ต้นแบบสถาปัตยกรรมแบบไม่รวมศูนย์โดยใช้บล็อกเชนสำหรับข้อมูลและการวิจัยด้านการแพทย์: กรณีศึกษาของเครือโรงพยาบาลเอกชนในประเทศไทย

A Prototype of Decentralized Architecture Using Blockchain for Medical Records and Research: A Case Study of a Private Hospital Group in Thailand

ไตรศักดิ์ ไตรเสนีย์, อมิตา มงคลปรีดาไชย และ ธีรพงศ์ ลีลานุภาพ

คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

1 ซอยฉลองกรุง 1 เขตลาดกระบัง แขวงลาดกระบัง กรุงเทพมหานคร 10520

E-mails: traisak_tra@outlook.com, amt_mpch@hotmail.com, teerapong@it.kmitl.ac.th

ABSTRACT – In Information Technology (IT), Blockchain does not only benefit economic but also greatly benefits other domains, such as scientific, political and medical. It provides a new solution with lower-cost, higher-scalability and greater-security for storing and controlling access to massive data. In medical domain, shifting from typical paper-based to electronic health records results in an increasing number of health data. Such health data, also collected by today's personal ubiquitous computing devices (e.g., smart-watches and phones), are very useful for preventive and proactive medical research. Therefore, health IT systems require a new architecture to record medical information. This article presents the design of a prototype of decentralized architecture for medical record using private Blockchain. As a case study of this research, the largest group of private hospitals in Thailand has been collaborating with us in providing their initial requirements for the design and implementation. Our design work involves the use of a private Blockchain and its smart contract as an access control manager to medical records. Different tools for Blockchain technologies are also reviewed. This article poses a next step towards the proof-of-concept in implementing Blockchain in Thailand as Blockchain-as-a-service (BaaS) for medical records.

KEYWORDS -- Medical Record; Ethereum; Blockchain; Smart contract; Decentralized Application; Encryption; Blockchain as a Service

บทคัดย่อ -- ในแวดวงเทคโนโลยีสารสนเทศ บล็อกเชนไม่ได้มีประโยชน์ต่อทางเศรษฐศาสตร์เท่านั้น แต่ยังมีประโยชน์ต่อสาขาวิชาอื่น ๆ เช่น วิทยาศาสตร์ การเมือง และการแพทย์ บล็อกเชนทำให้เกิดวิธีแก้ไขใหม่ในการจัดเก็บและควบคุมการเข้าถึงข้อมูลมหาศาล ซึ่งมีค่าใช้จ่ายที่น้อยลง ความสามารถในการขยายระบบที่สูงขึ้น และความปลอดภัยที่เพิ่มขึ้น ในสาขาด้านการแพทย์ การเปลี่ยนผ่านจากรูปแบบดั้งเดิมของการเก็บข้อมูลสุขภาพลงในกระดาษ ไปสู่รูปแบบอิเล็คทรอนิคส์นั้นทำให้เกิดการเพิ่มจำนวนของข้อมูลสุขภาพเป็นอย่างมาก โดยข้อมูลสุขภาพเหล่านี้ยังรวมถึงข้อมูลที่ได้มาจากอุปกรณ์ส่วนบุคคลต่าง ๆ ที่มีการใช้งานกันอย่างแพร่หลาย (เช่น สมาร์ทวอช์ทและสมาร์ทโฟน) ซึ่งส่งผลดีต่อการนำข้อมูลสุขภาพไปใช้ในงานวิจัยทางด้านการแพทย์ทั้งเชิงการป้องกันและรักษา ดังนั้น ระบบสารสนเทศสำหรับข้อมูลสุขภาพจึงต้องการสถาปัตยกรรมใหม่ในการบันทึกข้อมูลทางด้านการแพทย์ บทความนี้นำเสนอการออกแบบตัวต้นแบบของสถาปัตยกรรมแบบไม่รวมศูนย์กลางสำหรับการเก็บบันทึกข้อมูลสุขภาพโดยใช้บล็อกเชนส่วนตัว โดยได้รับความร่วมมือจากเครือโรงพยาบาลเอกชนที่ใหญ่ที่สุดในประเทศไทย ซึ่งผู้พัฒนาได้ใช้เป็นกรณีศึกษาของงานวิจัยนี้ บทความนี้ยังได้นำเสนอการ

ใช้งานบล็อกเชนส่วนตัวและสมาร์ทคอนแทรคท์เป็นเครื่องมือในการจัดการควบคุมการเข้าถึงข้อมูลทางด้านการแพทย์ รวมทั้งทำการทบทวนวรรณกรรมที่เกี่ยวข้องกับเครื่องมือต่าง ๆ ที่ใช้ในเทคโนโลยีบล็อกเชน โดยบทความนี้ได้สร้างความก้าวหน้าของการพิสูจน์แนวคิดของการพัฒนาบล็อกเชนในรูปแบบการให้บริการบล็อกเชนสำหรับข้อมูลด้านการแพทย์

คำสำคัญ-- ข้อมูลทางการแพทย์; บล็อกเชน; อีเธอเรียม; สมาร์ทคอนแทรคท์; แอปพลิเคชันแบบไม่รวมศูนย์; การเข้ารหัสลับ; การให้บริการบล็อกเชน

1. บทนำ

ในประเทศไทย โรงพยาบาลส่วนใหญ่ใช้วิธีการเก็บข้อมูลลงในฐานข้อมูลแบบรวมศูนย์ (Centralized Database) ทั้งนี้มีเหตุผลเนื่องมาจากการจัดเก็บฐานข้อมูลแบบรวมศูนย์นั้นมีการถูกนำมาใช้งานอย่างแพร่หลายและเป็นที่คุ้นเคยของนักพัฒนาและดูแลระบบเป็นระยะเวลานาน ทำให้การตั้งระบบและการดูแลรักษาฐานข้อมูลแบบรวมศูนย์นั้นจึงมีความง่ายและรวดเร็วกว่าแบบอื่น ๆ ที่มีความซับซ้อนมากกว่า โดยการออกแบบและใช้รูปแบบโครงสร้างข้อมูลของแต่ละโรงพยาบาลนั้นมีแตกต่างกันไปขึ้นอยู่กับผู้พัฒนา อย่างไรก็ตาม รูปแบบการเก็บข้อมูลแบบรวมศูนย์ดังกล่าวอาจส่งผลให้เกิดปัญหาจากการเติบโตข้อมูลในอนาคต ทั้งด้านค่าใช้จ่ายที่เพิ่มสูงขึ้น และด้านความเสถียรของระบบที่ลดลงตามปริมาณข้อมูลในฐานข้อมูลที่เพิ่มสูงขึ้น นอกจากนั้นหากเกิดข้อผิดพลาดกับคลังข้อมูล ก็อาจส่งผลให้ส่วนเชื่อมต่ออื่น ๆ ที่เหลือ ทำงานผิดพลาดตามไปด้วย นอกจากนี้ ระบบในปัจจุบันไม่ได้มีการอำนวยความสะดวกในการเข้าถึงจากผู้ใช้ประเภทต่าง ๆ โดยตรง ดังตัวอย่างเช่น

1) ผู้ป่วยซึ่งเป็นเจ้าของข้อมูลประวัติทางด้านการแพทย์หรือการรักษาของตนเองนั้น มีความยากลำบากในการเข้าถึงข้อมูลของตนเอง เมื่อมีความจำเป็นในการนำข้อมูลการรักษาของตนเองนั้น ไปใช้ในการเข้ารับการรักษาต่อที่โรงพยาบาลอื่น ๆ ที่ต้องการประวัติการรักษาของคนไข้ย้อนหลัง เพื่อประกอบการรักษา

2) นักวิจัยที่ต้องการข้อมูลทางด้านการแพทย์ ไปใช้เพื่อศึกษาและวิเคราะห์เพื่อทำงานวิจัย มีความความยากลำบากในการรวมรวมและการขอใช้ข้อมูล ทั้งนี้เพื่อไม่ให้เกิดปัญหาทางจริยธรรมทางด้านการแพทย์ จากการละเมิดความเป็นส่วนตัวผู้ป่วยซึ่งเป็นเจ้าของข้อมูล

3) บริษัทประกันสุขภาพ ที่ต้องการตรวจสอบประวัติการรักษาของคนไข้ เพื่อรักษาหรือสงวนสิทธิประโยชน์ของผู้ประกันตน โดยประวัติการรักษาของผู้ประกันตนนั้นจะต้องสามารถถูกตรวจสอบในแง่ของบูรณภาพของข้อมูลได้ (Data Integrity)

ในปี 2016 ได้มีงานวิจัยที่นำเสนอการนำบล็อกเชน (Blockchain) มาใช้จัดการเก็บข้อมูลทางการแพทย์แบบไม่รวมศูนย์ (Decentralized Database) Ekblaw et. al. [1] นำเสนอการพัฒนาตัวต้นแบบ MedRec ในการประยุกต์ใช้สมาร์ทคอนแทรคท์ (Smart Contract) เพื่อควบคุมการเข้าถึงข้อมูลทางการแพทย์ และทำการเก็บข้อมูลสำหรับการศึกษาระบบวินิจฉัยโรคแบบอัตโนมัติ ในขณะที่ยังคงความเป็นส่วนตัวของผู้ป่วยซึ่งเป็นเจ้าของข้อมูล รวมทั้งสร้างระบบการแลกเปลี่ยนข้อมูลสาธารณสุขที่มีความน่าเชื่อถือ

Palmer [2] นำเสนอการใช้โครงสร้างพื้นฐานข้อมูลลงลายเซ็นแบบไม่ใช้กุญแจ (Keyless Signature Infrastructure) กับระบบ e-Hospital ซึ่งถูกใช้งานระดับประเทศ ซึ่งถูกใช้ในประเทศเอสโตเนีย

โดยผลที่ได้ตามมาจากสองงานวิจัยข้างต้น คือ ระบบที่ใช้รูปแบบโครงสร้างแบบไม่รวมศูนย์ในการควบคุมสิทธิ์ให้เป็นไปตามที่แต่ละโรงพยาบาลกำหนด ช่วยลดต้นทุนค่าใช้จ่ายในด้านอุปกรณ์ ช่วยให้เกิดการเชื่อมต่อและทำงานร่วมกันได้ดีขึ้น และมีความน่าเชื่อถือโปร่งใสมากขึ้น เนื่องจากตรวจสอบย้อนหลังได้ง่ายและยากต่อการลักลอบแก้ไขจากบุคคลภายนอก ส่งผลให้ข้อมูลมีความน่าเชื่อถือ

2. เทคโนโลยี เครื่องมือ และงานวิจัยที่เกี่ยวข้อง

ส่วนนี้ทำการอธิบายถึงเทคโนโลยีและเครื่องมือที่นำมาใช้ในการออกแบบสถาปัตยกรรมแบบไม่รวมศูนย์ และหลักการพัฒนาสถาปัตยกรรมเพื่อจัดการควบคุมการเข้าถึงข้อมูล

รวมถึงงานวิจัยที่เกี่ยวข้องซึ่งนำเสนอการใช้เทคโนโลยีบล็อกเชนมาใช้ในการออกแบบสถาปัตยกรรมสำหรับข้อมูลสุขภาพของโรงพยาบาล

2.1 เทคโนโลยีที่เกี่ยวข้อง

2.1.1 Blockchain

บล็อกเชน [3] เป็นเทคโนโลยีสำหรับเก็บข้อมูลแบบไม่รวมศูนย์ซึ่งถูกออกแบบมาเพื่อเน้นความปลอดภัยโดยการเข้ารหัสลับ (Cryptographic security) ความถูกต้องของข้อมูล (Data validity) และจัดการข้อมูลที่มีความอ่อนไหวสูง (Highly sensitive information) นอกจากนั้นยังสามารถขยายเพื่อรองรับข้อมูลที่มีแนวโน้มที่จะเพิ่มสูงขึ้นอย่างต่อเนื่อง (Volume scalability) โครงสร้างของการเก็บข้อมูลในบล็อกเชนนั้นจะถูกจัดเก็บเป็นลักษณะของชุดข้อมูลที่มีหน่วยเป็น “บล็อก” (Block) ซึ่งมีการเก็บข้อมูลการลงเวลาที่แก้ไขล่าสุดและความสัมพันธ์ระหว่างบล็อกโดยแสดงการอ้างอิงของบล็อกข้อมูลก่อนหน้า ในปัจจุบันบล็อกเชนได้มีการพัฒนาอย่างต่อเนื่องจนถึงยุค 3.0

Blockchain 1.0 - Crypto Currency: ในช่วงแรกที่เทคโนโลยีบล็อกเชนถูกสร้างขึ้นมานั้นมีประโยชน์ในวงการการเงินการธนาคารสูงมาก ในการจัดเก็บข้อมูลธุรกรรมทางด้านการเงิน เนื่องจากมีธุรกรรมเกิดขึ้นเป็นจำนวนมากในทุก ๆ วินาที บัญชีของแต่ละบุคคลจะถูกเก็บเป็นความลับเสมอกันโดยมีเพียงธนาคารเท่านั้นที่ทราบ จึงถูกนำไปใช้ในรูปแบบของสกุลเงินที่ทุกคนสามารถทำธุรกรรมร่วมกันและช่วยกันตรวจสอบความถูกต้องได้ อีกทั้งยังไม่สามารถสร้างข้อมูลที่เป็นเท็จอีกด้วย ซึ่งในช่วงนี้จะมีสกุลเงินที่รู้จักกันทั่วไป เช่น BitCoin MasterCoin Ripple

Blockchain 2.0 - Smart Contract: ต่อมาเทคโนโลยีบล็อกเชนได้ถูกพัฒนาให้สามารถเขียนโปรแกรมเข้าไป เพื่อควบคุมเงื่อนไขต่าง ๆ ของบล็อกเชนได้โดยไม่จำเป็นต้องพึ่งตัวกลางใด ๆ เช่น ผู้ซื้อขายสินค้าออนไลน์และผ่านทางโซเชียลมีเดีย สามารถลดความเสี่ยงของการชำระเงินลงได้ การโอนเงินข้ามประเทศจะสามารถสำเร็จภายในไม่กี่นาทีจากเดิมที่ใช้เวลาหลายวัน การอนุมัติสินเชื่อที่สามารถทำได้อย่างรวดเร็ว หรือการเคลมประกันแบบอัตโนมัติ โดยไม่จำเป็นต้องใช้เอกสาร รวมถึงยังเป็นการเพิ่มความโปร่งใสในการทำธุรกรรม ที่จะช่วยให้การกำกับดูแลของหน่วยงานภาครัฐให้เป็นไปได้อย่างมีประสิทธิภาพ ซึ่งรูปแบบนี้สามารถบังคับใช้สิ่งที่กำหนดไว้ในสัญญาได้อัตโนมัติ มีความถูกต้อง โปร่งใส เพิ่มความน่าเชื่อถือและประสิทธิภาพการทำงาน ช่วยลดขั้นตอนการทำงานและลดข้อผิดพลาดที่อาจเกิดจากมนุษย์ (Human Error) ซึ่งกลายมาเป็นโครงสร้างพื้นฐานของแอปพลิเคชันแบบไม่รวมศูนย์ (Decentralized Application)

Blockchain 3.0 - แอปพลิเคชันแบบไม่รวมศูนย์ (Decentralized Application - Dapp) บล็อกเชนพัฒนาเข้าสู่ยุคที่นำเทคโนโลยีนี้มาพัฒนาในเชิงธุรกิจอย่างจริงจัง ในรูปแบบของแอปพลิเคชัน เช่น การจัดการสิทธิของที่ดิน การป้องกันการทุจริตในภาครัฐ และการพิสูจน์การเป็นเจ้าของผลงาน ซึ่งแอปพลิเคชันในยุคนี้จะต่างจากยุคก่อนที่ไม่จำเป็นต้องทำงานจากเซิร์ฟเวอร์ส่วนกลาง แต่กระจายการทำงานอยู่ในอุปกรณ์บนเครือข่ายคอมพิวเตอร์ที่เชื่อมต่อถึงกันทั่วโลกและใช้สื่อสารกันได้ โดยแอปพลิเคชันจะสามารถทำงานได้ตลอดเวลาไม่มีวันปิด ตราบใดที่ยังมีบุคคลใดบุคคลหนึ่งบนโลกยังสั่งการทำงานของแอปพลิเคชัน

อย่างไรก็ตาม บล็อกเชนก็ยังมีข้อจำกัด เหมือนกับปัญหาที่ยังคงมีอยู่ในสกุลเงินดิจิทัล เช่น BitCoin ที่ยังต้องได้รับการปรับปรุงพัฒนาแก้ไข ตัวอย่างเช่น

- ความช้า ซึ่งจำเป็นต้องใช้เวลาที่มากเพียงพอในการยืนยันธุรกรรมที่เกิดขึ้น
- การขยายการรองรับจำนวนธุรกรรมที่เกิดข้อต่อหนึ่งหน่วยเวลา (Throughput Scalability) ซึ่งปัจจุบันสามารถทำได้ 8-10 ธุรกรรมต่อหนึ่งวินาที
- ความยากในการพัฒนาอย่างยั่งยืน เนื่องจากการแยกห่วงโซ่ของบล็อกในบล็อกเชนโดยการไม่ยอมรับธุรกรรมจากโหนด (Node) ที่ไม่ยอมอัพเดทซอฟต์แวร์ตามโปรโตคอลใหม่ หรือเรียกว่า Hard fork

ถึงแม้ว่าบล็อกเชนยังคงมีข้อจัดกัด ดังตัวอย่างข้างต้น เทคโนโลยีบล็อกเชนก็ยังคงมีศักยภาพและประโยชน์อีกมากมาย ซึ่งควรค่าแก่ศึกษาและถูกนำมาประยุกต์ใช้ได้หลากหลายในภาคอุตสาหกรรมต่าง ๆ

2.1.2 Decentralized Storage

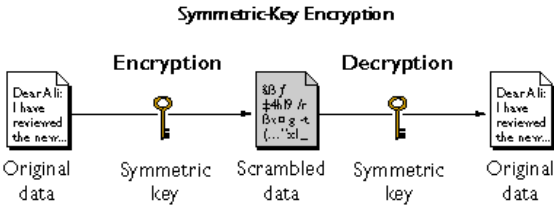
การจัดเก็บข้อมูลแบบไม่รวมศูนย์ [4] คือ ระบบฐานข้อมูลที่มีการเก็บข้อมูลไว้ในระบบคอมพิวเตอร์หลาย ๆ เครื่อง ซึ่งเครื่องที่ติดตั้งนั้นอาจตั้งอยู่ตามที่ต่าง ๆ โดยระบบคอมพิวเตอร์เหล่านี้จะสื่อสารกันผ่านระบบเครือข่ายคอมพิวเตอร์ ซึ่งผู้ใช้สามารถเรียกใช้ข้อมูลจากฐานข้อมูลที่

อยู่ในเครื่องใดก็ได้และผู้ใช้ไม่จำเป็นต้องรับรู้ว่าข้อมูลที่ตนต้องการนั้นจัดเก็บอยู่บนเครื่องใดและมองเห็นเป็นระบบการจัดเก็บข้อมูลแบบรวมศูนย์

ลักษณะสำคัญของการจัดเก็บข้อมูลแบบไม่รวมศูนย์ คือ มีความเป็นอิสระของแต่ละส่วนงาน ไม่ขึ้นกับส่วนกลาง ทำให้สามารถขยายระบบได้ง่ายขึ้นและสามารถลดข้อผิดพลาดเรื่องจุดเดียวของความล้มเหลว (Single point of failure) ได้ดีกว่าแบบรวมศูนย์ อย่างไรก็ตาม การจัดเก็บข้อมูลแบบไม่รวมศูนย์มีระบบที่ค่อนข้างซับซ้อน ฉะนั้นจะต้องให้ความสำคัญต่อการควบคุมความถูกต้องของข้อมูล

2.1.3 Encryption

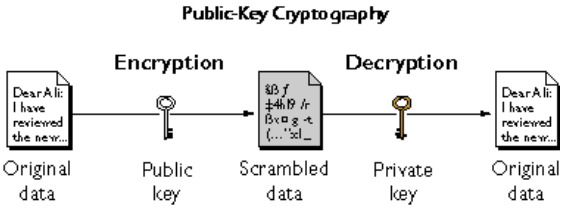
เทคโนโลยีการเข้ารหัสลับข้อมูล [5] เป็นกระบวนการสำหรับการแปรรูปข้อมูลอิเล็กทรอนิกส์ธรรมดาให้อยู่ในรูปที่บุคคลทั่วไป *ไม่* สามารถอ่านเข้าใจได้ โดยทั่วไปการเข้ารหัสลับจะกระทำก่อนการจัดเก็บข้อมูลหรือก่อนการส่งข้อมูล โดยการนำข้อมูลอิเล็กทรอนิกส์ธรรมดากับกุญแจ (Key) ซึ่งเป็นตัวเลขซึ่งถูกสุ่ม มาผ่านกระบวนการทางคณิตศาสตร์ และผลที่ได้คือข้อมูลที่ถูกเข้ารหัสลับ ขั้นตอนที่กล่าวมานี้จะเรียกว่า “การเข้ารหัสลับ” (Encryption) และเมื่อต้องการอ่านข้อมูล จะต้องนำเอาข้อมูลที่ถูกเข้ารหัสลับและกุญแจมาผ่านกระบวนการทางคณิตศาสตร์ จะได้ผลลัพธ์ที่เป็นข้อมูลดั้งเดิม ขั้นตอนนี้จะเรียกว่า “การถอดรหัส” (Decryption) ระบบเข้ารหัสลับสามารถแบ่งตามวิธีการใช้กุญแจได้เป็น 2 วิธีคือ ระบบเข้ารหัสลับแบบกุญแจสมมาตรและอสมมาตร



รูปที่ 1. ระบบเข้ารหัสลับแบบกุญแจสมมาตร

(IBM Knowledge Center)

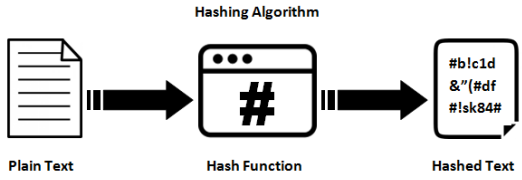
ระบบเข้ารหัสลับแบบกุญแจสมมาตร (Symmetric-key cryptography) คือ การเข้ารหัสลับข้อมูลด้วยกุญแจเดี่ยว ทั้งผู้ส่งและผู้รับใช้กุญแจแบบเดียวกันทั้งการเข้าและถอดรหัส โดยวิธีการนี้ผู้ส่งและผู้รับจะต้องตกลงวิธีในการเข้ารหัสลับข้อมูล ซึ่งรูปแบบในการเข้ารหัสลับข้อมูลที่ผู้ส่งและผู้รับตกลงกันแท้ที่จริงก็ คือ กุญแจ (แสดงดังรูปที่ 1.)



รูปที่ 2. ระบบเข้ารหัสลับแบบกุญแจอสมมาตร

(IBM Knowledge Center)

ระบบเข้ารหัสลับแบบกุญแจอสมมาตร (Asymmetric-key cryptography หรือ Public Key Technology) คือการเข้ารหัสลับวิธีนี้จะใช้หลักกุญแจคู่ทำการเข้ารหัสลับและถอดรหัส โดยกุญแจจะประกอบไปด้วย กุญแจส่วนตัว (Private key) และกุญแจสาธารณะ (Public key) โดยหลักการทำงานคือ หากใช้กุญแจใดเข้ารหัสลับ จะต้องใช้อีกกุญแจหนึ่งถอดรหัส สำหรับการเข้ารหัสลับและถอดรหัสด้วยกุญแจคู่นี้จะใช้ฟังก์ชันทางคณิตศาสตร์เข้ามาช่วยโดยที่ฟังก์ชันทางคณิตศาสตร์ที่นำมาใช้ ได้รับการพิสูจน์แล้วว่าจะมีเฉพาะกุญแจคู่ของมันเท่านั้นที่จะสามารถถอดรหัสได้ ไม่สามารถนำกุญแจคู่อื่นมาถอดรหัสได้อย่างเด็ดขาด (แสดงดังรูปที่ 2.)



รูปที่ 3. Hash Function

(File Change Detection and Remediation, 2017)

Hash Function คือการเข้ารหัสลับแบบกุญแจอสมาตรประเภทหนึ่ง ซึ่งเป็นการเข้ารหัสลับทางเดียว โดยจะสร้างลายเซ็นต์ดิจิตัล (Digital Signature) ของข้อมูลอิเล็กทรอนิกส์ที่ *ไม่* สามารถถอดรหัสกลับมาได้ คุณสมบัติหลักของ Hash Function คือ ค่า Hash ของข้อความเดิมจะต้องเหมือนกันเสมอ ในทางตรงกันข้าม ข้อความที่ไม่เหมือนกันจะไม่มีทางที่จะมี Hash ได้ค่าเดียวกัน ความยาวของข้อความที่ผ่านการ Hash จะต้องเท่ากันไม่ว่าจะเป็นเพียงข้อความสั้น ๆ หรือทั้งไฟล์ก็ตาม ซึ่งค่า Hash ที่ได้มาจะ *ไม่* สามารถนำไปถอดรหัสกลับเพื่อหาข้อความเดิมได้ Bitcoin ได้นำเอา

คุณสมบัติเหล่านี้มาใช้ในการตรวจสอบความถูกต้องของรายการ โดยการสร้างบัญชีแยกประเภท (Ledger Address) ของแต่ละบัญชีเพื่อใช้แทนเลขที่บัญชี โดยเลขที่บัญชีของ Bitcoin จะสร้างจาก Public key ที่นำมาผ่านกระบวนการ Hash Function (แสดงดังรูปที่ 3.)

2.2 เครื่องมือที่เกี่ยวข้อง

2.2.1 อีเธอเรียมบล็อกเชน (Ethereum Blockchain)

ภายในช่วงระยะเวลาหลายปีที่ผ่านมา นักพัฒนามากมายเริ่มนำบล็อกเชนมาใช้เป็นพื้นฐานในด้านการเงิน แต่สำหรับยุคต่อไปของการใช้บล็อกเชนจะมุ่งเน้นการสร้างแพลตฟอร์มสำหรับทั้งนักพัฒนาและผู้ใช้ทั่วไป ให้สามารถใช้ประโยชน์จากเครือข่ายแบบไม่รวมศูนย์ที่ช่วยกระจายความสามารถและอำนาจการควบคุมออกจากระบบแบบศูนย์กลาง ลดค่าใช้จ่ายจากการทำงานแบบ proof-of-work [6] ที่ต้องใช้อุปกรณ์ประมวลผลจำนวนมาก ในขณะที่สามารถเพิ่มขีดจำกัดด้านความรวดเร็วและยืดหยุ่น สิ่งเหล่านี้คือจุดประสงค์หลักของแนวคิดการสร้างบล็อกเชนแบบอีเธอเรียม

แนวคิดอีเธอเรียมบล็อกเชน เริ่มพัฒนาโดย Vitalik Buterin [6] นักพัฒนาชาวรัสเซียในปี 2013 ซึ่ง Buterin เคยเป็นผู้ร่วมพัฒนา Bitcoin มาก่อนแต่ก็เห็นข้อบกพร่องและข้อจำกัดของ Bitcoin ทำให้เขาเริ่มสร้างอีเธอเรียม ซึ่งมีสิ่งที่เพิ่มเติมเข้ามา คือ ฟีเจอร์ที่เรียกว่า สมาร์ทคอนแทรคท์ ที่อนุญาตให้เราสามารถเขียนโปรแกรมลงไปในข้อมูลของสกุลเงิน Ether ให้ทำงานอัตโนมัติเมื่อเจอเงื่อนไขที่กำหนด เทคโนโลยีที่ใช้ในอีเธอเรียมบล็อกเชน ประกอบไปด้วย

1) การเข้ารหัสลับข้อมูลด้วยกุญแจสาธารณะ
2) การเข้ารหัสลับด้วย Hash-function
3) เครือข่ายแบบ Peer-to-peer
4) EVM (Ethereum Virtual Machine) และ
5) Proof-of-stake

เมื่อผู้ใช้ทำการสร้างธุรกรรม (Transaction) รายการธุรกรรมนั้นจะถูกส่งไปยังเครือข่าย Peer-to-peer ที่สมาชิกในเครือข่ายจะเก็บข้อมูลเดียวกันและมีการใช้งาน EVM เพื่อที่จะคอยตรวจสอบการเปลี่ยนแปลง และสถานะต่างๆในเครือข่าย จากนั้นกลไก Proof-of-stake [8] ที่รวมอยู่ในระบบจะทำหน้าที่ในเลือกผู้ที่เหมาะสมมาทำการรับรองความถูกต้องโดยใช้หลักการ Merkle Tree [6] จากนั้นผู้ที่ทำการรับรองความถูกต้องและสามารถ Execute คำสั่งตามสมาร์ทคอนแทรคท์ได้ โดยจะได้รับค่าธรรมเนียมเป็นสิ่งตอบแทน หลังจากนั้นผู้ที่ได้รับค่าตอบแทนจะทำการจัดเก็บธุรกรรมดังกล่าวลงในบล็อก พร้อมกับ Hash ของส่วนหัวข้อมูลของบล็อกก่อนหน้า (Previous block header) และทำการประกาศออกไปในเครือข่ายให้สมาชิก เพื่อทำสำเนาข้อมูลรูปแบบเดียวกันขึ้นมาเพื่อป้องกันการลักลอบแก้ไข

2.2.2 Test RPC (Simulated Blockchain private network)

ในการพัฒนาและทดสอบคอนแทรคท์นั้นจำเป็นที่จะต้องใช้ Blockchain network ในการติดตั้งเพื่อเรียกใช้งาน แต่การทดสอบคอนแทรคท์ในเครือข่ายสาธารณะของอีเธอเรียม (Public network) หรือเครือข่ายสาธารณะสำหรับทดสอบ (Public test network) ที่มีผู้ใช้งานอยู่จริงนั้นมีค่าใช้จ่ายสูง และต้องใช้กำลังในการประมวลผลสูงมาก จึงมีการพัฒนาเครือข่ายแบบจำลองขึ้นมาเพื่อใช้ทดสอบคอนแทรคท์โดยเฉพาะ ซึ่ง Test RPC เป็น Node.js โดยมีพื้นฐานจากอีเธอเรียมที่ช่วยจำลองสภาวะการทำงานของเครือข่ายแบบเต็มรูปแบบและมีคำสั่งเบื้องต้นในการจัดการกับโหนดของอีเธอเรียม เพื่อให้สามารถตั้งค่าสภาวะของ Network ได้ตามต้องการ

2.2.3 Web3 library และ Web3 framework

Web3 API ถูกกำหนดให้เป็น Library หลักในการพัฒนา Dapp เพื่อสร้างปฏิสัมพันธ์กับ Ethereum node โดย Web3 library มีการพัฒนาให้รองรับหลายภาษาและมี Framework ให้เลือกใช้หลากหลายตามจุดประสงค์ของการสร้าง นอกจากนี้การใช้ Framework จะช่วยเพิ่มประสิทธิภาพด้านการทดสอบและความรวดเร็วในการพัฒนาเนื่องจาก Framework ส่วนมากมีคำสั่งพื้นฐานที่สำคัญต่อการสร้าง Dapp เช่น Migration Compiler Deploy Testing เป็นต้น

2.2.4 Geth (Go Ethereum)

Ethereum platform ทำงานแบบไม่รวมศูนย์ โดยจุดสำคัญคือการไม่มีเซิร์ฟเวอร์ตัวกลางในการทำงาน แต่อาศัยฮาร์ดแวร์ของสมาชิกในระบบทั้งหมด เสมือนเป็นเครื่องเซิร์ฟเวอร์ที่ทำงานร่วมกัน โดยเครื่องสมาชิกแต่ละโหนดจำเป็นต้องเปิดใช้งานโปรแกรม Ethereum client ที่ชื่อ Geth เพื่อสร้างเครือข่ายขึ้นมาทำหน้าที่ในการ จัดเก็บ ประมวลผล และสื่อสารกันด้วย Ethereum Protocol

นักพัฒนาได้ออกแบบให้ Geth สามารถเรียกใช้แบบ Standalone client บนระบบปฏิบัติการต่าง ๆ ทั้ง Windows macOS และ Linux อีกทั้งยังสามารถนำ Geth library เข้าไปทำการฝังตัวกับ iOS และ Android เพื่อใช้งานบนอุปกรณ์เคลื่อนที่ซึ่งไม่เพียงแต่ภาษา Go เท่านั้นที่ถูกนำมาสร้าง Ethereum client แต่นักพัฒนายังสามารถเลือกใช้ภาษา Python หรือ C++ ในการเข้าใช้งาน Ethereum Protocol ได้เช่นกัน

2.3 งานวิจัยที่เกี่ยวข้อง

2.3.1 MedRec

ในปี ค.ศ. 2016 นักวิจัยจากสถาบันเทคโนโลยีแมสซาชูเซตส์ได้พัฒนาระบบต้นแบบ เพื่อทดสอบความเป็นไปได้และความเหมาะสมในการใช้บล็อกเชน ร่วมกับข้อมูลด้านสาธารณสุข อีกทั้งเพื่อศึกษาวิธีการทำงานของ โครงสร้างแบบไม่รวมศูนย์ในบล็อกเชนว่าสามารถเพิ่มประสิทธิภาพด้านความปลอดภัย และใช้งานร่วมกับระบบอื่น ๆ ที่เกี่ยวข้องได้อย่างไร โดยนักวิจัยเลือกได้เลือกใช้อีเธอเรียมบล็อกเชนที่มีจุดเด่นเรื่องสมาร์ทคอนแทรคท์ซึ่งสามารถนำมาพัฒนาแอปพลิเคชันแบบไม่รวมศูนย์ได้ แตกต่างจากบล็อกเชนรูปแบบอื่นที่มีจุดประสงค์หลักเพื่อใช้ในการทำธุรกรรม นอกจากนี้ สมาร์ทคอนแทรคท์ยังมีชุดคำสั่งที่เอื้อต่อการสร้างระบบควบคุมการเข้าถึงข้อมูลระหว่างแหล่งข้อมูลกับส่วนเชื่อมต่อผู้ใช้และสามารถจัดทำแหล่งเก็บข้อมูลแบบไม่รวมศูนย์ที่มีข้อดีในด้านความพร้อมใช้งาน ความสอดคล้องของข้อมูลและป้องกันข้อมูลสูญหายได้อย่างมีประสิทธิภาพ ผลที่ได้จากการพัฒนาข้างต้น คือ ระบบ MedRec [1] ที่ช่วยให้ข้อมูลการรักษา ข้อมูลการใช้ยา และผลการทดสอบในห้องทดลอง สามารถตรวจสอบและทำงานร่วมกันได้ อีกทั้งยังสามารถกำหนดสิทธิ์การเข้าถึงข้อมูลได้อย่างเหมาะสม นอกจากนี้ผู้ป่วยซึ่งเป็นเจ้าของข้อมูลจะมีสิทธิ์ในการควบคุมการใช้งานข้อมูลอย่างที่ควรจะเป็น และสนับสนุนให้นักวิจัยสามารถนำข้อมูลส่วนตัวที่ได้รับอนุญาตจากผู้ป่วยไปใช้ได้โดยไม่เกิดปัญหาทางจริยธรรมในการละเมิดความเป็นส่วนตัว

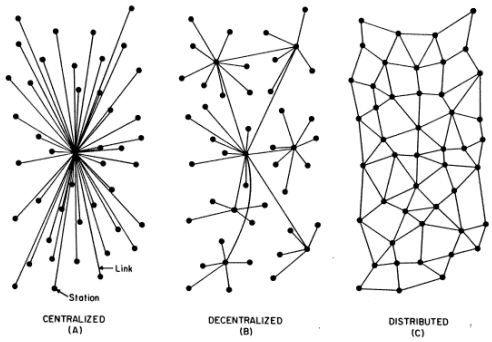
2.3.2 Estonia e-Health และ e-Prescription

สาธารณรัฐเอสโตเนียเป็นประเทศที่รัฐบาลให้ความสำคัญกับระบบสารสนเทศอย่างมาก มีการก่อตั้งมูลนิธิ e-Health foundation [2] ที่เป็นส่วนหนึ่งของ e-Government system ตั้งแต่ปี ค.ศ.1997 โดยระบบนี้ช่วยให้ประชาชนสามารถเข้าถึงทุกบริการและทุกหน่วยงานของภาครัฐ และมีการบังคับใช้รูปแบบการจัดการข้อมูลของหน่วยงานต่างภายใต้การควบคุม ให้มีความปลอดภัย สอดคล้องกัน และสามารถนำไปใช้งานร่วมกันได้ ในด้านสาธารณสุขก็เช่นกัน ประชาชนสามารถเข้าถึงประวัติการรักษา ข้อมูลการใช้ยาของตนเองที่เคยมีการบันทึกไว้จากสถานพยาบาลต่าง ๆ ได้ด้วย บัตรประชาชนที่มีการฝังชิปเอาไว้ นอกจากนี้แพทย์ผู้รักษาจากสถานพยาบาลต่าง ๆ รวมถึงนักวิจัยที่ต้องการข้อมูลสามารถทำการขอสิทธิ์ในการใช้งานได้

หลังจากนั้น ในปี ค.ศ. 2016 มูลนิธิ e-health มีการประกาศความร่วมมือกับบริษัทด้านความปลอดภัยในการปรับปรุงรูปแบบการเก็บข้อมูล โดยใช้ Keyless Signature Infrastructure (KSI) Blockchain เพื่อป้องกันการลักลอบแก้ไขข้อมูลการรักษาอย่างผิดกฎหมาย และสามารถตรวจสอบเส้นทางในการทุจริตได้ โดย KSI Blockchain นั้นถูกพัฒนาและใช้งานมาตั้งแต่ปี 2008 โดยเป็นบล็อกเชนที่ไม่มีบัญชีแยกประเภท ในทุกครั้งที่ไฟล์มีการเปลี่ยนแปลง จะมีการสร้างคีย์ใหม่แล้วจัดเก็บไว้ใน KSI Blockchain และการเพิ่มคีย์ให้กับสินทรัพย์ดิจิตอลอย่างเช่น ส่วนประกอบของโปรแกรม log file หรือ firmware นั้นยังช่วยให้สามารถสร้างระบบแจ้งเตือนอัตโนมัติเพื่อป้องกันผู้ไม่หวังดีทั้งหลาย นอกจากนี้ KSI Blockchain ยังมีกลไกการเข้ารหัสลับที่แตกต่างจากบล็อกเชนทั่วไปที่ใช้การเข้ารหัสลับแบบสมมาตร แต่ใช้เพียงแค่การเข้ารหัสลับแบบ Hash เท่านั้น

3. การเปรียบเทียบ

3.1 การเปรียบเทียบรูปแบบสถาปัตยกรรมแบบรวมศูนย์ ไม่รวมศูนย์ และกระจายศูนย์



รูปที่ 4. สถาปัตยกรรมแบบรวมศูนย์ ไม่รวมศูนย์ และกระจายศูนย์

ผู้คนส่วนมากมักคุ้นเคยกับคำว่า “แอปพลิเคชัน” ซึ่งแต่ละแอปพลิเคชันมีจุดประสงค์ที่ต่างกันออกไป ปัจจุบันมีมากกว่าหนึ่งพันล้านแอปพลิเคชันที่ใช้อยู่ในปัจจุบันและส่วนใหญ่มักอยู่ในรูปแบบของเว็บแอปพลิเคชันที่เป็นสถาปัตยกรรม client-server แบบรวมศูนย์หรือกระจายศูนย์บางส่วน และส่วนน้อยที่เป็นแบบไม่รวมศูนย์ รูปที่ 4 แสดงภาพรวมของ 3 รูปแบบสถาปัตยกรรมสำหรับแอปพลิเคชัน

ซอฟต์แวร์แอปพลิเคชันแบบรวมศูนย์ซึ่งเป็นที่นิยมมากที่สุดในปัจจุบัน จะควบคุมการทำงานของแต่ละหน่วยงานและการไหลของข้อมูลโดยตรงจากศูนย์กลางเดียว ทุกหน่วยงานจะขึ้นอยู่กับอำนาจกลางในการรับและการส่งข้อมูลจากคำสั่งที่ได้รับ องค์กรหลายแห่งทราบถึงประโยชน์ของการจัดเก็บไฟล์แบบรวมศูนย์ในไดร์ฟที่ใช้งานร่วมกันและยิ่งสะดวกมากยิ่งขึ้นเมื่อจัดเก็บข้อมูลแบบกลุ่มเมฆ เช่น Dropbox Google Drive แต่องค์กรเหล่านี้มักต้องออกแบบสภาพแวดล้อมสำหรับการประมวลผลและการจัดเก็บข้อมูลแยกออกจากกัน พื้นที่จัดเก็บข้อมูลแบบรวมศูนย์จะจัดเก็บไฟล์ข้อมูลและฐานข้อมูลที่ใช้ร่วมกันระหว่างเซิร์ฟเวอร์คอมพิวเตอร์ผ่านเครือข่ายหรือที่เรียกกันว่า การจัดเก็บข้อมูลแบบเครือข่าย (Networked storage) การจัดเก็บแบบรวมศูนย์ช่วยให้สามารถรวบรวมทรัพยากรการจัดเก็บข้อมูลได้ไม่ว่าจะอยู่ในระดับไฟล์หรือระดับบล็อก เพื่อให้สามารถใช้ที่เก็บข้อมูลได้อย่างมีประสิทธิภาพเนื่องจากเซิร์ฟเวอร์ทั้งหมดสามารถเข้าถึงพื้นที่เก็บข้อมูลเดียวกันได้ การจัดการและการสำรองข้อมูลทำได้ง่ายกว่าเนื่องจากเป็นศูนย์กลางและถูกมองว่าเป็นพื้นที่จัดเก็บข้อมูลเดียว อย่างไรก็ตาม ฐานข้อมูลแบบรวมศูนย์จะขึ้นอยู่กับการเชื่อมต่อเครือข่าย การเชื่อมต่ออินเตอร์เน็ตที่ช้าลงจะต้องใช้เวลาในการเข้าถึงฐานข้อมูลนานขึ้น อาจเกิดคอขวด (Bottlenecks) ขึ้นจากการเข้าชมที่สูง การเข้าถึงจะถูกจำกัดโดยผู้ใช้มากกว่าหนึ่งคนในชุดข้อมูลเดียวกันเนื่องจากมีสำเนาเพียงชุดเดียวและเก็บไว้ในที่เดียว ซึ่งอาจส่งผลให้ประสิทธิภาพการทำงานของระบบลดลงอย่างมาก ซึ่งหากไม่มีการตั้งค่าป้องกันความผิดพลาดและเกิดเสียหายของฮาร์ดแวร์ขึ้น ข้อมูลทั้งหมดภายในฐานข้อมูลจะสูญหายไป อีกทั้งมีข้อมูลที่เหลือน้อยที่สุดหากไม่มีข้อมูลสำรองหากชุดข้อมูลสูญหายโดยไม่คาดคิดเป็นเรื่องยากที่จะดึงข้อมูลกลับมา ซึ่งในกรณีส่วนใหญ่จะต้องทำด้วยตนเอง

สถาปัตยกรรมแบบไม่รวมศูนย์เป็นการประมวลผลที่เกิดจากการแบ่งการทำงานให้กับคอมพิวเตอร์ต่าง ๆ ภายในองค์กรหรือเครือข่ายเดียวกันโดยแต่ละหน่วยงายหรือแต่ละคอมพิวเตอร์สามารถจัดการเครื่องคอมพิวเตอร์ของตัวเองและไม่ต้องมีปฏิสัมพันธ์กับหน่วยงานอื่น ประเด็นหลักสำคัญเกี่ยวกับระบบแบบไม่รวมศูนย์คือการไม่มีศูนย์กลางควบคุมที่สำคัญ (Central Point of Control) ซึ่งลักษณะการเก็บข้อมูลเป็นไปตามหัวข้อย่อยที่ 2.1.2.

สำหรับสถาปัตยกรรมแบบกระจายศูนย์ ข้อมูลจะถูกจัดเก็บไว้ในหลาย ๆ เครื่องและจะเชื่อมต่อเข้าด้วยกันผ่านระบบเครือข่าย โดยแต่ละส่วนจะมีระบบจัดการฐานข้อมูลเป็นของตนเองและสามารถที่จะทำงานได้ด้วยตนเองหรือทำงานร่วมกันก็ได้ โดยคอมพิวเตอร์แต่ละเครื่องจะเรียกว่า “โหนด” (Node) แต่ละโหนดจะสามารถสื่อสารกันได้ซึ่งผู้ใช้สามารถเรียกใช้ข้อมูลจากฐานข้อมูลที่มีอยู่จากโหนดใดก็ได้และผู้ใช้ไม่จำเป็นต้องทราบว่าข้อมูลที่ตนต้องการนั้นจัดเก็บอยู่บนเครื่องใด หลักการพื้นฐานของฐานข้อมูลแบบกระจาย ได้ระบุไว้ว่า “สำหรับผู้ใช้นั้น ระบบกระจายควรจะให้เขามองเห็นเหมือนกันระบบที่ไม่ได้กระจาย” กล่าวอีกอย่างหนึ่งก็คือ ผู้ใช้ฐานข้อมูลในระบบกระจาย ควรจะสามารถกระทำสิ่งต่าง ๆ ได้เหมือนกับว่าระบบที่ใช้อยู่นั้นไม่ใช้ระบบกระจาย โดยเฉพาะอย่างยิ่งการกระทำที่เป็นการจัดกระทำข้อมูล (Data Manipulation Operation) เช่น ใช้คำสั่งภาษาสอบถามเชิงโครงสร้างเพื่อสืบค้นข้อมูล เป็นต้น ซึ่งการกระทำดังกล่าวควรจะให้ผู้ใช้รู้สึกว่าไม่แตกต่างจากระบบฐานข้อมูลแบบรวมศูนย์

ในขณะที่ฐานข้อมูลแบบรวมศูนย์นั้นเก็บข้อมูลไว้ในอุปกรณ์จัดเก็บข้อมูลที่อยู่ในตำแหน่งเดียวซึ่งเชื่อมต่อกับหน่วยประมวลผลเดียว ระบบฐานข้อมูลแบบกระจายศูนย์จะเก็บข้อมูลในอุปกรณ์จัดเก็บข้อมูลที่อาจอยู่ในตำแหน่งที่ต่างกันและจัดการโดยใช้ฐานข้อมูลส่วนกลาง ฐานข้อมูลแบบรวมศูนย์จะง่ายต่อการรักษาและปรับปรุงให้ดีขึ้นเนื่องจากข้อมูลทั้งหมดจะถูกเก็บไว้ในที่เดียว นอกจากนี้ยังง่ายต่อการรักษาความสมบูรณ์ของข้อมูลและหลีกเลี่ยงความซ้ำซ้อนของข้อมูล แต่คำขอทั้งหมดที่เข้ามาเพื่อต้องการเข้าถึงข้อมูลจะได้รับการประมวลผลโดยเอนทิตีเดียว เช่น เมนเฟรมเดียว จึงสามารถทำให้เกิดคอขวดได้อย่างง่ายดาย แต่ด้วยฐานข้อมูลแบบกระจายศูนย์ สามารถหลีกเลี่ยงคอขวดนี้ได้เนื่องจากฐานข้อมูลแบบขนานทำให้โหลดสมดุลระหว่างหลายเซิร์ฟเวอร์ แต่การเก็บรักษาข้อมูลให้ทันสมัยอยู่เสมอในระบบฐานข้อมูลแบบกระจายจำเป็นต้องใช้งานเพิ่ม

จึงทำให้ต้นทุนการบำรุงรักษาและความซับซ้อนเพิ่มขึ้นและต้องใช้ซอฟต์แวร์เพิ่มเติมเพื่อการนี้ด้วย นอกจากนี้การออกแบบฐานข้อมูลสำหรับฐานข้อมูลแบบกระจายมีความซับซ้อนมากกว่าฐานข้อมูลแบบรวมศูนย์

3.2 การเปรียบเทียบเครื่องมือพัฒนา

การพัฒนาแอปพลิเคชันแบบไม่รวมศูนย์โดยใช้สมาร์ทคอนแทรคท์สามารถนำเฟรมเวิร์คที่มีชุดคำสั่งสำเร็จรูปในการทดสอบ คอมไพล์และเรียกใช้งาน มาใช้เพื่อผลดีต่อการพัฒนาทั้งเรื่องความเร็วและความง่ายต่อการแก้ไขแอปพลิเคชัน นอกจากนี้ยังมีเฟรมเวิร์คหลายรูปแบบให้เรียกใช้ซึ่งมีข้อดีแตกต่างกันไป ดังตารางที่ 1

ตารางที่ 1. ตารางเปรียบเทียบเครื่องมือที่ใช้พัฒนา

Topic	Truffle	Embark	Dapple
Operation System	MacOS, Windows, Linux	Only Linux	MacOS, Linux
Package management	npm	npm	Nix
Dashboard	no	yes	no
Decentralize storage	Swarm, IPFS	Swarm, IPFS	IPFS
Automate testing	Mocha, Chai	no	DS-test

โดยผู้พัฒนาเลือกใช้ Truffle framework เนื่องด้วยความง่ายในการแก้ไขและใช้งานคอนแทรคท์ การสนับสนุนทุกระบบปฏิบัติการ ส่วนจัดการแพ็คเกจของ Node.js ส่วนเชื่อมต่อกับฐานข้อมูลแบบไม่รวมศูนย์ เช่น Ethereum Swarm และการที่มีผู้ใช้งานจำนวนมาก ทำให้มีตัวอย่างวิธีการแก้ไขปัญหาและการสนับสนุนจากกลุ่มผู้พัฒนาอย่างต่อเนื่อง

4. การวิเคราะห์ความต้องการจากกรณีศึกษา

ในการวิจัยและพัฒนามีการศึกษาถึงบริบทของโรงพยาบาลเพื่อหาแนวทางการออกแบบและพัฒนาสถาปัตยกรรมแบบกระจายศูนย์เพื่อจัดการควบคุมการเข้าถึงข้อมูลสุขภาพสำหรับเครือโรงพยาบาลโดยใช้บล็อกเชนส่วนบุคคล ซึ่งได้ทำความร่วมมือกับเครือโรงพยาบาลเอกชนแห่งหนึ่งในประเทศไทยเพื่อรวบรวมข้อมูลความต้องการของระบบ โดยโรงพยาบาลมีความตั้งใจที่จะพัฒนาระบบเพื่อที่จะแลกเปลี่ยนข้อมูลสุขภาพของผู้ป่วยระหว่างโรงพยาบาลในเครือและหน่วยงานที่เกี่ยวข้อง

ในปัจจุบันการที่คนไข้จากโรงพยาบาลหนึ่ง ต้องเปลี่ยนไปรักษาที่โรงพยาบาลในเครืออีกแห่งหนึ่ง จะต้องนำเอกสารจากโรงพยาบาลแรกไปยังโรงพยาบาลที่สองด้วยตนเอง จึงค่อนข้างลำบากและล่าช้าเนื่องจากยังไม่มีระบบมารองรับ และจากเดิมที่ทางโรงพยาบาลมีระบบฐานข้อมูลแบบรวมศูนย์ ปริมาณข้อมูลที่เพิ่มขึ้นในทุกวันจึงเป็นเรื่องยากในการดูแลและขยายระบบเพราะค่าใช้จ่ายด้านโครงสร้าง (Infrastructure) จะสูงขึ้นตามไปด้วย

4.1 ขอบเขตของข้อมูลทดสอบเบื้องต้น

จากการประชุมเพื่อรวบรวมข้อมูลความต้องการของระบบกับทางเครือโรงพยาบาล โดยขอบเขตของระบบคือ ระบบที่สามารถทำการเก็บข้อมูลผู้ป่วยและข้อมูลธุรกรรมต่าง ๆ ไว้ในบล็อกเชน สามารถกำหนดสิทธิ์การเข้าถึงข้อมูลและทำการโอนย้ายข้อมูลผู้ป่วยระหว่างโรงพยาบาลในเครือ โดยข้อมูลดังกล่าวจะต้องได้รับการยินยอมสิทธิ์จากผู้ป่วยก่อนเสมอ เพื่อพิสูจน์แนวคิดเบื้องต้น ผู้พัฒนาได้ตกลงกับทางศูนย์วิจัยของโรงพยาบาลในการใช้ข้อมูลการตรวจสุขภาพของผู้ป่วย (check-up data) เป็นการทดลองเบื้องต้นในการทดสอบการจัดเก็บและแลกเปลี่ยนข้อมูล ซึ่งมีรูปแบบการจัดเก็บข้อมูลดังตารางที่ 2 ซึ่งข้อมูลที่ใช้ในการศึกษาเหล่านี้จะไม่สามารถระบุชื่อผู้ป่วยได้ (Unidentified)

ตารางที่ 2. รายละเอียดของข้อมูลทดสอบเบื้องต้น

Field Key	Description	Possible Value
HN	Hospital Number	MD5
Employee	พนักงาน	Direct Employee
EN	Episode Number	MD5
Year	ปี ค.ศ.	Number
Date	วันที่	Date
Nation	สัญชาติ	THAI
Sex	เพศ	Male, Female
Age	อายุ	Number
Location	สถานที่ทำการตรวจ	String
weight	น้ำหนัก	Number
height	ส่วนสูง	Number
BMI	ค่าBMI	Number
sysBP	ค่าความดันตัวบน	Number
diaBP	ค่าความดันตัวล่าง	Number
Glucose (Fasting)	ค่าน้ำตาลหลังอดอาหาร	Number
Glucose (POCT)	ค่าน้ำตาลPOCT	Number
HbA1C	ค่าน้ำตาลสะสม	Number
LDL	ค่าไขมันชนิดไม่ดี	Number
HDL	ค่าไขมันชนิดดี	Number
Cholesterol	ค่า คลอเลสเตอรอล	Number
Triglycerides	ค่าไตรกลีเซอไรด์	Number
Hypertension	ความดัน	Number
Diabetes	เบาหวาน	Number
Vascular Disease	โรคหลอดเลือดหัวใจ	Number
Smoking	การสูบบุหรี่	String
DOB	วันเดือนปีเกิด	Date
Waist circumference	เส้นรอบเอว	Number
Waist size	เส้นรอบเอว	Number
Hypertension	ผลโรคความดันโลหิตสูง	0, 1
Diabetes	ผล โรคเบาหวาน	0, 1

4.2 การกำหนดสิทธิ์การเข้าถึงข้อมูล

จากการวิเคราะห์ความต้องการของระบบพบว่าข้อมูลที่นำมาทดสอบเบื้องต้นแต่ละส่วนจะถูกควบคุมการกำหนดสิทธิ์เข้าถึงข้อมูลโดยแต่ละบุคคลจะมีสิทธิ์เข้าถึงข้อมูลต่างกันไป ดังตารางที่ 3.

*ตารางที่ 3. ตารางกำหนดสิทธิ์การเข้าถึงข้อมูล**

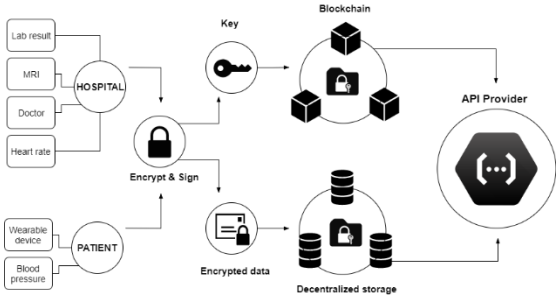
Data	Role					
	PC	N	PH	TC	LS	MRS
Lab result	CRUD	- R - -	- - - -	- R - -	- R U -	- R - -
Prescription	CRUD	- R - -	- R U -	- - - -	- - - -	- R - -
X-ray	CRUD	- R - -	- - - -	- R U -	- - - -	- R - -
Patient record	CRUD	CRUD	CRUD	CRUD	CRUD	- R - -

*PC แทน Physician; N แทน Nurse; PH แทน Pharmacist; TC แทน Technician; LS แทน Lab Staff; และ MRS แทน Medical Record Staff

กรณีศึกษานี้จะแบ่งผู้เกี่ยวข้องออกเป็น 6 บทบาท ได้แก่ แพทย์ (Physician) พยาบาล (Nurse) เภสัชกร (Pharmacist) นักเทคนิคการแพทย์ (Technician) เจ้าหน้าที่ตรวจผลการรักษาเชิงปฏิบัติ (Lab staff) และเจ้าหน้าที่ตรวจผลการรักษา (Medical Record staff) แบ่งประเภทของข้อมูลออกเป็น 4 ประเภท ได้แก่ ผลการรักษาเชิงปฏิบัติ (Lab result) ใบสั่งยา (Prescription) ผลภาพฉายรังสี (X-ray) และข้อมูลผู้ป่วย (Patient Record) แต่ละบทบาทสามารถเข้าถึงข้อมูลแต่ละประเภทได้แตกต่างกันออกไปตามสิทธิ์ของตนที่ถูกกำหนดไว้ตามตัวอักษร C R U และ D ซึ่งตัวอักษรเหล่านี้หมายถึง สิทธิ์ในการสร้างข้อมูล สิทธิ์ในการอ่านข้อมูล สิทธิ์ในการแก้ไขข้อมูล และสิทธิ์ในการลบหรือยกเลิกข้อมูลตามลำดับ เช่น พยาบาลสามารถอ่านข้อมูลผลการรักษาเชิงปฏิบัติ ใบสั่งยา ผลภาพฉายรังสี และสามารถสร้าง อ่าน แก้ไข และลบข้อมูลผู้ป่วยได้

5. ผลการออกแบบสถาปัตยกรรมของระบบ

จากการวิเคราะห์เทคโนโลยีบล็อกเชนและความต้องการของเครือโรงพยาบาลเป็นกรณีศึกษา ผู้พัฒนาได้สร้างแผนภาพแสดงรูปแบบสถาปัตยกรรม โดยออกแบบให้มีการเข้ารหัสลับข้อมูลเพื่อความปลอดภัยและความเป็นส่วนบุคคลของข้อมูลทางการแพทย์ นอกจากนี้ยังมีการจัดเก็บข้อมูลแบบกระจายศูนย์เพื่อป้องกันการลักลอบแก้ไขและเพิ่มความพร้อมใช้งานของข้อมูลซึ่งเป็นรูปแบบที่เหมาะสมต่อองค์กร



รูปที่ 5. ภาพรวมการทำงานเบื้องต้น

โดยระบบนี้จะทำการรับข้อมูลผู้ป่วยจากแหล่งต่าง ๆ ไม่ว่าจะมาจากอุปกรณ์สวมใส่ด้านสุขภาพหรือผลตรวจจากทางโรงพยาบาล โดยนำข้อมูลมาทำการเข้ารหัสลับและจัดเก็บแบบไม่รวมศูนย์ในเครือข่ายส่วนตัว (Private network) ที่เป็นการนำอีเธอเรียมบล็อกเชนมาใช้เนื่องจากเป็นบล็อกเชนแพลตฟอร์มที่มุ่งเน้นให้นำบล็อกเชนมาใช้กับอุตสาหกรรมอื่นนอกเหนือจากการเงิน เช่น ธุรกิจอสังหาริมทรัพย์ ธุรกิจประกันภัย รวมถึงด้านการแพทย์ โดยมีสมาร์ทคอนแทรคท์และ EVM เป็นหัวใจสำคัญซึ่งผู้พัฒนาต้องทำการปรับแต่งข้อกำหนดและการตั้งค่าต่าง ๆ ในไฟล์ตั้งต้น (Genesis file) ให้เข้ากับบริบทขององค์กร จากนั้นนำคีย์ในการเข้าถึงข้อมูลมาผูกกับบัญชีของผู้ป่วยในบล็อกเชน ซึ่งในส่วนนี้สามารถควบคุมสิทธิ์การเข้าถึงและติดตามการใช้งานข้อมูลได้ด้วยสมาร์ทคอนแทรคท์ ซึ่งผู้พัฒนาต้องการออกแบบให้ระบบสามารถเรียกใช้ได้โดยแอปพลิเคชันที่จะถูกใช้งานจากแผนกต่าง ๆ ของโรงพยาบาล และรองรับการเชื่อมต่อกับโรงพยาบาลอื่น ๆ ภายในเครือได้ในอนาคต จึงทำการออกแบบตามแนวคิด REST API [9] เนื่องจากสามารถทำงานบน HTTP โปรโตคอลและสนับสนุนรูปแบบข้อมูลแบบ JSON นอกจากนี้ยังวางตัวเป็นผู้ให้บริการ API สำหรับบล็อกเชนด้านข้อมูลทางการแพทย์ พร้อมกับจัดทำเครือข่ายบล็อกเชนส่วนตัว (Private Blockchain) ไว้สำหรับรับรองความถูกต้องของการทำรายการและเอาไว้จัดเก็บข้อมูล โดย API เบื้องต้นที่ได้จัดเตรียมไว้ประกอบไปด้วย การสร้างบัญชีผู้ใช้งาน การกำหนดสิทธิ์การเข้าถึงข้อมูล การเพิ่มข้อมูล การเข้ารหัสลับข้อมูล การเรียกดูข้อมูล และการแก้ไขข้อกำหนดการใช้ข้อมูล ทั้งนี้การพัฒนาแอปพลิเคชันแบบกระจายศูนย์ต้องใช้เวลาในการศึกษาและทดลองค่อนข้างมาก การใช้ API เหล่านี้จะช่วยร่นระยะเวลาในการพัฒนา เพื่อให้ผู้ที่ต้องการจะนำไปใช้ไม่จำเป็นต้องศึกษารายละเอียดการทำงานของบล็อกเชนหรือจัดทำเครือข่ายบล็อกเชนเอง แต่ยังสามารถใช้ความรู้พื้นฐานในการทำแอปพลิเคชันแล้วปฏิบัติตามแนวทางการใช้งานของแต่ละ API ที่ได้วางเอาไว้ และยังคงประโยชน์ของโครงสร้างแบบกระจายศูนย์ไว้ได้ทั้งหมด

5.1 Network Implementation

ดังที่ได้อธิบายไว้ในหัวข้อ 2.2.4 Ethereum platform มีรูปแบบการทำงานแบบไม่รวมศูนย์ โดยมีเน็ตเวิร์คโหนดที่ทำงานร่วมกันคือ Geth ซึ่งการทำงานร่วมกันของเน็ตเวิร์คโหนดนี้ นอกจากการรองรับภาษาโปรแกรมที่หลากหลายแล้วยังมีความสามารถด้านการประมวลผล เปรียบเสมือนเครื่องเซิร์ฟเวอร์ประสิทธิภาพสูงหนึ่งเครื่อง โดยมีชื่อเรียกว่า Ethereum Virtual Machine ในการสร้าง EVM ขึ้นมานั้น Geth จำเป็นต้องใช้ไฟล์ตั้งต้น (Genesis file) เดียวกันเพื่อทำงานบนข้อตกลงร่วมกัน Genesis file เปรียบเสมือนไฟล์ตั้งค่าสำหรับบล็อกเชน ที่อนุญาตให้ผู้พัฒนาสามารถนำไปสร้าง บล็อกเชนส่วนตัวได้เอง โดยผู้พัฒนาจำเป็นต้องกำหนดตัวแปรเบื้องต้น 4 ตัว คือ

1) Chain ID สำหรับบล็อกเชนส่วนตัวเพื่อป้องกันไม่ให้ผู้ไม่หวังดีทำการโจมตีด้วยวิธีการ Replay-attack ได้และทำการเลือกเวอร์ชันของ Ethereum protocol ที่ต้องการใช้

2) กำหนดความยากในการยืนยันความถูกต้องของธุรกรรม

3) กำหนดจำนวน Gas สูงสุดต่อบล็อก

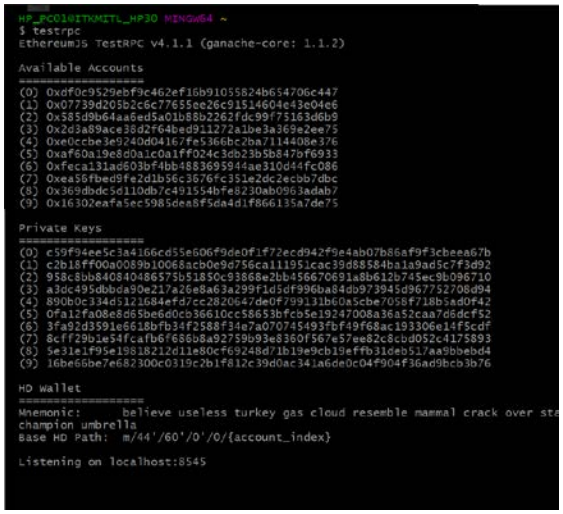
4) Address ตั้งต้นในการสร้าง Chain ใหม่

รูปที่ 6. ตัวอย่าง Genesis File

นอกจากนี้เรายังสามารถเพิ่มการตั้งค่าอื่น ๆ อย่างเช่น Coinbase, ExtraData, Nonce, Mixhash, parentHash และ Timestamp ลงใน Genesis file ได้อีกด้วย โดย Genesis file จะอยู่ในรูปของ JSON [10] เมื่อได้ Genesis file มาแล้ว โปรแกรม Geth จะมีคำสั่ง init ที่สามารถคอมไพล์ไฟล์ Genesis เพื่อสร้างบล็อกเชนส่วนตัวให้โดยอัตโนมัติ โดยจะมีการใช้พื้นที่บนเครื่องเพื่อทำการจัดเก็บข้อมูลธุรกรรมที่เกิดขึ้นเรียกว่าข้อมูล chain data และกุญแจเข้าใช้งานชื่อว่า keystore ที่เก็บข้อมูลบัญชีผู้ใช้อยู่ด้วย

จากนั้นเมื่อต้องการเพิ่มจำนวนเน็ตเวิร์คโหนดสามารถทำการใช้คำสั่ง $bootnode --genkey=boot.key เพื่อแสดงข้อมูล enode URL ซึ่งเปรียบเสมือนบ้านเลขที่ในการให้เน็ตเวิร์คโหนดอื่นเข้ามาติดต่อกับโหนดปัจจุบันได้ โดยภายในบล็อกเชนส่วนตัวเดียวกันจำเป็นต้องมีอย่างน้อย 1 โหนด ที่มีการทำงานอยู่ตลอดเวลา เพื่อประมวลผลและจัดเก็บข้อมูลธุรกรรมที่เกิดขึ้น

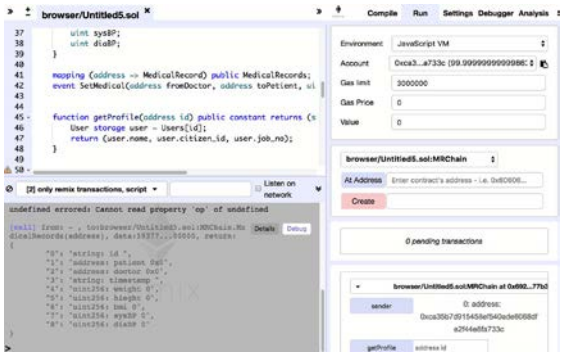
โดยเบื้องต้นแล้วในทางปฏิบัติผู้พัฒนาได้ทำการเลือกใช้ TestRPC ในการจำลองการทำงานของ EVM เพื่อความรวดเร็วในการทดสอบสมาร์ทคอนแทรคท์ และความง่ายในการติดตั้งเน็ตเวิร์คโหนด แต่ TestRPC มีข้อจำกัดในด้านการจำลองสภาพแวดล้อมการทำงานจริง เช่น ความรวดเร็วในการประมวลผลทรานแซคชัน รูปแบบการจัดเก็บไฟล์ keystore ที่ผู้ใช้ไม่สามารถกำหนด Passphrase ได้เอง การเพิ่มจำนวนเน็ตเวิร์คโหนด และการตั้งค่า Genesis file



รูปที่ 7. หน้าต่างการทำงานของ TestRPC

5.2 Smart Contract Implementation

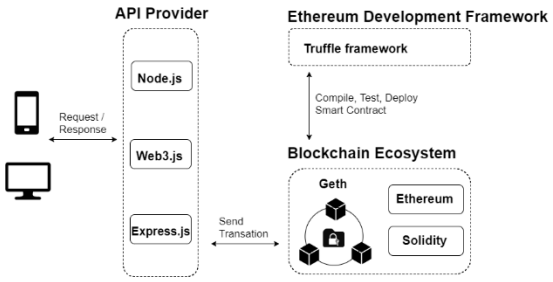
โครงสร้างระบบแบบอีเธอเรียมบล็อกเชนออกแบบมาให้ใช้สมาร์ทคอนแทรคท์เป็นตัวกำหนดบทบาทของผู้ใช้ภายในเครือข่ายบล็อกเชนเดียวกัน เสมือนเป็นข้อตกลงร่วมกันในเรื่องวิธีการทำงานของแอปพลิเคชันโดยผู้พัฒนาเลือกใช้ Remix IDE ดังรูปที่ 8. ในการเขียนสมาร์ทคอนแทรคท์เนื่องจากสามารถประมวลผลออนไลน์ได้ในทันทีและมีการจำลองสภาพแวดล้อมของเครือข่ายบล็อกเชนพร้อมกับประวัติการทำงานของคอนแทรคท์ให้เห็นถึงวิธีการในการสร้างธุรกรรมโดยบัญชีผู้ใช้ในระบบ



รูปที่ 8. Remix IDE

5.3 API provider Implementation

ในส่วนของการพัฒนา ผู้พัฒนาได้วางตัวเป็นผู้ให้บริการ API สำหรับบล็อกเชนด้านข้อมูลทางการแพทย์ พร้อมกับจัดทำเครือข่ายบล็อกเชนส่วนตัว (Private Blockchain) ไว้สำหรับรับรองความถูกต้องของการทำรายการและเอาไว้จัดเก็บข้อมูล ซึ่งได้เลือกใช้ Express Framework ในการเขียน API เนื่องจากมีส่วนจัดการแพ็คเกจเป็น Node.js เช่นเดียวกับ Web3 Library จากหัวข้อย่อยที่ 2.2.3 และ Truffle Framework จากหัวข้อย่อยที่ 3.2. ทำให้ง่ายต่อการเชื่อมต่อเข้ากับส่วนของสมาร์ทคอนแทรคท์ อีกทั้งมีจุดเด่นในเรื่องความเร็วในการตอบสนองและมีรูปแบบที่ง่ายต่อการใช้งาน



รูปที่ 9. รูปแบบสถาปัตยกรรมของระบบ

รูปที่ 9. คือส่วนของการออกแบบสถาปัตยกรรม และเครื่องมือที่ใช้งานจริง ส่วนหลักคือระบบนิเวศของบล็อกเชนและสมาร์ทคอนแทรคท์ โดยที่ใช้อีเธอเรียมเป็นเครือข่ายบล็อกเชนและมี Geth เป็นเน็ตเวิร์คโหนด มีชุดคำสั่งสมาร์ทคอนแทรคท์ที่เขียนโดยใช้ภาษา Solidity ซึ่งแปลงภาษาและติดตั้งบนเครือข่ายอีเธอเรียมโดย Truffle Framework จะเป็นตัวจัดการให้ทั้งหมด จากนั้นผู้ใช้งานภายนอกจะสามารถเรียกใช้งานสมาร์ทคอนแทรคท์ได้โดยเชื่อมต่อเข้ามาทาง API ที่ผู้พัฒนาออกแบบไว้ซึ่งจะสามารถนำไปพัฒนาต่อได้อย่างสะดวกและรวดเร็ว

6. บทสรุป

บทความนี้ได้นำเสนอการออกแบบและพัฒนาสถาปัตยกรรมแบบกระจายศูนย์เพื่อจัดการควบคุมการเข้าถึงข้อมูลสุขภาพสำหรับเครือโรงพยาบาล ซึ่งได้จัดเตรียมบล็อกเชนสำหรับข้อมูลด้านการแพทย์ไว้ให้บริการ (Blockchain-as-a-Service) เพื่อความสะดวกต่อในการที่แต่ละโรงพยาบาลสามารถนำบริการนี้ไปพัฒนาระบบใหม่หรือเชื่อมต่อเข้ากับระบบเดิม โดยได้เลือกใช้เทคโนโลยีที่มีศักยภาพอย่างบล็อกเชนในการพัฒนา ผลของการออกแบบเบื้องต้นสรุปได้ว่า เทคโนโลยีบลอคเชนสามารถสร้างความปลอดภัยที่สูง และสมาร์ทคอนแทรคช่วยให้สามารถกำหนดสิทธิ์การเข้าถึงข้อมูลได้อย่างอิสระ เหมาะอย่างยิ่งสำหรับข้อมูลด้านการแพทย์ แต่มีข้อจำกัดในเรื่องของความเร็วของจำนวนการทำธุรกรรมซึ่งต้องทำการปรับปรุงต่อไป และต้องการคอมพิวเตอร์ประสิทธิภาพสูงจำนวนมากในการยืนยันการทำธุรกรรม ซึ่งในอนาคตจำเป็นต้องทำการวัดผลประสิทธิภาพเทียบกับระบบรูปแบบเดิมในด้านความเร็ว ความปลอดภัย ค่าใช้จ่าย และ ประสิทธิภาพในการกู้คืนและแก้ไขระบบต่อไป

เอกสารอ้างอิง

[1] A. Ekblaw, A. Azaria, J. D. Halamka, MD, A. Lippman, “A Case Study for Blockchain in Healthcare: “MedRec” prototype for electronic health records and medical research data”, pp. 1 - 10, Aug. 2016.

[2] D. Palmer, “Blockchain Startup to Secure 1 Million e-Health Records in Estonia” [online], Retrieved from https://www.coindesk.com/blockchain-startup-aims-to-secure-1-million-estonian-health-records, Mar. 2016.

[3] M. Swan, “Blockchain: Blueprint for a New Economy”, O'Reilly Media, Inc., pp. ix – xi, 2015.

[4] S. Raval, “Decentralized Applications: Harnessing Bitcoin's Blockchain Technology”, O'Reilly Media, Inc., pp. 3 - 38, 2016.

[5] T. R. Peltier, “Information Security Fundamentals”, 2nd edition, pp. 29 – 42, 2014.

[6] V. Buterin , “Merkling in Ethereum” [online], Retrieved from https://blog.ethereum.org/2015/11/15/merkling-in-ethereum/, Dec. 2015

[7] V. Morabito, “Business Innovation Through Blockchain”, pp. 10 - 13, 2017

[8] V. Butein, “A Next Generation Smart Contract & Decentralized Application Platform” [online], Retrieved from https://github.com/ethereum/wiki/wiki/White-Paper, Dec. 2017

[9] M. Massé, “REST API Design Rulebook”, O'Reilly Media, Inc., pp. 5 - 33, 2011

[10] The go-ehereum Authors, “Go Ethereum” [online], Retrieved from https://github.com/ethereum/go-ethereum