Abyss Storage Cluster 기반 DataLake Framework의 Connected Data Architecture 개념 설계 및 검증

(Design and Verification of Connected Data Architecture Concept employing DataLake Framework over Abyss Storage Cluster)

차병래*,**, 차윤석**, 박선*, 신병춘***, 김종원*

(ByungRae Cha, Sun Park, Byeong-Chun Shin and JongWon Kim)

요 약

조직과 기업의 비즈니스 시스템의 규모가 커지면서 다양한 대량의 데이터들이 생성되는 비즈니스 환경의 변화와 데이터를 보다 스마트하게 처리하여 효율성을 높일 수 있는 방법으로 DataLake와 같이 단일 도메인 모델이 필요한 상황이다. 특히, 자원의 유한성과 공유 경제에 의한 물리적인 분할된 멀티 사이트의 데이터를 논리적인 단일 도메인 모델을 만드는 것은 컴퓨팅 자원의 효율적 운영 측면에서 매우 중요하다. 기존의 Data Lake 프레임워크의 장점을 기반으로 다양한 응용 영역의 멀티 사이트들을 통합 및 데이터 라이프 사이클을 관리하기 위한 Abyss Storage 기반 DataLake 프레임워크의 Connected Data Architecture 개념 (connected data architecture-concept)과 기능들을 정의하고, Connected Data Architecture 개념을 위한 인터페이스 설계 및 인터페이스(Interface) #2와 #3의 유효성 검증을 수행한다.

■ 중심어 : 데이터레이크 프레임워크; Abyss Storage Cluster; CDA-Concept (connected data architecture-concept)

Abstract

With many types of data generated in the shift of business environment as a result of growth of an organization or enterprise, there is a need to improve the data-processing efficiency in smarter means with a single domain model such as Data Lake. In particular, creating a logical single domain model from physical partitioned multi-site data by the finite resources of nature and shared economy is very important in terms of efficient operation of computing resources. Based on the advantages of the existing Data Lake framework, we define the CDA-Concept (connected data architecture concept) and functions of Data Lake Framework over Abyss Storage for integrating multiple sites in various application domains and managing the data lifecycle. Also, it performs the interface design and validation verification for Interface #2 & #3 of the connected data architecture-concept.

■ keywords : DataLake Framework; Abyss Storage Cluster; CDA-Concept (connected data architecture-concept)

Ⅰ. 서 론

데이터는 기업 또는 조직 운영의 여러 측면에서 중추적인 역할을 담당하면서 많은 기업과 조직들에게 중요성이 높아지고 있으며, 점차 기업과 조직의 가치는 데이터 중심으로 변화하고 있다. 기업에서는 초창기에 데이터 저장소(Repository)에 자료를 저장 및 사용하는 단순한 방법으로 이용하였다. 좀 더 진보한 형태로, 사용자의 의사 결정에 도움을 주기 위하여, 조직 또는 기업 시스템의 데이터베이스에 축적된 데이터를 공통의 형식으로 변환(Transformation)해서 관리하는 Data Warehouse를 이용하고 있다. 오늘날 우리는 빅데이터(Big Data)를 활용하는 사용 케이스(Use Case)들이 증가하고 있으며, 기업 또는 조직에서 처리되는 데이터들도 다양한 형태로

* 정회원, 광주과학기술원 전기전자컴퓨터공학부
** 정회원, 제노테크(주)
*** 정회원, 전남대학교 수학과
※ 본 연구는 과학기술정보통신부와 정보통신산업진흥원 및 광주정보문화산업진흥원의 “에너지신산업 SW융합클러스터조성사업(R&D, ITAS10101701100 70001000200200)”으로 수행된 연구결과입니다.
※ This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.R7117-16-0218, Development of Automated SaaS Compatibility Techniques over Hybrid/Multisite Clouds).

접수일자 : 2018년 09월 06일
수정일자 : 2018년 09월 19일
게재확정일 : 2019년 09월 27일
교신저자 : 김종원, e-mail : jongwon@gist.ac.kr

대량화되어짐에 따라서 이를 좀 더 효율적으로 처리할 수 있는 데이터 레이크(Data Lake)가 점차적으로 주목을 받고 있다.

기업 또는 조직은 비즈니스 시스템들 전반에 걸쳐 방대한 양의 데이터를 생성하며, 규모가 커지면서 서로 다른 시스템들에 존재하는 데이터를 보다 스마트하게 처리하기를 원하고 있다. 이를 위한 가장 기본적인 접근 방법 중 하나는 정보를 Enterprise Data화하여 데이터를 정확하게 표현하고 전체 비즈니스에 대한 가장 중요한 데이터를 설명할 수 있는 단일 도메인 모델(Single Domain Model)을 구축하는 것이다. 이러한 일반적인 Data Lake 모델들은 데이터를 캡처링(Capturing), 처리(Processing), 분석(Analyzing), 그리고 사용자들(Users) 또는 데이터를 소비하는 시스템들(Consuming systems)에 제공할 수 있어야 하며, 이를 위해서는 전사적 데이터 레이크(Enterprise-wide Data Lake)를 구축해야 하며, 다음의 7 가지 항목들을 고려해야 한다[1]:

① Data Governance & Data Lineage
② 비즈니스 인텔리전스(Business Intelligence)를 유도하기 위한 인공 지능(AI) 및 기계 학습의 적용
③ Predictive Analysis
④ Information Traceability & Consistency
⑤ Historical Analysis to derive Dimensional Data
⑥ 다양한 엔터프라이즈(Enterprise)의 데이터 전송을 위한 중앙 집중화된 데이터 근원 기반의 최적화된 데이터 서비스 제공
⑦ 빅데이터의 Batch 및 Streaming 처리를 위한 Lambda Architecture의 채용

Data Lake의 장점을 얻기 위해서는 위에 기술된 7가지의 Data Lake 요구 기능들을 어떤 구조로, 어떻게 동작 할 것인지에 대한 Data Lake의 구성 요소들을 정의하는 게 매우 중요하며[2], 추가적으로 Data Lake의 확장성 및 호환성 측면에서의 고려가 필요하다. 이와 관련하여, Connected Data Architecture 개념 (Connected Data Architecture-Concept, CDA-Concept)은 Hortonworks의 Mark Harring의 인터넷 기고(起稿)문에서 찾을 수 있으며, Data Lake 개념은 선구자인 데이터 웨어하우스의 진화의 결과이며, 더 나아가서 서로 연결된 Data Pools 또는 Connected Data Architectures에 대한 사고가 필요하다고 하였다[3, 4].

본 논문에서는 기존 Abyss Storage 기반의 DataLake 프레임워크의 장점을 기반으로 다양한 응용 영역의 멀티 사이트들을 통합 및 데이터 라이프 사이클을 관리하기 위한 Connected Data Architecture-Concept의 요구 사항을 분석하고, CDA-Concept을 위한 인터페이스 설계하며, DataLake와 퍼블릭 스토리지 간의 인터페이스 #2, 그리고 DataLake와 micro-Storage 간의 인터페이스 #3의 실효성 검증을 수행하였다. 인터페이스 #1 의 DataLake와 Data Lake 간의 유효성 검증은 단지 데이터 클러스터 복사와 같은 효과로 특별한 의미를 부여하지 않았다.

Ⅱ. 관련 연구

1. Abyss Storage Cluster와 DataLake Framework

SMB(Small and Medium-sized Business)를 위한 대용량 Abyss Storage Cluster는 [그림 1]과 같이 구성되었으며, 실제적으로 Abyss Storage Cluster의 H/W 프로토타입 개발이 완료됨과 동시에 제품의 양산이 가능한 상태이다. 또한 Abyss Storage의 성능 향상을 위하여 스토리지의 다양한 디스크 매체별 성능 테스트와 스토리지의 내부 네트워크의 가속화를 위한 네트워크 본딩(Bonding), 그리고 KOREN(Korea Advanced Research Network) 네트워크를 이용한 국내 및 국외 네트워크 트래픽 테스트를 완료한 상태이다[5, 6].

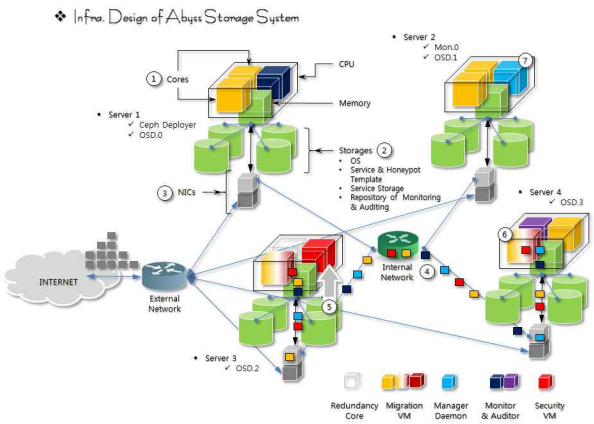

그림 1. Abyss Storage Cluster의 개념도

Abyss Storage Cluster 기반의 DataLake Framework는 전사적 데이터 레이크(Enterprise-wide Data Lake)를 구축하기 위한 다양한 요구 조건들을 고려하여 [그림 2]와 같이 정의 및 구성하였다[2].

DataLake Framework의 최 하단에 위치한 Physical Layer는 DataLake의 논리적 기능들을 지원하기 위한 실질적인 물리적인 자원들인 컴퓨팅, 스토리지, 그리고 네트워킹 등으로 구성되며, 본 연구에서는 Abyss Storage Cluster가 물리적인 자원을 지원하는 역할을 수행하게 된다. Physical Layer의 위에 존재하는 Distributed Storage Layer는 들어오는 이벤트

(Event) 및 데이터 스트림(Data Stream)에 대한 람다 아키텍처[7]의 전반적인 반응성(Reactivity)을 정의하며, Distributed Storage Layer가 충분히 빠르지 않은 경우, 람다 아키텍처(Lambda Architecture)의 Stream Layer의 동작이 둔해서 람다 아키텍처의 준 실시간(Near Real-time) 특성을 반영할 수 없게 된다.

DataLake의 Security Layer는 각 계층의 구성 요소들 간의 인증(Authentication), 권한(Authorization), 네트워크 고립(Network Isolation), 데이터 보호(Data Protection), 그리고 감사/진단(Auditing/Diagnose) 등의 다양한 보안 기능들이 제공되어야 한다. DataLake의 Data Acquisition Layer의 주요한 역할 중 하나는 데이터를 DataLake에서 추가 처리 할 수 있는 메시지로 변환 기능과 다양한 스키마 사양을 수용할 수 있는 유연성(Flexibility)이며, 동시에 모든 변환된 데이터 메시지를 DataLake에 원활하게 푸시(Push) 할 수 있는 빠른 연결 메커니즘을 제공해야 한다.

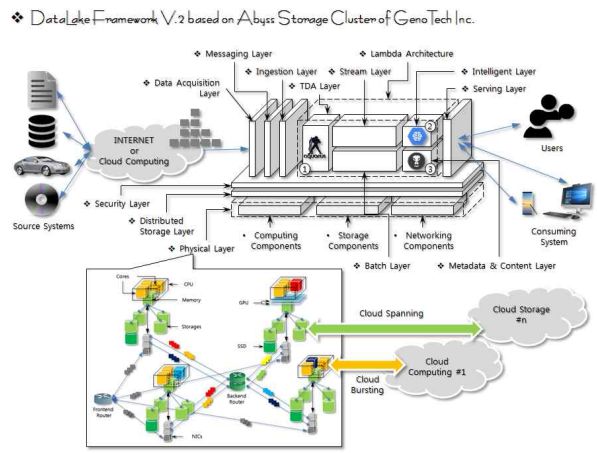

그림 2. DataLake Framework와 Abyss Storage Cluster 기반의 Cloud Bursting/Spanning

DataLake의 Messaging Layer는 Data Acquisition Layer에 연결된 외부와의 멀티 연결들을 분리하는 계층임과 동시에 메시지의 지속성(persistent)에 의한 메시지 전달을 보장하며, 메시지의 지속성의 특성은 대개 스토리지 매체에서 지원하게 된다. DataLake의 Ingestion Layer는 람다 아키텍처를 위한 매우 중요한 계층이며, 이 계층은 람다 아키텍처의 작업 모델로 데이터를 전달하는 속도 등을 제어할 수 있다. 람다 아키텍처는 [그림 3]과 같이 일괄 데이터 처리와 실시간 데이터 처리의 병합 문제를 해결할 수 있으며, 고성능의 분산 컴퓨팅 자원과 확장성으로 대규모 데이터 세트를 일괄적으로 거의 준 실시간 처리를 통한 일관된 데이터를 제공한다. 특히 람다 아키텍처는 저 지연(Low Latency)에 의한 엔터프라이즈 데이터의 다양한 데이터로드 프로파일(Data Load Profile)을 통한 Scale-Out 아키텍처를 구현 방법을 정의하고 있다. 본 연구에서 제안한 람다 아키텍처의 내부에는 DataLake Framework의 주요한 특성을 포함하고 있으며, TDA(Topology Data Analysis), Batch, Stream, Intelligent, 그리고 Meta-data & Content 등의 Layer들로 구성된다. 람다 아키텍처의 TDA Layer는 Topology data analysis[8]를 통한 원시 데이터를 모델링된 데이터(Modeled Data)로 변환하는 기능이 이 계층의 주요한 임무임과 동시에 다른 Data Lake 모델들과의 차별화된 핵심 기능이며, 모델링된 데이터는 람다 아키텍처의 Serving Layer에 의한 제공 가능한 데이터 모델을 의미한다. 람다 아키텍처의 Intelligent Layer는 고품질의 모델링된 데이터를 생성하기 위해 수집된 원시 데이터를 기반으로 기계 학습(Machine Learning) 및 데이터 과학 처리를 Batch & Stream Layer에서 지원하게 된다. 또한 고품질의 모델링된 데이터를 Distributed Storage Layer에 저장되며, 사용자들과 소비 시스템(Consumer System)을 위한 Service Layer에 제공하게 된다. 람다 아키텍처의 Meta-data & Content Layer는 DataLake 운영을 위한 생성된 메타데이터(Meta-data)의 저장/관리/검색 기능 등의 메타 데이터 관리 기능을 수행하며, DataLake를 구성하는 여러 Layer들의 다양한 역할을 수행할 S/W들을 가상화(Virtualization) 기술에 의한 CI/CD(Continuous Integration/Continuous Deployment) 기능을 제공하게 되며, 이러한 기능은 다른 Data Lake 모델들과의 차별화된 기능을 제공하게 된다. 그리고 람다 아키텍처의 Meta-data & Content Layer에서 제공되는 Cloud Bursting[9]과 Cloud Spanning[10]은 애플리케이션 작업 부하에 대한 원활한 운영과 컴퓨팅과 스토리지의 고확장성을 보장하도록 설계되었다.

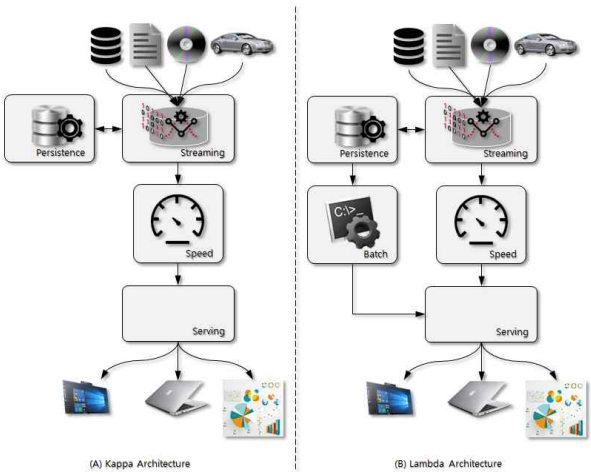

그림 3. 카파 아키텍처와 람다 아키텍처의 비교

DataLake의 Serving Layer는 DataLake로부터 데이터를 제

공하는 주요 책임을 가지며, 특히 람다 아키텍처에서 생성된 데이터를 사용자 또는 소비하는 애플리케이션에게 어떻게 전달 및 제공하는가가 매우 중요하게 된다. 데이터는 다양한 시스템들 간의 다중 방법으로 전송이 가능하며, 데이터 전송을 위한 일반적인 방법은 서비스(Service)를 이용하는 방법이며, 이러한 서비스를 데이터 서비스(Data Services)라고 한다.

2. Overloading/Overriding & OTA

Connected Data 개념을 물리적인 Abyss Storage Cluster 기반의 Connected DataLake로 확장 및 세부적인 서술을 위하여 Overloading/Overriding 그리고 OTA(Over-the-Air)에 대한 개념들을 간략하게 기술한다. 자바와 같은 객체지향 프로그래밍 언어에서 다형성을 지원하는 방법으로 메소드 오버로딩(Overloading) 오버라이딩(Overriding)이 있다. 오버로딩은 같은 이름의 메소드를 여러 개 가지면서 매개변수의 유형(Type)과 개수가 다르도록 하는 기술이며, 오버라이딩은 상속관계에 있는 상위 클래스가 가지고 있는 메소드를 하위 클래스가 재정의 해서 사용한다. 또한, OMA (Open Mobile Alliance)의 Download OTA는 유/무선 네트워크를 통한 임의의 형식과 크기의 미디어 객체(Media Object)들을 다운로딩하기위한 유연성한 메커니즘을 제공한다[11, 12]. 더불어, 최근 OTA 기술은 모바일과 Connected Car 분야에서는 세부적으로 FOTA (Firmware Over-the-Air)와 SOTA (Software Over-the-Air)로 구분되며, 네트워크 장비 분야에서는 Zero-Touch Provisioning 기술 등이 각광을 받고 있다.

III. Abyss Storage Cluster 기반 DataLake Framework의 CDA 개념 정의 및 설계

1. Abyss Storage Cluster 기반 DataLake Framework의 CDA 개념의 역할 분석

DataLake Framework의 CDA(Connected Data Architecture) 개념의 역할 및 필요성으로 단일 도메인 모델을 구축하는 것이며, 이러한 일반적인 Data Lake 모델들은 데이터를 캡처링, 처리, 분석, 그리고 사용자들 또는 데이터를 소비하는 시스템들에 제공할 수 있는 전사적 데이터 레이크를 구축에 고려해야 하는 2 가지 항목으로 “④ Information Traceability & Consistency“과 “⑥ 다양한 엔터프라이즈의 데이터 전송을 위한 중앙 집중화된 데이터 근원 기반의 최적화된 데이터 서비스 제공“이다. 특히, Abyss Storage 기반 DataLake의 Connected Data 개념의 초기 모델로 다양한 데이터 전송을 위한 데이터 전송 서비스에 초점을 맞추고 설계하고자 한다.

2. Abyss Storage Cluster 기반 DataLake Framework의 CDA 개념 정의 및 설계

Abyss Storage Cluster 기반 DataLake Framework의 Connected Data Architecture 개념(CDA-Concept)은 물리적으로 분할된 다양한 스토리지들(Cloud Storage, Data Center, micro-Storage 등)을 소프트웨어에 의한 논리적으로 묶을 수 있다는 개념이며 추상적으로 인터페이스라 명명하며, 구현 측면에서는 [그림 4]와 같이 3가지 형태의 인터페이스를 정의 및 설계한다.

- DataLake와 Data Lake 간의 Interface #1
- DataLake와 Cloud Storage 간의 Interface #2
- DataLake와 micro-Storage 간의 Interface #3

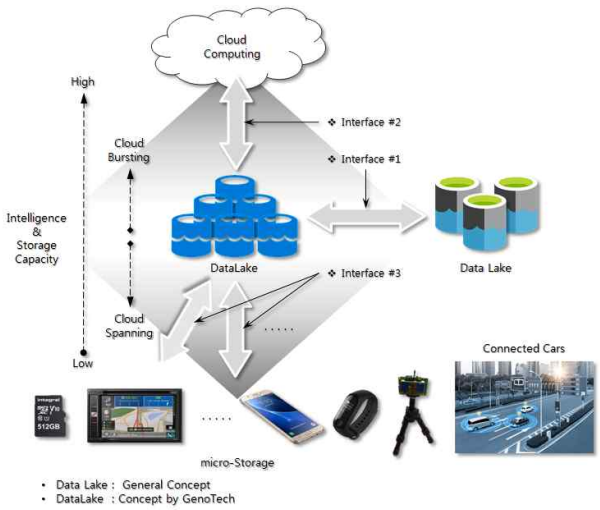

그림 4. Connected Data 개념의 인터페이스 정의

DataLake Framework의 CDA 개념의 사용 케이스들로는 Interface 사례 별로 각각의 제약 사항과 필요 기능들로 서술한다. 위의 [그림 4]에서 DataLake를 중심으로 Interface #1은 Abyss Storage Cluster의 물리적 스토리지를 기반 DataLake Framework와 다른 Data Center와의 네트워크 연결을 통한 데이터 공간의 확장 및 백업(Back-up), 그리고 이를 지원하기

위한 추가적인 안전한 데이터 전송 및 보안(Security)을 위한 컴퓨팅 지원이 필수적으로 필요하다.

Interface #2는 DataLake Framework와 퍼블릭 클라우드의 스토리지 자원과의 네트워크 연결을 통한 데이터 공간의 확장 및 안전한 데이터 전송, 고도화된 서비스(Advanced Services)를 위한 컴퓨팅 지원, 그리고 데이터 보안이 필요하다. 특히 Interface #2는 데이터 공간 확보 측면보다는 클라우드 컴퓨팅 자원을 이용한 Intelligence Analysis 또는 Prediction 등의 다양한 서비스 지원에 의미가 있다.

마지막으로 Interface #3은 DataLake와 micro-Storage를 지원하는 IoT 디바이스, 스마트폰, 네비게이션, 커넥티드 카(Connected Car) 등과의 무선 네트워크 연결을 통한 다양한 디바이스의 스토리지 공간을 제공하며, 추가적으로 Intelligence 서비스를 위한 컴퓨팅을 지원하게 되며, 가장 취약한 부분이 보안으로 추측된다.

Ⅳ. Abyss Storage Cluster 기반 DataLake Framework의 CDA 개념 검증

본 연구에서 제안하는 Abyss Storage Cluster 기반 DataLake Framework의 CDA 개념의 유효성 검증 측면에서는 CDA 개념에 정의된 Interface #2(DataLake 프레임워크와 퍼블릭 클라우드 스토리지)와 Interface #3(DataLake 프레임워크와 IoT 디바이스의 micro-Storage)의 시뮬레이션에 의한 유효성 검증을 진행하였다. 그러나 Interface #1 의 DataLake와 Data Lake 간의 테스트는 단지 데이터 복사 또는 내부 네트워크 트래픽과 같은 효과를 보일 것으로 추정하며, 유효성 검증과는 특별한 의미가 없어 생략하였다. 유효성 검증은 정확한 성능 테스트와 별개로 본 연구에서 제안하는 개념을 실체화하기 위한 가능성을 점검하는 과정으로 정의한다.

1. Interface #2 - DataLake Framework와 클라우드 스토리지(Google & AWS) 간의 CDA 개념 검증

CDA 개념의 Interface #2의 유효성 검증은 퍼블릭 클라우드 스토리지 AWS(Amazon Web Service)와 Google Storage의 버킷 생성을 통한 데이터 크기별로 1MB, 10MB, 100MB, 1GB, 2GB, 그리고 5GB와 Abyss Storage Cluster 기반 DataLake와의 파일 업로드 및 다운로드 테스트를 실시하여 Interface #2의 유효성을 검증하였다.

(1) AWS와 DataLake Framework 간의 CDA 개념의 Interface #2 유효성 검증

퍼블릭 클라우드 스토리지 AWS 버킷들과 Abyss Storage 기반 DataLake 간의 네트워크 연결을 통한 데이터 크기별로 파일의 업로드 및 다운로드 테스트를 실시한 결과를 [표 1]과 [표 2]에 나타내었으며, 단위는 파일의 업로드 및 다운로드에 소요된 시간(ms)을 표현하였다.

표 1. AWS의 데이터 업로드 테스트 결과

횟수	Data Size					
	1M	10M	100M	1G	2G	5G
1	1.51	3.68	12.13	105.02	355.53	816.23
2	1.51	3.55	12.21	100.55	201.91	496.71
3	1.93	4.37	13.3	99.95	199.9	498.82
4	1.96	3.86	12.09	99.54	536.98	536.36
5	2.12	3.48	12.59	257.57	215.07	526.06
6	2.29	3.65	17.4	600.25	237.7	606
7	1.89	6.53	12.45	163.25	199.82	1397.49
8	1.74	3.32	12.05	377.45	200.67	536.92
9	1.64	3.92	12.69	103.69	617.64	494.72
10	1.99	3.34	47.5	104.43	198.07	490.15
평균	1.858	3.97	16.441	201.17	296.329	639.946

표 2. AWS의 데이터 다운로드 테스트 결과

횟수	Data Size					
	1M	10M	100M	1G	2G	5G
1	1.88	3.6	15.78	155.49	1206.36	774.34
2	1.89	3.59	87.17	144.62	316.7	839.73
3	4.13	4.29	15.56	277.67	301.57	1100.68
4	1.9	4	19.78	134.18	407.47	728.18
5	2.09	3.68	19.24	152.65	300.7	735.41
6	2.5	3.62	22.37	397.38	576.1	949.95
7	1.88	3.87	17.82	168.96	342.33	689.19
8	1.68	3.57	20.71	188.31	310.37	789.24
9	1.73	4.27	20.69	245.37	374.28	923.28
10	1.79	8.21	16.5	237.35	427.76	963.16
평균	2.147	4.27	25.562	210.198	456.364	849.316

(2) Google 스토리지와 DataLake Framework 간의 CDA 개념의 Interface #2 유효성 검증

퍼블릭 클라우드 스토리지 Google Storage 버킷들과 Abyss Storage 기반 DataLake 간의 네트워크 연결을 통한 데이터 크기별로 파일의 업로드 및 다운로드 테스트를 실시한 결과를 [표 3]와 [표 4]에 나타내었으며, 단위는 파일의 업로드 및 다운로드에 소요된 시간(ms)을 표현하였다.

Abyss Storage 기반 DataLake Framework의 Connected Data 개념의 Interface #2의 유효성 검증에서 대체적으로 퍼블릭 클라우드와 Abyss Storage 클러스터와의 데이터 전송 간에 데이터 용량이 커짐에 따른 데이터 전송 시간의 점차적인 증가와 더불어 전송 속도의 분산(Variance) 또한 커지는 경향

을 보였으며([표 1]과 [표 2] 그리고 [표 3]와 [표 4]을 참조), 이것은 Abyss Storage Cluster 기반 DataLake Framework 측면에서 Meta-data & Content Layer의 Cloud Spanning 기능 구현의 데이터 전송 측면에서 외부 및 내부 네트워크 상황과 네트워크 안정성에 따른 영향력이 매우 크다는 것을 간접적으로 입증하고 있다.

표 3. Google Storage의 데이터 업로드 테스트 결과

횟수	Data Size					
	1M	10M	100M	1G	2G	5G
1	3.72	5.99	12.1	98.11	194.73	479.89
2	7.56	4.37	12.05	96.52	196.72	481.85
3	4.19	5.82	10.3	96.73	195.01	481.89
4	6.77	5.07	13.33	97.08	192.55	480.41
5	3.45	3.26	11.65	97.13	193.66	481.62
6	2.8	3.05	15.59	96.68	194.28	483.52
7	5.87	4.49	12.36	96.71	194.42	486.11
8	4.06	3.13	10.83	98.26	192.71	483.79
9	5.71	2.22	12.75	97.18	194.04	483.54
10	2.45	4.09	11.43	98.31	192.43	483.39
평균	4.658	4.149	12.239	97.271	194.055	482.601

표 4. Google Storage의 데이터 다운로드 테스트 결과

횟수	Data Size					
	1M	10M	100M	1G	2G	5G
1	2.67	2.41	12.19	93.29	184.94	488.82
2	2.54	3.33	12.16	92.85	187.45	459.17
3	2	3.97	12.06	93.24	185.49	459.8
4	2.28	3.92	12.5	100.53	184.43	467.28
5	1.37	3.92	11.4	95.68	185.52	461.4
6	2.15	3.96	11.54	95.05	184.59	521.15
7	2.55	4.95	12.03	100.02	184.38	459.4
8	2.34	4.75	10.55	94.06	184.44	463.17
9	3	1.85	10.27	94.81	194.51	461.01
10	20.8	3.93	10.29	92.82	186.1	459.12
평균	4.17	3.699	11.499	95.235	186.185	470.032

2. Interface #3 - DataLake Framework와 micro-Storage 간의 CDA 개념 검증

Abyss Storage 기반 DataLake의 2G bps 외부 네트워크 스위치에 초소형 PC인 Raspberry Pi 5대와의 네트워크 연결 및 Parallel-SCP를 이용하여 파일을 동시에 전송하는 테스트로 Interface #3의 유효성을 검증하였다. IoT 디바이스의 micro-Storage를 Raspberry Pi로 대처하였으며, 세분화하여 Raspberry Pi 3 B와 Raspberry Pi 3 B+를 통한 전송완료 시간(ms)을 [표 5]과 [표 6]에 비교하였다.

CDA 개념의 Interface #3 테스트 결과로 Raspberry Pi 3 B와 Raspberry Pi 3 B+의 성능을 비교하면 차이는 크게 나지 않지만 Raspberry Pi 3 B+가 평균 1 ~ 3초 빠르게 파일전송이 완료되는 것을 확인이 가능하였으며([표 5]과 [표 6]의 비교 참조), 이를 통하여 IoT 영역의 micro-Storage를 구성하는 H/W 및 S/W 성능이 CDA 개념의 Interface #3의 성능에 막대한 영향을 미치는 요소임을 짐작하게 하였다.

표 5. Raspberry Pi 3 B 파일전송 테스트 결과

횟수	node1	node2	node3	node4	node5
1	34	38	33	34	35
2	44	41	36	32	32
3	39	34	34	32	33
4	34	35	34	32	37
5	41	34	40	36	39
6	34	34	37	40	37
7	38	35	34	42	41
8	38	39	37	34	36
9	39	40	35	36	32
10	38	43	34	33	38
평균	37.9	37.3	35.4	35.1	36

표 6. Raspberry Pi 3 B+ 파일전송 테스트 결과

횟수	node1	node2	node3	node4	node5
1	35	35	31	37	30
2	36	36	34	33	36
3	32	32	33	33	36
4	32	32	35	37	37
5	38	38	37	42	39
6	38	38	40	36	33
7	33	33	34	31	33
8	33	33	33	37	33
9	33	33	39	35	36
10	33	33	37	37	34
평균	34.3	34.3	35.3	35.8	34.7

V. 결 론

최근 데이터는 기업 운영의 중추적인 역할과 중요성이 높아지고 있으며, 점차적으로 기업의 가치는 데이터 중심으로 이동하고 있다. Abyss Storage Cluster 기반 DataLake Framework의 Connected Data Architecture 개념은 물리적으로 분할된 다양한 스토리지들을 소프트웨어에 의한 논리적으로 묶을 수 있다는 개념을 수립하며, DataLake와 Data Lake 간의 Interface #1, DataLake와 Cloud Storage 간의 Interface #2, 그리고 DataLake와 micro-Storage 간의 Interface #3으로 정의하였다. 실제적 활용 측면에서 DataLake와 Public Cloud Storage, Data Center, 그리고 IoT 디바이스의 micro-Storage 등과의 인터페이스별로 사용 케이스들을 서술하였다. 또한 CDA 개념의 Interface #2와 Interface #3의 유효성 검증을 수행하였다.

REFERENCES

[1] omcy John, Pankaj Misra, *"Data Lake for Enterprises - Leveraging Lambda Architecture for Building Enterprise Data Lake,"* Packt Publishing, May 2017.

[2] 차병래 외 3인, “Abyss Storage Cluster 기반의 DataLake Framework의 설계,” *스마트미디어저널*, Vol. 7, No. 1, 9-15쪽, 2018년 3월
[3] Mark Harring, *"Connected Data Ponds: The evolution of Data Lakes,"* Hortonworks, Sept. 08, 2016.
[4] 차병래 외 3인, “Abyss Storage Cluster 기반 DataLake Framework의 Connected Data Architecture 개념 설계,” *2018 한국정보기술학회 하계공동학술대회*, 4-7쪽, 조선대학교, 대한민국, 2018년 6월 8일.
[5] 차윤석 외 4인, “Abyss Storage의 Disk 타입에 의한 Ceph RADOS의 Benchmarking,” *2017 한국통신학회 동계학술대회*, pp.1271-1273, 하이원리조트, 대한민국, 2018년 6월.
[6] 차병래 외 4인, “대용량 Abyss Storage의 KOREN 네트워크 기반 국내 및 해외 실증 테스트,” *스마트미디어저널*, 제6권, 제1호, 9-15쪽, 2017년 3월
[7] Lambda Architecture(2008). http://searchbusinessanalytics.techtarget.com/definition/Lambda-architecture (accessed Sep., 2018).
[8] 차병래 외 4인, “Idea Sketch to Improvement Image Learning based on Machine Learning using Topology Theory,” *SMA 2017,* pp.139-141, Boracay, Philippine, Dec. 2017.
[9] Cloud Bursting(2017). http://searchcloudcomputing.techtarget.com/definition/cloud-bursting (accessed Sep. 7, 2018).
[10] Cloud Spanning(2014), http://searchcloudcomputing.techtarget.com/definition/cloud-spanning (accessed Sep., 7, 2018).
[11] OMA(2011), *"Download Over the Air Specification,"* Open Mobile Alliance, March 2011.
[12] OMA(2006), "Download Over the Air Architecture," Open Mobile Alliance, Aug. 2006.

저 자 소 개

차병래

2004년 목포대학교 대학원 컴퓨터공학과 졸업(공학박사)
2005년 호남대학교 컴퓨터공학과 전임강사
2009년 ~ 현재 광주과학기술원 전기전자컴퓨터공학부 연구조교수
2012년 ~ 현재 제노테크(주) 대표이사
<주관심분야: 정보보안, IDS, Neural Network, Cloud Computing, VoIP, NFC, 대용량 스토리지 기술 등>

차윤석

2014년 고려대학교 컴퓨터정보학과 학사
2015년 ~ 현재 제노테크(주) 기업부설 연구소 연구원
<주관심분야 : IoT, BigData, Cloud Computing, VoIP, NFC, 대용량 스토리지 기술 등>

박 선

2007년 인하대학교 컴퓨터정보공학과 공학박사
2008년 호남대학교 컴퓨터공학과 전임강사
2010년 전북대학교 인력양성사업단 박사후 과정
2010년 목포대학교 정보산업연구소 연구전임교수
2013년 ~ 현재 광주과학기술원 NetCS연구실 연구교수
<주관심분야: 정보검색, 데이터마이닝, 해양IT정보융합, 클라우드 컴퓨팅, IoT, 스토리지 시스템>

신병춘

2002년 전남대학교 수학과 조교수
2011년 ~ 현재 전남대학교 수학과 교수
<주관심분야 : 수치해석, 인공신경망, 컴퓨터 비전>

김종원

1997년 University of Southern California 연구 조교수
1999년 Technology Consultant for VProtect Systems Inc.
2000년 Technology Consultant for Southern California Division of InterVideo Inc.
2001년 광주과학기술원 전기전자컴퓨터공학부 교수
2008년 ~ 현재 광주과학기술원 전기전자컴퓨터공학부 교수
<주관심분야: Future Internet, SDN & NFV, SDI>