รายงานการเข้าร่วมอบรม

Big Data Programming

จัดโดย

เขตอุตสาหกรรมซอฟต์แวร์ประเทศไทย (Software Park Thailand)

ณ อาคารเขตอุตสาหกรรมซอฟต์แวร์ประเทศไทย

ชั้น 3 ถนนแจ้งวัฒนะ จังหวัดนนทบุรี

วันที่ 20-23 ธันวาคม 2559

ผู้รายงาน

ผู้ช่วยศาสตราจารย์ ดร. วฤษาย์ ร่มสายหยุด

สาขาวิชาวิทยาศาสตร์และเทคโนโลยี

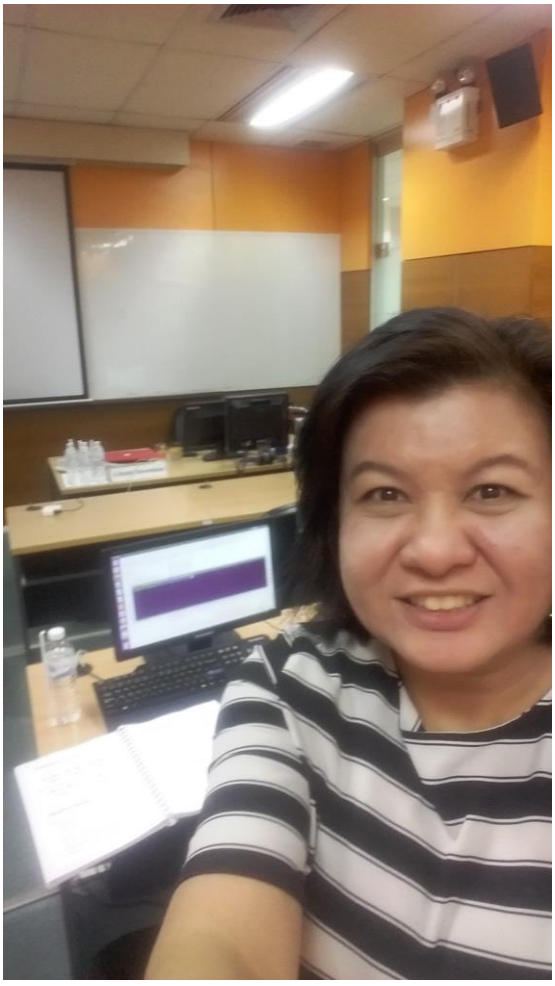
โครงการนี้ได้รับการสนับสนุนจากทุนพัฒนาบุคลากรตามความต้องการของหน่วยงาน

ประจำปี 2560

รายงานการไปอบรม
ตามระเบียบมหาวิทยาลัยสุโขทัยธรรมาธิราช ว่าด้วยการให้ทุนฝึกอบรม ดูงาน
และประชุมทางวิชาการแก่พนักงานมหาวิทยาลัยสุโขทัยธรรมาธิราช

---

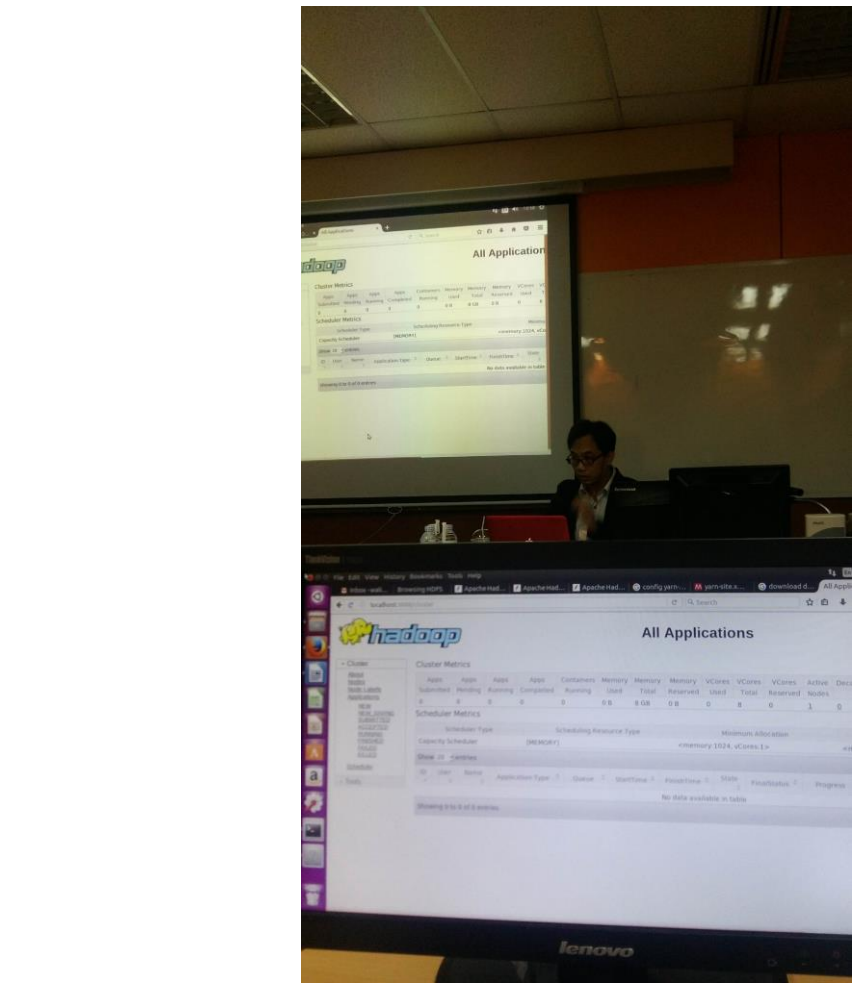
1. ชื่อ ผู้ช่วยศาสตราจารย์ วฤษาย์ ร่มสายหยุด อายุ 45 ปี ตำแหน่ง อาจารย์ สังกัดสาขาวิชาวิทยาศาสตร์และเทคโนโลยี โทร 8264 ไปเข้าร่วมอบรม เรื่อง Big Data Programming วันที่ 20 – 23 ธันวาคม 2559 รวมระยะเวลา 4 วัน

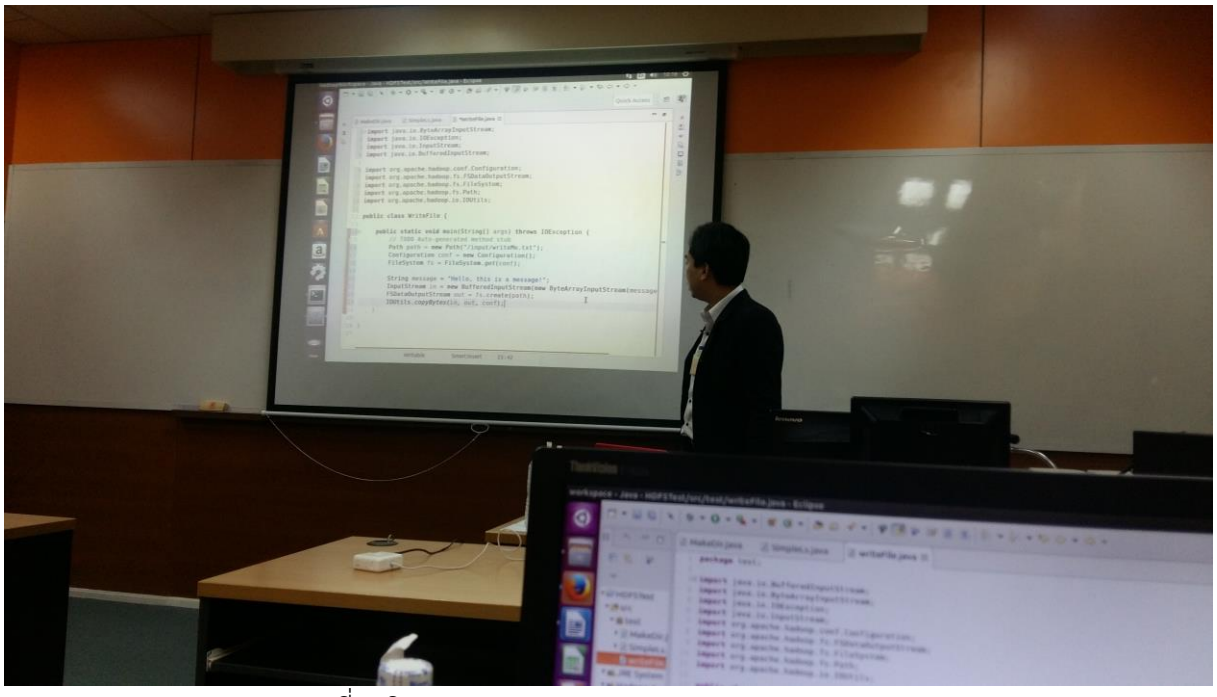


ภาพที่ 1 ผศ.ดร.วฤษาย์ ร่มสายหยุด ผู้เข้ารับการอบรม Big Data Programming

2. รายละเอียดเกี่ยวกับการไปฝึกอบรม ดูงาน ประชุม และสัมมนา

2.1 วิทยากร : Mr.Dendej Sawarnkatat



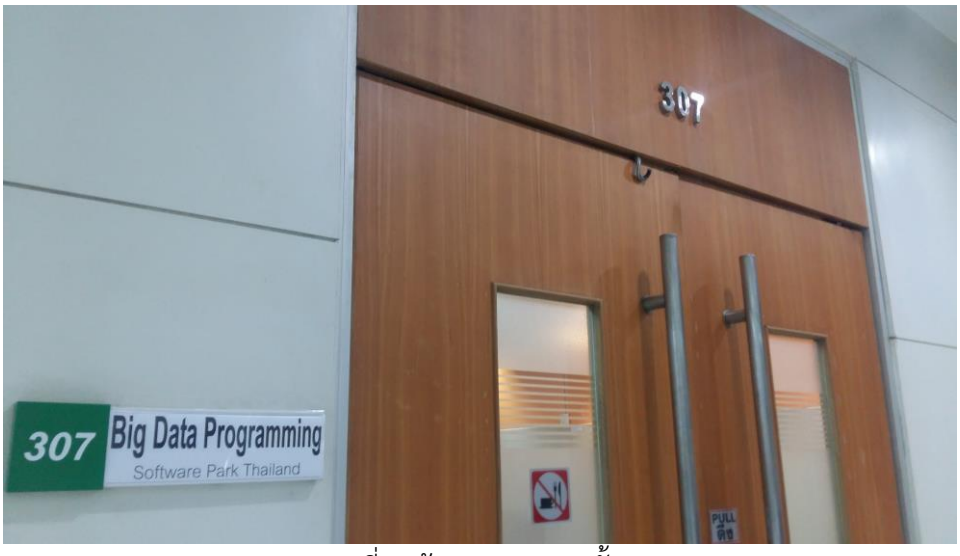


ภาพที่ 2 วิทยากรบรรยาย Big Data Programming

2.2 วัน-เวลา อบรม : วันที่ 20-23 ธันวาคม 2559 เวลา 9:00-16.30 น.

2.3 สถานที่อบรม : สำนักงานวิทยาศาสตร์และเทคโนโลยีแห่งชาติ ณ อาคารเขตอุตสาหกรรมซอฟต์แวร์ประเทศไทย ชั้น 3 ถนนแจ้งวัฒนะ จังหวัดนนทบุรี

ภาพที่ 3 สถานที่อบรมอาคารเขตอุตสาหกรรมซอฟต์แวร์ประเทศไทย



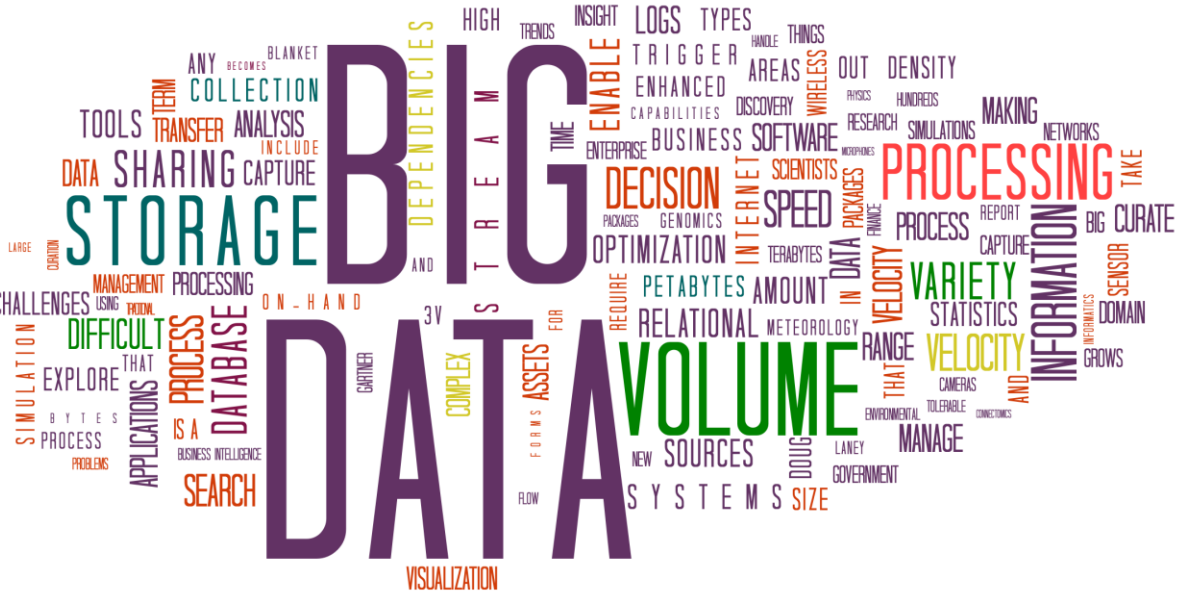
ภาพที่ 4 ห้องอบรม 307 ชั้น 3

2.4 เนื้อหาการบรรยาย

<u>วันที่ 1 (20 ธันวาคม 2559)</u>

2.4.1 ความรู้เบื้องต้นเกี่ยวกับข้อมูลใหญ่ (Big Data)

ข้อมูลใหญ่ (Big Data) คือการที่มีข้อมูลดิจิตอลขนาดมหาศาล โดย Big Data คือ เทคนิค หรือเทคโนโลยีในการกลั่น หรือวิเคราะห์ สกัด เอาคุณค่าออกมาจากข้อมูลขนาดใหญ่ ซึ่งเกินขอบเขต หรือขีดจำกัดของการจัดการข้อมูลแบบเดิมๆ ดังภาพที่ 5



ภาพที่ 5 ข้อมูลใหญ่

ที่มา : http://www.telecomjournalthailand.com/big-data-ใหญ่กว่าชื่อ/

โดยทาง Gartner ได้นิยามคุณลักษณะของ Big Data ไว้ด้วย 3V คือ high-volume, high-velocity และ high-variety ดังภาพที่ 6 ได้แก่

Volume คือปริมาณของข้อมูลมากมายมหาศาลจนไม่สามารถที่จะเก็บรวบรวมในฐานข้อมูลรูปแบบเดิมๆได้ หรือถ้าเก็บได้ก็อาจจะยากและซับซ้อน

Velocity หรือความเร็ว คือข้อมูลที่เปลี่ยนแปลงตลอดเวลา โดยข้อมูลนั้นจะมีค่าในเวลา ณ จุดนั้น และจดหมดค่าเมื่อเวลาผ่านไป การนำเอาข้อมูลชนิดนี้มาใช้ต้องมีการประมวลผลที่รวดเร็วทันต่อเหตุการณ์

Variety หรือความหลากหลาย คือข้อมูลที่มีความหลากหลายทางโครงสร้าง ทำให้ไม่สามารถที่จะจัดเก็บหรือแยกหมวดหมู่ได้ตามที่ต้องการ หรือมีรายละเอียดปลีกย่อย หรือรูปแบบในการจัดการข้อมูลที่เพิ่มมากขึ้นตามขนาดของข้อมูล

ภาพที่ 6 คุณลักษณะของ Big Data -3V

ที่มา : http://www.telecomjournalthailand.com/big-data-ใหญ่กว่าชื่อ/

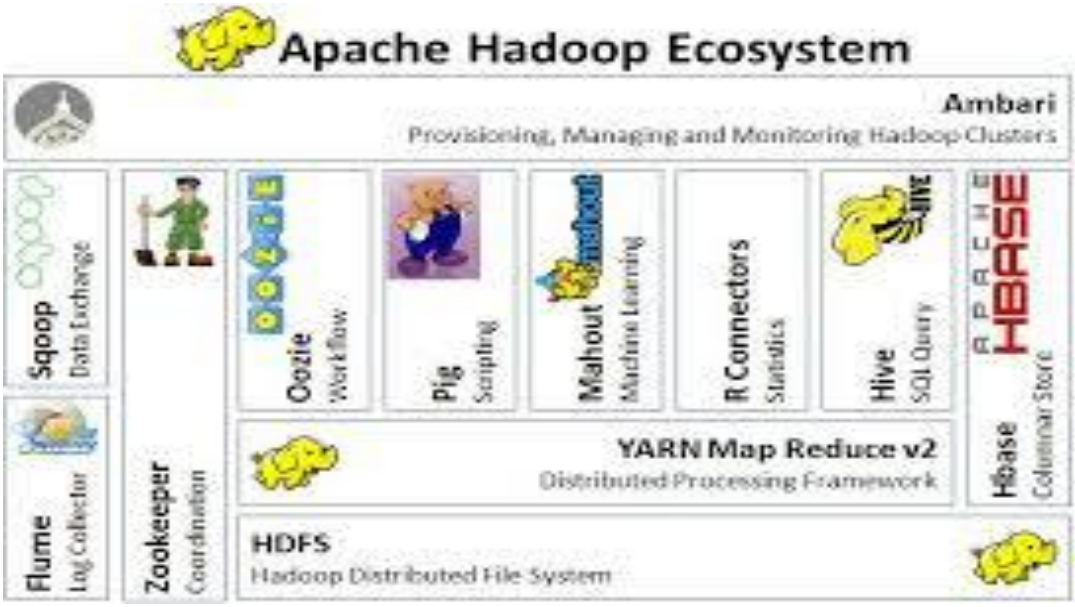
Big Data จะเป็นประโยชน์ต่อการใช้งานหลายประการ เช่น การใช้งานข้อมูลเกี่ยวกับการค้นคว้า วิจัย เอกสาร เครือข่ายทางสังคม หรือข้อมูลเฉพาะต่างๆ เช่น โรงพยาบาล คลังต่างๆ เป็นต้น ซึ่ง Big Data นี้เหมาะสำหรับการนำมาวิเคราะห์ข้อมูลดิบ หรือข้อมูลกึ่งโครงสร้างต่างๆ นำไปใช้ในการวิเคราะห์พฤติกรรมลูกค้า หรือธุรกิจที่เกี่ยวข้อง เพื่อหาวิธีการแก้ไขหรือหาวิธีการจัดการให้ธุรกิจเป็นไปตามที่คาดหวัง ไม่ว่าจะเป็นด้านธุรกิจ ที่จะเพิ่มโอกาสทางธุรกิจทำให้เกิดนวัตกรรมด้านเทคนิคที่สามารถรวบรวมและจัดเก็บข้อมูลได้ง่ายยิ่งขึ้น และทางด้านการเงินที่สามารถคิดเป็นเปอร์เซ็นต์ค่าใช้จ่ายไอทีได้ด้วย ซึ่งปัจจุบันนี้มีเครื่องมือที่ได้รับความนิยมเข้ามามีส่วนช่วยในการจัดการก็คือ Hadoop ที่ถูกพัฒนามาจาก Open Source Technology สามารถเก็บข้อมูลขนาดใหญ่และนำไปประมวลผลได้ แต่การวิเคราะห์ Big Data นั้นเป็นเพียงการวิเคราะห์ข้อมูลดิบแบบย่อยเท่านั้น หากต้องการข้อมูลที่เจาะลึกมากขึ้นไปอีกก็ต้องเพิ่มขั้นตอนการวิเคราะห์แบบ Analytics ที่จะทำให้ได้ข้อมูลในเชิงลึกมากขึ้นตามต้องการ

แนวโน้มในการนำ Big Data Analytics มาใช้งานนั้นมีเป้าหมายรองรับการขยายตัวธุรกิจอย่างชัดเจนดังนี้ ต้องมีความเร็วในการนำข้อมูลมาวิเคราะห์และใช้งานได้รวดเร็วและง่ายพอที่จะรับมือกับผู้ใช้งานแผนกต่างๆ ได้ และด้วยปริมาณข้อมูลที่เติบโตอย่างรวดเร็วเช่นกันหากสามารถนำมารวบรวมและวิเคราะห์ได้ก็จะเป็นการใช้ประโยชน์จากสินทรัพย์ขององค์กรเพื่อสร้างก้าวต่อไปที่แข็งแกร่งได้อีก และหากช่วยให้บริษัทสามารถเข้าใจลูกค้าของตนได้ลึกซึ้ง ได้ภาพรวมธุรกิจ และวิเคราะห์ตอบโจทย์เพื่อสร้าง Customer Experience ที่ให้ความแตกต่างทางธุรกิจได้ก็จะยิ่งเป็นการต่อยอดธุรกิจให้เหนือคู่แข่ง ที่สำคัญการนำข้อมูลที่เฉียบคมเหล่านี้มาใช้งาน ต้องสามารถตอบสนองทิศทางของแต่ละแผนกได้ดี และไม่ควรใช้งานยากเกินไป ทางที่ดี คือ ต้องง่ายจนแทบไม่ต้องเทรนเลย เพื่อให้เข้าถึงจิตใจลูกค้า หรือผู้บริโภคเป้าหมายของบริษัทได้แม้รายละเอียดเล็กน้อย ทุกวันนี้การทำการค้าที่เข้าถึงใจผู้บริโภคและมีความคล่องตัวในการให้การบริการแบบหลากหลายช่องทางนั้นคือกุญแจแห่งความสำเร็จ

2.4.2 ความรู้เบื้องต้นเกี่ยวกับอาปาเช่ฮาดูป (Apache Hadoop) และ Hadoop Distributed File System (HDFS) หรือเอชดีเอฟเอส

ซอฟต์แวร์ที่สำคัญตัวหนึ่งที่มีการนำมาใช้กันมาในระบบ Big Data คือ Hadoop เพราะ Hadoop เป็น Open Source Technology ที่จะทำหน้าที่เป็น Distributed Storage ที่สามารถเก็บข้อมูลขนาดใหญ่ที่เป็น Unstructured และนำมาประมวลผลได้ โดยองค์ประกอบหลักๆของ Hadoop จะประกอบด้วย Hadoop Distributed File System (HDFS) ที่ทำหน้าที่เป็น Storage และ MapReduce ที่ใช้ในการพัฒนาโปรแกรมประมวลผล ทั้งนี้โครงสร้างด้าน Hardware ของ Hadoop จะใช้เครื่อง Commodity Server จำนวนมากต่อเป็น Cluster กัน ในปัจจุบันหลายๆองค์กรจะใช้ Hadoop Technology ในการพัฒนา Big Data อาทิเช่น Facebook, Yahoo และ Twitter โดยจะมีเครื่อง Server 5 -1,000 เครื่อง ทั้งนี้ขึ้นอยู่กับขนาดข้อมูล นอกจากนี้ Technology Vendor ต่างๆอาทิเช่น Oracle, IBM, EMC หรือแม้แต่ Microsoft ต่างก็นำ Hadoop มาใช้ในเทคโนโลยีของตัวเองในการพัฒนาผลิตภัณฑ์ทางด้าน Big Data

ทั้งนี้ Hadoop จะไม่ได้นำมาแทนที่ระบบฐานข้อมูลเดิมแต่เป็นการใช้งานร่วมกันทั้ง Database แบบเดิมที่เป็น Structure Data และการนำ Unstructured Data ขององค์กรที่อาจเก็บไว้ในระบบอย่าง Hadoop เข้ามาพิจารณาร่วมกับข้อมูลอื่นๆภายนอกเช่น Facebook แล้วนำมาวิเคราะห์ข้อมูลโดยใช้เครื่องมืออย่าง Business Intelligence ดังภาพที่ 7



ภาพที่ 7 Apache Hadoop Ecosystem

ที่มา : http://www.siamhtml.com/getting-started-with-big-data-and-hadoop-spark-on-cloud-dataproc/

จากภาพที่ 7 Apache Hadoop Ecosystem เป็นการดำเนินการเกี่ยวกับ 3 ส่วนใหญ่ๆ ได้แก่

1. Storage คือ การจัดเก็บข้อมูล นั่นคือเรื่อง Volume และ Variety เนื่องจากข้อมูลนั้นไม่มีรูปแบบที่ชัดเจนและไม่สามารถกำหนดได้เหมือนกับ RDBMS ดังนั้นจึงต้องการที่จัดเก็บแบบใหม่ด้วยเทคโนโลยีที่อยู่ภายใต้ชื่อ Hadoop ซึ่งสามารถแบ่งตามคุณลักษณะได้ 3 กลุ่มใหญ่ ๆ ดังนี้

3. Distributed data ข้อมูลจะกระจายไปทำงานหลาย ๆ เครื่อง หรือ node
4. Cluster computing กระบวนการทำงานของแต่ละ node จะอยู่ภายใต้ cluster ซึ่งเป็น software ที่เชื่อมแต่ละ node เข้าด้วยกัน เหมือนกับว่าทำงานอยู่ในเครื่องหรือ ระบบเดียวกัน
5. Massive parallel processing ระบบการประมวลผลภายใน cluster สามารถทำงานแบบขนานกันได้ ซึ่งช่วยให้การทำงานเร็วขึ้น

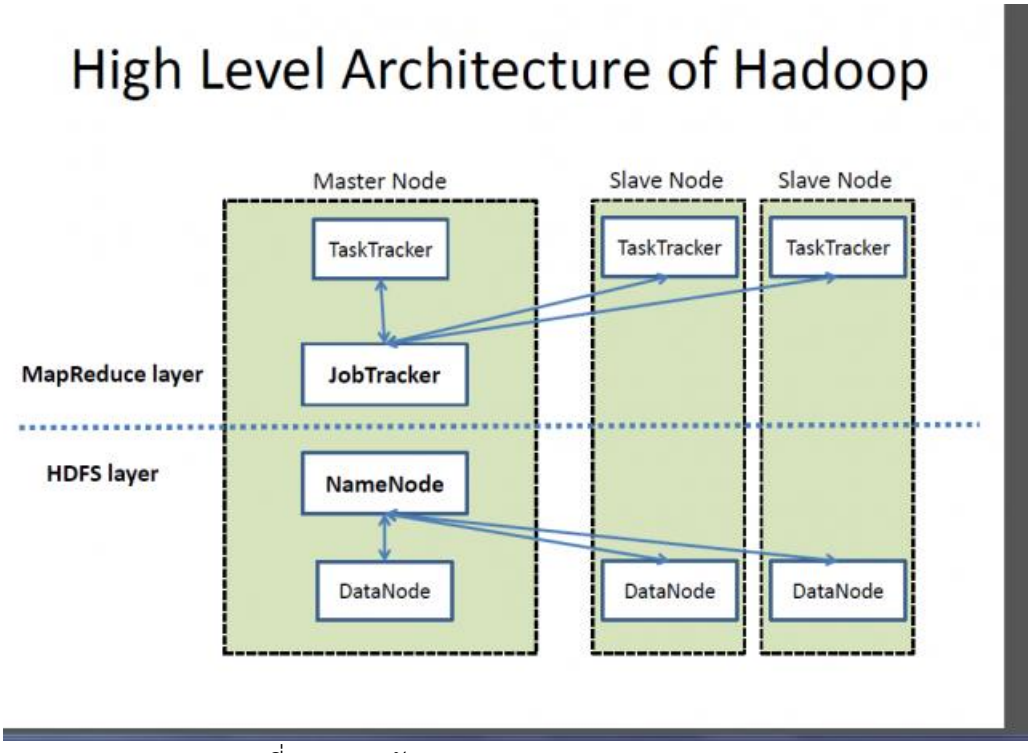
2. Processing คือ การประมวลผล นั่นคือเรื่อง Volume และ Velocity ข้อมูลจะไร้ค่าอย่างมาก ถ้าปราศจากการประมวลผล ซึ่งมีรูปแบบการประมวลผล 2 แบบ คือ

6. Batch เป็นการประมวลผลที่ใช้เวลานาน
7. Streaming เป็นการประมวลผลแบบ realtime

3. Analytic คือ การวิเคราะห์ นั่นคือกระบวนการวิธีสำหรับเข้าถึงข้อมูลเชิงลึกจาก 1 และ 2 ข้อมูลในโลกของ Big Data นั้น ไม่เหมาะสมอย่างยิ่งที่จะนำมาวิเคราะห์ ดังนั้น จึงต้องทำการแปลงข้อมูลไปอยู่ในข้อมูลที่มีรูปแบบก่อนเสมอ โดยเทคนิคในการวิเคราะห์ประกอบไปด้วย

8. Data mining
9. Predictive analytic
10. Text analytic
11. Video analytic
12. Social media analytic
13. Sentiment analytic
14. Location analytic
15. Machine learning

1.4.3 สถาปัตยกรรมของ Hadoop



ภาพที่ 8 : สถาปัตยกรรมของ Hadoop

ที่มา : https://opensource.com/life/14/8/intro-apache-hadoop-big-data

1.4.4 Hadoop Distributed File System (HDFS) : โมดูลนี้จะมีไว้เพื่อจัดเก็บข้อมูลที่จะนำมาวิเคราะห์ให้อยู่ในรูปที่สามารถเข้าถึงได้อย่างรวดเร็ว รวมไปถึงการสำรองข้อมูลดังกล่าวให้โดยอัตโนมัติ โดย MapReduce : ส่วนโมดูลนี้จะมีไว้เพื่อการประมวลผลข้อมูลปริมาณมหาศาลที่ได้เก็บเอาไว้

โดยใน Hadoop ประกอบด้วยโหนด (Node) นั้นหมายถึงเครื่องคอมพิวเตอร์ที่ประกอบไปด้วย CPU, RAM แล้วก็ Disk ซึ่ง Node ต่างๆ ใน Hadoop จะแบ่งออกเป็น 2 แบบด้วยกัน

Data Node: เป็น Node ที่ทำหน้าที่เก็บ Block ของไฟล์เอาไว้ และรับผิดชอบในการประมวลผล Block นั้นๆ แต่ตัว Data Node เอง จะไม่รู้ว่า Block ที่ตัวเองเก็บอยู่นั้น เป็นของไฟล์ไหน

Name Node : เป็น Node ที่ทำหน้าที่รวบรวมผลของการประมวลผล Block ต่างๆ จาก Data Node ทั้งหลาย ซึ่งแน่นอนว่า Name Node นี้ มันจะต้องรู้ทุกอย่างเกี่ยวกับไฟล์ต้นฉบับ ไม่ว่าจะเป็นชื่อไฟล์, ขนาด รวมไปถึงที่อยู่ของแต่ละ Block ที่ถูกกระจายออกไปตาม Data Node ต่างๆ หรือพูดง่ายๆ Name Node มันก็คือ Master ส่วน Data Node ก็คือ Slave นั่นเอง

1.4.5 การติดตั้งและใช้งาน Hadoop ใน Setting up a Single Node Cluster บน Ubuntu โดย Linux มีทั้งหมด 5 ขั้นตอน ดังต่อไปนี้

ขั้นตอนที่ 1 เตรียมซอฟต์แวร์ที่เกี่ยวข้อง

1) Ubuntu 16.04.1 : https://www.ubuntu.com/download/desktop
2) Hadoop 2.7.3 Release Notes : http://hadoop.apache.org/docs/r2.7.3/hadoop-project-dist/hadoop-common/releasenotes.html
3) JAVA version 8 update 111 : https://java.com/en/download/
4) Eclipse Neon : http://www.eclipse.org/downloads/eclipse-packages/

ขั้นตอนที่ 2 ติดตั้ง ssh เพื่อเขียนคำสั่ง Script ในการทำงานบน Ubuntu

```
$ sudo apt-get install ssh
$ sudo apt-get install rsync
```

ขั้นตอนที่ 3 Unpack JAVA

```
# set to the root of your Java installation
export JAVA_HOME=/usr/java/latest
```

ทดสอบการ run JAVA ใน Hadoop

```
$ bin/Hadoop
```

ขั้นตอนที่ 4 ทำการ Configure ที่ไฟล์ etc/hadoop/core-site.xml:

```
Hadoop-env.
JAVA_HOME=:readlink /usr/bin/java “s:bin/java::
core-site.xml
<property>
<name>fs.default.name</name>
```

```
<value>hdfs://localhost:9100</value>
</property>
<property>
  <name>mapred.job.tracker</name>
  <value>localhost:9101</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/var/hadoop_data/namenode</value>
</property>
<property>
  <name>dfs.datanode.name.dir</name>
  <value>file:/var/hadoop_data/datanode</value>
</property>
```

ขั้นตอนที่ 5 ติดตั้ง Hadoop YARN

คำสั่ง format ที่ NameNode

```
$ bin/hdfs namenode -format
```

คำสั่ง Start

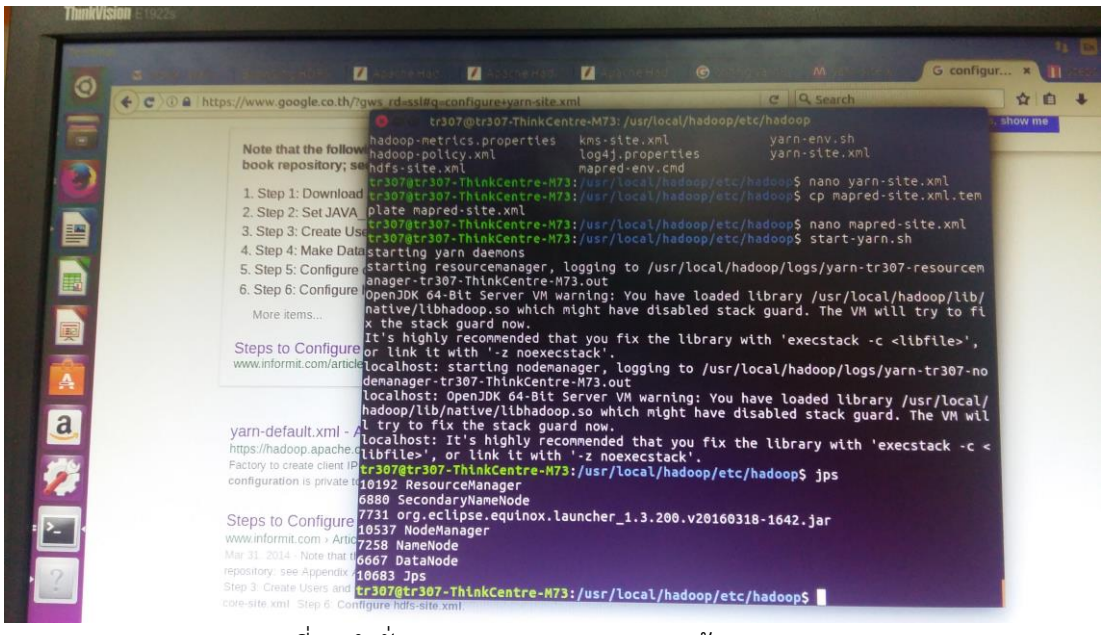
```
$ sbin/start-dfs.sh
```

คำสั่งกำหนดค่า Web

```
http://localhost:50070/
```

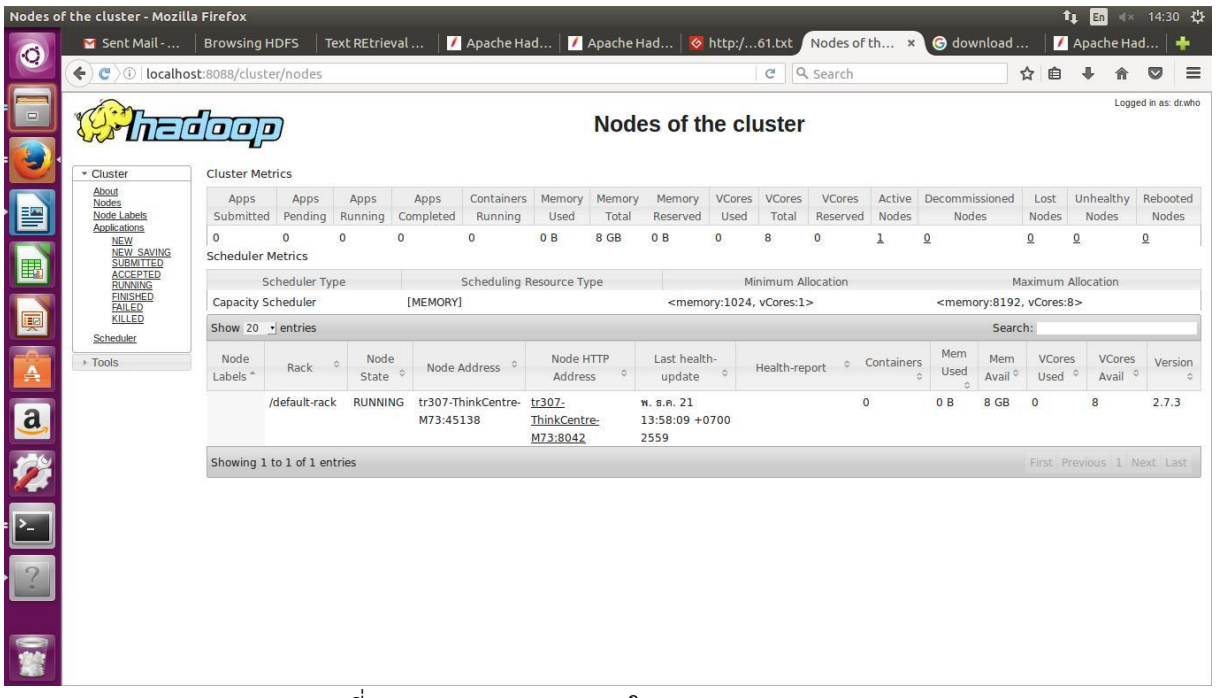
คำสั่ง jps ตรวจสอบสภาวะแวดล้อมของ Hadoop ดังภาพที่ 9

```
  Jps
```



ภาพที่ 9 คำสั่ง jps ตรวจสอบสภาวะแวดล้อมของ Hadoop

จะได้ผลลัพธ์การทำงานของ Hadoop Distributed File System ดังภาพที่ 10



ภาพที่ 10 การ run Hadoop ใน Single Node Cluster

จากภาพที่ 10 ทดสอบการ run Hadoop ที่ localhost:8088/cluster/nodes โดยแสดงสถานะ Node State เป็น Running และ Version เป็น 2.7.3

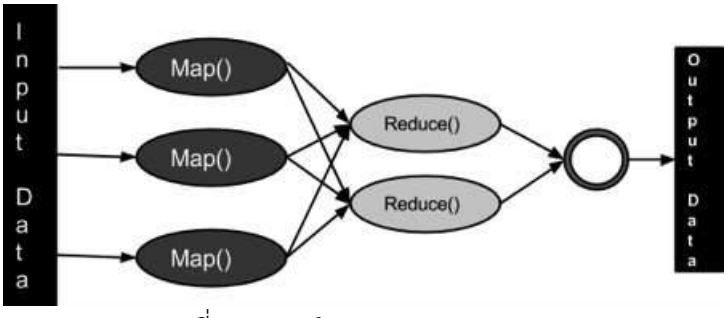
วันที่ 2 (21 ธันวาคม 2559)

1. หลักการทำงานของ MapReduce

แมพ์รีดิวซ์ (MapReduce) เป็นเทคนิคการประมวลผลและรูปแบบโปรแกรมสำหรับการคำนวณแบบกระจายบนพื้นฐาน Java โดยอัลกอริทึม MapReduce มีสองงาน ที่สำคัญคือแมพ์ และรีดิวซ์

โดยประการที่หนึ่งแมพ์หรือแผนที่จะนำชุดของข้อมูลและแปลงเป็นอีกชุดหนึ่งของข้อมูลที่แต่ละองค์ประกอบจะแบ่งออกเป็น tuples (คีย์ / คู่ค่า) และประการที่สองรีดิวซ์หรือลดงานซึ่งจะมีการส่งออกจากแผนที่เป็น input ที่และรวมผู้ tuples ข้อมูลลงในชุดเล็กของ tuples เป็นลำดับของชื่อ MapReduce หมายถึงการลดงานที่จะดำเนินการเสมอหลังจากงานแผนที่

ประโยชน์ที่สำคัญของ MapReduce คือช่วยในการปรับขนาดการประมวลผลข้อมูลผ่านคอมพิวเตอร์หลายโหนด ภายใต้รูปแบบ MapReduce ที่วิทยาการการประมวลผลข้อมูลที่เรียกว่าแมพ์รีดิวซ์สามารถประยุกต์ใช้ในการทำงานมากกว่าหลายร้อยหลายพันหรือแม้กระทั่งนับหมื่นของเครื่องในคลัสเตอร์เป็นเพียงการเปลี่ยนการตั้งค่านี้ และสามารถปรับขยายระบบได้ง่าย (scalability) จึงเป็นสิ่งที่ดึงดูดโปรแกรมเมอร์จำนวนมากที่จะใช้รูปแบบ MapReduce ดังภาพที่ 11



ภาพที่ 11 การทำงานของ MapReduce

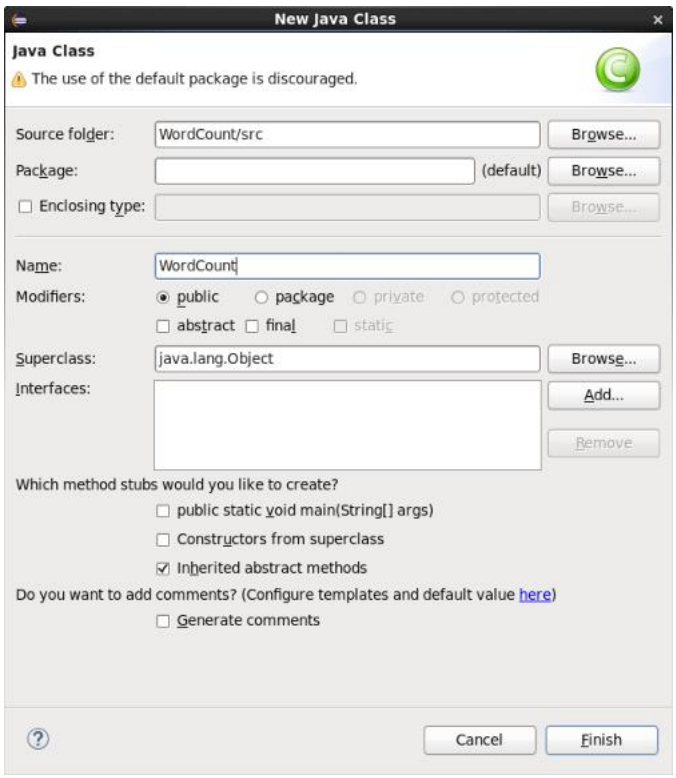
ที่มา : http://www.w3ii.com/th/hadoop/hadoop_mapreduce.html

จากภาพที่ 11 โดยทั่วไปกระบวนทัศน์ MapReduce จะขึ้นอยู่กับการส่งคอมพิวเตอร์เพื่อที่ข้อมูลที่อยู่โดยโปรแกรม MapReduce ดำเนินการในสามขั้นตอนคือ

1. ขั้นตอน Map คือการประมวลผลข้อมูลของท่าน โดยทั่วไปการป้อนข้อมูลที่อยู่ในรูปแบบของไฟล์หรือไดเรกทอรีหรือแฟ้ม และถูกเก็บไว้ในระบบไฟล์ Hadoop (HDFS) แฟ้มใส่จะถูกส่งผ่านไปยังบรรทัดฟังก์ชั่น Mapper โดยสาย แมปเปอร์ประมวลผลข้อมูลและสร้างชิ้นเล็ก ๆ หลายของข้อมูล
2. ขั้นตอน Shuffle (สับเปลี่ยน) คือการจัดเรียงข้อมูลจาก Maper
3. ขั้นตอน Reduce คือการรวมกันของเวทีการสับเปลี่ยนและลดขั้นตอน งานของ Reducer คือการประมวลผลข้อมูลที่มาจาก Mapper หลังจากการประมวลผลจะผลิตชุดใหม่ของการส่งออกซึ่งจะถูกเก็บไว้ใน HDFS

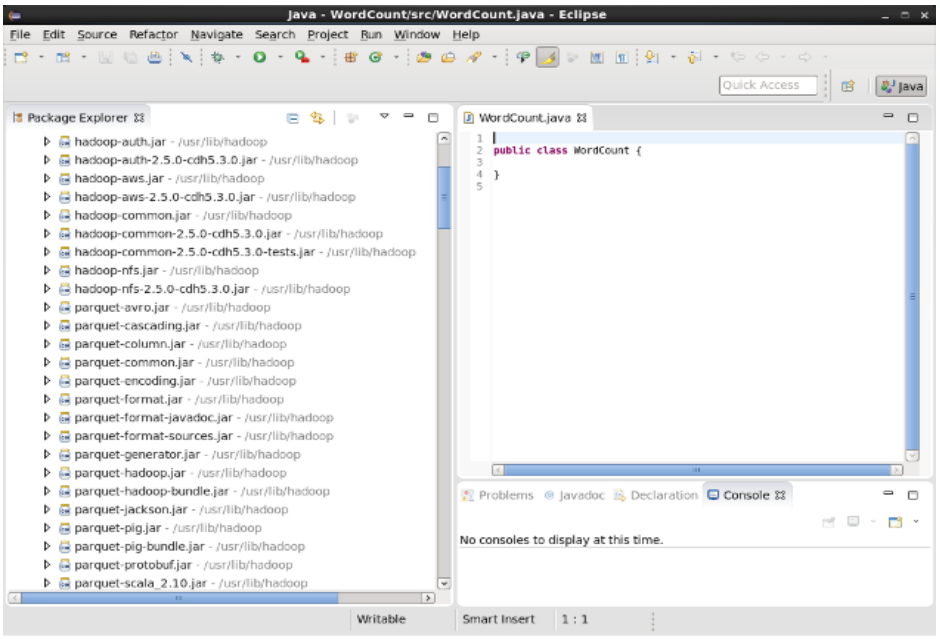
การเขียนโปรแกรม WordCount ในการทำงานของ MapReduce

ขั้นตอนที่ 1 เปิด Eclipse Neon ตั้งชื่อ WordCount



ภาพที่ 12 เริ่มต้นโปรแกรม WordCount

ขั้นตอนที่ 2 ทำการ import คลาสที่เกี่ยวข้องเช่น Hadoop



ภาพที่ 13 กำหนดคลาสที่เกี่ยวข้อง

ขั้นตอนที่ 3 เขียนคำสั่ง WordCount

1. StartsWithCountJob.java

```
package wordcount;


import org.apache.hadoop.conf.Configured;
```

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class StartsWithCountJob extends Configured implements Tool {

  @Override
  public int run(String[] args) throws Exception {
            // TODO Auto-generated method stub
            Job job = Job.getInstance(getConf(), "StartsWithCount");
                        job.setJarByClass(getClass());

TextInputFormat.addInputPath(job, new Path(args[0]));
job.setInputFormatClass(TextInputFormat.class);

TextOutputFormat.setOutputPath(job, new Path(args[1]));

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);

job.setMapperClass(StartsWithCountMapper.class);
job.setReducerClass(StartsWithCountReducer.class);
job.setCombinerClass(StartsWithCountReducer.class);

return job.waitForCompletion(true) ? 0:1;

  }

  public static void main(String[] args) throws Exception {
            // TODO Auto-generated method stub
```

```
int exitCode = ToolRunner.run(new StartsWithCountJob(), args);
System.exit(exitCode);
  }

}
```

2. StartsWithCountMapper.java

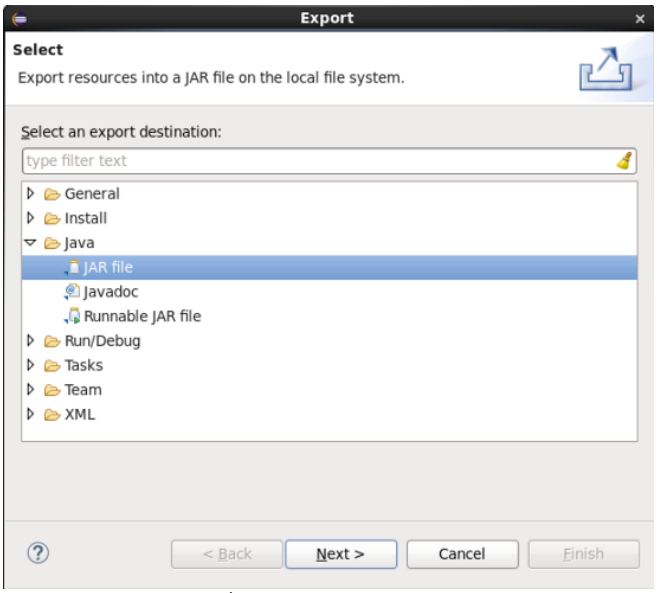
```
package wordcount;

import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class StartsWithCountMapper extends Mapper<LongWritable, Text, Text,
IntWritable> {
  private static Text text = new Text();
  private static IntWritable One = new IntWritable(1);
  protected void map(LongWritable key, Text value, Context context)
                        throws IOException, InterruptedException {
            // TODO Auto-generated method stub
            StringTokenizer tokenizer = new StringTokenizer(value.toString());
        while (tokenizer.hasMoreElements()) {
         text.set(tokenizer.nextToken());
         context.write(text, One);
        }
  }
}
```

3. StartsWithCountReducer.java

```
package wordcount;

import java.io.IOException;
```

```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class StartsWithCountReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

  @Override
  protected void reduce(Text token, Iterable<IntWritable> counts,
                        Context context) throws IOException, InterruptedException {
            // TODO Auto-generated method stub
            int sum = 0;
        for (IntWritable count : counts) {
          sum += count.get();
        }
                  context.write(token, new IntWritable(sum));
  }
}
```

ขั้นตอนที่ 4 สร้าง Jar ไฟล์

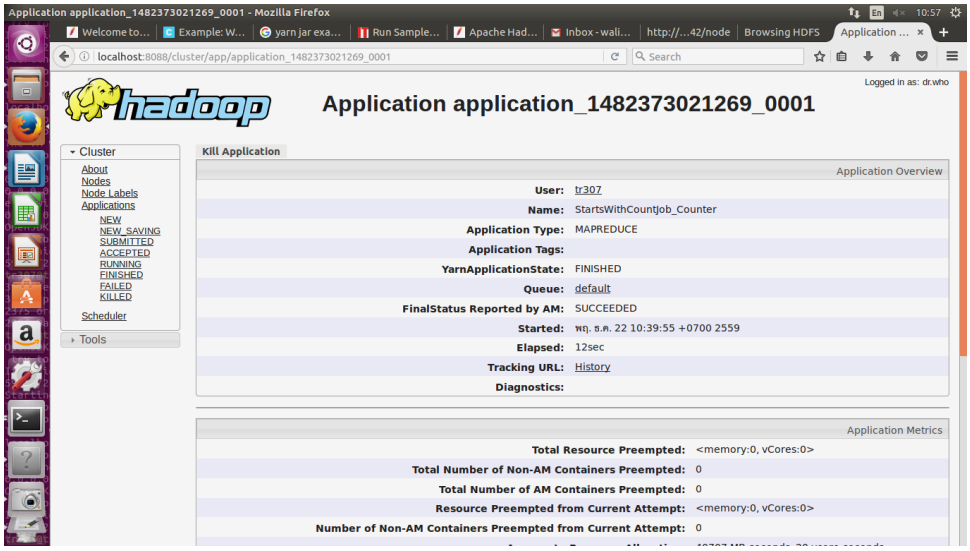


ภาพที่ 14 การสร้าง JAR ไฟล์

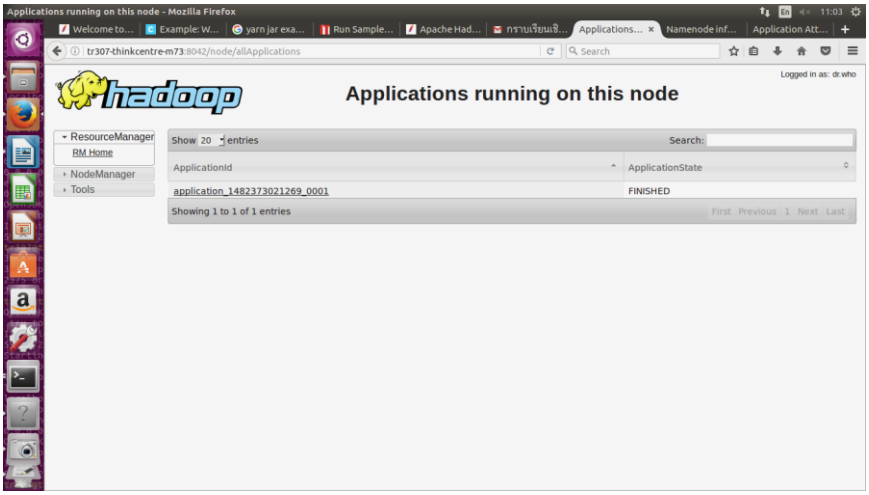
ขั้นตอนที่ 5 คำสั่ง RUN

```
hadoop jar /home/WordCount.jar WordCount input/wordcount.txt output
```

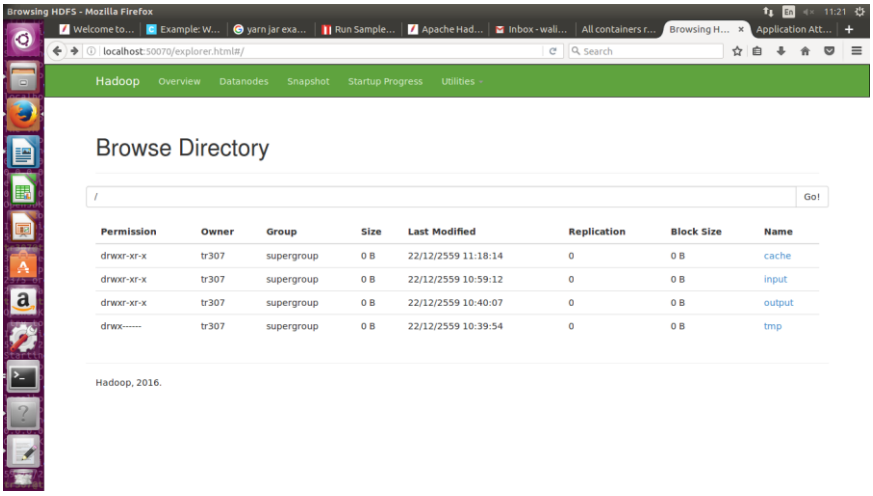
ขั้นตอนที่ 6 การแสดงผลที่ Browser (http://localhost:8088)



ก



ข



ค

ภาพที่ 15 ก-ค การแสดงผลที่ Browser

<u>วันที่ 3 (22 ธันวาคม 2559)</u>

2. การเขียนโปรแกรม Histogram

```
package weblog;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.GregorianCalendar;
import java.util.Iterator;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;
import org.apache.http.ParseException;

import weblog.FreqDistMapReduce.FreqDistMapper;
import weblog.FreqDistMapReduce.FreqDistReducer;

public class HistrogramGenMapReduce extends Configured implements Tool {
        public int run(String[] args) throws Exception {
                if (args.length != 2) {
                        System.err.println("Usage: <input_path> <output_path>");
                        System.exit(-1);
                }
                String inputPath = args[0];
                String outputPath = args[1];
```

```
                Job job = Job.getInstance(getConf(), "HistrogramGenMapReduce");
                job.setJarByClass(getClass());
                job.setMapperClass(HistGenMapper.class);
                job.setReducerClass(HistGenReducer.class);
                job.setNumReduceTasks(1);
                job.setMapOutputKeyClass(LongWritable.class);
                job.setMapOutputValueClass(LongWritable.class);
                // job.setOutputKeyClass(Text.class);
                // job.setOutputValueClass(LongWritable.class);
                FileInputFormat.setInputPaths(job, new Path(inputPath));
                FileOutputFormat.setOutputPath(job, new Path(outputPath));

                return job.waitForCompletion(true) ? 0 : 1;
        }
public static void main(String[] args) throws Exception {
int exitCode = ToolRunner.run(new Configuration(), new HistrogramGenMapReduce(), args);
                System.exit(exitCode);
        }

public static class HistGenMapper extends Mapper<Object, Text, LongWritable,
LongWritable> {

public static final Pattern httplogPattern = Pattern
.compile("([^\\s]+) - - \\[(.+)\\] \"[^\\s]+ (/[^\\s]*) HTTP/[^\\s]+\" ([^\\s]+) ([0-9]+)");
                //count time and key (set point of time) on group 2

                private final static LongWritable one = new LongWritable(1);
                private final static SimpleDateFormat dateFormatter = new
SimpleDateFormat("dd/MMMMM/yyyy:hh:mm:ss /"); //pattern of date format (day month
year hr mm ss and timezone

protected void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
Matcher matcher = httplogPattern.matcher(value.toString());
if (matcher.matches()) {
String timeStr = matcher.group(2); // check point of time on group 2
try {
```

```
Date time = dateFormatter.parse(timeStr);
Calendar calendar = GregorianCalendar.getInstance();
calendar.setTime(time);
int hour = calendar.get(Calendar.HOUR_OF_DAY);
context.write(new LongWritable(hour), one);
} catch (java.text.ParseException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}
}
}

public static class HistGenReducer extends Reducer<LongWritable, LongWritable,
LongWritable, LongWritable> {
@Override
protected void reduce(LongWritable key, Iterable<LongWritable> values, Context context)
                                        throws IOException, InterruptedException {
                                Long sum = 0;
                                for (LongWritable value : values) {
                                        sum += value.get();
                                }
context.write(key, new LongWritable(sum));
                                }
                        }
                }
}
```

```
tr307@tr307-ThinkCentre-M73:~/workspace/weblog$ yarn jar test-hist.jar
weblog.HistrogramGenMapReduce /input/log1 /outputHist55
OpenJDK 64-Bit Server VM warning: You have loaded library
/usr/local/hadoop/lib/native/libhadoop.so which might have disabled stack guard. The
VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with
'-z noexecstack'.
```

59/12/23 10:30:59 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

59/12/23 10:30:59 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032

59/12/23 10:31:00 INFO input.FileInputFormat: Total input paths to process : 1

59/12/23 10:31:00 INFO mapreduce.JobSubmitter: number of splits:2

59/12/23 10:31:00 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1482398609860_0007

59/12/23 10:31:01 INFO impl.YarnClientImpl: Submitted application application_1482398609860_0007

59/12/23 10:31:01 INFO mapreduce.Job: The url to track the job: http://tr307-ThinkCentre-M73:8088/proxy/application_1482398609860_0007/

59/12/23 10:31:01 INFO mapreduce.Job: Running job: job_1482398609860_0007

59/12/23 10:31:06 INFO mapreduce.Job: Job job_1482398609860_0007 running in uber mode : false

59/12/23 10:31:06 INFO mapreduce.Job: map 0% reduce 0%

59/12/23 10:31:16 INFO mapreduce.Job: map 40% reduce 0%

59/12/23 10:31:18 INFO mapreduce.Job: map 64% reduce 0%

59/12/23 10:31:19 INFO mapreduce.Job: map 74% reduce 0%

59/12/23 10:31:22 INFO mapreduce.Job: map 81% reduce 0%

59/12/23 10:31:23 INFO mapreduce.Job: map 100% reduce 0%

59/12/23 10:31:25 INFO mapreduce.Job: map 100% reduce 100%

59/12/23 10:31:27 INFO mapreduce.Job: Job job_1482398609860_0007 completed successfully

59/12/23 10:31:27 INFO mapreduce.Job: Counters: 50

File System Counters

FILE: Number of bytes read=33610920

FILE: Number of bytes written=67577879

FILE: Number of read operations=0

FILE: Number of large read operations=0

FILE: Number of write operations=0

HDFS: Number of bytes read=205246658

HDFS: Number of bytes written=213

HDFS: Number of read operations=9

HDFS: Number of large read operations=0

HDFS: Number of write operations=2

Job Counters

Killed map tasks=1

Launched map tasks=3

Launched reduce tasks=1

Data-local map tasks=3

Total time spent by all maps in occupied slots (ms)=28146

Total time spent by all reduces in occupied slots (ms)=4845

Total time spent by all map tasks (ms)=28146

Total time spent by all reduce tasks (ms)=4845

Total vcore-milliseconds taken by all map tasks=28146

Total vcore-milliseconds taken by all reduce tasks=4845

Total megabyte-milliseconds taken by all map tasks=28821504

Total megabyte-milliseconds taken by all reduce tasks=4961280

Map-Reduce Framework

Map input records=1891715

Map output records=1867273

Map output bytes=29876368

Map output materialized bytes=33610926

Input split bytes=194

Combine input records=0

Combine output records=0

Reduce input groups=24

Reduce shuffle bytes=33610926

Reduce input records=1867273

Reduce output records=24

Spilled Records=3734546

Shuffled Maps =2

Failed Shuffles=0

Merged Map outputs=2

GC time elapsed (ms)=1081

CPU time spent (ms)=29520

Physical memory (bytes) snapshot=735342592

Virtual memory (bytes) snapshot=5853548544

Total committed heap usage (bytes)=526909440

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0

File Input Format Counters

Bytes Read=205246464

File Output Format Counters

Bytes Written=213

tr307@tr307-ThinkCentre-M73:~/workspace/weblog$ hdfs dfs -cat /outputHist55/*

OpenJDK 64-Bit Server VM warning: You have loaded library /usr/local/hadoop/lib/native/libhadoop.so which might have disabled stack guard. The VM will try to fix the stack guard now.

It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.

59/12/23 10:34:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

0 119484

1 120813

2 119542

3 116474

4 96297

5 78201

6 70876

7 68930

8 70896

9 69716

10 68388

11 61364

12 52228

13 44734

14 36861

15 31839

16 31492

17 34851

18 53428

19 82904

20 98824

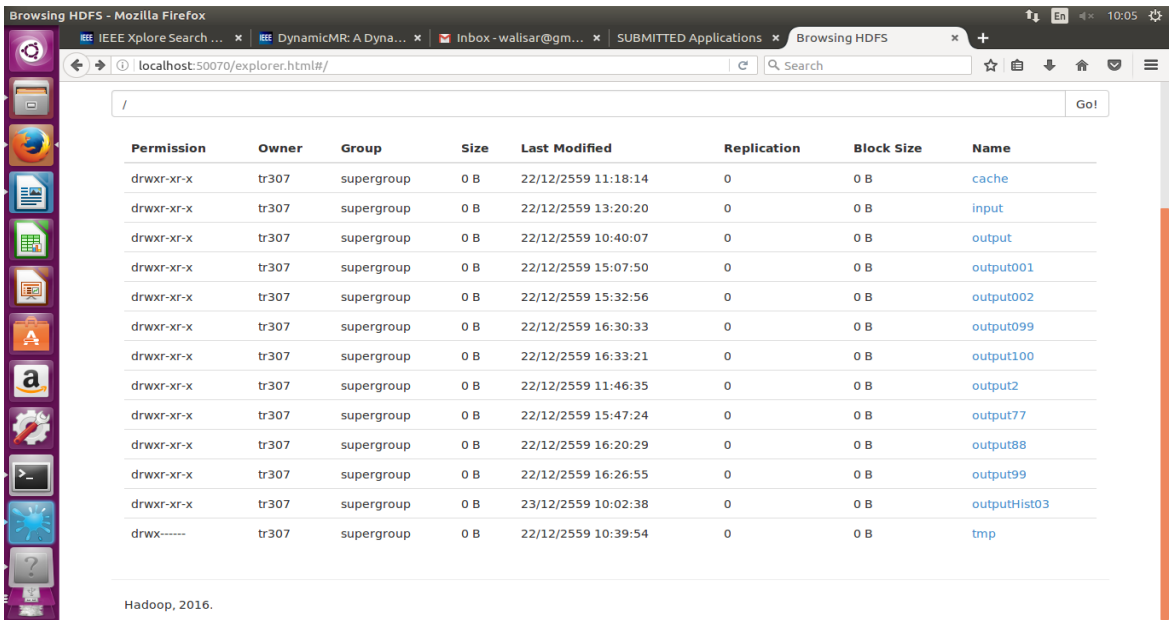
21 104291

22 114177

23 120663

tr307@tr307-ThinkCentre-M73:~/workspace/weblog$ ^C

tr307@tr307-ThinkCentre-M73:~/workspace/weblog$



ภาพที่ 16 การ run ของโปรแกรม Histogram

<u>วันที่ 4 (23 ธันวาคม 2559)</u>

3. เครื่องมืองของ Apache Hadoop

1. Hive เป็นเครื่องมือสำหรับผู้ต้องการสืบค้น (Query) ข้อมูลที่เก็บใน HDFS ด้วยภาษาลักษณะ SQL แทนที่จะต้องมาเขียนโปรแกรม Map/Reduce โดย Hive จะทำหน้าที่ในการแปล SQL like ให้มาเป็น Map/Reduce แล้วก็ทำการรันแบบ Batch
2. Pig เป็นเครื่องมือคล้ายๆกับ Hive ที่ช่วยให้ประมวลผลข้อมูลโดยไม่ต้องเขียนโปรแกรม Map/Reduce ซึ่ง Pig จะใช้โปรแกรมภาษา script ง่ายๆที่เรียกว่า Pig Latin แทน โดย Pigเหมาะกับการทำ ETL สำหรับการแปลงข้อมูลในรูปแบบต่างๆเช่น JSON
3. Sqoop เป็นเครื่องมือในการถ่ายโอนข้อมูลระหว่างฐานข้อมูลที่อยู่รูปแบบ Table บน RDBMS อย่าง SQL server, Oracle หรือ MySQL กับข้อมูลบน HDFS ของ Hadoop
4. Flume เป็นเครื่องมือในการดึงข้อมูลจากระบบอื่นๆแบบ Realtime เข้าสู่ HDFS เช่น การดึง Log จาก Web Server การดึงข้อมูลเหล่านี้จะต้องมีการติดตั้ง Agent ที่เครื่อง Server
5. HBase เป็นเครื่องมือที่จะทำให้ Hadoop สามารถอ่านและเขียนข้อมูลแบบ Realtime Random Access ได้โดยจะทำให้เป็น BigTable ที่เก็บข้อมูลได้ไม่จำกัด row หรือ column ซึ่ง HBase ก็จะเป็นเสมือนการทำให้ Hadoop เป็น NoSQL Database

6. Oozie เป็นเครื่องมือในการทำ Workflow จะช่วยให้เราเอาคำสั่งประมวลผลต่างๆของระบบ Hadoop เช่น Map/Reduce, Hive หรือ Pig มาเชื่อมต่อกันในรูปของ Workflow ได้
7. Hue ย่อมาจากคำว่า Hadoop User Experience เป็นเครื่องมือช่วยทำ User interface ของ Hadoop ให้ใช้งานได้ง่ายขึ้นกว่าการต้องใช้ command line
8. Mahout เป็นเครื่องมือของ Data Scientist ที่ต้องการทำPredictive Analytics ข้อมูลบน Hadoop โดยใช้ภาษาจาวา ทั้งนี้ Mahout สามารถใช้ Algorithm ที่เป็น Recommender, Classification และ Clustering ได้

4. การสร้างตารางด้วย Apache HIVE

คำสั่งสร้างตาราง (Create Table)

รูปแบบคำสั่ง

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.] table_name
[(col_name data_type [COMMENT col_comment], ...)]
[COMMENT table_comment]
[ROW FORMAT row_format]
[STORED AS file_format]
```

การเขียนคำสั่งที่ Apache HIVE

```
COMMENT ‘Employee details’
FIELDS TERMINATED BY ‘\t’
LINES TERMINATED BY ‘\n’
STORED IN TEXT FILE
```

```
hive> CREATE TABLE IF NOT EXISTS employee ( eid int, name String,
salary String, destination String)
COMMENT ‘Employee details’
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘\t’
LINES TERMINATED BY ‘\n’
STORED AS TEXTFILE;
```

2.5 ประโยชน์ที่ได้รับ

1) ประโยชน์ที่ได้รับจากการอบรม

- สามารถพัฒนา Big Data Application โดยใช้เครื่องมือ กระบวนการพัฒนาและเทคโนโลยีสมัยใหม่คือ Hadoop, HBase, Hive, Pig และ Oozie ที่นำไปใช้ทำปัญญาวิเคราะห์ (Analytic) หรือใช้งานร่วมกับ Business Intelligence เพื่อใช้เป็นเครื่องมือในการบริหารจัดการในด้านต่างๆ เช่น แนวทางการลด ต้นทุนและหรือจัดอันดับสินค้าและบริการในเชิงของผลตอบแทนได้จริง
- สามารถนำไประยุกต์ใช้งานในลักษณะ Public Cloud ที่มีให้บริการในท้องตลาด หรือ สามารถใช้งานภายในแบบ Private Cloud ได้ทันทีโดยไม่ต้องมีการเปลี่ยนแปลง
- มีทักษะและประสบการณ์ในการพัฒนาโปรแกรมอย่างถูกต้องตามหลักปฏิบัติระดับดีเลิศ (Best practices) อันเป็นที่ยอมรับทั่วโลก

2) ประโยชน์ที่มหาวิทยาลัยสุโขทัยธรรมาธิราช จะได้รับ

- สามารถนำความรู้ไปประยุกต์ใช้ในการทำวิจัยที่ได้รับทุนวิจัย 2560 จากมหาวิทยาลัยสุโขทัยธรรมาธิราช
- สามารถนำความรู้ไปประยุกต์ใช้ในการทำวิจัยร่วมกับนักศึกษาระดับปริญญาโท ในฐานะอาจารย์ที่ปรึกษาวิทยานิพนธ์และค้นคว้าอิสระ
- จัดอบรมการเขียนโปรแกรมข้อมูลใหญ่ (Big Data Programming) ให้แก่คณาจารย์ในสาขาต่างๆ เจ้าหน้าที่ของสำนักคอมพิวเตอร์ หน่วยงานที่เกี่ยวข้อง เพื่อเป็นการถ่ายทอดความรู้ให้แก่บุคคลากรมหาวิทยาลัยสุโขทัยธรรมาธิราช และประชาสัมพันธ์มหาวิทยาลัยสุโขทัยธรรมาธิราช

.....