บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 แบบจำลองวุฒิภาวะความสามารถบูรณาการ (Capability Maturity Model® Integration - CMMI®)

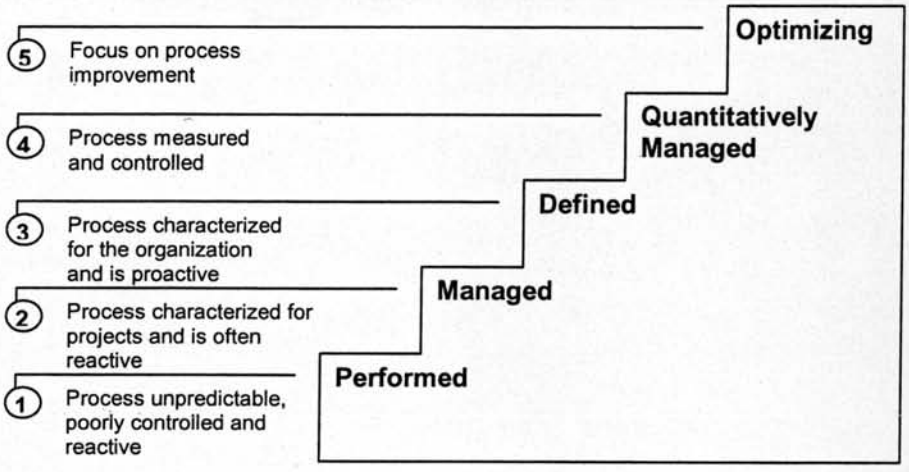
แบบจำลองวุฒิภาวะความสามารถบูรณาการ[1] เป็นกรอบงานสำหรับการปรับปรุงกระบวนการซอฟต์แวร์ที่คิดค้นโดยสถาบันวิศวกรรมซอฟต์แวร์ หรือ เอสอีไอ (software Engineering Institute - SEI) ในสหรัฐอเมริกาเพื่อมาแทนที่แบบจำลองวุฒิภาวะความสามารถ หรือ ซีเอ็มเอ็ม (Capability Maturity Model - CMM®) เนื่องจากแบบจำลองวุฒิภาวะความสามารถมีบางข้อจำกัดที่ทำให้ไม่สามารถใช้ร่วมกับแบบจำลองอื่นๆได้ ทางสถาบันวิศวกรรมซอฟต์แวร์จึงพัฒนาแบบจำลองวุฒิภาวะความสามารถบูรณาการขึ้นมาเพื่อแก้ปัญหาดังกล่าวโดยรวมเอาแบบจำลองต้นแบบ 3 แบบเข้าไว้ด้วยกันคือ

1. The Capability Maturity Model for Software (SW-CMM®) v.2.0 draft C
2. The System Engineering Capability Model (SECM)
3. The Integrated Product Development Capability Maturity Model (IPD-CMM) v.0.98

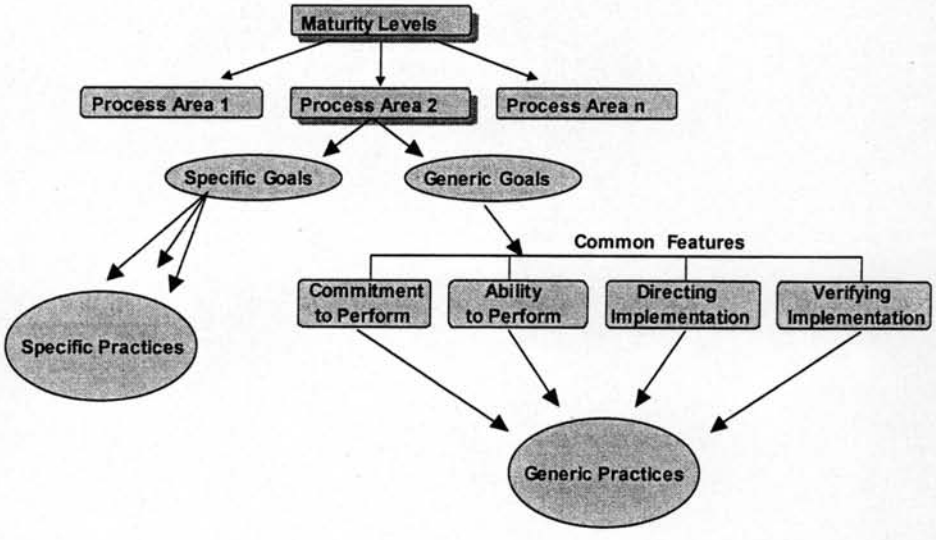
แบบจำลองวุฒิภาวะความสามารถบูรณาการช่วยในการจัดระดับวุฒิความสามารถขององค์กรและกำหนดแนวทางการปรับปรุงกระบวนการเพื่อก้าวไปสู่ระดับวุฒิความสามารถที่สูงขึ้น แบบจำลองวุฒิภาวะความสามารถบูรณาการมี 2 แบบให้องค์กรสามารถเลือกใช้ได้แล้วแต่เป้าหมายขององค์กร คือ

- **แบบขั้นบันได** (Staged Representation) [7] เป็นแนวทางปรับปรุงกระบวนการทีละขั้นซึ่งผลสำเร็จของแต่ละขั้นจะเป็นรากฐานสำหรับการปรับปรุงกระบวนการในขั้นถัดไป แต่ละขั้นหมายถึงระดับวุฒิภาวะซึ่งมี 5 ระดับ ดังรูปที่ 2.1 แต่ละระดับวุฒิภาวะประกอบด้วยกลุ่มกระบวนการที่ระบุไว้แน่นอน (Process Area - PA) ในแต่กลุ่มกระบวนการจะประกอบด้วยเป้าหมายเฉพาะ (Specific Goal - SP) และเป้าหมายทั่วไป (Generic Goal - GG) ทุกเป้าหมายจะมีแนวทางปฏิบัติสำหรับองค์กรเพื่อบรรลุผลสำเร็จตามเป้าหมาย โครงสร้างองค์ประกอบของแบบจำลองวุฒิภาวะความสามารถบูรณาการแบบขั้นบันไดเป็นดังรูปที่ 2. 1

แบบจำลองวุฒิภาวะความสามารถบูรณาการแบบขั้นบันไดเหมาะกับองค์กรที่คุ้นเคยกับแบบจำลองวุฒิภาวะความสามารถอยู่แล้วต้องการเปลี่ยนมาเป็นแบบจำลองวุฒิภาวะความสามารถบูรณาการ และองค์กรที่ไม่มีเป้าหมายว่าจะพัฒนากลุ่มกระบวนการใดเป็นหลัก



รูปที่ 2.1 ระดับวุฒิภาวะของแบบจำลองวุฒิภาวะความสามารถบูรณาการแบบขั้นบันได [7]



รูปที่ 2.2 องค์ประกอบของแบบจำลองวุฒิภาวะความสามารถบูรณาการแบบต่อเนื่อง [8]

- **แบบต่อเนื่อง** (Continuous Representation) [8] เป็นแนวทางปรับปรุงกระบวนการที่ยืดหยุ่นให้องค์กรสามารถเลือกปรับปรุงเพียงกลุ่มกระบวนการที่สอดคล้องกับวัตถุประสงค์ทางธุรกิจขององค์กรหรือเลือกกระบวนการที่เห็นว่าองค์กรยังบกพร่องอยู่ก็ได้ โดยจะวัดกระบวนการด้วยระดับความสามารถ 6 ระดับ คือ

  ระดับ 0 ไม่สมบูรณ์ (Incomplete)

ระดับ 1 ปฏิบัติ (Performed)

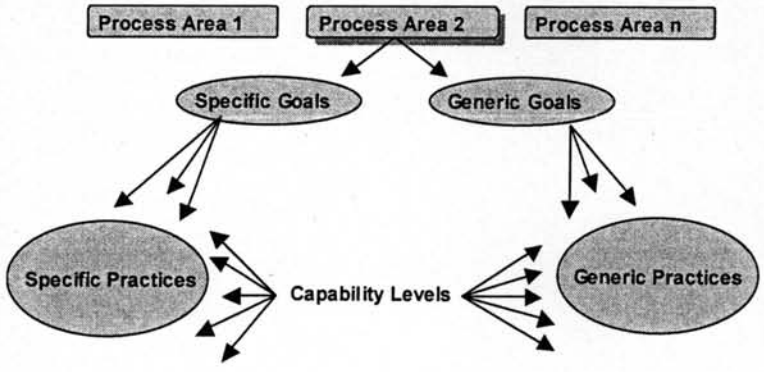
ระดับ 2 จัดการ (Managed)

ระดับ 3 จัดตั้ง (Established)

ระดับ 4 ทำนาย (Predictable)

ระดับ 5 เหมาะสม (Optimizing)

โดยรายละเอียดพื้นฐานของเป้าหมายและแนวทางปฏิบัติในการปรับปรุงกระบวนการแบบต่อเนื่องกับแบบขั้นบันไดไม่มีความแตกต่างกัน เพียงแต่การวางโครงสร้างนั้นต่างกัน โครงสร้างองค์ประกอบของแบบจำลองวุฒิภาวะความสามารถบูรณาการแบบต่อเนื่องเป็นดังรูปที่ 2.3



รูปที่ 2.3 องค์ประกอบของแบบจำลองวุฒิภาวะความสามารถบูรณาการแบบต่อเนื่อง [8]

การนำเสนอในรูปแบบต่อเนื่องนั้น จะมี

2.1.2 กระบวนการจัดการโครงแบบซอฟต์แวร์ (Software Configuration Management) โดย ANSI/IEEE Std 1042-1987 [9]

เป็นเอกสารแนะแนวการจัดการโครงแบบซอฟต์แวร์ (Software Configuration Management) ซึ่งสัมพันธ์กันกับ แผนจัดการโครงแบบ (Software Configuration Management Plans) [10] โดยเอกสารแนะแนวกระบวนการจัดการโครงแบบนี้ จะนำเสนอ 2 ส่วนด้วยกันคือ ส่วนของสิ่งที่ต้องนำมาพิจารณาขณะทำการวางแผนจัดการโครงแบบ และส่วนของการเตรียมและสร้างแผนจัดการโครงแบบ ซึ่งสิ่งที่จะถูกพิจารณาและจะถูกจัดการควบคุมภายใต้กระบวนการจัดการโครงแบบซึ่งสามารถแบ่งเป็น 3 ลำดับชั้น ดังนี้

1. คอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) เช่น เอกสารความต้องการ, โปรแกรม, รหัสต้นทาง (Source code)

2. ส่วนประกอบ (Component) เช่น Subsystems, Package, Software Tools

3. หน่วย (Unit) เช่น ฟังก์ชัน, โมดูล, routine

คณะควบคุมโครงแบบ (Change Control Boards) เป็นผู้ตัดสินใจที่จะใช้ระดับใดระดับหนึ่งที่ต้องการจะควบคุมการเปลี่ยนแปลง

จุดประสงค์ของการจัดการโครงแบบคือ การควบคุมผลผลิตต่างๆ ในโครงการเพื่อเพิ่มคุณภาพและลดความผิดพลาดของซอฟต์แวร์

การจัดการโครงแบบ จึงเป็นกิจกรรมที่ครอบคลุมตลอดช่วงกระบวนการผลิตซอฟต์แวร์ เนื่องจากความเปลี่ยนแปลงสามารถเกิดขึ้นเมื่อใดก็ได้ กิจกรรมการจัดการโครงแบบจึงถูกพัฒนาเพื่อ (1) ระบุการเปลี่ยนแปลง (2) ควบคุมการเปลี่ยนแปลง (3) สร้างความมั่นใจว่าการเปลี่ยนแปลงถูกกระทำอย่างเหมาะสม และ (4) การเปลี่ยนแปลงถูกรายงานไปยังผู้เกี่ยวข้อง

การจัดการโครงแบบ อาจมองในอีกมุมหนึ่งว่า คือ การทำกิจกรรมประกันคุณภาพของซอฟต์แวร์ (Software Quality Assurance) ที่ถูกประยุกต์ใช้ตลอดกระบวนการซอฟต์แวร์

การจัดการคอนฟิกูเรชันของซอฟต์แวร์แตกต่างจากการบำรุงรักษาซอฟต์แวร์ (Software Maintenance) อย่างชัดเจน การบำรุงรักษา คือ ชุดของกิจกรรมวิศวกรรมซอฟต์แวร์ที่เกิดขึ้นหลังจากซอฟต์แวร์ส่งมอบไปให้แก่ลูกค้าและถูกติดตั้งเพื่อปฏิบัติงานจริง แต่การจัดการคอนฟิกูเรชันของซอฟต์แวร์ คือ ชุดของกิจกรรมติดตามและควบคุม ที่เริ่มต้นขึ้นเมื่อโครงการซอฟต์แวร์เริ่ม และจะสิ้นสุดลงก็ต่อเมื่อซอฟต์แวร์พ้นการใช้งานแล้ว

หัวข้อสำคัญในกระบวนการจัดการโครงแบบสามารถอธิบายได้ดังนี้คือ

1. **คอนฟิกูเรชันไอเท็ม** (Configuration Item)

   ผลผลิตจากกระบวนการซอฟต์แวร์ คือข้อมูลข่าวสารที่อาจแบ่งหมวดหมู่ได้กว้างๆ เป็นสี่หมวดหมู่ได้แก่

   1.1 โปรแกรมคอมพิวเตอร์ ทั้งในรูปแบบของชุดคำสั่ง (Source Code) และโปรแกรมที่สามารถทำงานได้ (Executable File)

   1.2 เอกสารที่อธิบายโปรแกรม ทั้งที่มีเนื้อหาในระดับเทคนิค และระดับผู้ใช้

   1.3 ข้อมูลซึ่งบรรจุอยู่ในโปรแกรมและอยู่ภายนอกโปรแกรม

   1.4 เครื่องมือสำหรับพัฒนาซอฟต์แวร์

   ไอเท็ม (Item) ต่างๆ ที่กล่าวข้างต้น เมื่อรวมกันแล้วจะกลายเป็นส่วนสำคัญสำหรับกระบวนการซอฟต์แวร์ และถูกเรียกรวมๆ กันว่าเป็น คอนฟิกูเรชันไอเท็ม ซึ่งก็คือ สิ่งที่มีความสำคัญต่อโครงการในขณะใดขณะหนึ่ง หรือตลอดทั้งระยะของโครงการ

คอนฟิกูเรชันไอเท็ม จำเป็นต้องถูกระบุและแจ้งไว้ในแผลการทำงาน เพื่อเป็นที่สังเกตในขณะที่โครงการดำเนินไป และเป็นเป้าหมายของการจัดการคอนฟิกูเรชัน คอนฟิกูเรชันไอเท็มที่ควรมีในโครงการสามารถแสดงได้ดังต่อไปนี้

1. ข้อกำหนดของระบบ (System Specification)
2. แผนโครงการซอฟต์แวร์ (Software Project Plan)
3. ข้อกำหนดความต้องการของซอฟต์แวร์ (Software Requirements Specification)
    a. ต้นแบบภาพเพื่อการวิเคราะห์ (Graphical Analysis Models)
    b. ข้อกำหนดการประมวลผล (Process Specification)
    c. งานต้นแบบ (Prototype)
    d. ข้อกำหนดทางการคำนวณหรือคณิตศาสตร์ (Mathematical Specification)
4. คู่มือผู้ใช้เบื้องต้น (Preliminary Manual)
5. ข้อกำหนดการออกแบบ (Design Specification)
    a. คำอธิบายการออกแบบข้อมูล (Data Design Description)
    b. คำอธิบายการออกแบบสถาปัตยกรรม (Architecture Design Description)
    c. คำอธิบายการออกแบบส่วนประสาน (Interface Design Description)
    d. คำอธิบายออบเจ็คต์ (Object description) ถ้าใช้เทคนิคการออกแบบเชิงวัตถุ (Object-Oriented Design)
6. รายการชุดคำสั่ง (Source Code Listing)
7. ข้อกำหนดสำหรับการทดสอบ (Test Specification)
    a. แผนการทดสอบและขั้นตอน (Test Plan and Procedure)
    b. กรณีการทดสอบและผลที่ถูกบันทึก (Test Cases and Recorded Results)
8. คู่มือการติดตั้งและปฏิบัติงาน (Operation and Installation Manuals)
9. โปรแกรมที่ทำงานได้ (Executable Program)
    a. ชุดคำสั่งโมดูลที่ทำงานได้ (Module Executable Code)
    b. โมดูลเพื่อลิงค์ (Linked Module)
10. คำอธิบายฐานข้อมูล

a. โครงสร้างไฟล์และแนวคิด (Schema and File Structure)

b. ข้อมูลเบื้องต้น (Initial Content)

11. คู่มือผู้ใช้เมื่อสร้าง (As-built User Manual)

12. เอกสารการบำรุงรักษา (Maintenance Manual)

a. รายงานปัญหาซอฟต์แวร์ (Software Problem Report)

b. คำร้องขอการซ่อมบำรุง (Maintenance Request)

c. คำสั่งเปลี่ยนแปลงทางวิศวกรรม (Engineering Change Orders)

13. มาตรฐานและขั้นตอนปฏิบัติ (Proceure) สำหรับวิศวกรรมซอฟต์แวร์ นอกเหนือจากคอนฟิกูเรชันไอเท็มข้างต้น องค์กรที่ปฏิบัติวิศวกรรมซอฟต์แวร์บางแห่งเพิ่มเติมเครื่องมือซอฟต์แวร์ไว้ภายใต้การควบคุมคอนฟิกูเรชันดังที่เคยกล่าวไว้ข้างต้น ซึ่งจะเป็นการระบุเวอร์ชัน (Version) ของโปรแกรมบรรณาธิกรณ์ (editor) ตัวแปลโปรแกรม (Compiler) และเครื่องมือ CASE อื่นๆ ไว้ควบคุมกับคอนฟิกูเรชันอื่น

2. เบสไลน์ (Baseline)

เบสไลน์ (Baseline) เป็นแนวคิดการจัดการคอนฟิกูเรชันของซอฟต์แวร์ที่ช่วยให้การควบคุมความเปลี่ยนแปลงสามารถกระทำได้โดยปราศจากผลกระทบที่ร้ายแรงต่อส่วนที่ไม่เปลี่ยนแปลง มารฐาน IEEE [10] ให้คำจำกัดความแก่เบสไลน์ไว้ว่า

"ข้อกำหนดหรือผลผลิตที่ผ่านการทบทวนและได้รับความเห็นชอบแล้วอย่างเป็นทางการ ดังนั้นมันจึงใช้เป็นฐานของการพัฒนาต่อไปได้ การเปลี่ยนแปลงต่อข้อกำหนดหรือผลผลิตนั้นจะสามารถกระทำได้โดยผ่านขั้นตอนควบคุมการเปลี่ยนแปลงที่เป็นทางการเท่านั้น"

ในแนวคิดของวิศวกรรมซอฟต์แวร์ เบสไลน์เป็นไมล์สโตน (Milestone) ของการพัฒนาซอฟต์แวร์ ที่ใช้เพื่อทำเครื่องหมายไว้สำหรับการส่งมอบคอนฟิกูเรชันไอเท็มหนึ่งหรือหลายชิ้น และการอนุมัติคอนฟิกูเรชันไอเท็มเหล่านี้จะกระทำได้ภายหลังจากที่คอนฟิกูเรชันไอเท็มผ่านการทบทวนอย่างเป็นทางการแล้ว ยกตัวอย่างเช่น เมื่อเอกสารข้อกำหนดการออกแบบและได้รับการทบทวน และแก้ไขข้อผิดพลาดที่พบจากการทบทวนแล้วเอกสารจะได้รับการอนุมัติและจึงจัดให้เอกสารข้อกำหนดการออกแบบเป็นเบสไลน์ จากนั้นการเปลี่ยนแปลงใดที่จะเกิดขึ้นกับ

สถาปัตยกรรมของซอฟต์แวร์ซึ่งถูกระบุไว้เอกสารข้อกำหนดการออกแบบ จะต้องถูกกระทำโดยผ่านขั้นตอนการร้องขอการเปลี่ยนแปลง และถูกประเมิน เวลา ผลกระทบ และท้ายสุดคือได้รับอนุมัติเท่านั้น จึงจะสามารถแก้ไขเอกสารและนำเข้าโครงการได้

เบสไลน์สามารถกำหนดไว้ได้ที่หลายระดับขึ้นอยู่กับความละเอียดของการควบคุม ซอฟต์แวร์เบสไลน์ส่วนใหญ่สามารถได้ดังนี้

ก. Functional Baseline (FBL) จะถูกสร้างขึ้นหลังจากการทวนสอบความต้องการ (Requirement Review) และส่งมอบเอกสารความต้องการไปยังผู้จัดการโครงแบบ

ข. Allocated Baseline (ABL) จะถูกสร้างขึ้นท้ายสุดของระยะการออกแบบ โดยผู้จัดการโครงแบบจะตรวจสอบเอกสารที่ต้องการทั้งหมดรวมทั้งซีไอได้ถูกสร้างขึ้น โดยซีไอที่อยู่ในส่วนของ allocated baseline จะรวมถึงการออกแบบระบบเบื้องต้น, รายละเอียดการออกแบบ และส่วนที่เกี่ยวข้องกับแผนการทดสอบ

ค. Product Baseline (PBL) จะถูกสร้างขึ้นสำหรับแต่ละครั้งที่ปล่อยระบบหลังจากผ่านระยะการทดสอบการยอมรับ (Acceptance testing phase) ผู้จัดการโครงแบบตรวจสอบว่าการทดสอบว่าได้ทำการทดสอบตรงตามเอกสารเบสไลน์ ซึ่งเบสไลน์นี้ได้รวมซีไอ, รายงานแผนการทดสอบระบบ, รายงานแผนการทดสอบการยอมรับ

ง. Production Baseline จะถูกสร้างขึ้นท้ายสุดของแต่ละการปล่อย (Release) หรือเมื่อมีการเปลี่ยนแปลงครั้งใหญ่ที่เบสไลน์ โดยจะถูกสร้างขึ้นที่ท้ายสุดของการทดสอบการยอมรับของแต่ละการปล่อย ซึ่งจะมีซีไอ รายงานแผนการทดสอบระบบ, รายงานแผนการทดสอบการยอมรับ, คู่มือ และเอกสารแผนงานอื่นๆ

การสร้างคอนฟิกูเรชันไอเท็มให้เป็นเบสไลน์ หรือการสร้างเบสไลน์ เกิดขึ้นเมื่อคอนฟิกูเรชันชิ้นหนึ่งชิ้นใดหรือหลายชิ้นผ่านการทบทวน และต้องได้รับการอนุมัติ จากนั้นจึงนำชุดของคอนฟิกูเรชันไอเท็มนั้นไปเก็บไว้ในฐานข้อมูลของโครงการ (Project Database) หรือ ไลบรารีของโครงการ (Project Library) หรือ ที่รวบรวมซอฟต์แวร์ (Software Repository)

และเนื่องจากระหว่างกำหนดการสร้างเบสไลน์แต่ละครั้ง อาจมีคอนฟิกูเรชันไอเท็มที่ตามจุดประสงค์ของเบสไลน์ได้รับการทบทวนและอนุมัติแล้ว เช่น ถ้าก่อนถึงกำหนดการสร้างเบสไลน์เพื่อการทดสอบ แผนการทดสอบได้รับการอนุมัติเป็นต้น คอนฟิกูเรชันไอเท็มนั้นสามารถถูกจัดเก็บในฐานข้อมูลของโครงการได้ทันที เพื่อการควบคุมการเปลี่ยนแปลงที่มีประสิทธิภาพและรวดเร็ว

ดังนั้น ในทางปฏิบัติ การสร้างเบสไลน์ตามกำหนดการ จึงอาจเป็นการรวบรวมรายการของเบสไลน์หรือคอนฟิกูเรชันไอเท็ม เพื่อทบทวนและตรวจสอบว่าคอนฟิกูเรชันไอเท็มตามจุดประสงค์ของเบสไลน์ทั้งหมดนั้น ได้รับการอนุมัติแล้วจัดเก็บในฐานข้อมูลของโครงการอย่างถูกต้องและครบถ้วนหรือไม่

3. **การควบคุมเวอร์ชัน** (Version Control)

การควบคุมเวอร์ชัน ประกอบด้วยขั้นตอนปฏิบัติและเครื่องมือต่างๆ เพื่อจัดการเวอร์ชันที่หลากหลายของวัตถุคอนฟิกุเรชัน (Configuration objects) ที่ถูกสร้างขึ้นระหว่างกระบวนการของวิศวกรรมซอฟต์แวร์

4. **การควบคุมการเปลี่ยนแปลง** (Change Control)

สำหรับโครงการพัฒนาซอฟต์แวร์ขนาดใหญ่ ความเปลี่ยนแปลงที่เกิดขึ้นอย่างรวดเร็วและไร้การควบคุมก่อให้เกิดความยุ่งยาก การควบคุมความเปลี่ยนแปลง (Change Control) จึงเป็นสิ่งจำเป็น การควบคุมความเปลี่ยนแปลง ซึ่งสามารถดูได้จากรูป 4.3 เป็นผลจากการประสานกันของขั้นตอนปฏิบัติงานโดยมนุษย์ และการใช้เครื่องมืออัตโนมัติเพื่อสนับสนุนกลไกควบคุมความเปลี่ยนแปลง

4.1 การควบคุมการเปลี่ยนแปลงคอนฟิกูเรชันไอเท็มที่ได้รับการอนุมัติแล้ว

เมื่อคอนฟิกูเรชันไอเท็มผ่านการทบทวนทางเทคนิคอย่างเป็นทางการและได้รับการอนุมัติ ไอเท็มนั้นจะกลายเป็นเบสไลน์ของโครงการ การเปลี่ยนแปลงคอนฟิกูเรชันไอเท็มจะต้องกระทำอย่างเป็นทางการ หรือเรียกว่า การควบคุมการเปลี่ยนแปลงระดับโครงการ (Project Level Change Control) คือ นักพัฒนาจะต้องได้รับอนุมัติจากซีซีบีของโครงการ (CCB) ก่อนทำการเปลี่ยนแปลง ถ้าการเปลี่ยนแปลงนั้นไม่กระทบไอเท็มอื่น หรือได้รับอนุมัติจากหัวหน้าโครงการ (PM) ในกรณีเร่งด่วน ไม่สามารถรอการอนุมัติจาก CCB ก่อนได้

4.2 การควบคุมการเปลี่ยนแปลงคอนฟิกูเรชันที่ยังไม่ได้รับการอนุมัติ

ก่อนที่คอนฟิกูเรชันไอเท็มจะเป็นเบสไลน์ หรือก่อนที่คอนฟิกูเรชันไอเท็มจะได้รับการอนุมัติ การควบคุมการเปลี่ยนแปลง จะเป็นแบบ การควบคุมการเปลี่ยนแปลงที่ไม่เป็นทางการ (Informal Change Control) นักพัฒนาคอนฟิกูเรชันไอเท็มสามารถเปลี่ยนแปลงแก้ไขไอเท็มของตนเมื่อใดก็ได้ ตามต้องการของโครงการและความต้องการทางเทคนิค แต่ความเปลี่ยนแปลงจะต้องไม่กระทบต่อความต้องการของระบบในวงกว้างซึ่งอยู่นอกเหนือความรับผิดชอบของนักพัฒนาผู้นั้น

4.3 การควบคุมการเปลี่ยนแปลงภายหลังปล่อยซอฟต์แวร์

เมื่อผลิตภัณฑ์ซอฟต์แวร์ถูกปล่อยไปยังลูกค้าแล้ว การเปลี่ยนแปลงจะต้องกระทำภายใต้ การควบคุมการเปลี่ยนแปลงอย่างเป็นทางการ (Formal Change Control) ขั้นตอนควบคุมการเปลี่ยนแปลงอย่างเป็นทางการ ซึ่งก็คือการควบคุมการเปลี่ยนแปลงคอนฟิกูเรชันไอเท็มที่ได้รับการอนุมัติแล้ว

5. **การตรวจสอบคอนฟิกูเรชัน** (Configuration Audit)

การระบุคอนฟิกูเรชันไอเท็ม การควบคุมเวอร์ชัน และการควบคุมการเปลี่ยนแปลง ช่วยให้นักพัฒนาซอฟต์แวร์สามารถรักษาลำดับการทำงานได้ และไม่เกิดความยุ่งเหยิงขึ้น อย่างไรก็ตาม แม้ตามกลไกการควบคุมที่ประสบความสำเร็จก็ยังติดตามความเปลี่ยนแปลงได้จนกระทั่งสร้างคำสั่งการเปลี่ยนแปลงทางวิศวกรรม ดังนั้นเราจึงต้องสร้างความมั่นใจว่าการเปลี่ยนแปลงถูกกระทำอย่างถูกต้อง ซึ่งสามารถทำได้สองอย่าง คือ (1) การทวนสอบทางเทคนิคอย่างเป็นทางการ (Formal Technical Review) และ (2) การตรวจสอบคอนฟิกูเรชันของซอฟต์แวร์ (Software Configuration Audits)

การทบทวนทางเทคนิคอย่างเป็นทางการ มุ่งเน้นความถูกต้องทางเทคนิคของคอนฟิกูเรชันที่ถูกแก้ไข ผู้ทบทวนจะประเมินคอนฟิกูเรชันไอเท็มเพื่อพิจารณาความถูกต้องสอดคล้องกับคอนฟิกูเรชันไอเท็มอื่น และผลกระทบข้างเคียงที่เป็นไปได้ การทบทวนทางเทคนิคอย่างเป็นทางการถูกจัดขึ้น สำหรับทุกการเปลี่ยนแปลง

การตรวจสอบคอนฟิกูเรชันของซอฟต์แวร์ เป็นการทบทวนทางเทคนิคอย่างเป็นทางการโดยประเมินคอนฟิกูเรชันสำหรับลักษณะเฉพาะที่มักจะไม่ถูก

พิจารณาระหว่างการทบทวนโดยทั่วไป การตรวจสอบเป็นการถามและตอบคำถามดังต่อไปนี้

1. กระทำการเปลี่ยนแปลงที่ระบุในคำสั่งการเปลี่ยนแปลงทางวิศวกรรมหรือไม่? การเปลี่ยนแปลงเพิ่มเติมถูกรวมเข้าไปหรือยัง
2. มีการทบทวนทางเทคนิคอย่างเป็นทางการเพื่อประเมินความถูกต้องทางเทคนิคหรือไม่?
3. ปฏิบัติตามมาตรฐานของวิศวกรรมซอฟต์แวร์หรือไม่?
4. เน้นความเปลี่ยนแปลงที่เกิดขึ้นในคอนฟิกูเรชันไอเท็มให้เห็นได้ชัดเจนหรือไม่? มีการกำหนดวันและผู้ทำการเปลี่ยนแปลงหรือไม่? คุณสมบัติของคอนฟิกูเรชันแสดงการเปลี่ยนแปลงที่เกิดขึ้นหรือไม่?
5. ปฏิบัติตามขั้นตอนซีเอ็มเพื่อบันทึกการเปลี่ยนแปลง การเก็บหมายเหตุ และการรายงานการเปลี่ยนแปลงหรือไม่?
6. ปรับปรุงคอนฟิกูเรชันไอเท็มที่เกี่ยวข้องอย่างเหมาะสมหรือไม่?

ในบางกรณี คำถามเพื่อการตรวจสอบอาจถูกถามเป็นส่วนหนึ่งของการทบทวนทางเทคนิคอย่างเป็นทางการ และเมื่อซีเอ็มเป็นกิจกรรมที่เป็นทางการ การตรวจสอบซีเอ็มจะต้องกระทำแยกต่างหากจากการทบทวนโดยกลุ่มประกันคุณภาพ (Quality Assurance Group)

6. **รายงานสถานะ** (Status Report)

การรายงานสถานะของคอนฟิกูเรชัน (Configuration Status Reporting หรือ SCR) หรือบางครั้งเรียกว่า การทำรายการสถานะ (Status Accouting) เป็นงานซีเอ็มที่ตอบคำถามต่อไปนี้

1. เกิดอะไรขึ้น
2. ใครเป็นผู้ทำ
3. เกิดขึ้นเมื่อไหร่
4. มีผลกระทบอะไรบ้าง

ทุกครั้งที่มีการแจ้งคอนฟิกูเรชันไอเท็มใหม่หรือเปลี่ยนแปลง ทุกครั้งที่ผู้มีอำนาจควบคุมการเปลี่ยนแปลงหรือซีซีบีอนุมัติการเปลี่ยนแปลง (และผลิตคำสั่งการเปลี่ยนแปลงทางวิศวกรรม) และทุกครั้งที่คอนฟิกูเรชันถูกตรวจสอบ ผลของการเปลี่ยนแปลงและการตรวจสอบจะต้องปรากฏในรายงานสถานะของคอนฟิกูเรชัน ผลจากการรายงานสถานะของคอนฟิกูเรชันอาจถูกเก็บบันทึกในฐานข้อมูล ดังนั้น

นักพัฒนาซอฟต์แวร์หรือผู้ทำหน้าที่บำรุงรักษาซอฟต์แวร์จะต้องสามารถเข้าถึงข้อมูลที่เปลี่ยนแปลงได้ นอกจากนี้ รายงานของการรายงานสถานะคอนฟิกูเรชัน (SCR Report) ต้องถูกสร้างเป็นประจำ เพื่อที่จะส่งมอบข้อมูลสำคัญแก่ฝ่ายบริหารและผู้ปฏิบัติงาน

การรายงานสถานะของคอนฟิกูเรชันมีบทบาทสำคัญต่อความสำเร็จของโครงการพัฒนาซอฟต์แวร์ขนาดใหญ่ เพื่อให้ทุกคนทุกฝ่ายรู้ข้อมูลเท่าเทียมกัน เป็นการป้องกันปัญหาหลายประการ ยกตัวอย่างเช่น นักพัฒนาสองคนอาจพยายามเปลี่ยนแปลงคอนฟิกูเรชันไอเท็มเดียวกันด้วยจุดประสงค์ที่แตกต่างและขัดแย้งกัน

2.1.3 คำอธิบายตัวชี้บอกการปฏิบัติกระบวนการ (Process Implementation Indicator Descriptions – PIID)

วิธีประเมินที่เป็นมาตรฐานของซีเอ็มเอ็มไอสำหรับการปรับปรุงกระบวนการ หรือ สแคมพี (Standard CMMI® Appraisal Method for Process Improvement – SCAMPI) [10] เป็นส่วนหนึ่งของชุดผลิตภัณฑ์ซีเอ็มเอ็มไอ (CMMI® Product Suite) ซึ่งใช้เป็นเครื่องมือในการประเมินระดับความสามารถและระดับวุฒิภาวะขององค์กรซึ่งมีการปรับปรุงกระบวนการด้วยแบบจำลองวุฒิภาวะความสามารถบูรณาการ นอกเหนือจากวัตถุประสงค์หลักในการเป็นเครื่องมือสำหรับการประเมินกระบวนการแล้ว สแคมพียังสามารถถูกนำมาใช้ในแง่ของการระบุจุดแข็งและจุดอ่อนของกระบวนการภายในองค์กร แสดงถึงความเสี่ยงต่างๆที่สำคัญ และช่วยในการจัดลำดับความสำคัญของการปรับปรุงกระบวนการได้อีกด้วย

วิธีการประเมินแบบสแคมพีมีพื้นฐานอยู่บนการทวนสอบ (Verification) ข้อมูล ดังนั้น องค์กรที่ถูกประเมินจึงต้องจัดเตรียมข้อมูลและเอกสารเป็นจำนวนมากเพื่อให้ทีมประเมินทำการทวนสอบ และสิ่งที่เป็นตัวชี้บอกว่าองค์กรควรจะต้องจัดเตรียมข้อมูลใดบ้างเพื่อนำมาใช้ในการประเมินระดับความสามารถหรือระดับวุฒิภาวะของซีเอ็มเอ็มไอคือ คำอธิบายตัวชี้บอกการปฏิบัติกระบวนการหรือพีไอไอดี (Process Implementation Indicator Descriptions – PIID)

แนวความคิดพื้นฐานของพีไอไอดี คือ เป็นเครื่องมือที่ช่วยให้องค์กรรับทราบว่าการปฏิบัติกระบวนการได้ดำเนินการไปอย่างไรในโครงการหนึ่งๆ หรือ ภายในองค์กรทั้งหมดโดยรวม โดยพีไอไอดีประกอบด้วยรายการของหลักฐานที่สามารถใช้แสดงได้ว่ามีการปฏิบัติตามวิธีปฏิบัติที่กำหนดไว้จริง ในพีไอไอดี มีการระบุเป้าหมายและวิธีปฏิบัติไว้ทีละข้อ โดยในแต่ละข้อ มีการระบุสิ่งที่สร้าง

ทางตรง (Direct Artifact) สิ่งที่สร้างทางอ้อม (Indirect Artifact) และการยืนยัน (Affirmation) ดังตารางที่ 2.1

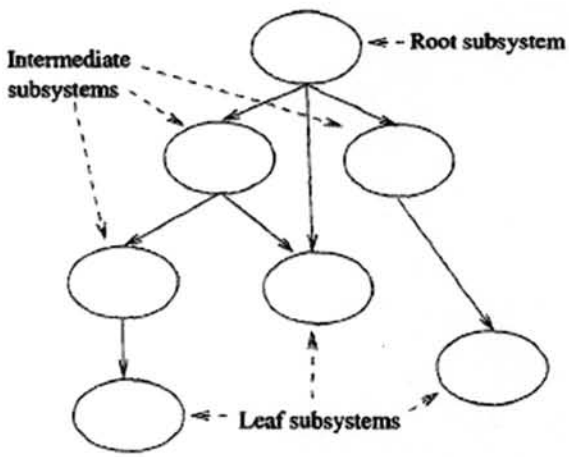
ตารางที่ 2.1 ตัวอย่างพีไอไอดีของกลุ่มกระบวนการจัดการโครงแบบ

Goal ID	CM SG 1 Baselines of identified work products are established.		
Practice ID	CM SP 1.1-1 *Identify the configuration items, components and related work products that will be placed under configuration management.*		
PII Type	Direct Artifacts	Indirect Artifacts	Affirmations / Corrections
Example Evidence (Look Fors / Listen Fors)	[1. Identified configuration items] • Configuration management lifecycle for controlled items (e.g., owner, point at which placed under control, degree of control, change approval.)	• Configuration management plan. • Configuration item identifiers, attributes and characteristics. • Documented criteria for selecting configuration items	-
Appraisal Considerations	• Be sure to consider configuration items representative of all disciplines and processes within the appraisal scope and context. In a sense, this SP specifies the constraints under which the remaining SPs should be considered and assessed. • See model for definition and description of configuration item and its work product components. • See model for typical examples of work products that may be part of a configuration item (e.g. process descriptions, requirements, design, tools) • See model overview material for GP2.6 for a description of the various levels of control that might be provided across the lifecycle, e.g. version control vs. formal configuration management. • "This process area applies not only to configuration management on projects, but also configuration management on organization work products such as standards, procedures, and reuse libraries." • Recall that this PA supports configuration management needs of all other process areas, as invoked by GP2.6		

2.2 งานวิจัยที่เกี่ยวข้อง

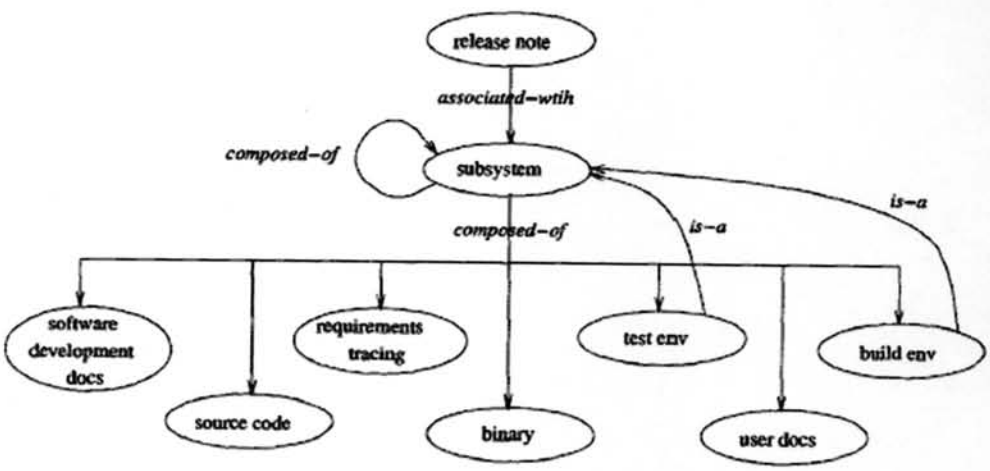
2.2.1 A Framework for Subsystem-based Configuration Management [11]

งานวิจัยนี้นำเสนอกรอบงาน (Framework) ซึ่งถูกพัฒนาขึ้นให้เหมาะกับเครื่องมือของการจัดการโครงแบบที่มีอยู่ [2,3,4,5] และสนับสนุน การจัดการโครงแบบ ของ subsystems โดยมีโครงสร้างเป็นลำดับชั้น (hierarchical) (รูปที่ 2.4) และมีลักษณะของ subsystem configuration items (CIs) ซึ่งใน subsystems จะเรียกว่า constituent configuration items หรือเรียกแบบสั้นๆ ว่า constituents



รูปที่ 2.4 โครงสร้างลำดับชั้นการจัดการโครงแบบของ subsystems [11]

กรอบงานที่นำเสนอสนับสนุน subsystems ภายใต้ configuration และ version Control โดย subsystems จะหมายถึง collections ของ software development artifacts ซึ่งประกอบด้วย รหัสต้นทาง เอกสาร และชุดแบบทดสอบ และ subsystem ดังรูปที่ 2.5



รูปที่ 2.5 ความสัมพันธ์ของ entity ใน subsystem [11]

- software development docs จะเป็นเอกสารจำพวก requirements, specification และ architecture/design documents ซึ่งจะพบใน root ของ subsystem
- รหัสต้นทาง (Source files) จะพบใน leaf subsystems ซึ่งอาจมีหรือไม่มีก็ได้
- Binaries ในส่วนของ root จะเป็นพวก execute file แต่ถ้าเป็น leaf จะเป็นพวก library
- เอกสารผู้ใช้งาน (User documents) เป็นเอกสารที่ควรจะอยู่ subsystem ที่มีการตอบสนองกับผู้ใช้ ซึ่งจะอยู่ใน

  Root subsystem แต่ก็อาจจะนำไปไว้กับ subsystem ที่ต้องการช่วยให้ผู้ใช้เข้าใจขึ้น
- Requirement tracing เป็นการติดตามความต้องการกับ constituents ของ subsystem
- Subsystems ซึ่งสามารถอยู่ใน subsystem ได้
- Test environment เป็นแบบทดสอบ ประกอบด้วย test plans, test drivers, test stubs, test case และ test result
- Build environment เป็นเอกสารเช่น Makefiles และ เอกสารที่เกี่ยวข้องกับการ build จำพวก compilers

ในแต่ละ subsystem จะบรรจุคอนฟิกูเรชันไอเท็ม หรือ ซีไอ โดยมีรายละเอียด ซึ่งเรียกว่า ข้อกำหนดระบบย่อย หรือ เอสซีเอส (Subsystem Configuration Specification – SCS) ดังตัวอย่างต่อไปนี้

Name: DNPMaster
Version: v1
Description: Master Unit part of DNP protocol.

Constituents	Version	Location
src-code1	v1	SrcCode
src-code2	v1	SrcCode
src-code3	v1	SrcCode
Makefile	v1	BuildEnv
protocol-spec	v4	Documentation
dnp-design	v3	Documentation
DnpTest	v1	TestEnv

รูปที่ 2.6 SCS สำหรับ DNPMaster [6]

งานวิจัยนี้นำเสนอโดยใช้ ชื่อของ subsystems และ คอนฟิกูเรชัน ของ Foxboro software เช่น จาก รูปที่ 2.6 SCS สำหรับ DNPMaster เป็นตัวอย่างของข้อกำหนดระบบย่อย ซึ่ง

จะบรรจุคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) โดยมี 3 source code , Makefile, protocol specification, design document และ test subsystem

เวอร์ชันที่ต่างกันระหว่าง 2 subsystem อาจจะบรรจุด้วยไอเท็มที่เหมือนหรือคนละเวอร์ชันกันดังรูปที่ 2.7 จากทั้งสองเวอร์ชันของ RTU+DNP นั้นจะมี DNPMaster และ DNPSlave ที่ต่างเวอร์ชันกัน ซึ่งคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item – CI) ทั้งหมดจะถูกเก็บในฐานข้อมูลการจัดการโครงแบบ (CM Database) อยู่แล้ว และงานวิจัยนี้จะเก็บในลักษณะที่สถานที่เก็บของ คอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) ซึ่งหลังจากมีการเปลี่ยนเวอร์ชันก็จะมีรายละเอียดการเปลี่ยนแปลงดังรูปที่ 2.8

Name: RTU+DNP
Version: v1
Description: Remote Terminal Unit using DNP protocol

Constituents	Version	Location
Core	v1	SrcCode
DNPMaster	v1	SrcCode
DNPSlave	v1	SrcCode

Name: RTU+DNP
Version: v2
Description: Remote Terminal Unit using DNP protocol

Constituents	Version	Location
Core	v1	SrcCode
DNPMaster	v2	SrcCode
DNPSlave	v2	SrcCode

รูปที่ 2.7 SCS ของ RTU+DNP ทั้งสองเวอร์ชัน [6]

Name: RTU+DNP
Current version: 2.0
Parent version: 1.0
Summary of changes: RTU+DNP changed in this release as a consequence of changes to both of the DNP subsystems. DNPMaster underwent trivial changes, however DNPSlave changed significantly (see DNPSlave change description for details).

Item	Change	Description
Core	none	
DNPMaster	modified	Supporting documentation within the subsystem was updated and made consistent
DNPSlave	modified	The source code was completely replaced in this release of DNPSlave

รูปที่ 2.8 รายละเอียดการเปลี่ยนแปลงของ RTU+DNP [6]

จากรูปที่ 2.8 จะเป็นรายละเอียดการเปลี่ยนแปลง (Subsystem Change Description) ซึ่งจะประกอบด้วย

1. เวอร์ชันปัจจุบันและเวอร์ชันก่อนหน้า
2. สรุปการเปลี่ยนแปลง
3. list ของ constituents พร้อมกับรายละเอียดของการเปลี่ยนแปลงใน subsystem

ประเภทของการเปลี่ยนแปลงในแต่ละ constituents นั้นจะมีดังนี้

- added ไอเท็มนี้ไม่พบในเวอร์ชันต้นแบบแต่ถูกเพิ่มมาในเวอร์ชันปัจจุบัน
- deleted ไอเท็มนี้พบในเวอร์ชันต้นแบบแต่ถูกตัดออกจากเวอร์ชันปัจจุบัน
- none ไอเท็มนี้ไม่มีการเปลี่ยนแปลง
- modify ไอเท็มนี้ถูกแก้ไข (หรือ ถูกแทนที่ด้วยเวอร์ชันที่อื่นของไอเท็มนี้)
- split ไอเท็มนี้เกิดจากการแบ่งไอเท็มนึงออกเป็นหลายไอเท็ม (เหมือนกับ delete แล้ว several add)
- combine ไอเท็มนี้เกิดจากการรวมกันของสองไอเท็มใดๆ ขึ้นไป
- move ไอเท็มนี้ถูกย้ายจาก subsystem ใดๆ มา subsystem หนึ่ง

2.2.2 กระบวนการจัดการโครงแบบ (Configuration Management Process) ของ SEPO [12]

Software Engineering Process Office (SEPO) เริ่มต้นก่อตั้งขึ้นเป็น Software Engineering Process Group แห่ง SSC San Diego ต่อมาได้ขยายบทบาทของกลุ่มออกไปในงานด้านอื่นๆนอกเหนือจากวิศวกรรมซอฟต์แวร์(Software Engineering) อันได้แก่ วิศวกรรมระบบ(Systems Engineering) การจัดการโครงการ(Project Management) และการรวมกลุ่มกระบวนการ(Corporate Processes) มีการปรับปรุงโครงสร้างจนกลายมาเป็นองค์กรในปัจจุบัน

แนวทางปฏิบัติเฉพาะในกลุ่มกระบวนการจัดการโครงแบบของแบบจำลองวุฒิภาวะความสามารถบูรณาการเมื่อนำมาเปรียบเทียบกับเอกสาร Configuration Management Process (Expert Mode) ของ SEPO [12] พบว่ามีความเชื่อมโยงกันดังตารางที่ 2.2

ตารางที่ 2.2 เปรียบเทียบแนวปฏิบัติเฉพาะในกลุ่มกระบวนการจัดการโครงแบบของซีเอ็มเอ็มไอ กับเอกสาร Configuration Management Process (Expert Mode) ของ SEPO [13]

ซีเอ็มเอ็มไอ	Configuration Management Process (SEPO)
SP1.1-1 : กำหนดคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) และสิ่งที่เกี่ยวข้องกับผลผลิตงาน ที่จะไว้ภายใต้ระบบการจัดการโครงแบบ	1. ปฏิบัติการการกำหนดโครงแบบ (Perform configuration identification)
SP1.2-1 : สร้างระบบจัดการโครงแบบ	
SP1.3-1 : สร้างหรือปลดปล่อยเบสไลน์	2. ปฏิบัติการควบคุมโครงแบบ (Perform Configuration Control)
SP2.1-1 : ติดตามการร้องขอการเปลี่ยนแปลง	
SP2.2-1 : การควบคุมคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI)	
SP3.1-1 : สร้างส่วนบันทึกการจัดการโครงแบบ	3. ปฏิบัติการการบันทึกสถานภาพโครงแบบ (Perform Configuration Status Accounting)
SP3.2-1 : การปฏิบัติการตรวจสอบบูรณาภาพของ configuration baselines	4. ปฏิบัติการการตรวจสอบและให้ความคิดเห็นโครงแบบ (Perform Configuration Audits and Reviews)

ซึ่ง SEPO ได้อธิบายรายละเอียดกระบวนการและกิจกรรมจัดการโครงแบบดังนี้

1. ปฏิบัติการการกำหนดโครงแบบ (Perform Configuration Identification)

1.1. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบกับผู้จัดการโครงการเห็นตรงกันที่จะกำหนดคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) ภายใต้ การจัดการโครงแบบ

1.2. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ กำหนดตัวระบุ (Identifier) ให้กับคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI)

2. ปฏิบัติการควบคุมโครงแบบ (Perform Configuration Control)

2.1. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ สร้าง CM libraries ตามที่เข้าใจตรงกันในแผนจัดการโครงแบบโครงการ (Project CMP)

2.2. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ จะนำคอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) หรือ technical artifacts ไว้ใน CM libraries

2.3. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ จะกำหนดสิทธิการเข้าใช้คอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) เพื่อเป็นการป้องกันการแก้ไขที่ไม่ได้รับการอนุญาต เมื่อขอเปลี่ยนแปลงคอนฟิกูเรชันเบสไลน์ (configuration baselines)

2.4. CCB จะต้องมีทวนสอบการขอแก้ไขเบสไลน์และตัดสินจากรายงานของปัญหา

2.5. CCB จะต้องให้สิทธิการของการขอเข้าเบสไลน์และทวนสอบและอนุมัติการร้องขอเปลี่ยนแปลง

2.6. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ สร้างเบสไลน์และสิ่งที่จะส่งมอบ

3. ปฏิบัติการการบันทึกสถานภาพโครงแบบ (Perform Configuration Status Accounting)

3.1. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ จะดูแลจัดการฐานข้อมูลสำหรับการออกรายงานสถานภาพโครงแบบ (CSA reports)

3.2. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบ จัดทำเอกสารข้อมูล ข้อกำหนดสำหรับฐานข้อมูลการออกรายงานสถานะโครงแบบ

3.3. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบและผู้จัดการโครงการ เลือกสื่อที่เหมาะสมเพื่อการเก็บและเข้าถึงฐานข้อมูล CSA

3.4. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบได้รับข้อมูลการเปลี่ยนแปลงเกี่ยวกับ คอนฟิกูเรชันไอเท็ม หรือ ซีไอ (Configuration Item - CI) เก็บไว้ในฐานข้อมูล เพื่อสนับสนุนการออกรายงานสถานภาพโครงแบบ (CSA reports)

3.5. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบสร้างรายงานสถานภาพโครงแบบ (CSA reports) เพื่อให้สถานภาพของเบสไลน์ปรากฏให้เห็นได้

3.6. กลุ่มผู้ปฏิบัติงานการจัดการโครงแบบแจกจ่ายรายงานสถานภาพโครงแบบ (CSA reports) ตามวาระเพื่อแสดงสถานภาพและประวัติของผลิตภัณฑ์ หมายเลขข้อกำหนดที่อนุมัติแล้ว ข้อมูลเบสไลน์ต่าง ๆ สถานภาพการปฏิบัติงาน CR การตัดสินใจของ CCB และข้อบกพร่องต่าง ๆ

4. ปฏิบัติการการตรวจสอบและให้ความคิดเห็นโครงแบบ (Perform Configuration Audits and Reviews)

4.1. ฝ่ายประกันคุณภาพ (QA) ตรวจสอบลักษณะการปฏิบัติงานของผลิตภัณฑ์เพื่อตรวจสอบว่า ทำงานตามข้อกำหนดที่ระบุไว้ในเอกสารโครงแบบหรือไม่

4.2. ฝ่ายประกันคุณภาพ (QA) ตรวจสอบโครงแบบของผลิตภัณฑ์ที่สร้างขึ้นกับข้อมูลทางเทคนิคเพื่อสร้างหรือทำให้เบสไลน์ของผลิตภัณฑ์ถูกต้อง

4.3. กลุ่มผู้ปฏิบัติงานโครงแบบ บันทึกกระบวนการที่ใช้ตรวจสอบข้อกำหนดโครงแบบของการตรวจสอบการปฏิบัติงานโครงแบบ (FCA) และการตรวจสอบทางกายภาพโครงแบบ (PCA) เพื่อเป็นไปในทางเดียวกับข้อกำหนดของ แผนการจัดการโครงแบบโครงการ

4.4. กลุ่มผู้ปฏิบัติงานโครงแบบสนับสนุน การตรวจสอบการปฏิบัติงานโครงแบบ (FCA) และการตรวจสอบทางกายภาพโครงแบบ (PCA)

4.5. กลุ่มผู้ปฏิบัติงานโครงแบบดูแลผลลัพธ์ของข้อบกพร่องที่ถูกรายงาน กับกิจกรรมต่าง ๆ ของการจัดการโครงแบบ