Online Journal of Applied Knowledge Management
A Publication of the International Institute for Applied Knowledge Management

Volume 2, Issue 1, 2014

Enhancing m-learning using GridFS for storing and streaming digital content

Milorad P. Stević, The Higher Education Technical School of Professional Studies, Novi Sad, Serbia, milorad.stevic@live.com

Abstract

Mobile devices transformed the way students access shared knowledge today. The initiative described through the usage of mobile technologies for learning purposes is called mobile learning or m-learning. It is a promising platform because it is learner-centred, enhances collaboration and flexibility of the learning process and enables students with physical disabilities to reach educational material. The platform offers not only the exposure of educational material from teachers to the students, but also utilizes the other way-students are now in a position to create educational material and share and discuss it with teachers and other students. The platform is suitable for traditional learning, lifelong learning, practical knowledge sharing and has many other applications. However, during experimental usage of a simple m-learning platform, content management issue arose-the process of expanding the storage was difficult and time consuming. Adding new capacity to the storage would not solve this issue permanently, because new content would eventually trigger the issue again. It was necessary to create an architectural change to solve the problem permanently. This paper describes the change of the underlying architecture of the information system and proposes the usage of MongoDB NoSQL database system and its GridFS specification for digital content management.

Keywords: M-learning, Unstructured data types, NoSQL, Service-oriented architecture, Document management, Information systems, Scalability

Introduction

The need for better inclusion of students into the learning process initiated m-learning experiment in our institution. A small number of students taking one course were selected for the experiment in introducing m-learning as a part of their class activities. Students were encouraged to participate in the process not only as consumers of the educational material, but as contributors as well. The results were promising and it appeared that students were very interested in the applied learning method. However, after some usage it was obvious that storage capacities were insufficient to hold the content students were producing. Adding new hard drives and/or servers to the information system (IS) infrastructure was inevitable. This approach would cause greater engagement of equipment and human resources. In addition, the cost of adding new hardware would be high, since efficient management of digital content involves high-end hardware usage. Even then, it is obvious that the issue will happen again and the process of adding new layers of hardware and complexity would be inevitable. Using relational database management system (RDBMS) for handling digital content and other unstructured data types is not efficient, because RDBMS is efficient for handling structured, not unstructured data. Applying vertical scaling for managing data growth does not solve the problem, only further burdens human and hardware resources, since it is resource consuming. Another problem is streaming data stored inside RDBMS, which is complicated, if possible at all. Other known solutions are established on various file system based repositories, but those have other issues. Some of them are security related, since files are stored in folders in a file system and are accessible directly, without IS control. Further, file systems have limitations regarding file sizes,

number of files in single directories and custom metadata about binary files. Managing data growth is also human and hardware intense.

Numerous authors believe that m-learning is going to be a promising platform for enhancing the learning experience because it is learner-centred (Naismith *et al.*, 2004) and it improves collaboration (Sharples *et al.*, 2005) and flexibility (Bull and Reid, 2004; Vavoula and Sharples, 2002). However, m-learning initiative must improve interoperability characteristics with already established Learning Management Systems (LMS) in order to promote further expansion of both, m-learning initiative and LMSs. This would also provision LMS evolution and its acceptance to new social needs and new technologies (Casany *et al.*, 2012). Nevertheless, there are concerns with interoperability between m-learning and LMS, for LMSs are usually built as monolithic systems (Alier *et al.*, 2010). Casany *et al*. (2012) defines two scenarios for good integration of m-learning with LMS: spreading LMS to mobile world using web services and interoperability initiatives or assimilating external m-learning applications into the LMS.

The proposed architecture change described in this paper offers the usage of NoSQL database system for managing unstructured data types, in this case, digital content in form of binary files. It is MongoDB NoSQL database system, which involves its GridFS specification for handling large unstructured data. This specification can be used for managing digital content for m-learning purposes. Because of the underlying MongoDB database system, the specification uses several important database features.

One important feature is horizontal scaling accompanied by partition tolerance. As a distributive system, MongoDB can scale horizontally, adding new servers and sharing data among servers without administrative intervention. Of course, rules for the deployment of this feature need to be defined in advance. Servers need not be expensive and they need not have similar hardware characteristics. Data is spread over servers in a way that enables load balancing of the content on different servers, which is important for horizontal scaling process.

Another important feature is the fact that data inside GridFS is stored in chunks, which makes streaming of data not only possible, but effective too. The size of chunks can be adjusted to better match concrete needs of the implemented file system. Chunk size should be synchronized with underlying operating system and file system for achieving the best results in file manipulation. Furthermore, using chunks empowers IS to implement features as rewinding content forward and backward without complicated client side implementation, which is useful for mobile platform in particular.

M-learning in distant education

In the beginning, distant education was based on print material and later evolved into e-learning, utilizing both open source and proprietary learning management system (LMS) platforms. With wider use of mobile technology, the possibility of using mobile devices in learning process appeared. It was a logical step forward, enabling participants to explore new possibilities brought by the new hardware platform. Nowadays, m-learning is growing innovation that allows users to participate in the learning process while on the move, thus expanding educational options

(Chandhok and Babbar, 2011). M-learning is interesting learning option not only for academic institutions, but for organizations too, since practical knowledge sharing and video on demand are becoming more and more interesting features for organizations (Elliott, 2012). One of interesting applications is the next generation of the hotel training system (Kim and Kizildag, 2011). Examples like this show that in the competitive and fluctuant world, organizations need to change the way in which they train their employees. Mobile devices, such as mobile phones, PDAs and MP3 players will increasingly be used to help train and support the performance of the personnel.

Although a number of m-learning projects have been initiated worldwide, guidance for m-learning adoption is still much needed. This is because technology alone does not enable m-learning; the key success element for adoption of m-learning is to comprehend the concerns of learners and to identify causes which lead to learners' motivation to adopt m-learning (Liu at al., 2010). The example of good appliance of the concept is the initiative for enrolling a small number of students in online graduation program with iPods as part of their learning materials, with positive experiences from the initiative (Richardson et al., 2013). In addition, systems behind m-learning paradigm need to be prepared for changes that are going to happen with wider adoption of m-learning. The amount of digital material for storing and efficient handling is becoming very large, since every participant in m-learning system can upload hundreds of MBs of digital content into the m-learning system. This is happening on a daily basis, creating difficulties for m-learning system to cope with this quantity of digital content. Uploaded content is uploaded for use in a learning process and clearly, uploading and storing this content is only a part of the problem. The next issue is efficient download and streaming of this content and this is going to be even more intense and more frequent because digital content is downloaded more often than uploaded.

Distributed systems based on commodity hardware provide an inexpensive computing resource (Kehagias et. al., 2006). Using schema-free systems based on a distributed architecture is a good solution for handling unstructured data (Hellerstein, Stonebraker & Hamilton, 2007). NoSQL databases are designed for handling large amount of unstructured data by horizontal scaling, thus removing the obstacles regarding future growth of data (Hellerstein, Stonebraker & Hamilton, 2007). This is where MongoDB GridFS specification is useful, because relaying on its automatic horizontal scalability and streaming content directly from the database can make the process of developing m-learning solution much easier and reliable. With its mirroring options deployed, MongoDB is even more desirable platform since download can be performed from every server in a mirror, independently. Configured like this, the proposed platform further increases its availability, since every server in a mirror serves as a potential download server. Once configured, this configuration can be maintained almost automatically.

Software architecture

Current architecture

In order to prove that proposed architecture offers improvements, the comparison with current architecture has to be conducted. Current architecture handles digital content using file system based repositories, which involves expensive hardware in order to maintain safety of collected content. This enables the protection of digital content in cases of hardware failures. However, there are still problems with conducting disaster recovery scenarios, achieving scalability of the system and enabling high availability of the system. Resolving problems that occur with this approach always implies acquisition of expensive hardware and is always human and time consuming. Current architecture tries to address these problems in several ways, as shown in table 1.

Table 1: Resolving issues in current architecture

Issue	Resolution	Disadvantages
Disaster recovery	• Use of expensive hardware capable of preserving files in case of hardware failures inside data centres, usually using drive mirroring • Deployment of a servers in a distant location for data center failures • Using synchronization jobs for replicating data to a distant location	• Expensive hardware acquisition • Synchronization jobs overlapping • Synchronization jobs times increases • Many simultaneous jobs overload servers or networks • Problems with coordinating additional steps needed after synchronization • Problems with monitoring jobs and alerting on failures • Human and time consuming operations
Scalability	• Adding new drives into existing storages • Adding new servers into existing data centres	• Expensive hardware acquisition • Adding new locations for storing files requires adjusting application/service layers of information system • Error prone operation considering that software needs adaptation • Human and time consuming operations
Availability	• Adding new data servers into existing data centres • Adding new application servers into existing data centres	• Expensive hardware acquisition • Adding new data/application servers requires adjusting application/service layers of the information system • Error prone operation considering that software needs adaptation • Human and time consuming operations

MongoDB GridFS architecture

MongoDB is an open-source document-oriented database system that enables automatic horizontal scaling through sharding and good performance and availability features. It is a NoSQL database that supports dynamic schemas and stores documents in a form of BSON

objects, which is similar to JSON objects. Main features of MongoDB are:

- Ad Hoc Queries-search by field, range queries and regular expression searches
- Indexing-any field in a MongoDB document can be indexed
- Replication-creating replica sets enables failover, better availability and data throughput
- Load balancing-using horizontal scaling MongoDB automates the process of load balancing over multiple servers
- File Storage-using GridFS specification, MongoDB can be used for storing files, taking advantage of automatic content streaming, load balancing and data replication features
- Server-side java script execution-MongoDB enables use of java-script in queries
- Capped collections-fixed size collections in MongoDB are called capped collections

GridFS is a specification for storing and retrieving files that exceed the BSON-document size limit of 16 MB. Instead of storing a file in a single document, GridFS divides a file into parts, or chunks and stores each of those chunks as a separate document. By default, GridFS limits chunk size to 256 KB, but this parameter can be altered. Altering default chunk size can increase computer network utilization and file manipulation operations. GridFS specifies two collections for storing files. The first collection stores file chunks, and the second one stores file metadata. This separation is important because searching operations based on metadata can be performed efficiently, without interacting with the collection that contains actual binary files.

When queried for a file, GridFS will, using the corresponding driver or client, reassemble chunks as needed. It is possible to perform range queries on files stored through GridFS. It is even possible to access information from arbitrary sections of files, which allows skipping into the middle of a video or audio file. Furthermore, GridFS is useful not only for storing files that exceed 16 MB, but also for storing any files for which it is necessary to enable access without having to load the entire file into memory.

In contrast to current approach which proposes vertical scaling of the system where new server replaces current server which is no more suitable for handling data due to small storage capacities or small amount of RAM, new approach introduces horizontal scaling of the system. Sharding, or horizontal partitioning, is the process of splitting data across multiple servers. Each server hosts a subset of data, thus decreasing the load on the servers. Reaching the storage capacity does not mean that a more powerful server is needed. Adding another less powerful server will trigger load balancing in the background, enabling the database system to handle more data. MongoDB with its GridFS specification offers specific approach in conducting disaster recovery scenarios, achieving scalability and enabling high availability of the system. GridFS tries to resolve these problems using its distributive architecture implemented on commodity hardware to achieve less expensive hardware implementation. Table 2 shows ways in which GridFS resolves issues with disaster recovery, scalability and availability.

Table 2: Resolving issues using MongoDB GridFS architecture

Issue	Resolution	Advantages
Disaster recovery	• Use of commodity hardware in distributed architecture enables preserving files in case of hardware failures inside data centres • Deployment of a commodity hardware in a distant location for data center failures • Using MongoDB replication for replicating data to a distant location	• Less expensive hardware acquisition • No synchronization jobs, data is replicated using automated MongoDB replication functionalities • No servers or network overload • No additional steps needed after synchronization, since there is no synchronization • Automated nature of replication process ensures small human involvement in the process
Scalability	• Adding new commodity servers into existing data centres	• Less expensive hardware acquisition • Adding new commodity servers does not require adjusting application/service layers of information system • Automated nature of sharding process ensures small human involvement in the process
Availability	• Adding new commodity data servers into existing data centres	• Less expensive hardware acquisition • Adding new commodity data servers does not require adjusting application/service layers of information system • Automated natures of replication and sharding processes ensure small human involvement in the process

Applying proposed architecture in a case study example

During the last semester in The Higher Education Technical School of Professional Studies in Novi Sad, a small group of students was established in order to estimate the potential for the usage of a simple m-learning platform. The group consisted from 60 students from information technology department enrolled in the final year of their studies-55 male and 5 female students, during the whole semester. They supposed to actively participate in the learning process using m-learning platform that was developed for this particular purpose. Participation consisted from uploading and sharing files and discussing already uploaded digital content. The purpose of the m-learning platform that was to be developed was to enrich existing distant learning system used in our institution. It was envisaged that m-learning system should use existing infrastructures, database and application servers, wifi connections and other infrastructure elements. The m-learning platform was not intended to be a platform itself, but another way of delivering content to the user.

After several months of usage, drives that were dedicated for storing digital content were exhausted and this issue needed resolution. Since it was an experiment, large drives were not used for storing files. The architecture was based on storing files on a file system and it was obvious that adding new drives/servers would cause the change in software that was developed.

Furthermore, this approach would not permanently solve the problem; it would rise again and adding new capacities would be necessary. However, since metadata was already handled using MongoDB database system and since this database system contained a specification for file handling, a new possibility appeared. Using MongoDB GridFS specification, files that were previously stored in a file system based repository were moved to MongoDB GridFS. The new architecture was completely relying on MongoDB database system: handling metadata was entrusted to MongoDB and handling binary files to MongoDB GridFS. Since all data was under control of MongoDB, it was possible to apply all MongoDB features on handling binary files. In the observed situation, important features were horizontal scaling, replication and file streaming. Streaming files was achievable as streaming complete files and as streaming specific portions of a file. The last feature appeared to be very useful in streaming large digital content, where user was interested in just a portion of a digital content, enabling the implementation of this feature in a very efficient manner. At the same time, horizontal scaling enabled adding new servers and new capacity to the system at the same time without having to change service layer or the application layer of the system, making the system maintenance less error prone and more stable during its use in production.

Since both approaches were analyzed, it was possible to compare time and human effort needed for deploying and maintaining the system. The results for two explained approaches, showing both time and programming effort are presented in a table 3. Time values in the table 3 represent time needed for one employee (programmer/administrator) to finish specific task.

Table 3: Time and programming effort needed for deploying and maintaining the system

Needed time and human resources	**File system based repository**		**MongoDB GridFS**		**Programming activity**
	Time (hours)	**Frequency**	**Time (hours)**	**Frequency**	
Adding one new server	2	Once per server	2	Once per server	No
Controlling file system usage (disk usage, acceptable number of files per folder…)	1	Weekly	-	-	No
Establishing synchronization/replication activities	1	Once per configuration	1	Once per configuration	No
Supervising synchronization/replication activities	0.5	Daily	-	-	No
Performing after synchronization/replication activities	0.5	Daily	-	-	No
Adjusting application because of new file locations	8	Once per event	-	-	Yes
Downtime during adjustment	0.5	Once per event	-	-	No

Discussion

Using file system based repositories for handling digital content has been the initial approach for handling binary files. However, after first needed scaling of the system, it was obvious that the intervention will be both human and time consuming, even on a very small system. While adjusting the system, errors were possible, downtime was present and the need for different approach arose. Proposed solution to the problem involves usage of MongoDB database systems and its GridFS specification for handling binary files. Once transferred to the database, files were handled together with the rest of database content, which enabled automatic replication, horizontal partitioning and streaming of digital content directly from the database. The change of underlying architecture brought zero time needed for adjustment of the system after scaling the system, which enabled the system to become more available, less error prone and stable.

Results presented in table 3 represent analysis of a small system, with only a couple of commodity servers and it is expected that in a bigger system with more servers measured values would be greater. Considering this, proposed solution is becoming more attractive with the increase of a number of servers in a system. This is expected behavior, since MongoDB was designed as a distributed system and performs better with the increase of the number of servers included in the system. Although the time needed for the establishment of the system was not reduced, elimination of time and human involvement in activities after synchronization is significant.

Conclusions

The approach to organize a small group of students and to create a small and functional m-learning application enabled easy transition of the system and efficient experimenting with different architectural approaches. After establishing two different architectures, measurement took place. Investigating measured values revealed that the proposed architecture is more efficient, more stable, less error prone and less human demanding. This is even more apparent with the increase of the number of servers involved in the process.

Handling large amount of digital content using MongoDB GridFS specification enabled the use of less expensive hardware for achieving disaster recovery, scalability and availability of the system. It was possible to deploy distributed architecture using commodity hardware that would enable information system to resolve noted problems. Furthermore, the process of streaming content was enhanced, making the proposed solution more efficient than the file system based repository architecture. Since handling binary files is important in any contemporary LMS, this approach is appropriate for those as well.

Implications for research

Further research could take place in measuring performances of a system that would rely on different NoSQL solution suitable for handling binary files.
Ecology is nowadays very important field of research. Possibility for deployment of used

hardware as server instances for NoSQL, knowing that system is partition tolerant and can survive server termination is very interesting application of ecology in IS. It would be valuable to develop an architecture that would consist from both, commodity and used hardware. The first one would be used for storing more important data and used hardware could be used for storing data that were not very important or that were rarely used. Still, there is no risk of losing data because of partition tolerance feature of the system.

Implications for practice

The idea of expanding existing IS to an area of document management using distributed architecture is deployable not only in m-learning solutions, but in any system that handles binary files and similar content. Implementing solutions that includes NoSQL database for managing documents has significant implications for practitioners:

- Existing information systems can be extended using MongoDB database for handling unstructured data
- MongoDB is suitable for handling large amount of documents because it contains horizontal scaling possibilities
- Horizontal scaling and distributive characteristics enable the system to survive server termination
- Disaster recovery capabilities are present even with large amount of data
- Even old and used hardware could take place in a system like this because of partition tolerance feature
- Streaming video and audio content directly from the database system is present

Limitations

The study shows acceptable adoption of NoSQL technology inside particular SOA and any insight is limited to this context. When applied in different surroundings, the model could require adaptation.

References

Alier, M.F., Guerrero, M.J.C., Gonzales, M.A.C., Penalvo, F.J.G. and Severance, C. (2010), "Interoperability for LMS: The missing piece to become the common place for e-learning innovation", *International Journal of Knowledge and Learning*, Vol. 6 No.2, pp. 130-141.

Bull, S. and Reid, E. (2004), "Individualized revision material for use on a handheld computer", *Proceedings of mLearn 2003, London, UK, May 19-20, 2003*.

Chandhok, S. and Babbar, P. (2011), "M-learning in distance education libraries: A case scenario of Indira Gandhi National Open University", *Electronic Library, The*, Vol. 29 No. 5, pp. 637-650.

Casany, M.J., Alier, M., Mayol, E., Piguillem, J., Galanis, N., Garcia-Penalvo, F.J. and Conde, M.A. (2012), "Moodbile: A Framework to Integrate m-Learning Applications with the

LMS", *Journal of Research and Practice in Information Technology*, Vol. 44 No.2, pp. 129-149.

Elliott, J. (2012), "Those 'difficult conversations': how can technology help managers to learn how to hold them in an effective manner?", *Development and Learning in Organizations*, Vol. 26 No. 4, pp. 17-19.

Hellerstein, J. M., Stonebraker, M. and Hamilton, J. R. (2007), "Architecture of a database system", *Foundations and Trends in Databases*, Vol. 1 No. 2, pp. 141–259.

Kim, J. and Kizildag, M. (2011), "M-learning: next generation hotel training system", *Journal of Hospitality and Tourism Technology*, Vol. 2 No. 1, pp. 6-33.

Kehagias, D., Grivas, M., Mamalis, B. and Pantziou, G. (2006), "DYNER: a DYNamic cluster for education and research", *Campus-Wide Information Systems*, Vol. 23 No. 2, pp. 92-101.

Liu, Y., Han, S. and Li, H. (2010), "Understanding the factors driving m-learning adoption: a literature review", *Campus-Wide Information Systems*, Vol. 27 No. 4, pp. 210-226.

Naismith, L., Lonsdale, P., Vavoula, G. and Sharples, M. (2004), "Literature review in mobile technologies and learning", *FutureLab: Innovation in Education*, Scientific Report, available at: http://archive.futurelab.org.uk/resources/publications-reports-articles/literature-reviews/Literature-Review203 (accessed 4 October 2013).

Richardson, P., Dellaportas, S., Perera, L. and Richardson, B. (2013), "Students' perceptions on using iPods in accounting education: a mobile-learning experience", *Asian Review of Accounting*, Vol. 21 No. 1, pp. 4-26.

Sharples, M., Taylor, J. and Vavoula, G. (2005), "Towards a theory of mobile learning", *Proceedings of mLearn 2005, Cape Town, South Africa, October 25-28, 2005*, pp. 1-9.

Vavoula, G.N. and Sharples, M. (2002), "KLeOS: A personal, mobile, knowledge and learning organization system", *Proceedings of the IEEE International Workshop on Mobile and Wireless Technologies in Education (WMTE2002) Vaxjo, Sweden, August 29-30, 2002*, pp. 152-156.

Biography

Milorad P. Stević is a Database Architect with 15 years of experience (Progress versions 8,9; Oracle version 10g, MSSQL Server versions 2005, 2008, 2012), He is also a .NET Software Architect with a 8 years of experience in development for the Windows platform, Web, Web Services, Client/Server and n-tiered distributed applications, broad experience in all parts of project life cycle including the analysis, design, development, implementation testing, debugging/profiling and support. He is a full-time faculty member at The Higher Education Technical School of Professional Studies, Novi Sad, Serbia.