity deals not only with the speed of data acquisition, but also with the speed of data processing, in order to allow using and visualizing the information. Data, streaming into an application, should be consumed without creating congestions and large queues. When critical events generate data, the propagation of the event description should take place as soon as possible and alert the business in order to take the appropriate actions.

Data velocity applies to all kinds of industries and operation workflows. A web site, operating an e-commerce application needs analytics details in order to measure performance speed, see the load in terms of disparate sessions and measure turnover and sales in near real time. For scrutinizing the sales figures in this example, a microservice can be used, that acts as a data producer. It may be triggered by the database management system after a sale has been concluded, it collects all additional information necessary from involved tables of the database and generates some form of a message. This message is serialized, pushed into a queue and the communication middleware makes it available to

[39] https://nordicgrowthhackers.com/from-zero-to-25m-users-know-your-purpose-and-grow-with-it/

the consumer for further processing and near real time visualization. If the information arrives on time the management can use it in order to take actions, view the current status, access options and eventually correct bad decisions.

Velocity means high speed of data acquisition, storage, and processing with low latency and responsiveness whenever data retrieval and visualization of results is considered critical.

Several frameworks are available in the Hadoop ecosystem for controlled data streaming in order to support velocity.

Today many traditional Hadoop or, Big Data Applications are based on the MapReduce algorithm. Hadoop is being conquering the world as a distributed parallel batch file system, using MapReduce as its “standard” engine computing intensive algorithms over clusters. Although almost 15 years have passed since the first Hadoop - MapReduce applications were launched, this software combination is still largely in operation. This happens not only because Hadoop managed to break the barrier of storage space, using a simple model for the user, but also because of the huge software ecosystem that was developed on top of it that deals with almost every aspect of the Big Data needs of the corporate world.

Hadoop/MR developers have managed so far to deal well with security, software complexity and installation maintenance issues of the software. However, MR responds relatively slow to queries and is not suited for dealing with small files because of its huge block size.

Apache Spark offers similar capabilities like MapReduce but shows lower latency and higher throughput. Spark usually runs on the Hadoop HDFS cluster platform, with YARN support like traditional Hadoop/YARN/MR solutions. It can certainly also run on single machines.

Apache Spark started as a research project in UC Berkeley in 2009. It finally was open sourced in 2010 and moved to the Apache Software Foundation in 2013 [71].

Spark is a unified analytics engine for large-scale data processing[40]. The ecosystem of Spark includes four additional packages:

- Spark **SQL**, a native SQL platform for direct or programmable data access and querying,
- Spark **Streams** for working with real time streaming data arriving from message queues, sensors, log files and other sources,
- **MLib** for running Machine Learning algorithm models and
- **GraphX** for graph computations and data analysis of properties attached to nodes and vertices.

These four frameworks together on top of Spark form an extremely versatile and very coherent environment with high abstraction. The code of the entire Spark ecosystem has been developed in Scala. Scala is a very expressive and flexible functional and object-oriented language, very well suited for heavy processing. All the framework packages extend the code base designed for Spark, share the same design philosophy and are based on the same ground and foundations like Spark.

One of the basic reasons Spark is much faster than MR, lies in the fact that Sparks' nodes use so called Resilient Distributed Datasets (RDD), which are immutable units of data residing on memory, rather than on persistent storage like the much slower hard disk drives. Spark is also easier to setup and operate and even less complicated to use.

It is possible to run Spark jobs on Hadoop or YARN clusters, on Apache Mesos or using a standalone scheduler for light applications and for learning purposes. Spark can be deployed when an already existing Hadoop/MR application needs to process streaming data, graph processing or simply SQL support.

The speed of Spark allows real-time processing of Big Data and brings Big Data one step closer to non-analytic applications that require low latency responses. Spark is not a substitute for Hadoop but should be viewed as a fast addon for performing searches, since it is memory bound. Hadoop FS is used for persisting the data.

[40] https://spark.apache.org/

Along with Volume, dealt with horizontal extension of the Hadoop file system, Variety solved with various NoSQL and NewSQL databases Spark hands a Velocity advantage to the Big Data programmer and technologist.

9.6. The Hadoop Ecosystem

The innovation of the Hadoop approach lies in the fact that the software is acting as an abstraction layer that simplifies most issues and problems of distributed processing and behaves like a huge single disk to the user. Hadoop offers fault tolerance and has scaling capabilities. It also supports redundancy by automated replication of files. The replication improves speed, because when multiple clients request the same resource, they can be served by multiple nodes. Apache Hadoop has an HDFS NFS Gateway, that supports NFSv3 and allows HDFS to be mounted as part of the client's local file system[41]. The logistics and complexities of storing and accessing large amounts of data on clusters of machines are hidden from the developer. Although the Hadoop/MR tool combination is a rigid approach, it can solve many types of problems involving querying huge data resources with reasonable response times. There are many applications build around Hadoop, used as addons. These applications form a suite of services, called the Hadoop Ecosystem. Some of the main projects of the Hadoop Ecosystem are:

- Monitoring tool: **Apache Zookeeper**
- NoSQL Databases: **Apache HBase**, **Cassandra, MongoDB**
- RDBMS import and export: **Sqoop**
- Data processing tools: **Apache Hive**, **Apache Pig, Impala**
- Data analysis tools: **Drill**, **Lucene**
- Data Serialization Components: **Avro** and **Thrift**
- Log file integration component: **Flume**
- Machine Learning and Data Mining Tool: **Mahout**

[41] https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsNfsGateway.html

Apache Zookeeper

Zookeeper is a distributed coordination service. It maintains configuration information and knows all names and nodes. It supports synchronization and acts as a name registry for all resources, guaranteeing consistency, reliability and timeliness of all updates.

Apache HBase

HBase is an open source, reliable and very fast real-time NoSQL database management system, that allows real-time reading and writing of Big Data. HBase supports processing of very large databases with millions of columns and billions of rows stored on top of Hadoop. HBase is coordinated by Zookeeper. HBase supports random reads and writes. Its integration with MapReduce allows applications to scale gracefully.

Cassandra

Apache Cassandra is a distributed NoSQL database system. Cassandra is a fast, distributed database that's highly fault tolerant as well as scalable. It provides high availability and linear scalability, twin goals that traditional relational databases cannot satisfy when handling very large data sets [72].

Sqoop

Apache Sqoop allows transferring structured data from relational databases to and from Hadoop HDFS [73].

Apache Hive

Hive is an SQL query engine, that converts SQL queries to MapReduce jobs. Thus, it simplifies the procedure of enabling Hadoop to be used as a data warehouse and supports queries. Tables correspond to HDFS directories. It is used for analytic jobs, unlike HBase, which is a true DBMS. Hive can easily integrate with traditional data center technologies with the use of the familiar JDBC interface. It is used as a bridge to integrate applications that use tabular data with Hadoop. The **Hive Query Language** (**HQL**) has similar semantics and functions as standard SQL in case of relational database, so that experienced database analysts can easily get their hands on it. Hive's query language can run on different computing engines, such as MapReduce, Tez, and Spark [74].

Pig – Pig Latin[42]

Apache Pig is a platform for analyzing large data sets and consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

At the present time, Pig's infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs, for which large-scale parallel implementations already exist. Pig's language layer currently consists of a textual language called Pig Latin.

Mahout

Mahout is a platform that supports executing machine learning algorithms. It allows finding meaningful patterns from data stored in HDFS, allows classification and clustering [75]. It supports Scala, and Apache Spark. It is a well-adjusted and complicated black box solution that allows further customization and building of new models, rules and features. [76].

Drill

Drill is a tool that supports direct ANSI SQL queries against any data stored in HDFS. Drill does not require schemas and allows processing complex records easily and fast.

Apache Flume

Apache Flume allows collecting, aggregating and moving large amounts of data from various sources, like log files, social media data, data from any web application to an HDFS.

This application and project-list is by no means exhaustive. The number of applications around Hadoop is already large and new applications are added constantly.

[42] https://pig.apache.org/

9.7. Big Data Infrastructure Vendors

The Apache Software Foundation (ASF) is a popular open source software vendor. It has been an open source technology incubator for 20 years, having a net worth of software products made available to the public at no cost estimated to be higher than 20 billion dollars. 300 different top-level projects, ranging from Java libraries and Application Programming Interfaces for various tasks, web servers and Database Management Systems to Big Data platforms and frameworks. Almost all the Big Data frameworks mentioned above are Apache projects. This means they all are open source and available for free, under Apache licenses. Apache licenses allow free personal or commercial use, modification, distribution and even selling. Both Hadoop and Spark are ASF projects.

Theoretically, a technically knowledgeable administration team has the potential to setup and run Hadoop and Spark with low installation costs. Still, the total expenses of operation and ownership include hardware and installation costs, eventually commercial software and maintenance costs. Spark clusters are usually comprised of commodity machines with a lot of RAM memory, which usually are more expensive than the ones used for Hadoop.

Although anybody can download a binary executable or even the full source code of any project and use it under the license conditions described here, there are commercial companies that sell enterprise-ready distributions, packaging Hadoop for the enterprises specifically. They also provide support and courses for training. The success of this market is due to the lack of expertise and skilled resources, along with the high complexity of deployment and the difficulty of choosing the right combination of tools for each application among the options offered in the ecosystems.

The main Big Data software and Cloud vendors today are:

- Cloudera - Hortonworks
- Amazon Web Services Elastic MapReduce Hadoop Distribution
- Microsoft
- MapR
- IBM InfoSphere Insights
- DataBricks

Theoretically a technically knowledgeable in cluster administration team has the potential to setup and run Hadoop and Spark with zero installation costs. Still, the total expenses of operation and ownership include hardware and installation costs, eventually commercial software and maintenance costs. Spark clusters are usually comprised of commodity machines with a lot of RAM memory, which usually are more expensive than the ones used for Hadoop.

As seen, there are many different options for the non-faint hearted to enter and join the world of Big Data. The Apache software packages can easily be installed to standalone Linux or Windows computers for educational purposes and a few simple commodity machines could be used as test beds for installing Hadoop FS on a cluster and learning how to work with YARN, Zookeeper and Spark and their entire ecosystems.

The latest technology shift are Cloud solutions, since the Cloud can always scale dynamically and provide computer resources fitted to the exact needs of the corporation, without having to maintain and support underutilized hardware or having bottlenecks at peak times.

The next chapter deals with extending the application of the analyzer by merging interaction data from customers of the shopping floor. The data collection devices and techniques are described, as well as the way data from the e-commerce site can be merged with data from the physical store customer interaction and provide better overview and support decision making for the entire corporation.

10. EXTENDING USE OF THE ANALYZER FOR PHYSICAL RETAIL STORES

"Extending Use of the Analyzer for Physical Retail Stores" deals with extending the application of the analyzer by polling interaction data of customers in the shopping floor. The information is collected automatically during the physical store customer interaction with the use of iBeacons and Near Field Communications for data acquisition in the retail shop floor. The generated retail data is pushed to the Analyzer in near real time. This data enhances the Analyzer input with physical store data and can be used in order to enrich customer clustering and behavioral analysis, by enhancing the available operational data sets, collected by a Web Analytics application, providing better overview and supporting decision making for the entire corporation.

Most retailers offer their products via e-Commerce applications in order to promote them better, globalize their clientèle, enhance and support sales. This paper explores and proposes techniques that will collect data from the physical shopping floor via low cost mobile technologies and promote them for processing to a customized sophisticated e-Commerce Analyzer. The Analyzer connects the worlds of physical and virtual shopping. The information is collected automatically during the physical store customer interaction with the use of iBeacons and Near Field Communications for data acquisition in the retail shop floor. The generated retail data is pushed to the Analyzer in near real time. This data enhances the Analyzer input with physical store data and can be used in order to enrich customer clustering and behavioral analysis, by enhancing the available operational data sets, collected by a Web Analytics application, providing better overview and supporting decision making for the entire corporation.

10.1. COMBINING E-COMMERCE SITES AND PHYSICAL STORES

Web analytics measurement and visualization techniques provide excellent tools and techniques in order to analyze, document, visualize and report almost every aspect of customer interaction and performance detail of any e-shop and any web application. Analytics applications are based upon measurements and logging of the customers actions. They are complicated software systems, with the ability to accurately record user actions, key-clicks and responses to databases and access log files. Many e-commerce sites have been implemented in order to enhance and complement physical retail shops. Their goal is to extend the operation capabilities and the customer base, providing services parallel to those of the physical shop. They provide low cost availability beyond working hours by offering 24/7 services and eliminate the time zone restrictions and customer location problems.

Physical stores use multiple techniques in order to analyze client behavior. Most of these techniques are based on capturing the data at the cash register, during the checkout process, when the customer is about to pay. If the customer uses a credit card the purchaser is uniquely identified. Even customers that pay with cash can be monitored to some extend precisely via discount coupons and the use of loyalty cards.

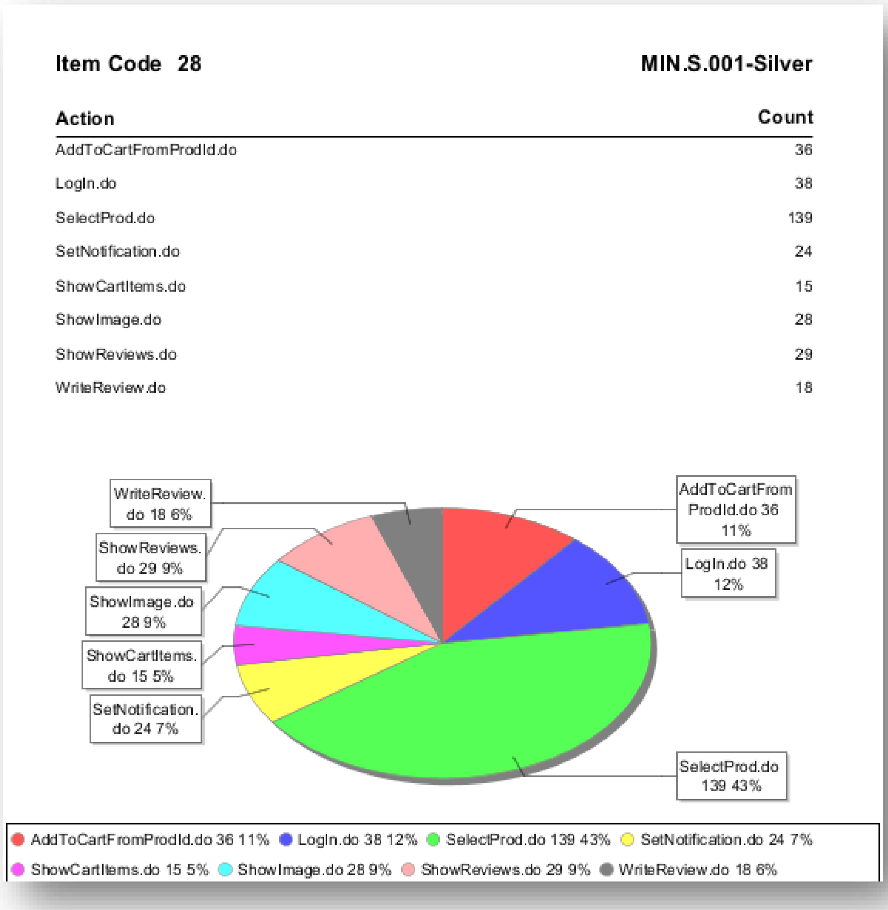
Every business transaction and step generate a large volume of data. As indicated by Provost and Foster [76], Data Mining is used for general customer relationship management to analyze customer behavior in order to manage attrition and maximize expected customer value. The finance industry uses data mining for credit scoring and trading, and in operations via fraud detection and workforce management. Major retailers from Walmart to Amazon apply data mining throughout their businesses, from marketing to supply-chain management. Many firms have differentiated themselves strategically with data science, sometimes to the point of evolving into data mining companies.

A versatile Log File Analyzer (LFA) is the tool that provides operational meta-data that allow the management to identify the way customers approach the offered products and measures their behavior and generates Customer Behavior Model Graphs [2]. The LFA used for this research was implemented with the ability to import and operate on both log file and e-shop data extracted either on demand or in near real time from the e-shop application.

The data visualization component of the LFA is customizable to a large extend, since requirements are usually volatile and new or modified output is often needed, and generates mainly:

- Graphical reports about products and actions,
- PDF output,
- Statistic Reports,
- Exception Reports and
- Traffic, Speed and Throughput Diagrams [63]

All reports are built around database queries and are fully customizable and extensible in order to provide visualization mechanisms that will allow the administrator or the management to extract knowledge about specific time spans and periods of operation. A typical example is visible in Figure 65 [63].


Item Code 28 **MIN.S.001-Silver**

Action	Count
AddToCartFromProdId.do	36
LogIn.do	38
SelectProd.do	139
SetNotification.do	24
ShowCartItems.do	15
ShowImage.do	28
ShowReviews.do	29
WriteReview.do	18

FIGURE 65: ACTIONS PER PRODUCT PIE CHART

10.2. MOBILE TECHNOLOGIES

Mobile technologies, like iBeacons, based on BLE (Bluetooth Low Energy) and NFC (Near Field Communications) tags, send data in order to activate mobile devices and support and provide the generation and capturing of streams of data that correlate a customer to a specific product. The store owner places them in appropriate locations, next to exposed products and through the uniqueness of their identification numbers, software applications can match the exhibited products and their geolocational information to the physical customers approaching them. Bluetooth Low Energy is a wireless personal area network technology used for transmitting data over short distances. As the name implies, it’s designed for low energy consumption and cost, while maintaining a communication range like that of its predecessor, Classic Bluetooth [77].

Streams of precise measurable data, automatically generated during shopping, can be used in order to describe the behavior of the customers and visitors of the physical store, in a similar fashion to e-commerce customer's clicks and selections in a website, are captured.

The main difference between the two technologies lays in the necessary proximity between the mobile device and the source. iBeacons are used in order to automatically activate apps on the smartphone of the shop, visitor while NFC tags require higher proximity. Both technologies leverage existing networks and enhance customer support and provide accurate product information.

10.3. The Physical Store

Back in the times where computers were introduced to the corporate world, among the first applications that became very quickly popular were Management Information Systems (MIS) for businesses. MIS's dealing with inventory, customer-maintenance, accounting, purchasing, invoicing etc., were introduced in the late fifties and have been widely used since the early sixties, running primarily on mainframe computers at that time. This long evolution, lead to today's contemporary highly integrated Enterprise Resource Planning (ERP) Systems, involving almost any hardware, software and networking technological innovation that has been achieved during the past 60 years. The centralized mainframe-based application model used initially has shifted to modern distributed systems, heavily based on the Internet, mobile communications and finally the Cloud and Internet of Things.

A general data flow diagram of the basic operations performed by a typical retail system supporting the operations at the shopping floor, mainly customer purchases, is shown in Figure 66. The very basic operations shown are: buying an item, payment processing and receiving an invoice. Customer transactions lead to inventory and customer data updates. In order to gain full auditing ability, for future data mining and analysis of transactions, the event information is captured and stored in a log file.

This research is focused on the type of e-commerce site operators, which operate both physical stores as well, and use their e-shop in order to enhance sales, expand clientele and improve customer support. Web sites usually are multilingual, deal with multiple currencies and operate 24/7 and are cheap.

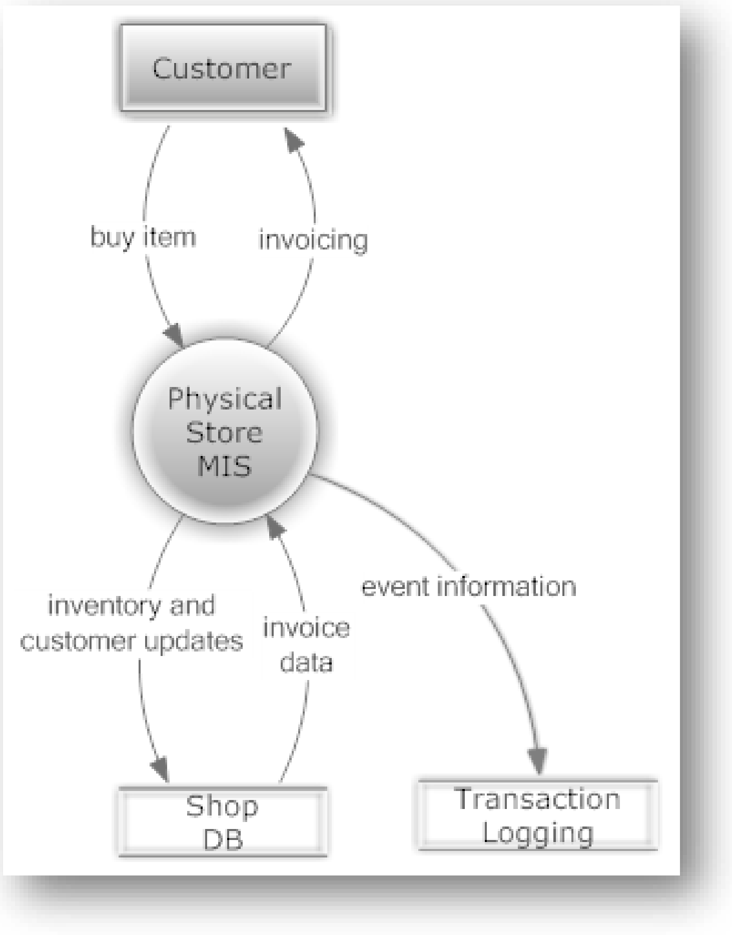


FIGURE 66: THE TYPICAL STORE MIS

Certainly, since the introduction of the Cloud as a new form of computing, there is a change in the way traditional MIS's are viewed conceptually. The desktop applications seem to evolve into web pages. The physical location of the server seems to become irrelevant.

10.4. THE ELECTRONIC COMMERCE SITE

A typical e-commerce site can easily be viewed as a metaphor of an MIS. The physical presence of the customer is not required and the Similarities between their DFDs shown in Figure 66 and Figure 67, are obvious, since the e-commerce site deals and supports product purchases like a typical MIS does. The e-commerce site operates on similar data repositories, including inventory and customer data as well. For any corporation that operates both a physical and a virtual shop, the virtual shop repository could be automatically fed by inventory data from the physical store database.

The goal of a business to customer (B2C) e-shop application is to promote retail sales and create profit. A virtual store allows buying products or services through a website, in analogy to a bricks-and-mortar retailer or a shopping mall [63]. E-commerce is big business and getting bigger every day. Growth estimates from eMarketer report that business-to-consumer (B2C) e-commerce sales worldwide will reach $1.5 trillion in 2014, increasing nearly 20% over 2013 [78].

The customer accesses the e-shop web page, searches the web site for a product and adds it to a shopping cart metaphor. This procedure is repeated and finally the customer checks out and the invoice is issued. The item search procedure is assisted by the e-shop categories, manufacturer selection, advanced search mechanisms and related product proposals. The customer of the physical store picks the items straight from the shelves.

Emphasis is given to the registration of events, which leads to full accountability of the operation and allows a toolbox of analytics to extract customer and system behavioral information.

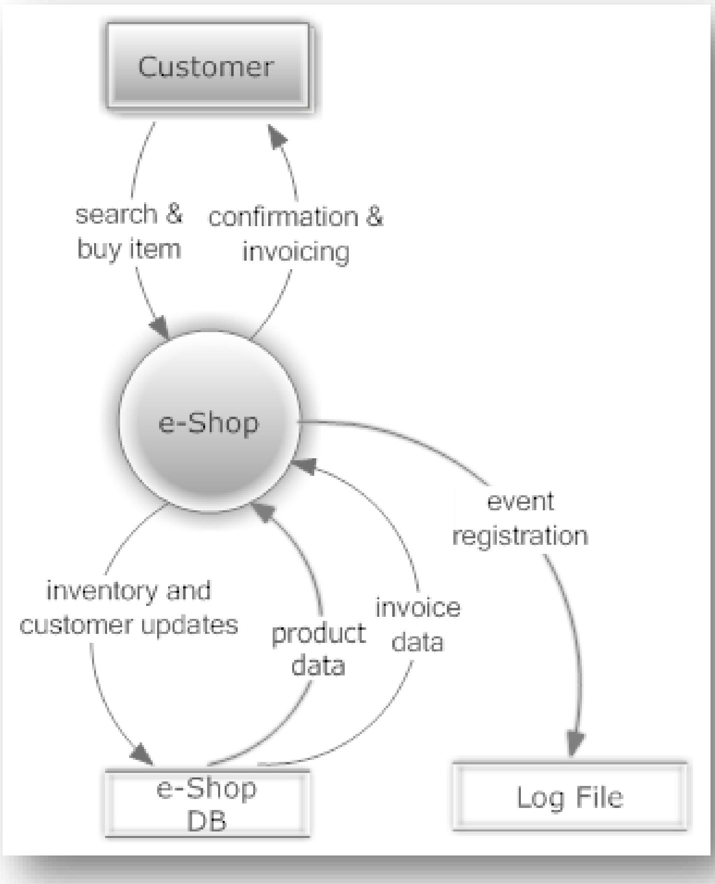


FIGURE 67: THE TYPICAL E-COMMERCE APPLICATION

10.5. THE LOG FILE ANALYZER

The log file analyzer (LFA), shown in Figure 68, is a toolbox, consisting of a menu driven application, with graphical user interface. The LFA is written in Java, in order to be portable across operating systems and architectures. It can reside on a separate server or on the server where the e-shop is running.

It is essential for the e-shop administrator to be given a user-friendly application to work with. It can quickly track the transaction histories of any e-shop, measure the required parameters, the performance of actions of the entire shop and reveal the results of measurements under various load conditions.

The architecture was chosen to be expandable in order to allow easy inheritance of the classes that do the data imports and the negotiations with the various web servers, so that whenever e-shops that run on new web server architectures are needed, their respective log files and any e-shop application databases can be adapted and evaluated relatively fast. In order to be usable, the toolbox contains a tool with the ability to load log file deltas as easy as possible and get the e-hops latest status for evaluation at any time.

The requirements for results, measurements, data mining and visualization of data etc. are under constant change, therefore the LFA is easily customizable and extensible, allowing standard interactive report generators to provide customized and semi-customized reports.

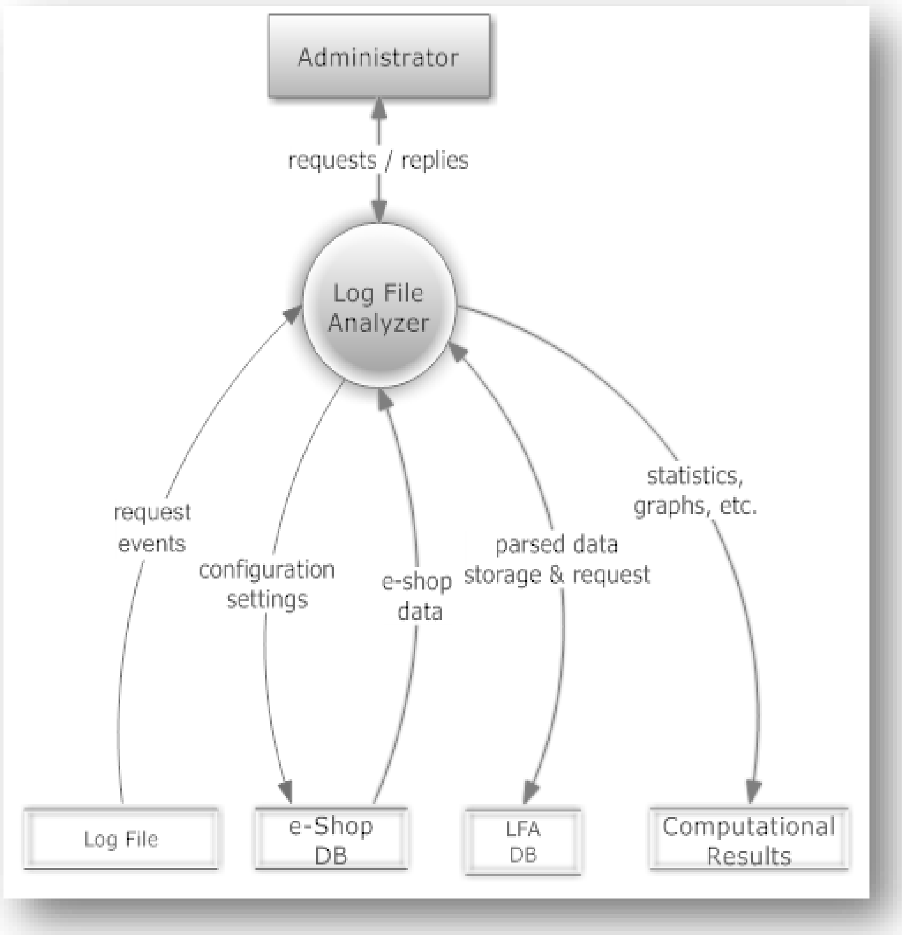


FIGURE 68: THE LOG FILE ANALYZER

The Log File Analyzer is a typical Extract Transform Load (ETL) application [79]. Its basic input is extracted straight from the log file of the e-shop and includes all requested events and transactions that took place during its operation. The LFA allows the administrator to configure the e-Shop in order to produce the appropriate log file details.

The analyzer application allows merging together information from the e-shop database with the log file entries. This feature allows the system to enrich the log file entries with familiar product codes, names and descriptions and produce easy comprehensible visualizations of the data.

The transformed information is stored to the LFA database transaction file, and is used in order to produce statistics, graphs and measurements.

The Analyzer is designed in the "Near Real Time ETL" fashion, described by Vassiliadis and Simitzis, [57] since it is compliant to the Quality-of-Service (QoS) characteristics:

1. Maximum freshness of data.
2. Minimal overhead of the source systems.
3. Guaranteed QoS for the warehouse operation.
4. Controlled environment.
5. Scalability in terms of sources involved, queries posed by end users and volumes of data to be processed.
6. Stable interface at the warehouse side.
7. Smooth upgrade of software at the sources.

10.6. The Combined Analyzer

Typical physical store and e-shop combo systems are comprised of two usually separate applications, which require some form of data communication bridges between them. Although the MIS and the e-shop are two distinct applications, viewed from the executive point of view, they are facets of the same information system. Daily sales figures, price and inventory updates, product reorder levels and stock from one system must update the other. In Figure 69 the interconnection between the log files can be seen. The analyzer reads data from the e-shop and the MIS databases, but the main data bulk is read from the two log files. The entries in the two log files are parsed in accordance to their layout format, which is customizable by the application and are merged into the LFA database. From then and on the data can be used for mining, cross-correlation and any analysis and visualization necessary.

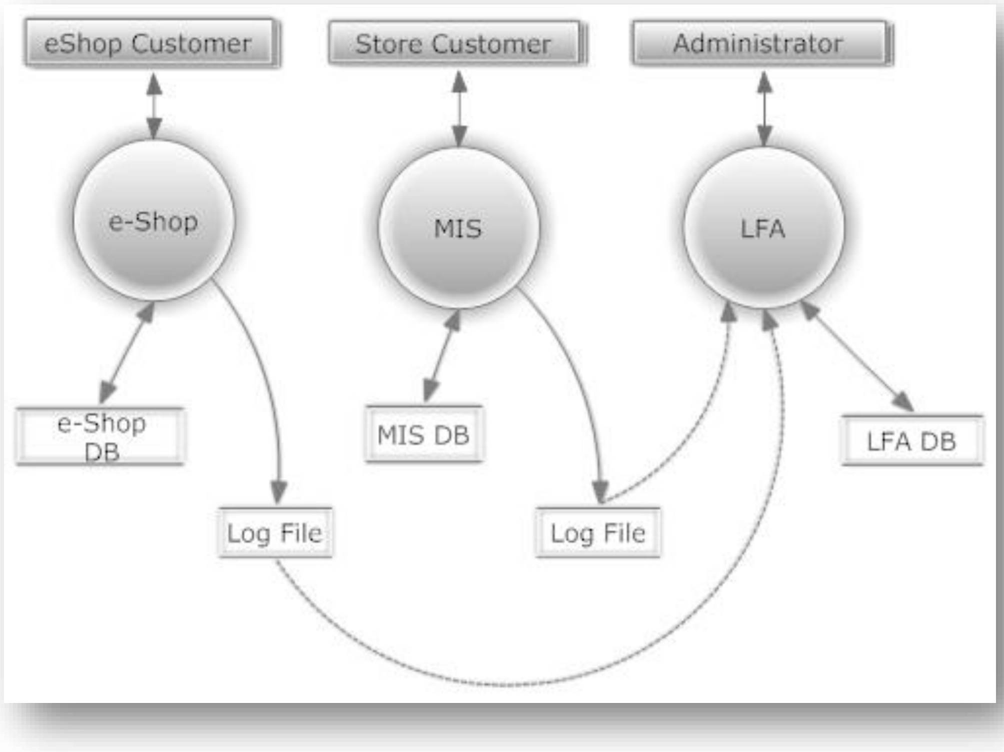


FIGURE 69: THE LOG FILE ANALYZER COMBINING E-SHOP AND MIS

The quality and precision and timeliness of the reported results by the LFA depend heavily on the quality and timeliness of its input.

Data collected by the e-shop, is pushed to the LFA DB in three distinct fashions [58]:

1. On demand from-to: The administrator requires from date and time and to date and time.
2. On demand, based on dates or sizes, where entire rotatable log files are loaded to the DB and
3. Automatically in near real time.

All three methods are available to the administrator, with the third fashion being the most convenient one, since it requires no administrator intervention. It parses and loads all required data into the LFA DB immediately after the data are collected as they are generated, allowing for real time measurements, warning and alarm generation in case of the occurrence of a problem or an exception event.

The topology of such a combined system is flexible. These three communicating applications could even reside on the same computer in small scale businesses, or even on three distinct servers with separate database servers and communication servers, all located in different locations. A large amount of companies operating e-shops prefer using virtual private networked (VPN) servers hosted by third party Internet hosting service providers, located geographically in different countries in order to lower operational costs.

The market for hosting companies is large, prices and services vary. Quality of service, reliability and the amount of freedom given by the provider to the customer to configure the rented systems are important selection criteria.

The latest tendency in computing is using Cloud services. The general idea behind moving to the Cloud is based on full elimination of the need to use customized applications running on self-administered servers locally, with pure web applications running on third party machines externally, with zero investment in hardware installations and administration, other than personal computers connected to the Internet equipped with just a browser and local printers. The Cloud also reduces costs, by allowing the e-shop operator to deal with peak loads without the need of huge servers.

10.7. Enhancing Input

There is a significant conceptual differentiation between the MIS and the e-commerce application, regarding the log file level of information detail granularity they can generate. While the MIS database and consequently the log file is updated whenever a transaction is committed to the database tables, that is whenever a retail customer moves to the cash register and every item of the cart is scanned by the clerk during checkout, the e-commerce system, as a pure web application has the capability of registering every single key click and every selection of any visitor, even if the visitor does never become a customer and leaves.

For every session a vast amount of data is registered, including time stamps and the entire history of requested items and categories. This information is pushed to the analyzer and allows an analyst administrator to evaluate the visitors' behavioral patterns and correlate information according to location, time, period of the year and gain insight into the complex selection mechanisms of the public that visits the e-shop. In order to gap this detail granularity differentiation between an MIS and an e-shop, receptors from the mobile telephony world can be used in the retail store floor, in order to capture and provide interaction information, like the detail information of the key clicks of the e-commerce application. This way the shopping cycle of any visitor of the store can be analyzed with a detail granularity equivalent to the one found in e-commerce applications.

The smart phone can either connect via the mobile network or any in house Wi-Fi network in order to obtain information about any product, discount coupons and price offers. The customer can also proceed to the e-shop link, complete the purchase online and pick up the products on the way out. In any case, the LFA is registering all transaction details and information and enhancing the patterns required for an accurate customer behavior analysis, in real time.

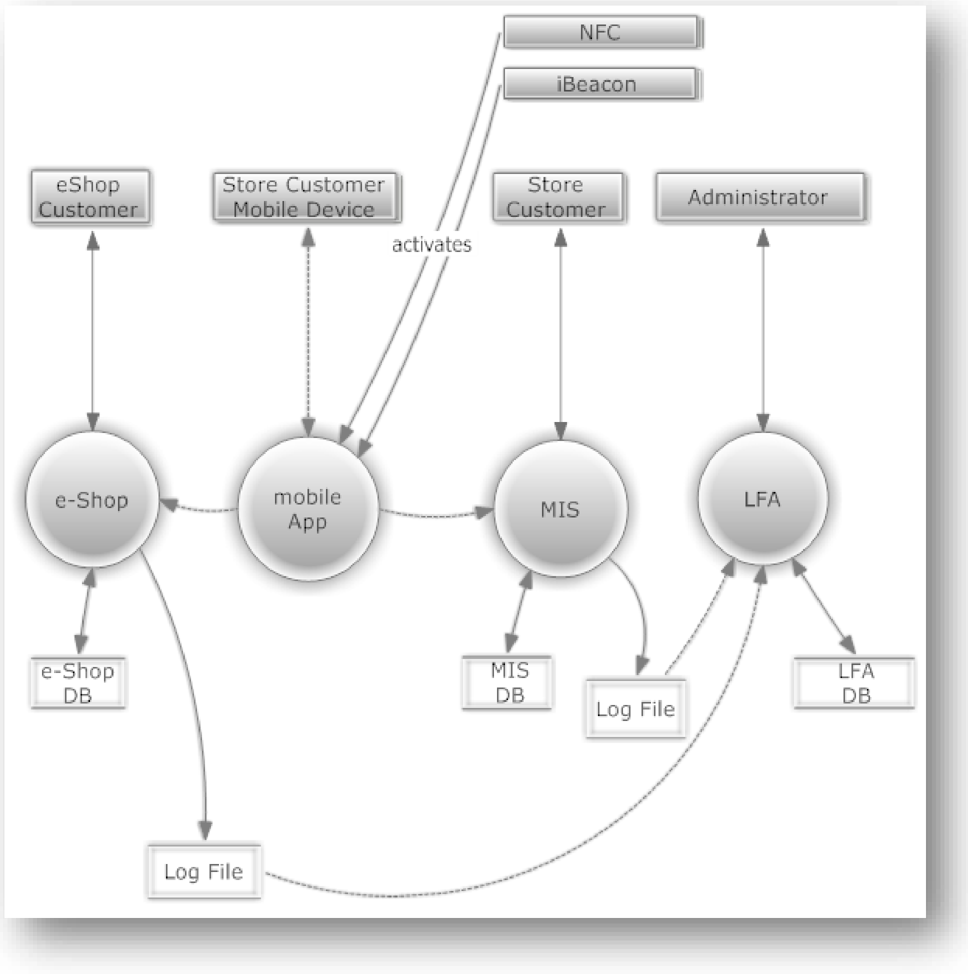


FIGURE 70: MOBILE TECHNOLOGIES AND LFA

Figure 70 shows the combined extended application architecture. A custom mobile application senses the iBeacon or the NFC tag and allows the customer to connect to either the MIS or the e-Shop multiple times during the shopping session as the customer moves in the shop. The iBeacons operate automatically via Bluetooth, while the NFC only when the user selects a specific product and wishes to obtain additional information about it or has the intention of purchasing it. This connection is either established via General Packet Radio Service (GPRS) or Wi-Fi if available.

Either way, they activate the mobile phone and allow its user to enhance the shopping experience by acquiring additional product information (Figure 71). The actions are recorded into the log file and are being transferred, in near real time, over to the LFA for further processing. This way the physical store obtains sensors that can trace the steps of a customer during in the same fashion as the e-shop can.

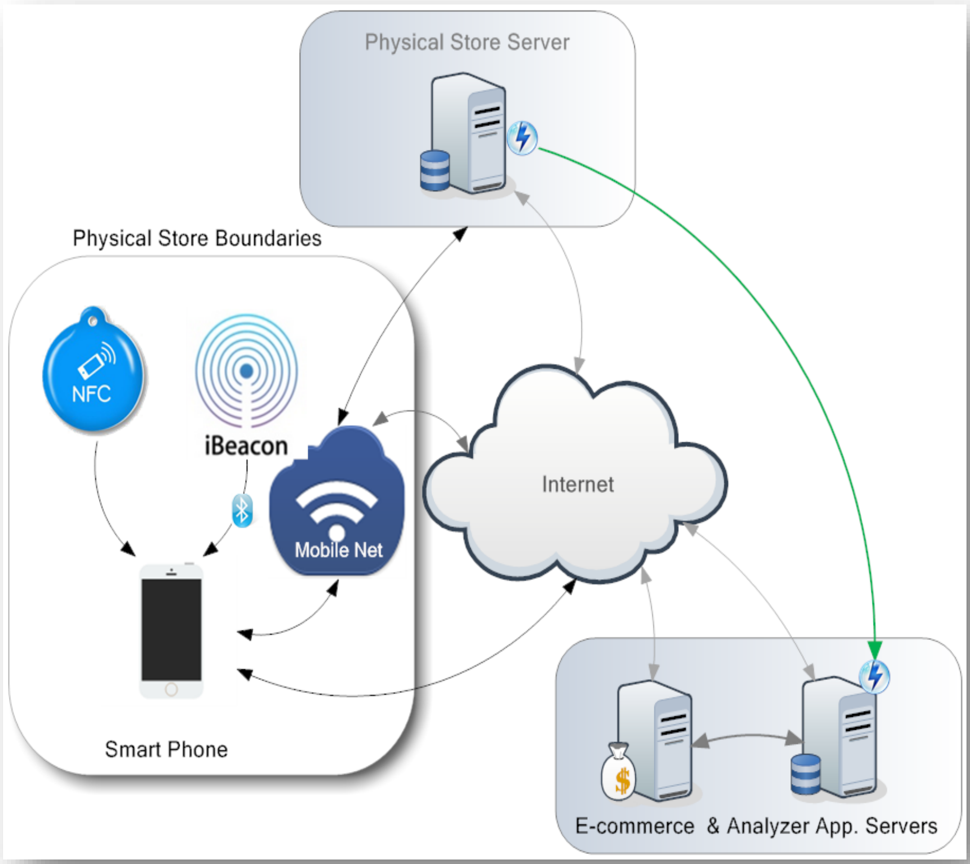


FIGURE 71: MOBILE DATA CAPTURING MECHANISM

An important issue that involves the evolution of web application frameworks has been resolved by the LFA we are developing. Web application systems, featuring the Rich Internet Application (RIA) environment model behave differently than traditional applications in the way they access the web server. The traffic to the web server is not as detailed as necessary, in order to capture user interaction to the same level as with non-RIA applications [58]. This new model of architecture shifts complicated graphical user interface (GUI) issues to the client. This way it allows using complicated widgets, like the ones used in desktop environments and allows the development of easier and more versatile user interfaces.

We use a specific queuing technique in order to implement the necessary near real time feed of data from the application to the Analyzer Application [58]. In addition to this data, entries obtained from the database log file are packaged and sent to the analyzed too [80]. The traditional application sends log file data via a named queue to the analytics application server. The RIA applications, on the other hand, have a less "interesting" log file to send. We try to avoid modifying the measured application and in order to generate the needed data, the applications database is being enhanced with triggers by the analyzer. An additional data table set is created. Once the triggers are activated, they write to the appropriate tables of this additional data table set the necessary information. This new set of data tables is constantly being polled by a data producer daemon and the information is fed to the analyzer in near real time.

A typical aspect of this implementation is presented in Figure 72 [58]. This example describes the steps that take place whenever the Customer places an order.

This following scenario takes place when the user places an order:

1. The placement of the order by the customer starts this cycle and is the event of interest for this metric.

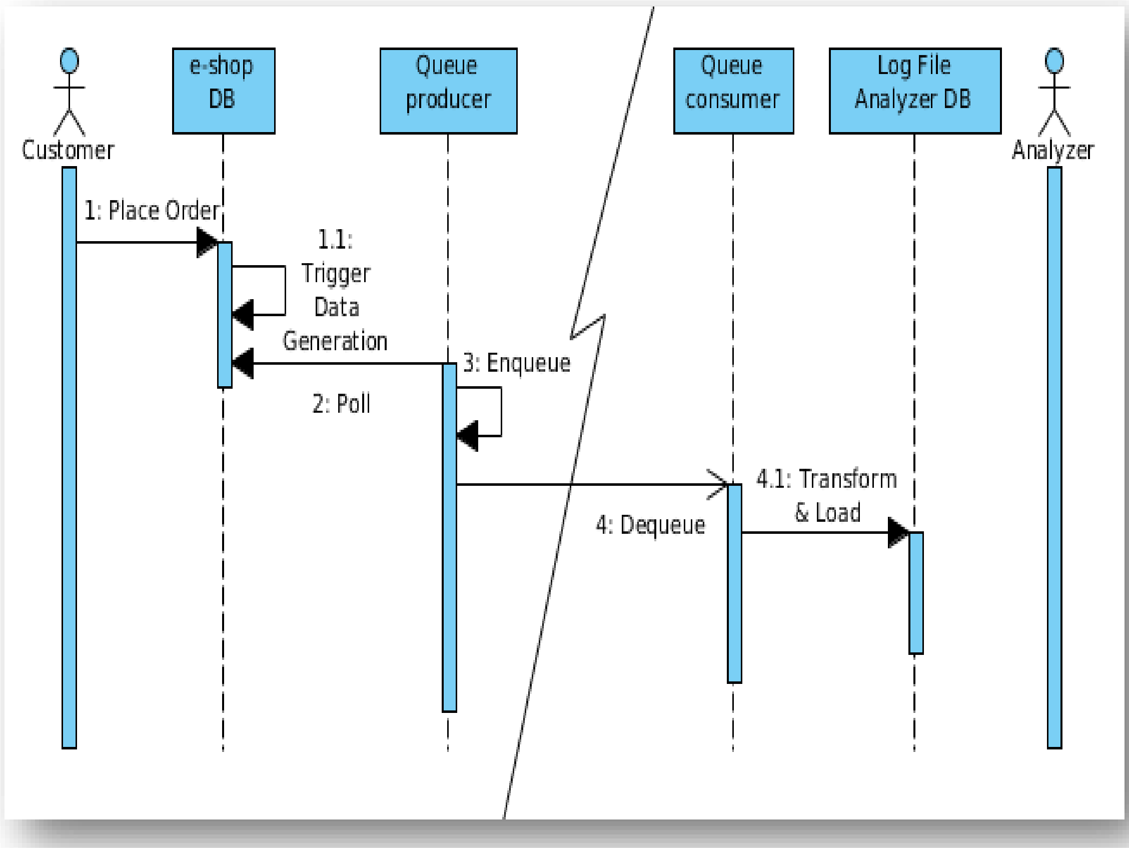


FIGURE 72: ORDER PROCESSING

1.1. In order to automatically intercept the order data, an after-insert trigger is placed in the orders table. This trigger copies the order-id auto number unique key to the new table "transmitted", which must be added to the e-shop database. This new table could also reside in a separate database, in case the database administrators want to keep the e-shop database changes and additions to a minimum. The contents of the transmitted table are kept simple. It includes just:

- An auto number as a primary key.
- The foreign key order number.
- One bit, indicating if the transmission to the LFA is pending or not (0 or 1).

2. This table is polled periodically for any pending records by the queue producer.

3. If pending entries are located, based on the order number, all required additional information is gathered with the appropriate left joins with other e-shop database tables and appropriate log file data and the resulting data structure is enqueued.

4. The queue consumer at the LFA server-side dequeues the information and

4.1. Transforms and loads the data to the LFA database. From there the order data is made available to the web application for presentation to the user.

10.8. Benefits

Mobile technologies and communication techniques commonly used in e-commerce applications can be extended in such a way that will significantly enhance the shopping experience of traditional commerce. They can be extended and can become hybrid systems, offering much better service to customers of traditional brick and mortar shops.

Typically, a consumer spends some time searching the Internet before making a purchase, comparing specifications and reading reviews. When visiting a traditional shop, the customer must rely on the words of the salesman. Three relatively low-cost techniques make product information available to customers that carry portable devices like smart-phones or tablets:

- Use of NFC tags for each product
- Use of iBeacons and
- Use of Quick Response codes (QR code) tags, attached to the exhibited products.

NFC tags provide the customer with a link which redirects the mobile web client to the product's web site entry, providing all necessary information and details about the specific item, also allowing the customer to complete the purchase electronically. The web server, hosting the e-shop, can only then have clues about the whereabouts of the customer, when special NFC-tags are being used, intended for specific locations of the product.

iBeacons on the other hand, are low cost indoor operating devices, acting as micro-location awareness and positioning systems, that transmit constantly in simplex fashion packages via Bluetooth low energy (BLE). Appropriate scanner-applications on the mobile device notify the customer automatically upon approaching the product or place. Multiple beacons can be processed simultaneously, and their unique codes can be used in order to identify the approximation of the customer to specific items. The constant package transmission enables the web server to easily measure the time a customer has spent studying any specific tagged item. The customer can automatically receive discount coupons or offers in order to make any seemingly product of interest more attractive. Accessing beacons provides an almost accurate geographical location of the customer inside the premises of the business and this geolocation is guaranteed. The mobile application can automatically offer technical specifications, details and information about the product as well as similar or cross selling products. Finally, the electronic payment can take place via the mobile phone and the personnel can arrange for the invoice, packaging and delivery of the product as soon as the customer reaches the exit gate. In other words, the customer can start a transaction online and conclude it off-line.

At the same time the route of each customer and the times she spends deciding what to purchase can be analyzed, similarly to the way it is analyzed by checking out the clicks on an e-shop browser. This can enhance the accuracy of the profile of the customer and add additional granularity if used together to other techniques, without requesting anything from the customer. If combined with data, extracted by the e-shop web analytics application will allow the company to provide better and more accurate services and make products proposals, which can lead to a more gratifying interaction and raise sales.

QR code tags can also allow the customer to receive information about products by redirecting straight to the e-shop page of the specific item. If approached this way, then the location of the customer is not guaranteed, since accessing through the special QR code could be requested by using a photograph of the code at any later time.

NFC and QR code reading are procedures that need to be demanded by the customer explicitly, while iBeacons will work automatically if the customer has the Bluetooth of her mobile device turned on.

The customer must receive adequate information about the advantages of using the system as well as the information that will be collected by the analytics application that will process the resulting interaction data.

Next section summarizes and redefines the problem and its dimensions. The input and output of the system is followed by given solutions given and the approaches. The new solution platform and its mutations are summarized and steps about developing and setting up the platform.

11. CONCLUSIONS

A summary and redefinition of the problem and its dimensions is attempted in this section. The input and output of the system is followed by given solutions given and the approaches. The new solution platform and its mutations are summarized and steps about developing and setting up the platform.

In the near past logfiles were often faced as unnecessary overhead that had to be removed every week in order to gain disk space. Their value was appreciated only in exception situations because they offered the only possibility to reveal why the exception occurred. For web applications the value of the logfiles lies in the fact that they hold time stamped details of the exact services provided and can be used as a solid base for analysis, in order to gain insight into the requests, responds and the sequence of all occurrences. Primary goal of Analytics applications is to collect metadata from every possible facet of the operational status and environment of an application system, process it and provide useful insight to the operators and management. Since this research is focused on web applications, the nature of the problem depends on technical details since it involves many layers of software: operating systems, web servers, database management systems, middleware and applications. Although log files are often viewed as byproducts of the operation of an application, the fact that they carry hard evidence and facts about who, when and how has visited a site, but also about how the web site has responded, render the as a very useful input for further analysis. Since access log files are very detailed semi-structured text files of temporal nature since they contain a progressing time stamp in every line they can be loaded to databases and running SQL queries can reveal metrics. (5. Web Analytics Systems / 5.1. Analytics Goals)

The Analytics application provides answers to questions like: Path of the visitors' session, total session time, number of sessions over time, average speed of service, existence of wrong links, known bot recognition etc.

Many commercial analytics applications operate only with input stemming just from log files. The access log file alone is enough as source of input for generating many metrics and visualizations, but the fact that vital information it carries about each web site is contained within the HTTP GET parameters in an encoded form poses a difficulty in comprehending the results. Nobody should be expected to know the product codes of the requested pages. Strings like “?prodId=8” are not specific enough for human understanding or for use as labels in visualizations.

Specifications and requirements were defined (7. Specifications and Requirements for the Analyzer Application / 7.2. Requirements). These specifications led to defining two classifications, (1) based on the way the analyzer imports data from the log file and (2) based on the scope of input data.

The first classification includes: on-demand, near real time and hybrid analyzers.

The second classification of analytics applications based on the scope of their input includes: type1 through 4. Type 1 operates solely on the log file, Type 2 imports product category and item records from the operational database and makes use of the description enhancing the visualizations and reports, Type 3 has a hybrid nature since it makes use of the web server log file but uses a tagging system API in parallel and Type 4 is social media aware.

A third special classification is discussed (10. Extending Use of the Analyzer for Physical Retail Stores), that merges clickstream input from a web site with input generated by portable devices interacting with iBeacons and NFC tags in the shopping floor of a businesses that operates e-commerce site in parallel with physical stores.

The input sources are the log file, database tables of the e-commerce site, web server configuration XML file and serialized streaming of the log file if installed. As noted, web applications that make heavy use of JavaScript and AJAX, usually rich Internet applications based on GWT for example generate reduced log files since the communication with the web server is sparse because their paradigm shift gives more responsibilities to the visitor's client. The proposed solution is to enhance the missing log file details is described in "7. Specifications and Requirements for the Analyzer Application / 7.4. Real-Time Analyzer" and is based on database triggers and use a message broker to send the appropriate data to the analyzer.

The approach of this research has dealt to a large extent with the steps of the Quantitative Analysis Cycle of an E-business Site [1] and provides insight into the operation of the web site.

The Analyzer provides the metrics and the insight necessary to get a performance overview of the status of the web site. It is easy to extend and add features, reports, visualizations and capabilities. The modular architecture based on free software tools and libraries allows easy restructuring if needed. The system is hybrid according to the first classification and can be converted to any type of the second classification by adding the appropriate components. Currently the system runs on a separate machine than the e-commerce sites and puts very little load on the web server where the producer daemons run.

Many metrics are built-in, and others are generated during the report generation phase, with SQL queries mostly embedded in the XML file of each report. The report generation process receives a database connection object from the connection pool and the appropriate parameters and is responsible for the query execution.

The output of the LFA includes reports, metrics, graphs, CBMG and SQL query results. Near real time statistics and graphs can be viewed by a browser on the web.

The proposed solution is a hybrid system, which according to Clifton is safer to operate, supports more than one e-commerce site, enhances the log file data with category and product descriptions from the e-commerce site. Has influence the pattern of the log file format, supports rich applications, offers near real time capabilities and is expandable and big data ready by moving the log file contents to a Hadoop HDFS and HBase.

12. FUTURE WORK

The next version of the Analytics application should make heavier use of state-of-the-art Big Data substructure available, should include machine learning mechanisms and neural networks in order to allow predictive Analytics algorithms and offer more additional data visualization options.

Software systems either evolve or die. The reason why all applications need permanent attention, refinement and enhancements is because the entire environment they operate in evolves with time and raises the need for them to adapt to these changes. These changes involve every level and aspect of the environment. Starting from the system software, they run on, the programming languages used to implement them, data sources, data repositories, input and output format, ending to computational requirements and specifications and needs of the users and the business. All these factors are subject to a constant evolution of their environments.

Analytics Application systems should also evolve in order to adapt to current technology trends and advances and must comply to the growing needs and capabilities of the computational environment and the progressing platforms. In order to make an application immune to the data volume issues, higher involvement of Big Data technologies is a necessary option. The Hadoop file system would allow the system to accommodate virtually unlimited sizes of log files and detail data over years of operation. Apache Spark ecosystem can substitute the streaming mechanisms, enhance real time processing speed and support machine learning algorithms.

A rewrite of the existing Java application in a functional programming language, preferably Scala is a future work that will ease the transition to Resilient Distributed Datasets of Spark, that can provide quicker responses and improved near real time speed for all requests even on extremely large data sets.

The current system, as has been developed, collects and prepares various data types that are used for providing Analytics measurements. This data can be used as a base for predictive Analytics. The Customer Behavioral Model Graph generates matrices that register the steps of the visitor during every session. Algorithms that study and classify users according to their behavior can be based on these matrices. Each is stamped with a unique session-ID. One when users log into the e-commerce site with their credentials can they be identified uniquely. Grouping all previous sessions collected over time for each user forms a base for predicting the future steps by applying models based on Markov chain analysis. The next version of the system will be able to do exactly that. Moving the data to the Hadoop File System, the Analytics application will become immune to data volume issues.

Data collected from various sources and streaming applications constantly extend the input and storage needs, so a total shift to Big Data instead of using single machine file systems seems an inevitable goal for every application with perpetually growing data on the long run. Log files and historical data grow in volume very fast and often need to be available on line as the years go by. The move to a practically huge distributed file system like Hadoop, can prevent historical data from the need to be stored in offline storage and will allow any prediction systems to give more accurate results, measure trends and tendencies, because they can be based on larger datasets.

Machine learning techniques allow the computer to learn from data. Different techniques and algorithms support this activity. Three common existing types of machine algorithms are: supervised, unsupervised and reinforcement algorithms.

Supervised learning algorithms process require input and output pairs for their training phase, while unsupervised systems require only input data and not outputs. In general, supervised algorithms are given example input and the associated output. This training step is repeated many many times and eventually, the algorithm becomes able to pick up a pattern between the inputs and outputs. At this point when feeding it with new input, it will eventually predict the output. The general way unsupervised algorithms work is by inputting only example input, without the associated output. By repetition of this step, eventually the algorithm receives the necessary training that allows it to cluster the input into groups and becomes capable after new input is inserted, to predict the cluster it belongs to. Finally, reinforcement algorithms perform similar to supervised, but are more sophisticated since they include agents that promote autonomous learning by trying to receive more positive than negative rewards.

Artificial neural networks are fundamental for deep learning. Their structure resembles that of the human brain and the interconnected biological neurons. The advances in computing speed and capacity and frameworks that deal with performing linear algebraic computations with huge matrices and large vectors of nodes allow applying different weights to large numbers of node neurons states.

Predictive Analytics would also enhance the value of the analyzer. Predictions of estimated expected load over periods of time may generate precaution reports for the e-commerce site and allow it to suggest ordering and reserving merchandise and items that are predicted to be needed before any reorder levels have been reached and before.

Since the analyzer application collects full sets of interaction data from multiple facets concerning the operation of e-commerce sites and all customer interactions over time this data can be easily used as the basis for training a neural network. The model could analyze customer reviews and classify them as positive, negative or neutral, find if customers have read them before purchasing items, cluster products according to popularity and provide recommendations based on previous purchases.

Predictions can be applied in order to reveal general tendencies, based upon past experience and trends and will offer advice, concerning product mix and stock, as well as point out slow moving items, that should be discontinued and have to be offered with discounted pricing, in order to reduce the existing product stock. A neural network can be used for the evaluation of customers loyalty and customize discounts for them as soon as they are recognized. At this point having

Visualizations are also becoming smarter and their interactivity can provide intuitive and useful interfaces for manipulating, selecting and summarizing large data quantities.

The data collection can be used as a base for setting up neural networks for recognizing patterns. The neural network can be trained on samples of the data and make predictions by detecting similar patterns in customer visits. The product reviews can be analyzed textually and

Another future goal is to apply machine learning algorithms on Analytics data. As the database grows, the which will provide support for business decisions and allow users to get the most out their big data.

Finally, the merger of the application with physical stores opens a very interesting range of connections with devices that is can be viewed as application extension in the area of Internet of Things.

In general, any modular Analytics application can be viewed as a large data and meta data source, that are prepossessed and easily capable of integration into any extension and any external application for further processing.

13. LITERATURE

[1] D. A. Menascé and V. A. F. Almeida, Scaling for E-Business, Upper Saddle River, New Jersey 07458: Prentice Hall PTR, 2000.

[2] D. Menascé and V. Almeida, "Challenges In Scaling EBusiness Sites," in *Proc. 2000 Computer Measurement Group*, Orlando, FL, 2000.

[3] "Global Trends in Online Shopping 2010," 31 Oct 2019. [Online]. Available: https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/Q1-2010-GOS-Online-Shopping-Trends-June-2010.pdf.

[4] "The Need For Speed II," *Zona Market Bulletin, ZONA Research, Issue 5,* p. 1, 2001.

[5] "The Web Robots Pages," 31 Oct 2019. [Online]. Available: https://www.robotstxt.org/.

[6] D. Menascé, R. Riedi and P. F., "Analyzing Web robots and their impact on caching," in *Proc. Sixth Workshop Citeseer*, 2001.

[7] L. Destailleur, "AWStats official web site," 2019. [Online]. Available: https://awstats.sourceforge.io/.

[8] S. Turner, "Analog 6.0," February 2010. [Online]. Available: http://www.analog.cx.

[9] J. Burby, S. Atchison and J. Sterne, Actionable web Analytics: using data to make smart business decisions, Willey Publishing, 2007.

[10] B. Erinle, Performance Testing with JMeter 2.9, Birmingham: Packt Publishing Ltd., 2013.

[11] S. Saxena and S. Gupta, Practical Real-time Data Processing and Analytics: Distributed Computing and Event Processing using Apache Spark, Flink, Storm, and Kafka, Birmingham: Packt Publishing Ltd., 2017.

[12] R. Kimball and R. Merz, The Data Webhouse Toolkit: Building the Web-Enabled Data Warehouse, New York: John Wiley & Sons, Inc. , 2000.

[13] R. Kimball and J. Caserta, The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data, Indianapolis: Wiley Publishing, Inc., 2004.

[14] R. Kimball and M. Ross, The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, Indianapolis: Wiley, 2013.

[15] R. Saxena and A. Srinivasan, Business Analytics: A Practitioner's Guide, New York: Springer-Verlag, 2013.

[16] M. E. Tyler and J. Ledford, Google Analytics, Indianapolis: Wiley Publications, 2006.

[17] J. L. Ledford and M. E. Tyler, Google Analytics 2.0, Indianapolis: Wiley Publishing, Inc., 2007.

[18] J. L. Ledford, J. Teixeira and M. E. Tyle, Google Analytics, 3rd Edition, Indianapolis: Wiley Publishing, Inc., 2010.

[19] B. Clifton, Advanced Web Metrics with Google Analytics, Indianapolis, Indiana: Wiley Publishing, Inc., 2008.

[20] B. Clifton, Advanced Web Metrics with Google Analytics, Second Edition, Indianapolis, Indiana: Wiley Publishing, Inc., 2010.

[21] B. Clifton, Advanced Web Metrics with Google Analytics, Indianapolis: John Wiley & Sons, Inc., 2012.

[22] B. Clifton, Successful Analytics: Gain Business Insights by Managing Google Analytics, Advanced Web Metrics Ltd, 2015.

[23] L. W. D. V. A. A. Daniel A. Menasce, Performance by Design: Computer Capacity Planning by Example, Prentice Hall, 2004.

[24] R. Sanders, 42 Rules for Applying Google Analytics. A practical guide for understanding web traffic, visitors and analytics, Silicon Valley, California: Super Star Press™, a Happy About® imprint, 2012.

[25] P. Baldi, P. Frasconi and P. Smyth, Modeling the Internet and the Web: Probabilistic Methods and Algorithms, Chichester, West Sussex: John Wiley & Sons, 2003.

[26] E. Ayanoglu, Y. Aytas and D. Nahum, Mastering RabbitMQ, Birmingham: Packt Publishing, 2015.

[27] S. Haunts, Message Queuing with RabbitMQ Succinctly, Morrisville: Syncfusion, Inc., 2015.

[28] G. M. Roy, RabbitMQ in Depth, Shelter Island, NY: Manning Publications Co., 2017.

[29] L. Grinshpan, Solving enterprise applications performance puzzles: queuing models to the rescue, Hoboken, New Jersey: IEEE Press, John Wiley & Sons, Inc., Publication, 2012.

[30] S. St. Laurent, Introducing Erlang: Getting Started in Functional Programming, O'Reilly, 2017.

[31] B. Ellis, Real-Time Analytics: Techniques to Analyze and Visualize Streaming Data, Indianapolis: John Wiley & Sons, Inc., 2014.

[32] G. Szabo, G. Polatkan, O. Boykin and A. Chalkiopoulos, Social Media Data Mining and Analytics, Indianapolis: John Wiley & Sons, Inc., 2019.

[33] I. Horton, Ivor Horton's Beginning Java, Indianapolis, Indiana: John Wiley & Sons, Inc., 2011.

[34] T. Khare, Apache Tomcat 7 Essentials, Birmingham: Packt Publishing Inc., 2012.

[35] A. Vukotic and J. Goodwill, Apache Tomcat 7, Apress, 2011.

[36] C. Mehta and others, MySQL 8 Administrator's Guide, Packt Publishing, 2018.

[37] J. Murach, Murach's MySQL 2nd Edition, Mike Murach & Associates, 2015.

[38] R. Bharathan, Apache Maven Cookbook, Packt Publishing, 2015.

[39] M. Helmke, A. Hudson and P. Hudson, Ubuntu Unleashed 2019 Edition: Covering 18.04, 18.10, 19.04, 13th Ed, Indianapolis, Indiana: SAMS, Pearson Education, 2019.

[40] J. LaCroix, Mastering Ubuntu Server Second Edition, Birmingham: Packt Publishing Ltd., 2018.

[41] T. Igoe, D. Coleman and J. Brian, Beginning NFC Near Field Communication with Arduino, Android, and PhoneGap, Sebastopol, CA: O'Reilly Media, 2014.

[42] C. Gilchrist, Learning iBeacon, Birmingham: Packt Publishing Ltd., 2014.

[43] M. S. Gast, Building Applications with iBeacon, Sebastopol, CA: O'Reilly Media, Inc., 2014.

[44] O. Campesato, Regular Expressions Pocket Primer, Dulles: Mercury Learning and inforMation LLC., 2019.

[45] Z. Nagy, Regex Quick Syntax Reference: Understanding and Using Regular Expressions, Berlin: Apress, 2018.

[46] B. Forta, Learning Regular Expressions, Addison-Wesley, 2018.

[47] K. Sharan, Beginning Java 8 Fundamentals: Language Syntax, Arrays, Data Types, Objects, and Regular Expressions, Apress, 2014.

[48] M. Fitzgerald, Introducing Regular Expressions, Sebastopol, CA: O'Reilly Media, Inc., 2012.

[49] B. Erinle, JMeter Cookbook, Birmingham: Packt Publishing Ltd., 2014.

[50] E. Halili, Apache JMeter, Birmingham: Packt Publishing Ltd., 2008.

[51] A. Tacy, R. Hanson, J. Essington and T. Anna, GWT in Action, Shelter Island, NY: Manning Publications Co., 2013.

[52] D. Guermeur and A. Unruh, Google App Engine Java and GWT Application Development, Birmingham, UK: Packt Publishing Ltd., 2010.

[53] "GWT Tutorial," Tutorials Point (I) Pvt. Ltd., 2015. [Online]. Available: https://www.tutorialspoint.com/gwt/index.htm. [Accessed 02 02 2018].

[54] B. Sosinsky, Cloud Computing Bible, Indianapolis, Indiana: Wiley Publishing, Inc., 2011.

[55] M. Schrenk, Webbots, Spiders, and Screen Scrapers: A Guide to Developing Internet Agents with PHP/CURL, San Francisco, CA: No Starch Press, Inc., 2007.

[56] B. Clifton, "Web Traffic Data Sources & Vendor Comparison," May 2008. [Online]. Available: http://webanalytic.yolasite.com/resources/data/web-data-sources.pdf.

[57] P. Vassiliadis and A. Simitzis, Near Real Time ETL, S.Kozielski, Robert Wrembel, Springer, 2009.

[58] C. J. Aivalis and A. C. Boucouvalas, "Future Proof Analyrics Techniques for Web 2.0 Applications," in *TEMU 2014*, Heraklion, 2014.

[59] G. Hohpe and B. Woolf, Enterprise Integration Patterns: Designing, Building and Deploying Messaging Solutions, Melrose: Addison-Wesley, 2003.

[60] B. Unhelkar, Software Engineering with UML, Boca Raton, FL: Auerbach Publications, CRC PRESS, Taylor & Francis Group, 2018.

[61] G. Toffoli, The Definitive Guide to iReport, New York: Apress, Springer-Verlag , 2007.

[62] B. Siddiqui, JasperReports 3.6 Development Cookbook, Birmingham: Packt Publishing Ltd., 2010.

[63] A. C. Boucouvalas and C. J. Aivalis, "An Eshop log file analysis toolbox," in *CSNDP*, Newcastle, 2010.

[64] P. Schieber, "The Wit and Wisdom of Grace Hopper," Yale University, April 1987. [Online]. Available: http://www.cs.yale.edu/homes/tap/Files/hopper-wit.html. [Accessed January 2020].

[65] D. Laney, "3D Data Management: Controlling Data Volume, Velocity, and Variety," Gartner Inc. META Group Inc, 6 February 2001. [Online]. Available: https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf. [Accessed 1 May 2020].

[66] S. Grimes, "Big Data: Avoid 'Wanna V' Confusion," 7 8 2013. [Online]. Available: https://www.informationweek.com/big-data/big-data-analytics/big-data-avoid-wanna-v-confusion/d/d-id/1111077?. [Accessed 01 05 2020].

[67] P. Warden, Big Data Glossary, Sebastopol, CA: O'Reilly Media Inc., 2011.

[68] S. Ghemawat and J. Dean, 2020. [Online]. Available: https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf. [Accessed 10 February 2020].

[69] A. Holmes, Hadoop in Practice, 2nd Edition, Shelter Island, NY: Manning, 2014.

[70] A. C. Murthy and others, Apache Hadoop YARN: moving beyond MapReduce and batch processing with Apache Hadoop 2, Addison-Wesley, 2014.

[71] "Spark Apache History," [Online]. Available: https://spark.apache.org/history.html . [Accessed 15 June 2019].

[72] S. R. Alapati, Expert Apache Cassandra Administration, Flower Mound: Apress, 2018.

[73] K. Ting and J. J. Cecho, Apache Sqoop Cookbook, Sebastopol, CA: O'Reilly Media, Inc., 2013.

[74] D. Du, Apache Hive Essentials Second Edition, Birmingham: Packt Publishing Ltd., 2018.

[75] P. Giacomelli, Apache Mahout Cookbook, Birmingham: Packt Publishing, 2013.

[76] D. Lyubimov and A. Palumbo, Apache Mahout: Beyond MapReduce, CreateSpace Independent Publishing Platform, 2016.

[77] F. Provost and T. Fawcett, Data Science for Business, O'Reilly, 2013.

[78] "A Guide to Beacons," 2015. [Online]. Available: http://www.ibeacon.com/what-is-ibeacon-aguide-to-beacons/.

[79] "“E-Commerce Shifts into Higher Gear", a global Nielsen Consumer," February 2015. [Online]. Available: http://ir.nielsen.com/files/doc_financials/Nielsen-Global-Ecommerce-Report-August-2014.pdf.

[80] R. Kimball and M. Ross, The Data Warehouse Toolkit 2nd Edition, Willey Computer Publishing, 2002.

[81] C. J. Aivalis and A. C. Boucouvalas, "Log File Analysis of E-commerce Systems in Rich Internet Web 2.0 Applications," in *PCI*, Kastoria, 2011.

[82] A. Holmes, Hadoop in Practice ISBN 9781617292224, Manning, 2014.

[83] "Apache Spark History," 15 June 2019. [Online]. Available: https://spark.apache.org/history.html.

[84] "Hadoop YARN," 14 June 2019. [Online]. Available: https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html.

[85] R. Kimball and M. Ross, Relentlessly Practical Tools for Data Warehousing and Business Intelligence, Second Edition, Indianapolis: John Wiley & Sons, Inc., 2016.

[86] M. Watson, "8 Key Application Performance Metrics & How to Measure Them," 12 June 2019. [Online]. Available: https://stackify.com/application-performance-metrics/.

[87] J. Li, L. Xu, L. Tang, S. Wang and L. L. a, "Big data in tourism research: A literature review," *Tourism Management,* pp. 301-323, 2018.

[88] S. Ghemawat and J. Dean. [Online]. Available: https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf. [Accessed 10 02 2020].

[89] A. C. Murthy and others, Apache Hadoop YARN: moving beyond MapReduce and batch processing with Apache Hadoop 2 ISBN-13: 978-0-321-93450-5, Addison-Wesley, 2014.

THE END