2015 Eighth International Conference on Internet Computing for Science and Engineering

SoSE Architecture of Data Intensive Computing for Healthcare Information System

Jun Ni, Ying Chen, Jie Sha, and Minghuan Zhang
[1]Big Data and Informatics Research Lab, Shanghai Sanda University
Shanghai, 201209, China
sageli@126.com

Abstract. **The architecture of healthcare information system (HIS) is important framework. The architecture implements the cutting-edge technologies of cloud computing and big data in parallel and distributed computing manors. The architecture scheme is proposed in SoSE-based systems engineering paradigm. The implementation of such architecture for healthcare data processing is given via multiple layers of big data implementations. This study is a primary work which provides basic insight into future intelligent system of HIS systems.**

Keywords: **healthcare information systems, HIS, big data, cloud computing, healthcare services, system architecture, system of systems, SoSE, data intensive computing, high performance computing, HPC, parallel computing.**

I. INTRODUCTION

1.1 Large-scale Data Intensive Computing Components

The concept of large-scale computing (massive computing) is not new coin. The term has been used throughout the whole domain of computations, especially parallel computing, supercomputing, and high performance computing. It usually refers to the scale of CPU computing capacity such as peta-scale computing exa-scale computing. It implicitly stands for how many processors employed and how to speed up computations with high scalability. However, the term of large-scale computing is also adopted in data intensive computing. For example, the conception of large-scale is applied to not only how many CPUs being used but also how large the datasets involved in single computation. It also includes multiple data storages participated in a single or large number of computing tasks to be accomplished using large number of computing capability and resources.

In general, a large-scale data intensive computing has five components of sub-systems. They described as follows.

High performance of computing systems equipment (as computer, calculation accelerated equipment, and data storage equipment, and high-speed data network passed and exchange equipment, and related auxiliary equipment).

High performance software system completed calculation task by needs (including system operation and management software, and application software).

High performance of network system and corresponding of facilities (including perimeter network system equipment, and calculation equipment of internal network configuration, and dynamic of resources monitoring , And deployment and management, and data passed of mechanism, and data entered and output).

High performance information management system (including information resources, and collection, and database save, and data structures information, and calculation method reserves, and built die simulation according to and parameter, and related of application calculation information and method reserves, as calculation method and means skills, and numerical calculation and calculation algorithm resources, and related calculation information and information, indicators).

Computational Expert is a HR system (taking care of professionals in relevant fields, calculations, professional technicians, management personnel and other. The traditional systems structure is illustrated in hierarchical manor, which is shown in Figure 1(a).

1.2 Concept of System of Systems

In order to aggregate all the all the HIS system components to work together, one needs to well understand each function of the system components (as we described above), and their inter-cooperative and interactions. It is analogic to the orchestra of all the “music instruments, each of which plays a special chord, rhythm, melody, harmony, and then structure, styles, and expressions. Due to the efforts of the music composer and conductor, and players, the orchestra’s theme is dynamically and expressively presented. Such assembling leads an integration of a concert system of systems. Pre-processing of the integration requires identify systems components inters of their attributes, functions, and behaviors, as

978-1-5090-0454-6/15 $31.00 © 2015 IEEE
DOI 10.1109/ICICSE.2015.37

well as their roles in the whole. That step is called systems engineering's pre-procedure. The second step is to synthesize or process (conduct) the systems as whole, in terms of the components' interconnection (mechanisms, and organism), interaction and interrelations, cooperation (synergy) and balances, and their roles in determining the whole systems' functions and features for the whole goal. It also leads us to consider the systems connections and interaction, as well as facility sharing, illustrated in Figure 1(b).

System is a set of interacting or interdependent components (elements) forming an integrated whole. System can be applied to different application domains. If the term is applied to form a set of engineering components, we usually call it an engineering system. If the term is applied to a biological entity, we refer it as a biological system, as such human body. In science and engineering, people often extract common abstract properties (characteristics) among various systems in terms of their matter, organization, behaviors, and principles which are independent of the specific domain, substance, type, or temporal scales of existence.

Most systems share common and abstractive characteristics, including system structures (defined by its components and their composition), behaviors (represented by inputs, processing, and outputs of transmitted media (including mater, energy, and information), interconnectivity which represents the structural interrelationships and interactions among system components (or elements), and operational functions to achieve system's goal.

A system also has its boundary. The system boundary wraps the system and classify the inner entities (system components or elements) from outer entities belong to surroundings. A close system boundary would not allow any information or data to be transmitted or transferred to the outside surroundings. That system is a close or isolated system. If the system boundary allows full or partial transmitting or transferring data to the surroundings, the system is open. An open system is a system which continuously interacts with its environment through its open boundary. The interaction can take the form of information, energy, or material transfers into or out of the system boundary. A concept of open system is contrasted with the one of an isolated system which exchanges neither energy, or matter, or information with its environment.

A system is composed of many system components such as out HIS subsystems or second layer sub-systems. These components can be basic elements or can be sub systems or componential systems which consist of their own element s or systems. The componential systems are called sub systems, while their parent becomes a super system. The componential elements or systems can be either homogeneous or heterogeneous to identify if the elements or sub systems are uniform or well-mixed. All the nature systems and most of the engineering systems contain heterogeneous components. These systems' components are various kinds in terms of their attributes, characteristics, behaviors, functions, and goals.

If a system consists of one or multiple subsystems, such super system is a large system which in many cases becomes a complex system. Based on this perspective view, a system is a super system, regarding to its "has-relationship" with its subsystems. This system can also be a regarded as a subsystem, associated with its "of-relationship" with its parent. Therefore, if the system is a super system or subsystem is based on its own properties and rules.

One cannot use systems classification to identify whether the system is natural or man-made or engineered. For example, as we all known, all the nature systems have subsystems, which also have their own subsystems. It forms an infinite hierarchic division, as long as matter can be classified into smaller entities. Based on this paradigm, an engineering system can also be infinitely classified into smaller matters, similar to biological classification. Therefore, the identification of nature or engineering systems is based on its self- and natural characteristics, behaviors, and goals. A natural system' components, for example, have their own naturally-grew, independent behaviors, functions, rules, and goals. If their attributes and functions are damaged, they can utilize the open system boundaries to self-adapt environmental changes, self-adjust their functions, or self-treat their damages, as well as self-control and govern their functions, and rules in order to sustain their survival until the life-cycle ended. However, many engineering systems do not have such properties and intelligent behaviors. Therefore, we believe natural systems are superior, complex and intelligent, even though look very simple. The human's great efforts are indent to develop engineering systems approach to the natural systems. That is the system engineering final destinations, although it has long way to go. That opinion implies the difference between the traditional systems engineering approach vs. today systems engineering methodology that is information-net-centric, intelligence-based, open systems architecting and integrations. That is the theme of system of systems engineering (SoSE). The concepts and principles in SoSE can be directly applied to our proposed integrated HIS super systems as we described.

II. SOSE BASED BIG DATA IN HIS

2.1 Hardware Architecture and System Design

As typical architecture of healthcare big data processing system can be sketched in Figure 2. In general, the architecture design of a computing system has two approaches. One is computation oriented architecture, and the other is a distributed architecture. Each has its characteristics. For example, centralized computing systems with data exchange speeds feature shared memory, relatively stable in technical operation, as well as easy to manage and maintain. Its obvious disadvantage is computational capacity is not strong to the cost, poor scalability, and lack of deployment of updated computing resource. A distributed computing system is precisely to make up for the shortcomings of centralized computing system. The system can integrate many computer processors and resources together to form a strong distributed-memory system. The system can easily expand to more servers, and easily composed of multi-core processor systems. Traditional computing systems cannot meet massive data processing. The integration of large-scale distributed computing resources for single computational task leads to the concept of grid computing, but only for intensive CPU-based computations in academia. On the other hand, industrials very often to deal with large amount of distributed data by also utilizing multiple computing resources. Based on their business profits, they come up with (1) services-based or utility computing for services charges, (2) technical architecture settings for distributed data intensive computing, and (3) supportive methodology to integrate resources behind, which leads to virtual technology. These three aspects of developments generate today's cloud computing with service operational modes (IaaS, PaaS, and SaaS), and business model, as well as technical treatment of big data using distributed computing methodology such as Hadoop/MapReduce, similar to HPC's parallel decomposition of computational tasks, but distributed-stored data -oriented tasks. The HPC's parallelism is for computing efficiency, while big data parallelism is for the capacity to data processing.

The development of a data intensive computing needs to consider both high-performance data-file storage system and high-performance computing system. Could computing technology in fact can be used to share the recourses to forming virtual large data and computing platform. This architecture requires a high-performance network crossing multiple platforms or servers. The multiple computational nodes within HPC subsystem can be dedicated to mapping and reducing tasks during processing management and mining.

For example, a HIS often distributes data over high-speed internet and integrate medical and health data for analysis. Hospitals and clinics can transmit data to a centralized system for data preprocessing, then to a high-performance computer for data processing and mining, and finally provide a timely feedback end users. The attributes of large amount of data, levels of timeliness, data types, data mining solutions, operational costs and benefits etc. should be considered when design and configure a system architecture, system maintenance for business suitability. Many big data computing solutions and system equipment can be migrated to form a platform for health data application services, equipment acquisition service, and applications services in today's healthcare environment.

Depending on each application scenario, the architecture of data intensive computing system may be designed differently. If the CPU computation is dominated, while data is in single format or in a straight steaming, it is better to design a HPC-based, data processing system architecture, shown in Figure 3.

In our research lab, since the major tasks are large sets of data files with minor intensive computation, we choose to design big-data based system. In order to fully use our available computing resources for both research and teaching, we also request to install and implement IBM-based could computing configuration with multiple cluster management systems.

In our experimental study, we made substantive changes to different system architectures. The main schema modifications are due three subsystems: data storage subsystems, high performance computing subsystem, and supportive database server subsystems. In our data storage subsystems, we adopted disk arrays (Redundant Arrays of Independent Disks, RAID) data storage, using different database software to support different data storage. We designed two data storage system, one is internal but with small data capability and external one which can host large dataset. An Ethernet data converter is used to connect external systems, while InfiniBand switch is used to link to all the internal computational odes within the HPC subsystem. Main objective is to improve efficiency, strengthen the effectiveness calculations, handling bulk data. Supporting server subsystem is used to participate in different types of calculations, including network servers, network data sharing, security gateway, network, data delivery, special peripheral data acquisition, data processing, data pre-processing, data visualization, data backup and GPU processing.

22 Architecture Configuration of Cloud-based Data Processing System with Big Data Configuration

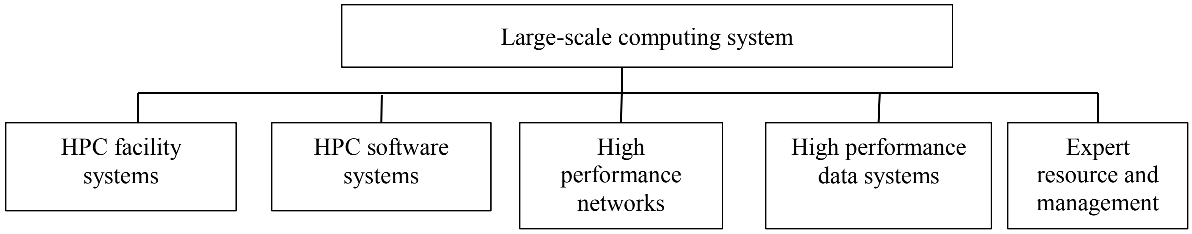
Since the two cores in today's big data technology are Hadoop/MapReduce and HDFS, respectively. Since MapReduce and Hadoop were first developed and released as open sources, they have been widely used in many big data applications. The popularity of using has grown with the development of additional tools such as HBase, Pig, Hive, etc. At present there is no uniform system of technical structures and common technical standards. Liu recently presents a Hadoop-based computing architecture for data processing, shown in Figure 4. This architecture has its universality, especially can be adapted to healthcare information system. This framework addresses the characteristics of distributed computing systems, like a grid computing system architectures, using layers, and specific methodologies mapped to the corresponding application fields. For example, in the data storage layer, one can use HBase and HCatalog technologies, which help to handle NoSQL data. We address the bottom-up layers in as follows:

Data Resource Layer: The bottom layer lies on various data resources including healthcare application data, private data of patients, families, and residents. These data are usually stored in the enterprise databases. The resources also include logs data to the systems. The types of data and their data structure are very diverse. The data characteristics are various in both data format and data structure. Some of data are unstructured, semi-structured data. The log data in both text formats and rich media formats are from Web or internet returns. Some of these data are stored centralized while others are stored directly in distributed file system (DFS), such as formatted log files. Some data can be parsed directly by MapReduce programs, while others not. But when it comes to deal with traditional healthcare applications, one can save the data in the primary database, such as historical data from operating system, since they are not stored in HDFS. MapReduce programs are required to access them through external API in the acceptable form of the data. This approach is neither flexible nor efficient, and Hadoop Framework is introduced as a data integration layer. Typical instances of using data integration layer like Sqoop tool are to import data that is stored in relational databases to Hadoop/MapReduce enabled format for subsequent handling procedures using Hive tool. The other approach is to directly import the data into HBase, which support exports access data stored in a relational database.

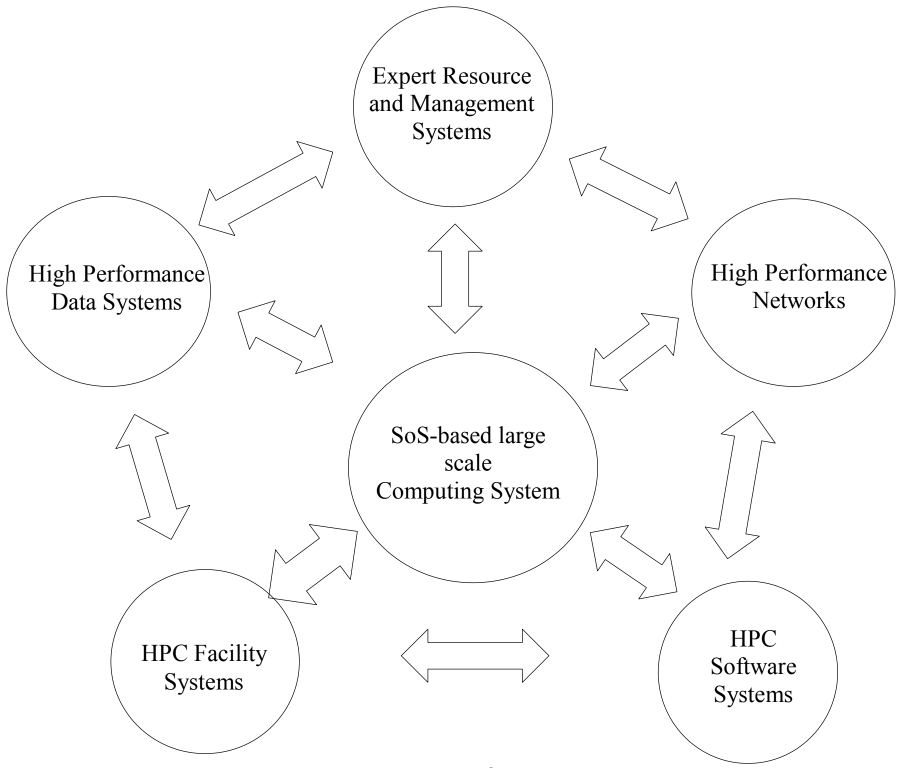
File Storage Layer: file storage layer a distributed file system. The system underlies a number of distributed files at different locations via a network connection. Through a unified interface, the system provides an object-level file access to application services. File storage application layer deals with the top shield storage device types and models, the interface protocols, and other technical details such as storage location. The system provides data backup, fault tolerance, and condition monitoring, security and other guarantees reliable document accessible to different service management units. Meanwhile, the distributed parallel files can be stored in a data cloud. The data processing environment should support efficient parallel access to huge amounts of large files. In this overall architecture, file storage layer can handle specific integration by connecting the desired data at remote data storage resources. HDFS is a core component of the file storage layer.

Data Storage Layer. It is responsible to provide distributed, scalable data storage and management capabilities. Different from traditional relational databases, cloud-based data storage do not require the full SQL support. The feature of its low cost management and quick reads and writes of the data offers two tiers of storage technology foundations, HBase and HCatalog. HBase implements a distributed, column-oriented database storage system, while HCatalog is for table-based data management.

The Programming Model Layer. It is reasonable for coding MapReduce-based tasks decomposition, data access and programming of protocols to access distributed files. It touches the base of algorithms used in big data, which depends on targeted applications.



(a) Traditional hierarchical framework of large-scale



(b) SOS-based perspective of systems and their relationships

Figure 1. SOS-based large-scale, high performance and data intensive computing

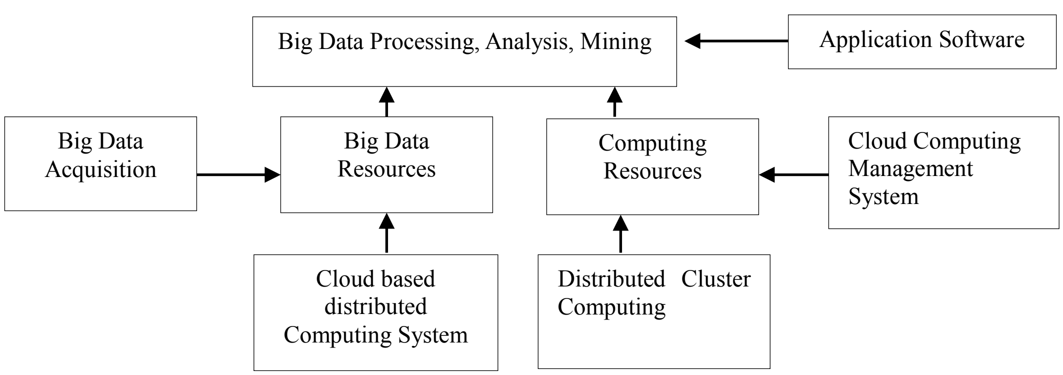
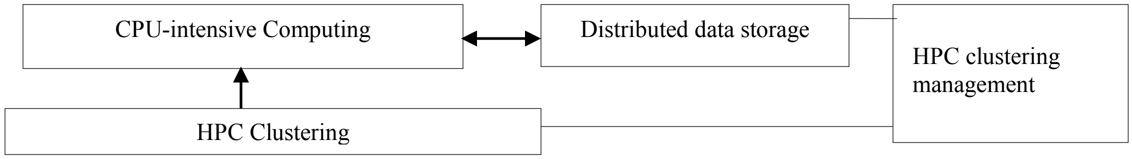
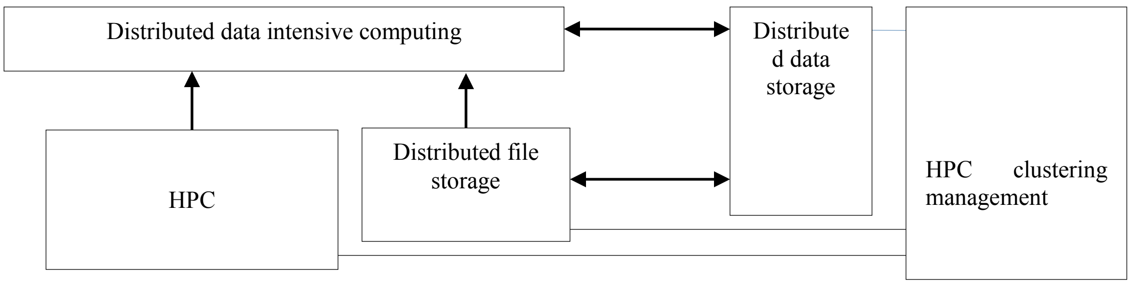


Figure 2. Architecture of Big data Processing System



(a) HPC-based data processing system architecture



(c) Distributed file-storage-based data computing architecture (big data)

Figure 3. Various architectures for data intensive computing system

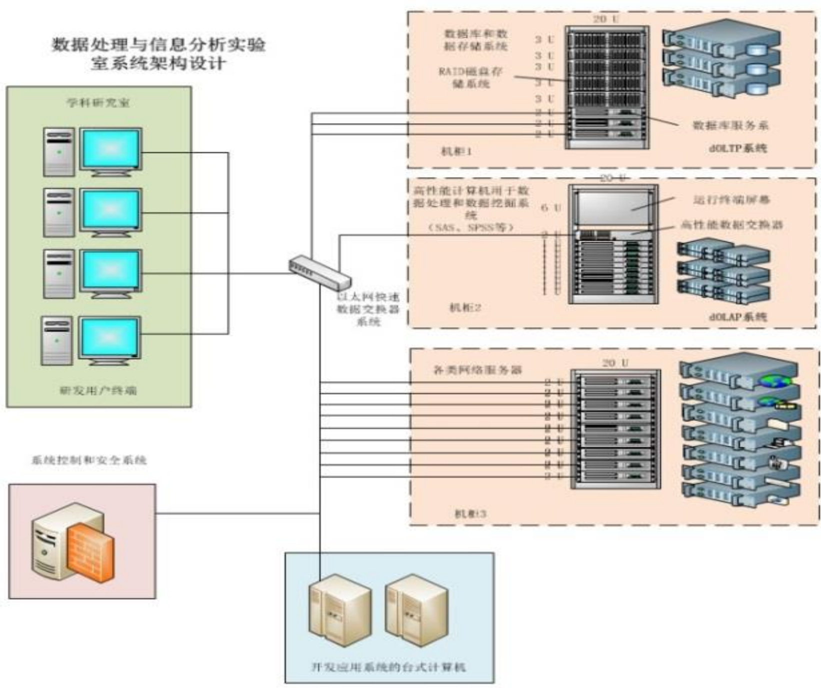


Figure 4. Big-data and Cloud combined configured data intensive computing system design and implementation.

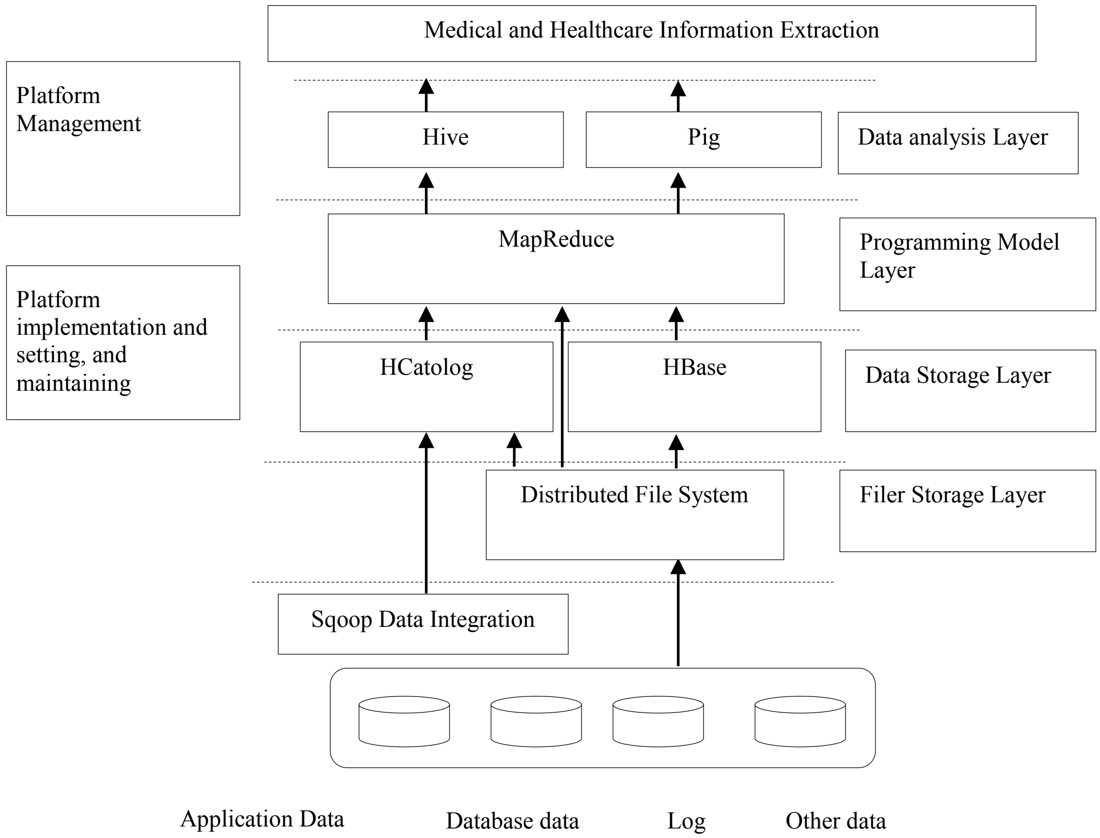


Figure 5 Architecture of cloud computing based healthcare data processing system

Data Analysis Layer. It enables to conduct data analysis with attentions on data processing and analysis models, business value, and information extraction, rather than supportive layers. It provides some advanced analysis tools for data analysts. Hadoop Pig and Hive are two major analysis tools. Pig provides a MapReduce abstracts based on a higher level of data-processing capability, including a data manipulation language and operating environment. Hive is the structured data that can be mapped to a data table for data analysts, such as providing a complete SQL query, and a query language to during MapReduce task execution.

Platform Management Component one the left-hand-side is the supportive system. It enables the smooth and safe operation for entire data processing. With other systems management components such as cloud, clustering, and monitoring etc., The PMC component handles system configuration and resources management, monitoring status and handling fault, performance tuning, ensuring security etc.

In the above presented four-level framework, according the addressed HIS environments and applications environment, appropriate amendments and local changes may be made. Each system and subsystems should maintain its independent in self-operation, self-adjusting, and self-governing. The data transferring and communication should be designed with open interface configuration with international standards such as DICOM, HL7, and other IT standards. Depending on special applications, developers can insert suitable layers as middleware settings. People can also develop special interface or plugins for deployment of remarkable enterprise data analysis and statistics software. In order to operate the system, some special design and consideration should be made in high performance networking. In computational model, people may add special interface to allow parallel computing during data intensive computing. Such combined computational models will be very useful in dealing with medical and healthcare applications, such as image processing like registration, classification and reconstructions etc. in clinic or hospital environments.

III. CONCLUSION

There exist huge demands of developing and constructing integrated healthcare information system which includes today large-scale computing for big data. Today's HIS system components should match with healthcare services. Each HIS subsystem is designed to map the corresponding services. The architectures of various data intensive computing are addressed with emphasis on targeted data amount, structure, and data storage. The cloud and big data enhanced architecture is proposed in a system engineering paradigm. The SoSE based architecture for healthcare data processing is designed via multiple layers with considerations. This study is a primary work which provides basic insight into future intelligent system of HIS systems.

Acknowledgement

This research is sponsored by Shanghai key research project at non-government-run university, Shanghai Municipal Education Commission, Research and Education fund, funding number, 2012-SHNGE-01ZD. Thanks must go to Ms. Xiaoming Bu, the Director of Research and Development, for her persistent help and support in pursuing this research grant and managing project throughout.

Reference

[1] Ananth Grama, George Karypis, Vipin Kumar, Anshul Gupta, Introduction to Parallel Computing (2nd Edition), ISBN 978-0201648652, 2003.

[2] Andrew Ryan, *Under the Hood: Hadoop Distributed file system reliability with Namenode and Avatarnode*, Facebook Blog, 2012.6.

[3] Apache Solr, http://lucene.apache.org/solr/.

[4] Banerjee J., R. I. Baum, and D. K. Hsiao, "Concepts and Capabilities of a Database Computer, " *ACM Transactions on Database Systems*, Volume 3 Issue 4, 1978.12.

[5] Baum, D.K. Hsiao, "Database Computers-A Step Toward Data Utilities," *IEEE Transactions on Computer*, Vol. C-25, No. 12, Pages 1254-1259, 1976.12.

[6] Behrooz Parhami, "A Highly Parallel Computing System for Information Retrieval," in *AFIPS Proceedings of the Fall Joint Computer Conference*, Pages 587-595, 1972.

[7] D. R. Brownbridge, L. F. Marshall, and B. Randell, "the Newcastle Connection," Software Practice and Experience12: 1147–1162, 1982.

[8] D. R. McGregor, R. G. Thomson, and W. N. Dawson, "High Performance Hardware for Database Systems," *Systems for Large Data Bases*, Pages 103-116, 1976.

[9] D.L. Slotnick, Logic per Track Devices, Advances in Computers, Vol. 10, Academic Press New York, pp. 291-296, 1970.

[10] D.L. Slotnick, Logicper, "Track Devices," *Advances in Computers*, Vol. 10, Frantz Alt, Ed., Academic Press, New York, pp. 291-296, 1970

[11] David J. DeWitt, "DIRECT –A Multiprocessor Organization for Supporting Relational Database Management Systems," in *Proceedings of the 5th annual symposium on Computer Architecture*, pp. 182 –189, 1978.

[12] David J. DeWitt, Paula B. Hawthorn, "A Performance Evaluation of Data Base Machine Architectures," in *Proceedings of the seventh international conference on Very Large Data Bases*, Volume 7, pp. 199 –214, 1981.

[13] E. A. Ozkarahan, S. A Schuster, and K. C. Smith, "RAP - associative processor for database management," in *AFIPS Conference Proc.*, Vol. 44, pp. 379-388, 1975.

[14] E. Babb, "Implementing a Relational Database by Means of Specialized Hardware," *ACM Transactions on Database Systems*, Volume 4 Issue 1, pp. 1-29, 1979.3.

[15] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," *Communications of the ACM -Special 25th Anniversary Issue*, Volume 26 Issue 1, pp. 64–69, 1983.1.

[16] Fay Chang, Jeffrey Dean, etc., "Bigtable: A Distributed Storage System for Structured Data," in *Proceeding of the Seventh Symposium on Operating System Design and Implementation*, 2006.11.

[17] Gene M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of AFIPS Conference*, Volume 30, pp. 483-485, 1967.4.

[18] Hans-Otto Leilich, Günther Stiege, Hans Christoph Zeidler, A Search Processor for Data Base Management Systems, Proceedings of the fourth international conference on Very Large Data Bases, Volume 4, Pages 280-287, 1978.

[19] High Performance Fortran Forum, http://hpff.rice.edu/index.htm.

[20] Ian Foster, "Designing and Building Parallel Programs," ISBN 978-0201575941, 1995.

[21] IBM DB2, http://www-01.ibm.com/software/data/db2/.

[22] International Data Corporation (IDC), the Expanding Digital Universe, 2007.3.

[23] ISO/IEC 9075-1:2011, Information technology-Database languages–SQL-Part 1, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=53681.

[24] J. McCarthy, R. Brayton, D. Edwards, etc., "LISP I Programmers Manual, Computation Center and Research Laboratory of Electronics," Massachusetts Institute of Technology, 1960.3.

[25] J. R. Goodman, "An Investigation of Multiprocessor Structure and Algorithms for Data Base Management," *Electronics Research Laboratory Memorandum* No. UCB/ERL M81/33, University of California, Berkeley, 1981.5.

[26] J. Satran, K. Meth, etc., Internet Small Computer Systems Interface (iSCSI), RFC3720.

[27] Jack Dongarra, Ian Foster, and Geoffrey C. Fox, etc., *The Sourcebook of Parallel Computing*, ISBN 978-1558608719, 2002.

[28] James L. Parker, "A Logic-Per-Track Retrieval System," *IFIP Congress*, Volume 1, Foundations and Systems, pp. 711-716, 1971.

[29] Jamshidi, M. 2005, Theme of the IEEE SMC 2005, Waikoloa, Hawaii, USA, http://ieeesmc2005.unm.edu/, October 2005

[30] Jeffrey Dean, Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in *Sixth Symposium on Operating System Design and Implementation*, 2004.12.

[31] Jim Gray, "Data Management: Past, Present, and Future," *IEEE Computer* 29(10): 38-46, 1996.

[32] John Darlington, Yi-ke Guo, Hing Wing To, Jin Yang, "Parallel skeletons for structured composition," in *Proceedings of the Fifth ACM SIGPLAN symposium on Principles and practice of parallel programming*, Pages 19–28, 1995.

[33] Krishnamurthi Kannan, "The Design of a Mass Memory for a Database Computer," in *Proceedings of the 5th Annual Symposium on Computer Architecture*, Pages 44-51,

1978.
[34] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, “the PageRank Citation Ranking: Bringing Order to the Web,” *Technical Report*. Stanford Info Lab, 1999.
[35] M. Missikoff, “An Overview of the Project DBMAC for a Relational Machine,” in Proceedings of the 6thWorkshop on Computer Architecture for Non-Numeric Processing, 1981.6.
[36] Message Passing Interface (MPI), http://www.mcs.anl.gov/research/projects/mpi/.
[37] Message Passing Interface Forum, http://www.mpi-forum.org/.
[38] Mike Burrows, “the Chubby Lock Service for Loosely-Coupled Distributed Systems,” in *the Seventh Symposium on Operating System Design and Implementation*, 2006.11.
[39] Mo Jamshidi, *Systems of Systems Engineering: Innovations for the 21st Century*, 2009, Wiley
[40] N. Minsky, “Rotating Storage Devices as Partially Associative Memories,” in AFIPS *Proceedings of the Fall Joint Computer Conference*, Pages 587-595, 1972.
[41] NDFS, http://wiki.apache.org/nutch/NutchDistributedFileSystem/.
[42] Oracle, Oracle Data Integrator
[43] Oracle: http://www.oracle.com/technetwork/middleware/data-integrator/overview/index.html.
[44] Parallel Virtual Machine, http://www.csm.ornl.gov/pvm/pvm_home.html.
[45] R. Shultz, “A Multiprocessor Computer Architecture for Database Support,” *PhD Dissertation*, Computer Sciences Department, Iowa State University, 1981.8.
[46] Rémy Card, Theodore Tso, Stephen Tweedie, “Design and Implementation of the Second Extended File system,” in *Proceedings of the First Dutch International Symposium on Linux*, 1994.12.
[47] Rob Pike, Sean Dorward, Robert Griesemer, and Sean Quinlan, “Interpreting the Data: Parallel Analysis with Sawzall,” *Scientific Programming Journal-Dynamic Grids and Worldwide Computing*, Volume 13 Issue 4, Pages277-298, 2005.10.
[48] S. A. Schuster, H. B. Nguyen, E. A. Ozkarahan, and K. C. Smith , “RAP.2 –An Associative Processor for Databases and its Applications,” in *IEEE Transactions on Computers*, Volume 28 Issue 6, Pages 446-458, 1979.6.
[49] S. C. Lin, D. C. P. Smith, and J. M. Smith, “The Design of a Rotating Associative Memory for Relational Database Applications,” *Transactions on Database Systems*, Volume 1 Issue 1, Pages 53-75, 1976.3.
[50] S. Y. W. Su, G.J. Lipovski, “CASSM: a cellular system for very large data bases,” in *Proceedings of the VLDB Conference*, Pages 456-472, 1975.
[51] SAGE, http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/sage/.
[52] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, “the Google File System,” in 19th ACM Symposium on Operating Systems Principles, 2003.10.
[53] Stuart E. Madnick, “the Infoplex Database Computer: Concepts and Directions,” in *Proceedings of the IEEE Computer Conference*, 1979.2.
[54] The Big Data Revolution in Healthcare, Report, Mckingsey Company
[55] The Open MPAPI specification for parallel programming, http://openmp.org/wp/.
[56] Ushi Inoue, Haruo Hayami, Hldeki Fukuoka, and Kenji Suzuki, Rinda, “A Relational Data-base Processor for Non-indexed Queries,” in *International Symposium on Database Systems for Advanced Applications*, Pages 382-386, 1989.4.
[57] W. Hell, “RDBM –A Relational Data Base Machine: Architecture and Hardware Design,” in *Proceedings of the 6thWorkshop on Computer Architecture for Non-Numeric Processing*, 1981.6.
[58] Zoie, http://code.google.com/p/zoie/.