รายงานสหกิจศึกษาฉบับสมบูรณ์

โครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึก
Big Data Infrastructure and Anomaly Detection Analytic

นางสาวรัตนาวดี ไวพาทิน

ภาควิชาวิศวกรรมคอมพิวเตอร์ สาขาวิศวกรรมสารสนเทศ
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2559

รายงานสหกิจศึกษาฉบับสมบูรณ์

โครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึก

Big Data Infrastructure and Anomaly Detection Analytic

นางสาวรัตนาวดี ไวพาทิน

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2559

ชื่อโครงงานสหกิจศึกษา โครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึก

ชื่อ-สกุล นักศึกษา นางสาวรัตนาวดี ไวพาทิน

คณะ วิศวกรรมศาสตร์ **ภาควิชา** วิศวกรรมคอมพิวเตอร์ **สาขาวิชา** วิศวกรรมสารสนเทศ

ชื่อ-สกุล อาจารย์นิเทศ ผศ.ดร.สุธีรา พันธุ์ธีรานุรักษ์

ชื่อ-สกุล ผู้นิเทศงาน นาย อลงกต บุรุษอาชาไนย

นาย ณัฐณพัชร์ กวิพรรธน์

ชื่อสถานประกอบการ บริษัท พีทีที ไอซีที โซลูชั่นส์ จำกัด

บทคัดย่อ

โครงงานฉบับนี้นำเสนอการพัฒนาโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกของบริษัท พีทีที ไอซีที โซลูชั่นส์ จำกัด โดยเป็นระบบที่พัฒนาขึ้นเพื่อนำมาใช้งานในส่วนของฐานเก็บข้อมูลและนำผลที่ได้จากการวิเคราะห์ความผิดปกติในเชิงลึกไปแก้ไขความผิดปกติ จากระบบเดิมที่มีฐานข้อมูลขนาดเล็กทำให้อาศัยพนักงานเข้ามาดูแลและจัดการฐานข้อมูลเป็นประจำ เช่น การบีบอัดไฟล์ในฐานข้อมูล การลบข้อมูลที่เกินระยะเวลาที่กำหนด การลบข้อมูลที่เกิดจากความผิดปกติ ทางผู้จัดทำจึงได้พัฒนาระบบนี้ขึ้น เพื่อให้สามารถส่งเสริมการทำงานของฐานข้อมูลเดิมที่มีอยู่ โดยสร้างโครงสร้างข้อมูลขนาดใหญ่เพื่อให้สามารถเก็บข้อมูลได้มากขึ้น และวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกเพื่อกำจัดข้อมูลที่ผิดปกติออกไปโดยที่พนักงานเข้ามาดูแลและจัดการลบข้อมูลต่าง ๆ น้อยที่สุด

คำสำคัญ: โครงสร้างข้อมูลขนาดใหญ่ เครื่องมือที่ใช้ในการรวบรวม เก็บ และวิเคราะห์ข้อมูล

Co-operative Title: Big Data Infrastructure and Anomaly Detection Analytic

Student Intern Name: Ms.Rattanawadee Waipatin

Faculty: Bachelor of Engineering **Department:** Computer Engineering

Program: Information Engineering

Advisor Name: Asst.Prof.Dr.Sutheera Puntheeranurak

Mentor Name: Mr.Alongkot Burutarchanai

Mr.Natnapat Gaviphatt

Company: PTT ICT Solutions Company Limited

ABSTRACT

This project presents Big Data Infrastructure and Anomaly Detection Analytic of PTT ICT Solutions Company Limited. The system is developed to help programmers decrease managing database by sending data to big data infrastructure. The legacy database is too small to keep all data and always needs programmers zipping files in the database, deleting the old data and deleting anomaly data. Implemented this infrastructure to support the database and detect the anomaly data instead of delete functions and for helping programmers to manage results.

Keywords: Big Data Infrastructure, tools for collecting, storing and analyzing data

กิตติกรรมประกาศ

การจัดทำโครงงานโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกนี้เป็นโครงงานของบริษัท พีทีที ไอซีที โซลูชั่นส์ จำกัด เกิดขึ้นจากการเริ่มต้นนำเทคโนโลยีใหม่ ๆ มาพัฒนาประสิทธิภาพการทำงานในองค์กรให้ดียิ่งขึ้น โดยผลงานดังกล่าวสำเร็จลุล่วงไปด้วยดี เพราะได้รับคำแนะนำและการให้คำปรึกษาตลอดจนติดตามผลความคืบหน้าจากหลาย ๆ ส่วน ทางคณะผู้จัดทำจึงขอขอบคุณทุกท่านที่ได้ให้ความช่วยเหลือตลอดระยะเวลาที่ได้มีโอกาสเข้าไปดำเนินงานโครงการสหกิจศึกษาและเรียนรู้ประสบการณ์การทำงานต่าง ๆ ตลอดจนสิ้นสุดโครงการ ตั้งแต่วันที่ 1 มิถุนายน 2559 จนถึง 25 พฤศจิกายน 2559

โครงงานนี้ไม่อาจสำเร็จลุล่วงได้หากขาดความกรุณาของคุณเขมรัฐ โชคมั่งมี ผู้จัดการส่วนแผนกออโตเมชั่น คุณอลงกต บุรุษอาชาไนย ที่ปรึกษาอาวุโส และนายณัฐณพัชร์ กวิพรรธน์ ผู้เป็นพี่เลี้ยงที่คอยช่วยเหลือ ดูแลเอาใจใส่ และให้คำปรึกษาต่าง ๆ มอบความทรงจำและประสบการณ์อันมีค่าตลอดระยะเวลาที่ผ่านมา

ขอขอบพระคุณผศ.ดร.สุธีรา พันธุ์ธีรานุรักษ์ ผู้เป็นอาจารย์นิเทศโครงการสหกิจศึกษาที่คอยให้การสนับสนุน ติดตามความคืบหน้าของโครงการ ให้คำปรึกษาและแนวทางแก้ไขปัญหาต่าง ๆ

สุดท้ายนี้ขอขอบคุณเพื่อนนักศึกษาและครอบครัว ที่คอยช่วยเหลือ ผลักดัน และเป็นกำลังใจสำคัญในการฟันฝ่าอุปสรรคต่าง ๆ ตลอดมา รวมถึงผู้มีพระคุณทุกท่านที่มิได้เอ่ยนามไว้ ณ ที่นี้ที่เป็นส่วนหนึ่งของความสำเร็จทั้งหมด จึงขอขอบคุณไว้ ณ โอกาสนี้

รัตนาวดี ไวพาทิน

สารบัญ

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

สารบัญภาพ(ต่อ)

หน้า

สารบัญภาพ(ต่อ)

หน้า

สารบัญภาพ(ต่อ)

หน้า

สารบัญภาพ(ต่อ)

หน้า

บทที่ 1

บทนำ

1.1 ข้อมูลสถานประกอบการที่เข้าร่วมปฏิบัติงานสหกิจศึกษา

บริษัท พีทีที ไอซีที โซลูชั่นส์ จำกัด ก่อตั้งขึ้นเมื่อวันที่ 7 กรกฎาคม 2549 โดยมีวัตถุประสงค์หลัก เพื่อสร้างมูลค่าเพิ่มทางธุรกิจให้กับบริษัทในกลุ่ม ปตท. ด้วยการปฏิบัติการร่วมกันเป็นหนึ่งเดียว (Group Synergy) ตามวิสัยทัศน์ของกลุ่มปตท. บริษัทพลังงานไทยข้ามชาติชั้นนำ (Thai Premier Multinational Energy Company) พีทีที ไอซีที โซลูชั่นส์ จึงก่อกำเนิดขึ้นจากความร่วมมือของบริษัทในกลุ่ม ปตท. ทั้ง 4 บริษัท อันได้แก่

1. บริษัท ปตท. จำกัด (มหาชน)
2. บริษัท ปตท. สำรวจและผลิตปิโตรเลียม จำกัด (มหาชน)
3. บริษัท พีทีที โกลบอล เคมิคอล จำกัด (มหาชน)
4. บริษัท ไทยออยล์ จำกัด (มหาชน)

1.2 ที่มาและความสำคัญของโครงงาน

ปัจจุบันเทคโนโลยีได้มีการพัฒนาไปอย่างรวดเร็วทำให้พนักงานด้านไอทีควรพัฒนาความรู้และการใช้งานอุปกรณ์ใหม่ ๆ อยู่เสมอ จากการที่องค์กรมีการนำอุปกรณ์ติดตามปริมาณการซื้อขายน้ำมันและก๊าซทำให้ข้อมูลการซื้อขายที่เกิดขึ้นทั่วประเทศส่งเข้ามาเก็บในฐานข้อมูลขององค์กรทุกวัน อีกทั้งยังมีข้อมูลอื่น ๆ อีก เช่น ข้อมูลเซนเซอร์จากรถขนส่งน้ำมัน ข้อมูลการขายกาแฟอเมซอนในแต่ละวัน ทำให้ฐานข้อมูลมีพื้นที่ไม่เพียงพอต่อปริมาณข้อมูล จึงจำเป็นต้องมีโปรแกรมเมอร์คอยบีบอัดไฟล์ในฐานข้อมูล ลบข้อมูลที่เกินระยะเวลาที่กำหนด ลบข้อมูลที่เกิดจากความผิดปกติเพื่อให้ฐานข้อมูลมีพื้นที่ว่างเพียงพอจะรับข้อมูลใหม่ ๆ ในแต่ละวันอยู่เสมอ ซึ่งบางครั้งการทำงานเหล่านี้ส่งผลให้เสียเวลาต่อการทำงานอื่น ๆ ของผู้ดูแล เพราะข้อมูลที่เข้ามาในแต่ละวันมีจำนวนมาก และต้องใช้เวลาในการเลือกลบข้อมูลที่ผิดปกติออกไป ผู้จัดทำจึงพัฒนาโครงงานโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึก (Big Data Infrastructure and Anomaly Detection Analytic) ขึ้นเพื่อเพิ่มพื้นที่ในการเก็บข้อมูลการซื้อขายน้ำมันและก๊าซ เพื่อช่วยลดการหาข้อมูลที่ผิดปกติ ทำให้พนักงานทำงานได้อย่างมีประสิทธิภาพ แม่นยำและรวดเร็ว

1.3 วัตถุประสงค์ของโครงงาน

1. เพิ่มพื้นที่เก็บข้อมูลให้กับองค์กรส่งผลให้สามารถเก็บข้อมูลได้จำนวนมากขึ้น

2. ลดเวลาในการทำงานของผู้ดูแลฐานข้อมูล และเพิ่มประสิทธิภาพในการทำงานอื่น

3. การเพิ่ม ลบ ข้อมูลสามารถทำได้ผ่านหน้าติดต่อประสานงานโดยไม่จำเป็นต้องมีความรู้ด้านโปรแกรมเมอร์ทำให้ผู้ใช้งานในแผนกอื่น ๆ สามารถเข้าใจได้ง่าย

1.4 ขอบเขตของโครงงาน

1. พัฒนาโครงสร้างข้อมูลขนาดใหญ่เพื่อใช้เก็บข้อมูลจาก 2 ส่วนหลัก ได้แก่

- ข้อมูลหัวจ่ายน้ำมัน เป็นข้อมูลการขายน้ำมันจากหัวจ่ายทุกหัวโดยจะประกอบด้วย เวลาเริ่มต้นการจ่ายน้ำมัน เวลาสิ้นสุดการจ่ายน้ำมัน จำนวนลิตรที่จ่าย หมายเลขถังเก็บน้ำมันหมายเลขหัวจ่าย ปริมาณน้ำมันที่เหลือในถังหลังจ่ายน้ำมัน เป็นต้น

- ข้อมูลหัวจ่ายก๊าซธรรมชาติ เป็นข้อมูลการขายก๊าซธรรมชาติจากหัวจ่ายทุกหัวซึ่งข้อมูลนี้จะเป็นหน้าที่ ของแผนกอื่นที่ต้องเอาไปดูแลต่อ แต่ในแผนกของข้าพเจ้ามีหน้าที่เก็บข้อมูลส่วนนี้เท่านั้น

2. โครงสร้างข้อมูลขนาดใหญ่สามารถเก็บข้อมูลได้ 1 เทระไบต์

3. โครงสร้างข้อมูลขนาดใหญ่ใช้เป็นที่เก็บข้อมูลสำรองในกรณีที่ฐานข้อมูลเดิมมีความผิดปกติ

4. การดูแลและจัดการข้อมูลในโครงสร้างข้อมูลขนาดใหญ่เดือนละ 1 ครั้ง

5. การวิเคราะห์หาความผิดปกติของข้อมูลทำให้การจัดข้อมูลทำได้ภายใน 2 ชั่วโมง

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 สำหรับผู้จัดทำโครงงาน

1. ได้รับความรู้จากการศึกษาการสร้างโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์ความผิดปกติของข้อมูลในเชิงลึก

2. ได้รับความรู้เรื่องการใช้งานบนระบบปฏิบัติการลินุกซ์

3. ได้รับความรู้เกี่ยวกับการติดตั้งโครงสร้างข้อมูลขนาดใหญ่

4. ได้รับความรู้เกี่ยวกับการทำงานและชนิดของเครื่องมือที่ใช้กับข้อมูลขนาดใหญ่

5. ได้รับความรู้การวิเคราะห์ข้อมูลในเชิงลึกด้วยภาษาสกาล่าโดยใช้ทฤษฎีเหมืองข้อมูล

6. มีความเข้าใจในขั้นตอนการทำงานและสามารถปรับตัวเข้ากับสภาพแวดล้อมการทำงานของบริษัทพีทีที ไอซีที โซลูชั่นส์ จำกัด

7. เรียนรู้อุปสรรคและการแก้ไขปัญหาในเหตุการณ์ต่าง ๆ ในการทำงาน ทั้งจากการปรึกษาเพื่อนร่วมงาน และศึกษาเรียนรู้ด้วยตนเอง

1.5.2 สำหรับผู้ใช้งาน

1. เพิ่มพื้นที่จัดเก็บข้อมูล

2. เพิ่มประสิทธิภาพในการจัดเก็บ ค้นหา และการเรียกดูข้อมูล

3. ลดภาระการดูและและจัดการข้อมูลในฐานข้อมูลลง

1.6 ขั้นตอนการดำเนินงาน

ในการดำเนินงานจัดทำโครงงานโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกประกอบด้วยขั้นตอนและระยะเวลาการดำเนินการดังนี้

ตารางที่ 1.1 ตางรางแสดงขั้นตอนการดำเนินการ

กิจกรรม	ระยะเวลา (เดือน)					
	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.
วางแผนโครงงาน	→					
ศึกษาข้อมูลเครื่องมือและโครงสร้างข้อมูลขนาดใหญ่	→	→				
ทดสอบการติดตั้งและการใช้งานเครื่องมือพื้นฐาน			→	→		
สร้างโครงสร้างข้อมูลขนาดใหญ่เพื่อการใช้งานจริง					→	
วิเคราะห์ความผิดปกติของข้อมูลในเชิงลึก						→
จัดทำเอกสารประกอบโครงงาน		→	→	→	→	→

1.7 อุปกรณ์ที่ใช้ในการพัฒนา

1.7.1 ฮาร์ดแวร์

1. คอมพิวเตอร์โน้ตบุ๊กสำหรับพัฒนาโปรแกรม

2. เซิร์ฟเวอร์ของบริษัท

1.7.2 ซอฟต์แวร์

1. อาปาเช่ฮาดูป (Apache Hadoop)

2. อาปาเช่นายฟาย (Apache Nifi)

3. อาปาเช่สปาร์ก (Apache Spark)

4. อาปาเช่อัมบารี (Apache Ambari)

5. อินเทลลิเจไอเดีย (IntelliJ idea)

6. วาแกรนต์ (Vagrant)

1.7.3 ภาษาที่ใช้พัฒนา

1. ภาษาจาวา (Java)

2. ภาษาสกาล่า (Scala)

4. ภาษาเชลล์สคริปต์ (Shell Script)

บทที่ 2

แนวคิด ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 อาปาเช่ฮาดูป (Apache Hadoop)

อาปาเช่ฮาดูปเป็นโอเพนซอร์สซอฟต์แวร์แพลตฟอร์ม (Open source software platform) สำหรับตัวเก็บข้อมูลแบบกระจายและการประมวลผลเซตของข้อมูลที่มีขนาดใหญ่มากแบบกระจายบนกลุ่มของคอมพิวเตอร์ที่สร้างจากฮาร์ดแวร์ บริการของฮาดูปจะประกอบไปด้วยส่วนเก็บข้อมูล ส่วนประมวลผลข้อมูล ส่วนนำเข้าข้อมูล ระบบจัดการข้อมูล ระบบความปลอดภัยและระบบการดำเนินงาน

ฮาดูปแบ่งความสามารถในการทำงานได้เป็น 4 ประเภท ดังนี้

2.1.1 หน่วยเก็บข้อมูล (Data storage) หรือเอชดีเอฟเอส (HDFS)

หน่วยเก็บข้อมูลสามารถรองรับการเพิ่มขยายได้ในอนาคต คงทนต่อความเสียหาย และใช้ที่เก็บข้อมูลที่จะนำไปวิเคราะห์ได้อย่างมีประสิทธิภาพและคุ้มค่าเอชดีเอฟเอสถูกออกแบบมาเพื่อขยายคลัสเตอร์ (cluster) ขนาดใหญ่ของเซิร์ฟเวอร์โดยเพิ่มไปถึงหลักร้อยเพตะไบต์และหลักพันของเซิร์ฟเวอร์

2.1.2 หน่วยประมวลผลข้อมูล (Data processing) หรือแมพรีดิวซ์ (MapReduce)

หน่วยประมวลผลข้อมูลเป็นเฟรมเวิร์ค (Framework) พื้นฐานสำหรับเขียนแอปพลิเคชันคู่ขนานที่ประมวลผลกลุ่มของข้อมูลขนาดใหญ่ทั้งที่มีโครงสร้างและไม่มีโครงสร้างที่เก็บใน เอชดีเอฟเอส แมพรีดิวซ์ใช้ข้อดีของการที่อยู่ใกล้ที่เก็บข้อมูลทำให้สามารถนำข้อมูลมาในแต่ละโหนดบนคลัสเตอร์ได้ ซึ่งลดระยะทางและระยะเวลาในการขนส่งข้อมูล

แต่เมื่อไม่นานมานี้ได้มีการนำยาร์น (YARN) มาใช้โดยให้ฮาดูป สามารถใช้งานควบคู่กับเครื่องประมวลผลอื่น ๆ เช่น อาปาเช่สปาร์ก (Apache Spark) ไปด้วยได้ทำให้สามารถประมวลผลข้อมูลได้ในหลาย ๆ วิธีในเวลาเดียวกันยาร์นจะจัดการทรัพยากรส่วนกลางที่ทำให้การประมวลผลงานหลายอย่างในปริมาณที่เหมาะสมในเวลาเดียวกันเกิดขึ้นได้ ยาร์นจึงเป็นพื้นฐานของฮาดูปรุ่นใหม่

อาปาเช่เทซ (Apache Tez) เป็นเฟมเวิร์คที่สามารถขยายได้ ใช้สำหรับสร้างแฟ้มคำสั่งรวม (Batch file) ที่มีประสิทธิภาพสูงและประสานงานกับ YARN ในการทำงานร่วมกันกับแอปพลิเคชันสำหรับประมวลผลข้อมูล ซึ่งเทซทำให้แมพรีดิวซ์ทำงานได้ดีขึ้น มีความเร็ว เพิ่มขึ้นและรักษาความสามารถในการรองรับข้อมูลระดับเพตะไบต์ได้ ในการใช้งานแมพรีดิวซ์จะถูกบังคับให้ใช้เทซร่วมด้วย

2.1.3 การเข้าถึงและการวิเคราะห์ข้อมูล (Data access and analysis)

แอปพลิเคชันสามารถมีปฏิกิริยาต่อข้อมูลในฮาดูปได้โดยใช้แฟ้มคำสั่งรวมหรือซีควล (SQL) ผ่านอาปาเช่ไฮฟ์ (Apache Hive) หรือการเข้าถึงแบบความหน่วงเวลาน้อยจะใช้โนซีควล (NoSQL) ผ่าน อาปาเช่เอชเบส (Apache HBase)

ไฮฟ์จะยอมให้ทั้งผู้ใช้งานทั่วไปและผู้เชี่ยวชาญในการวิเคราะห์ข้อมูลใช้งานการวิเคราะห์ข้อมูล การรายงาน และการแสดงผลบนเครื่องมือของฮาดูปและการสืบค้นข้อมูลในส่วนเก็บข้อมูลของฮาดูปซึ่งก็คือเอชดีเอฟเอสสามารถทำได้โดยใช้อาปาเช่โซลาร์ (Apache Solr)

2.1.4 ระบบการจัดการและความปลอดภัยของข้อมูล (Data governance and security)

อาปาเช่เรนเจอร์ (Apache Ranger) จะบริหารความปลอดภัยในการนำเข้าข้อมูลและการนำข้อมูลแต่ละส่วนมารวมเข้าด้วยกันในฮาดูปและอาปาเช่นอกซ์ (Apache Knox) จะช่วยควบคุมการเข้าใช้งานของผู้ใช้งานแต่ละคน

ประโยชน์ของฮาดูปที่เป็นเหตุผลให้หลาย ๆ องค์กรได้เลือกใช้เป็นเพราะความสามารถในการเก็บ การจัดการ และการวิเคราะห์ทั้งข้อมูลที่มีโครงสร้างและข้อมูลที่ไม่มีโครงสร้างในปริมาณมหาศาลนั้น มีดังนี้

- ความสามารถในการรองรับการเพิ่มขยายได้ในอนาคต (Scalability) และประสิทธิภาพการประมวลผลแบบกระจายของแต่ละโหนด (Node) ในคลัสเตอร์ทำให้ฮาดูปสามารถที่จะเก็บ จัดการและวิเคราะห์ข้อมูลในหน่วยของเพตะไบต์

- ความเชื่อมั่น (Reliability) ปกติแล้ว คลัสเตอร์ประมวลผลขนาดใหญ่มีแนวโน้มที่แต่ละโหนดในคลัสเตอร์จะมีทำงานขัดข้องเกิดขึ้นได้ โดยฮาดูปนั้นสามารถที่จะฟื้นคืนสภาพได้เร็ว คือเมื่อโหนดที่ประมวลผลขัดข้องนั้นเชื่อมต่อโหนดที่เหลือในคลัสเตอร์อีกครั้ง ข้อมูลจะถูกทำสำเนาโดยอัตโนมัติเพื่อเตรียมการสำหรับการทำงานขัดข้องของโหนดในอนาคต

- ความยืดหยุ่น (Flexibility) จะแตกต่างจากระบบฐานข้อมูลเชิงสัมพันธ์ คือ ไม่ต้องสร้างโครงสร้างข้อมูลใด ๆ ก่อนจะเก็บข้อมูลเพราะสามารถเก็บข้อมูลในรูปแบบใดก็ได้ รวมไปถึงรูปแบบกึ่งโครงสร้าง (Semi-structured format) และรูปแบบไม่มีโครงสร้าง (Unstructured format) แล้วสามารถแปลงและใส่รูปแบบให้กับข้อมูลเมื่อถูกอ่านได้

- ต้นทุนต่ำ (Low cost) ฮาดูปเป็นโอเพนซอร์สและทำงานบนฮาร์ดแวร์ราคาถูกได้

2.2 อาปาเช่ฮาดูปเอชดีเอสเอฟ (Apache Hadoop HDFS)

อาปาเช่ฮาดูปเอชดีเอฟเอสเป็นระบบแฟ้มภาษาจาวาซึ่งเป็นที่เก็บข้อมูลที่สามารถปรับขนาดและมีความเชื่อถือได้ มันถูกออกแบบมาเพื่อขยายเป็นคลัสเตอร์ขนาดใหญ่หลายคลัสเตอร์ในเซิร์ฟเวอร์ เอชดีเอฟเอสสามารถมีที่เก็บข้อมูลที่รองรับได้ถึง 200 เพตะไบต์และในหนึ่งคลัสเตอร์นั้นมี 4500 เซิร์ฟเวอร์ ซึ่งสามารถรองรับไฟล์ได้เกือบล้านล้าน เมื่อข้อมูลที่มีทั้งปริมาณและคุณภาพมาเก็บในเอชดีเอฟเอสแล้วยาร์นจะให้แอปพลิเคชันต่าง ๆ เข้ามาเอาข้อมูลไปประมวลผลพร้อมกันได้ เพราะฉะนั้นผู้ใช้ส่วนใหญ่จะมั่นใจได้เลยว่ารูปแบบข้อมูลบนเอชดีเอฟเอสนั้น เป็นที่ยอมรับและนิยมใช้กันมากในปัจจุบัน

ข้อดีของเอชดีเอฟเอสคือสามารถขยายพื้นที่เก็บข้อมูลได้ มีความคงทนต่อความเสียหาย และการที่เป็นระบบไฟล์แบบกระจายทำให้มีการทำงานที่กว้างขวางและหลากหลายต่อการทำงานของแอปพลิเคชันประมวลผลข้อมูลอื่นซึ่งควบคุมโดยยาร์น กล่าวคือเอชดีเอฟเอสสามารถทำงานภายใต้ความหลากหลายทางกายภาพและบนระบบที่มีสถานการณ์ที่หลากหลาย ด้วยกระบวนเก็บข้อมูลแบบกระจายตลอดในหลายเซิร์ฟเวอร์ ที่เก็บข้อมูลที่ถูกรวมเข้าด้วยกันนั้นสามารถเติบโตเป็นเส้นตรงควบคู่ไปกับความต้องการได้และยังที่เหลือพื้นที่เก็บข้อมูลเพียงพอต่อการใช้งานอื่น ๆ

2.3 อาปาเช่ฮาดูปแมพรีดิวซ์ (Apache Hadoop MapReduce)

อาปาเช่ฮาดูปแมพรีดิวซ์เหมาะสำหรับการประมวลผลแบบเชิงกลุ่ม (Batch processing) ในระดับเทระไบต์หรือเพตะไบต์ของข้อมูลที่เก็บในฮาดูป แมพรีดิวซ์จะทำงานโดยแบ่งเซตของข้อมูลขนาดใหญ่ออกเป็นก้อนของข้อมูลขนาดใหญ่ที่เป็นอิสระต่อกันและจัดระบบโดยการใส่กุญแจและค่าให้กับก้อนข้อมูลสำหรับการประมวลผลแบบขนาน การประมวลผลแบบนี้ทำให้ความเร็วและความน่าเชื่อถือของคลัสเตอร์นั้นดีขึ้น ให้ผลลัพธ์ที่รวดเร็วและมีความเที่ยงตรงมากขึ้น

ฟังก์ชันแมพจะแบ่งข้อมูลอินพุตออกเป็นช่วงโดยใช้อินพุตฟอร์แมทและทำการแมพข้อมูลของแต่ละช่วงของข้อมูลอินพุต ซึ่งจะมี JobTracker ที่จะแจกจ่ายงาน ให้กับโหนดเวิร์กเกอร์ (Worker node) ผลลัพธ์ที่ได้จากงานของแต่ละแมพจะถูกแบ่งโดยกลุ่มเป็นคู่ของค่าและกุญแจจากนั้นจะเข้าสู่ฟังก์ชันรีดิวซ์

ฟังก์ชันรีดิวซ์จะรวบรวมผลลัพธ์ที่หลากหลายจากการแมพสำหรับแต่ละกุญแจ และเอามารวมเป็นคำตอบที่ใช้แก้ปัญหาของโหนดมาสเตอร์ โดยแต่ละรีดิวซ์จะดึงส่วนที่สำคัญจากตำแหน่งที่ Map ทำงานจากนั้นจะเขียนผลลัพธ์กลับไปที่เอชดีเอฟเอช

ข้อดีของการทำแมพรีดิวซ์ คือ

2.3.1 ความเรียบง่าย (Simplicity) นักพัฒนาสามารถเขียนแอปพลิเคชันตามภาษาที่ต้องการได้ เช่น จาวา ซี หรือไพทอน และงานของแมพรีดิวซ์นั้นง่ายต่อการดำเนินงาน

2.3.2 ความสามารถในการรองรับการเพิ่มขยายได้ในอนาคต (Scalability) แมพรีดิวซ์ประมวลผลข้อมูลระดับเพตะไบต์ที่เก็บในเอชดีเอฟเอสบนหนึ่งคลัสเตอร์ได้

2.3.3 ความเร็ว (Speed) แมพรีดิวซ์อาจเจอปัญหาที่ต้องใช้เวลาแก้เป็นวันหรือหลาย ๆ วัน แต่การประมวลผลแบบขนานสามารถแก้ได้โดยใช้เวลาเป็นชั่วโมงหรือแค่ระดับนาที

2.3.4 การกู้คืนระบบ (Recovery) ถ้าเครื่องจักรเครื่องหนึ่งไม่สามารถหาคำตอบของข้อมูลชุดหนึ่งได้ แต่อีกเครื่องจักรหนึ่งสามารถหาคำตอบของข้อมูลได้และคู่ของค่าและกุญแจเหมือนกับข้อมูลชุดที่ไม่สามารถหาได้ แมพรีดิวซ์ จะคอยติดตามปัญหานี้ทั้งหมด

2.3.5 การเคลื่อนย้ายข้อมูลให้น้อยที่สุด (Minimal data motion) การประมวลผลของแมพรีดิวซ์สามารถเกิดบนโหนดที่ข้อมูลอยู่ใกล้ซึ่งเป็นจุดสำคัญทำให้ลดการเกิดรูปแบบอินพุตเอาต์พุตของเน็ตเวิร์คและมีผลทำให้การประมวลผลของฮาดูปรวดเร็วยิ่งขึ้น

2.4 อาปาเช่เอชเบส (Apache Hbase)

อาปาเช่เอชเบสเป็นฐานข้อมูลโนซีควลโอเพนซอร์สที่รองรับการอ่านหรือเขียนไปยังเซตข้อมูลขนาดใหญ่ในทันที เอชเบสสามารถขยายขนาดได้แบบเชิงเส้นเพื่อรองรับเซตของข้อมูลขนาดใหญ่ที่ประกอบไปด้วยล้านล้านแถวและล้านคอลัมน์ และง่ายต่อการแบ่งข้อมูลที่มีความหลากหลายในโครงสร้างที่แตกต่างกัน เอชเบสถูกสร้างขึ้นเพื่อรวมเข้ากับฮาดูปและทำงานควบคู่กันไปโดยมียาร์นเป็นตัวควบคุม

การเข้าถึงข้อมูลในฮาดูปนั้น เอชเบสจะเข้าถึงแบบสุ่มไปในทันที โดยจะสร้างตารางขนาดใหญ่ขึ้นมาเพื่อใช้เก็บข้อมูลที่มีโครงสร้างที่หลากหลายหรือแม้กระทั่งข้อมูลที่ไม่สมบูรณ์ ผู้ใช้สามารถสืบคนข้อมูลจากเอชเบสในตำแหน่งที่ต้องการได้ จึงเป็นคุณสมบัติที่ดีในการเลือกที่จะเก็บข้อมูลแบบกึ่งโครงสร้าง เช่น ข้อมูลจราจรคอมพิวเตอร์ (Log data)

เอชเบสยังมีคุณลักษณะเด่นดังนี้

2.4.1 การคงทนต่อความเสียหาย

- มีการทำสำเนาไว้
- มีความเป็นอันหนึ่งอันเดียวกัน (Atomic) มีการทำงานที่สอดคล้องกัน (Consistent)
- มีอัตราการพร้อมใช้งานที่สูงมาก
- มีการแบ่งและกระจายงานให้ทำงานได้พร้อม ๆ กัน

2.4.2 ความเร็ว

- การเรียกค้นสามารถทำได้ทันที

- มีหน่วยความจำในตัวเอง

- ประมวลผลควบคู่ไปกับการทำงานของเซิร์ฟเวอร์

2.4.3 ความพร้อมใช้งาน

- แบบจำลองข้อมูล

- การส่งข้อมูลออกในรูปแบบไฟล์

- ง่ายต่อการใช้ภาษาจาวาในการเชื่อมต่อแอปพลิเคชัน

2.5 อาปาเช่ไฮฟ์ (Apache Hive)

อาปาเช่ไฮฟ์จะสามารถสืบค้นข้อมูล สรุป สำรวจ และวิเคราะห์ข้อมูลในเชิงลึกได้ และเปลี่ยนผลลัพธ์ที่ได้ให้เป็นความเข้าใจในมุมมองเชิงธุรกิจได้ โดยจะทำงานร่วมกับฮาดูปที่ถูกสร้างมาเพื่อจัดระบบและเก็บข้อมูลมหาศาลในทุกรูปร่าง ทุกขนาด และทุกรูปแบบ สามารถทำงานร่วมกับเทคโนโลยีอื่น ๆ ที่มีเอพีไอหรือไลบรารีในจาวาที่ใช้สำหรับติดต่อกับฐานข้อมูลเชิงสัมพันธ์ เช่น เอ็มเอส ซีควล ออราเคิล และมายซีควล

ข้อดีของการใช้ไฮฟ์มีดังนี้

2.5.1 คุ้นเคยและง่ายต่อการสืบค้นข้อมูลเพราะใช้ภาษาซีควลเป็นพื้นฐาน ซึ่งเรียกว่าภาษาไฮฟ์ซีควล

2.5.2 มีการตอบสนองอย่างรวดเร็วแม้จะทำงานบนข้อมูลที่มีขนาดมหาศาล

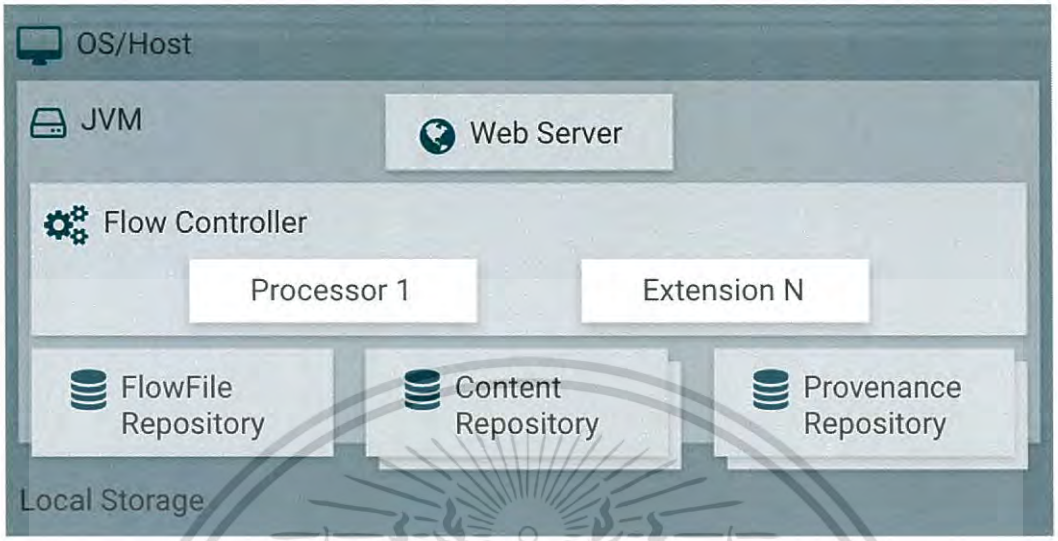
2.5.3 เนื่องจากข้อมูลมีความหมากหลายและเพิ่มจำนวนขึ้น จึงสามารถเพิ่มเครื่องจักรในการทำงานได้โดยไม่ทำให้ประสิทธิภาพในการทำงานลดลง

2.5.4 ทำงานร่วมกับเครื่องมือวิเคราะห์ข้อมูลอื่น ๆ ได้ โดยนิยมใช้ร่วมกับอาปาเช่ฮาดูปแมพรีดิวซ์

2.6 อาปาเช่นายฟาย (Apache Nifi)

อาปาเช่นายฟายเป็นแพลตฟอร์มของการส่งข้อมูลที่ใช้สำหรับการเคลื่อนย้ายอัตโนมัติระหว่างระบบที่ไม่เหมือนกันโดยสิ้นเชิง การควบคุมการทำงานสามารถทำได้ในทันที (Real-time control) ทำให้การบริหารการเคลื่อนย้ายข้อมูลระหว่างต้นทางและปลายทางเกิดขึ้นได้ สามารถรองรับข้อมูลที่มาจากการกระจายจากแหล่งต่าง ๆ ข้อมูลที่มีลักษณะไม่เหมือนกันได้ทั้ง รูปแบบ โปรโตคอล ความเร็ว และขนาด เช่น

ข้อมูลเครื่องจักร ข้อมูลตำแหน่งอุปกรณ์บนพื้นโลก ข้อมูลการคลิกเมาส์ ไฟล์ต่าง ๆ ข้อมูลโซเชียลเน็ตเวิร์ค ข้อมูลการติดต่อสื่อสารของคอมพิวเตอร์ (Log file) ข้อมูลวีดีโอ และข้อมูลต่าง ๆ อีกมากมาย



ภาพที่ 2.1 สถาปัตยกรรมของอาปาเช่นายฟาย[18]

จากภาพที่ 2.1 อาปาเช่นายฟายดำเนินการโดยระบบปฏิบัติการที่ทำให้สามารถใช้ซอฟต์แวร์เพื่อจำลองการทำงานของคอมพิวเตอร์เครื่องอื่น และมีส่วนประกอบพื้นฐานบนระบบปฏิบัติการดังนี้

1. เว็บเซิร์ฟเวอร์ จุดประสงค์ของการมีเว็บเซิร์ฟเวอร์คือการเข้าใช้งานผ่านโปรโตคอลเอชทีทีพี (http protocol) และควบคุมการทำงานผ่านช่องทางการเชื่อมต่อระหว่างเว็บไซต์

2. โฟลวคอนโทรลเลอร์ (Flow Controller) เป็นเหมือนสมองในการสั่งการที่จะทำหน้าที่จัดการการส่งสกุลไฟล์ต่าง ๆ และบริหารตารางการดำเนินการของสกุลไฟล์

3. เอกซ์เทนชั่นส์ (Extensions) คือสกุลไฟล์ของนายฟายซึ่งมีความหลากหลายเป็นอย่างมาก

4. โฟลวไฟล์รีโพสิทอรี่ (FlowFile Repository) เป็นที่ใช้สำหรับเก็บสถานะของการติดตามการทำงานส่งไฟล์ต่าง

5. คอนเทนต์รีโพสิทอรี่ (Content Repository) เป็นที่บรรจุโฟลวไฟล์ที่แท้จริงทั้งหมดไว้

6. โพรวิแนนซ์รีโพซิทอรี่ (Provenance Repository) เป็นที่เก็บแหล่งกำเนิดข้อมูลเหตุการณ์ทั้งหมด

อาปาเช่ซอฟท์แวร์ฟาวน์เดชั่น (Apache Software Foundation) พัฒนาเทคโนโลยีนายฟายที่มีพื้นฐานมาจาก “Niagara Files” ที่ถูกพัฒนาและใช้ในสำนักงานความมั่นคงแห่งชาติของประเทศ

สหรัฐอเมริกาเมื่อ 8 ปีที่แล้ว ซึ่งการพัฒนาในช่วงแรกตั้งเป้าหมายให้มีความยืดหยุ่น เหมาะสม และครอบคลุมต่ออุปกรณ์ที่มี ตั้งแต่อุปกรณ์ขนาดเล็ก เช่น ราสเบอร์รี่พายไปจนถึงคลัสเตอร์ข้อมูลขององค์กร และการทำงานร่วมกันของเซิร์ฟเวอร์จำนวนมาก การเชื่อมต่อเน็ตเวิร์คที่มีอัตราการเปลี่ยนแปลงขึ้นลง จะส่งผลกระทบต่อการสื่อสารและการส่งข้อมูลแต่นายฟายสามารถปรับตัวได้ดีต่อสิ่งนี้ ทำให้การสื่อสารเป็นไปได้ดี

คุณสมบัติของนายฟายมีดังนี้

1. เป็นโปรแกรมสำหรับติดต่อกับผู้ใช้ในรูปแบบเว็บไซต์ ง่ายต่อการออกแบบและควบคุม

2. การกำหนดคุณสมบัติมีประสิทธิภาพสูง มีการการันตีการส่งข้อมูล ความหน่วงเวลาน้อย มีการจัดลำดับความสำคัญของงาน

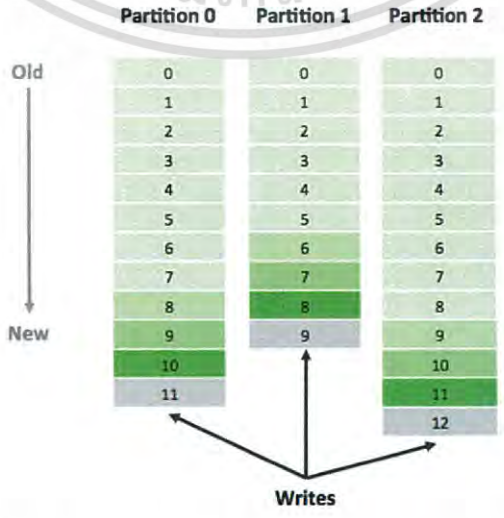
3. สามารถติดตามการเคลื่อนย้ายของข้อมูลตั้งแต่ต้นทางจนถึงปลายทางได้

4. ออกแบบได้ตามชนิดของไฟล์ที่เราต้องการ

5. มีความปลอดภัยสูง

2.7 อาฟาเช่คาฟกา (Apache Kafka)

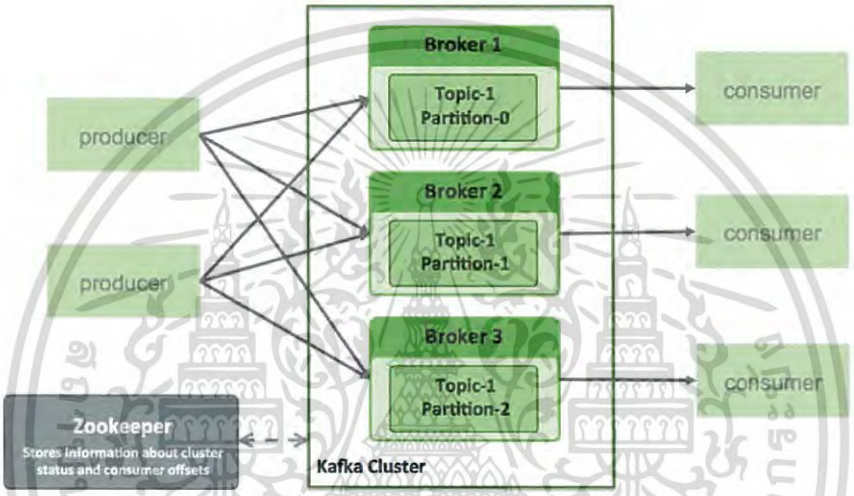
อาปาเช่คาฟกาเป็นระบบการส่งข้อความแบบพับลิชซับสไครบ์ (publish-subscribe) ที่มีความเร็ว ทนทาน สามารถขยายขนาดได้ สามารถทำงานต่อได้หากมีส่วนใดส่วนหนึ่งเสียหาย นิยมใช้ในการสื่อสารเช่น จาวาแมสเซจเซอร์วิส (Java Message Service) หรือแอดวานซ์แมสเซจคิวอิ้งโปรโตคอล (Advanced Message Queuing Protocol) เพราะความสามารถที่มีอัตราปริมาณงานที่ส่งได้สูง มีความน่าเชื่อถือ และสามารถทำซ้ำได้



ภาพที่ 2.2 แผนภาพการแบ่งส่วนของอาปาเช่คาฟกา [19]

จากภาพที่ 2.2 แสดงให้เห็นว่าการออกแบบระบบของคาฟกา ทำให้การเขียนข้อมูลบนดิสก์มีการเรียงลำดับอย่างเป็นขั้นตอนและการส่งไฟล์จะเป็นแบบเข้าก่อนออกก่อนเข้าหลังออกหลังตามลำดับ

การทำงานของคาฟกาเป็นการผสมผสานระหว่างอุปกรณ์สามตัวตั้งแต่เอชเบส สตอร์มและสปาร์กเพราะมีการวิเคราะห์ข้อมูลแบบเชิงลึกในทันทีและส่งข้อมูลแบบสตรีมมิ่งสตอร์มและเอชเบสจะทำงานได้ดีขึ้นเมื่อทำงานร่วมกับคาฟกาโดยจะมีเว็บไซต์สำหรับติดตามการทำงาน และมีกลุ่มข้อมูลจราจรคอมพิวเตอร์



ภาพที่ 2.3 แผนภาพตัวแทนของอาปาเช่คาฟกา [20]

จากภาพที่ 2.3 แสดงให้เห็นว่าคาฟกามี 4 ส่วนของโปรแกรมที่เกี่ยวข้องกับการการเคลื่อนย้ายเข้าและออกของข้อมูล ดังนี้

1. หัวเรื่อง (Topic) คือชื่อช่องทางที่จะทำการส่งข้อมูลหรือที่รู้จักกันในชื่อแชนแนล (Channel)

2. ผู้ผลิต (Producer) คือการส่งข้อมูลไปยังหัวเรื่องของคาฟกาซึ่งอาจส่งได้หนึ่งหรือหลายหัวเรื่องก็ได้

3. ผู้บริโภค (Consumer) คือผู้รับข้อมูลที่มาจากหัวเรื่องคาฟกาและประมวลผลข้อมูลที่รับมา นอกจากนี้ยังกำหนดรหัสที่ระบุข้อความในแต่ละส่วนโดยเฉพาะ เนื่องจากข้อมูลทั้งหมดนั้นเก็บบนดิสก์ ผู้บริโภคจึงลดปัญหาการการประมวลผลที่มีความเร็วไม่เท่ากันโดยการกระโดดข้ามไปในจุดใดจุดหนึ่งในแต่ละส่วนได้จากรหัสเฉพาะที่กำหนด

4. ตัวแทน (Broker) เนื่องจากคลัสเตอร์ของคาฟกาที่ประกอบด้วยหนึ่งหรือหลายเซิร์ฟเวอร์ จะมีตัวแทนที่บริหารการคงอยู่ของข้อมูลและมีการทำสำเนาข้อมูล มีการเรียงข้อมูลตามลำดับและไม่สามารถเปลี่ยนลำดับข้อมูลได้ทำให้ลดเวลาในการค้นหาความรับผิดชอบของตัวแทนที่กล่าวมานี้เป็นจุดเด่นในประสิทธิภาพการทำงานของคาฟกา

2.8 อาปาเช่สคูป (Apache Sqoop)

อาปาเช่สคูปเป็นเครื่องมือที่ถูกออกแบบมาเพื่อประสิทธิภาพในการส่งผ่านข้อมูลขนาดใหญ่ระหว่างฮาดูปและที่เก็บข้อมูลแบบมีโครงสร้าง เช่น ฐานข้อมูลเชิงสัมพันธ์ สคูปสามารถช่วยกำจัดงานหนัก เช่น การประมวลผลแบบ ETL จากคลังข้อมูลไปยังฮาดูปเพื่อประสิทธิภาพในการดำเนินงานด้วยต้นทุนที่น้อยลง นอกจากนี้สคูปยังสามารถดึงข้อมูลจากฮาดูปและนำข้อมูลออกมาใส่ในที่เก็บข้อมูลภายนอกแบบมีโครงสร้างได้ เช่น เทราดาต้า เน็ตทีซ่า ออราเคิล มายซีควล และโพสต์เกรซ

หน้าที่ของสคูปประกอบด้วย 7 อย่างดังนี้

1. นำเข้ากลุ่มข้อมูล (dataset) แบบเป็นลำดับและต่อเนื่องกันจากเมนเฟรม

2. นำเข้าข้อมูลโดยแปลงให้เป็นไฟล์ข้อความ มีการบีบอัดข้อมูลให้ดีขึ้น และทำดัชนีเพื่อประสิทธิภาพในการสืบค้นข้อมูล

3. นำข้อมูลเข้าและส่งข้อมูลออกได้พร้อมกันทำให้ประสิทธิภาพของระบบเร็วขึ้นและมีการใช้งานระบบได้เหมาะสมที่สุด

4. คัดลอกข้อมูลจากระบบภายนอกสู่ฮาดูปได้อย่างรวดเร็วเนื่องจากสคูปและฮาดูปถูกพัฒนาขึ้นมาใช้ร่วมกัน

5. เพิ่มประสิทธิภาพให้กับการวิเคราะห์ข้อมูลโดยรวมข้อมูลที่มีโครงสร้างและไม่มีโครงสร้างลงในทะเลข้อมูล

6. แบ่งและกระจายงานโดยย้ายที่เก็บข้อมูลและที่ประมวลผลที่มากเกินกว่าจะรับได้ไปยังระบบอื่น

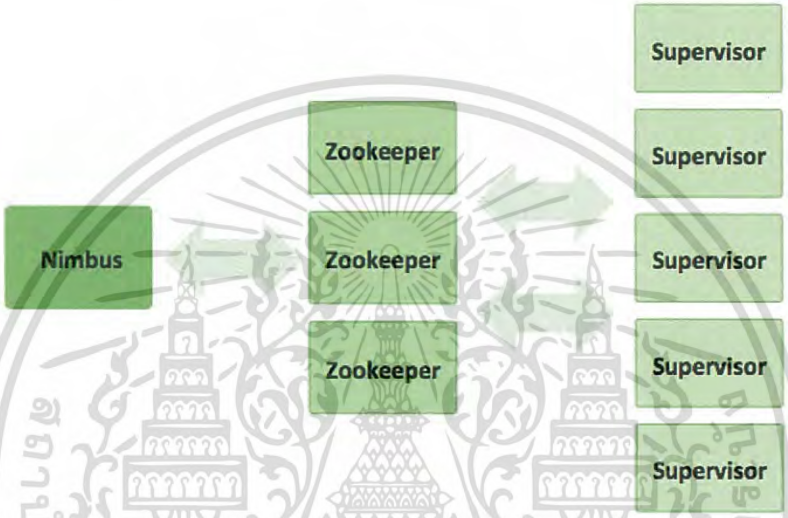
2.9 อาปาเช่สตอร์ม (Apache Storm)

อาปาเช่สตอร์มเป็นระบบประมวลผลข้อมูลปริมาณมากแบบกระจายในทันที มีความเร็วสูง สามารถประมวลผลมากกว่าล้านเอกสารต่อ 1 วินาที ต่อ 1 โหนดบนคลัสเตอร์ที่ขนาดพอประมาณ บริษัทใช้ประโยชน์จากข้อดีตรงนี้รวม Storm เข้ากับแอปพลิเคชันนำเข้าข้อมูลอื่น ๆ เพื่อป้องกันการเกิดเหตุการณ์ไม่คาดฝันหรือปรับผลลัพธ์ให้เหมาะสม ประกอบด้วยโหนด 3 กลุ่มดังนี้

1. โหนดนิมบัส หรือโหนดมาสเตอร์คล้ายกับ JobTracker ในฮาดูป สามารถกระจายโค้ดข้ามคลัสเตอร์ได้ ทำงานข้ามคลัสเตอร์ได้ เฝ้าสังเกตุผลที่ได้จากการคำนวณแล้วจัดสรรงานแต่ละผู้ทำงานตามต้องการ

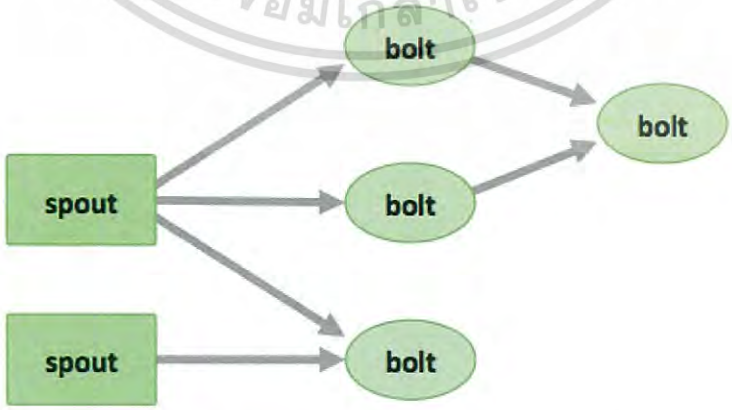
2. โหนดซูคีพเพอร์ (ZooKeeper Node) ประสานงานกับคลัสเตอร์ของสตอร์ม

3. โหนดซูเพอร์ไวเซอร์ (Supervisor Node) สื่อสารกับนิมบัสผ่านซูคีพเพอร์และกำหนดการเริ่มและหยุดทำงานของผู้ทำงานให้สอดคล้องกับสัญญาณที่ได้รับจากนิมบัส



ภาพที่ 2.4 คลัสเตอร์ของอาปาเช่สตอร์ม [21]

จากภาพที่ 2.4 แสดงคลัสเตอร์ของสตอร์มที่ประกอบด้วยโหนดนิมบัส โหนดซูคีพเพอร์และโหนดซูเพอร์ไวเซอร์ ซึ่งนิมบัสจะทำงานเป็นหัวใจหลักและทำงานเชื่อมต่อกับซูคิปเปอร์ ส่วนซูเพอร์ไวเซอร์จะเป็นส่วนภายนอกที่ต้องติดต่อผ่านซูคิปเปอร์



ภาพที่ 2.5 การประมวลผลข้อมูลของสตอร์ม [22]

จากภาพที่ 2.5 แสดงการประมวลผลข้อมูลของสตอร์มซึ่งทำให้การประมวลผลมีความรวดเร็วและเป็นแบบกระจายประกอบด้วย 5 ส่วนดังนี้

1. ทูเพิล (Tuples) เช่น “4 ทูเพิล” คือ (7 1 3 7)

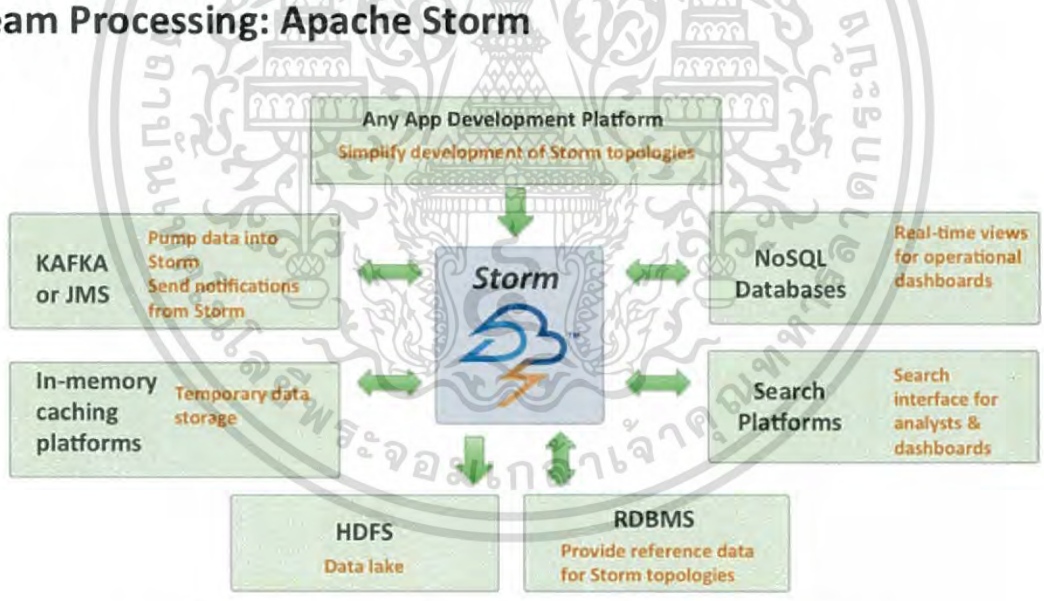
2. สตรีม (Streams) คือ ทูเพิลที่เกิดขึ้นอย่างต่อเนื่องและไม่มีขอบเขตจำกัด

3. สเปาท์ (Spouts) คือ แหล่งของสตรีมในการนำมาคำนวณหาผลลัพธ์ มันจะอ่านสตรีมจากคิวของตัวแทน เช่น แรบบิทเอ็มคิว คาฟกา แต่สเปาท์ก็สามารถสร้างสตรีมของตัวเองได้หรืออ่านจากที่อื่น เช่น Twitter streaming API และมีระบบการเข้าคิวของตัวเอง

4. โบลท์ (Bolts) จะประมวลผลค่าของสตรีมเข้าใด ๆ ก็ตามแล้วผลิตค่าของสตรีมออกใหม่ขึ้นแทน ส่วนใหญ่โลจิกต่าง ๆ ของการคำนวณอยู่ในส่วนนี้ เช่น ฟังก์ชัน ฟิลเตอร์ การรวมสตรีม (Streaming join) การสื่อสารกับฐานข้อมูล

5. โทโพโลยี (Topology) การคำนวณทั้งหมดทำให้เห็นถึงเน็ตเวิร์คของสเปาท์และโบลท์

ผู้ใช้สตอร์มจะเป็นคนกำหนดโทโพโลยีตั้งแต่สตรีมข้อมูลเข้าแล้วจะไปประมวลผลส่วนใดและผลลัพธ์ที่ได้จะเข้าไปสู่ฮาดูป



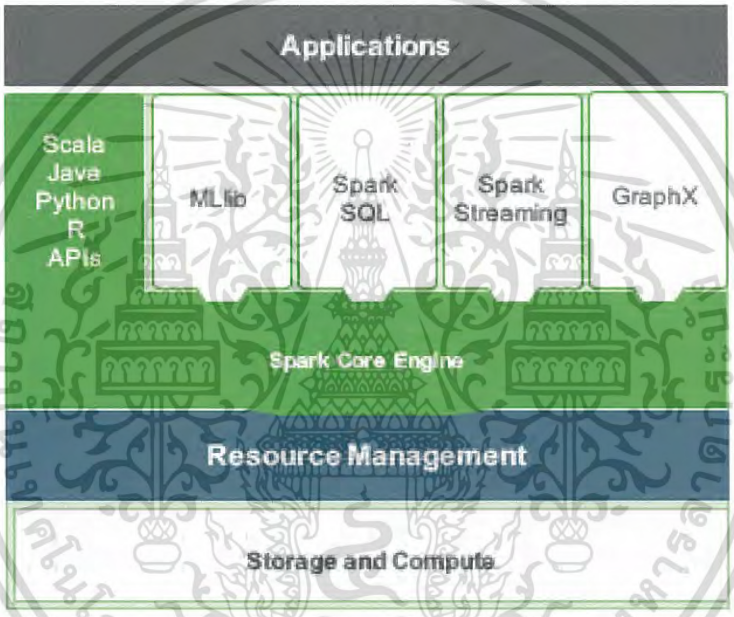
ภาพที่ 2.6 แสดงการประมวลผลแบบสตรีมมิ่งผ่านอาปาเช่สตอร์ม [23]

จากภาพที่ 2.6 แสดงให้เห็นว่าสตอร์มสามารถประมวลผลและส่งข้อมูลไปยังระบบอื่น ๆ ได้หลากหลาย ไม่ว่าจะเป็นในเครือของอาปาเช่ ได้แก่ เอชดีเอฟเอส คาฟกา หรือฐานข้อมูลเชิงสัมพันธ์ และรูปแบบแอปพลิเคชันต่าง ๆ

2.10 อาปาเช่สปาร์ก (Apache Spark)

อาปาเช่สปาร์กเป็นเครื่องมือที่ประมวลผลในตัวเองซึ่งมีช่องทางการเชื่อมระหว่างแอปพลิเคชันที่พัฒนามาอย่างดีเพื่อยอมให้ผู้ทำงานเกี่ยวกับข้อมูลสามารถเข้าถึงเซตข้อมูลได้อย่างรวดเร็วและมีประสิทธิภาพด้วยวิธีดำเนินการแบบสตรีม การเรียนรู้ของเครื่อง (Machine learning) หรือซีควล สปาร์กทำงานอยู่บนยาร์นทำให้นักพัฒนาสามารถเอาความสามารถของสปาร์กมาสร้างแอปพลิเคชันและพัฒนาให้ดีขึ้นโดยใช้เซตข้อมูลของฮาดูป

สปาร์กจะประกอบด้วยแกนของสปาร์กและเซตของไลบรารี แกนของสปาร์กจะเป็นเครื่องจักรที่ทำงานแบบกระจายและรองรับการสื่อสารด้วยภาษา จาวา สกาล่า และไพทอน ซึ่งจะเสนอแพลตฟอร์มของการพัฒนาแอปพลิเคชันดึงข้อมูลจากแหล่งข้อมูลต่าง ๆ เข้าสู่คลังข้อมูลแบบกระจาย



ภาพที่ 2.7 สถาปัตยกรรมของอาปาเช่สปาร์ก [24]

จากภาพที่ 2.7 จะเห็นได้ว่าสปาร์กถูกออกแบบมาเพื่อการทำวิทยาศาสตร์ข้อมูลได้ง่ายขึ้น นักวิทยาศาสตร์ส่วนใหญ่เลือกใช้การทำการเรียนรู้ของเครื่องคือเซตของเทคนิคและอัลกอริทึมที่สามารถเรียนรู้จากข้อมูลซึ่งอัลกอริทึมเหล่านี้มักจะถูกทำซ้ำอยู่บ่อย ๆ และสปาร์กสามารถที่จะแคชกลุ่มของข้อมูลในหน่วยความจำที่ใช้เป็นประจำหรือเคยเข้าใช้งานแล้วส่งผลให้มีการทำงานที่เร็วขึ้น และสปาร์กถือเป็นเครื่องจักรประมวลผลที่ดีที่สุดในการดำเนินงานด้านอัลกอริทึม อีกทั้งสปาร์กนั้นยังประกอบด้วย MLlib ซึ่งเป็นไลบรารีที่จัดหาเซตของอัลกอริทึมที่เป็นเทคนิคการทำวิทยาศาสตร์ข้อมูลเป็นหลักและมีครอบคลุมไลบราลีเกือบทุกประเภทที่ใช้ในการทำเหมืองข้อมูล ตัวอย่างเช่น การจำแนกประเภท การถดถอย การแบ่งกลุ่ม เป็นต้น

ความสามารถของสปาร์กที่เป็นจุดรวมเข้ากับฮาดูปมีดังนี้

1. รองรับรูปแบบไฟล์ ORC

2. มีความปลอดภัยสูง

3. การดำเนินงานสามารถจัดการและดูแลโดยใช้อาปาเช่อัมบารี

4. การเข้าถึงข้อมูลทำได้อย่างดีควบคู่ไปกับไฮฟ์

5. ทำงานบนยาร์นทำให้มีการปรับปรุงขนาดและความน่าเชื่อถือ

2.11 อาปาเช่อัมบารี (Apache Ambari)

อาปาเช่อัมบารีเป็นเครื่องมือที่จะช่วยทำให้การใช้งานฮาดูปง่ายขึ้นจึงมีการพัฒนาอัมบารีขึ้นมาเป็นซอฟต์แวร์ที่จัดหาเงื่อนไขในการติดตั้งฮาดูปมาให้ ใช้จัดการและดูแลคลัสเตอร์ของฮาดูป อีกทั้งยังมีการใช้งานที่ง่ายผ่านหน้าติดต่อประสานงาน การบริการระบบผ่านอัมบารีสามารถทำได้ดังนี้

1. การจัดหาคลัสเตอร์ของฮาดูปมาให้

- อัมบารีเสนอขั้นตอนการลงเซอร์วิสของฮาดูปท่ามกลางโฮสต์หลายเครื่องอย่างเป็นลำดับ
- อัมบารีดูแลการตั้งค่าสำหรับเซอร์วิสของฮาดูปในแต่ละคลัสเตอร์

2. การจัดการคลัสเตอร์ของฮาดูป

- อัมบารีมีการบริหารจากส่วนกลางสำหรับการเริ่ม หยุด และตั้งค่าใหม่ของเซอร์วิสของฮาดูปข้ามคลัสเตอร์

3. การดูแลคลัสเตอร์ของฮาดูป

- อัมบารีมีหน้ากระดานสำหรับดูและสุขภาพและสถานะของคลัสเตอร์ของฮาดูป
- อัมบารีใช้ประโยชน์จากระบบของอัมบารีเมตริกซ์สำหรับการรวมกันของเมตริกซ์
- อัมบารี ใช้ประโยชน์จากเฟรมเวิร์คการแจ้งเตือนสำหรับระบบ จะแจ้งเตือนเมื่อต้องการการักษาระบบ เช่น โหนดไม่ทำงาน พื้นที่ในดิสก์เหลือน้อย เป็นต้น

2.12 อีลาสติกเสิร์ช (Elasticsearch)

อีลาสติกเสิร์ชเป็นการทำงานแบบกระจาย สามารถเก็บข้อมูลและค้นคืนได้ ทำหน้าที่เป็นโปรแกรมสืบค้นข้อมูล รับค่าข้อมูลที่เป็นระบบแอนาลอกจากเซ็นเซอร์แล้วเปลี่ยนเป็นระบบดิจิตอลได้ และการวิเคราะห์ข้อมูลในเชิงลึกมีความปลอดภัยสูง ข้อมูลที่เก็บในอีลาสติกเสิร์ชจะเก็บในรูปแบบเจสันซึ่งเป็นระบบมาตรฐาน การเก็บข้อมูลจะต้องระบุชื่อดัชนี ชื่อประเภท และไอดีด้วยเพราะจะทำให้สืบค้นได้ง่าย แต่

ถ้าไม่ระบุ เครื่องมือจะใส่ไอดีให้โดยอัตโนมัติ ระบบสามารถขยายขนาดได้ ไม่จำเป็นต้องเริ่มที่หลายโหนด สามารถทำงานโดยเริ่มต้นที่โหนดเดียวก่อนได้แล้วค่อย ๆ เพิ่มโหนดขึ้นไปได้ หากต้องการเพิ่มพื้นที่ในการจัดเก็บและประมวลผลซึ่งง่ายต่อการจัดการ อีกทั้งยังสามารถค้นหาคำที่สะกดคล้ายคลึงกันได้ เช่น วรรณ กับ วัน นิยมใช้ร่วมกับคีบานาและลอคสแตช

วิธีการใส่ข้อมูลบนอีลาสติกเสิร์ชมี 2 แบบหลัก ๆ ดังนี้

1. เจสันบนเอชทีทีพี

ใช้คำสั่ง PUT Request เพื่อสร้างชื่อดัชนี ชื่อประเภท ตามด้วยไอดี แต่ถ้าไม่ใส่หมายเลขไอดีให้ใช้คำสั่ง POST แทน PUT

การเรียกข้อมูลโดยใช้หมายเลขไอดี สามารถทำได้โดยใช้คำสั่ง GET

curl –XGET http://localhost:9200/ชื่อดัชนี/ชื่อประเภท/หมายเลขไอดี

การลบเอกสารออกโดยใช้หมายเลขไอดี สามารถทำได้โดยใช้คำสั่ง DELETE

curl –XDELETE http://localhost:9200/ชื่อดัชนี/ชื่อประเภท/หมายเลขไอดี

2. เนทีฟไคลแอนท์ (Native client)

วิธีนี้ไม่ค่อยเป็นที่นิยม ส่วนใหญ่ผู้ใช้งานจะใช้วิธีแรกมากกว่า

2.13 ลอคสแตช (Logstash)

ลอคสแตชเป็นโอเพนซอร์สซอฟต์แวร์แพลตฟอร์มที่ใช้ในการเก็บข้อมูลโดยสามารถส่งผ่านข้อมูลแบบทันที สามารถเก็บข้อมูลที่หลากหลายและมีรูปแบบต่างกันโดยสิ้นเชิง แล้วนำมาทำให้เป็นมาตรฐานเดียวกันแบบที่ผู้ใช้ต้องการได้ สามารถทำความสะอาดและนำข้อมูลมาวิเคราะห์ในเชิงลึกและแสดงผลข้อมูลการทำงานของผู้ใช้ระบบได้ การทำงานสามารถรับข้อมูลเข้าในรูปแบบใดก็ได้แล้วนำมาแปลงเป็นอินพุตแบบแปลงอาร์เรย์ทั่ว ๆ ไป จากนั้นนำไปกรอง จะได้เอาต์พุตที่มีความสามารถเฉพาะอย่าง การลงโปรแกรมมีความสะดวกสบาย ไม่มีการตั้งค่าที่ยุ่งยาก สามารถใช้งานได้เลย รองรับภาษาจาวา และรูบี้ รองรับระบบปฏิบัติการวินโดวส์ ลินุกซ์ และ โอเอสเอ็ก ใช้ระยะเวลาในการนำเข้าข้อมูลประมาณ 5 ถึง 20 วินาที นิยมใช้งานร่วมกับอีลาสติกเสิร์ชและคีบานา

2.14 สกาล่า (Scala)

ภาษาสกาล่าเปิดตัวสู่สาธารณะตั้งแต่ปี 2004 ซึ่ง ณ ปัจจุบันก็ผ่านมากว่า 12 ปีแล้ว โดยลักษณะของภาษารองรับการเขียนโปรแกรมทั้งแบบ functional และ imperative ทั้งสองแบบ ตัวภาษา สกาล่า

เป็นภาษาที่ทำงานอยู่บนโปรแกรมที่ใช้อ่านคำสั่งจาวา ทำให้สามารถเรียกใช้งานไลบรารีทั้งหมดที่สามารถทำงานบนโปรแกรมที่ใช้อ่านคำสั่งจาวาได้ทันที

แต่ถึงอย่างนั้นนักพัฒนาก็เลือกที่จะเขียนไลบรารีขึ้นมาใหม่เพราะความต้องการทางด้านประสิทธิภาพมากกว่า หลักการเขียนของสกาล่าจะอิงกับหลักการของการเขียนโปรแกรมเชิงวัตถุในภาษาจาวา แต่จะมีรูปแบบของภาษาที่สั้นมากทำให้บางครั้งอาจจะอ่านยากเกินไปรวมไปถึงภาษาสกาล่าเองยินยอมให้ใช้อักขระพิเศษในการตั้งชื่อได้อีกด้วยทำให้การแก้บัคในบางครั้งเป็นเรื่องน่าปวดหัวอยู่พอสมควร แต่ตรงนี้ผมมองว่าเราแก้ปัญหาได้ด้วยการกำหนดมาตรฐานการโค้ดในการพัฒนาโปรแกรมแทน

อีกหนึ่งข้อที่สำคัญคือภาษาสกาล่ามีความเป็นภาษาที่ต้องกำหนดชนิดในการประกาศตัวแปรค่อนข้างมากทำให้ลดโอกาสเกิดข้อผิดพลาดในการพัฒนาโปรแกรมได้เป็นอย่างดี ในสมัยก่อนภาษาโปรแกรมเองก็มาพร้อมกับความสามารถต้องกำหนดชนิดในการประกาศตัวแปรอยู่แล้วเนื่องจากติดปัญหาด้านทรัพยากร

ในเวลาต่อมาทรัพยากรที่มากขึ้นของคอมพิวเตอร์ส่งผลให้เกิดการสร้างภาษาโปรแกรมที่มีภาษาพลวัตร (Dynamic typing) ทำให้การพัฒนาโปรแกรมเป็นเรื่องง่ายเพราะตัวภาษาจะทำหน้าที่คอยทำกำหนดชนิดในการประกาศตัวแปรให้เองโดยอัตโนมัติแต่นั้นก็สร้างปัญหาให้การพัฒนาระบบที่มีขนาดใหญ่หรือนักพัฒนาที่ขาดความชำนาญทำให้เกิดข้อผิดพลาดในโปรแกรมอยู่บ่อยครั้ง

จนตอนนี้โลกของการพัฒนาซอฟต์แวร์เปลี่ยนไปสู่ยุคของการใช้งานคลาวด์และเน้นการปล่อยบ่อยขึ้น ทำให้การพัฒนาโปรแกรมต้องการความรัดกุมในการทำงานมากขึ้นและใช้งานทรัพยากรน้อยลงเพราะคลาวด์คิดค่าใช้จ่ายตามปริมาณทรัพยากรที่ถูกใช้ในการประมวลผล ทำให้ภาษาโปรแกรมที่มีลักษณะเป็นการกำหนดชนิดในการประกาศตัวแปรเริ่มกลับมาได้รับความนิยมอีกครั้งหนึ่ง

2.15 วาแกรนต์ (Vagrant)

วาแกรนต์ เป็นโปรแกรมที่ใช้สร้างทรัพยากรในการเขียนโปรแกรมซึ่งคล้ายกับเวอร์ชวลบ้อกซ์ (VirtualBox) วีเอ็มแวร์ (VMware) หรือคือการสร้างเครื่องเสมือนขึ้นมาทำอะไรซักอย่างแต่ไม่ใช่แบบเต็มตัวแต่จะเอามาเฉพาะที่พอทำงานได้

ข้อดีของการใช้วาแกรนต์ เมื่อเราต้องการพัฒนาโปรแกรมบนแพลตฟอร์มอื่น ระบบปฏิบัติการอื่นที่เราไม่ได้ใช้ประจำก็สร้างวีเอ็มขึ้นมาพัฒนาทดสอบได้เลยโดยไม่ต้องยุ่งหรือระบบปฏิบัติการเครื่องเราไม่ต้องหาเครื่องจริงมาลงระบบปฏิบัติการด้วย ซึ่งสมัยนี้การลำบากหาเครื่องจริงมาทดสอบนั้นเป็นเรื่องเชยไปแล้ว เทคโนโลยีวีเอ็มไปไกลมากทำให้สามารถทำได้เกือบทุกอย่างของเครื่องจริงและปรับสเปคได้ยืดหยุ่น

ตามที่เราต้องการ นิยมเอามาใช้ในการพัฒนาซอฟต์แวร์เพราะมันปรับแต่งง่าย ไม่ซับซ้อน คำสั่งมีน้อยและการเชื่อมโฟลเดอร์และเชื่อมพอร์ทระหว่างโฮสต์กับผู้อื่นง่ายด้วย

บทที่ 3

การติดตั้งเครื่องมือและนำไปใช้

การดำเนินการในโครงงานโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึก ประกอบด้วยขั้นตอนดังนี้

1. การเลือกเครื่องมือที่เหมาะสมต่อการใช้งาน

2. การติดตั้งเครื่องมือ

3. การนำเครื่องมือไปใช้งาน

ในบทนี้จะนำเสนอการดำเนินงานบนระบบปฏิบัติการอูบุนตู (Ubuntu) ซึ่งหากคอมพิวเตอร์ใช้ระบบปฏิบัติการอื่น สามารถใช้วาแกรนต์ควบคู่กับเวอร์ชวลบ้อกซ์ เพื่อจำลองระบบปฏิบัติการอูบุนตูบนเครื่องเสมือนขึ้นมา

3.1 การเลือกเครื่องมือที่เหมาะสมต่อการใช้งาน

ก่อนเริ่มติดตั้งโครงสร้างข้อมูลขนาดใหญ่ทั้งหมด ผู้จัดทำได้ศึกษาความสามารถในการทำงานของเครื่องมือชนิดต่าง ๆ จำแนกออกเป็น 4 ประเภท ดังนี้

3.1.1 เครื่องมือนำเข้าข้อมูล

การนำเข้าข้อมูลมีทั้งรูปแบบที่มีโครงสร้าง รูปแบบที่ไม่มีโครงสร้าง รูปแบบกึ่งโครงสร้าง ทั้งนี้การทำงานของเครื่องมือแต่ละตัวจะมีความสามารถที่แตกต่างกันออกไป ได้แก่ อาปาเช่สคูป อาปาเช่คาฟกา อาปาเช่นายฟายและอีลาสติกเสิร์ช

3.1.2 เครื่องมือเก็บข้อมูล

เครื่องมือที่ใช้เก็บข้อมูลจะเป็นเครื่องมือหลักและเป็นตัวกลางในการเชื่อมต่อไปยังเครื่องมืออื่น ๆ ได้แก่ อาปาเช่ฮาดูปเอชดีเอฟเอส อาปาเช่เอชเบส อาปาเช่ไฮฟ์และอีลาสติกเสิร์ช

3.1.3 เครื่องมือประมวลผลข้อมูล

การประมวลผลข้อมูลมีทั้งการปะมวลแฟ้มคำสั่งรวมและการประมวลผลในทันที ซึ่งสิ่งที่เครื่องมือประมวลผลควรทำได้คือการประมวลผลแฟ้มคำสั่งรวมและการเลือกใช้ส่วนใหญ่ขึ้นอยู่กับความเร็วในการประมวลผลเนื่องจากข้อมูลมีขนาดใหญ่มากจึงต้องใช้การประมวลที่มีประสิทธิภาพและความเร็วสูง ได้แก่ อาปาเช่ฮาดูปแมพรีดิวซ์ อาปาเช่สปาร์ก อาปาเช่ฟลิงค์และอาปาเช่สตอร์ม

3.1.4 เครื่องมือแสดงผลข้อมูล

เครื่องมือแสดงผลที่เหมาสำหรับการใช้งานข้อมูลขนาดใหญ่คือคีบานา

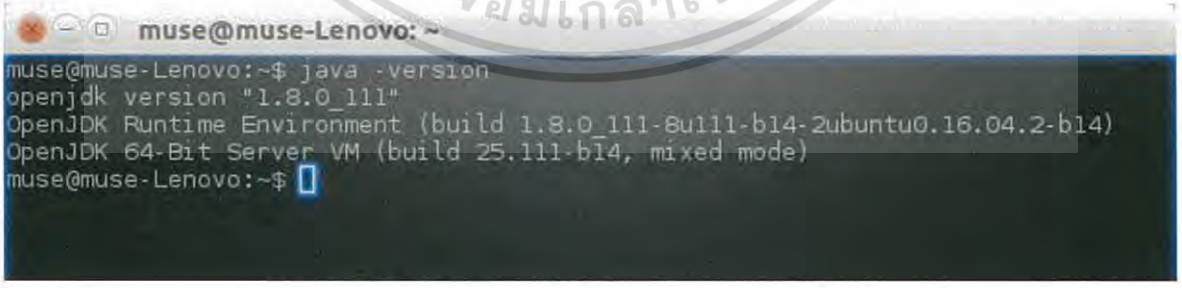
3.2 การติดตั้งเครื่องมือ

หลังจากศึกษาความสามารถในการทำงานของเครื่องมือต่าง ๆ แล้ว ผู้จัดทำได้ดำเนินงานติดตั้งเครื่องมือแต่ละชนิดที่ถูกคัดเลือกแล้วว่าเหมาะสมต่อการนำไปใช้งานต่อข้อมูลน้ำมันและก๊าซของบริษัท ซึ่งมีขั้นตอนการติดตั้ง ดังนี้

3.2.1 อาปาเช่ฮาดูป

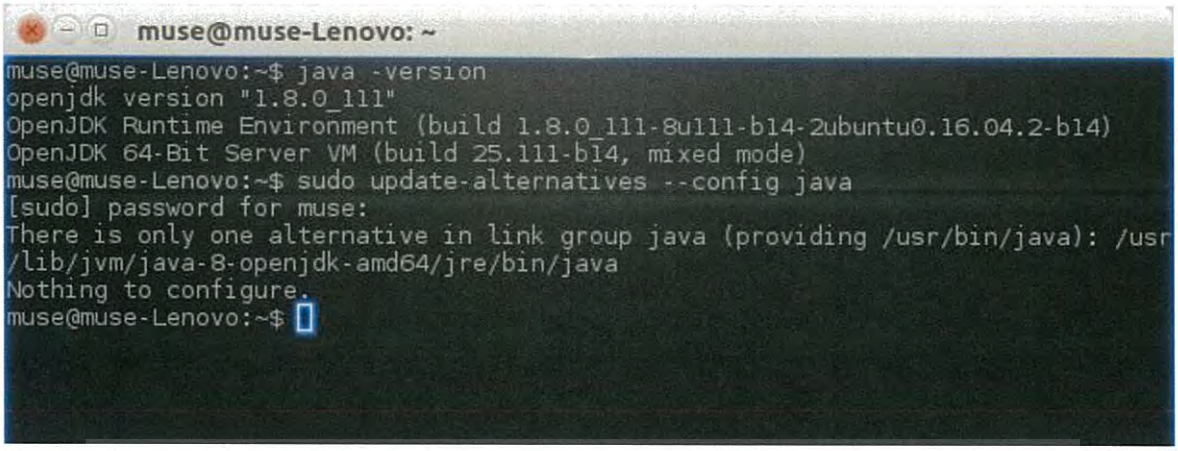
3.2.1.1 ติดตั้งจาวา

ในขั้นแรกต้องมีการเข้าไปเป็น root ก่อน จากนั้นจึงมีการอัพเดตและติดตั้งจาวาตามคำสั่ง sudo su – จากนั้น apt-get update และ apt-get install default-jdk



ภาพที่ 3.1 ผลลัพธ์คำสั่ง java –version

มีการตรวจสอบเวอร์ชั่นของจาวา หากลงแล้วจะขึ้นผลลัพธ์ดังภาพที่ 3.1จากนั้นจึงตั้งค่าJAVA_HOME

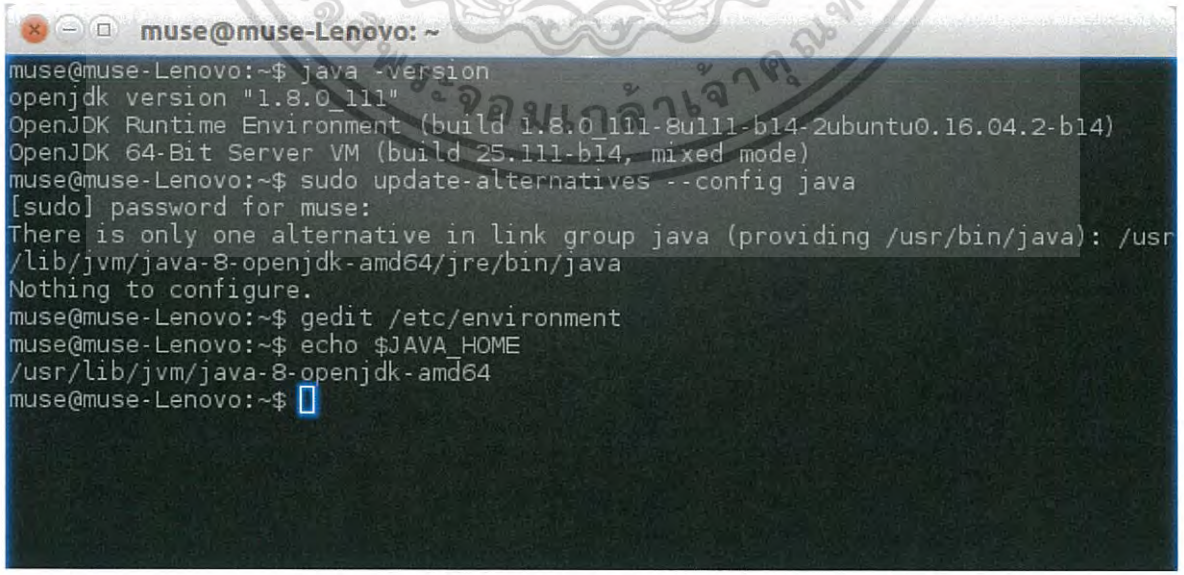


ภาพที่ 3.2 ผลลัพธ์คำสั่ง sudo update-alternatives --config java

จากภาพที่ 3.2 เมื่อใช้คำสั่ง sudo update-alternatives --config java แล้วให้คัดลอกเส้นทางที่ได้และไปแก้ไขในไฟล์ /etc/environment

ภาพที่ 3.3 หน้าต่างของ /etc/environment

จากภาพที่ 3.3 ให้เพิ่ม export JAVA_HOME = แล้วใส่เส้นทางที่คัดลอกได้ แต่ให้ลบ /jre/bin/java ออกไป จากนั้นบันทึกและรีโหลดไฟล์ด้วยคำสั่ง source /etc/environment



ภาพที่ 3.4 ผลลัพธ์คำสั่ง echo $JAVA_HOME

จากภาพที่ 3.4 เป็นการทดสอบการตั้งค่า JAVA_HOME

3.2.1.2 สร้างและติดตั้ง SSH

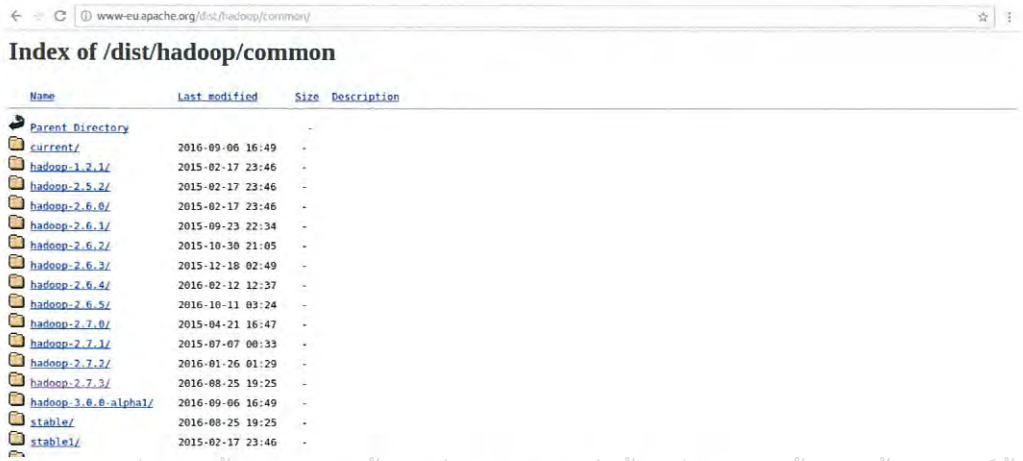
ขั้นตอนการสร้าง SSH ต้องมีการคัดลอกกุญแจไปยังไฟล์โดยเมื่อพิมพ์คำสั่ง ssh-keygen -t rsa -P '' หากถูกถามให้ระบุตำแหน่งไฟล์ให้กด Enter จากนั้นใช้คำสั่ง cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

3.2.1.3 ติดตั้งฮาดูป

สร้างโฟลเดอร์เพื่อเก็บไฟล์ที่ติดตั้งทั้งหมดโดยใช้คำสั่ง mkdir hadoopenv และสามารถดาวน์โหลดไฟล์ได้จาก http://www.apache.org/dyn/closer.cgi/hadoop/common/ หรือ wget http://www-eu.apache.org/dist/hadoop/common/hadoop-2.7.3/hadoop-2.7.3.tar.gz

ภาพที่ 3.5 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดฮาดูป

จากภาพที่ 3.5 แสดงหน้าเว็บดาวน์โหลดไฟล์ฮาดูปซึ่งควรเลือกตำแหน่งลิงก์ที่มีลูกศรชี้



ภาพที่ 3.6 หน้าเว็บไซต์ไฟล์ฮาดูป

จากภาพที่ 3.6 แสดงไฟล์ของฮาดูปเวอร์ชั่นต่าง ๆ ซึ่งแนะนำให้โหลดเวอร์ชั่นล่าสุด จากนั้นแตกไฟล์โดยใช้คำสั่ง tar xfz Downloads/hadoop-2.7.3.tar.gz และย้ายไปใส่ในโฟลเดอร์ hadoopenv โดยใช้คำสั่ง mv Downloads/hadoop-2.7.3 hadoopenv/ ซึ่งชื่อของไฟล์จะเป็นไปตามชื่อของโฟลเดอร์ที่ดาว์นโหลดมาทำให้อาจแตกต่างกันตามเวอร์ชั่นที่ดาวน์โหลดได้ หากชื่ออ่านยากให้แก้ไขให้เป็นไปตามที่ต้องการเพื่อการเข้าไปใช้งานในโฟลเดอร์ที่ง่ายขึ้น

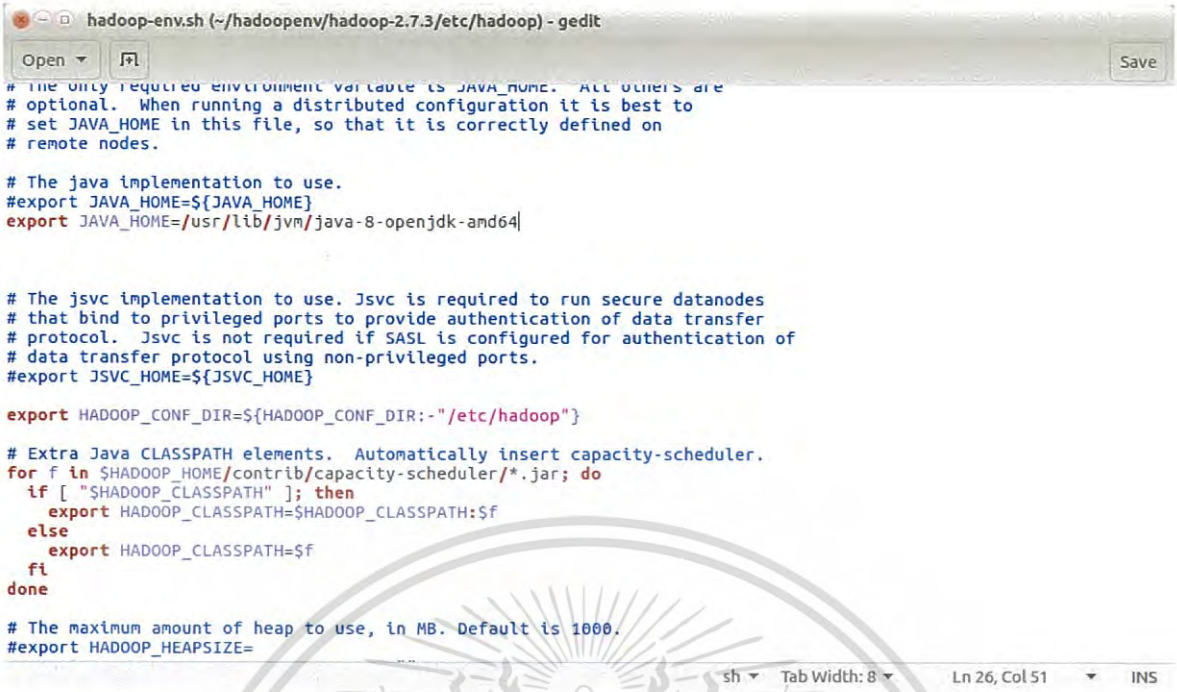
3.2.1.4 แก้ไขและตั้งค่าไฟล์ดังนี้

1. ไฟล์~/.bashrc ให้คัดลอกผลลัพธ์ที่ได้จากคำสั่ง echo $JAVA_HOME เพื่อในไปตั้งค่าใน ~/.bashrc จากนั้นเปิดไฟล์ ~/.bashrc เพื่อแก้ไขโดยใช้คำสั่ง gedit ~/.bashrc

ภาพที่ 3.7 ตัวแปรฮาดูป

จากภาพที่ 3.7 ให้นำตัวแปรฮาดูปไปใส่ในไฟล์ ~/.bashrc แล้วบันทึกและรีโหลดไฟล์ใหม่อีกครั้งด้วยคำสั่ง source ~/.bashrc

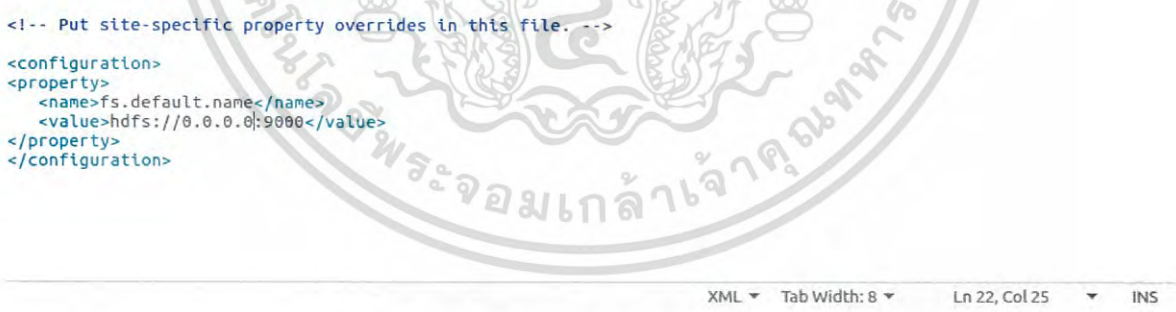
2. ไฟล์ hadoop-env.sh ให้เปิดไฟล์ hadoop-env.sh เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hadoop-2.7.3/etc/hadoop/hadoop-env.sh จากนั้นเข้าไปตั้งค่า JAVA_HOME ในไฟล์ hadoop-env.sh เพื่อให้ไฟล์ทราบตำแหน่งของไฟล์จาวาและสามารถอ่านค่าได้



ภาพที่ 3.8 การตั้งค่า $JAVA_HOME ใน hadoop-env.sh

จากภาพที่ 3.8 เป็นการตั้งค่า JAVA_HOME ใน hadoop-env.sh

3. ไฟล์ core-site.xml ทำการเปิดไฟล์ core-site.xml เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hadoop-2.7.3/etc/hadoop/core-site.xml ซึ่งเป็นการตั้งค่าชื่อและพอร์ตของฮาดูปซึ่งนิยมใช้ที่พอร์ต 9000 เป็นสากล



ภาพที่ 3.9 การตั้งค่าใน core-site.xml

จากภาพที่ 3.9 เป็นการแก้ไขข้อความระหว่าง<configuration></configuration>

4. ไฟล์ yarn-site.xml ทำการเปิดไฟล์ yarn-site.xml เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hadoop-2.7.3/etc/hadoop/yarn-site.xml

```
-->
<configuration>
<property>
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
</property>
<property>
   <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
   <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<!-- Site specific YARN configuration properties -->

</configuration>
```

XML ▾ Tab Width: 8 ▾ Ln 23, Col 12 ▾ INS

ภาพที่ 3.10 การตั้งค่าใน yarn-site.xml

จากภาพที่ 3.10 เป็นการแก้ไขข้อความระหว่าง<configuration></configuration>

5. ไฟล์ mapred-site.xml ทำการคัดลอก mapred-site.xml.template ด้วยคำสั่ง cp hadoopenv/hadoop-2.7.3/etc/hadoop/mapred-site.xml.template hadoopenv/ hadoop-2.7.3/etc/hadoop/mapred-site.xml แล้วสร้าง mapred-site.xml เพื่อตั้งค่าใหม่เพราะไฟล์เก่าควรเก็บไฟล์ก่อนแก้ไขไว้ โดยเปิดไฟล์ mapred-site.xml เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hadoop-2.7.3/etc/hadoop/mapred-site.xml

```
<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
   <name>mapreduce.framework.name</name>
   <value>yarn</value>
</property>
</configuration>
```

XML ▾ Tab Width: 8 ▾ Ln 23, Col 12 ▾ INS

ภาพที่ 3.11 การตั้งค่าใน mapred-site.xml

จากภาพที่ 3.11 เป็นการแก้ไขข้อความระหว่าง<configuration></configuration>

6. ไฟล์ hdfs-site.xml ทำการสร้างโฟลเดอร์ 2 โฟลเดอร์ สำหรับ namenode 1 โฟลเดอร์จากคำสั่ง mkdir -p hadoopenv/hadoop-2.7.3/hadoop_store/hdfs/namenode และ datanode จากคำสั่ง mkdir -p hadoopenv/hadoop-2.7.3/hadoop_store/hdfs/datanode จากนั้นเปิดไฟล์hdfs-site.xml เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hadoop-2.7.3/etc/hadoop/hdfs-site.xml

```
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
   <name>dfs.replication</name>
   <value>1</value>
 </property>
 <property>
   <name>dfs.namenode.name.dir</name>
   <value>file:/home/muse/hadoopenv/hadoop-2.7.3/hadoop_store/hdfs/datanode</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/home/muse/hadoopenv/hadoop-2.7.3/hdfs/datanode</value>
 </property>
</configuration>
```

XML ▾ Tab Width: 8 ▾ Ln 30, Col 50 ▾ INS

ภาพที่ 3.12 การตั้งค่าใน hdfs-site.xml

จากภาพที่ 3.12 เป็นการแก้ไขข้อความระหว่าง<configuration></configuration> และระบบไฟล์ของฮาดูปต้องถูกฟอร์แมตก่อนด้วยคำสั่ง sudo hadoopenv/hadoop-2.7.3/bin/hdfs namenode –format ถึงจะสามารถใช้งานได้

```
muse@muse-Lenovo: ~
muse@muse-Lenovo:~$ hadoopenv/hadoop-2.7.3/sbin/start-dfs.sh
17/03/13 18:41:39 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [0.0.0.0]
0.0.0.0: Ubuntu 16.04.1 LTS
0.0.0.0: starting namenode, logging to /home/muse/hadoopenv/hadoop-2.7.3/logs/ha
doop-muse-namenode-muse-Lenovo.out
localhost: Ubuntu 16.04.1 LTS
localhost: starting datanode, logging to /home/muse/hadoopenv/hadoop-2.7.3/logs/
hadoop-muse-datanode-muse-Lenovo.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: Ubuntu 16.04.1 LTS
0.0.0.0: starting secondarynamenode, logging to /home/muse/hadoopenv/hadoop-2.7.
3/logs/hadoop-muse-secondarynamenode-muse-Lenovo.out
17/03/13 18:41:55 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
muse@muse-Lenovo:~$
```

ภาพที่ 3.13 ผลลัพธ์คำสั่ง start-dfs.sh

จากภาพที่ 3.13 เป็นการเปิดใช้งานเอชดีเอฟเอชด้วยคำสั่ง start-dfs.sh

```
muse@muse-Lenovo: ~
muse@muse-Lenovo:~$ hadoopenv/hadoop-2.7.3/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/muse/hadoopenv/hadoop-2.7.3/logs/yarn
-muse-resourcemanager-muse-Lenovo.out
localhost: Ubuntu 16.04.1 LTS
localhost: starting nodemanager, logging to /home/muse/hadoopenv/hadoop-2.7.3/lo
gs/yarn-muse-nodemanager-muse-Lenovo.out
muse@muse-Lenovo:~$
```

ภาพที่ 3.14 ผลลัพธ์คำสั่ง start-yarn.sh

จากภาพที่ 3.14 เปิดใช้งาน Yarn ด้วยคำสั่ง start-yarn.sh

ภาพที่ 3.15 แสดงผลลัพธ์คำสั่ง jps

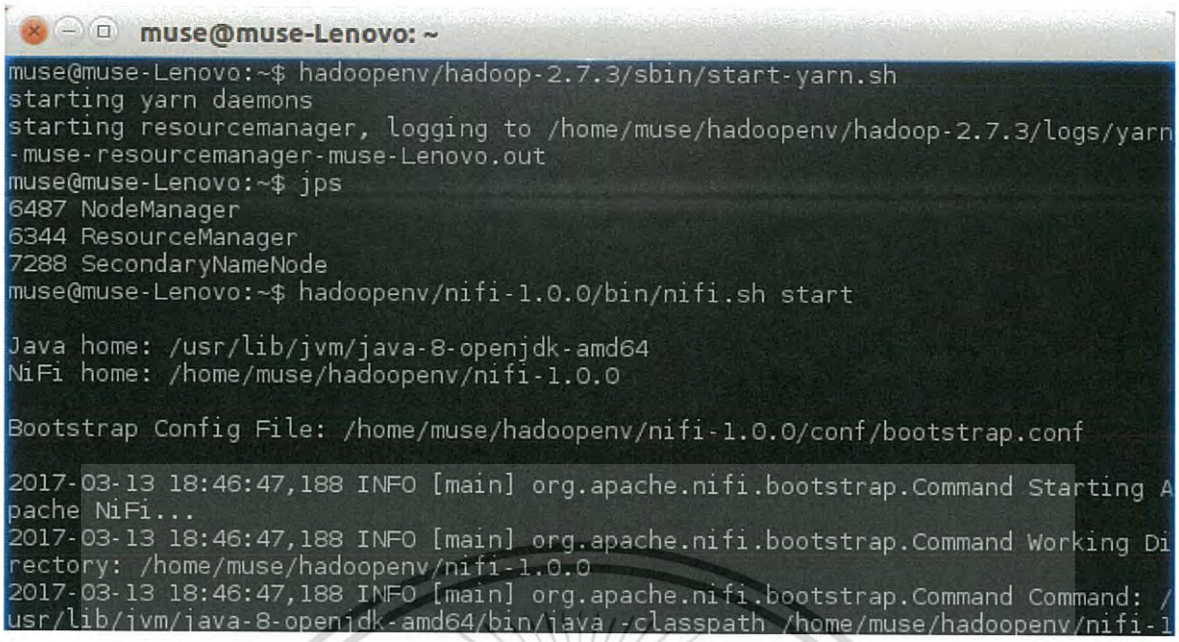
ภาพที่ 3.15 เมื่อใช้คำสั่ง jps แล้ว จะแสดงให้เห็นเลขพอร์ทต่าง ๆ ของแต่ละโหนด

3.2.2 อาปาเช่นายฟาย

สามารถดาวน์โหลดไฟล์ได้ที่ http://nifi.apache.org/download.html หรือ wget https://www.apache.org/dyn/closer.lua?path=/nifi/1.1.0/nifi-1.0.0-bin.tar.gz

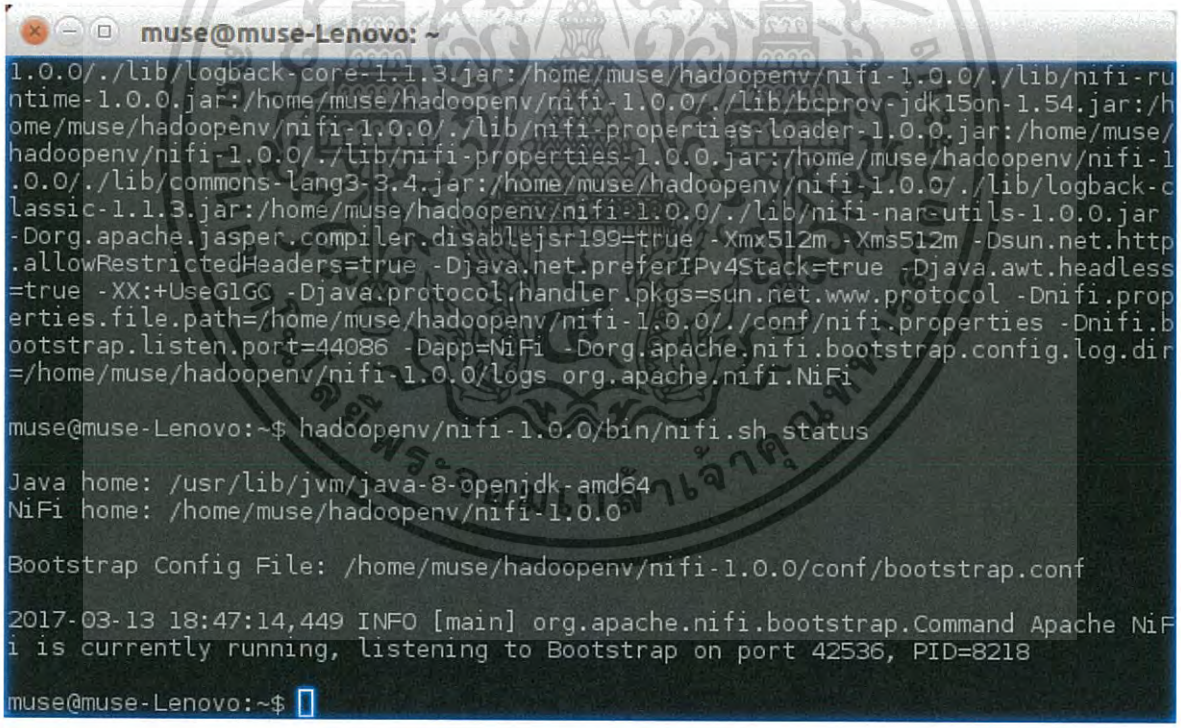
ภาพที่ 3.16 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดนายฟาย

จากภาพที่ 3.16 เป็นหน้าเว็บไซต์ดาวน์โหลดไฟล์นายฟายควรเลือกไฟล์นามสกุล .tar.gz จากนั้นแตกไฟล์ด้วยคำสั่ง tar xfz Downloads/nifi-1.0.0-bin.tar.gz และย้ายไปใส่ในโฟลเดอร์ hadoopenv ด้วยคำสั่ง mv nifi-1.0.0-bin.tar.gz/ hadoopenv/



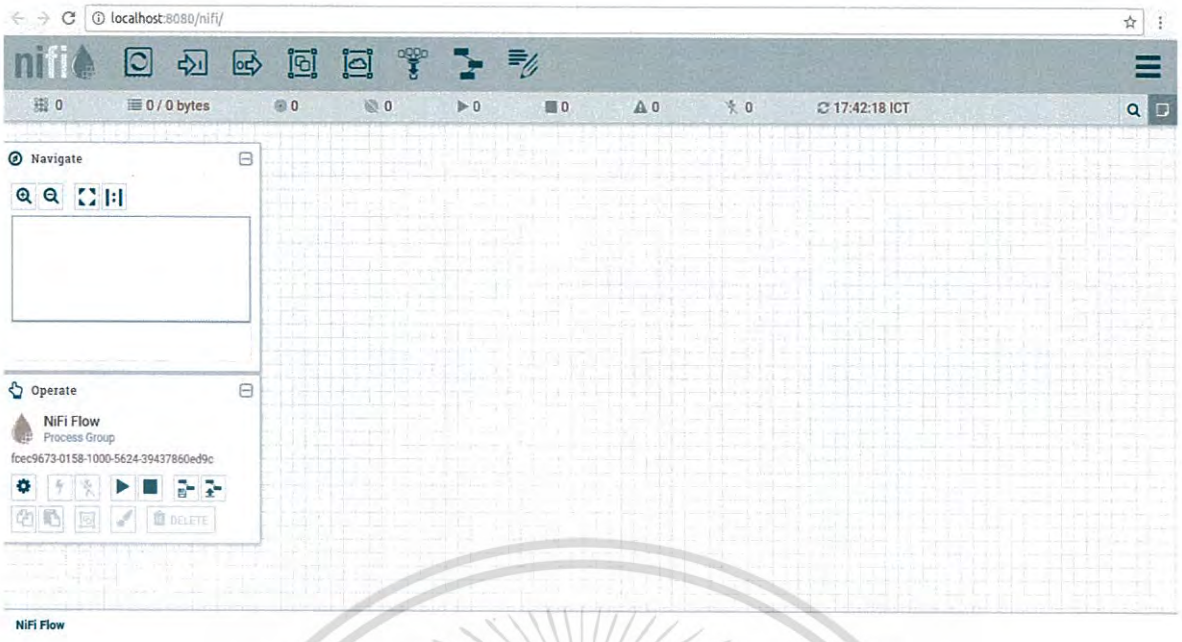
ภาพที่ 3.17 ผลลัพธ์คำสั่ง nifi.sh start

จากภาพที่ 3.17 เปิดใช้งานนายฟายด้วยคำสั่ง nifi.sh start



ภาพที่ 3.18 ผลลัพธ์คำสั่ง nifi.sh status

จากภาพที่ 3.18 ดูสถานะการทำงานของนายฟายด้วยคำสั่ง nifi.sh status

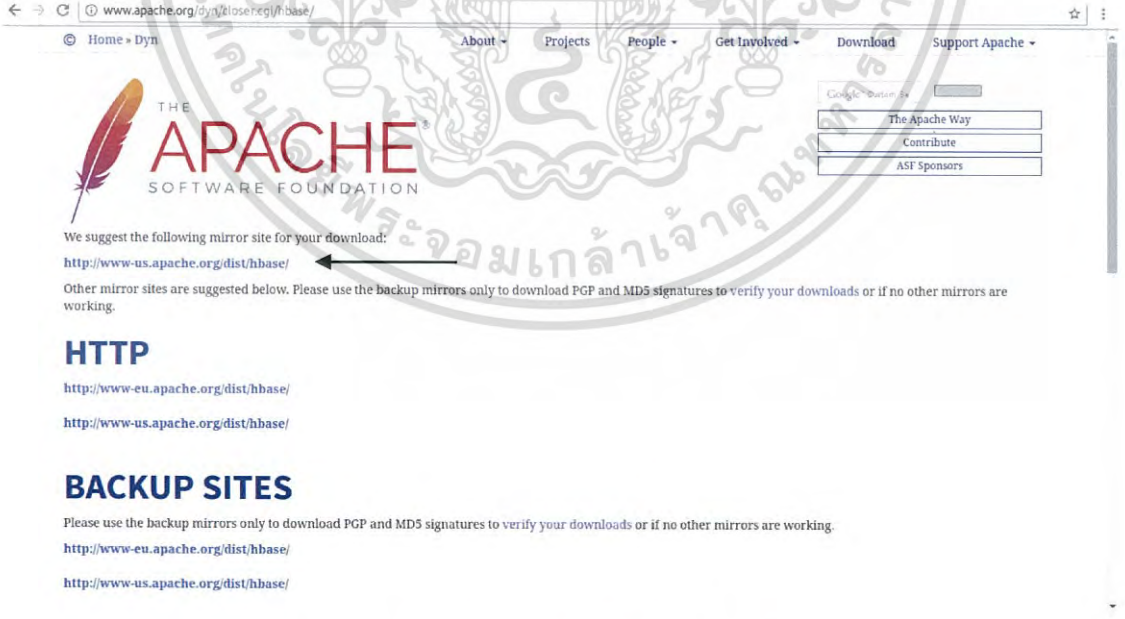


ภาพที่ 3.19 หน้าเว็บอินเตอร์เฟสระหว่างนายฟายและผู้ใช้งาน

จากภาพที่ 3.19 เข้าใช้งานนายฟายผ่านหน้าเว็บอินเตอร์เฟสที่พอร์ท 8080

3.2.3 อาปาเช่เอชเบส

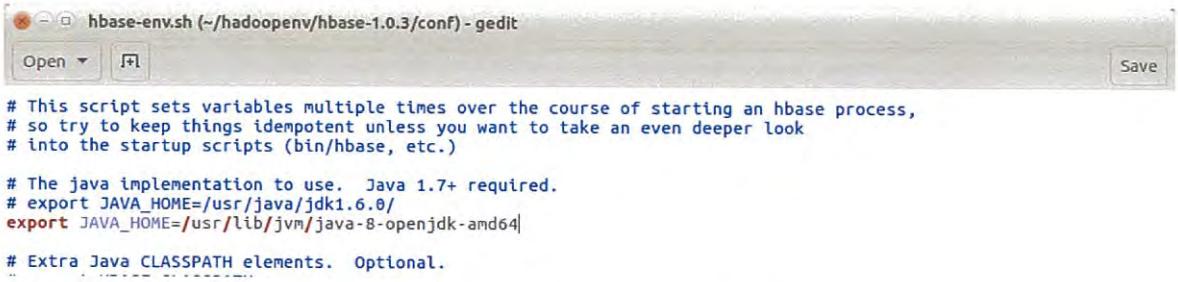
ดาวน์โหลดไฟล์ที่ http://www.apache.org/dyn/closer.cgi/hbase/ หรือ wget http://www-us.apache.org/dist/hbase/hbase-1.0.3/



ภาพที่ 3.20 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดเอชเบส

จากภาพที่ 3.20 แสดงหน้าเว็บดาวน์โหลดไฟล์เอชเบสซึ่งควรเลือกตำแหน่งลิงก์ที่มีลูกศรชี้ จากนั้นทำการแตกไฟล์ด้วยคำสั่ง hadoopenv ด้วยคำสั่ง tar xfz Downloads/hbase-1.0.3-

bin.tar.gz และย้ายไปใส่ในโฟลเดอร์ด้วยคำสั่ง mv hbase-1.0.3/ hadoopenv/ แล้วเปิดไฟล์ hbase-env.sh เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hbase-1.0.3/conf/hbase-env.sh

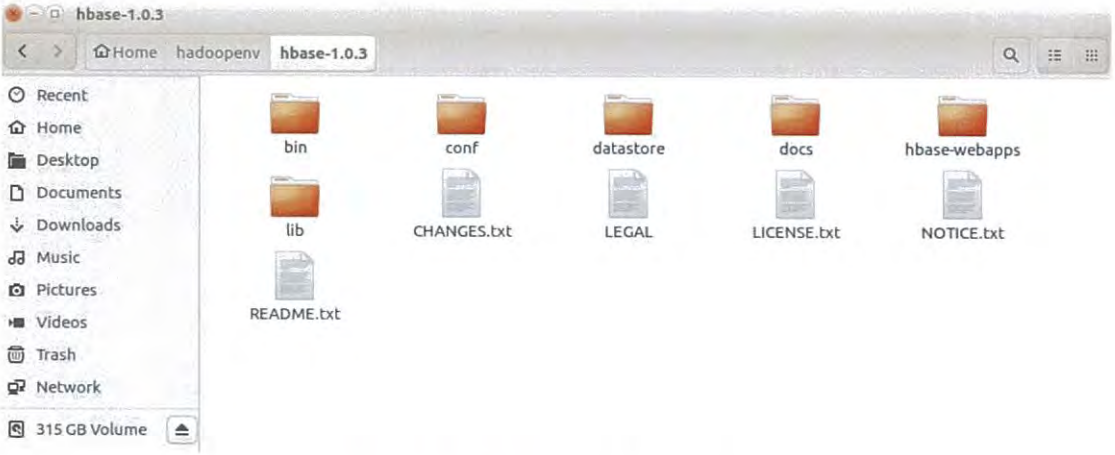


ภาพที่ 3.21 $JAVA_HOME ใน hbase-env.sh

จากภาพที่ 3.21 เป็นการตั้งค่า JAVA_HOME โดยใส่ที่อยู่ไฟล์ของจาวาจากนั้นเปิดไฟล์ ~/.bashrc เพื่อตั้งค่าเอชเบส

ภาพที่ 3.22 ตัวแปรเอชเบสที่ใส่ใน ~/.bashrc

จากภาพที่ 3.22 เป็นการตั้งค่าเอชเบสโดยใส่ HBASE_HOME และ PATH จากนั้นสร้างโฟลเดอร์สำหรับเก็บข้อมูลจากเอชเบสด้วยคำสั่ง mkdir hadoopenv/hbase-1.0.3/datastore



ภาพที่ 3.23 โฟลเดอร์ datastore ในเอชเบส

จากภาพที่ 3.23 เป็นผลลัพธ์การสร้างโฟลเดอร์สำหรับเก็บข้อมูลจากเอชเบส ชื่อ datastore จากนั้นเปิดไฟล์ hbase-site.xml เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/hbase-1.0.3/conf/hbase-site.xml

```
 */
-->
<configuration>
<property>

<name>hbase.rootdir</name>

<value>file:///home/muse/hadoopenv/hbase-1.0.3/datastore</value>

</property>

<property>

<name>hbase.zookeeper.property.dataDir</name>

<value>/home/hduser/HBASE/zookeeper</value>

</property>
</configuration>
```

XML ▾ Tab Width: 8 ▾ Ln 38, Col 12 ▾ INS

ภาพที่ 3.24 การตั้งค่าใน hbase-site.xml

จากภาพที่ 3.24 เป็นการแก้ไขข้อความระหว่าง<configuration></configuration> จากนั้นเปิดใช้งานเอชเบสด้วยคำสั่ง hadoopenv/hbase-1.0.3/bin/start-hbase.sh

3.2.4 อาปาเช่ไฮฟ์

ทำการแตกไฟล์ด้วยคำสั่ง tar xfz Downloads/apache-hive-1.2.1-bin.tar.gz และย้ายไปใส่ในโฟลเดอร์ hadoopenv ด้วยคำสั่ง mv apache-hive-1.2.1-bin/ hadoopenv/ แล้วจึงเปิดไฟล์ ~/.bashrc ด้วยคำสั่ง gedit ~/.bashrc เพื่อตั้งค่าไฮฟ์

```
fi

#HADOOP VARIABLES START
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_INSTALL=/home/muse/hadoopenv/hadoop-2.7.3
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
#HADOOP VARIABLES END

#HBASE VARIABLES START
export HBASE_HOME=/home/muse/hadoopenv/hbase-1.0.3
export PATH=$PATH:$HBASE_HOME/bin
#HBASE VARIABLES END

#HIVE VARIABLES START
export HIVE_HOME=/home/muse/hadoopenv/apache-hive-1.2.1-bin
export PATH=$PATH:$HIVE_HOME/bin
#HIVE VARIABLES STOP
```

ภาพที่ 3.25 ตัวแปรไฮฟ์ที่ใส่ใน ~/.bashrc

จากภาพที่ 3.25 เป็นการตั้งค่า HIVE_HOME และ PATH จากนั้นรีโหลดไฟล์ ~/.bashrc ด้วยคำสั่ง source ~/.bashrc แล้วเปิดไฟล์ hive-config.sh เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/apache-hive-1.2.1-bin/bin/hive-config.sh

ทำการใส่ HADOOP_HOME ใต้ข้อความข้างล่างนี้

```
# Allow alternate conf dir location.

        HIVE_CONF_DIR="${HIVE_CONF_DIR:-$HIVE_HOME/conf}"

        export HIVE_CONF_DIR=$HIVE_CONF_DIR

        export HIVE_AUX_JARS_PATH=$HIVE_AUX_JARS_PATH
```

```
# Allow alternate conf dir location.
HIVE_CONF_DIR="${HIVE_CONF_DIR:-$HIVE_HOME/conf}"

export HIVE_CONF_DIR=$HIVE_CONF_DIR
export HIVE_AUX_JARS_PATH=$HIVE_AUX_JARS_PATH
export HADOOP_HOME=/home/muse/hadoopenv/hadoop-2.7.3

# Default to use 256MB
export HADOOP_HEAPSIZE=${HADOOP_HEAPSIZE:-256}
```

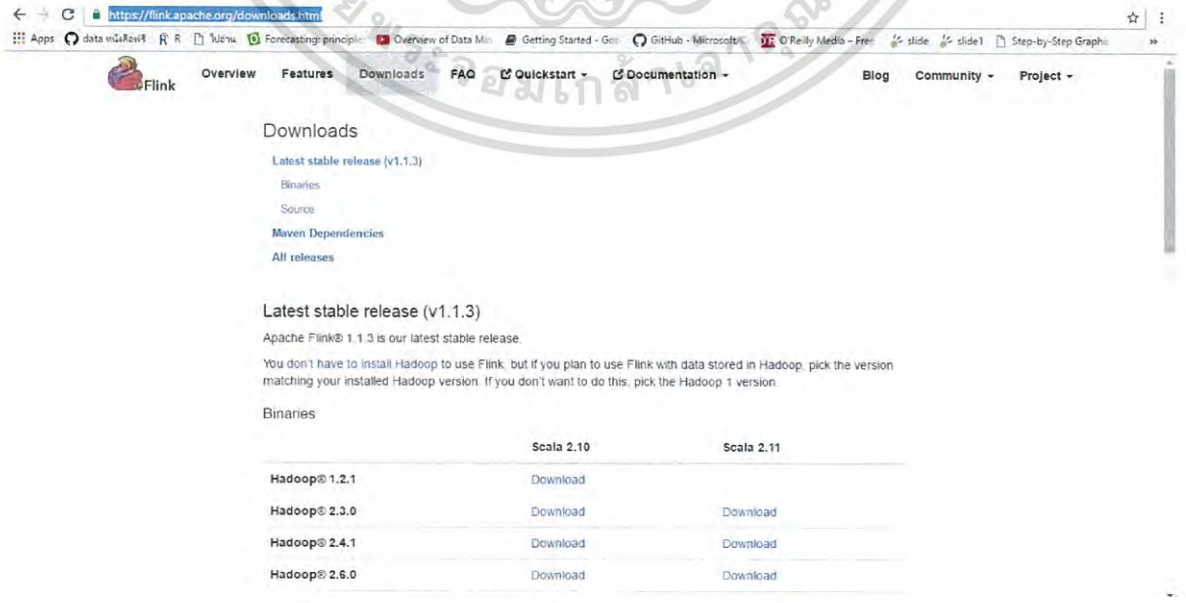
sh ▾ Tab Width: 8 ▾ Ln 54, Col 14 ▾ INS

ภาพที่ 3.26 การตั้งค่าใน hive-config.sh

จากภาพที่ 3.26 เป็นการตั้งค่า HADOOP_HOME จากนั้นเปิดคอมมานด์ไลน์ของไฮฟ์ด้วยคำสั่ง hadoopenv/apache-hive-1.2.1-bin/bin/hive

3.2.5 อาปาเช่ฟลิงค์

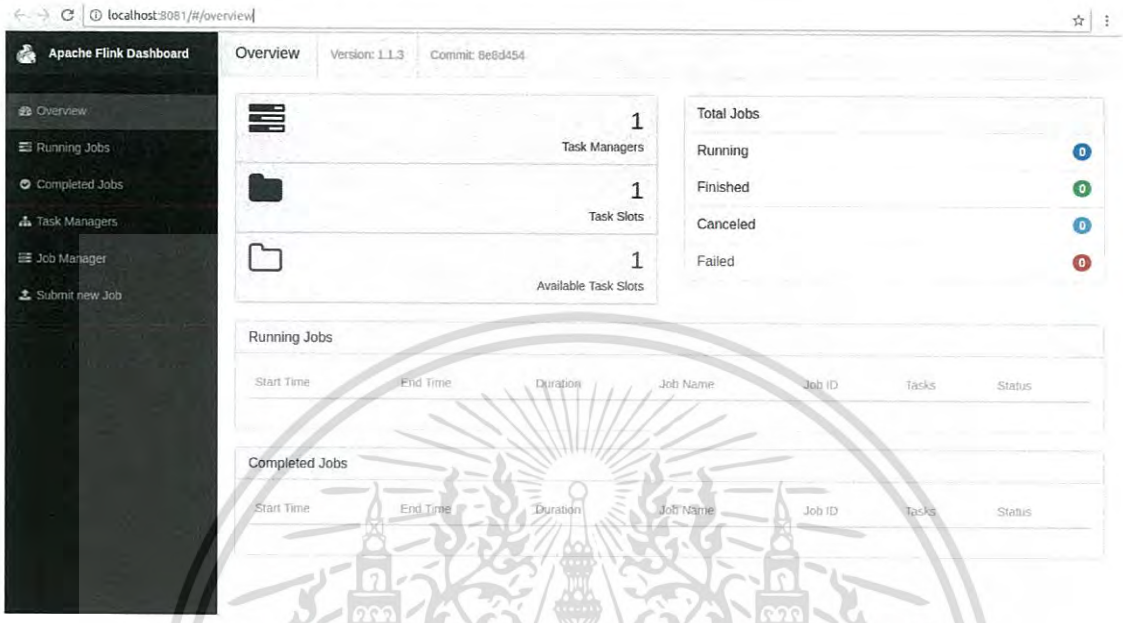
ดาวน์โหลดได้ที่ https://flink.apache.org/downloads.html โดยต้องดูเวอร์ชั่นของสกาล่าให้ตรงกัน หากต้องการดูเวอร์ชั่น พิมพ์ scala -version



ภาพที่ 3.27 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดฟลิงค์

ทำการแตกไฟล์ด้วยคำสั่ง tar xfz Downloads/flink-1.1.3-bin-hadoop27-scala_2.11.tgz และย้ายไปยังโฟลเดอร์ hadoopenv/ ด้วยคำสั่ง mv flink-1.1.3/ hadoopenv/

จากนั้นเปิดใช้งานฟลิงค์ด้วยคำสั่ง hadoopenv/flink-1.1.3/bin/start-local.sh



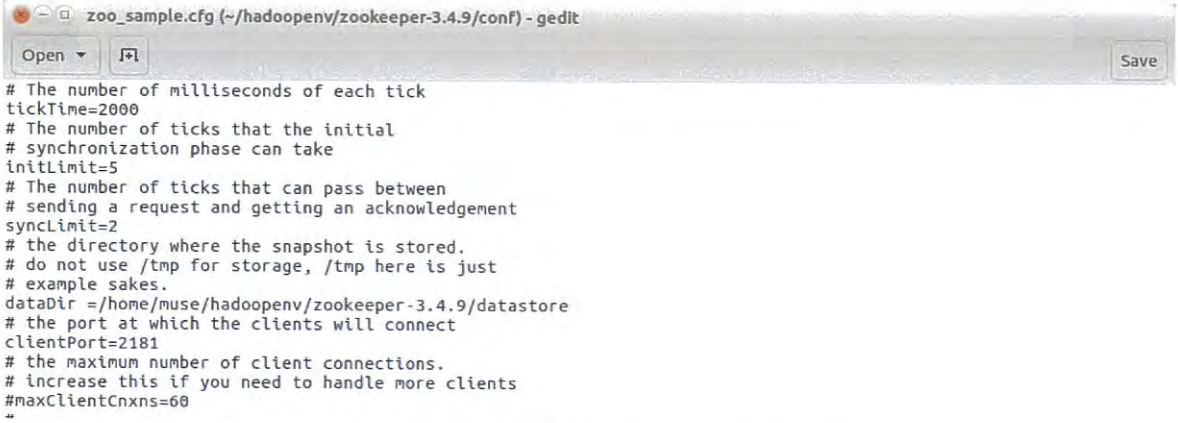
ภาพที่ 3.28 หน้าติดต่อประสานงานระหว่างฟลิงค์และผู้ใช้งาน

จากภาพที่ 3.28 เป็นการเข้าใช้งานฟลิงค์ผ่านหน้าเว็บอินเตอร์เฟสที่พอร์ท 8081

3.2.6 อาปาเช่ซูคิปเปอร์

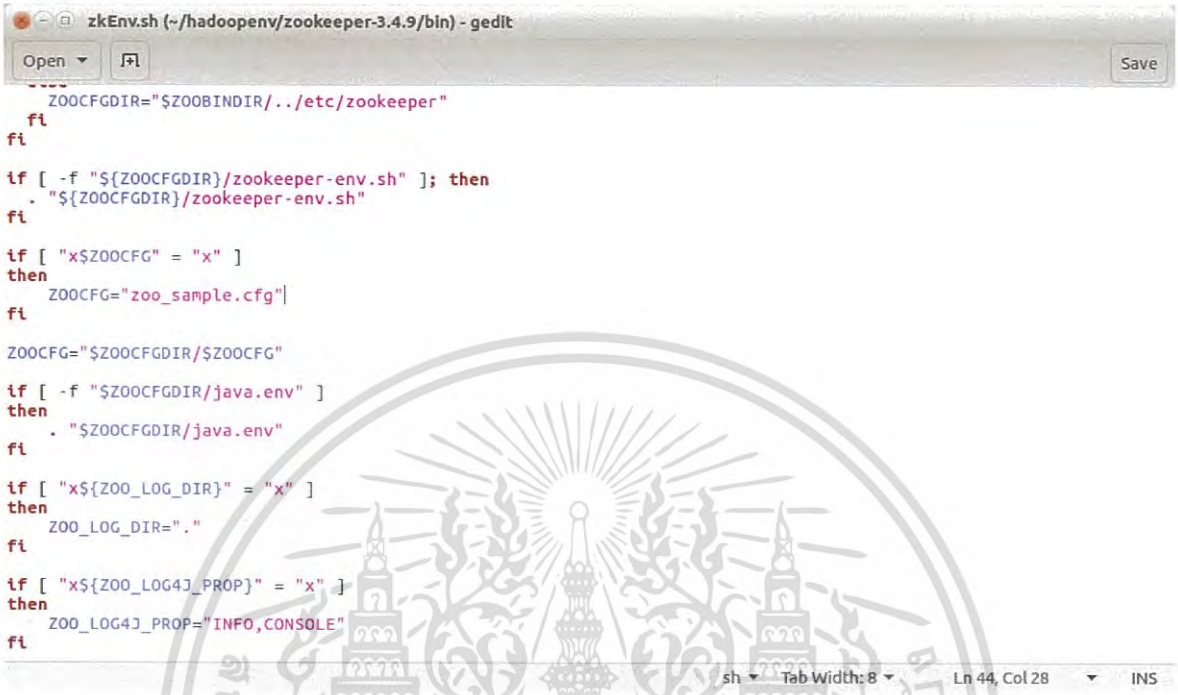
ทำการดาวน์โหลดไฟล์ที่ http://www.apache.org/dyn/closer.cgi/zookeeper/ จากนั้นแตกไฟล์และย้ายไปยังโฟลเดอร์ hadoopenv/ ด้วยคำสั่ง mv zookeeper-3.4.9/ hadoopenv/

สร้างโฟลเดอร์สำหรับเก็บข้อมูลซูคิปเปอร์ด้วยคำสั่ง mkdir hadoopenv/zookeeper-3.4.9/datastore และเปิดไฟล์เพื่อแก้ไข gedit hadoopenv/zookeeper-3.4.9/conf/zoo_sample.cfg



ภาพที่ 3.29 การตั้งค่าใน zoo_sample.cfg

จากภาพที่ 3.29 เป็นการตั้งค่า tickTime dataDir clientPort initLimit และ syncLimit ทำการเปิดไฟล์ zkEnv.sh เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/zookeeper-3.4.9/bin/zkEnv.sh

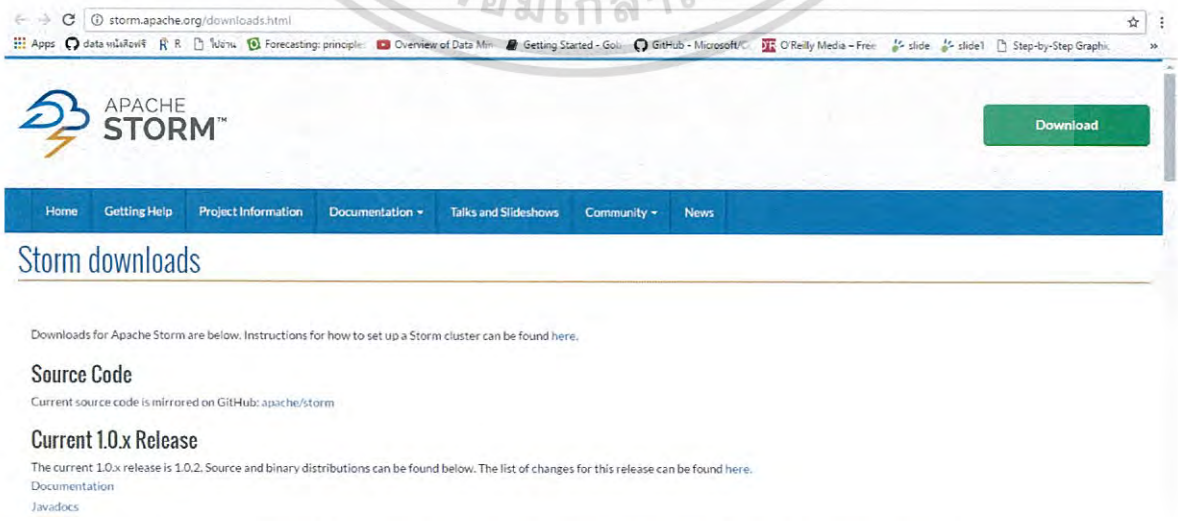


ภาพที่ 3.30 การตั้งค่าใน zkEnv.sh

จากภาพที่ 3.30 เป็นการตั้งค่า ZOOCFG="zoo_sample.cfg" จากนั้นทำการเปิดใช้งานซูคิปเปอร์ด้วยคำสั่ง hadoopenv/zookeeper-3.4.9/bin/zkServer.sh start

3.2.7 อาปาเช่สตอร์ม

ทำการดาวน์โหลดไฟล์ได้ที่ http://storm.apache.org/downloads.html



ภาพที่ 3.31 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดสตอร์ม

จากภาพที่ 3.31 เป็นหน้าต่างแสดงลิงก์ดาวน์โหลดไฟล์ของสตอร์มให้เลือกไฟล์ที่มีนามสกุล .tar.gz จากนั้นแตกไฟล์ด้วยคำสั่ง tar xfz Downloads/apache-storm-1.0.2.tar.gz และย้ายไปยังโฟลเดอร์ hadoopenv/ ด้วยคำสั่ง mv apache-storm-1.0.2/ hadoopenv/ เพื่อจัดเก็บให้อยู่ในโฟลเดอร์เดียวกับที่เราใช้งานโปรแกรมอื่น ๆ

สร้างโฟลเดอร์สำหรับเก็บข้อมูลจากสตอร์มด้วยคำสั่ง mkdir hadoopenv/apache-storm-1.0.2/datastore แล้วจึงเปิดไฟล์ storm.yaml เพื่อแก้ไขด้วยคำสั่ง gedit hadoopenv/apache-storm-1.0.2/conf/storm.yaml

```
########### These MUST be filled in for a storm configuration
storm.zookeeper.servers:
      - "localhost"
#     - "server1"
#     - "server2"
#
storm.local.dir:"/home/muse/hadoopenv/apache-storm-1.0.2/datastore"
nimbus.host: "localhost"
supervisor.slots.ports:
 - 6700
 - 6701
 - 6702
 - 6703
nimbus.seeds: ["host1", "host2", "host3"]

#
#
# ##### These may optionally be filled in:
#
## List of custom serializations
# topology.kryo.register:
```

YAML ▾ Tab Width: 8 ▾ Ln 30, Col 22 ▾ INS

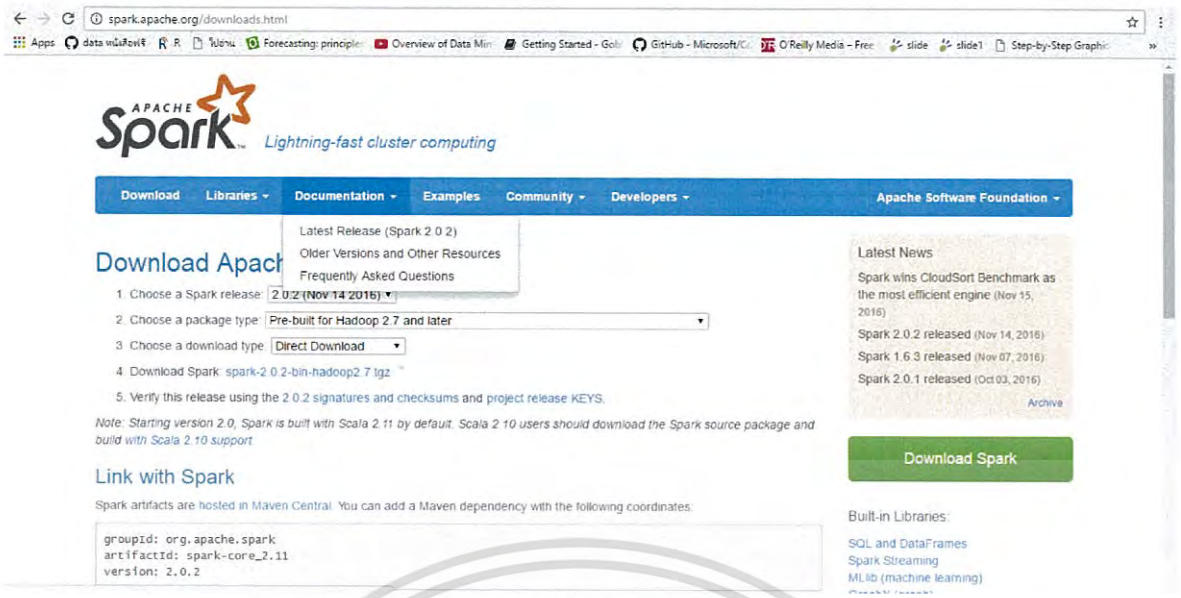
ภาพที่ 3.32 การตั้งค่าใน storm.yaml

จากภาพที่ 3.32 เป็นการตั้งค่า storm.zookeeper.servers storm.local.dir nimbus.host และ supervisor.slots.ports

เปิดใช้งานนิมบัสของสตอร์ม ซูเปอร์ไวเซอร์ของสตอร์มและอินเตอร์เฟสของสตอร์มด้วยคำสั่งดังนี้ nimbus hadoopenv/apache-storm-1.0.2/bin/storm nimbus ซูเปอร์ไวเซอร์ hadoopenv/apache-storm-1.0.2/bin/storm supervisor และสุดท้ายคือ hadoopenv/apache-storm-1.0.2/bin/storm ui จากนั้นเข้าใช้งานสตอร์มผ่านหน้าเว็บอินเตอร์เฟสที่ http://localhost:6700

3.2.8 อาปาเช่สปาร์ก

ทำการดาวน์โหลดได้ที่สปาร์กได้ที่ http://spark.apache.org/downloads.html ดังภาพที่ 3.3 จากนั้นให้แตกไฟล์ด้วยคำสั่ง tar xfz Downloads/spark-2.0.2-bin-hadoop2.7.tgz และย้ายไปยังโฟลเดอร์ hadoopenv/ ด้วยคำสั่ง mv spark-2.0.2-bin-hadoop2.7/ hadoopenv/ เพื่อรวบรวมไฟล์ให้อยู่ในโฟลเดอร์เดียวกัน ต้องทำการติดตั้งและแก้ไขไฟล์ ~/.bashrc เนื่องจากต้องการให้สิ่งแวดล้อมของเครื่องเก็บการตั้งค่าของสปาร์กไว้และการตั้งค่านี้ยังจะทำให้เครื่องมืออื่นก็สามารถใช้งานร่วมกับสปาร์กได้เพราะเป็นการตั้งค่าในไฟล์ของเครื่องเอง



ภาพที่ 3.33 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดสปาร์ก

เปิดไฟล์ ~/.bashrc เพื่อตั้งค่าสปาร์กด้วยคำสั่ง gedit ~/.bashrc

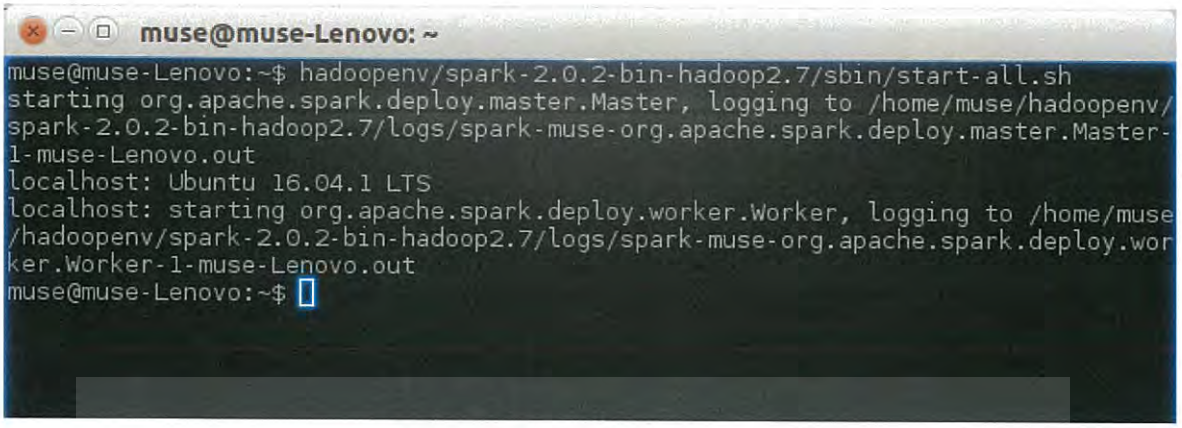
```
#HADOOP VARIABLES START
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_INSTALL=/home/muse/hadoopenv/hadoop-2.7.3
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
#HADOOP VARIABLES END

#HBASE VARIABLES START
export HBASE_HOME=/home/muse/hadoopenv/hbase-1.0.3
export PATH=$PATH:$HBASE_HOME/bin
#HBASE VARIABLES END

#HIVE VARIABLES START
export HIVE_HOME=/home/muse/hadoopenv/apache-hive-1.2.1-bin
export PATH=$PATH:$HIVE_HOME/bin
#HIVE VARIABLES STOP

#SPARK VARIABLES START
export SCALA_HOME=/home/muse/hadoopenv/spark-2.0.2-bin-hadoop2.7
export PATH=$SCALA_HOME/bin:$PATH
#SPARK VARIABLES STOP
```

ภาพที่ 3.34 ตัวแปรสปาร์กที่ใส่ใน ~/.bashrc

จากภาพที่ 3.34 เป็นการตั้งค่าสปาร์กโดยตั้งค่า SCALA_HOME และ PATH และทำการรีโหลดไฟล์ ~/.bashrc ด้วยคำสั่ง source ~/.bashrc

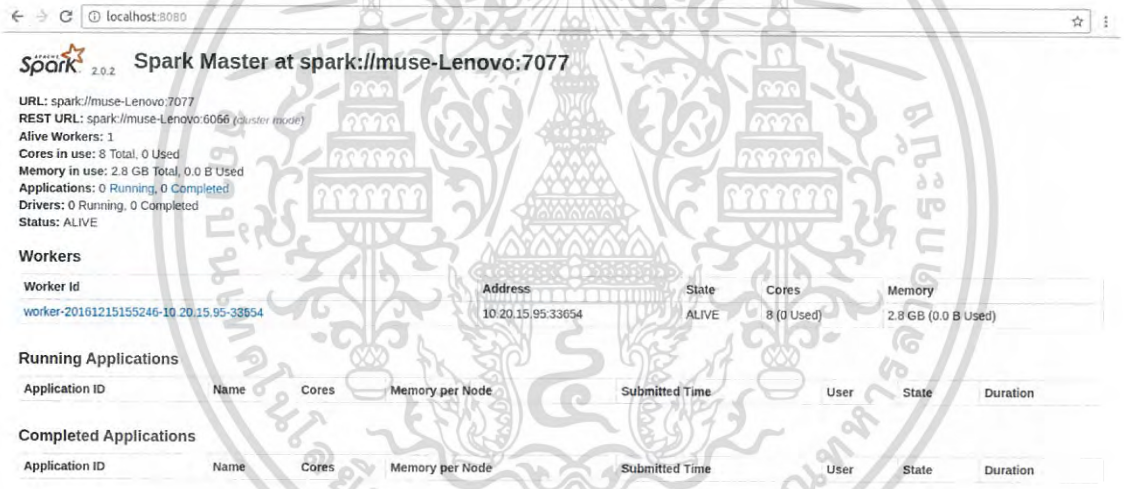
ภาพที่ 3.35 ผลลัพธ์คำสั่ง scala –version

จากภาพที่ 3.35 เป็นการตรวจสอบผลลัพธ์การตั้งค่าของสกาล่า



ภาพที่ 3.36 ผลลัพธ์คำสั่ง start-all.sh

จากภาพที่ 3.36 เป็นการเปิดใช้งานสปาร์กด้วยคำสั่ง start-all.sh จะต้องขึ้นผลลัพธ์ดังภาพถือว่าได้ทำการเปิดการใช้งานสปาร์กแล้ว

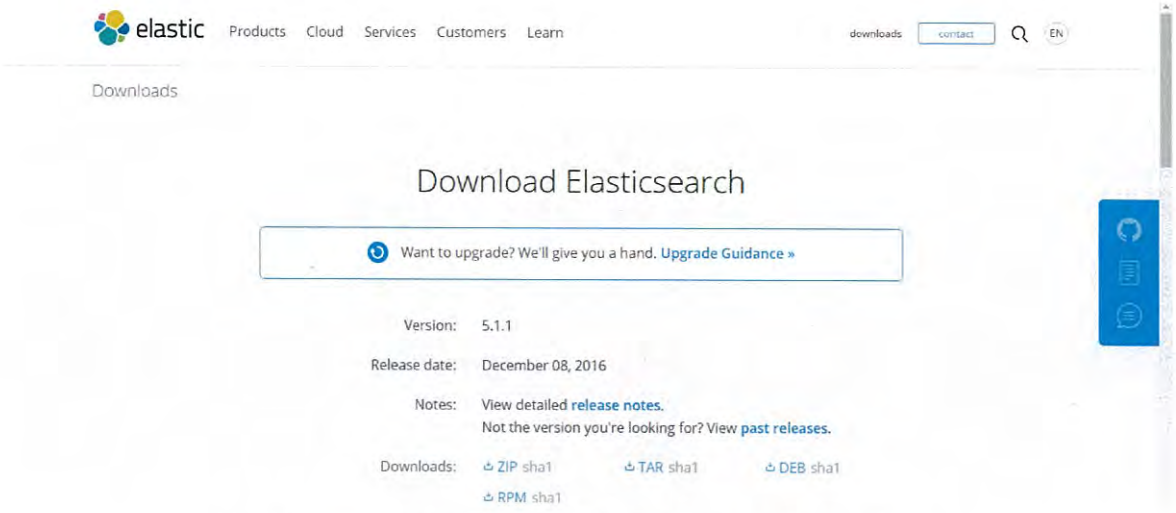


ภาพที่ 3.37 หน้าเว็บอินเตอร์เฟสระหว่างผู้ใช้งานและสปาร์ก

จากภาพที่ 3.37 เป็นการเข้าใช้งานสปาร์กผ่านหน้าติดต่อประสานงานที่พอร์ต 8080 จะสามารถเข้าไปบังคับและควบคุมการทำงานของสปาร์กมาตเตอร์และหน่วยย่อยของสปาร์กได้

3.2.9 อีลาสติกเสิร์ช

ทำการดาวน์โหลดได้ที่ https://www.elastic.co/downloads/elasticsearch หรือ wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-5.0.1.tar.gz จากนั้นแตกไฟล์ด้วยคำสั่ง tar -xzf Downloads/elasticsearch-5.0.1.tar.gz และย้ายไปยังโฟลเดอร์ hadoopenv/ ด้วยคำสั่ง mv elasticsearch-5.0.1.tar.gz hadoopenv/



ภาพที่ 3.38 หน้าเว็บไซต์แสดงลิงก์ไปยังที่ดาวน์โหลดอีลาสติกเสิร์ช

จากภาพที่ 3.38 เป็นหน้าดาวน์โหลดไฟล์อีลาสติกเสิร์ชควรโหลดไฟล์นามสกุล .tar.gz จากนั้นเริ่มใช้งานอีลาสติกเสิร์ชโดยใช้คำสั่ง hadoopenv/elasticsearch-5.0.1/bin/elasticsearch start

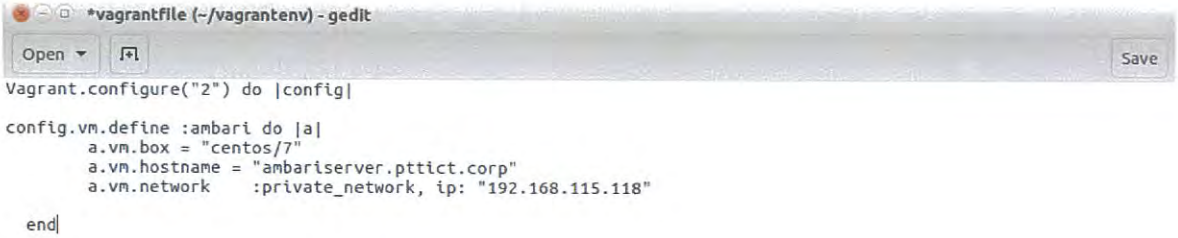
```
{
  "name" : "IjcZlB5",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "jODxX0C5T32ts0kBk1dGlw",
  "version" : {
    "number" : "5.0.1",
    "build_hash" : "080bb47",
    "build_date" : "2016-11-11T22:08:49.812Z",
    "build_snapshot" : false,
    "lucene_version" : "6.2.1"
  },
  "tagline" : "You Know, for Search"
}
```

ภาพที่ 3.39 หน้าอินเตอร์เฟสระหว่างอีลาสติกเสิร์ชและผู้ใช้งาน

จากภาพที่ 3.39 เป็นการเข้าใช้งานอีลาสติกเสิร์ชผ่านหน้าอินเตอร์เฟสที่พอร์ต 9200

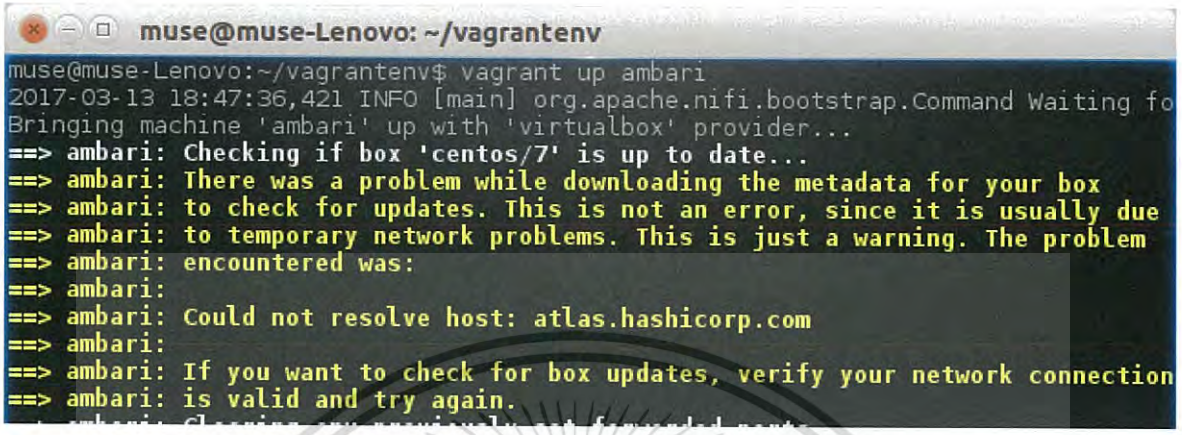
3.2.10 อาปาเช่อัมบารี

เข้าไปที่ โฟลเดอร์วาแกรนต์ที่สร้างไว้ด้วยคำสั่ง cd vagrantenv/ จากนั้นเปิด vagrantfile เพื่อตั้งค่าด้วยคำสั่ง gedit vagrantfile ซึ่งจะเป็นการตั้งต่าโฮสต์เนม เน็ตเวิร์ก และตั้งชื่อวาแกรนต์ที่เราต้องการสร้างขึ้น



ภาพที่ 3.40 การตั้งค่าใน Vagrantfile

จากภาพที่ 3.40 เป็นการตั้งค่าเครื่องเสมือนที่เราต้องการสร้างขึ้นใน Vagrantfile จากนั้นเข้าใช้งานเครื่องเสมือนที่ตั้งค่าไว้โดยจากภาพที่ 3.40 ได้ตั้งค่าไว้ชื่อ ambari จึงใช้คำสั่ง vagrant up ambari

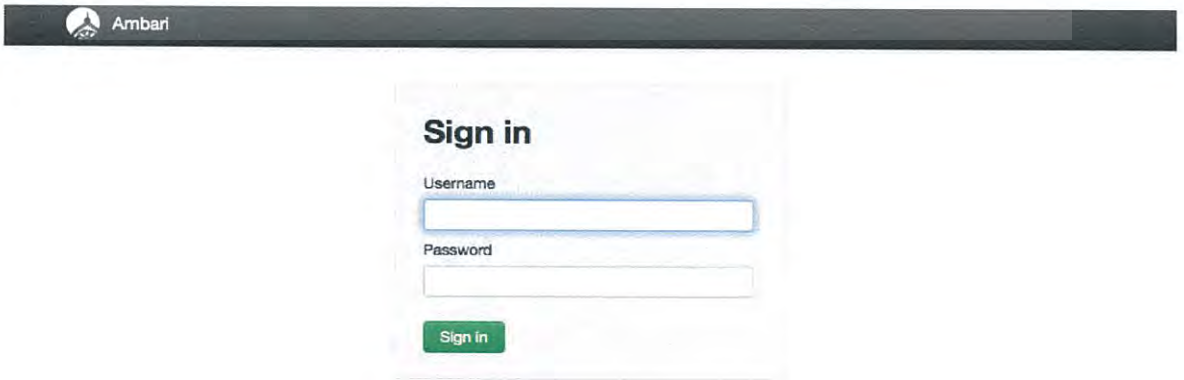


ภาพที่ 3.41 ผลลัพธ์คำสั่ง vagrant up ambari

จากภาพที่ 3.41 เป็นการเปิดการทำงานเครื่องเสมือนและติดตั้งระบบปฏิบัติการ จากนั้นเข้าใช้งานแบชเชลล์ของอัมบารีเป็น root ด้วยคำสั่ง vagrant ssh ambari และ sudo su – แล้วทำการลงโปรแกรม wget ntp และ net-tools ด้วยคำสั่ง yum -y install wget ntp net-tools

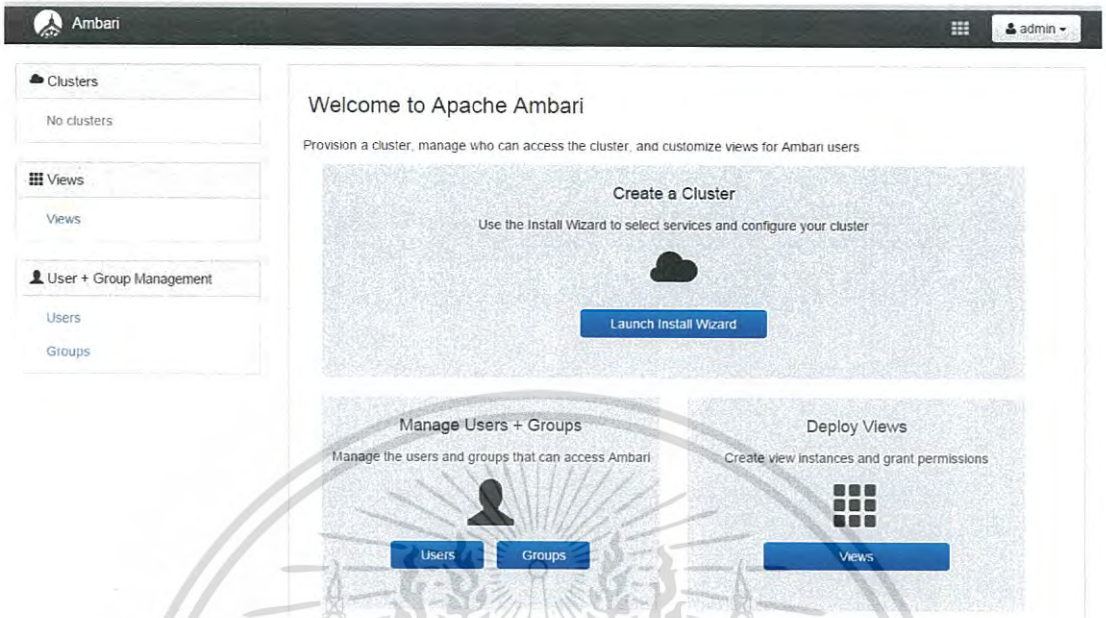
ทำการดาวน์โหลด ambari.repo wget -O /etc/yum.repo.d/ambari.repo http://public-repo-1.hortonworks.com/ambari/ centos7/2.x/updates/2.2.1.0/ambari.repo จากนั้นอัปเดตและติดตั้งอัมบารีเซิร์ฟเวอร์และอัมบารีเอเจนด้วยคำสั่ง yum -y update และ yum install ambari-server ambari-agent แล้วจึงทำ ambari-server setup

เปิดการใช้งานอัมบารีโดยใช้คำสั่ง ambari-server start และ ambari-agent start และถ้าระบบเตือนให้ลง ntpd จึงใช้คำสั่งดังนี้ systemctl start ntpd แล้วไปยังพอร์ต 8080 เพื่อติดตั้งเซอร์วิสของฮาดูปผ่านอัมบารี



ภาพที่ 3.42 หน้าก่อนลอกอินอัมบารี

จากภาพที่ 3.42 แสดงหน้าลอกอินการเข้าใช้งานอัมบารีโดยใช้ username: admin password: admin ซึ่งเป็นค่าเริ่มต้นของการใช้งาน



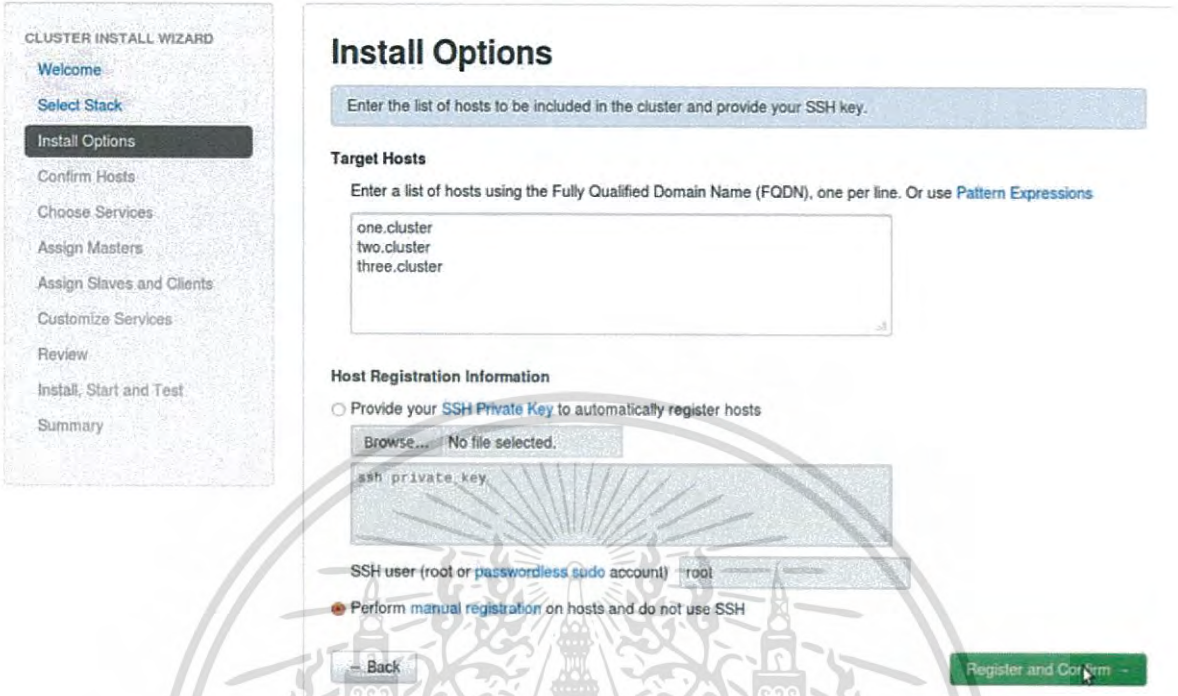
ภาพที่ 3.43 หน้าหลังลอกอินอัมบารี

จากภาพที่ 3.43 แสดงการลอกอินเข้ามาแล้ว จากนั้นติดตั้งฮาดูปและเครื่องมืออื่น ๆ โดยเลือก Launch install wizard ซึ่งจะเป็นการแนะนำและติดตั้งเครื่องมือที่เราต้องการได้แบบอัตโนมัติ โดยที่เราไม่ต้องไปตั้งค่าสิ่งแวดล้อมของแต่ละเครื่องมือเอง แต่อัมบารีจะตั้งค่าสิ่งแวดล้อมของเครื่องมือให้ใช้ร่วมกันได้หมด

ภาพที่ 3.44 ขั้นตอน Select Stack

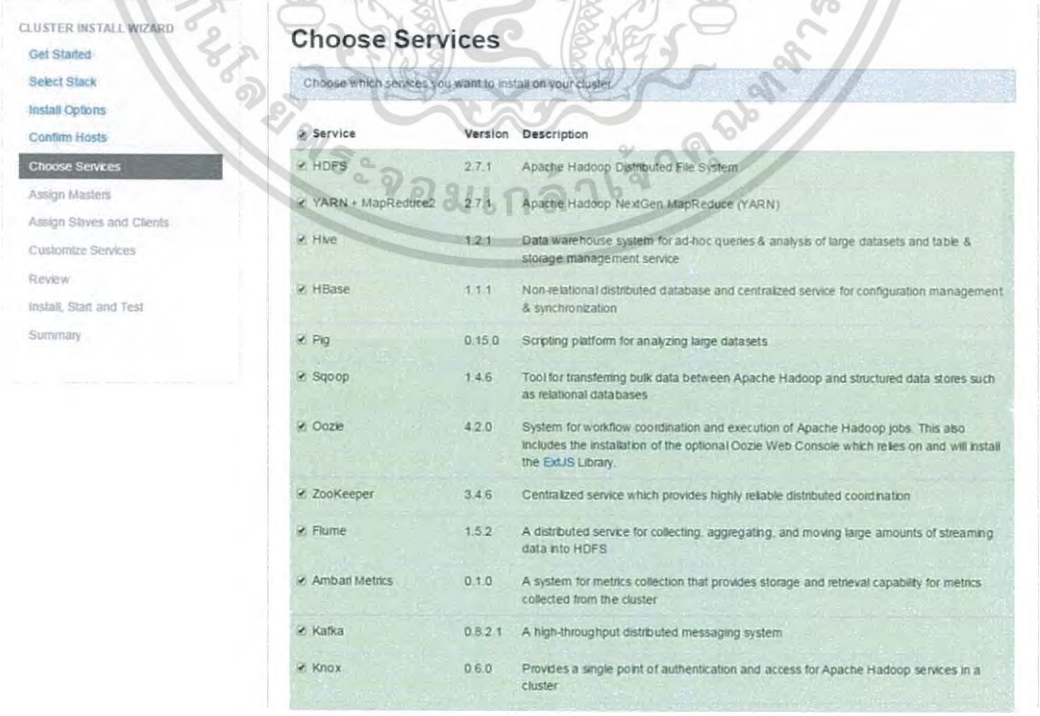
จากภาพที่ 3.44 ให้เลือกสแตกของฮาดูปเวอร์ชั่นล่าสุดเพราะการลงเวอร์ชั่นที่ใหม่ที่สุดจะช่วยลดการเกิดปัญหาการใช้งานเกิดขึ้นและมีการอัพเดตปัญหาต่าง ๆ จากเวอร์ชันก่อนแล้ว จากนั้นใส่

Target Hosts ตามที่ตั้งค่าไว้ในวาแกรนต์และเลือก do not use SSH เนื่องจากการใส่คีย์ยังมีปัญหาต่อการลงโปรแกรมจึงเลือกที่จะไม่ใส่



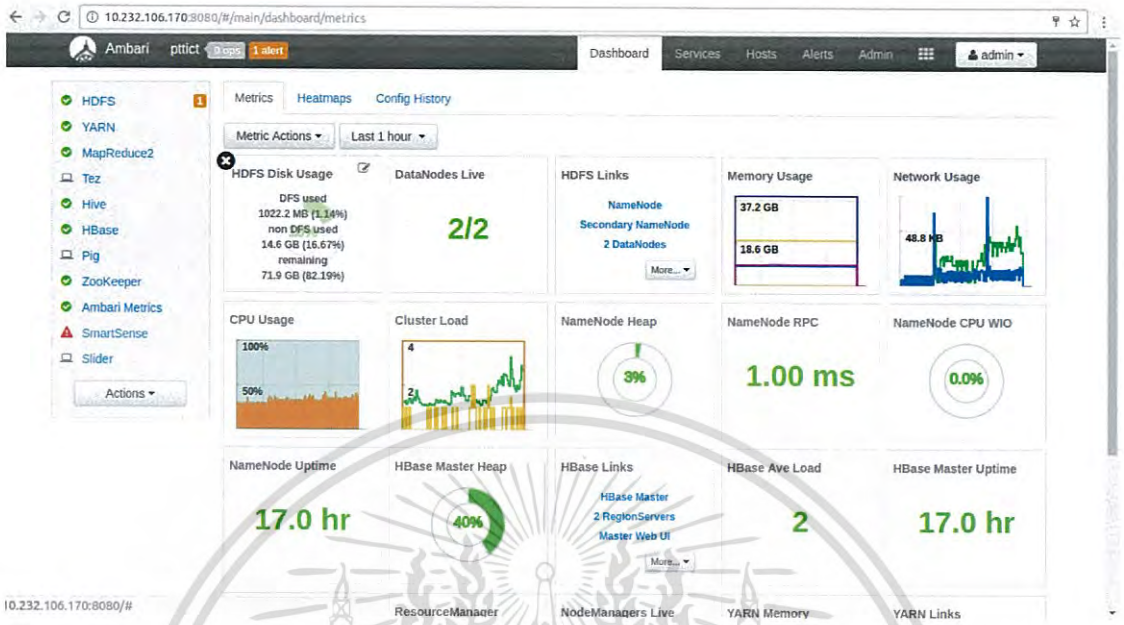
ภาพที่ 3.45 ขั้นตอน Install Options

จากภาพที่ 3.45 เมื่อเข้าหน้า Confirm Hosts รอโปรแกรมตรวจสอบความถูกต้องแล้วกดตกลง จากนั้นเลือกเซอร์วิสที่เราต้องการลง หากยังไม่ลงตอนนี้ก็สามารถที่จะลงเพิ่มในอนาคตได้



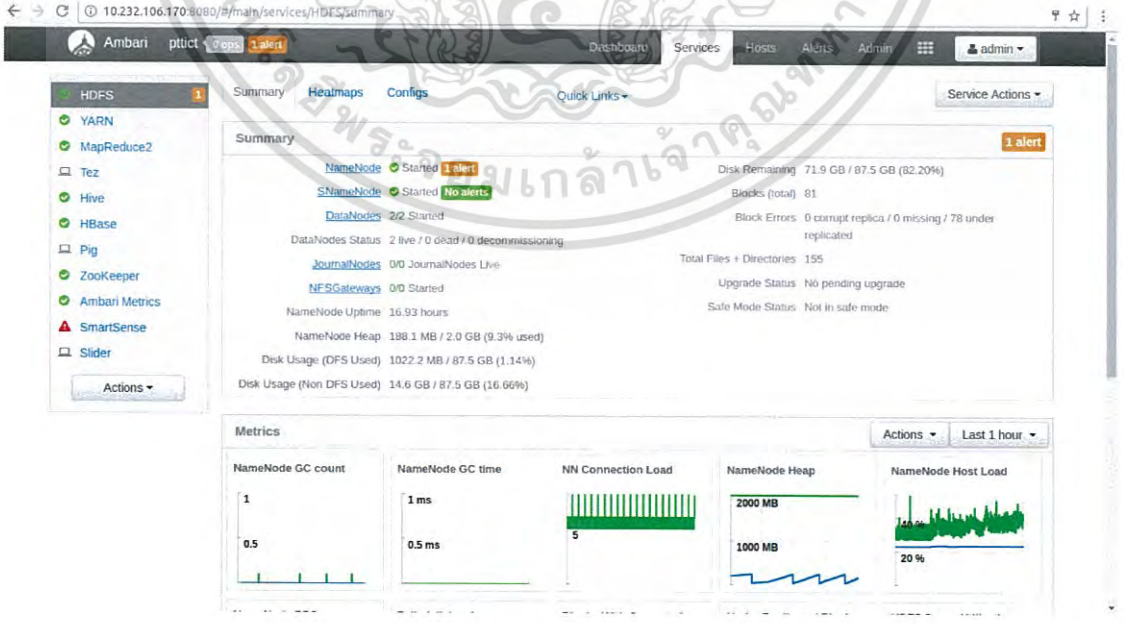
ภาพที่ 3.46 ขั้นตอน Choose Services

จากภาพที่ 3.46 เป็นการเลือกเครื่องมือที่ต้องการจะติดตั้งโดยหากเครื่องมือไหนที่จำเป็นต่อการใช้งาน จะมีการแจ้งเตือนให้เลือกเครื่องมือนั้น ก่อนจะข้ามไปขั้นตอนต่อไป



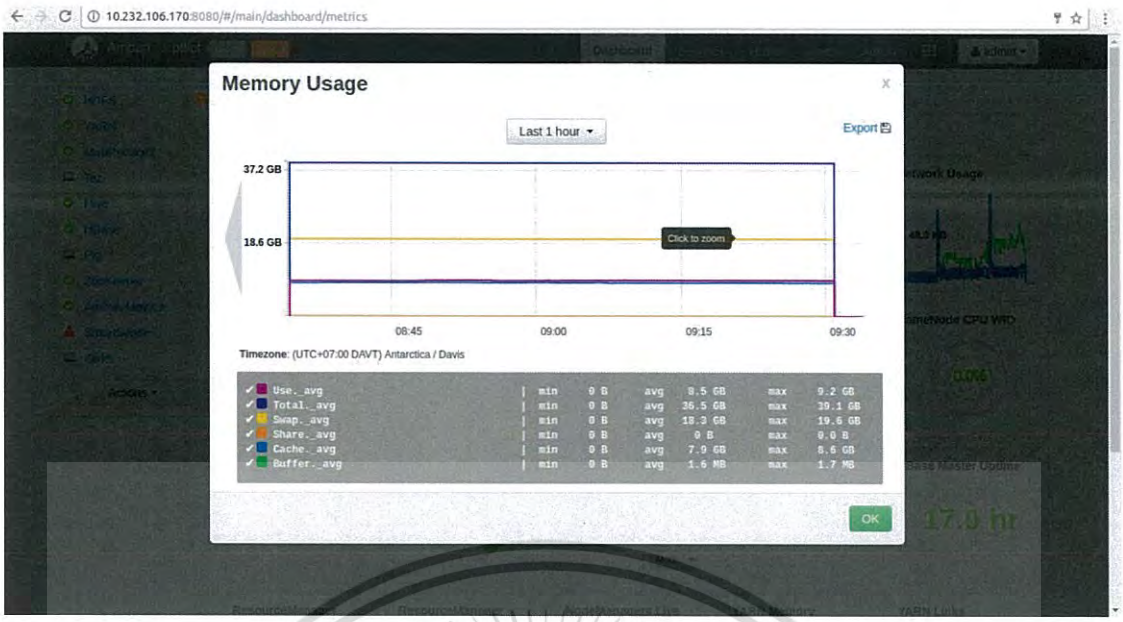
ภาพที่ 3.47 หน้ากระดานของอัมบารี

จากภาพที่ 3.47 Assign Masters Assign Servers and Clients Custom Services Review จะถูกตั้งค่าให้อัตโนมัติแล้ว แต่ถ้าหากผู้ใช้งานอยากเปลี่ยนค่าก็สามารถทำได้ แล้วให้กดตกลงจนไปถึงหน้า Install Start and Test เพื่อรออัมบารีจัดการลงโปรแกรมให้ เมื่อลงสำเร็จแล้วก็สามารถเข้าไปหน้าใช้งานได้



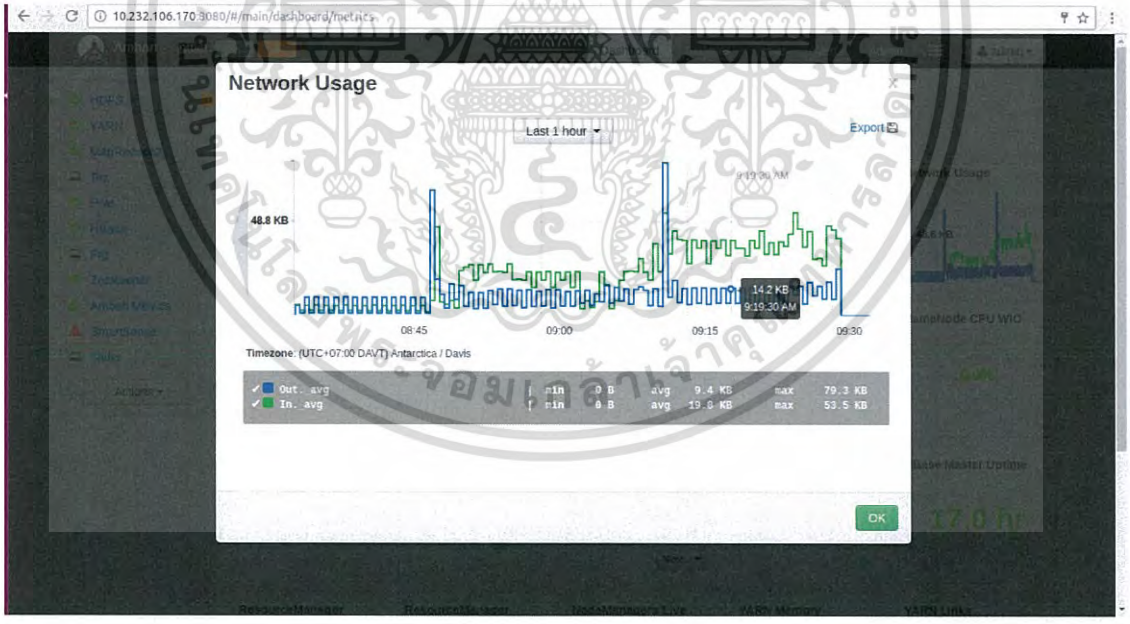
ภาพที่ 3.48 เซอร์วิสของฮาดูป

จากภาพที่ 3.48 เป็นรายละเอียดของเซอร์วิสของฮาดูปซึ่งสามารถมอนิเตอร์ฮาดูปได้



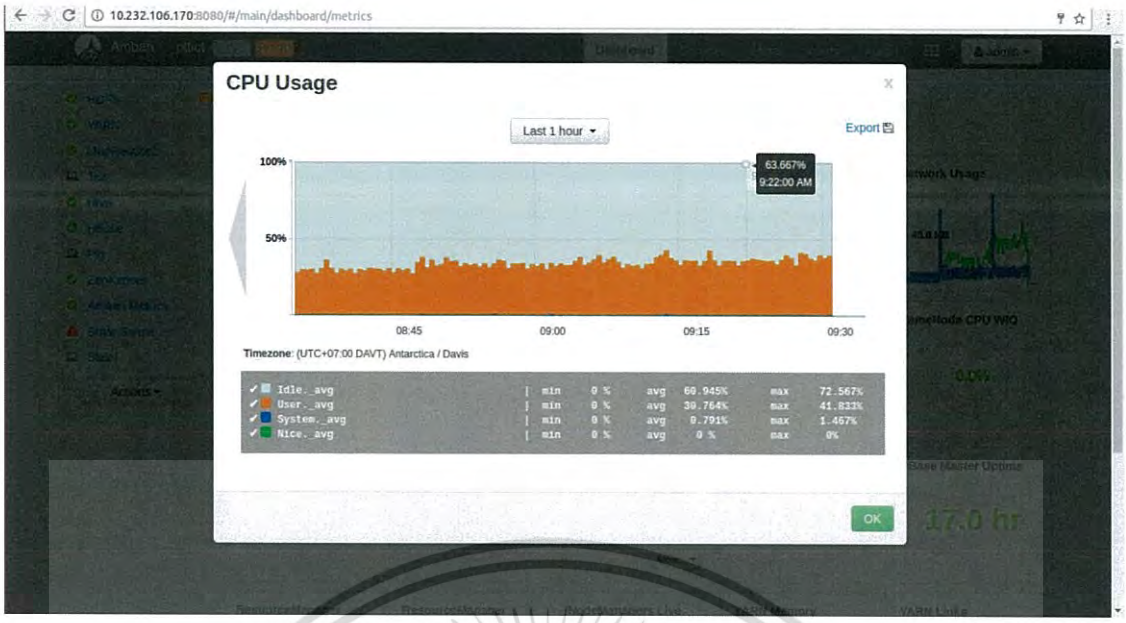
ภาพที่ 3.49 การใช้งานหน่วยความจำ

จากภาพที่ 3.49 เป็นการแสดงการใช้งานหน่อยความจำทั้งหมดของเอชดีเอฟเอสซึ่งจะคอยบอกสถานะของหน่วยความจำของระบบ ทั้งปริมาณหน่วยความจำที่ถูกใช้ไป หน่วยความจำที่เหลือว่าง หน่วยความจำที่กำลังถูกใช้งาน ซึ่งจะมีการแจ้งเตือนขึ้นหากระบบมีปัญหาการใช้งานหน่วยความจำเกิดขึ้น



ภาพที่ 3.50 การใช้งานเน็ตเวิร์ก

จากภาพที่ 3.50 เป็นการแสดงการใช้งานเน็ตเวิร์กทั้งหมดของเอชดีเอฟเอสซึ่งจะคอยบอกสถานะของเน็ตเวิร์กของระบบแบบทันทีทันใด หากระบบมีการทำงานของเน็ตเวิร์กที่ผิดพลาดจะมีการแจ้งเตือนผ่านอัมบารีเกิดขึ้น เพื่อให้ผู้ดูแลระบบรีบดำเนินการแก้ไขในทันทีทันใด โดยจะสามารถเข้าไปดูปัญหาได้โดยตรงทำให้ง่ายต่อการมอนิเตอร์



ภาพที่ 3.51 การใช้งานซีพียู

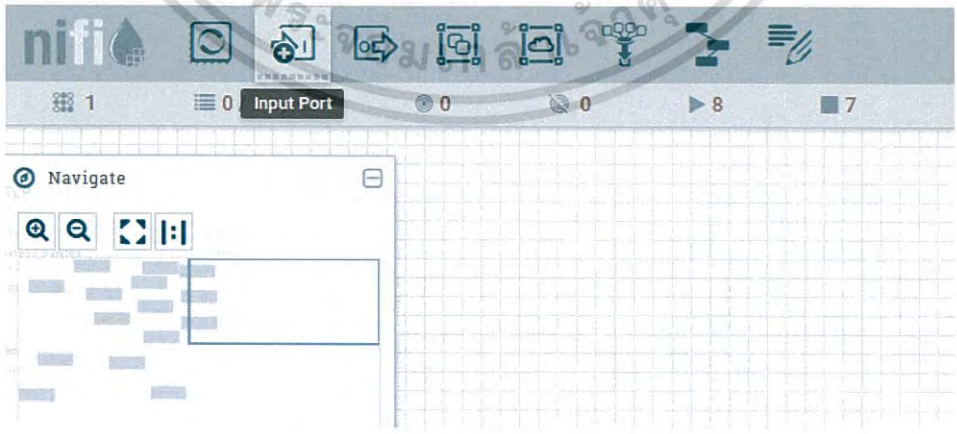
จากภาพที่ 3.51 เป็นการแสดงการใช้งานซีพียูทั้งหมดของเอชดีเอฟเอสซึ่งจะคอยบอกสถานะของซีพียูของระบบ

3.3 การนำเครื่องมือไปใช้

เมื่อติดตั้งเครื่องมือเสร็จแล้ว ผู้จัดทำได้มีการนำเครื่องมือมาใช้โดยแบ่งเป็น 2 ส่วนได้แก่

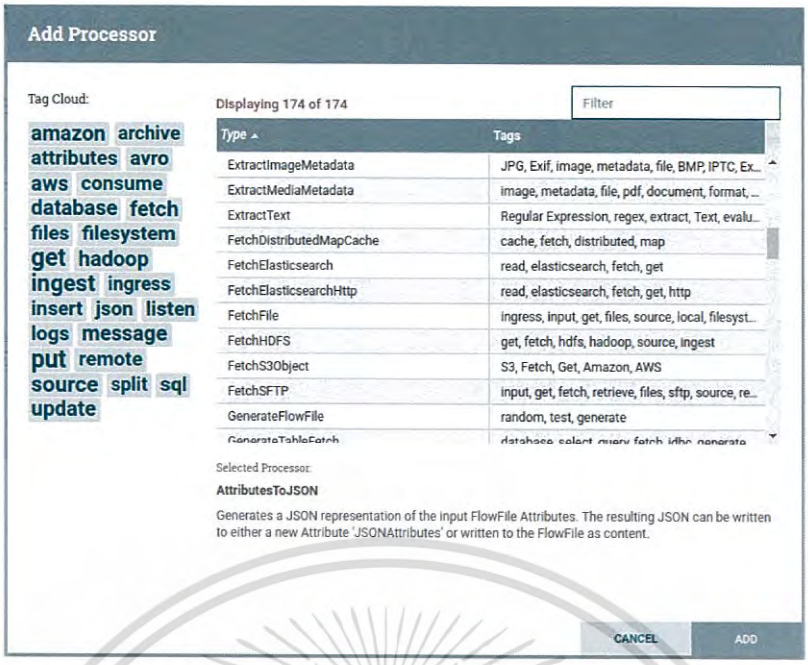
3.3.1 การนำข้อมูลเข้าเอชดีเอฟเอสโดยใช้อาปาเช่นายฟาย

ทางผู้จัดทำได้เลือกเครื่องมือนี้มาใช้นำข้อมูลเข้าเอชดีเอฟเอส เพราะข้อมูลของบริษัทนั้นมีรูปแบบหลายแบบ ซึ่งเครื่องมือนี้ตอบโจทย์ต่อการใช้งานมาก ผู้จัดทำจึงจะแสดงวิธีการนำข้อมูลเข้า ดังนี้



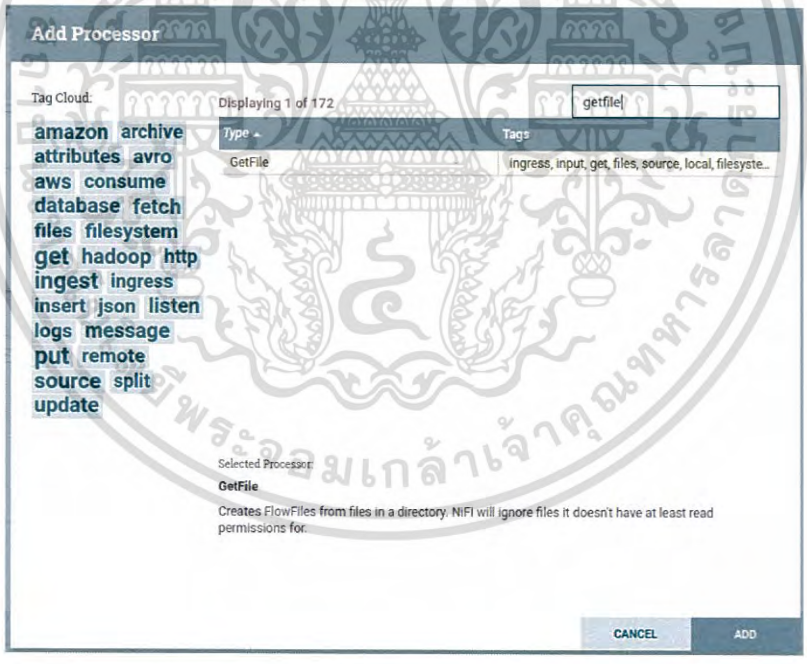
ภาพที่ 3.52 การเลือกโพรเซสเซอร์ GetFile

จากภาพที่ 3.52 เป็นการเลือกโพรเซสเซอร์แล้วลากมาวางบนพื้นที่ทำงานจากนั้นจะขึ้นหน้าตั้งค่าโพรเซสเซอร์อัตโนมัติ



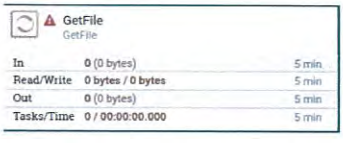
ภาพที่ 3.53 หน้าการตั้งค่าหลังจากเลือกโพรเซสเซอร์

จากภาพที่ 3.53 เป็นหน้าตั้งค่าโพรเซสเซอร์ที่ใช้ในการเลือกการทำงานต่าง ๆ



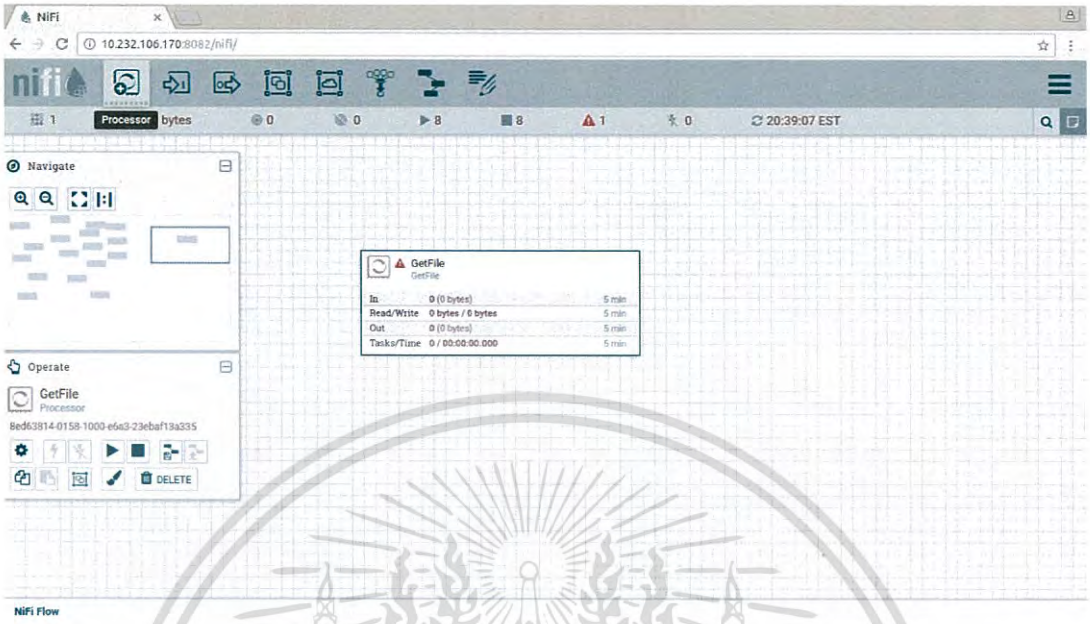
ภาพที่ 3.54 การตั้งค่าโพรเซสเซอร์ให้ทำหน้าที่ GetFile

จากภาพที่ 3.54 เป็นการเลือกตัวนำเข้าข้อมูลเข้าให้กับโพรเซสเซอร์



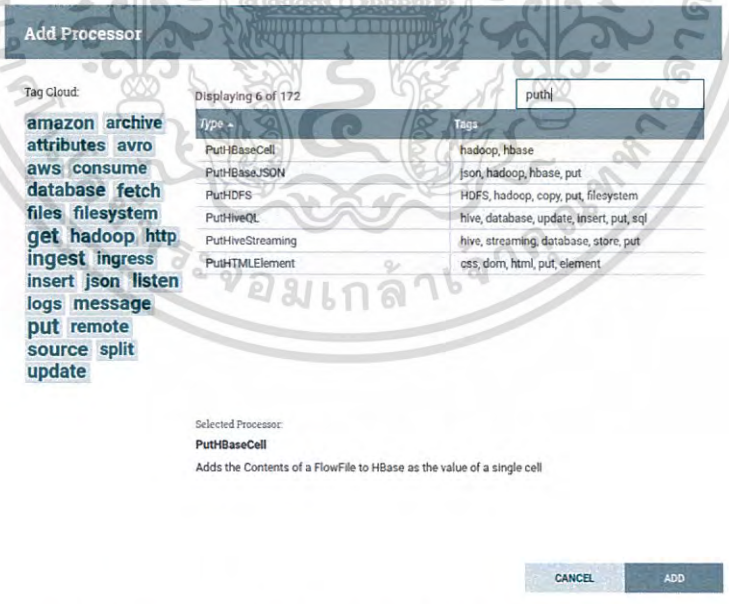
ภาพที่ 3.55 ผลลัพธ์จากการเลือกการทำงานแบบ GetFile

จากภาพที่ 3.55 เป็นผลลัพธ์ที่ได้จากการเลือกการตั้งค่าให้โพรเซสเซอร์ GetFile โดยจะมีค่าสถานะการอ่าน เขียน ไฟล์เข้า ไฟล์ออก และเวลาในการส่งออกของไฟล์บอกไว้



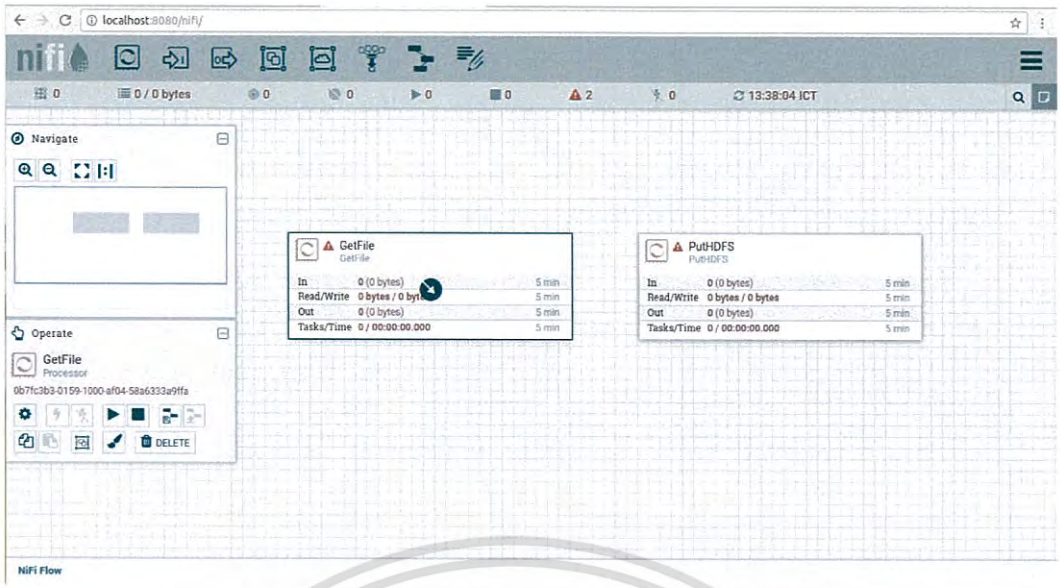
ภาพที่ 3.56 การเลือกโพรเซสเซอร์ในการส่งข้อมูลออก

จากภาพที่ 3.56 เป็นการเลือกโพรเซสเซอร์อีกตัวมา เพื่อเป็นตัวส่งข้อมูลออกไปยังเอชดีเอฟเอส



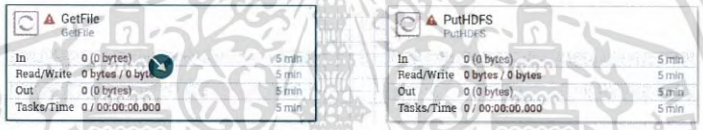
ภาพที่ 3.57 การตั้งค่าโพรเซสเซอร์ให้ทำหน้าที่ PutHDFS

จากภาพที่ 3.57 เป็นการตั้งค่าโพรเซสเซอร์ PutHDFS เพื่อใช้ส่งข้อมูลไปยังเอชดีเอฟเอส โดยการเลือกโพรเซสเซอร์ที่เหมาะสมนั้นให้เลือกจากตำแหน่งปลายทางที่จะไปเก็บว่าเป็นที่เก็บข้อมูลชนิดไหน เพราะแต่ละชนิดมีการเก็บโครงสร้างของไฟล์แตกต่างกัน



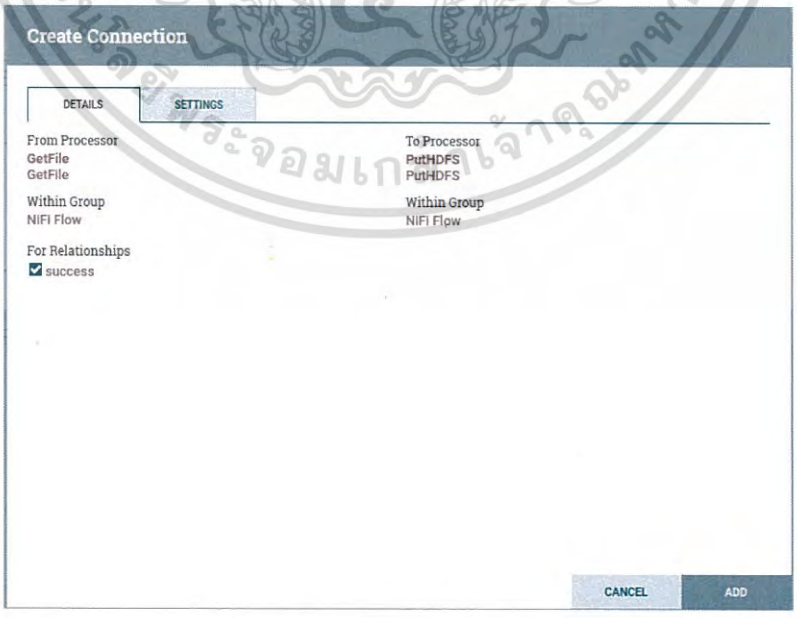
ภาพที่ 3.58 โพรเซสเซอร์ GetFile และ PutHDFS

จากภาพที่ 3.58 เป็นพื้นที่การทำงานที่แสดงโพรเซสเซอร์ GetFile และ PutHDFS



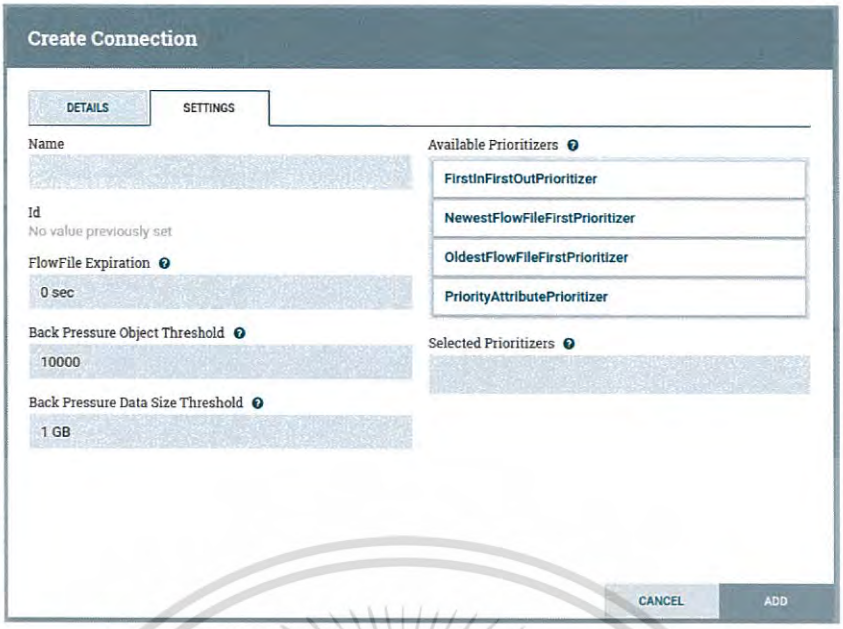
ภาพที่ 3.59 การเชื่อมต่อของโพรเซสเซอร์ GetFile และ PutHDFS

จากภาพที่ 3.59 ให้คลิกที่โพรเซสเซอร์ GetFile จะมีลูกศรปรากฏขึ้น ให้ลากไปเชื่อมกับโพรเซสเซอร์ PutHDFS เพื่อสร้างการเชื่อมต่อ



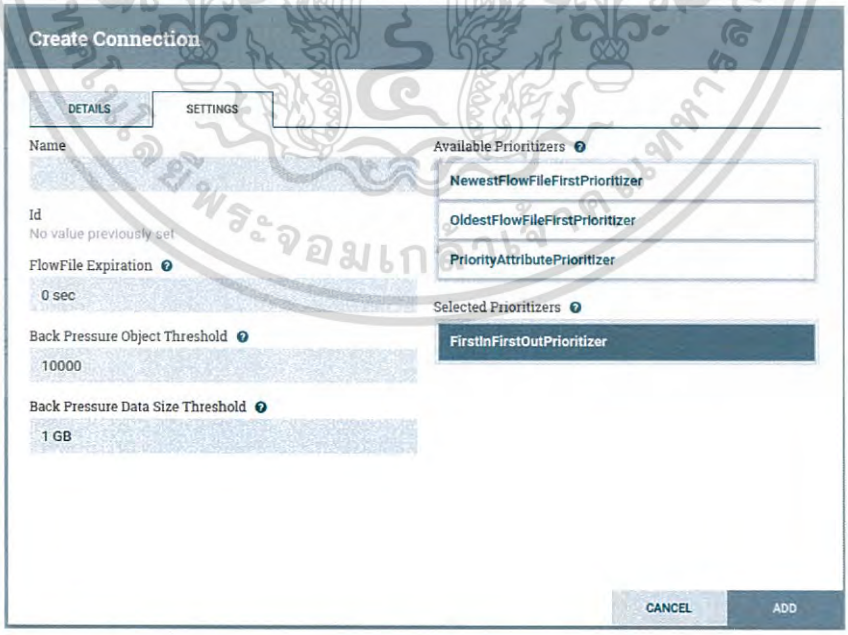
ภาพที่ 3.60 การเลือกความสัมพันธ์การเชื่อมต่อของโพรเซสเซอร์ GetFile และ PutHDFS

จากภาพที่ 3.60 จะขึ้นหน้าตั้งค่าการเชื่อมต่อโดยเลือกให้เลือก success



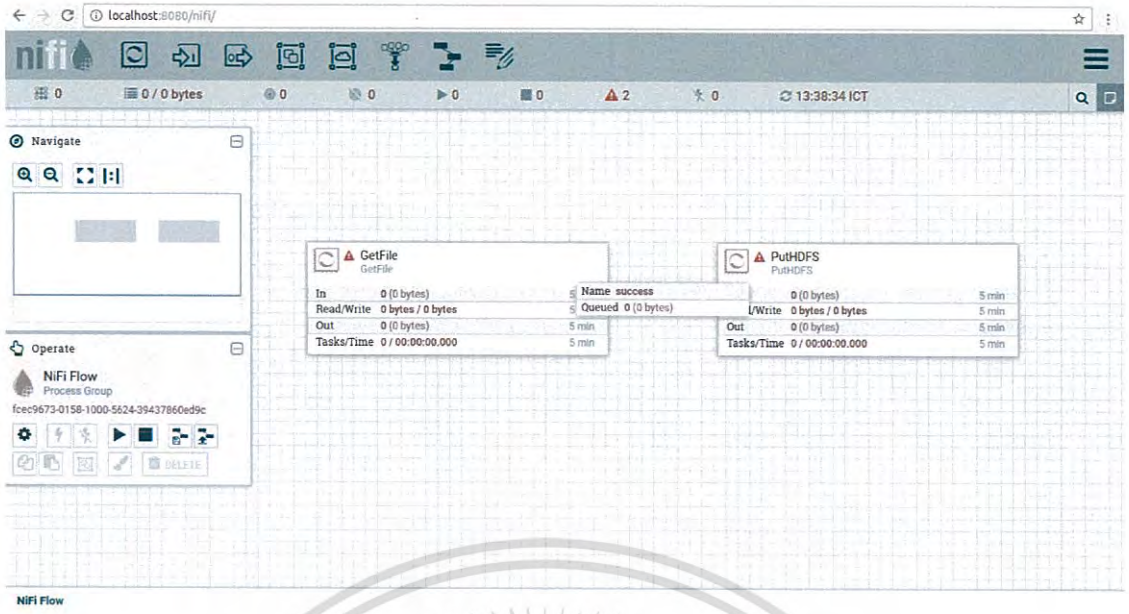
ภาพที่ 3.61 การตั้งค่าการจัดลำดับการส่งข้อมูลของโพรเซสเซอร์ GetFile และ PutHDFS

จากภาพที่ 3.61 ให้ไปที่ตั้งค่าซึ่งเป็นการตั้งค่าของโพรเซสเซอร์แต่ละตัวโดยเราสามารถตั้งค่า ชื่อ เวลาในการใช้งาน จำนวนสูงสุดของการใช้งาน ขนาดข้อมูลของการใช้งาน และการจัดลำดับการส่งข้อมูลผ่านโพรเซสเซอร์แต่ละตัวได้ เช่น ไฟล์ที่เข้าล่าสุดออกก่อน ไฟล์ที่เข้าหลังสุดออกก่อน แต่ในกรณีนี้ต้องการให้ไฟล์เป็นการทำงานที่ไฟล์ไหนเข้าก่อนให้ไฟล์นั้นออกก่อนจึงเลือกที่จะลาก FirstInFirstOutPrioritizer ลงมาที่ช่อง Selected Prioritizers



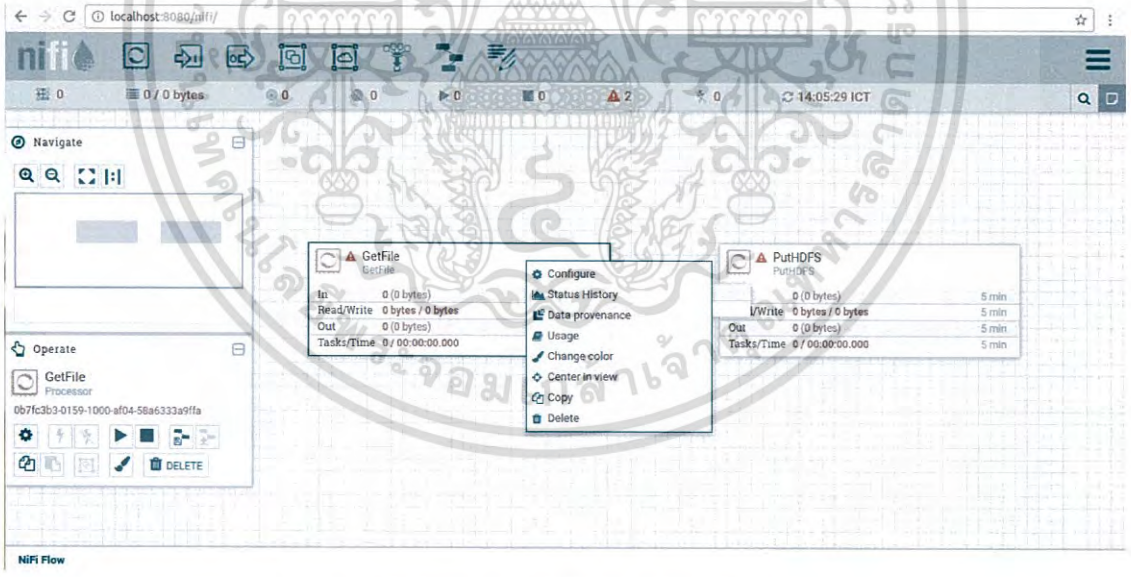
ภาพที่ 3.62 การเลือกการจัดลำดับการส่งข้อมูลเป็น FirstInFirstOutPrioritizer

จากภาพที่ 3.62 เป็นผลลัพธ์ของการลาก ลาก FirstInFirstOutPrioritizer ลงมาที่ช่อง Selected Prioritizers



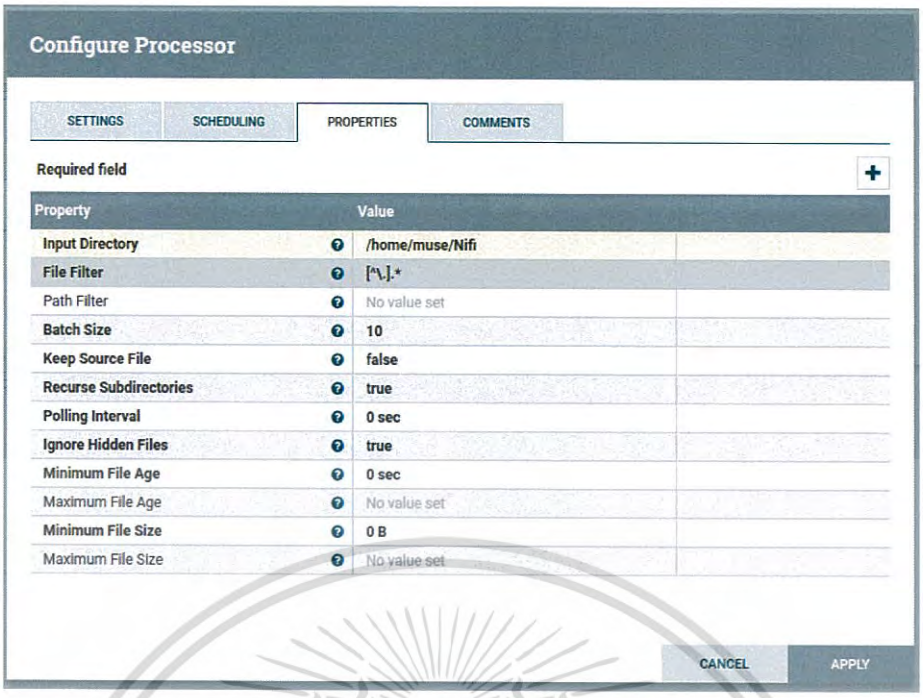
ภาพที่ 3.63 การเชื่อมต่อระหว่างโพรเซสเซอร์ GetFile และ PutHDFS

จากภาพที่ 3.63 เป็นผลลัพธ์จากการตั้งค่าโพรเซสเซอร์หลังจากกด add จากภาพที่ 3.62 ซึ่งแสดงให้เห็นถึงหน้าตาของการเชื่อมต่อระหว่างโพรเซสเซอร์ GetFile และ PutHDFS โดยจะขึ้นผลของการส่งไฟล์สำเร็จให้เห็นจากส่วนที่เชื่อมกัน



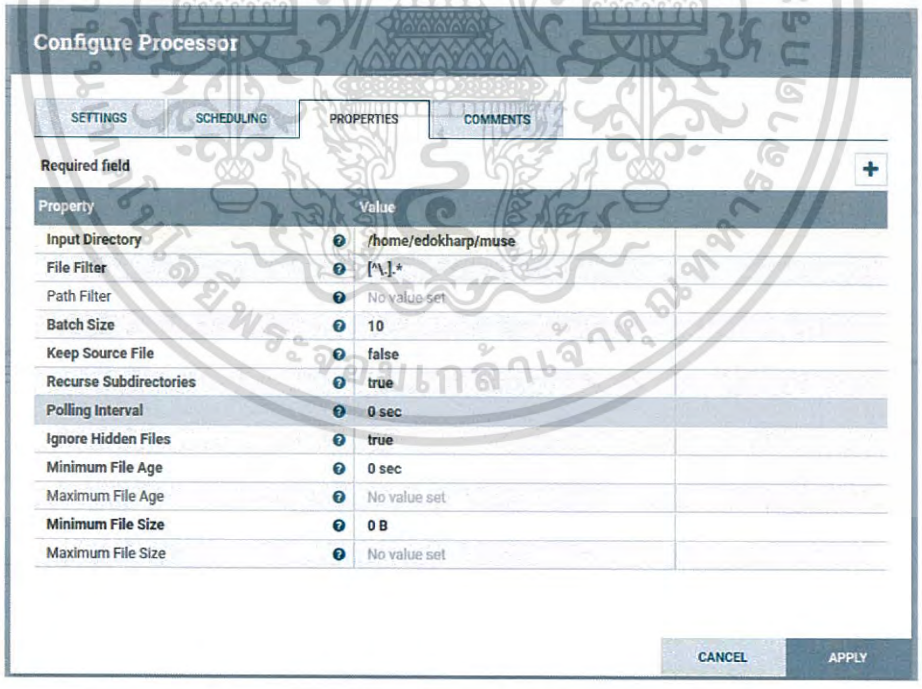
ภาพที่ 3.64 การตั้งค่าโพรเซสเซอร์ GetFile

จากภาพที่ 3.64 เป็นการตั้งค่าโพรเซสเซอร์ GetFile ให้รู้จักกับโฟลเดอร์ต้นทางซึ่งต้องเป็นโฟลเดอร์หรือตำแหน่งที่เป็นจุดเริ่มต้นในการส่งข้อมูลเข้านายฟาย การตั้งค่าสามารถเชื่อมกับต้นทางใดก็ได้ เพราะนายฟายสามารถรับไฟล์ต่างชนิดเข้ามาแล้วนำไปเก็บที่ปลายทางที่ต่างกันได้โดยที่รูปแบบของไฟล์ไม่เปลี่ยนแปลง และยังสามารถดูคิวการส่งข้อมูลได้ การตั้งค่าจะต้องคลิกขวาที่โพรเซสเซอร์แล้วเลือก configure



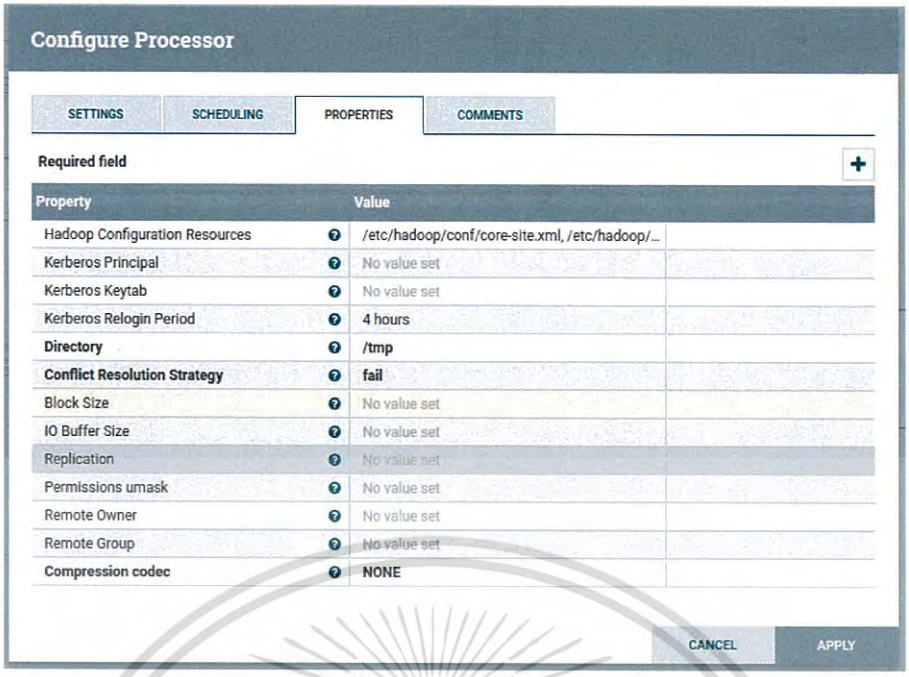
ภาพที่ 3.65 การตั้งค่าที่อยู่ของไฟล์ในการนำเข้าข้อมูล

จากภาพที่ 3.65 เป็นการตั้งค่าให้นายฟายรู้จักกับตำแหน่งที่จะไปดึงไฟล์มาเข้าตัวมันเอง ซึ่งต้องค่าการใส่ตำแหน่งของโฟลเดอร์ต้นทางที่ Input Directory



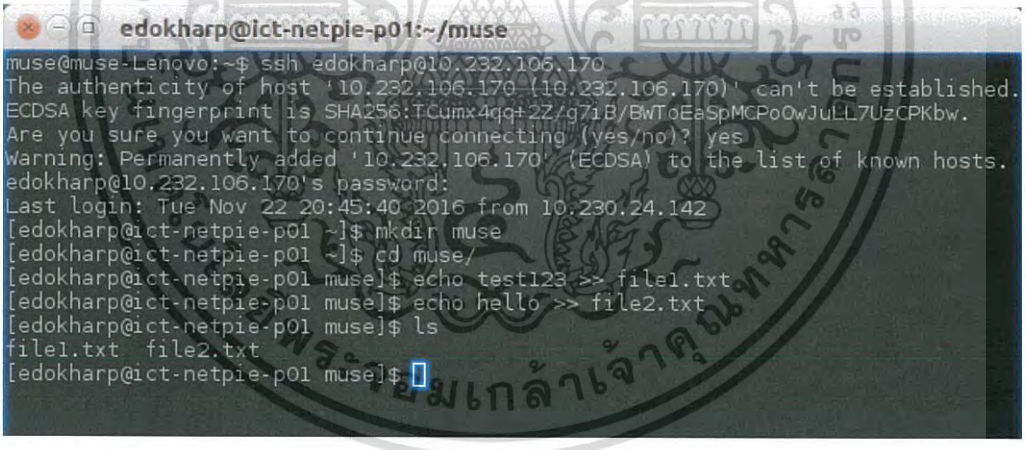
ภาพที่ 3.66 การตั้งค่าที่อยู่ของไฟล์ในการส่งออกข้อมูล

จากภาพที่ 3.66 เป็นการตั้งค่าให้นายฟายรู้จักตำแหน่งที่เก็บโฟลเดอร์ปลายทางซึ่งจะเป็นที่เก็บไฟล์เอชดีเอฟเอสที่โพรเซสเซอร์ PutHDFS จึงใส่ตำแหน่งของโฟลเดอร์เอชดีเอฟเอสที่ Input Directory



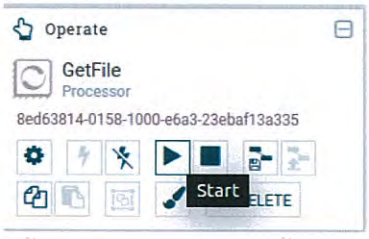
ภาพที่ 3.67 การตั้งค่า Hadoop Configuration Resources

จากภาพที่ 3.67 เป็นการใส่ตำแหน่งไฟล์ core-site.xml และ hdfs-site.xml ที่ Hadoop Configuration Resources



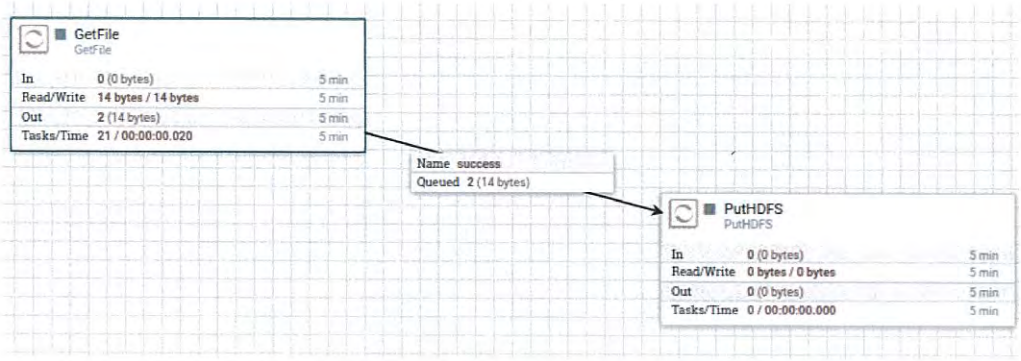
ภาพที่ 3.68 ชื่อไฟล์ที่อยู่ในโฟลเดอร์ต้นทาง

จากภาพที่ 3.68 เป็นการทดสอบการส่งข้อมูลโดยสร้างไฟล์ใส่ในโฟลเดอร์ต้นทาง file1.txt และ file2.txt



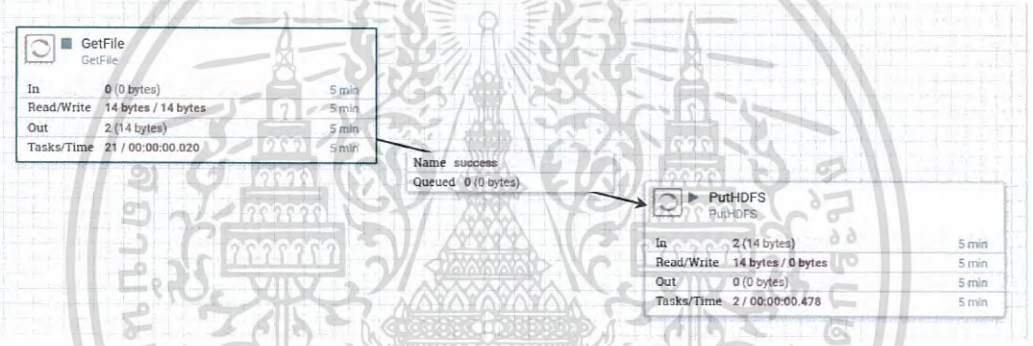
ภาพที่ 3.69 ปุ่มปฏิบัติงานของ โพรเซสเซอร์ GetFile

จากภาพที่ 3.69 เมื่อต้องการจะส่งไฟล์ให้คลิกโพรเซสเซอร์ GetFile แล้วเลือก Start ที่ Operate



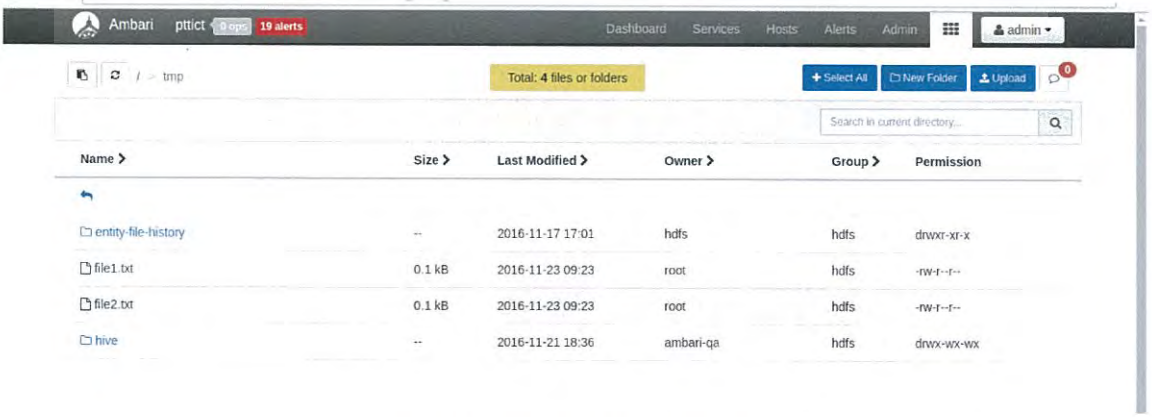
ภาพที่ 3.70 ข้อมูลเข้าคิวเพื่อส่งออกไปยังเอชดีเอฟเอส

จากภาพที่ 3.70 ไฟล์จะถูกส่งไปรอคิวตรงกลางระหว่างทั้ง 2 โพรเซสเซอร์



ภาพที่ 3.71 การส่งไฟล์ไปยังเอชดีเอฟเอส

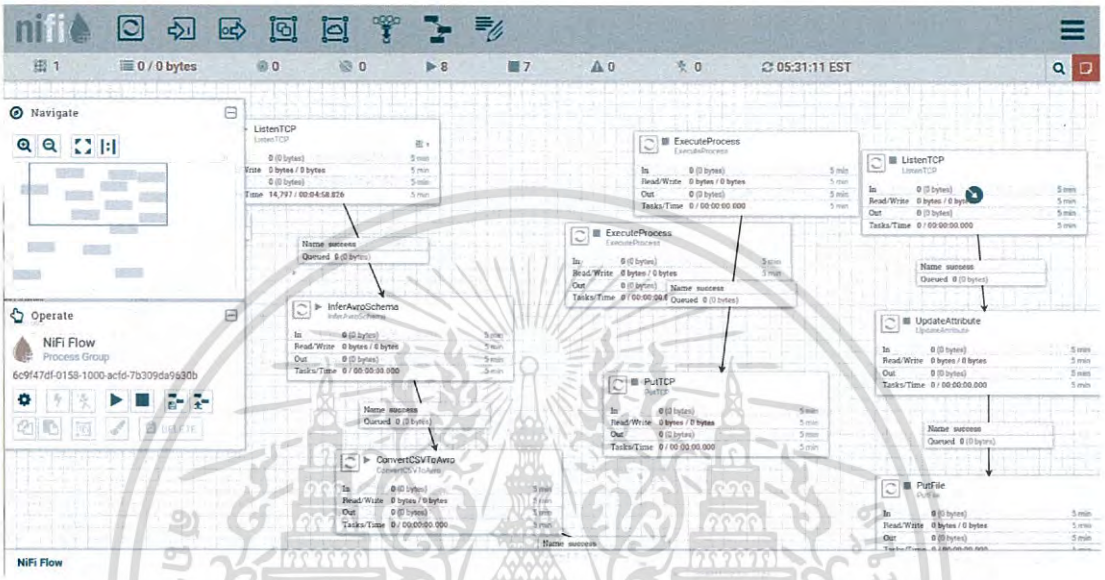
จากภาพที่ 3.71 เมื่อต้องการส่งไฟล์เข้าเอชดีเอฟเอสให้กด Start ที่ PutHDFS ไฟล์ที่เข้าคิวอยู่จะถูกส่งออกไป หากต้องการส่งไฟล์แบบทันทีให้กด Start ที่โพรเซสเซอร์ทั้งสองตัว ก่อนจะใส่ไฟล์ลงโฟลเดอร์ต้นทาง จากรูปจะสังเกตได้ว่า ไฟล์ออกจาก GetFile 2 ไฟล์ โดยดูจากค่า Out เท่ากับ 2 และไฟล์ได้เข้า PutHDFS 2 ไฟล์โดยดูจากค่า In เท่ากับ 2



ภาพที่ 3.72 ไฟล์ในเอชดีเอฟเอส

จากภาพที่ 3.72 ตรวจสอบไฟล์ที่ส่งไปโฟลเดอร์ปลายทางที่เอชดีเอฟเอสโดยสามารถมอนิเตอร์ผ่านอัมบารีได้

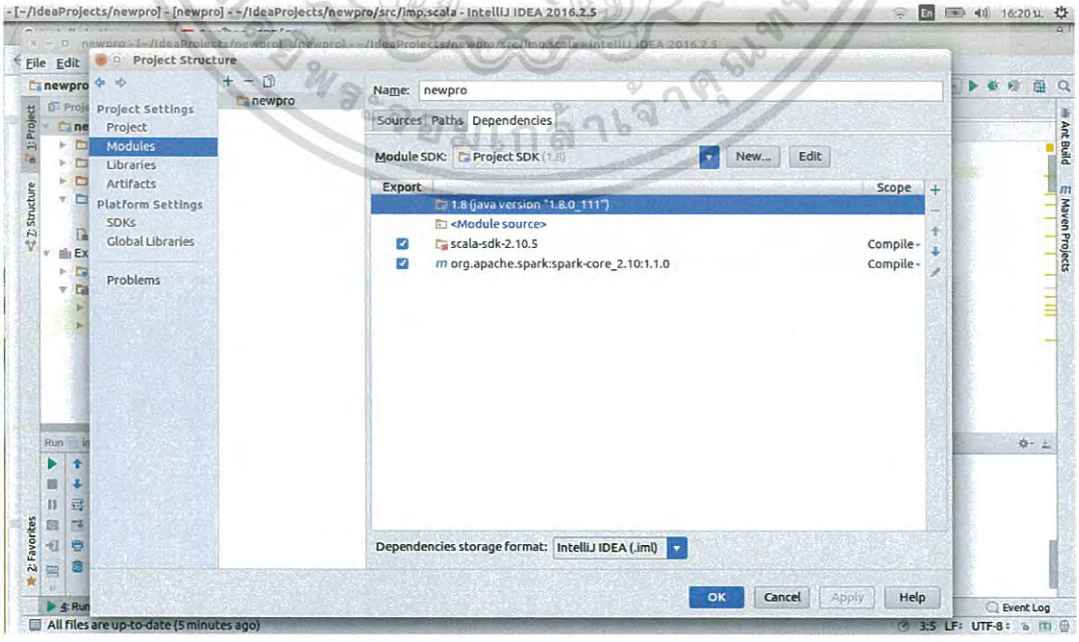
เบื้องต้นเป็นการทำงานพื้นฐานของนายฟายเท่านั้น ส่วนในงานของบริษัทนั้น ผู้จัดทำได้ตั้งค่าการทำงานของนายฟายให้ดึงข้อมูลจากระบบ TAS และระบบ MAS และส่งข้อมูลเข้าเอชดีเอฟเอสที่มีดูแลการทำงานผ่านอัมบารี ดังภาพที่ 3.73



ภาพที่ 3.73 การดึงข้อมูลจากระบบ TAS และระบบ MAS และส่งข้อมูลเข้าเอชดีเอฟเอส

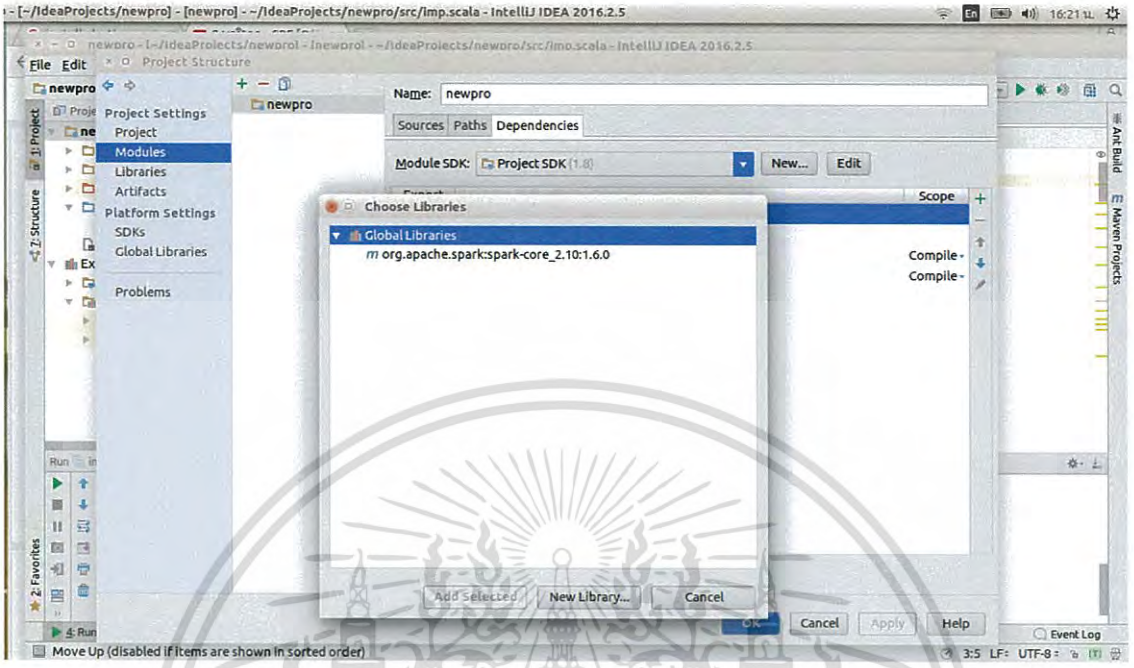
3.3.2 การวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกด้วยอาปาเช่สปาร์ก

ทำการเปิด IntelliJ idea ขึ้นมา



ภาพที่ 3.74 การตั้งค่าเอ็นไวรอนเม้นท์ของสกาล่า

จากภาพที่ 3.74 เป็นการตั้งค่าสิ่งแวดล้อมของสกล่าโดยเลือก file > project structure > dependencies



ภาพที่ 3.75 การตั้งค่าเอนไวรอนเม้นท์ของสกาล่า

จากภาพที่ 3.75 เป็นการเลือกที่เครื่องหมายบวก add > Libraries > New Library > From Maven > ค้นหา Spark core จากนั้นเลือก Spark core ให้ตรงกับเวอร์ชั่นสปาร์ก ของผู้ใช้งาน แล้วเลือก Spark core แล้ว add Selected > คลิกเลือก export Spark core ให้ขึ้นเครื่องหมายถูก > Ok

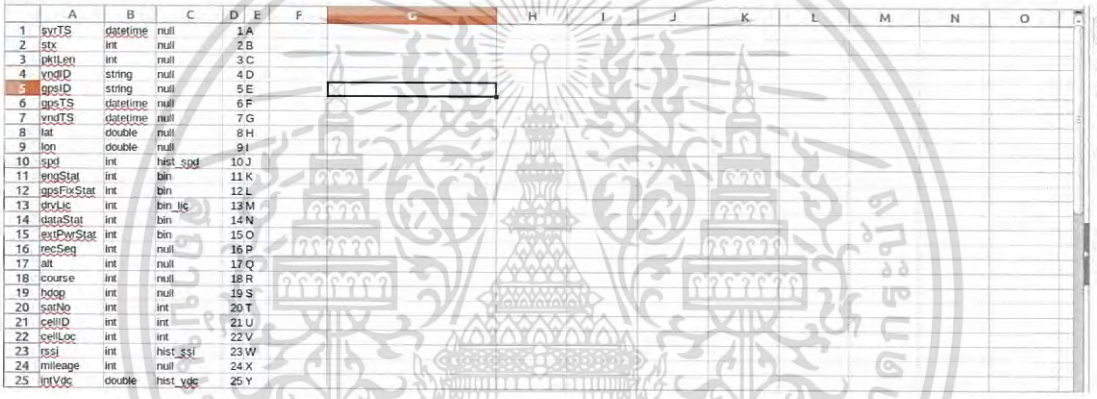
ภาพที่ 3.76 ไลบรารีภายนอก

จากภาพที่ 3.76 เป็นผลลัพธ์เมื่อทำสำเร็จจะได้ไลบรารีของสปาร์กขึ้นอยู่ฝั่งซ้าย

จากนั้นจึงนำข้อมูลมาวิเคราะห์หาความผิดปกติโดยข้อมูลที่นำมาใช้เป็นข้อมูลจากกล่องเก็บติดตามในรถขนน้ำมัน (IVMS) แต่ละคันของบริษัท ข้อมูลเหล่านี้ถูกเก็บไว้วิเคราะห์พฤติกรรมการขับรถ

ของคนขับแต่ละคัน วิเคราะห์อุบัติเหตุหรือวิเคราะห์ระยะทางและเวลาในการขนส่ง ซึ่งมีการใช้ข้อมูลหลายประเภท เช่น ตำแหน่งละติจูด ลองจิจูด เวลา ณ ตำแหน่งนั้น เวลาที่ข้อมูลถึงฐานเก็บข้อมูล ความเร็วของรถ การเบรกของรถ เลขประจำตัวของรถแต่ละคัน สถานะเครื่องยนต์ เลขประจำตัวคนขับ จำนวนครั้งที่ความเร็วเกินกำหนด จำนวนครั้งที่เปิดไฟซ้าย จำนวนครั้งในการคาดเข็มขัด จำนวนครั้งที่เปิดไฟหน้า จำนวนครั้งที่เปิดไฟหลัง เป็นต้น ซึ่งจากการแปลงข้อมูลมาเป็นกราฟแล้วสังเกตดู จะพบว่าบางครั้งข้อมูลตำแหน่งของรถคลาดเคลื่อนในเวลาที่ใกล้กัน สถานะเครื่องยนต์เป็นศูนย์ คือไม่ได้ติดเครื่องยนต์แต่รถกลับมีความเร็วหรือมีการเปิดไฟหน้าในเวลากลางวัน ทำให้เราค้นพบว่าข้อมูลในแต่ละแถวมีการขัดแย้งกันเองซึ่งเป็นการหาความผิดพลาดในเบื้องต้นจากการจับคู่ชนิดของข้อมูลและนับจำนวนที่เกิดขึ้นในแต่ละแถว

ข้อมูลที่ส่งมาจาก IVMS จะผ่านทางอินเตอร์เน็ตจากซิมการ์ดของเครือข่ายโทรศัพท์ ข้อมูลดังกล่าวได้ส่งมาเป็นตัวซึ่งแบ่งชนิดข้อมูลออกเป็น 49 คอลัมน์ ดังภาพที่ 3.77



	A	B	C	D	E
1	syrTS	datetime	null	1	A
2	stx	int	null	2	B
3	pktLen	int	null	3	C
4	vndID	string	null	4	D
5	gpsID	string	null	5	E
6	gpsTS	datetime	null	6	F
7	vndTS	datetime	null	7	G
8	lat	double	null	8	H
9	lon	double	null	9	I
10	spd	int	hist_spd	10	J
11	engStat	int	bin	11	K
12	gpsFixStat	int	bin	12	L
13	drvLic	int	bin_lic	13	M
14	dataStat	int	bin	14	N
15	extPwrStat	int	bin	15	O
16	recSeq	int	null	16	P
17	alt	int	null	17	Q
18	course	int	null	18	R
19	hdop	int	null	19	S
20	satNo	int	int	20	T
21	cellID	int	int	21	U
22	cellLoc	int	int	22	V
23	rssi	int	hist_rssi	23	W
24	mileage	int	null	24	X
25	intVdc	double	hist_vdc	25	Y

ภาพที่ 3.77 ไฟล์ที่เก็บชื่อและค่าของคอลัมน์ข้อมูล IVMS

จากภาพที่ 3.77 คอลัมน์ A เป็นชนิดของข้อมูลทั้ง 49 ชนิดที่ถูกนำมาใช้แบ่งข้อมูลตัวเลขในแต่ละแถวและคอลัมน์ C คือชนิดของตัวเลขที่ส่งเข้ามาซึ่งส่วนใหญ่จะเป็นเลขฐานสองและจำนวนเต็ม แต่ในส่วนของคอลัมน์ที่เป็น null จะไม่ถูกนำมาวิเคราะห์เพราะไม่สามารถบ่งบอกถึงความปกติหรือไม่ปกติได้

```
2016-08-18T00:00:00.007 171 198 3 359769034447166 2016-08-17T23:59:56.932 2016-08-18T00:00:00.000 16.878437 99.128740 0 0 1  1 1 19753 115 315
8 10 0 0 24 8694657 4.200000 25 0 0 0 0.000000 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.018 171 198 3 359769033398832 2016-08-17T23:59:57.987 2016-08-18T00:00:00.000 12.897088 101.125365 0 0 1  1 1 18072 80 44 8
10 0 0 12 16672321 4.000000 25 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.075 171 198 3 359769033387488 2016-08-17T23:59:58.009 2016-08-18T00:00:00.000 16.525835 100.143270 0 0 1  1 1 17278 39 205
8 10 0 0 19 2291687 4.300000 25 0 0 0 0.000000 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.080 171 198 3 359769033394518 2016-08-17T23:59:58.029 2016-08-18T00:00:00.000 13.897832 101.591750 56 1 1  1 1 17760 24 271
8 10 0 0 17 7488749 4.100000 27 0 0 0 0.000000 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.215 171 198 3 359769033396356 2016-08-17T23:59:57.165 2016-08-18T00:00:00.000 15.201643 104.859330 0 0 1  1 1 17893 129 234
8 11 0 0 22 24784096 4.300000 26 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.219 171 198 3 359769033389740 2016-08-17T23:59:57.155 2016-08-18T00:00:00.000 13.769238 100.691727 49 1 1  1 1 17468 0 332
8 10 0 0 25 6787213 4.300000 27 0 0 0 0.000000 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.259 171 198 3 359769034310083 2016-08-17T23:59:58.211 2016-08-18T00:00:00.000 8.089927 98.329607 0 0 1  1 1 18955 48 195 8
11 0 0 22 10579041 4.100000 26 0 0 0 0.000000 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.341 171 198 3 359769033389534 2016-08-17T23:59:57.287 2016-08-18T00:00:00.000 17.982568 100.042313 0 0 1  1 1 16994 155 18
9 10 0 0 17 28452724 4.000000 25 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.350 171 198 3 359769033397644 2016-08-17T23:59:57.294 2016-08-18T00:00:00.000 14.523372 100.856020 0 0 1  1 0 17990 9 87 8
10 0 0 16 13576758 3.700000 5 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.359 171 198 3 359769033397313 2016-08-17T23:59:56.295 2016-08-18T00:00:00.000 15.412720 102.479690 59 1 1 0024 1 00000559
58601 1 1 17962 155 27 8 10 0 0 16 24535770 4.300000 28 0 0 0 0.000000 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.385 171 198 3 359769034436300 2016-08-17T23:59:57.316 2016-08-18T00:00:00.000 16.214998 99.702395 48 1 1 0024 1 00003946
10400 1 1 19671 56 162 8 10 0 0 15 710474 4.200000 27 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.389 171 198 3 359769033387561 2016-08-17T23:59:57.330 2016-08-18T00:00:00.000 9.113025 99.344703 0 0 1  1 1 17285 17 58 8
10 0 0 21 28755674 4.000000 25 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.391 171 198 3 359769033388403 2016-08-17T23:59:58.322 2016-08-18T00:00:00.000 9.137123 99.243457 0 1 1 0024 1 00018852
80200 1 1 17355 -5 81 8 10 0 0 23 22329801 4.100000 28 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2016-08-18T00:00:00.393 171 198 3 359769034437399 2016-08-17T23:59:57.299 2016-08-18T00:00:00.000 14.528345 100.856475 0 0 1  1 1 19699 13 265
8 10 0 0 21 22736272 4.000000 26 0 0 0 0.000000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```


ภาพที่ 3.78 ข้อมูล IVMS

จากภาพที่ 3.78 เป็นข้อมูล IVMS ที่เรานำมาหาความผิดปกติมีทั้งหมด 3773461 แถว

และจากข้อมูลจะเห็นได้ว่ามีการเว้นวรรคของข้อมูลในแต่ละคอลัมน์แต่ต้องนำมาจัดเรียงใหม่เพื่อให้ตรงตามคอลัมน์และมีข้อมูลที่คอลัมน์ 13 บางแถวจะมีเลขเกินมา 3 ตัวและเว้นไม่เป็นระเบียบทำให้ต้องเขียนโปรแกรมเพื่อจัดคอลัมน์เพิ่ม โค้ดที่เขียนจะนำคอลัมน์ในรูปบนมาเก็บไว้ แล้วนำข้อมูลที่ผ่านการแบ่งช่วงและแก้ไขเลขที่คอลัมน์ 13 แล้วมาใส่ให้ตรงกัน จากนั้นมีการนำแต่ละคอลัมน์มาจับคู่กันเพื่อนับแถวที่มีทั้งสองคอลัมน์และมีค่าตรงกัน

บทที่ 4

ผลการวิจัย

โครงสร้างขนาดใหญ่และเครื่องมือต่าง ๆ ที่ทางผู้จัดทำและพี่เลี้ยงได้ศึกษาการทำงานและติดตั้งนั้น ทำให้การจัดการข้อมูลในฐานข้อมูลมีประสิทธิภาพมากขึ้น อีกทั้งการติดต่อประสานงานในเครือของบริษัทต้องมีการแสดงศักยภาพของแผนกและนำเสนอเทคโนโลยีใหม่ ๆ อยู่เสมอ เพื่อที่จะถูกเลือกให้ทำงานที่ตรงกับความสามารถในการพัฒนาเทคโนโลยี และจากการถูกเชิญไปประชุมและดูงานจากบริษัทต่าง ๆ เกี่ยวกับเทคโนโลยีข้อมูลขนาดใหญ่ พบว่าบริษัทส่วนใหญ่ยังไม่มีพนักงานที่มีความเชี่ยวชาญในด้านข้อมูลขนาดใหญ่ การนำเสนอของบริษัทส่วนใหญ่จะเป็นการซื้อเครื่องมือสำเร็จรูปของค่ายต่าง ๆ เช่น แซพ ไอบีเอ็ม ไมโครซอฟท์ และนำมาให้พนักงานในบริษัทใช้งาน แต่เครื่องมือเหล่านี้ อาจจะทำให้เกิดข้อจำกัดในการทำงานและความคิด การวิเคราะห์ข้อมูลต่าง ๆ อีกทั้งยังพบว่าบริษัทส่วนใหญ่ยังไม่มีการนำฮาดูปมาใช้งาน เพราะยังขาดพนักงานที่มีความรู้ในการติดตั้งและการใช้งาน ทางบริษัท พีทีที ไอซีที โซลูชั่นส์ จำกัด จึงเป็นบริษัทส่วนน้อยที่มีศักยภาพและความพร้อมในการทำงานมากกว่าบริษัทอื่น และได้รับการติดต่อจากบริษัทที่สนใจในเทคโนโลยีข้อมูลขนาดใหญ่ ได้รับการติดต่อว่าจ้างงานต่าง ๆ เพิ่มขึ้น ผู้จัดทำจึงจะนำเสนอผลลัพธ์จากโครงงานโครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกดังนี้

4.1 ความสามารถของเครื่องมือต่าง ๆ

เครื่องมือถูกแบ่งตามลักษณะการใช้งานหลัก ๆ ออกเป็น 4 ประเภท ดังนี้

4.1.1 เครื่องมือนำเข้าข้อมูล

อาปาเช่สคูปเป็นเครื่องมือนำเข้าข้อมูลที่สามารถเชื่อมต่อเข้ากับฐานข้อมูลได้ สามารถใช้งานได้ดีในการเคลื่อนย้ายข้อมูลระหว่างฐานข้อมูลเชิงสัมพันธ์กับฮาดูปแต่มีข้อเสียคือทำงานได้ช้าและระหว่างประมวลผลอาจเกิดความผิดพลาดบ่อยครั้งแล้วทำให้การประมวลผลหยุดลง จึงไม่เป็นที่นิยมใช้ในปัจจุบัน

อาปาเช่คาฟกาเป็นระบบการส่งข้อความแบบพับลิชซับสไครบ์ โดยยอมให้ผู้ใช้งานเชื่อมต่อโปรแกรมที่มีภาษาที่แตกต่างกันเข้าด้วยกันได้ สามารถทำงานได้หลาย ๆ เครื่องพร้อมกัน การทำงานมีความเร็วสูงมาก รองรับข้อมูลในปริมาณมากได้ สามารถนำมาเป็นเครื่องมือในการเพิ่มประสิทธิภาพ และลดความซับซ้อนของไปป์ไลน์ของข้อมูลในระบบขององค์กร แต่ข้อเสียคือคาฟกาไม่มีเลขประจำข้อความ ข้อความในคาฟกาจะถูกกำหนดที่อยู่โดยข้อมูลจราจรทำหน้าที่เป็นโบรกเกอร์คือเก็บข้อมูลไว้ในคิวแล้วรอให้คนมาดึงไป

อาปาเช่นายฟายเป็นเครื่องมือที่สามารถส่งไฟล์ได้หลากหลายรูปแบบ ไม่ว่าจะมีโครงสร้างหรือไม่มีโครงสร้าง โดยคุณสมบัติเด่นคือสามารถรับข้อมูลจากโปรโตคอลที่หลากหลาย เช่น FTP SFTP HTTP JMS UDP เป็นต้น และส่งข้อมูลออกไปโดยใช้โปรโตคอลที่หลากหลายจึงเหมาะสำหรับเป็นตัวกลางระหว่างระบบ ย้ายไปแพลตฟอร์มใด ๆ ได้ ใช้ได้กับอุปกรณ์ที่รันจาวาได้ เป็นที่นิยมใช้งานในปัจจุบัน ใช้งานง่ายเพราะเว็บอินเตอร์เฟส มีการการันตีการส่งข้อมูล มีการเข้าคิว อีกทั้งยังมีความปลอดภัยสูงมากอีกด้วย

ลอคสแตชเป็นท่อประมวลผลข้อมูลแบบการทำงานบนเซิร์ฟเวอร์ที่นำเข้าข้อมูลจากหลาย ๆ แหล่งพร้อมกัน โดยรองรับการนำเข้าข้อมูลที่หลากหลาย จากนั้นจะมีการแปลงข้อมูล โดยจะมีการ ฟิลเตอร์แล้วระบุชื่อของพื้นที่ในการสร้างโครงสร้างขึ้นมา และแปลงข้อมูลให้เหมาะสมต่อการนำไปทำการวิเคราะห์ข้อมูล การนำเข้าและการส่งออกจะครอบคลุมแทบทุกรูปแบบที่นิยมใช้งาน เช่น sylog http tcp udp csv standard output

4.1.2 เครื่องมือเก็บข้อมูล

เครื่องมือที่ใช้เก็บข้อมูลจะเป็นเครื่องมือหลักและเป็นตัวกลางในการเชื่อมต่อไปยังเครื่องมืออื่น ๆ มีดังนี้

ตารางที่ 1.2 ตารางแสดงความแตกต่างของคุณสมบัติระหว่างเครื่องมือเก็บข้อมูล

เครื่องมือเก็บข้อมูล	**ระบบไฟล์**	**ภาษา**	**การวิเคราะห์ข้อมูล**
อาปาเช่เฮชดีเอฟเอส	แบบกระจาย	จาวา	ทำได้โดยใช้แมพรีดิวซ์
อาปาเช่เอชเบส	แบบกระจาย	โนซีควล	ทำไม่ได้
อาปาเช่ไฮฟ์	แบบกระจาย	ไฮฟ์ซีควล	ทำไม่ได้
อีลาสติกเสิร์ช	เก็บที่เดียวโดยใช้ชื่อดัชนีในการเก็บ	จาวา	ทำได้ด้วยตัวเอง

อาปาเช่เฮชดีเอฟเอสเป็นเครื่องมือที่ใช้งานได้ฟรี และบริษัทในปัจจุบันนิยมใช้งานกันอย่างกว้างขวาง เช่น เฟสบุค ทวิตเตอร์ ด้วยการใช้งานอย่างแพร่หลายทำให้มีเครือข่ายช่วยเหลือปัญหาและตอบคำถามการใช้งานของฮาดูป การทำงานของเอชดีเอฟเอสเป็นการทำงานแบบกระจาย สามารถขยายพื้นที่การทำงานให้เพิ่มขึ้นหรือลดลงได้ง่าย ซึ่งจะมีประโยชน์ต่อการติดต่องานที่มีปัญหาพื้นที่ทำงานขนาดใหญ่ แต่ด้วยเฟรมเวิร์คที่มีประสิทธิภาพทำให้การจะนำระบบภายนอกมาทำงานร่วมกับฮาดูปนั้น

เป็นไปได้ยากเพราะบริษัทส่วนใหญ่ในปัจจุบันยังนิยมใช้งานซีควลอยู่ และยังขาดฟังก์ชันการลำดับความปลอดภัยของข้อมูลซึ่งเป็นสิ่งสำคัญในการปกป้องข้อมูล แต่ทั้งนี้บริษัทมีข้อมูลเกิดขึ้นจำนวนมากในทุก ๆ วัน จึงต้องมีการปรับตัวแพลตฟอร์มในการบริหารและจัดเก็บข้อมูล และฮาดูปช่วยแก้ปัญหานี้และเข้ามามีบทบาทสำคัญในปัจจุบัน

อาปาเช่เอชเบสสามารถเก็บข้อมูลและทำการวิเคราะห์ข้อมูลเชิงลึกควบคู่กับการทำงานของแมพรีดิวซ์ได้ รองรับการขยายขนาดกับเอชดีเอฟเอสได้ มีความคงทนต่อความเสียหาย มีความยืดหยุ่นเพราะไม่กำหนดรูปแบบที่ตายตัว ใช้โนซีควลในการทำงาน เหมาะกับข้อมูลที่ต้องการทำงานแบบทันที สามารถทำงานร่วมกันกับไฮฟ์สำหรับการสืบค้นข้อมูลแบบซีควลมีการทำการแบ่งและกระจายงานให้เหมาะสม แต่หากมีปัญหาเกิดขึ้นเอชเบสไม่มีเครือข่ายช่วยเหลือในการทำงาน สามารถเรียงและตั้งค่าดัชนีได้แค่กุญแจ แต่ขณะที่ฐานข้อมูลเชิงสัมพันธ์สามารถตั้งค่าดัชนีในส่วนไหนก็ได้ และยังไม่มีการยืนยันตัวตนภายในเอชเบสทำให้ยังไม่ปลอดภัยต่อการใช้งานซึ่งเป็นจุดสำคัญที่ควรจะมี ไม่แนะนำสำหรับผู้ใช้งานทั่วไปเพราะใช้งานยาก

อาปาเช่ไฮฟ์ทำงานคาบคู่กับเอชดีเอฟเอสได้ สนับสนุนการทำสืบค้นข้อมูลแบบซีควล จึงเรียกกันว่าไฮฟ์คิวแอล (HQL) ทำให้สามารถสืบค้นข้อมูลโดยผู้ใช้งานหลาย ๆ คนได้พร้อมกัน โครงสร้างตารางข้อมูลของไฮฟ์คล้ายคลึงกับโครงสร้างในฐานข้อมูลจึงเหมาะกับโปรแกรมเมอร์ที่ถนัดการใช้ซีควล รองรับข้อมูลเฉพาะที่มีรูปแบบโครงสร้างได้หลายแบบ รองรับการบันทึกทับแต่ไม่สามารถลบและอัพเดตได้ อีกทั้งการค้นหาข้อมูลทำได้เฉพาะการสืบค้นพื้นฐาน ไม่สามารถใส่เงื่อนไขในการสืบค้นได้สำหรับไฮฟ์คิวแอล

ตารางที่ 1.3 ตารางแสดงความแตกต่างของคุณสมบัติระหว่างฐานข้อมูลเชิงสัมพันธ์และอีลาสติกเสิร์ช

ฐานข้อมูลเชิงสัมพันธ์	อีลาสติกเสิร์ช
ฐานข้อมูล (Database)	ดัชนี (Index)
ตาราง (Table)	ชนิด (Type)
แถว (Row)	เอกสาร (Document)
คอลัมน์ (Column)	เขตข้อมูล (Field)
โครงสร้างข้อมูล (Schema)	การแปลงรูป (Mapping)

อีลาสติกเสิร์ชการนำเข้าข้อมูลต้องมีการใส่ชื่อดัชนี ชื่อประเภท และไอดีให้ข้อมูลด้วย ทำให้มีการจัดระเบียบข้อมูลและความสามารถในการค้นหาข้อมูลทำได้ดีมาก เป็นเทคนิคการค้นหา

ข้อความแบบใหม่ที่ครอบคลุมกว่าเดิม ใช้เทคนิคการค้นหาด้วยคำศัพท์หรือวลีที่ถูกต้องตามพจนานุกรม และค้นหาคำที่สะกดใกล้เคียงกันได้ ซึ่งเป็นจุดเด่นของการทำงานของอีลาสติกเสิร์ช การนำเข้าข้อมูลหากไม่ใส่เลขไอดีเพื่อระบุตำแหน่งในการเก็บข้อมูล อีลาสติกเสิร์ชจะใส่ให้อัตโนมัติ ซึ่งไม่แนะนำเพราะจะส่งผลให้ค้นหาข้อมูลได้ยาก ข้อเสียของอีลาสติกเสิร์ชคือ ต้องมีการอัพเดตบ่อย และไม่มีความทันทีหรือการทำงานตลอดเวลา การวิเคราะห์ข้อมูลในเชิงลึกยังมีประสิทธิภาพไม่สูงเท่าเครื่องมืออื่น ๆ ที่ใช้ในการวิเคราะห์ข้อมูลในเชิงลึกโดยตรง

4.1.3 เครื่องมือประมวลผลข้อมูล

การประมวลผลข้อมูลมีทั้งการประมวลแฟ้มคำสั่งรวมและการประมวลผลในทันที ซึ่งสิ่งที่เครื่องมือประมวลผลควรทำได้คือการประมวลผลแฟ้มคำสั่งรวมและการเลือกใช้ส่วนใหญ่ขึ้นอยู่กับความเร็วในการประมวลผลเนื่องจากข้อมูลมีขนาดใหญ่มากจึงต้องใช้การประมวลที่มีประสิทธิภาพและความเร็วสูงมีดังนี้

ตารางที่ 1.4 ตารางแสดงความแตกต่างของคุณสมบัติระหว่างเครื่องมือประมวลผลข้อมูล

เครื่องมือประมวลผลข้อมูล	รูปแบบการประมวลผล	ความเร็ว
แมพรีดิวซ์	ชุดคำสั่งของไฟล์	ช้า
อาปาเช่สปาร์ก	ชุดคำสั่งของไฟล์ และ เรียลไทม์	เร็วกว่าแมพรีดิวซ์ 10-100 เท่า
อาปาเช่ฟลิงค์	ชุดคำสั่งของไฟล์ และ เรียลไทม์	เร็วใกล้เคียงอาปาเช่สปาร์ก
อาปาเชสตอร์ม	ชุดคำสั่งของไฟล์ และ เรียลไทม์	ช้า

แมพรีดิวซ์รองรับการใช้ภาษาจาวา และสามารถประมวลผลแบบเชิงกลุ่มได้ และประมวลผลเฉพาะข้อมูลที่เป็นข้อความได้ดี เป็นส่วนหนึ่งของฮาดูปทำให้เชื่อว่าการประมวลผลข้อมูลมีความเร็วสูงเพราะสามารถดึงข้อมูลจากเอชดีเอฟเอสได้โดยตรง แต่จากการทดสอบและใช้งานจริงกับข้อมูลปริมาณมาก ปรากฏว่าแมพรีดิวซ์มีความเร็วต่ำกว่าสปาร์ก 10 ถึง 100 เท่า

อาปาเช่สปาร์กประมวลผลชิ้นของข้อมูลที่รู้จักกันว่าชุดข้อมูลกระจายความยืดหยุ่น (RDDs) ทำงานโดยการมีส่วนควบคุมหลักและจะกระจายไฟล์ของชุดข้อมูลกระจายความยืดหยุ่นให้ส่วนอื่นช่วยประมวลผลด้วย มีความเร็วสูงเมื่อเทียบกับแมพรีดิวซ์สามารถประมวลผลแบบเชิงกลุ่มและประมวลผลแบบทันทีได้ รองรับการดำเนินการเรียนรู้ของเครื่อง (MLlib) รองรับการสืบค้นด้วยซีควล (SparkSQL)

ประมวลผลกราฟด้วย GraphX รองรับทั้งภาษาจาวา ไพทอนและสกาล่า ซึ่งในบางภาษายังมีข้อจำกัดในการใช้งานอยู่ การประมวลผลในตัวเองอาจเกิดสภาวะคอขวดขึ้นได้ในกรณีที่มีข้อมูลมหาศาลและสปาร์กไม่มีระบบจัดการไฟล์จึงต้องทำงานร่วมไปกับฮาดูปหรือแพลตฟอร์มข้อมูลอื่น ๆ

อาปาเช่ฟลิงค์เป็นการทำงานประมวลผลแบบทำซ้ำ สามารถประมวลผลแบบเชิงกลุ่มและประมวลผลแบบทันทีได้ ซึ่งการประมวลผลแบบสตรีมมิ่งเป็นการประมวลผลที่ใช้สตรีมข้อมูลจริง เฟรมเวิร์คการประมวลผลข้อมูลในเชิงลึกสามารถขยายไปทำงานร่วมกันกับฮาดูปได้ดี ทำให้ทำงานร่วมกับสตอร์มและแมพรีดิวซ์ได้ ทั้งนี้การใช้งานฟลิงค์เพิ่มขึ้นในเวลาอันรวดเร็วเพราะความสามารถในการทำงานของตัวมันเอง แต่สปาร์กก็ยังเป็นที่นิยมใช้งานมากกว่าในบริษัทต่าง ๆ

อาปาเชสตอร์มเป็นเฟรมเวิร์คประมวลผลแบบสตรีมมิ่ง การสตรีมข้อมูลจะเป็นแบบทูเปิล การทำงานสามารถรองรับได้หลายภาษา เช่น จาวา สกาล่า ไพทอน รูบี้ เป็นต้น หากมีการทำงานล้มเหลวจะสามารถกลับมาทำงานใหม่ได้อย่างรวดเร็ว

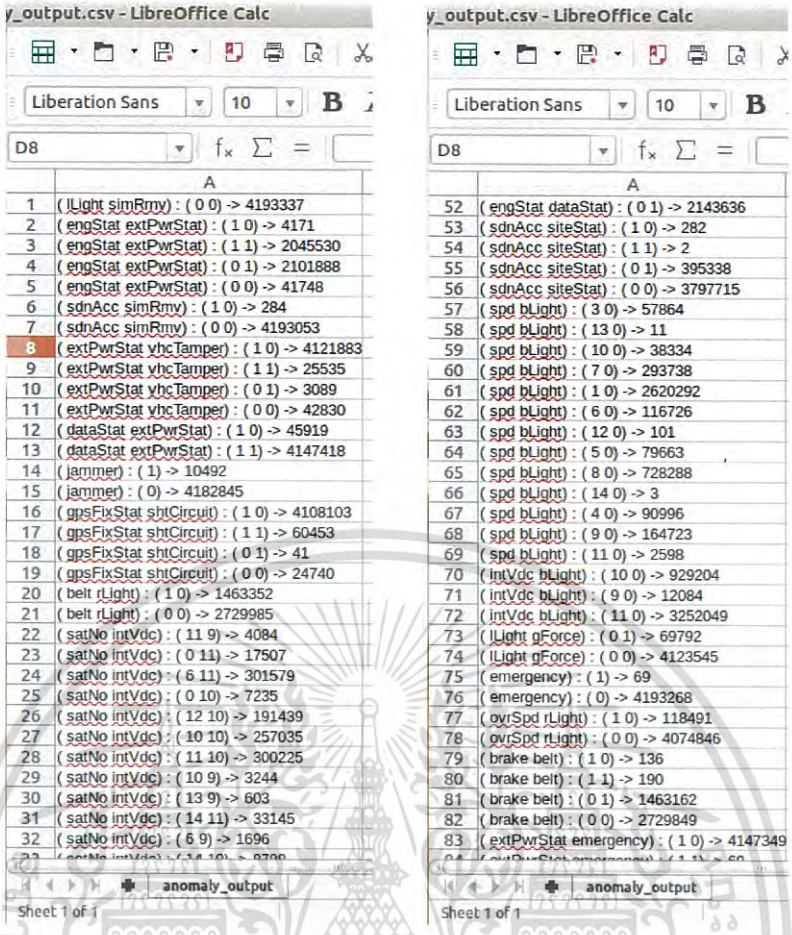
4.1.4 เครื่องมือแสดงผลข้อมูล

เครื่องมือแสดงผลที่เหมาสำหรับการใช้งานข้อมูลขนาดใหญ่คือคีบานาเพราะสามารถแสดงผลได้หลายรูปแบบตามต้องการ สามารถทำกราฟต่าง ๆ ได้ เรียกดูข้อมูลตามความต้องการได้ นำแผนที่จริงจากแผนที่ของกูเกิลมาใส่เสริมได้ โดยหากข้อมูลเรามีละติจูด ลองจิจูด ก็สามารถบอกตำแหน่งของข้อมูลได้อย่างเที่ยงตรง สามารถนำผลข้อมูลที่ได้ทำเป็นลิงก์แล้วไปใส่ในเว็บได้ หรือแปลงเป็นไฟล์ชนิดอื่นเพื่อการใช้งานที่เหมาะสมได้ แต่ยังมีข้อจำกัดคือใช้งานยาก ต้องมีความเชี่ยวชาญทางด้านนี้พอสมควร การนำข้อมูลเข้าต้องกำหนดค่าของข้อมูลก่อนโดยต้องเข้าใจความหมายของข้อมูลนั้น และแปลความหมายของข้อมูลให้คีบานาเข้าใจด้วย จึงจะสามารถนำมาใช้งานในการแสดงผลต่าง ๆ ได้อย่างดี

ทั้งนี้การจะติดตั้งเครื่องมือสามารถทำได้ง่ายโดยผู้จัดทำแนะนำให้ติดตั้งอัมบารีก่อน จากนั้นเข้าใช้งานอัมบารีผ่านเวปไซต์แล้วทำการติดตั้งเครื่องมืออื่น ๆ ผ่านอัมบารีจะเป็นการลดระยะเวลาในการศึกษาวิธีการติดตั้งและการตั้งค่าเครื่องมือต่าง ๆ ให้เชื่อมถึงกัน โดยเราสามารถดูแลจัดการเครื่องมือต่าง ๆ ผ่านอัมบารีมีความสะดวกรวดเร็วและง่ายต่อการใช้งาน

4.2 ผลของการวิเคราะห์หาความผิดปกติของข้อมูลน้ำมันในเชิงลึก

ผลลัพธ์ที่ได้จากการหาความผิดปกติของข้อมูลในเชิงลึกทำการจับคู่ข้อมูลได้ทั้งหมด 959 ชุด โดยข้อมูลบางชนิดอาจมี 2 ค่าหรือหลายค่าก็ได้ ข้อมูลที่เป็นจำนวนเต็มส่วนใหญ่จะถูกแบ่งออกเป็นหลายค่า เช่น เปิดไฟหน้าเป็น1 ปิดไฟหน้าเป็น0 ความเร็วในช่วง0ถึง60เป็น0 ความเร็วในช่วง61ถึง110เป็น1 ความเร็วในช่วง111ถึง150เป็น2 เป็นต้น การกระทำนี้ส่งผลให้เราสามารถหาความผิดปกติเพิ่มได้อีก



ภาพที่ 4.1 ผลลัพธ์ที่ได้จากการหาความผิดปกติของข้อมูล

จากภาพที่ 4.1 จะสังเกตเห็นว่าแถวที่ 1 มี จำนวนข้อมูลที่เหมือนกันของ lLight เป็น 0 และ simRmv เป็น 0 มีทั้งหมด 4193337 ซึ่งถือว่ามีจำนวนมาก ไม่ถือเป็นข้อมูลที่ผิดปกติ แต่กลับกันกับแถวที่ 54 ซึ่งมีข้อมูลอยู่ 2 แถวจากทั้งหมด 3773461 แถว ถือเป็นข้อมูลผิดปกติอย่างแน่นอน ซึ่งในส่วนของการจัดการและนำผลลัพธ์ไปใช้งาน ทางผู้จัดทำได้ส่งผลลัพธ์ต่อให้กับผู้ดูแลและจัดการข้อมูลเพื่อนำไปตรวจสอบและลดระยะเวลาในการดูข้อมูลทีละจุดว่าส่วนไหนมีความผิดปกติเกิดขึ้นบ้าง และนำไปพัฒนาและแก้ไขความผิดปกติของอุปกรณ์ที่ใช้งานต่อไป ไม่ว่าจะเป็นสัญญาณเครือข่ายในการส่งข้อมูล เซนเซอร์ต่าง ๆ ในรถ หรือกล่อง IVMS ในรถขนส่งน้ำมัน

บทที่ 5

บทสรุป

5.1 สรุปผลการดำเนินงาน

โครงสร้างข้อมูลขนาดใหญ่และการวิเคราะห์หาความผิดปกติของข้อมูลในเชิงลึกเป็นเครื่องมือที่พัฒนาขึ้นมาเพื่อเก็บข้อมูลที่เกิดจากอุปกรณ์ต่าง ๆ ที่เกิดขึ้นในแต่ละวันและมีปริมาณมากขึ้น ช่วยลดเวลาในการดูแลจัดการฐานข้อมูลต่าง ๆ อีกทั้งยังช่วยป้องกันการสูญหายของข้อมูลในกรณีที่ฐานข้อมูลไม่สามารถรองรับข้อมูลเพิ่มได้อีก และยังเป็นการพัฒนาศักยภาพของบริษัทโดยเป็นการนำเทคโนโลยีใหม่มาปรับใช้กับการดำเนินงานของบริษัทเอง แสดงให้เห็นถึงความก้าวหน้าและความพร้อมในการรับมือกับเทคโนโลยีใหม่ ส่งผลให้ได้รับมอบหมายงานใหม่ ๆ จากบริษัทในเครือมากยิ่งขึ้น ซึ่งการดำเนินการของโครงงานนี้แบ่งออกเป็น 2 ส่วนหลัก คือ

5.1.1 โครงสร้างข้อมูลขนาดใหญ่

1. อาปาเช่ฮาดูปเอชดีเอฟเอสเป็นโครงสร้างสำหรับเก็บข้อมูลขนาดใหญ่ ใช้จัดเก็บข้อมูลที่ส่งมาจากอุปกรณ์ต่าง ๆ โดยสามารถดูแลการทำงานผ่านอาปาเช่อัมบารีได้โดยตรง การเข้าใช้งานและจัดการ สามารถจำกัดสิทธิ์การเข้าถึงของผู้ใช้งานแต่ละคนได้ ทำให้ผู้ใช้งานในระดับต่าง ๆ สามารถเข้าไปทำงานในส่วนของตนเองได้โดยไม่ต้องทำเรื่องอนุมัติขอข้อมูลซึ่งต้องใช้เวลาอย่างน้อย 1 วัน

2. อาปาเช่นายฟายเป็นเครื่องมือนำข้อมูลจากภายนอกเข้าสู่ฮาดูปข้อมูลที่นำเข้าจะเป็นข้อมูลที่มีรูปแบบโครงสร้างเป็นส่วนใหญ่

5.1.2 วิเคราะห์ความผิดปกติของข้อมูล

การวิเคราะห์ความผิดปกติของข้อมูลจะทำโดยใช้อาปาเช่สปาร์กซึ่งจะใช้ภาษาสกาล่าในการจัดเรียงคอลัมน์ข้อมูลและตัวเลขข้อมูลจากอุปกรณ์ได้ตรงกัน จับคู่คอลัมน์ของข้อมูลทั้งหมดแล้วนับข้อมูลที่มีคอลัมน์และค่าตรงกัน ผลลัพธ์ที่ได้จะถูกนำส่งต่อไปยังคนดูแลจัดการฐานข้อมูล

5.2 ข้อเสนอแนะและแนวทางในการพัฒนา

1. เครื่องคอมพิวเตอร์ที่ใช้ติดตั้งอาปาเช่อัมบารีและอาปาเช่ฮาดูปควรมีประสิทธิภาพสูง และใช้อินเตอร์เน็ตแลนเพื่อป้องกันอินเตอร์เน็ตขาดการเชื่อมต่อระหว่างติดตั้ง

2. การนำเข้าข้อมูลเป็นการรับข้อมูลจากอุปกรณ์ต่าง ๆ ซึ่งในหลายครั้งเกิดจากความผิดพลาด

ในการส่งของอุปกรณ์ ในอนาคตจึงควรติดตั้งมินิฟายไว้ที่อุปกรณ์ เพื่อให้การส่งข้อมูลเข้านายฟายเกิดขึ้นได้ในทันที

เอกสารอ้างอิง

[1] "Apache Hadoop" [ออนไลน์]. เข้าถึงได้จาก http://hortonworks.com/apache/hadoop/ (วันที่สืบค้นข้อมูล 15 มิถุนายน 2559)

[2] "Apache Hadoop HDFS" [ออนไลน์]. เข้าถึงได้จาก http://hortonworks.com/apache/hdfs/ (วันที่สืบค้นข้อมูล 16 มิถุนายน 2559)

[3] "Apache Nifi" [ออนไลน์]. เข้าถึงได้จาก https://nifi.apache.org/docs/nifi-docs/html/getting-started.html (วันที่สืบค้นข้อมูล 17 มิถุนายน 2559)

[4] "Apache Kafka" [ออนไลน์]. เข้าถึงได้จาก http://hortonworks.com/apache/kafka/#section_2 (วันที่สืบค้นข้อมูล 20 มิถุนายน 2559)

[5] "Apache Storm" [ออนไลน์]. เข้าถึงได้จาก http://storm.apache.org/about/simple-api.html (วันที่สืบค้นข้อมูล 21 มิถุนายน 2559)

[6] "Apache Spark" [ออนไลน์]. เข้าถึงได้จาก http://hortonworks.com/apache/spark/#section_3 (วันที่สืบค้นข้อมูล 22 มิถุนายน 2559)

[7] "Apache Hbase" [ออนไลน์]. เข้าถึงได้จาก https://hbase.apache.org/ (วันที่สืบค้นข้อมูล 23 มิถุนายน 2559)

[8] "Apache Hive" [ออนไลน์]. เข้าถึงได้จาก https://cwiki.apache.org/confluence/display/Hive/Home (วันที่สืบค้นข้อมูล 24 มิถุนายน 2559)

[9] "Apache Ambari" [ออนไลน์]. เข้าถึงได้จาก https://ambari.apache.org/ (วันที่สืบค้นข้อมูล 12 กรกฎาคม 2559)

[10] "Elasticsearch" [ออนไลน์]. เข้าถึงได้จาก

https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-elasticsearch-on-ubuntu-14-04 (วันที่สืบค้นข้อมูล 1 กรกฎาคม 2559)

[11] "Apache Sqoop" [ออนไลน์]. เข้าถึงได้จาก

http://hortonworks.com/apache/sqoop/#section_1 (วันที่สืบค้นข้อมูล 5 กรกฎาคม 2559)

[12] "รู้จักกับภาษา Scala พี่น้องของ Java" [ออนไลน์]. เข้าถึงได้จาก

https://www.nomkhonwaan.com/2016/02/13/scala-1-getting-started-with-scala (วันที่สืบค้นข้อมูล 7 กรกฎาคม 2559)

[13] "IntelliJ idea" [ออนไลน์]. เข้าถึงได้จาก

https://www.jetbrains.com/idea/ (วันที่สืบค้นข้อมูล 14 กรกฎาคม 2559)

[14] "Apache Flink" [ออนไลน์]. เข้าถึงได้จาก

https://www.quora.com/What-is-the-difference-between-Apache-Flink-and-Apache-Spark (วันที่สืบค้นข้อมูล 19 กรกฎาคม 2559)

[15] "Kibana" [ออนไลน์]. เข้าถึงได้จาก

https://www.digitalocean.com/community/tutorials/how-to-use-kibana-dashboards-and-visualizations (วันที่สืบค้นข้อมูล 21 กรกฎาคม 2559)

[16] "Virtual box" [ออนไลน์]. เข้าถึงได้จาก

http://pariwatvirtualbox.blogspot.com/2016/07/virtual-box.html (วันที่สืบค้นข้อมูล 5 สิงหาคม 2559)

[17] "Vagrant คืออะไร" [ออนไลน์]. เข้าถึงได้จาก

http://www.bongbank.net/tech/development/setup-development-environment-with-vagrant/ (วันที่สืบค้นข้อมูล 5 สิงหาคม 2559)

[18] "Apache Nifi Architecture" [ออนไลน์]. เข้าถึงได้จาก

http://nifi.apache.org/docs/nifi-docs/html/overview.html (วันที่สืบค้นข้อมูล 20 สิงหาคม 2559)

[19] "Apache Kafka Partition" [ออนไลน์]. เข้าถึงได้จาก

https://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.3.2/bk_kafka-user-guide/content/ch_using_kafka.html (วันที่สืบค้นข้อมูล 24 สิงหาคม 2559)

[20] "Apache Kafka Cluster" [ออนไลน์]. เข้าถึงได้จาก

http://www.whizlabs.com/blog/apache-kafka-what-is-it/ (วันที่สืบค้นข้อมูล 25 สิงหาคม 2559)

[21] "Apache Storm Cluster" [ออนไลน์]. เข้าถึงได้จาก

http://storm.apache.org/releases/current/Tutorial.html (วันที่สืบค้นข้อมูล 25 สิงหาคม 2559)

[22] "Spouts and Bolts" [ออนไลน์]. เข้าถึงได้จาก

http://hortonworks.com/hadoop-tutorial/processing-streaming-data-near-real-time-apache-storm/ (วันที่สืบค้นข้อมูล 25 สิงหาคม 2559)

[23] "Apache Storm Stream Processing" [ออนไลน์]. เข้าถึงได้จาก

http://www.openscalability.com/p/kafka-storm-twitter-geeknight/storm.html#/step-1 (วันที่สืบค้นข้อมูล 26 สิงหาคม 2559)

[24] "Apache Spark Architecture" [ออนไลน์]. เข้าถึงได้จาก

http://hortonworks.com/apache/spark/ (วันที่สืบค้นข้อมูล 26 สิงหาคม 2559)