Exploring the Consistent behavior of Information Services

Sarantos Kapidakis[1a]

[1]*Laboratory on Digital Libraries and Electronic Publishing, Department of Archive, Library and Museum Sciences, Faculty of Information Science and Informatics, Ionian University, 72 Ioannou Theotoki Str., Corfu 49100, Greece*

Abstract. Computer services are normally assumed to work well all the time. This usually happens for crucial services like bank electronic services, but not necessarily so for others, that there is no commercial interest in their operation. In this work we examined the operation and the errors of information services and tried to find clues that will help predicting the consistency of the behavior and the quality of the harvesting, which is harder because of the transient conditions and the many services and the huge amount of harvested information. We found many unexpected situations. The services that always successfully satisfy a request may in fact return part of it. A significant part of the OAI services have ceased working while many other serves occasionally fail to respond. Some services fail in the same way each time, and we pronounce them dead, as we do not see a way to overcome that. Others also always, or sometimes fail, but not in the same way, and we hope that their behavior is affected by temporary factors, that may improve later on. We categorized the services into classes, to study their behavior in more detail.

1 Introduction and Related work

The computer services do not always have the right behavior. The users may notice delays or unavailability – but they have no idea how often these happen, and assume that each problem is rare and temporary.

The users usually have higher operation expectations from the commercial services than from the free services. Therefore, the non commercial service providers do not have the same motivation, and even no resources, to ensure reliable and fast operation.

The big diversity of computer services, their different requirements, designs and interfaces and also network problems and user-side malfunctions, make it hard to know when a service behavior is normal or not and what to measure in each case.

To overcome some of these restrictions, we examined the behavior of a large number of similar services of a specific type: data providers using the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH).

Metadata harvesting is used very often, to incorporate the resources of small or big providers to large collections. The metadata harvesters, like Science Digital Library and Europeana, accumulate metadata from many collections (or sources) through the appropriate services, belonging to metadata providers mostly memory institutions, by automatically contacting their services and storing the retrieved metadata locally. Their goal is to enable searching on the huge quantity of heterogeneous content, using only their locally store content. Metadata harvesting is very common nowadays and is based on the Open Archives Initiative Protocol for Metadata Harvesting. As examples, we mention the Directory of Open Access Repositories (OpenDOAR) that provides an authoritative directory of academic open access repositories[a], and the OAIster[b] database of OCLC with millions of digital resources from thousands of providers.

The reliability of the services is important for ensuring current information. If the metadata harvesting service is not responding, the corresponding metadata records will not be updated at that time. If the reason is temporary, this may not be a very serious deficiency, as the updates will eventually be performed on the next successful metadata exchange, and will be normally used afterwards. Nevertheless, the unreliability - downtime of the metadata harvesting services usually indicate a proportional unreliability or downtime of the resource providing service, which always resides on the local sites, where both the local and the harvested metadata link to. When the resources are not available, the corresponding user requests are not satisfied, affecting the quality of the service.

In [7] Lagoze et al. discuss the NSDL development and explains why OAI-PMH based systems are not relatively easy to automate and administer with low people cost, as one would expect from the simplicity of

[a] Corresponding Author : sarantos@ionio.gr

[a] http://www.opendoar.org
[b] http://www.oclc.org/oaister.en.html

the technology. It is interesting to investigate the deciciencies of the procedure.

National or large established institutions consistently try to offer their metadata and data reliably and current and to keep the quality of their services as high as possible, but local and smaller institutions often do not have the necessary resources for consistent quality services – sometimes not even for creating metadata, or for digitizing their objects. In small institutions, the reliability and quality issues are more prominent, and decisions often should also take the quality of the services under consideration.

The behavior and the reliability of a service, as well as the quality of its content, is important to the outcome and the satisfaction from the service. The evaluation and quality of metadata is examined as one dimension of the digital library evaluation frameworks and systems in the related literature, like [2], [8] and [11]. Fuhr et al. in [2] propose a quality framework for digital libraries that deal with quality parameters. The service reliability falls under their System quality component, and the metadata update under their Content quality component.

In [9] Ward describes how the Dublin Core is used by 100 Data Providers registered with the Open Archives Initiative and shows that is not used to its fullest extent. In [4] Kapidakis presents quality metrics and quality measurement tool, and applied them to compare the quality in Europeana and other collections, that are using the OAI-PMH protocol to aggregate metadata. In [5] Kapidakis further studies the responsiveness of the same OAI services, and the evolution of the metadata quality over 3 harvesting rounds between 2011 and 2013.

From the different aspects of the quality of digital library services, the quality of the metadata has been mostly studied. Some approaches are applied on OAI-PMH aggregated metadata: Bui and Park in [1] provide quality assessments for the National Science Digital Library metadata repository, studying the uneven distribution of the one million records and the number of occurrences of each Dublin Core element in these. Another approach to metadata quality evaluation is applied to the open language archives community (OLAC) in [3] by Hughes that is using many OLAC controlled vocabularies. Ochoa and Duval in [9] perform automatic evaluation of metadata quality in digital repositories for the ARIADNE project, using humans to review the quality metric for the metadata that was based on textual information content metric values.

In [6] Kapidakis examines how solid is the metadata harvesting procedure, making 17 harvesting rounds, over three years, from 2014 to 2016, and exploiting the results to conclude on the quality of their metadata as well as on their availability, and how it evolves over these harvesting rounds. The list of working services was decreasing every month almost constantly, and less than half of the initial services continued working.

Nevertheless, the services did not have a solid behavior, which make any assumption and conclusion harder to reach. In this work we explore the behavior of the information services over a small period of time, so that no permanent changes to their behavior are expected and we are able to study their consistency without considering their evolution.

The rest of the paper is organized as follows: In section 2 we describe our methodology and the data we collected for that purpose. In section 3 we analyze the data to find patterns of behavior. In section 5 we classify the services into classes according to their behavior over consecutive observations, in many harvesting rounds, and study each class for further behavior details. We conclude on section 5.

2 Methodology and Collected Data

It is difficult to understand, analyze or predict the behavior of network services, because it depends on many factors, many of which may be external to the service and unknown. Nevertheless, there may be some significant factors of the service configuration or maintenance, or their environment (including the accessing network) that can be considered.

To reveal common behavior patterns, we used several harvesting rounds, where on each one we asked each of the 2121 OAI-PMH sources listed in the official OAI Registered Data Providers[c] on January of 2014 for a similar task: to provide 1000 (valid) metadata records. Such tasks are common for the OAI-PMH services, which periodically satisfy harvesting requests for the new or updated records, and involve the exchange of many requests and responses, starting from the a negotiation phase for the supported features of the two sides.

Our sequential execution of all these record harvesting tasks from the corresponding services, usually takes more then 24 hours to complete, and sometimes some tasks time out (we set a timeout deadline to 1 hour for each task, and interrupted any incomplete task just afterwards, so that it will not last forever) resulting to abnormal termination of the task.

Ideally, a task will complete normally, returning all requested metadata records – 1000, or all records if fewer (even zero) are only available on that service. Other behaviors are also possible - and of interest when studying the behavior of each service. A task may return an error, declaring that the final task goal cannot be satisfied. We also consider as error the situation of the abnormal termination of a timed out task, as it is not clear if any useful part of the requested goal was completed, and if the communication sides have the same view about it. If, because of an erroneous situation, both communication sides do not agree on the harvested records, the have to be re harvested in the future, and are not useful in the meanwhile.

Finally, it is also possible to have a situation that the service actually returns less records than the ones available and requested, but reports that the task completes normally. Nevertheless, we classify this behavior with the successful ones, because both sides acknowledge the harvesting of these records, and on periodic normal harvesting – while the remaining of the records will eventually also be harvested, even on a later effort. This situation looks also similar to the expected

[c] https://www.openarchives.org/Register/BrowseSites

behavior of returning all available records, when they are fewer than the requested ones, and the two can only be distinguished when examining other responses of the same service.

We repeat a new harvesting round with all the task to the services in constant intervals, asking the exact same services every three days for a period of two months. We selected this interval so that we do not overload the services and also leave enough time for the services to correct temporary problems – especially if they need human attention.

Below, the behavior of the participating services is examined collectively, to better understand the possible outcomes, and their likelihood.

3 Collective Behavior of the Services

On our 44541 tasks on all 21 harvesting rounds, 21478 tasks (48%) completed declaring a success, with average response time 60.5 seconds, although only 20409 (95%) of them returned records. For tasks that returned records, the minimum response time was less than a second and the maximum was 3175 seconds, in a case that the information service returned only 200 of the 1000 records, while there were more records (as they were returned on all other requests). Because of the short interval between our harvesting records, in most cases neither the service software nor the records in each service were modified between rounds. We still consider responses with fewer than the requested records as successful, because the remaining of the requested records can be returned on a successive request, in the common OAI-PMH configuration.

We study the successful responses for each harvesting round, to detect if some of the rounds had much higher or lower success rates: the successful responses on each of the 21 rounds range from 977 to 1040, with an average of 1022.8 and standard deviation 15.8. The similar results from all rounds indicate that there was no special conditions (possibly on our network) that will affect the outcome of our tasks and contaminate our analysis. Otherwise, we should have excluded them.

We study the distribution of the number of records returned by each service for common values, other than the requested records, 1000. Figure 1 presents the distribution of the number of records returned by each of the 1079 services that responded successfully on any of the 21 rounds. 324 of the services were successful only on some rounds, and 125 of them were successful on all rounds, but did not provide the same number of records on each round. In these cases, the maximum number of records returned was considered. The remaining 630 services always returned the same number records. The more rounds a service responded successfully – especially when their maximum is repeated in many rounds, the safer it is to assume that we have found the number of records that should be returned on each task.

Only 633 of the 1079 services returned 1000 records, and may actually have more records. On the other hand, 61 services always returned 0 records, and considered it as a successful task. It is not clear if all their records are deleted, or there is an incompatibility on accessing the service that is not detected properly. The remaining 385 services returned from 2 to 994 records (217 of them consistently in all 21 rounds – the others occasionally returned less) and with no obvious pattern: as can be seen in Figure 1, 203 services had a unique number of records, 55 pairs of services shared the number of returned records, etc, and, on the maximum, 6 services returned the same number of records (34). The number of returned records seems to vary unpredictably.

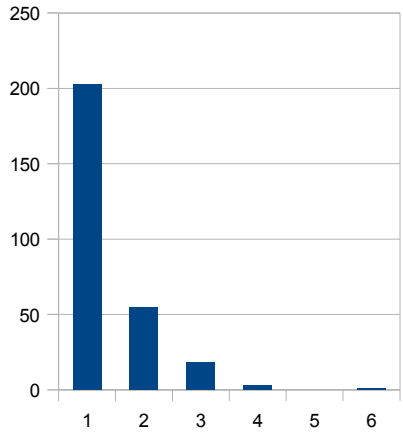

Figure 1. The distribution of the number of records returned by a service and its frequency for the 385 services.

The other 23063 times the service ended with an error: in 22073 (96%) of these tasks, no records were retrieved at all, while the remaining 990 times some records (57 distinct numbers ranging from 20 to 1000, with an average around 400 – in most cases it was a round number though) were indeed returned. Among these cases, in 62 the service did not terminate but timed out after an hour of trying, and out of them in 45 cases, no records were retrieved at all, while the remaining 17 times some records (ranging 20 to 1000) were indeed returned. The records that were returned when the service ended with an error were ignored in our study.

As each task consists of many actions, its outcome may not be trivially classified as full success or failure, but may be something in between, with contradicting indicators, like when retrieving records and getting an error status, or when retrieving no records and getting a success status, resulting to partial success or failure. Therefore, the success and the failure of a task has to be clearly defined, so that it can be used later on. In this work, we decided to adopt the status returned from the task in order to characterize it as success or failure, not considering the number of records actually returned. This has the advantage that on the failures we can interpret the returned error status to further explore the reason for the failure. The outcome of a task is further classified into more cases later on.

When a task reports a failure, there is a variety of errors that appear, as returned by each service, that are briefly listed in Table 1 with their frequencies. The error messages are service specific and the exact semantics of each error message is not globally defined, although we

can assume the semantics of most errors by their complete message.

Apart from the normal responses and the timed-out responses, we present the remaining errors by splitting them into two categories, the ones related to the network communication, that may apply to more services, and the ones that are specific to the OAI-PMH operation, and its data interpretation and exchange. There are errors that their short name is too generic to explain. In order to decide in which category each error belong, we examined the complete error message, although on the table we only show its short form, as returned by the software of the service.

Table 1. The error messages by category.

Error short name	*Frequency*
Normal Completion	
	21478
Timed-out Execution	
Timeout	62
Communication Errors	
Error	170
error	332
IncompleteRead	10
HTTPError	6985
SSLError	48
URLError	7401
OAI-PMH Protocol Errors	
UnicodeEncodeError	125
BadArgumentError	356
BadStatusLine	54
BadVerbError	3507
CannotDisseminateFormatError	42
NoRecordsMatchError	682
XMLSyntaxError	2771
DatestampError	438
BadResumptionTokenError	80

We observe that the communication errors are about twice more often than the OAI-PMH protocol errors.

Some of the errors may be permanent, requiring human intervention (in the data or the software) to correct them, while other errors are temporary, and may not occur on another request. The temporary errors may affect not only the successful tasks, but also those with permanent errors, temporarily hiding them, making the analysis of each behavior more complex.

4 Consistency in the Behavior

We want to consider the behavior of each service over time, our harvesting rounds, to see how consistent it is. The temporary errors may hide the permanent behavior , but there are many services with a consistent behavior: in fact, we found that 1668 of the services had the “same” behavior each time: either they always succeeded or they always failed with the same way - by timeout or by reporting the same error. We did not expect the behavior of these services to change much.

To get a more accurate picture, we split our services into groups, according with their response behavior, that we called classes, and made a separate analysis for each one of these classes, which are shown in Table 2 with the number of services in each class.

Table 2. Services by behavior type.

#	class	services
1	full	630
2	some	125
3	both	324
4	fail	129
5	dead	913
	total	**2121**

The classes are defined as follows:

1. full: the services that always succeeded and always returned the same number of records, been as consistent as one could wish for.
2. some: the services that always succeeded but did not always return the same number of records each time. Their behavior is good, but could be improved to be more consistent.
3. both: the services that sometimes succeeded and sometimes failed and reported an error. Their errors should be temporary, as they do not prevent the successful behavior on other times.
4. fail: the services that always fail, but report different reason / errors – some of the errors should be temporary, but also some permanent errors may be hidden below them.
5. dead: the services that always fail, but always report the same error – which must be (in most cases) permanent.

The class with the most services (913) is the dead class, the services that seem to have stopped responding for one specific reason. The second class in size is the full class (with 630 services), with the services that work as expected. The classes with more transient / less consistent behavior follow in size. The behavior characteristic of each class can be further examined.

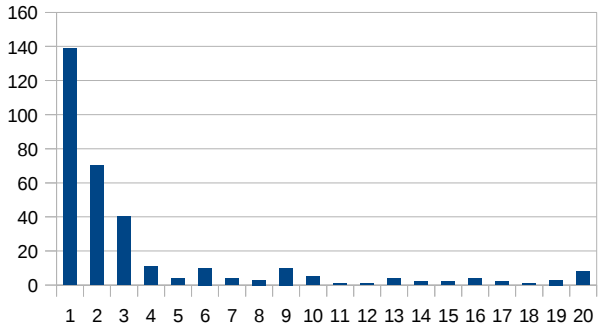

Figure 2. The number of failed tasks per service (class both).

We initially examine the return status of the tasks: only in class both, the successful and failed tasks coexist for each service, as the other classes contain the services that always failed or always succeeded. The number of failed tasks per service for class both is shown in Figure 2. We observe that, although most services failed just one time, and a significant number of them two or three times, there are services that failed even all but one time. Although most services fail occasionally, some of them only succeed occasionally.

The number of failed tasks per round for class both is shown in Figure 3. The average number of failed tasks per round is 56.2, with minimum 39 and maximum 102. Rounds 4 and 6 had a higher number of failures, but not too high to indicate a problem on the client side.

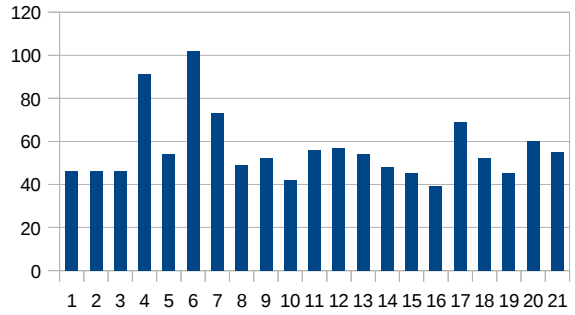

Figure 3. The number of failed tasks per round (class both).

We also examine the outcome of the harvesting: the average records retrieved from each successful task for the services of each class are presented in Table 3. Their number always ranges from 0 to 1000, and the average and standard deviation shows that the services in the class full usually provide more records, than the other, less consistent, services. It looks like the consistency problems affects the number of returned records.

Table 3. Records per successful task.

class	min	max	average	sdev	#tasks
full	0	1000	752.175	394.709	13230
some	0	1000	381.698	317.244	2625
both	0	1000	681.923	420.788	5623

In Table 4 we examine for each service the number of times that the number of returned records is different from that of the previous harvesting round (that was only 3 days earlier , and in most cases denote a problem in the harvesting and not a change in the records).

Table 4. Record changes per service.

class	min	max	average	sdev	#services
full	0	0	0.0	0	630
some	1	15	2.712	2.764	125
both	0	17	0.481	1.728	324

For the 630 services in the class full, this is, by definition, 0: only these services are included in the class. In the class some, this is from 1 up to 15, but with a much lower average and standard deviation. Similarly for class both, although it may be as high as 17, its average is less than 0.5.

In class some all tasks are successful, but some of them only partially, returning less records than those available and requested. The number of partially successful tasks per service is shown in Figure 4. We observe that the services may partially succeed quite often, from 1 to the maximum of 20 times, and on average 11.7 times (standard deviation is 6.7), often having more often partially than complete success. The partial success of the tasks seems quite unpredictable.

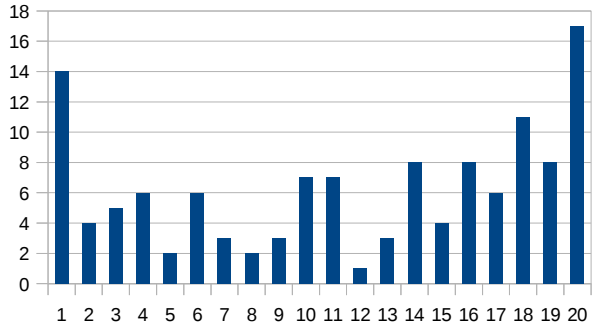

Figure 4. The number of partially successful tasks per service (class some).

The number of failed tasks per round is shown in Figure 5. It looks like that the services get improved on each round, and decrease the number of partial success, and increase the number of complete success on each round. The strict decrease of the number of partial success was not predicted. We assume that the class contains mostly services that do not work perfectly, but are improved somehow, while the services that get worse will eventually fail and will be classified under the class both.

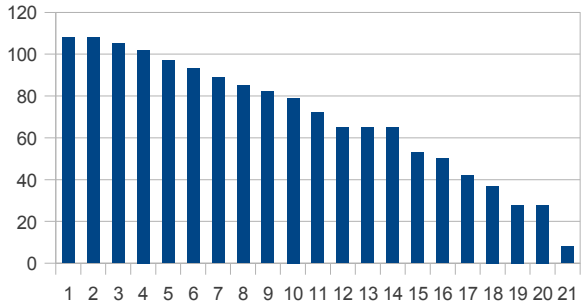

Figure 5. The number of partially successful tasks per round (class some).

The response time of the 21478 successful tasks, per service class, is shown in Table 5 . We observe that the average response time is smaller on the classes that are not consistently returning all the asked records, probably because they perform less work in this case, and return less records on the average. We conclude that some services are consistently more stable than others, although they so a similar job and often use similar software.

Table 5. Response time per successful task.

class	min	max	average	sdev	#tasks
full	0.43	2669.9	65.53	149.5	13230
some	0.34	234.5	27.82	32.7	2625
both	0.07	3175.4	63.95	122.1	5623

The remaining 23063 tasks return an error. These errors are shown in Table 6 and their frequency per service class. From the class dead, we see that only the errors IncompleteRead and Timeout are not always persistent in the same service, and all others may be persistent, which usually happen in the permanent errors. We cannot claim that a specific error occurs more often in the services of any class, even considering the total number of errors in each class.

Table 6. Frequency of service errors in each class.

error / class	both	fail	dead
BadArgumentError		20	336
BadResumptionTokenError	17		63
BadStatusLine	11	1	42
BadVerbError	99	384	3024
CannotDisseminateFormatError			42
DatestampError	7	158	273
Error	25	19	126
HTTPError	297	934	5754
IncompleteRead	10		
NoRecordsMatchError	140	143	399
SSLError		6	42
Timeout	37	25	
URLError	400	449	6552
UnicodeEncodeError		20	105
XMLSyntaxError	101	486	2184
error	37	64	231

Finally, we examine how often the errors do change to a different error (or to a success), between successive tasks of a service. This, by definition of the classes, only happens in services of the classes both and fail, and does not change in the other 3 classes.

Table 7. Service error changes per service

class	min	max	average	sdev	#services
both	1	13	3.056	1.931	324
fail	1	12	2.612	1.724	129

In Table 7 we examine for each service the number of times that the returned error is different from that of the previous harvesting round. Although it can change as high as 13 times, on average it changes from 2.5 to 3 times, with higher average and standard deviation on the class both: this class seems to have more unpredictable behavior.

5 Conclusions and Future Work

Communication of user and services through networking is complex, and we often do not have a picture of all the involved factors and situations. Understanding how the services behave, either qualitatively or quantitatively, help us understand the current situation and better design future services.

We studied the behavior of the services in many rounds of record harvesting, which is a complex procedure, with many requests through the network. We examined the number of returned records, and realized that in some cases less records are returned. Also, the service may report an error, and still some of the requested records may have been transferred. Therefore, the successful task is not obvious, and we had to define it, considering what is “successful enough”.

The tasks are executed in multiple rounds, which are close enough to each other to avoid significant changes on the services. Analyzing the success of the tasks in a round demonstrates if unexpected circumstances occurred during this round. The behavior of a service over multiple rounds of tasks is not always consistent, and it is not rare that it changes from round to round, because of temporary problems. We categorized the services into classes according to the consistency of their behavior, and how the inconsistent behavior varies, and studied aspects of the behavior that appear in each class.

No obvious patterns of behavior were detected. Nevertheless, that some services are consistently more

stable than others, although they do a similar job and often use similar software.

In the future, we plan to further analyze the errors as permanent or temporary, exclude the tasks with the temporary errors and analyze the others for more permanent characteristics, as well as to examine the error distribution per service.

References

1. Y. Bui & J. Park, *An assessment of metadata quality: a case study of the National Science Digital Library Metadata Repository*, In Haidar Moukdad (Ed.) CAIS/ACSI 2006 Information Science Revisited: Approaches to Innovation. Proceedings of the 2005 annual conference of the Canadian Association for Information Science held with the Congress of the Social Sciences and Humanities of Canada at York University, Toronto, Ontario (2005)
2. N. Fuhr, G. Tsakonas, T. Aalberg, M. Agosti, P. Hansen, S. Kapidakis, P. Klas, L. Kovács, M. Landoni, A. Micsik, C. Papatheodorou, C. Peters and I. Sølvberg, *Evaluation of Digital Libraries*, International Journal of Digital Library, Springer-Verlag, vol. **8**, no 1, November 2007, pp. 21-38 (2007)
3. B. Hughes, *Metadata quality evaluation: experience from the open language archives community*, Berlin: Springer. *Lecture Notes in Computer Science* vol. **3334**. ISBN 978-3-540-24030-3. doi: 10.1007/b104284 (2005)
4. S. Kapidakis, *Comparing Metadata Quality in the Europeana Context*, Proceedings of the 5th ACM international conference on PErvasive Technologies Related to Assistive Environments (PETRA 2012), Heraklion, Greece, June 6-8 2012, ACM International Conference Proceeding Series; vol. **661** (2012)
5. S. Kapidakis, *Rating Quality in Metadata Harvesting*, Proceedings of the 8th ACM international conference on PErvasive Technologies Related to Assistive Environments (PETRA 2015), Corfu, Greece, July 1-3 2015, ACM International Conference Proceeding Series; ISBN 978-1-4503-3452-5 (2015)
6. S. Kapidakis, *Exploring Metadata Providers Reliability and Update Behavior*, Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL 2016), September 5-9, to appear (2016)
7. C. Lagoze, D. Krafft, T. Cornwell, N. Dushay, D. Eckstrom & J. Saylor, *Metadata aggregation and "automated digital libraries": a retrospective on the NSDL experience*" Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries (JCDL 06), pp. 230-239 (2006)
8. B. L. Moreira, M. A. Goncalves, A. H. F. , Laender & E. A. Fox, *Automatic evaluation of digital libraries with 5SQual*, Journal of Informetrics, vol. 3, 2, pp. 102-123 (2009)
9. X. Ochoa & E. Duval, *Automatic evaluation of metadata quality in digital repositories*, International Journal on Digital Libraries, vol. **10**(2/3), pp. 67–91 (2009)
10. J. Ward, *A quantitative analysis of unqualified dublin core metadata element set usage within data providers registered with the open archives initiative*, Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries (JCDL 03), ISBN:0-7695-1939-3, pp. 315-317 (2003)
11. Y. Zhang, *Developing a holistic model for digital library evaluation*, Journal of the American Society for Information Science *and Technology*, vol. **61**, 1, pp. 88-110 (2010)