World Wide Web Journal manuscript No.
(will be inserted by the editor)

Towards Building a Blog Preservation Platform

Nikos Kasioumis · Vangelis Banos · Hendrik Kalb

Received: yyyy-mm-dd / Accepted: yyyy-mm-dd

Abstract Social media content and user participation has increased dramatically since the advent of Web 2.0. Blogs have become relevant to every aspect of business and personal life. Nevertheless, we do not have the right tools to aggregate and preserve blog content correctly, as well as to manage blog archives effectively. Given the rising importance of blogs, it is crucial to build systems to facilitate blog preservation, safeguarding an essential part of our heritage that will prove valuable for current and future generations. In this paper, we present our work in progress towards building a novel blog preservation platform featuring robust digital preservation, management and dissemination facilities for blogs. This work is part of the BlogForever project which is aiming to make an impact to the theory and practice of blog preservation by creating guidelines and software that any individual or organization could use to preserve their blogs.

Keywords Blog preservation · Web archiving

1 Introduction

In this paper, we present how specialised blog archiving can overcome problems of current web archiving. We introduce a specialised platform that exploits the characteristics of blogs to enable improved archiving.

Nikos Kasioumis
CERN, CH-1211 Geneva 23, Switzerland
E-mail: nikos.kasioumis@cern.ch

Vangelis Banos
Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
E-mail: vbanos@gmail.com

Hendrik Kalb
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany
E-mail: hendrik.kalb@tu-berlin.de

Web archiving is an important aspect in the preservation of cultural heritage [22]. Web preservation is defined as the capturing, management and preservation of websites and web resources. Web preservation must be a start-to finish activity, and it should encompass the entire lifecycle of the web resource [4]. The most notable web archiving initiative is the Internet Archive[1] which has been operating since 1996. In addition, a large variety of projects from national and international organizations are working on web preservation related activities. For instance, many national memory institutions such as national libraries understood the value of web preservation and developed special activities towards this goal [32]. All active national web archiving efforts, as well as some academic web archives are members of the International Internet Preservation Consortium (IIPC)[2]. However, it is a complex task that requires a lot of resources. Therefore, web archiving is always a selective process, and only parts of the existing web are archived [11,2]. Contrary to traditional media like printed books, web pages can be highly dynamic. Therefore, the selection of archived information comprises of not only the decision of what to archive (e.g. topic or regional focus) but also additional parameters such as the archiving frequency per page, and parameters related to the page request (e.g. browser, user account, language etc.) [22]. Thereby, the selection seems often to be driven by human publicity and search engine discoverability [2]. The archiving of large parts of the web is a highly automated process, and the archiving frequency of a webpage is normally determined by a schedule for harvesting the page. Thus, the life of a website is not recorded appropriately if the page is updated more often than it is crawled [14]. Next to the harvesting problem of web archiving, the access of the archived information is inadequate for sophisticated retrieval. Archived information can be accessed only on site or page level according to a URI because analysis and management of current web archiving does not distinguish between different kinds of web pages. Thus, a page with a specific structure like blogs is handled as a black box.

Blog archiving is a subcategory of web archiving. The significance of blogs in every aspect of business and private life cannot be understated. Blogs are used from teaching physics in Latin America [34] to facilitating fashion discussions by young people in France [7]. The collective outcome of all blogs, their content, interconnections and influences constitute the oeuvre of the blogosphere, a unique socio-technical artefact of high significance [17]. However, current web archiving tools cause several problems for blog preservation. First, the tools for aquisition and curation use a schedule based approach to determine the point in time when the content should be captured for archiving, leading to loss of information in case it is updated more often than it is crawled [14]. On the other hand, unnecessary harvests and archived duplicates occur if the blog is less often updated than the crawling schedule, and if the whole blog is harvested again instead of a selective harvest of the new pages. Therefore, an approach which considers update events (e.g. new post,

[1] `http://archive.org`

[2] `http://netpreserve.org`

new comment, etc.) as triggers for crawling activities would be more suitable. Thereby, Rich Site Summary (RSS) feeds and ping servers can be utilized as triggers. Secondly, the general web archiving approach considers the webpage as the digital object that is preserved and can be accessed. However, a blog consists of several smaller entities like posts and comments which can be exploited by a specialised archiving system to facilitate further analysis, e.g. the dissemination of issues or events. In summary, we identify several problems for blog archiving with current web archiving tools:

- Aggregation is performed with a schedule based approach without considering web site updates. This causes incomplete content aggregation if the update frequency of the contents is higher than the schedule predicts [14, 32].
- Traditional aggregation uses brute-force methods to crawl without taking into account what is the updated content of the target website. Thus, the performance of both the archiving system and the crawled system are affected unnecessarily [32].
- Current web archiving solutions do exploit the potential of the inherent structure of blogs. Therefore, while blogs provide a rich set of information entities, structured content, APIs, interconnections and semantic information [21], the management and end-user features of existing web archives are limited to primitive features such as URL Search, Keyword Search, Alphabetic Browsing and Full-Text Search [32].

Our research and development aims to overcome these problems of current web archiving through the exploitation of blog characteristics. However, while specialisation can solve existing problems, it causes additional challenges for the interoperability with other archiving and preservation facilities. For example, the vision of a seamless navigation and access of archived web content requires the support and application of accepted standards [28]. Therefore, our targeted solution aims on

- improving blog archiving through the exploitation of blog characteristics, and
- supporting the integration to existing archiving and preservation facilities.

The rest of this document is structured as follows: In Section 2, we evaluate prevalent web archiving software and initiatives in the context of blog preservation. In Section 3, we present the rationale and identified user requirements behind design decisions. In Section 4, we analyse the system architecture and the key components of the resulting system. In Section 5, we outline the procedure and findings of the platform evaluation before we reveal limitations of our solution in section 6. Finally, we draw conclusions and outline future work.

2 Related Work

In the following section, we review projects and initiatives related to blog preservation. Therefore, we inspect the existing solutions of the IIPC[3] for web archiving and the ArchivePress blog archiving project. Furthermore, we look into EC funded research projects such as Living Web Archives (LiWA)[4], SCalable Preservation Environments (SCAPE)[5] and Collect-All ARchives to COmmunity MEMories (ARCOMEM)[6] which are focusing on preserving dynamic content, web scale preservation activities and how to identify important web content that should be selected for preservation. Table 1 provides an overview of the related initiatives and projects we examine in this section.

Table 1 Overview of related initiatives and projects

Initiative	Description	Started
ArchivePress	Explore practical issues around the archiving of weblog content, focusing on blogs as records of institutional activity and corporate memory.	2009
ARCOMEM	Leverage the Wisdom of the Crowds for content appraisal, selection and preservation, in order to create and preserve archives that reflect collective memory and social content perception, and are, thus, closer to current and future users.	2011
IIPC projects	Web archiving tools for acquisition, curation, access and search.	1996
LiWA	Develop and demonstrate web archiving tools able to capture content from a wide variety of sources, to improve archive fidelity and authenticity and to ensure long term interpretability of web content.	2009
SCAPE	Developing an infrastructure and tools for scalable preservation actions	2011

The International Internet Preservation Consortium (IIPC)[7] is the leading international organization dedicated to improving the tools, standards and best practices of web archiving. The software they provide as open source comprises tools for acquisition (Heritix[8]), curation (Web Curator Tool[9] and

[3] `http://www.netpreserve.org/web-archiving/tools-and-software`

[4] `http://liwa-project.eu`

[5] `http://www.scape-project.eu`

[6] `http://www.arcomem.eu`

[7] `http://netpreserve.org/`

[8] `http://crawler.archive.org`; an open-source, extensible, Web-scale, archiving quality Web crawler

[9] `http://webcurator.sourceforge.net/`; a tool for managing the selective Webharvesting process

NetarchiveSuite[10]), access and finding (Wayback[11], NutchWAX[12], and WERA[13]). They are widely accepted and used by the majority of internet archive initiatives [11].

However, the IIPC tools cause several problems for blog preservation. First, the tools for acquisition and curation use a schedule based approach to determine the point in time when the content should be captured for archiving. Thus, the life of a website is not recorded appropriately if the page is updated more often than it is crawled [14]. Given that a lot of blogs are frequently updated, an approach which considers update events (e.g. new post, new comment, etc.) as triggers for crawling activities would be more suitable. Secondly, the archiving approach of IIPC considers the webpage as the digital object that is preserved and can be accessed. However, a blog consists of several smaller entities like posts and comments. Therefore, while archives like the Internet Archive enable an analysis of the structure of the blogosphere (how blogs are connected and change over time), a specialised archiving system based on the internal structure of blogs facilitates also further analysis like the dissemination of issues or events in posts or by authors [33].

The ArchivePress[14] project was an initial effort to attack the problem of blog archiving from a different perspective than traditional web crawlers. It is the only existing open source blog-specific archiving software according to our knowledge. ArchivePress utilises Extensible Markup Language (XML) feeds produced by blog platforms in order to achieve better archiving [25]. The scope of the project explicitly excludes the harvesting of the full browser rendering of blog contents (headers, sidebars, advertising and widgets), focusing solely on collecting the marked-up text of blog posts and blog comments (including embedded media). The approach was suggested by the observation that blog content is frequently consumed through automated syndication and aggregation in news reader applications, rather than by navigation of blog websites themselves. Chris Rusbridge, then Director of the Digital Curation Centre[15] at Edinburgh University, observed, with reason, that blogs represent an area where the content is primary and design secondary [26].

Contrary to the solutions of IIPC, ArchivePress utilises update information of the blogs to launch capturing activities instead of a predefined schedule. For this purpose, it is taking advantage of RSS feeds, a ubiquitous feature of blogs. Thus, blogs can be captured according to their activity level, and it is more likely that the whole lifecycle of the blog can be preserved.

[10] `https://sbforge.org/display/NAS/Releases+and+downloads`; a curator tool allowing librarians to define and control harvests of web material

[11] `http://archive-access.sourceforge.net/projects/wayback/`; a tool that allows users to see archived versions of web pages across time

[12] `http://archive-access.sourceforge.net/projects/nutch/`; a tool for indexing and searching Web archives

[13] `http://archive-access.sourceforge.net/projects/wera/`; a Web archive search and navigation application

[14] `http://archivepress.ulcc.ac.uk/`

[15] `http://www.dcc.ac.uk`

However, ArchivePress has also a strong limitation because it does not access the actual blog page but only its RSS feed. Thus, ArchivePress does not aggregate the complete blog content but only a portion of it which is published in RSS feeds because feeds potentially contain just partial content instead of the full text, and do not contain advertisements, formatting markup, or reader comments [9]. Even if blog preservation does not necessarily mean to preserve every aspect of a blog [4], and requires instead the identification of significant properties [29], the restriction to RSS feeds would prevent a successfull blog preservation in various cases. RSS references only recent blog posts and not older ones. What is more, static blog pages are not listed at all in RSS.

Certain EC funded digital preservation projects are also relevant to blog preservation in various ways. The LiWA project focuses on creating long term web archives, filtering out irrelevant content and trying to facilitate a wide variety of content. Its approach is valuable as it focuses on many aspects of web preservation but its drawback is that it provides specific components which are integrated with the IIPC tools and are not generic. On the other hand, the ARCOMEM project focuses mainly on social web driven content appraisal and selection, and intelligent content acquisition. Its aim is to detect important content regarding events and topics, in order to preserve it. Its approach is unique regarding content acquisition and could be valuable to detect important blog targets for preservation but it does not progress the state of the art regarding preservation, management and dissemination of archived content. Another relevant project is SCAPE, which is aiming to create scalable services for planning and execution of preservation strategies. SCAPE does not directly advance the state of the art with a new approaches to web preservation but aims at scaling only. Its outcome could assist in the deployment of web scale blog preservation systems.

Besides the presented initiatives, there is an entire software industry sector focused on commercial web archiving services. Representative examples include Hanzo Archives: Social Media Archiving[16], Archive-it: a web archiving service to harvest and preserve digital collections[17] and PageFreezer: social media and website archiving[18]. Due to the commercial nature of these services though, it is not possible to find much information on their preservation strategies and technologies. Furthermore, it is impossible to know how long these companies will support these services or even will be in business. Thus, we believe that they cannot be considered in our evaluation.

Based on our review of the state of the art in blog preservation, we have identified issues that hinder the appropriate preservation of blogs. In the following section, we present the user requirements and platform specifications we identified in order to overcome these difficulties and create a robust blog preservation platform.

[16] http://www.hanzoarchives.com/products/social_media_archiving
[17] http://www.archive-it.org/
[18] http://pagefreezer.com/

3 User Requirements

The requirements presented in this section, although identified in the context of BlogForever, can be applied to any blog preservation platform. Despite the fact that we identified issues in current web archiving, we have to further determine requirements to aggregate and preserve blog content correctly. Therefore, we examine domain specific user requirements which influence the design and development of the BlogForever platform.

The user requirements phase of the BlogForever platform[16] involved conducting 26 semi-structured interviews with representatives from six different stakeholder groups, identified as key to the project. These groups are researchers, libraries, blog authors and readers, businesses, and blog hosting services. The resulting requirements were first grouped into two main categories, functional and non-functional requirements. Furthermore, the non-functional requirements were divided into several subcategories included in table 2.

Table 2 Overview of user requirements

Requirements Category	Description	Number of Requirements
Data	Describe visible data and data the user needs to export or to process[24]	23
Interoperability requirements	Specify the ability of the system to share information and services[12]	7
(End-)user interface	Specify the interface between the software and the user[23].	35
Performance	Specify speed, performance, capacity, scalability, reliability and availability[23]	8
Operational	Specify required capabilities of the system to ensure a stable operation of the system[1]	6
Security	Specify the factors that protect the software from accidental or malicious access, use, modification, destruction, or disclosure	9
Legal	Cover any constraints through legal regulation, e.g. licenses, laws, patents, etc [10]	5
Preservation	Specify digital preservation specific requirements[10]	17

Due to the amount of identified requirements, it is not possible to explain them in detail. Therefore, we focus on important requirements regarding preservation, interoperability and performance because they specify the possible solutions for the aims of the research presented in this paper.

3.1 Preservation Requirements

BlogForever is utilizing the Open Archival Information System (OAIS) reference model [20] as a conceptual guidance for its weblog digital repository construction and management. Therefore, one of the main concerns while gath-

ering requirements was to specify four specific OAIS functions: Ingest, Data Management, Archival Storage and Access [19].

A blog preservation platform should be able to:

- Receive crawled blog content in the form of Submission Information Packages (SIPs). In BlogForever, the SIP will be a Metadata Encoding and Transmission Standard[19] (METS) wrapper which contains the original XML data as crawled by the spider, along with Machine-Readable Cataloging (MARC) and Metadata for Images in XML (MIX) metadata, and links to locally-attached files.
- Perform quality assurance checking to ensure that the transmission was successful and that the content is eligible for admission to the repository. This validation is necessary as the content source is not controlled by the administrators of the blog preservation platform and may cause issues. Virus and splog[20] detection should be performed.
- Generate Blog Archival Information Packages (AIPs) from SIPs to preserve the content.
- Produce blog Dissemination Information Packages (DIPs) upon user request. It should be possible to retrieve the original METS file from the Ingestion Database, enrich it with extracted information, and export it to the designated community.

3.2 Interoperability Requirements

Interoperability is one of the most crucial aspects of any digital preservation related system. BlogForever poses greater interoperability challenges due to the fact that the archive has to interoperate with multiple 3rd party systems which are hosting blogs to retrieve content in real time. In more detail, the following requirements are deemed necessary:

- Capturing has to be possible for various platforms. Blogs are available on different platforms. The archive should not be restricted to only one kind of platform or software because it would severely limit the amount of blogs that can be archived. Therefore, the spider and respectively the archive should be able to capture blogs ideally from every platform or at least from the most common platforms.
- Export blog elements. Expose parts of the archive via Open Archives Initiative - Protocol for Metadata Harvesting (OAI-PMH) based on specified criteria. The BlogForever platform should enable exposing different parts of the archive to different clients via OAI-PMH according to specified parameters and policy (e.g. client identity, admin settings).

[19] `http://www.loc.gov/standards/mets/`

[20] Spam blog, a blog with no real content but only used to promote associated websites and sell links and advertisements.

- Support Pingback/Trackback[21]. The archive should enable Pingback/Trackback to facilitate connections between the archived content and external resources. Acting as the author, for the sake of these methods, the archive can thereby show if external blogs are referencing archived content.
- Export links between blog content. The BlogForever platform should be able to provide users with a graph of links between blogs and blog posts archived in the system. The format of this graph is not restricted.

3.3 Performance Requirements

Performance is crucial for a system that needs to perform real time actions such as archiving. What is more, a platform that needs to be able to capture large amounts of blogs per day needs to have certain performance characteristics. These aspects of BlogForever clearly differentiate it from other archiving platforms where aggregation is performed in controlled, scheduled intervals. Real time capturing can produce a lot of data in peak times. Therefore, it should be possible to:

- Connect to multiple blogs in parallel,
- Retrieve and process blog content from multiple sources in parallel,
- Store data concurrently in the archive, and
- The archive should be scalable in terms of the size of data it stores, the number of blogs it monitors, the volume of content it retrieves and the number of its visitors.

The user requirements specified in this step of the development were considered for the architecture of the BlogForever platform, presented in the following section.

4 System Architecture

In this section, we first introduce the general concepts of our system architecture, consisting of two main components, comparing it with other current solutions and explaining how we address common issues in novel ways. Then we give a more detailed description of the design of each software component justifying our design decisions.

4.1 The BlogForever Software Platform

In the following, we give an overview of the general architecture of the BlogForever platform before we explain its two components - the blog spider and the blog repository - in detail. The BlogForever platform provides all the necessary

[21] Pingback and Trackback are both types of the Linkback method which enables authors to be notified when other authors link to, or refer to, their material.

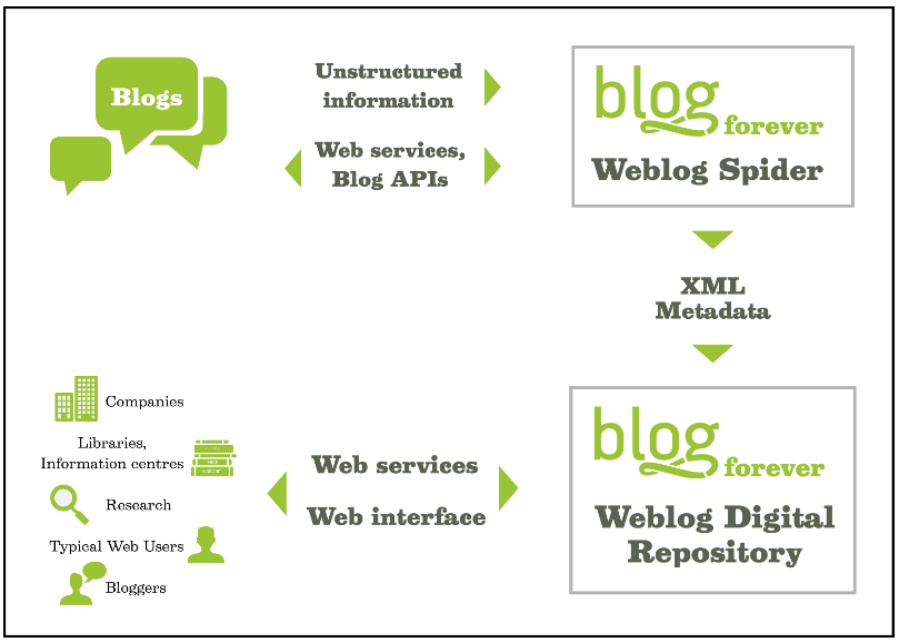

Fig. 1 A general overview of the BlogForver platform, featuring the blog spider and the blog repository

functionalities for collecting, storing, managing, preserving, and disseminating large volumes of information from blogs. Decoupling the two platform components allows the platform to be configured, run and managed in a very flexible and scalable way, as the spider and digital repository can be entirely independent while at the same time working together in harmony. A general overview of the BlogForever platform is illustrated in figure 1.

In a nutshell, given a predefined list of blogs, the spider analyses the blogs, manages the capturing process, harvests the required information in an event driven manner and hands them over to the repository together with additional metadata and in a standardised and widely accepted format. In turn the repository ingests, processes and stores the blog content and metadata, and facilitates their long-term preservation. Furthermore, the content is provided to end-users in various ways including sophisticated blog-specific functionalities.

The BlogForever platform is based on the well established model of harvesting, storing and presenting information from the world wide web. However, unlike other solutions, the BlogForever software treats blogs as web resources of a unique nature throughout their existence in the platform, from the moment of harvesting, to ingesting and processing and finally to presenting to the final user, while at the same time focusing in long term digital preservation as a key aspect of this process.

A selective harvesting process that captures only new content without missing relevant information is the main challenge for harvesting blogs. Therefore,

the BlogForever spider uses RSS and Atom feeds, ping servers, blog APIs as well as traditional HTML scraping to harvest blog content. As a result, real-time - or close to real-time[22] - archiving can be achieved to comply with the highly dynamic and volatile nature of blogs, expanding the traditional methods which use a schedule based approach to decide when new content should be harvested [14]. Furthermore, only the parts of a blog that contain new information are crawled instead of the whole page. Thus, the amount of crawled data can be reduced to the necessary data.

The initial crawl of a new blog harvests and archives the complete pre-existing blog content. This is achieved by crawling all existing blog pages to retrieve the location of past resources. Alternatively, blog APIs as well as RSS/Atom feeds can be used to recover historic content when enabled by the publisher. Once the location of past resources has been retrieved the platform uses its regular methods to harvest all pre-existing content. After the initial crawl, further crawls only consider new content as previously described.

A detailed data model has been developed based on existing conceptual models of blogs, data models of open source blogging systems, an empirical study of web feeds, and an online survey with blogger and blog reader perceptions [31]. While the full data model comprises twelve categories with over forty single entities, figure 2 shows just a high level view of the blog core as a Unified Modeling Language (UML) diagram. The primary identified entities of a weblog and the interrelation between them is shown and described by the connected lines. The small triangles indicate the directions of the relationships. Each entity has several properties, e.g. title, URI, aliases, etc. of a blog, which are not described here.

Using an exhaustive data model, which is unique for blogs, BlogForever does not limit itself to capturing online blog content as it is given to it. New material is analyzed and the different parts, as defined in the data model, are identified. All metadata are stored in a relational database on the repository component and are instantly available for discovery and exploitation. Furthermore, searching and navigation functionalities are available for the inherent entities of a blog (e.g. posts, comments), and, thus, enable sophisticated analysis of the blogosphere. Additionally, an extended range of personalized services, social and community tools are made available to the end-user.

BlogForever's design that features two fully independent and configurable components that work together in harmony, as described earlier, make it a highly flexible and scalable solution for private as well as institutional blog preservation.

[22] Taking into account the overhead incurred by the various network connections, and the latency incurred by the tools processing the information it is accurate to say that archiving is as close to real time as these factors allow.

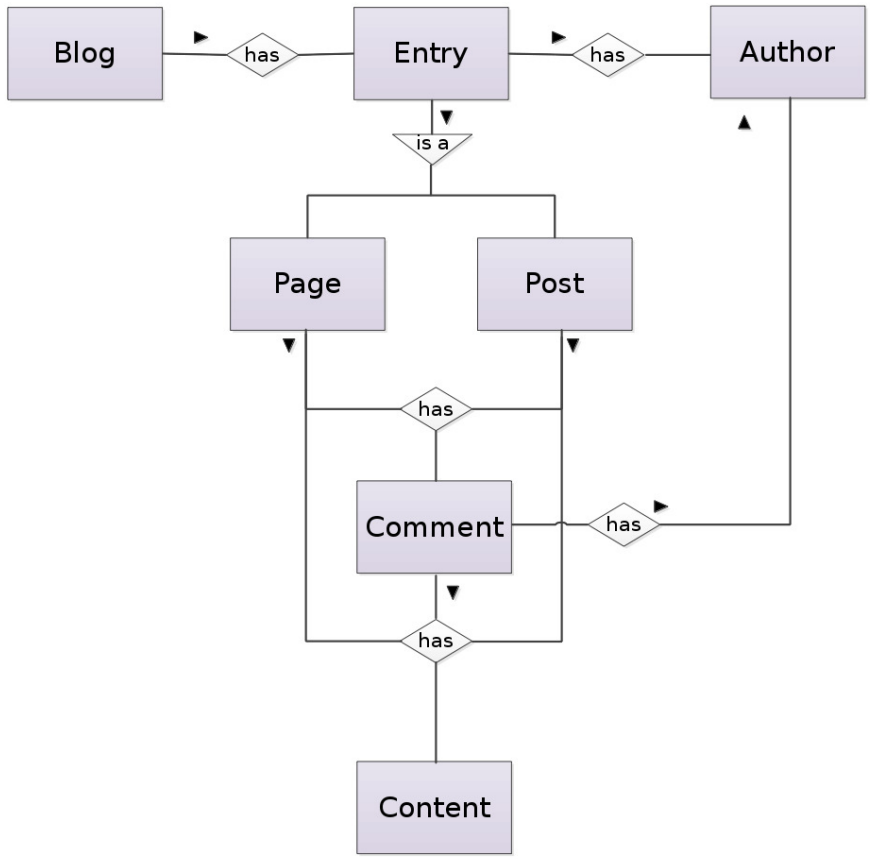

Fig. 2 Core entities of the BlogForever data model [31]

4.2 Blog Spider

The blog spider is the component of the BlogForever software platform that harvests a defined set of blogs. However, harvesting new blogs that are connected to the already archived blogs is also possible. It comprises of several subcomponents as shown in figure 3. The Inputter is the entry point, where the list of blog URLs is given to the spider. The Host Analyzer parses each blog URL, collects information about the blog host and discovers RSS feeds that the blog may provide. The Fetcher harvests the actual content of each blog. In the core of the spider, the Source Database handles all the blog URLs as well as information on the structure of the blogs and the filters used to parse their content. The Scheduler ensures that all the given blogs are checked when new content is available or at regular intervals. The Parser analyzes HTML content, semantically extracts all the useful information and encapsulates it into XML. Finally, the Exporter serves all the parsed content and also provides

an API for programmable access to it. In the next paragraphs the internal architecture of the spider will be made clearer.

Inputter: The Inputter's main function is to serve as the entry point to the spider, where the list of blogs to be monitored can be set. There are two ways to achieve that: either through a web interface that allows the manual insertion of new blogs, or automatically by fetching new blogs from a broader list hosted at the digital repository. The idea behind this design decision is that the administrator should have full control over the set of blogs he wants to have harvested. Currently, the automatic fetching of new blogs from the repository is achieved through the use of the Open Archives Initiative - Protocol for Metadata Harvesting[23] (OAI-PMH), which allows for the spider to pull information from the repository. However, mechanisms for the repository to push information on new blogs to the spider, such as the use of a web service, are also being explored. At the same time, the option to insert new blogs discovered through ping servers is also being implemented. The Inputter is also capable of identifying and accepting a set of options that affect the crawling of each blog. These options, which can be either set at the repository or directly at the spider, include fetching embedded objects of specific data types, producing and delivering snapshots of the content at harvest time etc.

Host Analyzer: All blog URLs collected by the Inputter have to pass through the Host Analyzer, which approves them or blacklists them as incorrect or inappropriate for harvesting. The Host Analyzer discovers and identifies RSS feeds for the given blog and, consequently, detects all content associated to it, such as blog posts, comments and other related pages. This is achieved through a process of analyzing the HTML code and using statistical classification techniques to detect and identify URLs. As a result of these techniques, the Host Analyzer also detects the blog authoring software platform used and/or the blog host as well as the link extraction patterns and filters that are used for future harvesting. Finally, this is the first stage where a blog may be flagged as potential spam, given basic information such as its URL.

Source Database: At the core of the spider's system we find the system manager, which comprises of the Source Database, that contains all the monitored blogs, and the Scheduler, that notifies the Fetcher when new content is available for each blog or otherwise at regular intervals. The Source Database essentially holds the internal catalog of all the blog sources that the Spider harvests. For each blog, information on blog post URLs, how blog comments relate to their parent blog posts, various metadata as well as filtering rules and extraction patterns are stored. Additionally, information related to how the blogs were inserted and processed by the spider is saved. It should be noted here that blogs which had previously been blacklisted as incorrect or inappropriate are also kept in the Source Database.

Scheduler: The Scheduler ensures that new content is captured when available based on update events, extending the traditional method of capturing content at regular intervals. These events can either be notifications received

[23] `http://www.openarchives.org/pmh/`

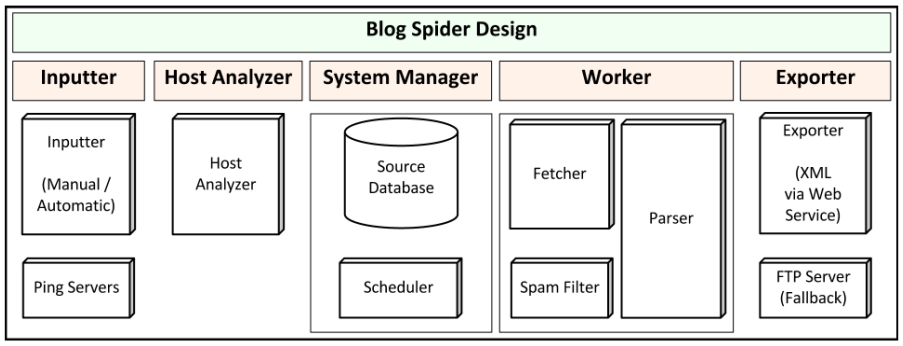

Fig. 3 The outline of the blog spider component design

from registered ping servers or updates in systematically retrieved RSS feeds. The Scheduler schedules the feed polling frequency depending on how active the it is (adaptive rate control), and uses protocols such as pubsubhubbub[24] when available to instead get notified when the feed is updated and limit the use of server resources. At the same time, the Source Database feeds the Scheduler with information on which blogs should be checked for updates manually at regular intervals when update events are not provided as previously described. Therefore, unless a ping server delivers updates automatically or an RSS feed provides such information when checked systematically for updates, the Scheduler makes sure that all blogs are checked at a regular basis for new material, including new posts as well as new comments. When required, the manual update frequency can be automatically deduced and updated by the historically measured frequency of new material on the given blog. Otherwise it may be configured by the administrator, defaulting to several times a day depending on the server load.

Fetcher: All actual fetching and parsing of content takes place at the Worker unit of the spider, which is divided to the subcomponents of the Fetcher and the Parser. The Fetcher's main operation is to capture and analyze RSS feeds and HTML content. This subcomponent's services are used in two distinct places in the spider's design: during the Host Analyzer's detection process, as seen before, and most notably during the actual downloading and analyzing of the blog content. For its first use, the Fetcher uses heuristic techniques, such as the ID3 decision tree [27], to create and use rules that extract useful links within a blog. These links usually are RSS feeds, links to posts, comments and other related information. For its second and main use, the Fetcher analyzes the full captured content to identify distinct blog elements. To this end, platform-specific rules as well as heuristic rules are deployed to discover and define a post's full text, attached material, related comments, author and date information etc. This process includes comparing content extracted from the RSS feed with content extracted from HTML sources, using

[24] `https://code.google.com/p/pubsubhubbub/`

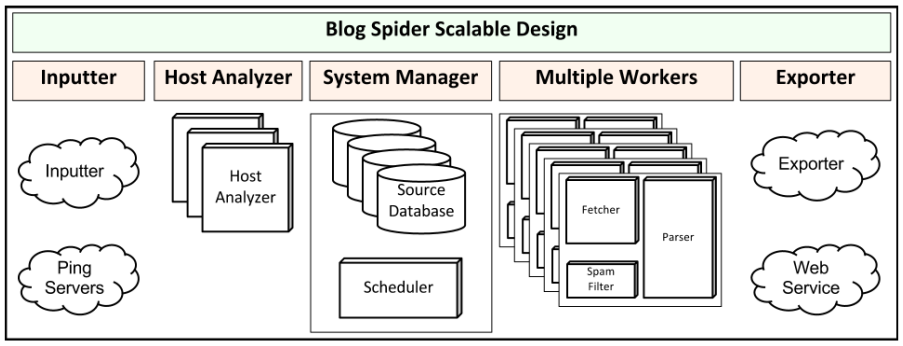

Fig. 4 High level outline of a scalable set up of the blog spider component

the Levenshtein distance string metric, to ensure the integrity of the final results [30].

Parser: Once the detailed content has been extracted, the Parser is put in use to perform an even more exhaustive discovery of information and metadata as well as encapsulate everything into a given XML format and send it to the Exporter. This is achieved through a series of additional checks which include: detecting further links to related content that had not been previously checked, detecting additional embedded material such as pictures, videos, blog rolls, social networks widgets etc. and discovering microformats and microdata semantic markup. Another function of the Parser is to encapsulate all the collected data and metadata into an exportable XML file of a given structure. In the current implementation we have chosen the METS standard as the default container for this information. More information on this design decision will be provided at the next section. Finally, in order to compliment the first level of spam detection and filtering performed by the Host Analyzer an additional dedicated Spam Filter is being developed as part of the Parser. Filtering will be performed at the level of the content through the detection of common spam patterns.

Exporter: At the end of the processing chain of the Spider we find the Exporter subcomponent which makes the parsed content available to the Spider's clients. This is achieved through a web service that supports an interoperable machine-to-machine interaction between the Spider and the potential client. The Spider provides detailed description of all the offered functionality through a machine-readable Web Services Description Language[25] (WSDL) description file. Clients can use Simple Object Access Protocol[26] (SOAP) messages to search for newly updated material and download it on demand. As a fallback option for technologically simpler clients, the Exporter is able to make the XML files produced by the Parser temporarily available through an File Transfer Protocol (FTP) server.

[25] `http://www.w3.org/TR/wsdl`
[26] `http://www.w3.org/TR/soap12-part1/`

Scalability: In the previous paragraphs we described a Blog Spider software design that can support crawling a limited amount of blogs. In order to be able to monitor and harvest large amounts of blog information a multiple server architecture is needed. At the core of the Spider the high load of the Source Database can be distributed over a number of dedicated database servers. Another congestion point can appear when fetching and parsing blog content. Thanks to the smart Worker unit design described previously, an arbitrary number of Workers can be deployed to work together so that the various Fetcher and Parser subcomponents can work in parallel on harvesting and analyzing resources from different sources. A schematic representation of the above can be seen in figure 4.

4.3 Digital Repository

The digital repository is the component of the BlogForever software platform that facilitates the ingestion, management and dissemination of blog content and information. Once the blog spider has finished processing new blog content, the repository programmatically imports this semi-structured information through its ingestion mechanism. New blogs to be harvested can be submitted to the repository and approved by the administrators. Within the repository, blog data and metadata is stored, checked, analyzed and converted in order to fit the repository's internal management needs in the most efficient way possible. Long term digital preservation is a key aspect of this process and is therefore reflected in many of the actions performed. Dedicated indexing and ranking services are continuously executed to ensure the most accurate search results for the end users through the web interface. Searching is the main functionality of the web interface of the repository, part of a series of services offered to the administrator and end user, such as community and collaborative tools and value-added services. Finally, interoperability is an important aspect of the repository, which supports many widely used open standards, protocols and export formats. A graphical summary of the above can be seen in figure 5.

The blog repository is based on the Invenio[27] software suite. Invenio is a free and open source digital library software suite. The technology offered by the software covers all aspects of digital library management from document ingestion through classification, indexing, and curation to dissemination. Invenio complies with standards such as the Open Archives Initiative metadata harvesting protocol (OAI-PMH) and uses MARC[28] as its underlying bibliographic format. The flexibility and performance of Invenio make it a comprehensive solution for management of document repositories of moderate to large sizes (several millions of records). Invenio has been originally developed at CERN to run the CERN document server, managing over 1,000,000 bibliographic records in high-energy physics since 2002, covering articles, books, journals,

[27] `http://invenio-software.org`
[28] `http://www.loc.gov/marc/`

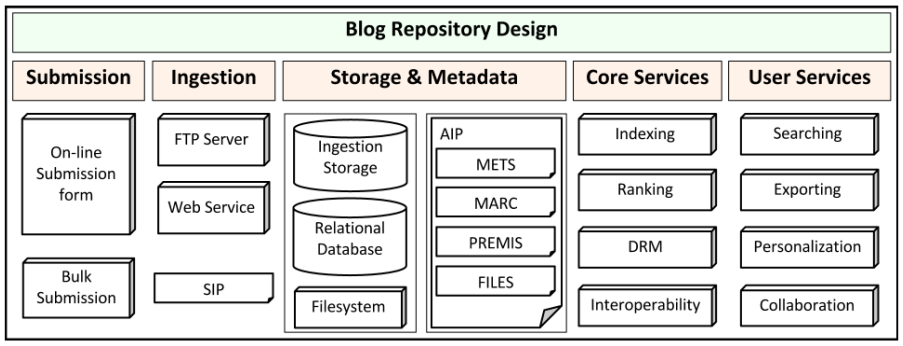

Fig. 5 The outline of the blog repository component design

photos, videos and more. Invenio is being co-developed by an international collaboration comprising institutes such as CERN[29], DESY[30], EPFL[31], FNAL[32], SLAC[33] and is being used by about thirty scientific institutions worldwide. As it is outside the scope of this paper, basic Invenio concepts will not be discussed here.

Metadata: One of the core design decisions of the BlogForever digital repository was the internal representation of metadata, as well as at the communication level with the Spider. Among the approaches relevant for transferring data across software applications and networks is the Extensible Markup Language[34] (XML). XML is a widely adopted machine/human-readable markup language. Its simplicity and suitability for transactions over the network spawned a large number of XML-based languages and standards. Among those standards are METS and MARCXML[35]. Both of the standards have been developed by the Library of Congress[36] and are widely adopted for representing or describing data.

METS categorizes metadata into two high level sections: a section for descriptive metadata (dmdSec) and a section for administrative metadata (amdSec). The administrative metadata section, in turn, is divided into further subcategories: a subsection related to provenance metadata (digiProv) and a subsection related to technical specifications of embedded content (techMD). In BlogForever all metadata for record will be wrapped, encoded, and exposed using METS. The policies on how these are wrapped within METS was described thoroughly in [19].

[29] `http://www.cern.ch/`
[30] `http://www.desy.de/`
[31] `http://www.epfl.ch/`
[32] `http://www.fnal.gov/`
[33] `http://www.slac.stanford.edu/`
[34] `http://www.w3.org/XML/`
[35] `http://www.loc.gov/standards/marcxml/`
[36] `http://www.loc.gov/`

In addition to the METS object associated to each record, it is essential that a METS profile describing the purpose of the repository and the types of objects and formats supported within the repository is made available at a fixed URI within the repository. The principles governing the management of the information ingested into the BlogForever repository are proposed in the METS profile included in [19], along with an example. In short, the profile stipulates:

- The controlled vocabularies and syntaxes to be used for metadata field values (e.g. ISO standards for describing time, date, geographic locations).
- The set of external metadata schemas that are being incorporated into the METS standard (e.g. MARCXML, MIX etc), and how they will be included within the native METS syntax.
- Any tools that might be employed for object characterization.
- The description and structural rules for employment in writing METS objects.

The BlogForever METS profile should be considered a work in progress, to be adapted if necessary as the implementation of the repository continues. The final profile should be made available at a fixed public URI within the repository and also submitted for inclusion in a public registry such as the Library of Congress, so that future projects involved in blog preservation activities may profit from our research.

MARCXML is the main descriptive metadata standard for BlogForever. The fact that the MARC format is the chosen internal representation of metadata in Invenio played an important role in our decision. MARC is well-established, flexible enough to guarantee long-term reliability and can be thoroughly extended to adapt to any metadata structure. For the context of blogs we have defined a draft mapping between the conceptual data model from [31] and MARC fields. For this reason we have redefined a series of MARC fields in the 9xx field range[37]. Other suggested standards for specific object types are: MIX for images, TextMD[38] for structured text, AES57-2011[39] for audio and MPEG/7[40] for moving images.

BlogForever's arsenal of standards is completed with the Preservation Metadata: Implementation Strategy (PREMIS) standard. PREMIS consists of a core set of standardized data elements that are recommended for repositories to manage and perform the preservation function. These crucial functions include actions to make the digital objects usable over time, keeping them viable, or readable, displayable and kept intact, all for the purpose of future access. Additionally, a PREMIS schema can be wrapped up in the METS profile. The detailed reasons and analysis for choosing these standards are described in [19].

[37] http://www.loc.gov/marc/bibliographic/bd9xx.html
[38] http://www.loc.gov/standards/textMD/
[39] http://www.loc.gov/standards/amdvmd/audiovideoMDschemas.html
[40] http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm

Submission: As mentioned previously, one of the design decisions was that the users cannot interfere with the list of target blogs. Therefore, the administrators of the platform can provide the option to groups of registered or all users to submit new blogs they think should be archived in the repository, which can then be approved or rejected by the administrators. The option of bulk submissions by the administrators is available too, through the use command line tools. Invenio already features an elaborate submission management system that allows for fully customizable forms to be created. Based on that system, the BlogForever repository offers a simple yet complete submission form that allows the user to suggest a new blog using its URL with the following optional information:

- Topic - a selection of existing topics is offered, based on previous user input, or the user can name their own topic.
- License - in order to respect ownership copyright and potential local rules and laws that may apply, the repository offers a knowledge of licenses the user can select from. More information on how licenses are enforced will be given in the following paragraphs.

Ingestion: The spider's Exporter subcomponent, which makes the parsed content available to the spider's clients, was previously presented and the technologies used to make data and metadata available were described. On the repository side, a well defined client is employed, able to programmatically access the web service, send requests using SOAP messages and interpret the responses. In case an FTP server is needed to export blog content, the repository may also use a dedicated FTP client to connect and download all new content. In both scenarios, the repository is periodically pulling information from the spider. A two-way communication is currently being designed to allow the spider to push new information to the repository and ensure real time archiving when that is desired.

Once new content is received by the repository in the form of individual packages (set of resources) they can can be considered as SIPs according to the Open Archival Information System (OAIS) specification. To ensure their validity, quality assurance checks may be performed on the SIP such as the verification of their MD5 checksum. A dedicated ingestion storage engine has been introduced in the repository to permanently store all SIPs. At the end of the ingestion process an AIP is generated from a single SIP by extracting descriptive information like descriptive metadata for search and retrieval, attached files, etc. The AIP is then finally transferred to the internal storage.

Storage: All data and metadata in BlogForever are stored in databases as well as in the filesystem. Multiple copies of each resource are kept in the form of replication as well as frequent incremental backups. To ensure the long term digital preservation of these resources, versioning is also introduced in various stages of the ingestion and archival process. Any possible change on any resource creates a new version of that resource and keeps all old versions. For example, nothing can be deleted unless the system administrator literally does so directly on the server.

As mentioned before, all SIPs are permanently stored in a dedicated ingestion storage engine. Any database server can be configured to act as the ingestion storage engine. In the current implementation MongoDB[41] has been chosen as a highly scalable and agile document database server[42]. When the AIPs are generated from the SIPs, descriptive metadata (in MARCXML) are extracted and stored separately in a database to provide fast search and retrieval services. In the current implementation the well established open source relational database management system (RDBMS) MySQL[43] is used. During the next stages of development an object-relational mapping software solution will be deployed to provide for easy integration of other popular RDBMSs such as PostgreSQL[44] and Oracle[45].

All attached files are managed by a dedicated software module which acts as an abstraction layer between the physical storage on the filesystem and the logical names of those files as part of the AIP. As for the metadata, versioning and replication are also provided for all physical files. BlogForever takes care of the internal connections between all the data and metadata that amount to an AIP in a seamless way, using various identifiers.

Within its role as an archive, BlogForever ingests and stores all content as it is sent from the spider. Upon reception all data are treated equally. Content that may potentially later be declared inappropriate at its original source can be flagged as such in the platform either automatically by following spider updates or manually at the repository by the administrators.

Indexing & Ranking: BlogForever features specially designed indexes to provide high speed searching. Indexes can be set up on any logical field of the descriptive metadata after associating it to one of the MARC tags. Indexing may also be performed on all text files such as PDF documents. This leads to a combined metadata and fulltext high speed search. Fuzzy indexing is also available through multi-lingual stemming. On top of indexing, search results may be sorted and ranked according to a series of criteria such as the classical word-frequency based vector model that allows to retrieve similar documents, the number of views and downloads etc.

Searching: The most important user feature of the software platform is provided through a simple yet intuitive interface. A single text field offers access to all the digital repository's resources. Behind the scenes, a powerful search engine makes sure accurate results are returned as quickly as possible. The user may search across all metadata and data on the repository or can focus their search on specific logical fields such as author or title, once defined by the administrator, as is for indexing. The option to use regular expressions is also available for advanced users. Finally, all documents on the BlogForever repository can be organized in collections, based on any descriptive metadata

[41] `http://www.mongodb.org/`
[42] `http://www.mongodb.org/display/DOCS/Use+Cases`
[43] `http://www.mysql.com/`
[44] `http://www.postgresql.org/`
[45] `http://www.oracle.com/products/database/`

the administrator choses. This extra granularity can help the user browse the contents or limit their search in specific collections.

Interoperability: In order to work together with diverse systems and ensure wider acceptance, interoperability is a key aspect of the BlogForever software platform. Various standards and common technologies are supported. Most notably:

- the Open Archives Initiative[46] (OAI) protocols can be used for the acquisition and dissemination of resources
- the OpenURL standardized format can be used to locate resources in the repository
- the Search/Retrieval via URL[47] (SRU) protocol is supported
- Digital Object Identifiers (DOI) can be used to uniquely identify resources in the repository

Exporting: To further enhance interoperability, BlogForever offers the possibility to export metadata in various standard formats not native to the platform, such as Dublin Core[48] and MODS[49]. A powerful conversion module quickly provides the various export formats on demand.

Personalization: BlogForever proposes a number of personalization and collaborative features. Like the repository-wide search collections, users may create their personal collections of documents (baskets) to use privately or share with other users. Periodical notifications on newly added documents (alerts) are also available and can also be used in conjunction with the aforementioned baskets. Alternatively, an RSS feed on any search query is available to be used with any given reader. Finally, commenting and rating on all repository content is offered.

Digital Rights Management (DRM): Basic management of digital rights in the BlogForever software platform is achieved through the configuration of a standard Role-Based Access Control (RBAC) system, which controls access to all repository's resources and allows for complicated access rules to be set. Full DRM support is currently being designed based on the research and analysis of common copyright issues in the blogosphere.

Scalability: The digital repository component of the BlogForever software platform is based on Invenio, which in turn, is based on widely used open source software projects such as the MySQL relational database management system and the Apache HTTP server. High availability and scalability are key features for MySQL with several success stories[50] to date and a complete guide[51] for developers including replication and clustering techniques. Apache also uses many well proven internal and external techniques to scale, including

[46] `http://www.openarchives.org/`
[47] `http://www.loc.gov/standards/sru/`
[48] `http://dublincore.org/`
[49] `http://www.loc.gov/standards/mods/`
[50] `http://www.mysql.com/why-mysql/scaleout/`
[51] `http://dev.mysql.com/doc/mysql-ha-scalability/en/ha-overview.html`

load balancing and caching. Invenio can run in multi-node load-balanced architecture with several front-end worker nodes and with database replication nodes in the back end[52]. Furthermore, the indexing and searching capabilities of Invenio can already be augmented by plugging in a scalable search server such as Apache Solr[53].

5 Evaluation

In this section, we reveal the procedure and current findings of the evaluation of the platform. Aim of the evaluation is the assessment of the overall platform quality and addresses therefore more aspects than the research problems highlighted in this paper.

We conduct the evaluation as a mixed method research which is defined as "the class of research where the researcher mixes or combines quantitative and qualitative research techniques, methods, approaches, concepts or language into a single study." [15, p. 17]. We follow the pragmatic view of choosing the method which is most helpful to examine the specific platform aspect instead of adhering to traditional philosophical paradigms (e.g. pure positivists or constructivists). Thus, we are able to combine in the evaluation process exploratory aspects (e.g. how can the usability of features be improved?) with confirmatory aspects (e.g. does the platform provide the intended improvement of web archiving?).

The evaluation is conducted in six case studies, as presented in Table 3. The case studies increase in their complexity and, thus, in the requirements of the software. Furthermore, the case studies are conducted in coordination with the implementation process in order to integrate the case studies' findings further improvements of the software. Thus, early case studies evaluate less functionalities than later and more satisfactory results are expected in later phases of the evaluation.

In the following, we present results for the first four out of six case studies because the remaining two are still running at the time of this paper.

Overall, the case studies are intended to test the BlogForever platform extensively, generate feedback from users, and minimize system problems. The reason behind this extensive software testing is to follow any activities aimed at evaluating a particular attribute or capability of the BlogForever platform, and determining that its functionality is sufficient for user requirements, that it meets its required results, and it is judged to be fit for purpose. The purpose of this testing is quality assurance, verification and validation. We take the view that software testing is more than just debugging. As Hetzel explains, with software testing we can test related factors to make quality visible [13].

This challenge is tackled with an evaluation plan which includes the following steps and will be revealed in the following paragraphs.

[52] `https://indico.cern.ch/getFile.py/access?contribId=17\&sessionId=9\&resId=0\&materialId=slides\&confId=183318`

[53] `http://lucene.apache.org/solr/`

Table 3 Overview of the planned BlogForever Case Studies [3]

ID	Nature	Blog Count	Domain & Content	Work Period
1	Small & simple	58	Higher Further Education UK	Jul - Nov 2012
2	Small & simple	70	Higher Further Education UK	Jul - Nov 2012
3	Small & complex	356	Higher Multilingual focused	Oct 2012 - Mar 2013
4	Small & complex	1000	Higher Multimedia focused	Jan - Apr 2013
5	Large & complex	2000	Wide range of topics	May - Jul 2013
6	Large & complex	500,000	Wide range of topics	May - Jul 2013

1. An initial set of Research Questions (RQs),
2. The design and implementation of six case studies of increasing size and complexity,
3. The definition of ten Themes which help rationalise the outputs of all evaluation reports and connect them with the Research Questions,
4. The data collection via internal and external testing,
5. The evaluation and conclusions.

The original BlogForever evaluation plan defined a set of general Research Questions to guide the research team members and keep them focused on the evidence and data needed to validate the BlogForever platform features impact. In this work, we are focusing particularly on the following specific Research Questions which are different from the original ones:

- RQ1: Are we improving blog archiving through the exploitation of blog characteristics?
- RQ2: Are we supporting the integration to existing archiving and preservation facilities?

According to the BlogForever evaluation plan, a set of Themes were devised in order to assist in rationalising the outputs of the evaluation and help reach substancial conclusions.

- T1: Using blog records,
- T2: System integrity,
- T3: Sharing and interaction,
- T4: Searching,
- T5: Access,
- T6: Data integrity,
- T7: Preservation,
- T8: Functionality,
- T9: System navigation,
- T10: System terminology.

RQ1 relates directly to T1: Using blog records and indirectly to all Themes as the mission of BlogForever is to support blog archiving and every aspect of the system described in the Themes is geared towards this purpose.

RQ2 relates directly to T3: Sharing and interaction. Moreover, RQ2 is indirectly related to T2: System integrity, T5: Access, T6: Data integrity and T7: Preservation, as these features are necessary prerequisites to archiving and interoperability with other systems.

After defining the necessary research tools, internal testing evaluated all implemented features from the partners point of view. An indicative testing record is presented in Table 4. Due to space restrictions, it is not possible to present all 89 internal testing records but only their scores summary in Table 5. For the score column, observers at the internal testing stage scored results comparing expected description of the features as described in the design of the platform versus actual outputs using a scale of 1 to 5, where 1 equates to "Did not work as expected" and 5 equates to "Worked better than expected".

Table 4 Internal testing record example [3]

Case Study	CS1
Aspect	Differentiation
Feature	RF4: Bibformat output templates to display blogs and blog posts differently.
Version	BF1
Observation	Template works effectively but lacks contextual detail.
Suggestions for improvement	Post should have the name of the main blog embedded within it; otherwise there is no context about the post other than the title of the blog post and authors name.
Further actions	The information is there, but probably needed to be more prominent. This has been solved in BF5.
Score	3

Table 5 Internal testing scores summary [5]

Themes	Score
T1: Using blog records	3.4
T2: System integrity	4.25
T3: Sharing and interaction	3
T4: Searching	3.75
T5: Access	3.28
T6: Data integrity	3.89
T7: Preservation	2.5
T8: Functionality	4.11
Average	3.66

External Testing evaluated the BlogForever platform from an outside user point of view. Unlike internal tests, all the external tests have been performed by independent users with experience related to preservation and archiving

but with diverse levels of expertise and specialisation. An example user questionnaire is presented in Table 6. Due to space restrictions, it is not possible to present all 24 external testing records but only their scores summary in Table 7.

Table 6 External user questionnaire parts B and C from Case Study 2

Question
Complete the registration process.
Randomly select a blog.
Visit the blog itself.
Try to find it in BF repository on their own.
Get to the detailed record of a blog.
Show citation description within a blog.
Please elaborate on how well or badly you feel you performed the exercises/solve the tasks set for you?
What aspects of the system supported you to perform the exercises or solve the tasks set for you today?
What aspects of the system made it difficult for you to perform the exercises or solve the tasks?
How could the system be improved?

Table 7 External testing scores summary [5]

Themes	Score
T1: Using blog records	3.61
T2: System integrity	3.61
T3: Sharing and interaction	3.94
T4: Searching	3.54
T5: Access	3.75
T6: Data integrity	3.53
T7: Preservation	3.67
T8: Functionality	3.59
T9: System navigation	3.62
T10: System terminology	3.54
Average	3.64

Some interesting observations on the final scores can be summarised as follows:

- The average score in internal and external testing is almost equal, 3.64 and 3.66. This score is rather favourable as 3 equates to "Most areas worked as expected" and 4 equates to "All work as expected".
- On average, the scores of all external testing scores are very consistent, with a minimum of 3.53 in T6: Data integrity and a maximum of 3.94 in T3: Sharing and interaction (range 1 to 5). On the other hand, internal testing scores vary more, with a minimum of 2.5 in T7: Preservation and a maximum of 4.11 in T8: Functionality.
- There seems to be consensus on the scoring of most themes, except T3: Sharing and interaction and T7: Preservation where the different between

the external and internal testing is around 1. In all other cases, the differences are much smaller, strengthening the outcomes of the evaluation.

To conclude, we consider the rating of all Themes to average out between 3 ("Most areas worked as expected") and 4 ("All work as expected"). Any deviations between the different themes, evaluation methods and case studies are not significant. From these scores, we can conclude that the majority of users are satisfied with the performance of the BlogForever platform. The trend of our success in WP5 indicates that any outstanding issues embodied in the Research Questions will be addressed by future project work.

In general, we may with good confidence assert that the BlogForever Case Studies show the credibility of the BlogForever approach for a blog aggregation, preservation, management and dissemination platform. Detailed reports on all internal and external testing are presented in BlogForever D5.2 Implementation of Case Studies [3]

6 Limitations

While our solution overcomes the identified problems of current web archiving, some limitations can be identified and are revealed in this section.

The first limitation concerns the definition of the target blogs to be preserved. As we described earlier in 4.1, the BlogForever spider is provided with a predefined list of blogs, which then analyses and harvests. The problem lies in the fact that the list of target blogs has to be predefined explicitly. This means that the administrators need to already have the list of blogs, which may not be always the case. When BlogForever is deployed to preserve the blogs of a specific organization, the list of target blogs is predefined but when BlogForever is deployed in a different context, e.g. to create a repository of major Mathematics blogs, then the definition of target blogs is a big issue. The solution to this issue would be to have a mechanism able to generate and curate such a list in a semi-automatic or fully automatic way, based on configurable topic hierarchies. This way, the administrator would define the topics of interest and let the platform handle the specifics of blog collection management.

The Blog Spider has also limitations regarding new blog content detection and processing. First of all, it uses RSS and ping servers to receive indications that blog content has been updated. Nevertheless, these methods cannot provide indications for layout changes. Moreover, the provision of RSS is applied in different ways. While some blogs provide separate RSS feeds for posts and comments, others provide RSS feeds just for posts. Thus, the detection of new comments in real-time is problematic in such cases. We also face issues during the processing of unknown blog platforms or "'exotic implementations"' because the identification/analysis process for the entities of blogs is a knowledge-intensive process and has to be adapted to new developments in blog platforms. Therefore, the amount of necessary adaptations is dependent on the actual domain of blog platforms that should be archived. In addition to

this, while the identification of structural blog entities like posts, comments, etc. is achieved, the validation of whether the author uses a real name or an alias can not be applied automatically. To sum up, there are some difficulties to evaluate the validity of certain elements.

Another limitation we identified is relevant to the scalability of the BlogForever platform. Scalability is the ability of a system, network, or process to handle a growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth [6]. As we described in 4.3, the BlogForever repository component is based on the Invenio software suite which has been in production since 2002 and many popular repositories hosting millions of records such as CERN Document Server[54] and INSPIRE [55] use it with great success. Nevertheless, due to its reliance on MySQL RDBMS, the BlogForever repository architecture is inherently not scalable to web scale datasets. New database technologies such as NoSQL databases are better fit for this purpose [8]. Thus, it would not be possible to deploy BlogForever in a large cluster of servers in order to create internet scale blog archives.

In summary, we identified several limitations that should be addressed in further research and development. However, they do not refute the claim that the BlogForever system solves the identified problems of current web archiving. Therefore, we proceed with a conclusion in the following section and give additional indications for further research.

7 Conclusions and future work

The paramount importance of the blogosphere has turned blog preservation into a pressing issue. However, in this work we presented that the current web archiving tools and methods are ineffective and inconsistent in order to preserve blogs correctly. We identified several problems which can be summarized as a) blog aggregation scheduling inefficiencies, b) reduced blog data extraction performance due to suboptimal web crawling methods and c) inabilities to exploit the potential of the inherent structure and modern technologies of blogs.

To resolve the identified issues which have a great impact on the current state of blog preservation, we first outlined requirements for a blog preservation system in the context of our current work in the BlogForever project. One observation is that the requirements that we defined are not constrained into our platform. On the contrary, they can be taken into consideration when designing and developing any blog preservation platform. What is more, we presented BlogForever platform as a novel blog preservation platform featuring robust digital preservation, management and dissemination facilities for blogs. The BlogForever platform encompasses a set of new solutions to address the identified problems. Most notably these are: a) a novel blog data model, b) a blog spider implementing selective harvesting processes which optimize the

[54] `http://cds.cern.ch/`

[55] `http://inspirehep.net/`

procedure scheduling and performance, c) a digital repository which facilitates the ingestion, management and dissemination of blog content.

Finally, our work could be expanded in several ways, that we are planning to explore in future work. First of all, we can improve on our limitation to handle the target blogs for preservation in a more automated way. As we described in the previous section, the target blogs are a relatively 'static' list defined by the administrators of the system. An automated way of manipulating the target blog list by defining topics and rules would be welcome, especially for larger and more complex blog repositories.

Second, the Blog Spider could be developed considerably in order to alleviate new content detection and processing issues. A new way to detect layout changes in blogs would be a valuable addition to existing update detection mechanisms. What is more, blog entities analysis and detection could be optimised to cope with a wider range of possible instances.

Third, the BlogForever platform could become considerably more scalable by switching the database component from a relational database system (MySQL) to a more scalable database architecture such as NoSQL. As we described in the previous section, this is an important limitation in order to deploy web scale blog repositories using the BlogForever platform.

Fourth, we can expand the targets of the BlogForever platform to include micro-blogs. Microblog differ from traditional blogs in that their contents are typically smaller in both actual and aggregate file size [18]. Therefore, it will be possible to preserve microblogs as well without altering significantly the requirements, architecture and implementation of the BlogForever platform.

Acknowledgements Special thanks to all the BlogForever project partners and especially Yunhyong Kim and Ed Pinsent for their efforts. The research leading to these results has received funding from the European Commission Framework Programme 7 (FP7), BlogForever project, grant agreement No.269963.

References

1. IEEE recommended practice for software requirements specifications. IEEE Std 830–1998 (1998)
2. Ainsworth, S.G., Alsum, A., SalahEldeen, H., Weigle, M.C., Nelson, M.L.: How much of the web is archived? In: Proceeding of the 11th annual international ACM/IEEE joint conference, p. 133. ACM Press, New York, New York, USA (2011)
3. Arango, S., Pinsent, E., Sleeman, P., Gkotsis, G., Stepanyan, K., Rynning, M., Kopidaki, S.: Blogforever: D5.2 implementation of case studies. Tech. rep. (2013)
4. Ashley, K., Davis, R., Guy, M., Kelly, B., Pinsent, E., Farrell, S.: A guide to web preservation. Tech. rep., Joint Information Systems Committee (JISC) (2010)
5. Banos, V., Arango-Docio, S., Pinsent, E., Sleeman, P.: Blogforever: D5.5 case studies comparative analysis and conclusions. Tech. rep. (2013)
6. Bondi, A.B.: Characteristics of scalability and their impact on performance. In: Proceedings of the 2nd international workshop on Software and performance, pp. 195–203. ACM (2000)
7. Chittenden, T.: Digital dressing up: Modelling female teen identity in the discursive spaces of the fashion blogosphere. Journal of Youth Studies **13**(4), 505–520 (2010)

8. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with YCSB. In: Proceedings of the 1st ACM symposium on Cloud computing, pp. 143–154. ACM (2010)
9. Elsas, J.L., Arguello, J., Callan, J., Carbonell, J.G.: Retrieval and feedback models for blog feed search. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '08, pp. 347–354. ACM, New York, NY, USA (2008)
10. Fitzner, D.: Requirements specification of the TELEIOS user community. Tech. rep., Fraunhofer IGD (2010)
11. Gomes, D., Miranda, J., Costa, M.: A survey on web archiving initiatives. In: S. Gradmann, F. Borri, C. Meghini, H. Schuldt (eds.) Research and Advanced Technology for Digital Libraries, *Lecture Notes in Computer Science*, vol. 6966, pp. 408–420. Springer Berlin / Heidelberg (2011)
12. Group, T.O.: Interoperability requirements. URL `http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap29.html`
13. Hetzel, W.C.: The complete guide to software testing. QED Information Sciences, Wellesley, Mass. (1988)
14. Hockx-Yu, H.: The past issue of the Web. In: Proceedings of the ACM WebSci'11. Koblenz, Germany (2011)
15. Johnson, R.B., Onwuegbuzie, A.J.: Mixed Methods Research: A Research Paradigm Whose Time Has Come. Educational Researcher **33**(7), 14–26 (2004)
16. Kalb, H., Kasioumis, N., Llopis, J.G., Postaci, S., Arango-Docio, S.: Blogforever: D4.1 user requirements and platform specifications. Tech. rep. (2011)
17. Kalb, H., Trier, M.: The blogosphere as œuvre: Individual and collective influences on bloggers. In: ECIS 2012 Proceedings (2012). URL `http://aisel.aisnet.org/ecis2012/110`
18. Kaplan, A.M., Haenlein, M.: The early bird catches the news: Nine things you should know about micro-blogging. Business Horizons **54**(2), 105–113 (2011)
19. Kim, Y., Ross, S., Stepanyan, K., Pinsent, E., Sleeman, P., Arango-Docio, S., Banos, V., Trochidis, I., Llopis, J.G., Kalb, H.: Blogforever: D3.1 preservation strategy report. Tech. rep. (2012)
20. Lavoie, B.: The open archival information system reference model: Introductory guide. Microform & Digitization Review **33**(2), 68–81 (2004)
21. Lindahl, C., Blount, E.: Weblogs: simplifying web publishing. Computer **36**(11), 114 – 116 (2003)
22. Masanès, J.: Web Archiving. Springer-Verlag, Berli, Heidelberg (2006)
23. McEwen, S.: Requirements: An introduction (2004). URL `http://www.ibm.com/developerworks/rational/library/4166.html`
24. (OPFRO), O.P.F.R.O.: Data requirement. URL `http://www.opfro.org/index.html?Components/WorkProducts/RequirementsSet/Requirements/DataRequirements.html~Contents`
25. Pennock, M., Davis, R.M.: Archivepress: A really simple solution to archiving blog content. In: Sixth International Conference on Preservation of Digital Objects (iPRES 2009), pp. 148–154. California Digital Library, San Francisco, USA (2009)
26. Rusbridge, C.: Preservation for scholarly blogs. URL `http://www.gavinbaker.com/2009/03/30/preservation-for-scholarly-blogs/`
27. Rynning, M., Banos, V., Stepanyan, K., Joy, M., Gulliksen, M.: Blogforever: D2.4 weblog spider prototype and associated methodology. Tech. rep. (2011)
28. Sanderson, R., Shankar, H., Ainsworth, S., McCown, F., Adams, S.: Implementing Time Travel for the Web. The Code4Lib Journal (13) (2011). URL `http://journal.code4lib.org/articles/4979/comment-page-1`
29. Stepanyan, K., Gkotsis, G., Kalb, H., Kim, Y., Cristea, A., Joy, M., Trier, M., Ross, S.: Blogs as objects of preservation: Advancing the discussion on significant properties. In: 9th International Conference on Preservation of Digital Objects (iPRES 2012). Toronto, Canada (2012)
30. Stepanyan, K., Gkotsis, G., Pincent, E., Banos V. abd Davis, R.: Blogforever: D2.6 data extraction methodology. Tech. rep. (2012)
31. Stepanyan, K., Joy, M., Cristea, A., Kim, Y., Pinsent, E., Kopidaki, S.: Blogforever: D2.3 weblog data model. Tech. rep. (2011)

32. Vangelis Banos, N.B., Manolopoulos, Y.: Trends in blog preservation. In: ICEIS Conference Proceedings (2012)
33. Weltevrede, E., Helmond, A.: Where do bloggers blog? platform transitions within the historical dutch blogosphere. First Monday **17**(2-6) (2012). URL `http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/3775/3142`
34. Zúñiga, V.T.: Blogs as an effective tool to teach and popularize physics: a case study. Latin-American Journal of Physics Education **3**(2), 4 (2009)