Design & Implementation Plan Document

Simplified vCenter Data with Graph Database

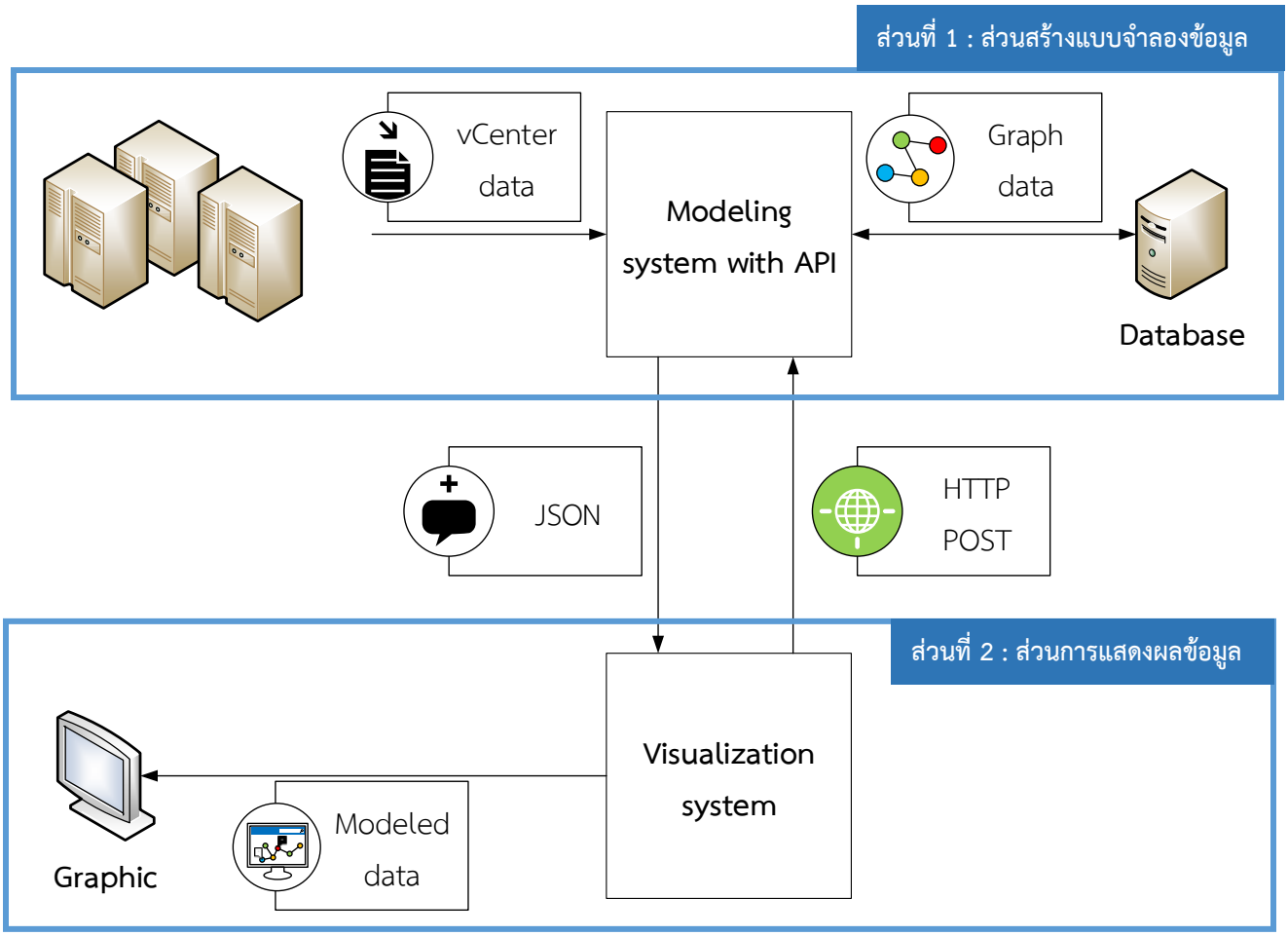
วิชา : Cloud Computing Fall 2014 @ KMITL

กลุ่ม : B_MW

สมาชิก : นายนพกร ใช้บุญเรือง 54010656

นางสาวนนทยา วิไลเลิศสมบัติ 54010654

ภาพรวมของระบบ



ระบบที่นำเสนอสามารถแบ่งการทำงานออกเป็นสองส่วนหลักๆ คือ ส่วนสร้างแบบจำลองข้อมูล และ ส่วนแสดงผลข้อมูลต่อผู้ใช้งาน

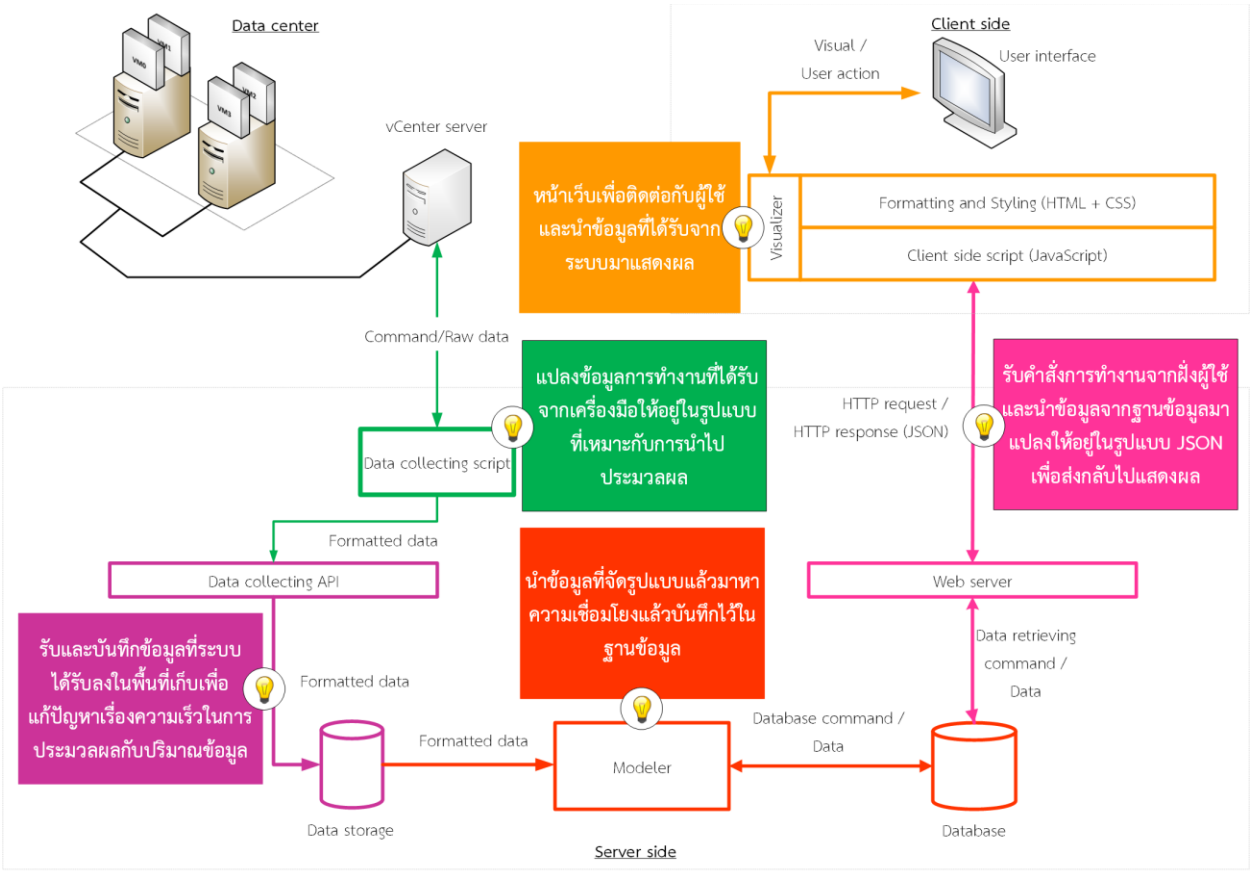
1) ส่วนสร้างแบบจำลองข้อมูล

ข้อมูลที่มาจาก vCenter จะถูกดึงเข้ามายังระบบโดยใช้ vSphere Web Service SDK เพื่อติดต่อกับ vCenter และถูกนำไปประมวลผลเพื่อปรับปรุงแบบจำลองข้อมูลซึ่งเก็บไว้ใน Graph Database นอกจากนี้จะมีส่วนที่ใช้ในการติดต่อกับส่วนแสดงผลข้อมูลเพื่อให้สามารถนำแบบจำลองดังกล่าวไปแสดงผลต่อผู้ใช้

2) ส่วนการแสดงผลข้อมูลให้กับผู้ใช้งาน

การแสดงผลจะแสดงผลผ่านเว็บโดยในโปรเจคนี้จะเน้นการแสดงผลเพื่อนำเสนอข้อมูลให้สามารถเข้าใจได้ง่ายขึ้นเป็นหลัก โดยไม่เน้นในเรื่องความสวยงามหรือฟังก์ชันอื่นๆในการแสดงผล

โครงสร้างภายในของระบบ



1. Data collecting script, Data collecting API และ Data Storage

เป็นส่วนที่ทำหน้าที่รับและบันทึกข้อมูลที่ส่งมาจาก vCenter โดยมีจุดประสงค์ให้เป็นที่พักข้อมูลชั่วคราว (buffer) ก่อนที่ระบบจะนำไปประมวลผล ส่วนนี้จะช่วยป้องกันข้อมูลสูญหายหรือผลกระทบต่อระบบในกรณีที่มีข้อมูลเข้ามาเป็นจำนวนมาก

Data collecting script เป็นสคริปต์ที่ทำหน้าที่ดึงข้อมูลจาก vCenter โดยใช้งาน vSphere Web Service SDK โดยดึงข้อมูลต่างๆของเครื่องในศูนย์ข้อมูล

Data collecting API ออกแบบให้ใช้งาน HTTP ที่กำหนดในรูปแบบของ RESTful API โดยจะสามารถใช้รายละเอียดใน URL ในการคัดแยกข้อมูลที่ส่งมา เพื่อให้ง่ายแก่การนำไปประมวลผลต่อไป

Data Storage เป็นส่วนที่ใช้เก็บข้อมูลดิบที่ผ่านการคัดแยกจาก Data Collecting API ส่วนนี้ถูกออกแบบในรูปแบบของไฟล์

2. Modeler และ Graph Database

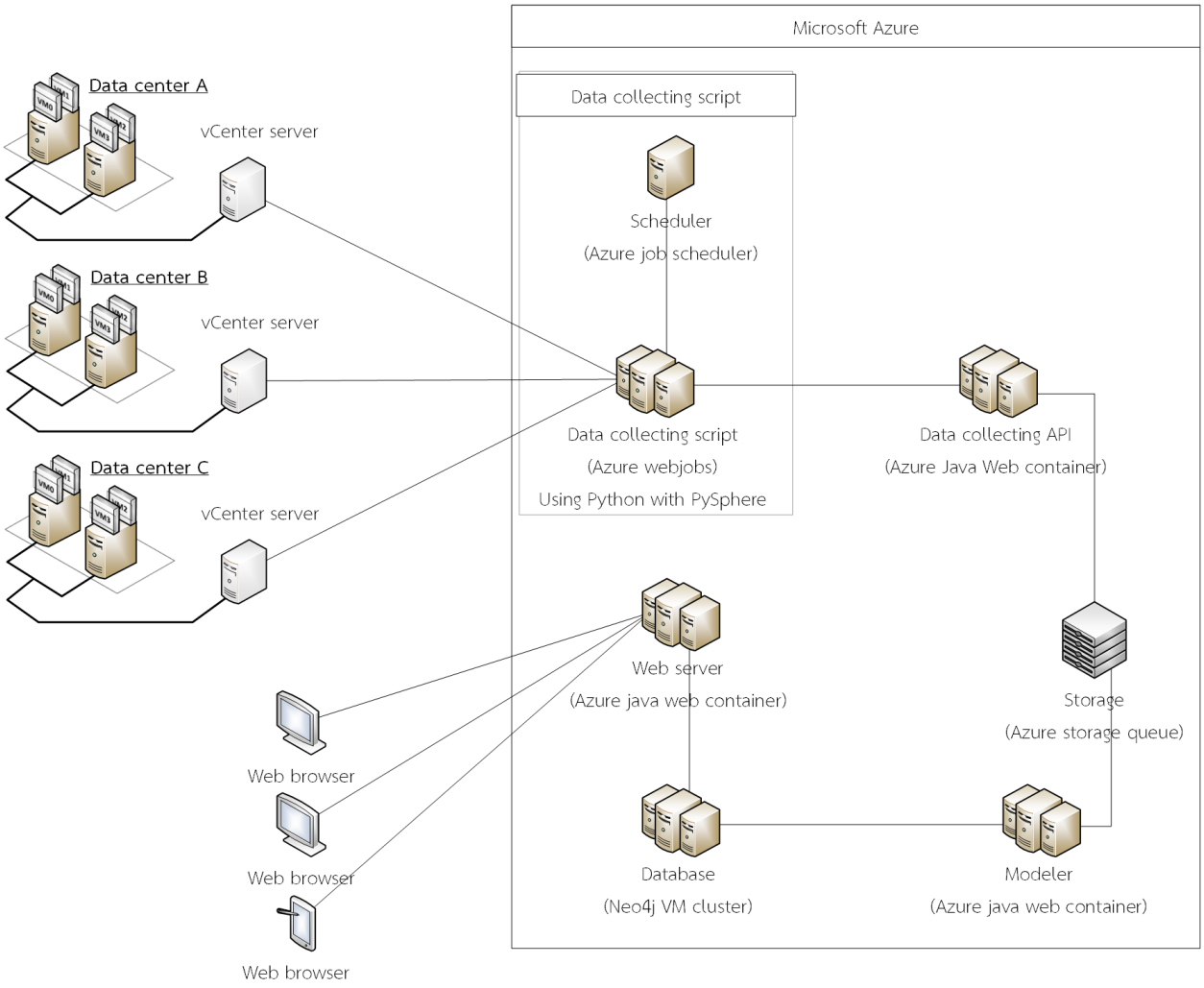
ทำหน้าที่นำข้อมูลออกจาก Data Storage เพื่อที่จะประมวลผลและแปลงให้อยู่ในโครงสร้างที่กำหนดไว้ เพื่อเหมาะแก่การนำมาแสดงผลต่อผู้ใช้โดยเก็บข้อมูลเหล่านี้ไว้ใน Graph Database โดยในส่วนของ Modeler จะเลือกพัฒนาโดยใช้ Java และใช้ neo4j เป็น Graph Database (โครงสร้างของข้อมูลอยู่ในขั้นตอนการออกแบบ)

3. Visualizer

Visualizer จะติดต่อกับ Database ผ่านทาง Web server ซึ่งใช้งาน neo4j REST API โดย Visualizer เป็นส่วนที่ทำหน้าที่แปลงข้อมูลจากระบบซึ่งอยู่ในรูปแบบของ JSON ให้กลายเป็นภาพและส่วนติดต่อกับผู้ใช้โดยอยู่ในรูป Web interface ซึ่งใช้งาน HTML5 ร่วมกับ JavaScript และ CSS

เพื่อให้ข้อมูลถูกปรับปรุงตลอดเวลาจึงออกแบบให้ทำงานแบบ asynchronous โดยทำการร้องขอเฉพาะข้อมูลที่เปลี่ยนแปลงจาก server เป็นระยะๆ เพื่อลดปริมาณข้อมูลที่ส่งผ่านเครือข่าย

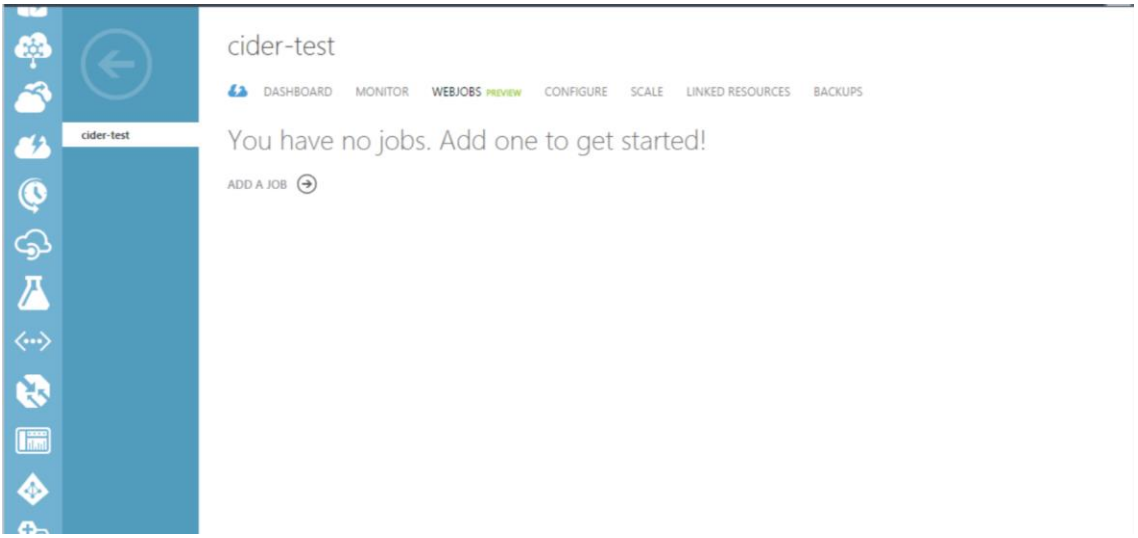
การพัฒนาระบบบน Microsoft Azure



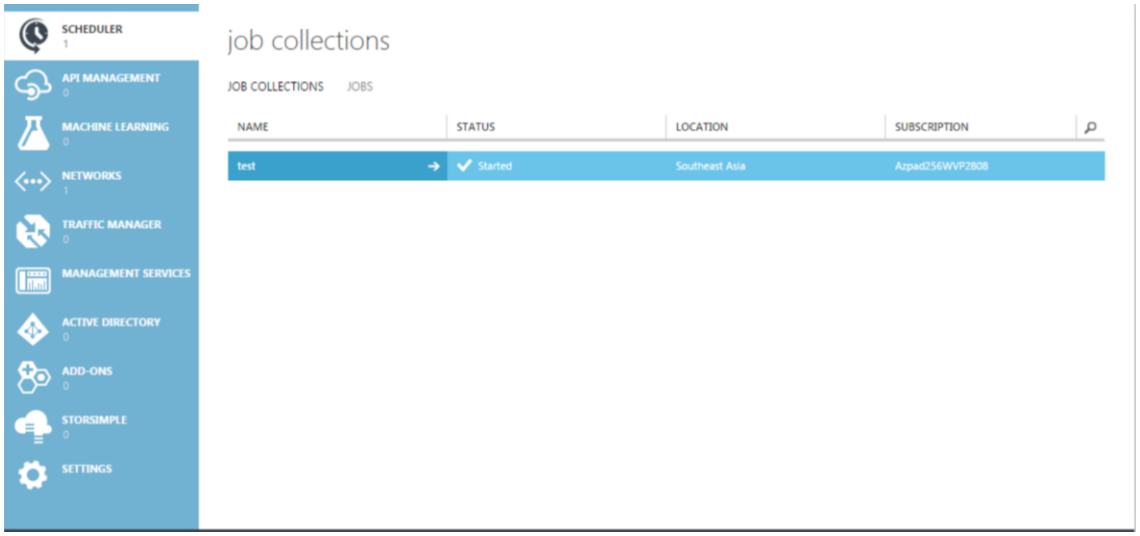
Data collecting script

แบ่งส่วนการทำงานออกเป็น 2 ส่วนย่อยๆดังนี้

1. Data collecting script
    a. หน้าที่: ดึงข้อมูลจาก vCenter เพื่อนำข้อมูลเข้าสู่ระบบ
    b. ภาษาที่ใช้เขียน: Python โดยใช้งาน PySphere ซึ่งเป็น Python API ที่ติดต่อกับ vSphere Web Service SDK
    c. Cloud service ที่ใช้งาน: ฝังคำสั่งที่ต้องการใช้ในการดึงข้อมูลไว้บน Azure webjob ซึ่งเป็นการทำงานรูปแบบหนึ่งบน Azure website



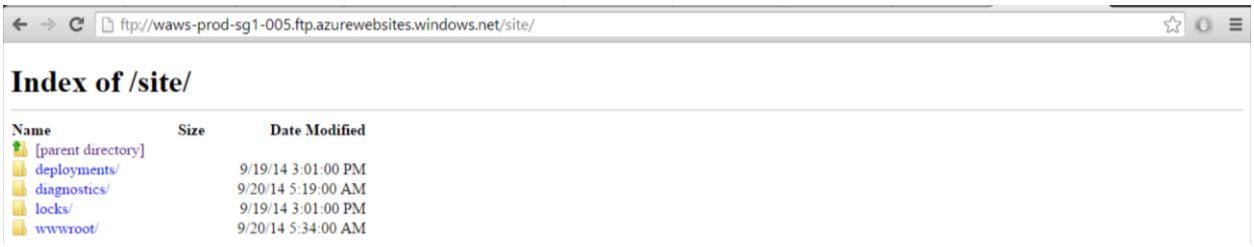
2. Scheduler
    a. หน้าที่: ควบคุมให้ Data collecting script ทำงานเมื่อถึงเวลาที่กำหนด
    b. Cloud service ที่ใช้งาน: ใช้งาน Scheduler ของ Azure โดยสามารถตั้งค่าช่วงเวลาในการทำงานได้

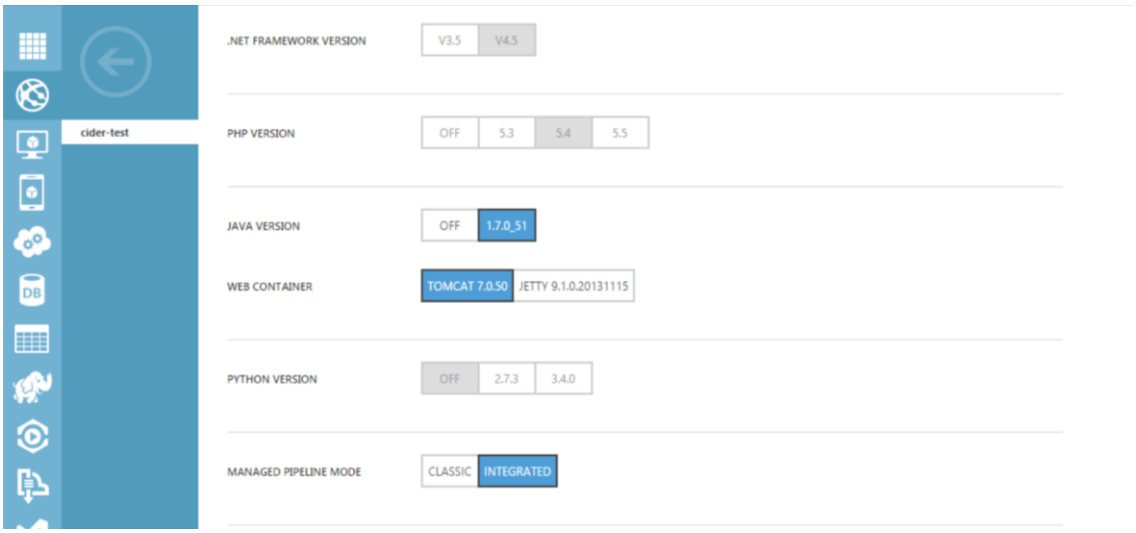


Data collecting API

a. หน้าที่: เป็น API สำหรับรับข้อมูลต่างๆจาก Data collecting script โดยจัดข้อมูลให้อยู่ในรูปแบบที่นำประมวลผลต่อได้ง่าย
b. ภาษาที่ใช้เขียน: Java โดยใช้งาน JAX-RS (The Java API for RESTful Web Services) เขียนให้รับข้อมูลในรูปแบบของ REST API และ Jersey เพื่อสร้าง Dispatcher ซึ่งจะทำให้คำสั่งที่เขียนสามารถทำงานบน Web container ได้

c. Cloud service ที่ใช้งาน: Azure Java web container โดยใช้ FTP เพื่อนำเว็บไซต์ขึ้นไปยัง Azure website โดยใช้ Apache Tomcat เป็น web container

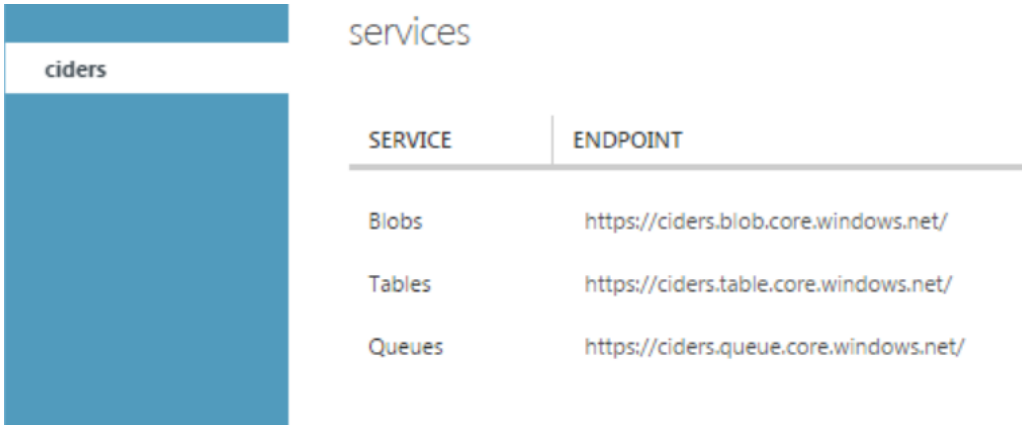




Data Storage

a. หน้าที่: เป็นที่พักข้อมูลชั่วคราว (buffer) ก่อนที่ระบบจะนำไปประมวลผล

b. Cloud service ที่ใช้งาน: Azure storage โดยน่าจะเลือกใช้การทำงานแบบ queue storage ผ่านทาง queue service rest API ของ Azure



Modeler

a. หน้าที่: ประมวลผลและแปลงข้อมูลใน Data storage ให้อยู่ในโครงสร้างที่กำหนดไว้ เพื่อเหมาะแก่การนำมาแสดงผลต่อผู้ใช้โดยเก็บข้อมูลเหล่านี้ไว้ใน Graph Database

b. ภาษาที่ใช้เขียน: Java

c. Cloud service ที่ใช้งาน: Azure Java web container โดยใช้ FTP เพื่อนำเว็บไซต์ขึ้นไปยัง Azure website โดยใช้ Apache Tomcat เป็น web container

Database (Neo4j)

a. หน้าที่: เก็บข้อมูลที่ผ่านการประมวลผลของ modeler

b. Cloud service ที่ใช้งาน: Azure virtual machine โดยสร้าง virtual machine ไว้จำนวนหนึ่งและกำหนดเป็นการทำงานแบบ cluster โดยอาศัยการตั้งค่าผ่าน Neo4j และใช้งาน HAProxy (Neo4j สามารถเพิ่มประสิทธิภาพของการอ่านข้อมูลได้ในรูปแบบของ scale-out แต่การเขียนข้อมูลสามารถทำได้โดย scale-up เท่านั้น ดังนั้นการทำ cluster ของ Neo4j ร่วมกับการใช้ HAProxy ส่งคำสั่งที่ต้องมีการเขียนข้อมูลไปยัง master และส่งคำสั่งที่เป็นเพียงการอ่านข้อมูลไปยัง slave ใน cluster จะช่วยให้เครื่องที่เป็น master ของ cluster นั้นๆสามารถเขียนข้อมูลได้เต็มประสิทธิภาพมากขึ้น เนื่องจากโอนถ่ายงานที่เกี่ยวข้องกับการอ่านข้อมูลไปยัง slave เพียงอย่างเดียว)

Web server

a. หน้าที่: เก็บหน้าเว็บของ visualizer เพื่อเป็นส่วนติดต่อกับผู้ใช้ โดย Web server จะเป็นส่วนที่คั่นระหว่างผู้ใช้ (และ Visualizer) กับ Database เพื่อแปลง request จากฝั่งผู้ใช้เป็นการติดต่อไปยัง Neo4j REST API

b. ภาษาที่ใช้เขียน: Java

c. Cloud service ที่ใช้งาน: Azure Java web container โดยใช้ FTP เพื่อนำเว็บไซต์ขึ้นไปยัง Azure website โดยใช้ Apache Tomcat เป็น web container

Implementation Plan

Activities\Time (Month-week)	September				October				November			
	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th
1) ศึกษาและออกแบบโครงสร้างของระบบ												
2) สร้างส่วน Data Collecting เพื่อดึงข้อมูลมาจาก vCenter												
3) สร้างส่วนตัวสร้างแบบจำลองและดาต้าเบส												
4) ทดสอบส่วนแบบจำลองข้อมูลและดาต้าเบส												
5) สร้างส่วนแสดงผลข้อมูล												
6) ทดสอบส่วนแสดงผลข้อมูล												
7) ทดสอบภาพรวมของระบบทั้งหมด												
8) ส่งชิ้นงาน												

รายละเอียดของแต่ละส่วนมีดังนี้

1) ศึกษาและออกแบบโครงสร้างของระบบ

ศึกษาภาพรวมการใช้งาน Azure Service ว่ามี service ใดบ้าง มี feature การทำงานเป็นอย่างไร และออกแบบส่วนต่างๆ ของโปรเจค ความสัมพันธ์ของแต่ละส่วน Software, Framework หรือ API ใดที่จะใช้ในการติดต่อกันในแต่ละส่วน

ผู้รับผิดชอบ : นายนพกร , นางสาวนนทยา

2) ศึกษาและสร้างส่วน Data Collecting เพื่อดึงข้อมูลมาจาก vCenter

ศึกษาส่วนของ Data Collecting ว่าต้องใช้ API ใดในการติดต่อกับ vCenter เพื่อดึงข้อมูลออกมา, ศึกษาส่วน Format ของข้อมูลให้ได้รูปแบบที่ง่ายต่อการนำไปใช้, ศึกษาการจัดเรียงลำดับข้อมูลในการประมวลผล เพื่อให้การทำงานมีประสิทธิภาพมากที่สุด

Data collecting script

Input : ข้อมูลจาก vCenter ที่ผ่านการจัดรูปแบบแล้ว

Operation : รวบรวมข้อมูลต่างๆจาก vCenter และจัดให้อยู่ในรูปแบบที่มีลำดับเช่นต้องมีการสร้าง vm ก่อนทำการตั้งค่าต่างๆให้ vm ดังกล่าวเป็นต้น

Output : คำสั่งเพื่อติดต่อ API ของ vSphere

Software/Language/Framework/API : Python (PySphere)

Data collecting API

Input : ข้อมูลที่จัดรูปแบบแล้ว

Operation : รวบรวมข้อมูลที่จัดรูปแบบแล้วจากหลายๆ vCenter Serverและจัดเก็บลง Data Storage

Output : การเขียนข้อมูลลงบน Data Storage

Software/Language/Framework/API : Java (JAX-RS และ Jersey)

ผู้รับผิดชอบ : นายนพกร

3) สร้างส่วนตัวสร้างแบบจำลองและดาต้าเบส

ออกแบบแบบจำลองในการเก็บข้อมูลและศึกษาดาต้าเบส (neo4j) ทั้งการ insert update, delete, query, และ performance

Modeler

Input : 1) ข้อมูลที่จัดรูปแบบแล้วจาก Data Storage

2) ข้อมูลจาก Database

Operation : ดึงข้อมูลบางส่วนจาก Database มาเพื่อวิเคราะห์ ข้อมูลที่มาจาก Data storage แล้วประมวลผลเป็นคำสั่งเพื่อ update ข้อมูลใน Database

Output : คำสั่งเพื่อ update ข้อมูลใน Database

Software/Language/Framework/API : Java, Cipher query

ผู้รับผิดชอบ : นายนพกร

Database

input : 1) Cipher query จาก Modeler

2) คำสั่งจาก Web Server

Operation : update, insert, delete หรือ retrieve ข้อมูลใน Database.

Output : ข้อมูล (ในรูปแบบ JSON)

Software/Language/Framework/API : ใช้ neo4j ในการ implement Database และใช้ Cypher Query Language

ผู้รับผิดชอบ : นางสาวนนทยา

4) ทดสอบส่วนแบบจำลองข้อมูลและดาต้าเบส

ทดลองดึง data ที่ต้องการออกมาเพื่อตรวจสอบว่าเก็บข้อมูลที่ต้องการใส่ลงไปได้ถูกต้องหรือไม่ โดยจะใส่ชุดข้อมูลทดลองเข้าไปในระบบเพื่อตรวจสอบว่าโปรแกรมสามารถทำงานได้ถูกต้องหรือไม่

ผู้รับผิดชอบ : นางสาวนนทยา

5) สร้างส่วนแสดงผลข้อมูล

ศึกษาและเขียนโปรแกรมเพื่อติดต่อกับ Database โดยใช้ HTTP response/request ที่เป็น JSON ผ่านทาง Web server ร่วมกับ neo4j REST API และแสดงผลผ่านหน้าเว็บ

Visualizer

Input : 1) คำสั่งจากผู้ใช้
2) ข้อมูลจาก Database

Operation : รับข้อมูลมาประมวลผลเพื่อแสดงภาพตามที่ผู้ใช้ ร้องขอ

Output : 1) สร้างกราฟและแสดงผลที่ตามที่ผู้ใช้ร้องขอ
2) HTTP request to Database

Software/Language/Framework/API : Java, Neo4j Graph-Style-Sheet & JavaScript

ผู้รับผิดชอบ : นายนพกร(Web server), นางสาวนนทยา(Visualizer: CSS, HTML และ JavaScript)

6) ทดสอบส่วนแสดงผลข้อมูล

ทดลองแสดงผลลัพธ์ในส่วนที่ต้องการว่าแสดงผลได้ถูกต้องตรงตามข้อมูลหรือไม่ และถูกต้องตรงตาม query ที่ต้องการ

ผู้รับผิดชอบ : นางสาวนนทยา

8) ทดสอบภาพรวมของระบบทั้งหมด

ทดสอบทุกระบบว่าสามารถทำงานได้ถูกต้อง ทั้งการเก็บข้อมูล แสดงผลข้อมูล โดยทุกส่วนสามารถทำงานติดต่อกันได้อย่างไม่มีปัญหา

ผู้รับผิดชอบ : นายนพกร , นางสาวนนทยา

9) ส่งชิ้นงาน

จัดทำรูปเล่มสรุปรายงานผล และตัวโครงงานเพื่อส่ง