Project Number 318736

D4.1 – Data Collected for Thread Analysis

Version 1.2
13 February 2013
Final

Public Distribution

University of Manchester

Project Partners: **Centrum Wiskunde & Informatica, SOFTEAM, Tecnalia Research and Innovation, The Open Group, University of L'Aquila, UNINOVA, University of Manchester, University of York, Unparallel Innovation**

PROJECT PARTNER CONTACT INFORMATION

Centrum Wiskunde & Informatica Paul Klint Science Park 123 1098 XG Amsterdam, Netherlands Tel: +31 20 592 4126 E-mail: paul.klint@cwi.nl	**Softeam** Alessandra Bagnato Avenue Victor Hugo 21 75016 Paris, France Tel: +33 1 30 12 16 60 E-mail: alessandra.bagnato@softeam.fr
Tecnalia Research and Innovation Jason Mansell Parque Tecnologico de Bizkaia 202 48170 Zamudio, Spain Tel: +34 946 440 400 E-mail: jason.mansell@tecnalia.com	**The Open Group** Scott Hansen Avenue du Parc de Woluwe 56 1160 Brussels, Belgium Tel: +32 2 675 1136 E-mail: s.hansen@opengroup.org
University of L'Aquila Davide Di Ruscio Piazza Vincenzo Rivera 1 67100 L'Aquila, Italy Tel: +39 0862 433735 E-mail: davide.diruscio@univaq.it	**UNINOVA** Pedro Maló Campus da FCT/UNL, Monte de Caparica 2829-516 Caparica, Portugal Tel: +351 212 947883 E-mail: pmm@uninova.pt
University of Manchester Sophia Ananiadou Oxford Road Manchester M13 9PL, United Kingdom Tel: +44 161 3063098 E-mail: sophia.ananiadou@manchester.ac.uk	**University of York** Dimitris Kolovos Deramore Lane York YO10 5GH, United Kingdom Tel: +44 1904 325167 E-mail: dimitris.kolovos@york.ac.uk
Unparallel Innovation Nuno Santana Rua das Lendas Algarvias, Lote 123 8500-794 Portimão, Portugal Tel: +351 282 485052 E-mail: nuno.santana@unparallel.pt	

DOCUMENT CONTROL

Version	Status	Date
0.9	First version with initial access info	10 January 2013
1.0	Complete version for review	28 January 2013
1.1	Revisions addressing review comments	11 February 2013
1.2	Final QA version for EC submission	13 February 2013

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Communication forums about Open Source Software (OSS) projects are valuable resources when it comes to evaluating OSS project quality. OSS newsgroups and mailing lists are message boards for improvement and upgrade announcements as well as for instructions and discussions about problems and defects encountered by the users. Bug tracking systems address the same issues in a more organised manner. Analysing messages in newsgroups, mailing lists and bug tracking systems that concern OSS projects can provide quality indicators, such as the speed at which user requests are addressed, the size of the developer and user community and the level of user satisfaction.

In this document, we present a review of methods to access OSS-related communication forums programmatically. We provide an overview of communication protocols and then discuss in detail the Application Programming Interfaces (API) supported by newsgroups, mailing lists and a number of popular bug tracking systems including Bugzilla[1], Trac[2], JIRA[3], Sourceforge Tracker[4], Google code Issue Tracker[5], GitHub Issue Tracker[6] and custom web-based forums. The API presentation focuses on Java, since Java is the programming language of choice in OSSMETER. Information about existing APIs is useful to develop software for obtaining text from communication channels for further processing.

Secondly, in this document we present an overview of the text processing workflow of work package 4. We focus on identifying data that needs to be stored in databases after each execution of the text processing workflow. We discuss the functionality and the output of each processing component separately. This presentation is valuable for the design of the part of the OSSMETER database where textual processing output will be stored. The complete description of storage facilities will be the topic of deliverable 5.2.

Finally, we present some early statistic results obtained by accessing a number of *Network News Transfer Protocol* (*NNTP*) newsgroups. The access mechanism was implemented as described in the first part of this document, in section 2.3.

[1] Bugzilla URL: bugzilla.org

[2] Trac URL: trac.edgewall.org

[3] JIRA URL: atlassian.com/software/jira

[4] Sourceforge tracker URL: sourceforge.net/apps/trac/sourceforge/wiki/Tracker

[5] Google code Issue Tracker URL: code.google.com/p/support/wiki/IssueTracker

[6] GitHub Issue Tracker URL: github.com/blog/411-github-issue-tracker

1. INTRODUCTION

1.1 OVERVIEW

In this document, we present our progress in relevance to the OSSMETER project task 4.1: "Data Collection, Domain Analysis and Infrastructure Development". Work package 4 aims to develop measures for evaluating the quality of user support and the level of user satisfaction over time in relation to Open Source Software (OSS), by analysing threads in OSS online support, discussion and bug tracking forums and relevant mailing lists. Task 4.1 focuses on the following aspects of data preparation:

- Access to newsgroups, bug tracking systems and mailing lists relevant to specific pieces of Open Source Software.
- Selection of metadata and other processing output that need to be stored for each piece of Open Source Software.

In section 2, we present the findings of our survey for existing interfaces for accessing data sources holding information valuable for the evaluation of the quality of OSS i.e. newsgroups, bug tracking systems and mailing lists. In section 3, we present an overview of the text processing component of OSSMETER, i.e. the outcome of work package 4, focusing on the information to be stored in the OSSMETER platform database. In section 4, we present statistics computed from a number of NNTP newsgroups. In Section 5 we present some risks and in Section 6 we conclude the deliverable

1.2 INTENTIONS

We intend to use the findings presented in section 2, as seed information to implement software for accessing textual information that relates closely to specific OSS. We then plan to exploit these resources automatically, by means of text processing methods. Research about existing access mechanisms to online discussions, support offered and bugs identified and addressed is indicative of the quality of an OSS and, according to the OSSMETER DoW, needs to be taken into account when evaluating OSS. We have already implemented software for accessing text in newsgroups and we present some statistics in section 4.

As far as our intentions relevant to section 3 are concerned, we intend to set the requirements for design databases to store the output of our text processing workflow. Persistence services will be provided by the OSSMETER platform.

1.3 OUTCOME

The outcomes of the research presented in this document are the following:

- In section 2, a detailed review of the most popular OSS communication channels and bug tracking systems. The review will simplify the implementation of crawlers able to download text from these sources and store it locally to be processed.

- In section 3, a presentation of the text processing workflow that will be implemented in work package 4, focussing on the processing output of components that needs to be stored in the OSSMETER platform database (i.e. the storage requirements of the text processing part of OSSMETER).
- In section 4, some early statistics of a number of newsgroups accessed programmatically.

2. ACCESS TO NEWSGROUPS, BUG TRACKING SYSTEMS AND MAILING LISTS

2.1 OVERVIEW

In this section, we present in detail our findings on existing access mechanisms to online repositories of textual data related to Open Source Software (OSS) projects. As such repositories, we consider newsgroups, mailing lists and a number popular bug tracking systems including Bugzilla, Trac, JIRA, Sourceforge Tracker, Google code Issue Tracker, GitHub Issue Tracker and custom web-based forums. Before presenting our findings in sections 2.3 - 2.5, we briefly discuss the access protocols employed by existing Application Programming Interfaces (APIs) in section 2.2.

2.2 ACCESS PROTOCOLS

Web service data exchanges mostly employ the following protocols or APIs: the Simple Object Access Protocol (SOAP), the Representational State Transfer (REST) web services, the Extensible Markup Language - Remote Procedure Calling (XML-RPC) protocol and the JavaScript Object Notation - Remote Procedure Calling (JSON-RPC) protocol.

The **Simple Object Access Protocol (SOAP)** is a specification for exchanging structured information in networks, such as the World Wide Web. It is applied extensively in the implementation of Web Services. It relies on Extensible Markup Language (XML) for its message format, and usually relies on other Application Layer protocols, most notably Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission.

The **Extensible Markup Language - Remote Procedure Calling (XML-RPC)** is a protocol for data exchange over the Internet, implementing messages as HTTP-POST requests. The message body of either a request or a reply is in XML format[7].

Another remote procedure calling protocol is the **JavaScript Object Notation - Remote Procedure Calling (JSON-RPC)**. The mechanism consists of two peers establishing a data connection. While the connection is established, each peer can use methods provided by the other peer, by exchanging HTTP requests.

The **Representational State Transfer (REST)** is an architecture for distributed software systems that communicate over networks. Similarly to SOAP, REST is also very

[7] An introduction in using XML-RPC in Java is available at: tldp.org/HOWTO/XML-RPC-HOWTO/xmlrpc-howto-java.html

popular in Web service design. RESTful web services, also called RESTful web APIs, are implemented using HTTP and the principles of REST: (a) the base URI for the web service, (b) the data format supported by the web service and (c) the set of operations supported by the web service using HTTP methods, e.g. GET, PUT, POST, or DELETE. Most web services support the XML format but any other data format can be supported.

REST is ideal for exposing public APIs over the internet to handle crude operations on data. In contrast, SOAP focusses on exporting functions, i.e. pieces of application logic. REST is usually preferred than SOAP, XML-RPC and JSON-RPC. Main reasons for this design choice are the advantages of REST in terms of performance and scalability as well as its flexibility with reference to data formats. REST supports many different data formats whereas SOAP supports XML, only.

2.3 NNTP NEWSGROUPS

Newsgroups are public message boards, i.e. repositories of messages posted from a group of users. The Network News Transfer Protocol (**NNTP**) is a popular application protocol used for transferring news articles between news servers and end user client applications. NNTP is also used as an API to access newsgroups programmatically. During the OSSMETER kick-off meeting, that took place on 24 and 25 October 2012 in York, it was collectively decided to start our study on the automated processing of the content of communication channels by focusing on OSS newsgroups that conform to the NNTP protocol.

The Apache Commons Net library (commons.apache.org/net) provides a comprehensive API for accessing content stored in NNTP servers. For example, the library allows retrieving the header, contents, senders and the unique identity number of each article, as well as the articles following or preceding it.

2.4 BUG TRACKING SYSTEMS

A Bug Tracking System (aka issue tracker) is an application for storing and managing discussions and actions related to particular software defects or enhancement requests. Bug Tracking Systems are useful to software users and developers to keep track of software defects (a.k.a. bug reports), support or enhancement requests, technical-support requests, development tasks (a.k.a. feature requests), patches and other issues to be addressed by the project development team.

In contrast to newsgroup and mailing list articles, bug tracking system requests capture additional metadata, such as the status and resolution of each request. In a simplistic scenario, one would expect only two status of request, either *open* or *fixed* and *closed*. However, in practice there are usually many more: *open*, *pending*, *closed*, *deleted* and others. Sometimes, bug status information is complemented by resolution. Resolutions for *open* status can be: *later*, *postponed*, *remind*, etc. Resolutions for *closed* status can be: *accepted*, *duplicate*, *fixed*, *invalid*, *later*, *out of date*, *rejected*, *won't fix*, etc.

In the remainder of this section, we discuss existing methods to access the textual data of the most popular bug tracking systems: Bugzilla, Trac, JIRA, Sourceforge Tracker,

Google code Issue Tracker, GitHub Issue Tracker and custom web-based forums in general. Of course there are more minor bug trackers, the majority of which can be accessed similarly to the ones presented below[8].

2.4.1 Bugzilla

Bugzilla is a web-based general-purpose bug tracking system and testing tool developed by the Mozilla foundation and originally used by the Mozilla project. Bugzilla provides a remote RESTful API called Bugzilla:REST API[9] in the form of a Bugzilla::WebService and also supports the following protocols: SOAP, XML-RPC and JSON-RPC[10].

The Bugzilla remote API consists of several methods that are grouped into five packages: bug, product, group, user, and bugzilla. Each API method is annotated with one of the following labels that describe the stability or the maturity of the method definition:

- **Stable**: the parameters and return values will not to change between versions of Bugzilla. Possible additions are guaranteed to be backward compatible.
- **Experimental**: the methods are almost stable, but might change in the future.
- **Unstable:** method definitions that are not guaranteed to exist in other Bugzilla versions.
- **Deprecated**: The use of these methods is discouraged because they have already been replaced by other methods and will not be included in future versions.

There are several open source Java libraries for consuming Bugzilla web services. Each client library offers a different level of coverage of the API and can be used with different versions of Bugzilla. The following libraries are available under the Apache License 2.0 or GNU LGPL v3: J2Bugzilla[11], B4J[12] (Bugzilla for Java), Bugzilla Library[13] and LightingBugAPI[14].

2.4.2 Trac

Trac is a wiki and issue tracking system that uses a minimalistic approach to web-based software project management. It is designed to improve the quality of produced software without interfering much with the development process and policies adopted by

[8] A extended comparison of a large number of bug tracking systems is available at: en.wikipedia.org/wiki/Comparison_of_issue-tracking_systems

[9] More information about the API are available at: wiki.mozilla.org/Bugzilla:REST_API

[10] A thorough discussion on Java access to BugZilla is posted at Nandana Mihindukulasooriya's blog: nandana.org/2012/10/bugzilla-web-service-interface-and-java-client-libraries.html

[11] J2Bugzilla is available at: code.google.com/p/j2bugzilla

[12] B4J is available at: techblog.ralph-schuster.eu/b4j-bugzilla-for-java

[13] The Bugzilla Library is available at: conqat.cs.tum.edu/index.php/Bugzilla_Library

[14] LightingBugAPI is available at: code.google.com/p/lightningbugapi

the software development team. Trac supports XML-RPC[15] and MYLIN. *TracDrops*[16] is a Java wrapper for the XML-RPC API.

2.4.3 JIRA

JIRA is a project management and bug tracking system developed and maintained by Atlassian since 2002. JIRA supports the following communication protocols: REST[17], SOAP[18], XML-RPC, JSON-RPC and MYLIN. Atlassian, provides a tutorial for developing a JIRA SOAP client[19] but suggests that REST should be preferred, since it is decided that future development will focus on REST[20]. Examples of using the JIRA REST API in a variety of programming languages including Java are also available[21].

2.4.4 SourceForge Tracker

SourceForge is a web-based source code repository, i.e. a centralized location for managing free and open source software development. SourceForge provides an embedded bug tracking system, SourceForge Tracker[22].

SourceForge comes with a custom read-only API for public project data[23]. It uses standard XML data formats, such as DOAP and RSS. Table 1 reports several entities accessible by the API, some of the parameters of each and the formats in which the corresponding data is available. Unfortunately, it is reported that the API is not actively maintained[24].

2.4.5 Google Code Issue Tracker

Google Code is Google's code hosting service, conceptually the same as SourceForge. The site contains documentation, discussion groups and blogs about developing open source software. Google code is not limited to Google products; any open source project can be hosted as long as it is open source.

[15] Details about the Trac XML-RPC API are discussed at: www.hossainkhan.info/content/trac-xml-rpc-api-reference

[16] TracDrops is available at: code.google.com/p/tracdrops

[17] JIRA REST API documentation is available at: docs.atlassian.com/jira/REST/latest

[18] Documentation of the JIRA SOAP service is available at: docs.atlassian.com/rpc-jira-plugin/latest/com/atlassian/jira/rpc/soap/JiraSoapService.html

[19] JIRA SOAP client tutorial: developer.atlassian.com/display/JIRADEV/Creating+a+JIRA+SOAP+Client

[20] Source: developer.atlassian.com/display/JIRADEV/JIRA+XML-RPC+Overview

[21] Some JIRA REST API examples are available at: confluence.atlassian.com/display/DOCSPRINT/The+Simplest+Possible+JIRA+REST+Examples

[22] Further information about the SourceForge Tracker is available at: sourceforge.net/apps/trac/sourceforge/wiki/Tracker

[23] Further information about the SourceForge API is available at: sourceforge.net/apps/trac/sourceforge/wiki/API

[24] Source: sourceforge.net/apps/trac/sourceforge/wiki/API

Google Code provides a custom bug tracking system called Google Code Issue Tracker[25]. In contrast to other bug tracking systems, Google Code Issue Tracker is designed to use a minimal set of fields only, and allows users to define and store their own labels according to the needs of their projects.

Google Code Issue Tracker is accessible programmatically via the Issue Tracker Data API[26]. The API allows client applications to view and update issues on Google Code in the form of Google Data API feeds. Applications can use the Issue Tracker Data API to create new issues & issue comments, request a list of issues, request issue comments for an issue, edit existing issues, and query for issues that match particular criteria. The Issue Tracker Data API has been deprecated and will be shut down on June 14, 2013[27]. Unfortunately, no alternative API has been announced, yet.

The Google Code Issue Tracker client library for .Net[28] allows getting all the issues from a project and submitting new ones, getting all comments associated with a project and creating new comments on an issue. Although there is no java client library available, there are detailed instructions how to use the Google Data API in Java[29].

Entity	Parameters	Formats
Project	changed_since, new_since, name, id	doap, json
User	username, id	json, rss
File Releases	project-id, package-id, path, recursive, order_by	rss
Project News	project-id, since	rss
Forum Posts	forum-id	rss
Tracker Artifacts	tracker-id, since	rss
Mailing List Messages	list-name	rss
Activity	project-id, since	rss

Table 1: Entities accessible by the SourceForge API

2.4.6 GitHub Issue Tracker

GitHub is a web-based hosting service for code management and development of software projects. It offers both paid plans for private repositories, and free accounts for open source projects.

The bug tracking system integrated in GitHub is GitHub Issue Tracker. Github supports a custom API, whose latest version is GitHub API v3. All API access is over HTTPS,

[25] Google Code Issue Tracker is available at: code.google.com/p/support/wiki/IssueTracker

[26] More information about the Google Code Issue Tracker Data API is available at: code.google.com/p/support/wiki/IssueTrackerAPI

[27] Source: googleblog.blogspot.co.uk/2012/12/winter-cleaning.html

[28] The Google Code Issue Tracker client library for .Net is available at: code.google.com/p/google-code-issue-tracker/

[29] Instructions for using the Google Data API in Java is available at: code.google.com/p/support/wiki/IssueTrackerAPIJava

and all data is sent and received in JSON format. There are several GitHub API client libraries in java:

- **GitHub Java API**[30]: This library aims to cover the entire GitHub v3 API. The library is currently used by the GitHub Mylyn connector.
- **GitHub API for Java**[31]: This library defines an object oriented representation of the GitHub API. The library offers limited coverage of the GitHub API, but is easily extensible.
- **POTD: GitHub API for Java**[32]: This library covers a small subset of the entire GitHub API, but it is argued that can be extended for the remaining parts.

2.4.7 Web-based forums

Except for the bug tracking systems described so far, there are several web-based forums used for reporting defects of specific pieces of OSS. For example, Eclipse uses FUDForum, which is based on NNTP[33]. Except for NNTP access, FUDForum can be accessed using FUDAPI[34], an API consisting of external calls. Another popular forum is PhPBB, used by KDE[35] and OpenOffice[36] among others. A JSON-RPC API for PhPBB is available[37], but no client library for Java.

2.5 MAILING LISTS

Some OSS projects, including all Apache projects[38], use mailing lists for communication between users and developers. Messages posted to such mailing lists are usually available in the form of an online archive[39]. Unfortunately, in most cases there is no standard API for accessing these archives. To gain access programmatically we would need to develop a crawler to download the HTML content of each webpage and then build parsers tailored to the HTML page format in question.

[30] The GitHub Java API is available at: org.eclipse.egit.github.core

[31] The GitHub API for Java is available at: github-api.kohsuke.org

[32] The POTD: GitHub API for Java is available at: weblogs.java.net/blog/kohsuke/archive/2010/04/18/potd-github-api-java

[33] A FUDForum is available at: www.eclipse.org/forums/index.php?t=thread&frm_id=22

[34] FUDAPI is available at: cvs.prohost.org/index.php/FUDAPI

[35] The PhPBB forum about KDE is available at: forum.kde.org

[36] The PhPBB forum about OpenOffice is available at: forum.openoffice.org/en/forum

[37] A JSON API for PhPBB is available at: github.com/pcrumm/phpbb.json

[38] Example mailing lists for an Apache project: velocity.apache.org/mail-lists.html

[39] Mailing list archive example: lists.w3.org/Archives/Public/html-tidy

3. STORAGE OF METADATA AND PROCESSING OUTPUT

3.1 OVERVIEW

In this section, we present an outline of the text processing workflow of OSSMETER that will be implemented in work package 4. The workflow consists of processing components each of which outputs a distinct kind of information, but all kinds are useful to assess different aspects of the quality of Open Source Software projects. Each of the components listed below is presented separately, focussing on the type of its output:

- article threader
- sentence splitter
- tokeniser
- part-of-speech tagger
- lemmatiser
- term extractor
- named entity recognizer
- question and answer extractor
- thread article classifier
- sentiment analyser

Processing component output sets the requirements for designing a database for storing it.

3.2 TEXT PROCESSING WORKFLOW

Since the volume of all articles in a newsgroup, bug tracking system or mailing list is theoretically unlimited and can practically be very large, we choose not to store their entire content. Instead, we prefer to only store information useful to compute quality indicators at any post-processing stage. Articles are processed on the fly, i.e. without storing their content permanently. While the entire text processing workflow is executed, the content of each article is stored temporarily in memory so that it is available to all processing components. In addition, every component has access to the stored output of all preceding components. Figure 1 illustrates a block diagram of the entire text processing system.

The processing components output that needs to be stored can range from statistics and counts of simple properties of the articles, such as the number of senders and articles per sender and the number of articles per thread, to the output of complex text processing components, such as terms, named entities and other classification labels.

In the following sections, we present a number of text processing components that will be applied to the textual data downloaded from newsgroups, bug tracking systems and mailing lists. We focus on the input and output of each component as well as the output that needs to be stored, summarised in table 2. Specifying this information is a prerequisite for creating the database schema.

In our description, we use associative arrays, also known as maps or dictionaries. Associative arrays are abstract data structures composed of a collection of (key, value) pairs, such that each possible key appears at most once in the collection. There are several implementations of associative arrays available, and also other equivalent ways of describing the same data.

3.3 ARTICLE METADATA

For each article, the following information should be stored in a database:

- **Article Statistics**: A tuple of metadata values associated with each article, such as the user that posted it, the timestamp of posting, etc. These values might be different for articles coming from diverse sources: newsgroups, mailing lists or bug tracking systems.

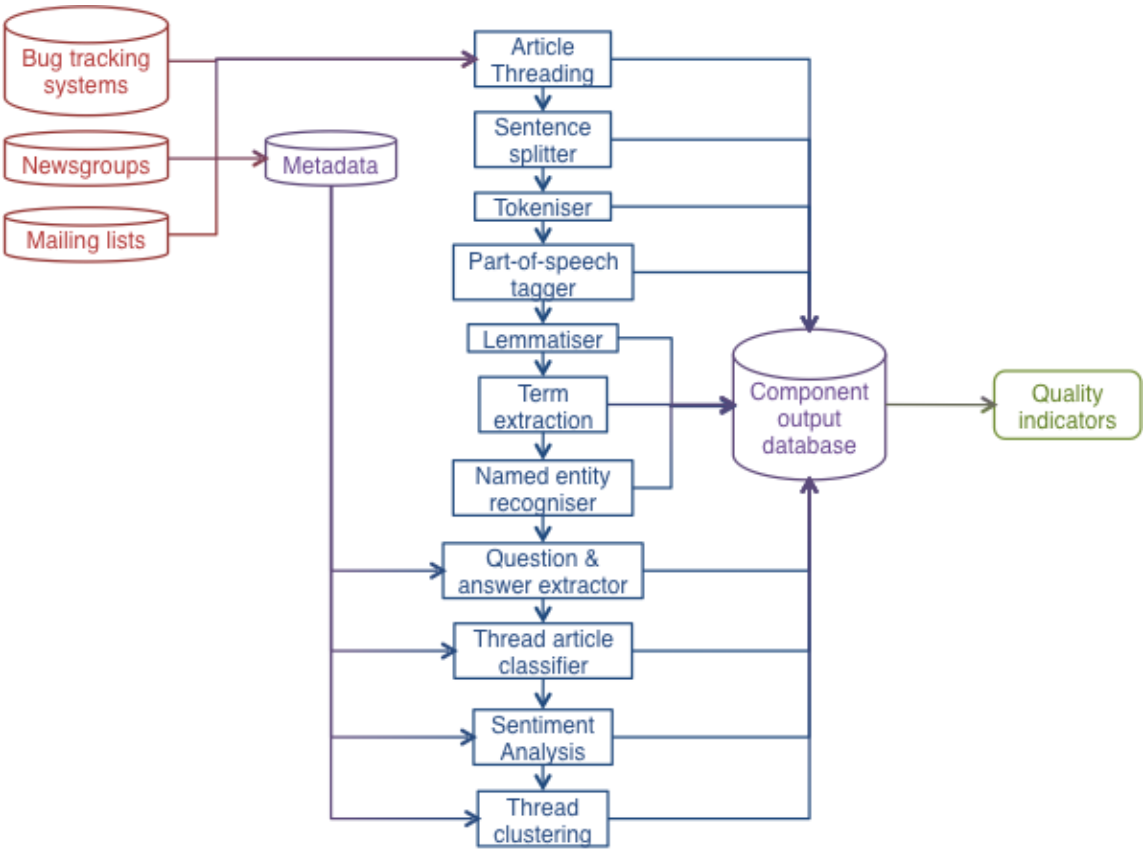

Figure 1: Block diagram of the entire system with a focus on the inputs and outputs of each component.

3.4 ORGANISATION IN THREADS

Bug tracking instances in sophisticated bug tracking systems are usually organised in threads, called issues, as discussed previously. By contrast, articles in newsgroups and mailing lists are not organized in threads. However, threading is very important for understanding the issues discussed and being able to evaluate the speed and quality of developer answers in response to user questions. To organise articles in threads we plan to reuse the Java implementation of an accurate and widely used message-threading algo-

rithm, originally devised by Zamie Zawinski. The algorithm uses the email subjects of emails, or the headings of articles. It trims iteratively the prefixes and suffices of these sequences and exploits their similarities[40].

A threading algorithm outputs one or more threads, i.e. sets of article Ids. Thus, the following information needs to be stored:

- **Articles - Threads**: A many-to-one associative array of article Ids to thread Ids. This associative array is useful to retrieve the thread to which an article was assigned.
- **Thread Statistics**: A tuple of statistic values needs to be stored for each thread, holding counts of features useful to evaluate the quality of communication. Examples of such features are: the number of messages in a thread, the number of users and/or developers participating, the duration between the first and last message etc. To compute these statistics no sophisticated components are needed.

The threading algorithm of Zamie Zawinski does not support updating. Every time new articles are added, the algorithm needs to be executed from scratch. We plan to inspect the running time of the algorithm for various newsgroup sizes and if needed develop a version of the algorithm that supports updating.

#	Name	Relation type
1	Articles - Threads	n - 1
2	Article Statistics	n - 1
3	Term dictionary	1 - 1
4	Lemma dictionary	1 - 1
5	Documents - Terms	1 - n
6	Terms - Lemmas	1 - n
7	Term Statistics	n - 1
8	Named Entity dictionary	1 - 1
9	Named Entity Type dictionary	1 - 1
10	Documents - Named Entities	1 - n
11	Named Entity Types	n - 1
12	Articles - Questions or Answers	n - 1
13	Articles - Topics	n - 1
14	Articles - Content Classes	n - 1
15	Articles - Polarity Scores	n – 1
16	Threads - Clusters	n - 1
17	Clusters - Cluster Labels	1 - 1

Table 2: Information to be stored in the database, per OSS

[40] More information about the threading algorithm can be found at: jwz.org/doc/threading.html

3.5 SENTENCE SPLITTING, TOKENISATION PART-OF-SPEECH TAGGING AND LEMMATISATION

Part-of-speech tagging is the task of deciding for the part-of-speech of each word in text. This information is very important, as it is the basis for the majority of subsequent text processing components such as Name Entity recognisers, sentiment analysers, and most types of text classifiers. Part of speech taggers consume text split into sentences and subsequently in distinct units equivalent to words. These tasks are performed by sentence-splitters and tokenisers, respectively.

Lemmatisers consume tokens and produce the lemma, i.e. a basic linguistic form of each input token. Using lemmatized tokens instead of surface forms reduces the variability of language, since more than one surface forms are mapped to a single lemma. Lemmas are useful to many subsequent text processing components, such as term extractors and named entity recognisers.

Sentence splitting, tokenization, part-of-speech tagging and lemmatisation are nowadays performed relatively quickly, since these preliminary tasks are simple and considered solved. Thus, there is no particular meaning in storing the output of these components. Instead, we choose to split the text of each NNTP article into sentences and tokens, tag tokens with parts-of-speech, lemmatise tokens and forward the output to subsequent components, without storing.

The text processing component repository of National Centre for Text Mining (NaCTeM), the University of Manchester, contains a state-of-the-art sentence splitter and a combined tokeniser, lemmatiser and part-of-speech tagger both developed in the Tsujii Laboratory, the University of Tokyo, a laboratory that no longer exists and used to cooperate closely with NaCTeM. The two research groups used to share a number of academics and researchers and have been merged into NaCTeM lately. The Genia Sentence Splitter[41] is based on a Maximum entropy model and achieves top results in the biomedical domain [1]. It will be retrained on text of the OSS domain, so as to be able to capture the sentence margins used in blogs, newsgroups and bug tracking systems. The Genia tagger[42] is reported to achieve state-of-the-art performance in a variety of textual domains [2]. Thus, we shall inspect its accuracy in the OSS domain and retrain if needed.

3.6 TERM EXTRACTION

Terms are sequences of words that refer to a domain specific concept. Consequently, they hold much of the semantic content of text. Terms are useful as features for training classifiers for topic segmentation and identification of content, as well as candidates for named entity recognition.

We plan to use Termine[43], a state-of-the-art term extractor developed at the NaCTeM.

[41] More information about the Genia Sentence Splitter is available at: nactem.ac.uk/y-matsu/geniass/

[42] More information about the Genia Tagger is available at: nactem.ac.uk/GENIA/tagger

[43] More information about Termine is available at: nactem.ac.uk/software/termine

Termine is based on the C-Value term extraction algorithm [3]. The algorithm computes a score for each candidate term based on the frequency of occurrence of the candidate both as a maximal candidate and as a substring of other longer candidates. Candidate terms are lemmatized sequences of tokens whose parts-of-speech conform to a pre-specified part-of-speech regular expression pattern.

For each OSS, we need to store the following information:

- **Term dictionary**: A one-to-one associative array of unique candidate term strings to unique term Ids. This associative array assigns a unique Id to each term candidate. In all other associative arrays, the term Ids are used instead of the terms candidates themselves for reduced size.
- **Lemma dictionary**: A one-to-one associative array of unique candidate term lemmas to unique lemma Ids. This associative array assigns a unique Id to each lemma. In all other associative arrays, the lemma Ids are used instead of the lemmas themselves for reduced size.
- **Documents - Terms**: A one-to-many associative array of unique NNTP article Ids to unique candidate term ids. This associative array is useful to retrieve which candidate terms occurred in which NNTP article.
- **Terms - Lemmas**: A one-to-many associative array of unique candidate term Ids to lemma ids. This associative array contains an entry for every lemma occurring in a tem candidate.
- **Term Statistics**: For each candidate term, five statistics need to be stored: the *length* of the term in tokens, its *marginal frequency*, its *nested cardinality*, its *nested frequency* and its *C-Value score*. All but the last statistics are integer valued, while C-Value is real valued. Marginal frequency, is the number of times the candidate term occurs as maximal candidate, i.e. not nested in longer candidates. Nested frequency is the number of times a term candidate occurs nested in linger candidates and nested cardinality is the number of these distinct longer candidates.

Storing the information above allows processing new documents by adding any possible new terms and then updating the C-Value scores of the whole collection.

3.7 NAMED ENTITY RECOGNITION

Named Entity Recognition is the task of identifying sequences in text that represent ontological concepts of pre-specified types/classes. Identifying Named Entities is closely related to ontologies. For example, consider a classification of OSS for various tasks, such as the Free/Open Source Software for Library and Information Management[44]. As shown in figure 2, *Apollo CMS*, *Apache::PageKit* and *SPIN CMS* are Content/Knowledge Management systems, *ExtPhr32*, *NEPHIS32*, *TexNet32* and *TheW32* are intended for Automated Classification and Thesaurus Construction, and *XBase*, *genSQL*, *MDBMS* and *SQLite* are Database Systems. A Named Entity Recogniser (NER) could either identify the sequences "*Apollo CMS*" and "*XBase*" as a Content/Knowledge Management system and Database System, respectively, or both sequences as OSS. NERs are usually based on dictionary matching and trainable Data Mining components, such as Support Vector Machines or Conditional Random Fields.

[44] Source: www.ncsi.iisc.ernet.in/raja/opendl/free-software.htm

To recognize the Named Entities in OSS related text, we plan to use NEMine[45] [4] as a basis, a biomedical NER trained to recognize gene and protein names, that was developed in NaCTeM. The recognizer needs to be extended, re-implemented and trained to recognize entities of the OSS domain.

For each OSS we need to store the Named Entities that occur in each article, their frequencies and their types. In particular, we need to store the following information:

- **Named Entity dictionary**: A one-to-one associative array of unique named entities to unique Ids. This associative array assigns a unique Id to each named entity. In all other associative arrays, the named entity Ids are used instead of the named entity strings for reduced size.

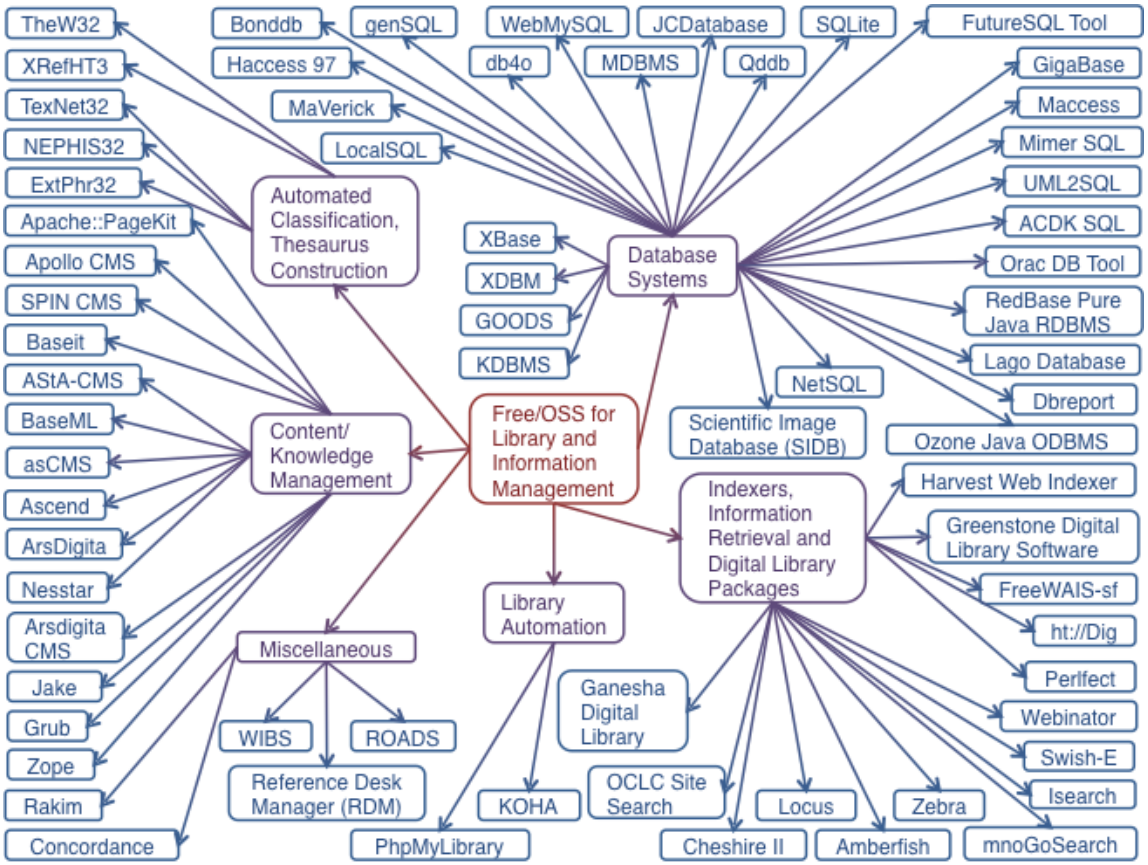

Figure 2: Ontology of free software and OSS for Library and Information Management

- **Named Entity Type dictionary**: A one-to-one associative array of unique named entities types to unique Ids. This associative array assigns a unique Id to each named entity type.
- **Documents - Named Entities**: A one-to-many associative array of unique NNTP article Ids to tuples, each consisting of a unique named entity id and an integer. Integers in tuples represent the frequency of occurrence of each named entity in

[45] A demonstrator of NEMine is available at: text0.mib.man.ac.uk/~sasaki/bootstrep/nemine.html

an article. This associative array is useful to retrieve which named entities occurred in which NNTP article and the frequency of this occurrence.

- **Named Entity Types**: A many-to-one associative array of unique named entity ids to named entity type Ids. This associative array holds information about the type of each named entity recognized.

3.8 EXTRACTION OF QUESTIONS AND ANSWERS FROM THREADS

In this task, each message in a thread will be classified in one of two coarse grained classes: questions or answers. For this purpose we shall develop classifiers based on syntactic and linguistic features, inspired by research for question-answering. In succession, we plan to match the questions to answers based on the similarity of semantic context. For brevity, we call **topic** a group consisting of a question and its corresponding answers. Although a thread ideally deals with a single topic, in practice threads often contain more topics, typically related with each other.

This processing component outputs the following information to be stored:

- **Articles - Questions or Answers**: A many-to-one associative array of unique article Ids to Booleans, modeling the question or answer.
- **Articles - Topics**: A many-to-one associative array of unique article Ids to Integers. Each integer represents a topic Id.

Due to the fact that we intend to identify topics within threads, the above associative arrays should be combined with "Articles - Threads", discussed in subsection 3.3, to draw the entire picture.

3.9 THREAD ANALYSIS AND CLASSIFICATION

This step aims to classify the questions and answers, identified in subsection 3.8, into a larger number of more fine-grained classes, such as: problems, solutions, complaints and feedback. Similarly to the previous task, machine learning models will be again trained to perform this classifications task. However, this fine grained classification will be based mostly on semantic features, in contrast to the previous task which was based mainly on syntactic ones. Due to this difference in the nature of features, we opt to address this classification in two levels. The information to be stored is the following:

- **Articles - Content Classes**: A many-to-one associative array of unique article Ids to Integers, each standing for a content class, e.g. problems and solutions.

3.10 THREAD SENTIMENT ANALYSIS

Sentiment analysis is the task of assigning a sentiment polarity score to sequences of words. In our case it is meaningful to assign scores to each thread article. Sentiment polarity can be one- or multi-dimensional, e.g. can model positive versus negative sentiment as opposed to interest, happiness, disappointment and anger. Sentiment polarity scoring is very important for the purposes of OSSMETER, because it can be used directly an OSS clustering indicator.

NaCTeM's sentiment analyser for social sciences [5] will be used as a basis. The output

information that needs to be stored is:

- **Articles - Polarity Scores**: A many-to-one associative array of unique article Ids to real numbers, in the case of one-dimensional polarity scores, or to tuples of positive real numbers, in the case of multi-dimensional scores.

3.11 THREAD CLUSTERING

As a last task, we plan to apply unsupervised clustering on threads. The result of this process will a set of clusters, i.e. unordered sets containing one or more threads. Each cluster will be labelled with a short phrase, indicative of its contents. This clustering will be useful to identify coarse content classes of threads, e.g. clusters might contain threads of closely related topics, and thus will allow evaluating the width of the variety of issues covered in the respective newsgroup, bug tracking system or mailing list.

We plan to use a clustering algorithm developed in the National Centre for Text Mining (NaCTeM) that has been successfully integrated in a number of customized search applications [5]. The main innovation of the algorithm lies in its ability to combine features of different nature coming from diverse sources. Thus, we expect that it will successfully combine the output of all preceding components, i.e. named entities, thread article classifications, sentimental polarities as well as other metadata statistics.

The output of the thread clustering component that needs to be stored is the following:

- **Threads - Clusters**: A many-to-one associative array of thread Ids to integer cluster Ids. The thread Ids should match the ones described in subsection 3.4.
- **Clusters - Cluster Labels**: A one-to-one associative array of integer cluster Ids to string cluster labels.

4. NNTP NEWSGROUP STATISTICS

As mentioned earlier, during the kick-off OSSMETER meeting it was collectively decided to start implementing software to access NNTP newsgroups. Following the feedback received, we computed various early statistics, useful for OSS quality evaluation.

The software implemented uses the Apache Commons Net library (commons.apache.org/net). Initially, it connects to an NNTP server, specified by the user, so as to be able to access the articles of any newsgroup hosted on it. Each article is associated with a number of fields: header, body, date and sender. The software occupies the article threading algorithm discussed in subsection 3.4 to organise the articles of each newsgroup in threads. In succession, as discussed in subsection 2.3, the software computes various early statistics, useful for OSS quality evaluation. In later stages of development, the content of each newsgroup of interest will be consumed by the text processing workflow discussed in section 3. At present, the following early statistics were computed:

- **number of articles**
- **number of threads**
- **minimum thread size**: the smallest number of articles in a thread
- **maximum thread size**: the largest number of articles in a thread
- **average thread size**: the largest number of articles in a thread
- **number of users**
- **minimum number of users in a thread**
- **maximum number of users in a thread**
- **average number of threads per user**
- **duration in days**: the difference in days between the timestamp of the earliest and the latest article
- **minimum thread duration in days**
- **maximum thread duration in days**

Some of these statistics (i.e. the number of threads and messages, the number of users in general and per thread) are crude indicators of the activity level in a newsgroup, i.e. the interest of the developers and users to discuss about it. The minimum, maximum and average thread size indicates the quality of responses submitted by developers to user requests. Similarly, the durations are indicative of the maturity of the OSS project and of the support provided.

Statistics for a number of NNTP newsgroups are shown in table 3. Some of these newsgroups are representative of newsgroups we expect to work with in OSSMETER, since they are newsgroups of real OSS projects. Such newsgroups are: netscape.public.mozilla.qa.editor, mozilla.dev.apps.calendar, mozilla.support.bugzilla and mozilla.dev.apps.thunderbird.

#	newsgroup	number of articles	number of threads	min thread size	max thread size	avg thread size	users	min users in thread	max users in thread	avg threads per user	duration (days)	min duration in thread (days)	max duration in thread (days)
1	netscape.public.mozilla.newsclips	43	15	2	2	2.87	16	2	2	0.94	3090	0	25
2	mozilla.tools.pulse	163	58	1	9	2.81	21	1	5	2.76	696	0	14
3	mozilla.test.multimedia	10	5	1	1	2.00	4	1	1	1.25	97	0	0
4	mozilla.dev.tech.editor	366	113	1	9	3.24	110	1	6	1.03	2515	0	636
5	netscape.public.mozilla.xpfe.checkins	70	36	2	2	1.94	32	1	1	1.13	2642	0	0
6	mozilla.reps.general	5302	977	1	65	5.43	506	1	28	1.93	686	0	369
7	mozilla.community.ignite	5	4	1	1	1.25	2	1	1	2.00	319	0	0
8	mozilla.hispano.labs	448	65	1	52	6.89	43	1	12	1.51	253	0	199
9	netscape.public.mozilla.qa.editor	50	26	1	2	1.92	26	1	2	1.00	2295	0	0
10	mozilla.dev.tech.crypto	11329	1930	1	178	5.87	1020	1	31	1.89	2558	0	1778
11	mozilla.dev.tech.java	794	306	1	14	2.59	258	1	5	1.19	2475	0	288
12	mozilla.mozillians	1208	236	1	51	5.12	174	1	18	1.36	685	0	253
13	mozilla.dev.extensions	17409	5652	1	32	3.08	3066	1	15	1.84	2572	0	2264
14	mozilla.dev.apps.calendar	5948	1578	1	74	3.77	862	1	19	1.83	2562	0	275
15	mozilla.dev.tech.js-engine	6276	1922	1	31	3.27	1230	1	12	1.56	2558	0	612
16	netscape.public.mozilla.jobs	324	238	1	6	1.36	184	1	3	1.29	2719	0	1102
17	mozilla.education	4	2	3	3	2.00	3	2	2	0.67	1379	2	2
18	mozilla.community.german	13	9	1	2	1.44	8	1	2	1.13	5	0	2
19	mozilla.webapps.pt-br	7	4	1	2	1.75	4	1	1	1.00	155	0	0
20	mozilla.dev.tech.rdf	294	88	1	18	3.34	86	1	8	1.02	2462	0	1835
21	mozilla.dev.embedding	3906	1340	1	56	2.91	959	1	24	1.40	2543	0	576
22	mozilla.qa.webapps	25	14	2	7	1.79	6	1	5	2.33	78	0	0
23	mozilla.community.belgium	873	237	1	16	3.68	62	1	7	3.82	719	0	49
24	netscape.public.mozilla.xpinstall	587	261	1	7	2.25	269	1	5	0.97	2642	0	2486
25	netscape.public.mozilla.nspr	606	234	1	15	2.59	197	1	5	1.19	2569	0	237
26	mozilla.community.australia	17	9	1	5	1.89	13	1	4	0.69	575	0	390
27	mozilla.dev.automation	670	226	1	29	2.96	49	1	7	4.61	526	0	112
28	mozilla.community.drumbeat	2176	668	1	55	3.26	275	1	22	2.43	1007	0	788
29	mozilla.dev.tree-management. tracemonkey	69	69	0	0	1.00	4	-	-	17.25	376	-	-
30	netscape.public.mozilla.qa.general	873	398	1	11	2.19	383	1	8	1.04	2571	0	632
31	mozilla.dev.tech.xpcom	4193	1427	1	46	2.94	989	1	11	1.44	2527	0	1201
32	netscape.public.mozilla.wishlist	11364	3876	1	55	2.93	2866	1	24	1.35	2642	0	1017
33	mozilla.reps.students	4	3	0	0	1.33	1	0	0	3.00	80	0	0
34	mozilla.hispano.general	873	118	1	168	7.40	125	1	44	0.94	59	0	47
35	mozilla.dev.webapps	1310	230	1	140	5.70	133	1	22	1.73	400	0	163
36	mozilla.support.bugzilla	25424	8875	1	38	2.86	5488	1	19	1.62	1259	0	1259
37	mozilla.community.games	204	74	1	14	2.76	39	1	11	1.90	322	0	89
38	mozilla.dev.planning	22257	5195	1	264	4.28	1533	1	70	3.39	2552	0	726
39	mozilla.dev.popcorn	270	125	1	11	2.16	28	1	6	4.46	248	0	13
40	mozilla.community.turkey	78	37	1	8	2.11	25	1	6	1.48	488	0	52
41	mozilla.community.arab-world	573	178	1	18	3.22	69	1	10	2.58	650	0	399
42	mozilla.drumbeat.website	332	122	1	13	2.72	62	1	8	1.97	1043	0	144
43	netscape.public.mozilla.crypto	4808	1209	1	94	3.98	865	1	13	1.40	2752	0	2132
44	mozilla.dev.l10n.de	446	120	1	21	3.72	84	1	10	1.43	1918	0	1496
45	mozilla.community.philippines	794	249	1	20	3.19	99	1	8	2.52	1190	0	711
46	mozilla.tools	517	173	1	24	2.99	88	1	11	1.97	1171	0	331
47	mozilla.announce.compatibility	8	8	0	0	1.00	3	-	-	2.67	594	-	-
48	mozilla.test	8149	2361	1	860	3.45	1463	1	355	1.61	2581	0	2561
49	netscape.public.mozilla.unix	2427	710	1	23	3.42	709	1	9	1.00	2842	0	150
50	netscape.public.mozilla.os2	14750	2461	1	73	5.99	914	1	39	2.69	2222	0	2222
51	mozilla.community.ireland	60	11	1	29	5.45	7	1	4	1.57	369	0	104
52	mozilla.dev.l10n.pt-br	566	141	1	20	4.01	69	1	8	2.04	615	0	389
53	mozilla.support.webtools	1833	686	1	17	2.67	495	1	9	1.39	2571	0	1022
54	mozilla.dev.quality	3171	1336	1	44	2.37	426	1	17	3.14	2547	0	616
55	mozilla.community.greece	104	34	1	22	3.06	16	1	7	2.13	487	0	18
56	mozilla.dev.builds	5028	1480	1	44	3.40	1047	1	20	1.41	2551	0	1590

57	mozilla.dev.webapi	1809	273	1	78	6.63	188	1	14	1.45	525	0	455
58	mozilla.support.seamonkey	76968	10532	1	230	7.31	3106	1	36	3.39	2620	0	2332
59	mozilla.dev.tech.xforms	3191	758	1	59	4.21	391	1	8	1.94	2552	0	546
60	mozilla.accessibility	25	16	1	3	1.56	16	1	3	1.00	343	0	2
61	mozilla.events	218	60	1	4	3.63	43	1	3	1.40	1900	0	16
62	netscape.public.mozilla.general	40364	8146	1	443	4.96	6346	1	55	1.28	2979	0	2979
63	mozilla.dev.mdc	2884	892	1	65	3.23	505	1	16	1.77	2557	0	960
64	netscape.public.mozilla.xml	2093	726	1	156	2.88	832	1	156	0.87	2830	0	789
65	mozilla.wishlist	2602	813	1	37	3.20	709	1	15	1.15	2532	0	1018
66	mozilla.dev.tech.js-engine.rhino	2559	787	1	31	3.25	498	1	10	1.58	1656	0	819
67	mozilla.tools.socorro	348	140	1	20	2.49	47	1	5	2.98	706	0	73
68	mozilla.dev.tech.xml	614	144	1	22	4.26	146	1	7	0.99	2528	0	168
69	mozilla.community.artzilla	2	2	0	0	1.00	2	-	-	1.00	3	-	-
70	mozilla.dev.tech.xpinstall	256	81	1	7	3.16	82	1	4	0.99	2531	0	185
71	mozilla.dev.developer-tools	31	9	2	10	3.44	15	2	6	0.60	37	0	0
72	netscape.public.mozilla.xpcom	3176	1243	1	28	2.56	857	1	8	1.45	2642	0	200
73	mozilla.community.indonesia	211	66	1	15	3.20	34	1	8	1.94	399	0	156
74	mozilla.community.uganda	180	102	1	14	1.76	34	1	4	3.00	464	0	56
75	netscape.public.mozilla.mathml	582	243	1	24	2.40	225	1	7	1.08	2697	0	168
76	mozilla.community.sweden	7	7	0	0	1.00	4	-	-	1.75	201	-	-
77	mozilla.dev.tech.js-engine.internals	975	183	1	68	5.33	91	1	14	2.01	787	0	357
78	mozilla.dev.tree-management	24282	20017	1	82	1.21	295	1	17	67.85	1464	0	855
79	mozilla.community.web-standards	250	45	1	39	5.56	60	1	16	0.75	1149	0	898
80	mozilla.reps.webdev	121	36	1	16	3.36	38	1	7	0.95	674	0	15
81	netscape.public.mozilla.xbl	843	234	1	30	3.60	171	1	10	1.37	2752	0	69
82	mozilla.dev.l10n.my	118	67	1	5	1.76	20	1	4	3.35	575	0	107
83	mozilla.dev.tech.network	1424	354	1	45	4.02	264	1	8	1.34	2519	0	1963
84	mozilla.support.calendar	8580	2339	1	87	3.67	1880	1	18	1.24	2571	0	1682
85	netscape.public.dev.xul	3217	1151	1	22	2.79	731	1	19	1.57	2928	0	1928
86	mozilla.community.b2g	7	6	2	2	1.17	7	2	2	0.86	244	0	0
87	mozilla.dev.tech.crypto.checkins	7738	2598	2	101	2.98	97	1	19	26.78	2557	0	2533
88	mozilla.dev.accessibility	3147	946	1	32	3.33	402	1	17	2.35	2459	0	1339
89	mozilla.community.tunisia	514	191	1	14	2.69	90	1	13	2.12	679	0	21
90	mozilla.dev.extensions.br	183	59	1	3	3.10	45	1	2	1.31	1270	0	198
91	mozilla.dev.identity	3681	806	1	54	4.57	356	1	17	2.26	655	0	328
92	netscape.public.mozilla.rt-messaging	158	79	1	2	2.00	67	1	2	1.18	2254	0	22
93	mozilla.dev.tech.layout	2489	647	1	58	3.85	348	1	13	1.86	2531	0	1436
94	mozilla.community.morocco	15	6	1	4	2.50	6	1	3	1.00	121	0	0
95	netscape.public.mozilla.rhapsody	169	118	1	3	1.43	113	1	3	1.04	2642	0	22
96	mozilla.dev.tech.plugins	2099	911	1	36	2.30	685	1	8	1.33	2544	0	770
97	netscape.public.mozilla.prefs	534	260	1	18	2.05	263	1	7	0.99	1859	0	2048
98	mozilla.reps.council	2051	431	1	28	4.76	41	1	11	10.51	668	0	280
99	mozilla.dev.apps.thunderbird	13521	3113	1	188	4.34	1675	1	58	1.86	2561	0	1174
100	mozilla.community.bangladesh	387	80	1	14	4.84	25	1	8	3.20	394	0	158
101	mozilla.it	40	8	2	9	5.00	20	2	8	0.40	200	1	148
102	mozilla.webmaker.canada.bc	40	23	1	5	1.74	14	1	5	1.64	60	0	11
103	netscape.public.mozilla.xpfe	2300	768	1	17	2.99	431	1	6	1.78	2765	0	763
104	mozilla.legal	464	112	1	26	4.14	128	1	10	0.88	2558	0	617
105	mozilla.dev.tech.xul	7667	2336	1	33	3.28	1288	1	10	1.81	2556	0	1734
106	mozilla.dev.shumway	10	3	2	5	3.33	7	1	3	0.43	50	0	0
107	mozilla.dev.apps.chatzilla	336	79	1	25	4.25	85	1	6	0.93	2480	0	565
108	netscape.public.mozilla.documentation	1152	388	1	84	2.97	395	1	19	0.98	2169	0	790
109	mozilla.dev.ports.os2	18462	2310	1	117	7.99	513	1	27	4.50	2562	0	2307
110	mozilla.dev.l10n	13674	2798	1	150	4.89	1126	1	64	2.48	2561	0	1646
111	netscape.public.mozilla.patches	223	134	1	8	1.66	158	1	8	0.85	2569	0	832
112	mozilla.dev.platforms.mobile	1855	654	1	53	2.84	514	1	28	1.27	1928	0	458
113	mozilla.community.algeria	21	16	1	2	1.31	10	1	2	1.60	426	0	0
114	mozilla.dev.l10n.uk	141	43	1	7	3.28	17	1	4	2.53	1050	0	46
115	netscape.public.mozilla.announce	154	95	1	6	1.62	95	1	6	1.00	914	0	50
116	netscape.public.mozilla.plugins	1539	737	1	22	2.09	758	1	18	0.97	2572	0	1502
117	netscape.public.mozilla.oji	137	70	1	7	1.96	70	1	5	1.00	2765	0	25
118	mozilla.dev.themes	736	132	1	99	5.58	200	1	54	0.66	2561	0	261
119	mozilla.community.armyofawesome	1	1	0	0	1.00	1	-	-	1.00	0	-	-
120	mozilla.community.ghana	29	12	1	8	2.42	17	1	7	0.71	530	0	4

121	netscape.public.mozilla.qt	235	138	1	6	1.70	146	1	4	0.95	2836	0	1893
122	netscape.public.mozilla.l10n	6263	1753	1	106	3.57	691	1	42	2.54	2785	0	2785
123	mozilla.community.new-zealand	13	9	1	3	1.44	7	1	2	1.29	543	0	0
124	mozilla.dev.tech.mathml	664	147	1	89	4.52	133	1	11	1.11	2480	0	334
125	mozilla.tools.l10n	16	10	1	2	1.60	5	1	2	2.00	369	0	2
126	mozilla.dev.usability	2691	426	1	412	6.32	467	1	110	0.91	2290	0	640
127	mozilla.dev.tech.js-engine.rivertrail	30	6	1	11	5.00	7	1	4	0.86	74	0	1
128	mozilla.community.senegal	155	80	1	11	1.94	54	1	8	1.48	603	0	290
129	netscape.public.mozilla.mstone	107	58	1	3	1.84	60	1	3	0.97	2569	0	34
130	mozilla.community.europe	7	3	1	3	2.33	7	1	3	0.43	334	0	0
131	mozilla.dev.marketplace	78	24	1	9	3.25	28	1	6	0.86	358	0	19
132	mozilla.community.brasil	4804	864	1	94	5.56	190	1	34	4.55	615	0	376
133	mozilla.dev.l10n.cs	498	144	1	15	3.46	100	1	6	1.44	2469	0	205
134	mozilla.dev.l10n.km	44	24	1	5	1.83	10	1	3	2.40	492	0	6
135	netscape.public.mozilla.unix.checkins	131	85	2	2	1.54	81	1	2	1.05	2642	0	7
136	mozilla.reps.mentors	872	162	1	26	5.38	52	1	13	3.12	423	0	68
137	mozilla.dev.l10n.vi	294	92	1	23	3.20	34	1	8	2.71	576	0	272
138	mozilla.dev.webdev	330	93	1	19	3.55	83	1	12	1.12	140	0	70
139	mozilla.dev.l10n.ta	3	2	0	0	1.50	2	-	-	1.00	62	-	-
140	mozilla.community.switzerland	12	7	1	2	1.71	7	1	2	1.00	49	0	0
141	mozilla.dev.b2g	4194	1059	1	166	3.96	607	1	25	1.74	521	0	207
142	mozilla.dev.l10n.pl	274	82	1	6	3.34	61	1	5	1.34	2164	0	323
143	mozilla.dev.security.policy	7901	699	1	154	11.30	326	1	24	2.14	1457	0	705
144	mozilla.dev.l10n.lo	13	9	4	4	1.44	7	3	3	1.29	383	0	0
145	netscape.public.mozilla.jsdebugger	271	128	1	17	2.12	127	1	4	1.01	2985	0	124
146	mozilla.dev.sumo	11	7	1	2	1.57	6	1	2	1.17	167	0	0
147	mozilla.tools.elmo	48	29	1	4	1.66	5	1	3	5.80	451	0	3
148	mozilla.drumbeat.barcelona	250	104	1	18	2.40	33	1	10	3.15	585	0	17
149	mozilla.reps.comms	30	15	1	7	2.00	15	1	3	1.00	100	0	0
150	mozilla.dev.pdf-js	275	116	1	23	2.37	83	1	8	1.40	231	0	139
151	mozilla.dev.l10n.fa	221	56	1	7	3.95	29	1	5	1.93	1890	0	44
152	netscape.public.mozilla.security	1889	507	1	58	3.73	490	1	15	1.03	2713	0	342
153	netscape.public.mozilla.browser	21583	5437	1	115	3.97	4669	1	26	1.16	2985	0	3116
154	netscape.public.mozilla.ui	682	298	1	19	2.29	336	1	10	0.89	2985	0	476
155	mozilla.dev.media	388	118	1	19	3.29	65	1	7	1.82	310	0	186
156	mozilla.support.other	485	126	1	11	3.85	146	1	6	0.86	2036	0	503
157	mozilla.dev.l10n.new-locales	117	38	1	14	3.08	37	1	8	1.03	524	0	258
158	netscape.public.mozilla.editor	985	383	1	26	2.57	369	1	7	1.04	2985	0	321
159	mozilla.community.india	961	291	1	39	3.30	194	1	16	1.50	1403	0	1069
160	netscape.public.mozilla.qa.browser	281	148	1	6	1.90	166	1	4	0.89	2985	0	142
161	mozilla.support.instantbird	43	20	1	6	2.15	16	1	3	1.25	72	0	47
162	mozilla.dev.privacy	54	23	1	10	2.35	19	1	6	1.21	264	0	119
163	mozilla.dev.apps.firefox	20491	4199	1	356	4.88	3318	1	102	1.27	2571	0	1532
164	mozilla.community.kenya	833	289	1	20	2.88	114	1	14	2.54	866	0	373
165	netscape.public.mozilla.layout	1342	448	1	30	3.00	425	1	11	1.05	3425	0	3425
166	netscape.public.mozilla.license	536	233	1	18	2.30	227	1	5	1.03	2985	0	168
167	netscape.public.mozilla.jseng	5275	1697	1	35	3.11	1130	1	9	1.50	2985	0	3008
168	mozilla.community.romania	2	2	0	0	1.00	1	-	-	2.00	0	-	-
169	mozilla.dev.apps.bugzilla	2819	717	1	57	3.93	390	1	19	1.84	2570	0	1521
170	mozilla.dev.security	3138	597	1	159	5.26	476	1	25	1.25	2521	0	377
171	netscape.public.mozilla.accessibility	864	171	1	317	5.05	157	1	17	1.09	1343	0	317
172	mozilla.support.firefox	151762	22126	1	448	6.86	14019	1	90	1.58	2572	0	2368
173	netscape.public.mozilla.dom	2678	809	1	105	3.31	671	1	28	1.21	2985	0	2569
174	mozilla.dev.tech.xbl	1004	231	1	20	4.35	163	1	8	1.42	2540	0	1674
175	mozilla.dev.servo	246	36	1	34	6.83	31	1	9	1.16	362	0	45
176	mozilla.dev.identity.performance	2	2	0	0	1.00	2	-	-	1.00	2	-	-
177	netscape.public.mozilla.checkins	75	38	1	4	1.97	35	1	3	1.09	2573	0	0
178	mozilla.drumbeat.privacy-icons	10	6	2	4	1.67	7	2	3	0.86	588	5	14
179	mozilla.dev.mdn	864	288	1	30	3.00	65	1	11	4.43	693	0	72
180	mozilla.airmozilla	1	1	0	0	1.00	1	-	-	1.00	0	-	-

Table 3: Statistics for 180 NNTP newsgroups

5. RISKS

In this section we summarise several risks that we came across during our research about newsgroup, mailing list and bug tracking system APIs. Of course, each risk is valid only if it is decided to evaluate projects of the corresponding repository.

The Sourceforge API is not actively maintained. As a result, we might come across some difficulties to use it, especially for Sourceforge versions later than the last API maintenance. However, since we are interested in minimal API functionality, i.e. we just need to read content without modifying any information, we expect a minor delay, only.

The Google Code Issue Tracker API has been deprecated and will be shut down on June 14, 2013[46]. Absence of an API to the Issue Tracker will cause significant delay, since we would need to develop a custom HTML parser to parse webpages directly. However, we expect that the deprecated API will be replaced by some other access mechanism probably closer to the deadline.

Finally, we identified that there is no Java client library available for the PhPBB JSON-RPC API. As a result, we expect some minor delay for implementing it in Java, using the functions of the JSON-RPC protocol.

6. CONCLUSION

In this document, we have presented our progress concerning OSSMETER Task 4.1. Section 2 discussed a thorough review of existing Application Programming Interfaces (API) supported by newsgroups, mailing lists and bug tracking systems related to OSS projects. This is a first step in the process of accessing textual data in these communication resources. In the sequel, posts will be downloaded and processed by the text processing workflow introduced in section 3. We discussed each component separately and focussed on the type of output data that needs to be stored in the OSSMETER platform database. The type of this data poses design requirements to be addressed by the OSSMETER data persistence infrastructure, which will be specified in deliverable 5.2. We have already implemented the software for accessing newsgroups and we presented statistics computed about a number of them in section 4.

[46] Source: googleblog.blogspot.co.uk/2012/12/winter-cleaning.html

REFERENCES

[1] *Sætre, Rune, Kazuhiro Yoshida, Akane Yakushiji, Yusuke Miyao, Yuichiro Matsubayashi and Tomoko Ohta.* 2007. **AKANE System: Protein-Protein Interaction Pairs in BioCreAtIvE2 Challenge, PPI-IPS subtask.** In Proceedings of the Second BioCreative Challenge Evaluation Workshop. pp. 209--212, CNIO.

[2] *Yoshimasa Tsuruoka, Yuka Tateishi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun'ichi Tsujii.* 2005. **Developing a Robust Part-of-Speech Tagger for Biomedical Text.** Advances in Informatics - 10th Panhellenic Conference on Informatics, LNCS 3746, pp. 382-392.

[3] *Katerina Frantzi, Sophia Ananiadou, and Hideki Mima.* 2000. **Automatic recognition of multi-word terms.** International Journal of Digital Libraries 3(2), pp.117-132.

[4] *Yutaka Sasaki, Yoshimasa Tsuruoka, John McNaught, and Sophia Ananiadou*. 2008. **How to make the most of NE dictionaries in statistical NER.** BMC bioinformatics, 9 Suppl 11.

[5] *Scott Piao, Yoshimasa Tsuruoka, and Sophia Ananiadou*. **Sentiment analysis with knowledge resource and NLP tools**. 2009. The International Journal of Interdisciplinary Social Sciences, 4(5):17–28.

[6] *Ioannis Korkontzelos, Tingting Mu, Angelo Restificar, and Sophia Ananiadou*. 2011. **Text mining for efficient search and assisted creation of clinical trials**. In Proceedings of the ACM fifth international workshop on Data and text mining in biomedical informatics, DTMBIO ’11, pages 43-50, New York, NY, USA. ACM.