บทที่ 2

แนวทางการพัฒนาซีเอ็มเอ็ม

ความเป็นมาของซีเอ็มเอ็ม

ในเดือนพฤศจิกายน 1986 สถาบันซอฟต์แวร์เอ็นจิเนียริ่ง (Software Engineering Institute - SEI) ได้ถูกขอจากรัฐบาลให้มีส่วนรับผิดชอบในการจัดเตรียมวิธีการประเมินประสิทธิภาพของบริษัทที่รับทำซอฟต์แวร์ให้กับรัฐบาล จึงได้ร่วมมือกับมิททรีคอร์เปอเรชั่น (Mitre Corporation) เริ่มต้นทำการพัฒนาโครงร่างการเติบโตของกระบวนการ (The Process Maturity Framework) ซึ่งจะช่วยในการปรับปรุงพัฒนากระบวนการซอฟต์แวร์ขององค์กร

ต่อมาในเดือนกันยายน 1987 SEI ได้เผยแพร่รายละเอียดอย่างคร่าวๆของโครงร่างการเติบโตของกระบวนการ และแบบสอบถามเกี่ยวกับการเติบโต (Maturity Questionnaire) โดยที่ SEI ตั้งใจให้แบบสอบถามเกี่ยวกับการเติบโตนี้เป็นเครื่องมือแบบง่ายๆสำหรับกำหนดว่ากระบวนการซอฟต์แวร์ใดที่ต้องการการปรับปรุงพัฒนา

หลังจากมีประสบการณ์เกี่ยวกับโครงร่างการเติบโตของกระบวนการถึง 4 ปี และการทำแบบสอบถามเกี่ยวกับการเติบโตในชุดแรกๆ SEI จึงได้พัฒนาโครงร่างการเติบโตของกระบวนการซอฟต์แวร์ (Software Process Maturity Framework) ไปเป็น เคปปะบิลิตี้ แมตชัวลิติโมเดล (Capability Maturity Model for Software) หรือซีเอ็มเอ็ม โดยที่ซีเอ็มเอ็มอาศัยพื้นความรู้ที่ได้รับจากการประเมินค่ากระบวนการซอฟต์แวร์ (Software Process Assessments) และการตอบสนองอย่างมากจากทั้งภาคอุตสาหกรรมและภาครัฐบาล

ชุดแรกที่เริ่มต้นของซีเอ็มเอ็ม คือ เวอร์ชั่น 1.0 ซึ่งได้ถูกทบทวนและใช้โดยกลุ่มองค์กรผู้ผลิตซอฟต์แวร์ในระหว่างปี 1991 และ 1992 การจัดทำโครงงาน(Workshop) ของซีเอ็มเอ็มเวอร์ชั่น 1.0 ได้ถูกจัดขึ้นในเดือนเมษายน 1992 เข้าร่วมโดยผู้ทรงคุณวุฒิด้านซอฟต์แวร์ประมาณ 200 ราย ซึ่งผลการตอบสนองจากโครงงาน และจากกลุ่มองค์กรผู้ผลิตซอฟต์แวร์ก่อให้เกิด ซีเอ็มเอ็มเวอร์ชั่น 1.1 ซึ่งเป็นเวอร์ชั่นปัจจุบัน

ซีเอ็มเอ็มเป็นพื้นฐานของการสร้างชุดของเครื่องมือที่เป็นระบบ ซึ่งมีประโยชน์ในการปรับปรุงพัฒนากระบวนการซอฟต์แวร์ โดยจะใช้วัดกระบวนการซอฟต์แวร์ขององค์กรว่าปัจจุบันอยู่ในระดับใด เพื่อกำหนดปัจจัยที่เกี่ยวข้องที่จะทำให้กระบวนการซอฟต์แวร์พัฒนาไปในระดับที่สูงขึ้นไป และเพื่อจะ

หอสมุดกลาง สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ได้รับการสนับสนุนในการปรับปรุงพัฒนากระบวนการซอฟต์แวร์จากองค์กร นอกจากนี้ยังมีประโยชน์ในการประเมินค่าความสามารถทางซอฟต์แวร์ (Software Capability Evaluation) ซึ่งจะใช้ในการกำหนดคู่สัญญา (Contractors) ว่าบริษัทใดมีความสามารถเหมาะสมในงานซอฟต์แวร์นั้น หรือใช้ในการตรวจสอบสถานะ(Monitor)ของกระบวนการซอฟต์แวร์ที่ทำอยู่ เพื่อกำหนดจุดแข็ง จุดอ่อน และความเสี่ยงของกระบวนการซอฟต์แวร์ขององค์กร

แนวความคิดของซีเอ็มเอ็ม

1. โครงร่างการเติบโตของกระบวนการ (The Process Maturity Framework)

หลังจาก 2 ทศวรรษที่ผ่านมาองค์กรทั้งภาครัฐบาลและภาคเอกชน ตระหนักว่าปัญหาพื้นฐานที่ประสบคือ ไม่สามารถจัดการกับกระบวนการซอฟต์แวร์ (Software Process)ได้ หลายองค์กรและหลายโครงการ ประสบปัญหาความล่าช้า และค่าใช้จ่ายเกินงบประมาณที่ตั้งไว้ ซึ่งองค์กรไม่สามารถที่จะหาทางที่จะช่วยให้โครงการเหล่านั้นหลีกเลี่ยงปัญหาดังกล่าวได้

1.1 ข้อเปรียบเทียบระหว่างองค์กรผลิตซอฟต์แวร์ที่เติบโตเต็มที่ (Mature Software Organizations) และองค์กรผลิตซอฟต์แวร์ที่ยังเติบโตไม่สมบูรณ์ (Immature Software Organizations)

ก่อนที่จะไปถึงการพัฒนากระบวนการ (Process Improvement) ต้องเข้าใจความแตกต่างระหว่างองค์กรผลิตซอฟต์แวร์ที่เติบโตเต็มที่และองค์กรผลิตซอฟต์แวร์ที่ยังเติบโตไม่สมบูรณ์เสียก่อน

องค์กรผลิตซอฟต์แวร์ที่ยังเติบโตไม่สมบูรณ์ (Immature Organization) จะไม่มีเป้าหมายพื้นฐานที่จะพิจารณาคุณภาพของผลิตภัณฑ์ หรือเพื่อจะแก้ไขปัญหาของผลิตภัณฑ์หรือกระบวนการ ฉะนั้นแล้วคุณภาพของสินค้าจะยากต่อการคาดเดา กิจกรรมในการที่จะเพิ่มคุณภาพ เช่น การทบทวน (Reviews) และการทดสอบ (Testing) มักจะถูกกำจัดหรือละทิ้ง เมื่อโครงการล่าช้าจากตารางหรือแผนงานที่ได้กำหนดไว้

ส่วนองค์กรผลิตซอฟต์แวร์ที่เติบโตเต็มที่ (Mature Software Organization) จะมีความสามารถมากในการจัดการดูแลการพัฒนาซอฟต์แวร์ (Software Development) และกระบวนการบำรุงรักษาระบบ (Maintenance Process) กระบวนการพัฒนาซอฟต์แวร์จะดำเนินการตามแผนงานที่ตั้งไว้เหมาะสมสำหรับการใช้งานและเป็นไปอย่างต่อเนื่อง ในองค์กรที่เจริญเติบโตเต็มที่ ผู้จัดการจะคอยติดตามคุณภาพของผลิตภัณฑ์ซอฟต์แวร์ และความพอใจของลูกค้าโดยมีวัตถุประสงค์หรือเป้าหมายพื้นฐานที่จะพิจารณาตรวจสอบคุณภาพของผลิตภัณฑ์และวิเคราะห์ปัญหาของผลิตภัณฑ์หรือกระบวนการ

การจัดทำตารางแผนงานและงบประมาณอาศัยจากผลการดำเนินงานในอดีต และความเป็นจริงที่เป็นไปได้ ซึ่งจะทำให้สามารถที่จะคาดหวังในเรื่องต้นทุน ตารางแผนงาน และคุณภาพของผลิตภัณฑ์ได้

1.2 แนวความคิดพื้นฐาน ที่เกี่ยวกับการเติบโตของกระบวนการ (Fundamental Concepts Underlying Process Maturity)

กระบวนการ (Process) คือ ระบบของการปฏิบัติการในการผลิตบางสิ่งบางอย่าง เป็นชุดของการกระทำ ความเปลี่ยนแปลง หรือ หน้าที่ที่บรรลุยังผลลัพธ์

กระบวนการซอฟต์แวร์ (Software Process) สามารถจำกัดความว่าเป็นกลุ่มของกิจกรรม วิธีการ การฝึกฝนปฏิบัติ ที่ใช้สำหรับพัฒนา และบำรุงรักษาซอฟต์แวร์และผลิตภัณฑ์ที่เกี่ยวข้อง (ตัวอย่าง เช่น แผนงานโครงการ (Project Plans) เอกสารการออกแบบ (Design Document) รหัสโปรแกรม ชุดทดสอบ และคู่มือการใช้งาน

ขีดความสามารถของกระบวนการซอฟต์แวร์ (Software Process Capability) หมายถึง ช่วงของผลลัพธ์ที่คาดว่าจะสามารถบรรลุได้ โดยการดำเนินการตามกระบวนการซอฟต์แวร์

ผลการดำเนินการกระบวนการซอฟต์แวร์ (Software Process Performance) หมายถึง ผลลัพธ์ที่บรรลุ สำเร็จจริง จากการดำเนินการตามกระบวนการซอฟต์แวร์

การเติบโตของกระบวนการซอฟต์แวร์ (Software Process Maturity) ครอบคลุมไปถึง การดำเนินงานที่สามารถวัดผลได้ จัดการควบคุมได้ และเพิ่มประสิทธิภาพได้

การเติบโต (Maturity) แสดงถึงศักยภาพในการเจริญเติบโตในด้านความสามารถ(Capability) ในการทำให้องค์กรผลิตซอฟต์แวร์ได้อย่างสมบูรณ์ ตามที่คาดหมายไว้ โดยที่มีการจัดทำเอกสาร จัดฝึกอบรม ตรวจสอบและปรับปรุงอยู่ตลอดเวลา ซึ่งสามารถใช้ได้กับโครงการต่างๆ ขององค์กร

1.3 แนวคิดเบื้องต้นของ แบบจำลองการเจริญเติบโตด้านความสามารถขององค์กรซอฟต์แวร์ (Overview of Capability Maturity Model)

แม้ว่าวิศวกรซอฟต์แวร์และผู้จัดการ มักจะตระหนักในรายละเอียดปัญหาต่างๆของเขา แต่มักจะไม่เห็นด้วยกับการพัฒนาซึ่งเป็นสิ่งสำคัญ ถ้าองค์กรไม่มีนโยบาย หรือกลยุทธ์เพื่อการพัฒนาก็ทำให้เป็นการยากที่จะรู้ว่ากิจกรรมเพื่อการพัฒนาจะต้องเริ่มต้นอย่างไร การที่จะบรรลุผลในการพัฒนากระบวนการ จำเป็นต้องพัฒนาแนวทางในการประเมินการเติบโตของกระบวนการซอฟต์แวร์ขององค์กรเป็นขั้นๆ โครงร่างการเติบโตของกระบวนการซอฟต์แวร์ (Software Process Maturity Framework)

ได้ทำการจัดเรียงลำดับขั้นตอนเหล่านี้เป็นขั้นๆ เพื่อให้การพัฒนาในแต่ละขั้น มีแนวทางพื้นฐานเพื่อเตรียมพร้อมที่จะพัฒนาไปยังขั้นต่อไป โดยมีแนวทางในการพัฒนากระบวนการอย่างเป็นขั้นตอนต่อเนื่อง แต่ไม่ได้มุ่งเพื่อแก้ปัญหาให้กับโครงการแบบจำลองการเติบโตด้านความสามารถสำหรับซอฟต์แวร์ (Capability Maturity Model for Software) หรือซีเอ็มเอ็มถูกออกแบบมาเพื่อแนะนำแนวทางให้กับองค์กรซอฟต์แวร์ว่าจะควบคุมกระบวนการพัฒนาและบำรุงรักษาซอฟต์แวร์อย่างไร และเป็นแนวทางในการเลือกกลยุทธ์สำหรับปรับปรุงพัฒนากระบวนการโดยการประเมินการเติบโตของกระบวนการปัจจุบัน และแจกแจงปัจจัยที่สำคัญต่อคุณภาพของซอฟต์แวร์และการปรับปรุงพัฒนากระบวนการ

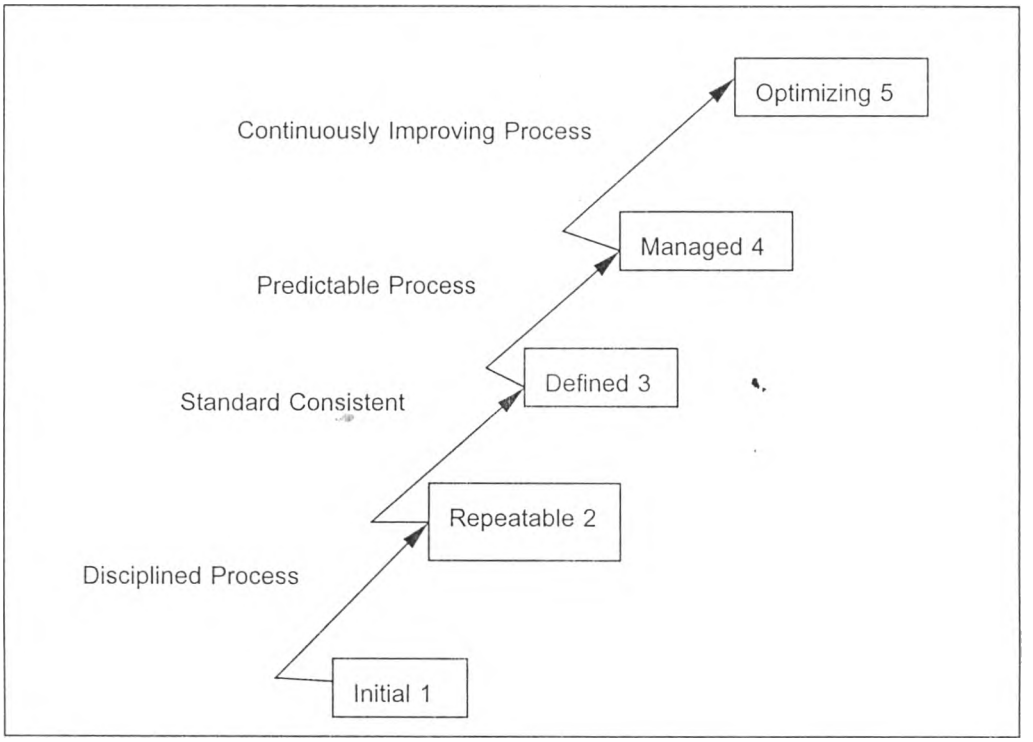
โครงสร้างของซีเอ็มเอ็มอาศัยหลักการของคุณภาพผลิตภัณฑ์ (Product Quality) ซึ่งเกิดขึ้นเมื่อประมาณ 60 ปีที่แล้ว ใน ค.ศ.1930 Walter Shewart ได้ประกาศหลักการของการควบคุมคุณภาพเชิงสถิติ (Statistical Quality Control) ซึ่งหลักการนี้ได้ถูกดัดแปลงโดย SEI เป็นโครงร่างการเติบโต (Maturity Framework) ซึ่งก่อให้เกิดการควบคุมโครงการ (Project Management) และแนวคิดทางวิศวกรรมสำหรับการควบคุมเชิงปริมาณของกระบวนการซอฟต์แวร์ ซึ่งเป็นพื้นฐานสำหรับการปรับปรุงพัฒนากระบวนการอย่างต่อเนื่องต่อไป

โครงร่างการเติบโตในหลักการทางคุณภาพได้ถูกนำไปดัดแปลงใช้ครั้งแรกในหนังสือ "Quality is Free" ของ Philip Crosby การเติบโตด้านการจัดการควบคุมคุณภาพของ Crosby อธิบายขั้นตอน 5 ขั้นของแบบฝึกปฏิบัติด้านคุณภาพ ต่อมาโครงร่างการเติบโตถูกดัดแปลงมาเป็นกระบวนการซอฟต์แวร์ โดย Ron Radice และคณะทำงานภายใต้แนวทางของ Watts Humphrey ที่ IBM และในปี 1986 Humphrey ได้นำเอาโครงร่างการเติบโตไปยังสถาบันซอฟต์แวร์เอ็นจิเนียริ่ง และได้เพิ่มเติมแนวคิดเกี่ยวกับระดับของการเติบโต (Maturity Levels) เข้าไป จนกลายมาเป็นแบบจำลองการเติบโตด้านความสามารถสำหรับซอฟต์แวร์ หรือ ซีเอ็มเอ็ม ในปัจจุบัน

นอกจากนั้นตั้งแต่ปี 1990 เป็นต้นมา SEI โดยความร่วมมือของบุคคลต่างๆ ทั้งภาครัฐบาลและภาคเอกชน ยังได้ขยายและขัดเกลา พื้นฐานของแบบจำลอง (Model Based) ไปเป็นการปรับปรุงพัฒนากระบวนการซอฟต์แวร์ (Software Process Improvement) อีกด้วย

2. ห้าระดับของการเติบโตกระบวนการซอฟต์แวร์ (The Five Levels Of Software Process Maturity)

ซีเอ็มเอ็มแบ่งออกเป็น 5 ระดับ เพื่อจะพัฒนาปรับปรุงกระบวนการอย่างเป็นขั้นตอนต่อเนื่อง ห้าระดับของการเติบโต แสดงถึงเกณฑ์วัดอย่างเป็นลำดับ สำหรับวัดการเติบโตของกระบวนการซอฟต์แวร์ขององค์กรและเพื่อประเมินความสามารถของกระบวนการซอฟต์แวร์



รูปภาพ 2.1 แสดง 5 ระดับของการเติบโตของกระบวนการซอฟต์แวร์

จากรูปภาพ2.1แสดงถึงกิจกรรมการพัฒนา ที่ต้องกระทำเป็นลำดับเพื่อเพิ่มการเติบโตของกระบวนการซอฟต์แวร์ ลูกศรในรูปภาพ แสดงถึงประเภทของความสามารถของกระบวนการในแต่ละระดับของการเติบโต

ลักษณะของระดับการเติบโต 5 ขั้น ที่ชี้ให้เห็นถึงการเปลี่ยนแปลงของกระบวนการในแต่ละระดับ มีดังต่อไปนี้

1. การเริ่มต้น (Initial)

กระบวนการซอฟต์แวร์จะมีลักษณะไม่ได้วางแผนไว้ล่วงหน้าและมักจะยุ่งเหยิงสับสน และความสำเร็จขึ้นอยู่กับความพยายามของแต่ละบุคคล

2. การทบทวน (Repeatable)

เริ่มมีกระบวนการบริหารโครงการขั้นพื้นฐาน เพื่อติดตามเรื่องต้นทุน ตารางการทำงานและการปฏิบัติงาน ระเบียบปฏิบัติของกระบวนการที่จำเป็นถูกนำมาใช้กับโครงการที่คล้าย ๆ กับที่เคยทำมาก่อน

3. การกำหนด (Defined)

ขั้นตอนกระบวนการซอฟต์แวร์ สำหรับการจัดการ และกิจกรรมทางวิศวกรรมถูกจัดทำเป็นเอกสารมาตรฐาน และรวบรวมไปเป็นขั้นตอนกระบวนการซอฟต์แวร์มาตรฐานขององค์กรทุกโครงการใช้ขั้นตอนกระบวนการซอฟต์แวร์มาตรฐาน ที่ผ่านการอนุมัติและจัดทำขึ้นโดยเฉพาะเพื่อพัฒนาและบำรุงรักษาซอฟต์แวร์

4. การจัดการ (Managed)

มีการเก็บรวบรวมรายละเอียดมาตรฐานการวัดกระบวนการซอฟต์แวร์และคุณภาพของผลิตภัณฑ์เป็นอย่างดีขั้นตอนซอฟต์แวร์และผลิตภัณฑ์เป็นที่เข้าใจ และสามารถควบคุมได้

5. การปรับปรุง (Optimizing)

การปรับปรุงกระบวนการเป็นไปอย่างต่อเนื่องโดยได้รับการตอบรับจากกระบวนการและจากแนวคิดที่เปลี่ยนแปลงและเทคโนโลยีต่าง ๆ

3. ลักษณะพฤติกรรมของระดับการเติบโต (Behavioral Characterization of the Maturity Levels)

การเติบโตระดับ 2 ถึงระดับ 5 สามารถแสดงถึงลักษณะกิจกรรมที่ทำโดยองค์กรเพื่อที่จะสร้างหรือปรับปรุงกระบวนการซอฟต์แวร์โดยดูจากกิจกรรมที่ทำในแต่ละโครงการ และจากผลของความสามารถของกระบวนการในแต่ละโครงการที่เกี่ยวข้องกัน ลักษณะพฤติกรรมของระดับ 1 จะรวมการสร้างฐานสำหรับการเปรียบเทียบเพื่อปรับปรุงพัฒนากระบวนการไปยังระดับการเติบโตที่สูงกว่าด้วย

3.1 ระดับที่ 1 - การเริ่มต้น (Level 1 - The Initial Level)

ในระดับการเริ่มต้น องค์กรไม่ได้จัดเตรียมสภาพแวดล้อมที่คงที่ สำหรับการพัฒนาและบำรุงรักษาซอฟต์แวร์ เมื่อองค์กรขาดการฝึกฝนปฏิบัติด้านการจัดการ คุณประโยชน์ของการปฏิบัติด้านวิศวกรรมซอฟต์แวร์ที่ดีจึงถูกละเลย

ในระหว่างวิกฤตกาล โครงการโดยทั่วไปจะละทิ้งขั้นตอนที่ได้วางแผนไว้และกลับไปสู่การเขียนรหัสโปรแกรม (Coding) และการทดสอบ (Testing) ความสำเร็จขึ้นอยู่กับการมีผู้จัดการที่พิเศษและการมีทีมงานที่มีประสิทธิภาพ บางครั้งผู้จัดการที่มีความสามารถและเด็ดเดี่ยวมั่นคงจะสามารถต้านทานต่อแรงกดดันต่าง ๆเพื่อที่จะหาทางลัดในกระบวนการซอฟต์แวร์ แต่เมื่อเขาออกจากโครงการนั้น ๆอิทธิพลต่าง ๆที่เคยมีอย่างสม่ำเสมอก็หมดลง รวมทั้งกระบวนการทางด้านวิศวกรรมที่เคยกระทำอย่างเข้มแข็งก็หายไปด้วย

ความสามารถของกระบวนการซอฟต์แวร์ในระดับ 1 นี้ องค์กรไม่สามารถคาดการณ์ได้ เพราะว่ากระบวนการซอฟต์แวร์มักจะถูกเปลี่ยนแปลงหรือแก้ไขอยู่ตลอดเวลาในการดำเนินงาน (กระบวนการไม่ได้ถูกวางแผนไว้ล่วงหน้า) ตารางเวลา งบประมาณ อำนาจหน้าที่ และคุณภาพของผลิตภัณฑ์ไม่สามารถคาดการณ์ได้ ผลงานขึ้นอยู่กับความสามารถของแต่ละบุคคลซึ่งจะมากน้อยแตกต่างกันไปตามทักษะ ความรู้และแรงจูงใจของแต่ละคน จะมีกระบวนการเพียงไม่กี่กระบวน การที่ชัดเจนและผลงานสามารถคาดการณ์ได้ซึ่งขึ้นอยู่กับความสามารถของแต่ละบุคคลมากกว่าความสามารถขององค์กร

3.2 ระดับที่ 2 - การทบทวน (Level 2 - The Repeatable Level)

ในระดับการทบทวน นโยบายสำหรับการบริหารโครงการซอฟต์แวร์และขั้นตอนต่างๆที่จะนำนโยบายไปใช้ได้ถูกจัดทำขึ้น การวางแผนและการจัดการโครงการใหม่ๆ อาศัยประสบการณ์พื้นฐานจากโครงการที่คล้ายๆกัน เป้าหมายในระดับที่ 2 นี้คือ การจัดให้มีกระบวนการในการจัดการบริหารโครงการซอฟต์แวร์ที่มีประสิทธิภาพ โดยการให้องค์กรกระทำซ้ำตามแบบที่เคยปฏิบัติในการพัฒนาที่ประสบความสำเร็จในโครงการก่อนหน้านั้น แม้นว่ารายละเอียดของกระบวนการที่นำไปใช้ในโครงการอาจแตกต่างกันกระบวนการที่มีประสิทธิภาพจะมีคุณลักษณะดังนี้คือ ต้องมีการฝึกฝนปฏิบัติ จัดทำเป็นเอกสาร บังคับควบคุมให้ทำตามได้ มีการฝึกอบรม มีการวัดผลงาน และต้องสามารถปรับปรุงพัฒนาได้

โครงการในระดับที่ 2 นี้ องค์กรจะต้องเริ่มใช้การควบคุมจัดการซอฟต์แวร์ขั้นพื้นฐาน พันธะสัญญาที่สามารถทำจริงได้ อาศัยผลลัพธ์ที่สังเกตจากโครงการก่อนๆ และความต้องการของโครงการปัจจุบัน ผู้จัดการซอฟต์แวร์จะคอยติดตามตรวจสอบต้นทุนซอฟต์แวร์ของโครงการ ตารางการทำงาน และมีหน้าที่รับผิดชอบในการแก้ไขปัญหาต่างๆที่เกิดขึ้น โดยมีเป้าหมายให้ความต้องการทางด้านซอฟต์แวร์และงานที่พัฒนาสนองตอบความต้องการ ความพอใจ ที่ได้ตั้งไว้ มาตรฐานของโครงการซอฟต์แวร์ได้ถูกกำหนดขึ้น และองค์กรมั่นใจว่าทุกคนจะปฏิบัติตามอย่างซื่อสัตย์ และมาตรฐานของการทำงานโครงการซอฟต์แวร์กับบุคคลภายนอก (Subcontractors) ได้ถูกกำหนดขึ้น (ถ้ามี) เพื่อความสัมพันธ์ที่ดีระหว่างลูกค้าและผู้ผลิต

ความสามารถของกระบวนการซอฟต์แวร์ระดับที่ 2 นี้ สามารถสรุปได้ว่าองค์กรต้องมีระเบียบวินัย เพราะว่า การวางแผน และการติดตามตรวจสอบโครงการซอฟต์แวร์ ต้องดำเนินการอย่างสม่ำเสมอและความสำเร็จก่อนหน้านี้สามารถกระทำซ้ำได้ กระบวนการต่างๆของโครงการดำเนินภายใต้การควบคุมที่มีประสิทธิภาพของระบบการบริหารโครงการ และดำเนินการตามแผนงานที่สามารถเป็นจริงได้ โดยอาศัยผลงานความสำเร็จของโครงการก่อนๆ

3.3 ระดับที่ 3 - การกำหนด (The Defined Level)

ในระดับการกำหนดนี้ มาตรฐานของกระบวนการในพัฒนาและการบำรุงรักษาซอฟต์แวร์ของทั้งองค์กร รวมทั้งวิศวกรรมซอฟต์แวร์ และกระบวนการทางด้านบริหารต่างๆ ได้ถูกจัดทำขึ้นเป็นเอกสาร (ลายลักษณ์ อักษร)มาตรฐานนี้รู้จักกันในนามของมาตรฐานกระบวนการซอฟต์แวร์ขององค์กร (Organization' s Standard Software Process) กระบวนการต่างๆที่ถูกสร้างขึ้นในระดับที่ 3 นี้ (สามารถเปลี่ยนแปลงได้ตามความเหมาะสม) จะช่วยให้ผู้จัดการซอฟต์แวร์และพนักงานฝ่ายเทคนิคสามารถปฏิบัติงานได้อย่างมีประสิทธิภาพมากขึ้นองค์กรจะใช้การฝึกฝนปฏิบัติทางด้านวิศวกรรมซอฟต์แวร์ที่มีประสิทธิภาพในขณะที่กำหนดมาตรฐานกระบวนการซอฟต์แวร์ จะมีการจัดตั้งกลุ่มคนที่มีหน้าที่รับผิดชอบกิจกรรมต่างๆ ในกระบวนการซอฟต์แวร์ขององค์กร ตัวอย่างเช่น กลุ่มกระบวนการวิศวกรรมซอฟต์แวร์ (Software Engineering Process Group หรือ SEPG) นอกจากนี้ยังมีการจัดโปรแกรมฝึกอบรมทั่วทั้งองค์กร เพื่อให้มั่นใจได้ว่าพนักงานและผู้จัดการมีความรู้และทักษะที่ต้องการตรงกับบทบาทหน้าที่ที่ได้รับมอบหมาย

โครงการหลายโครงการที่มีการจัดทำมาตรฐานกระบวนการซอฟต์แวร์ขององค์กรโดยเฉพาะเพื่อพัฒนากระบวนการซอฟต์แวร์ของตนเอง ซึ่งมีรายละเอียดที่เป็นเอกลักษณ์เฉพาะของโครงการนั้นซึ่งจะรู้จักกันในนามกระบวนการซอฟต์แวร์ที่ถูกกำหนดขึ้นของโครงการ (Project 's Defined Software Process) กระบวนการซอฟต์แวร์ที่ถูกกำหนดนี้จะประกอบด้วย ชุดของการรวมกันระหว่างวิศวกรรมซอฟต์แวร์ที่ถูกกำหนดเป็นอย่างดี และกระบวนการทางด้านบริหาร ซึ่งจะมีคุณลักษณะที่รวมทั้งเกณฑ์ในการบอกถึงความพร้อม (Readiness Criteria) การนำเข้า (Inputs) มาตรฐานและขั้นตอนในการปฏิบัติงาน กลไกในการตรวจสอบ (Verification Mechanisms) เช่น การตรวจสอบทบทวนการทำงาน (Peer Reviews) ผลลัพธ์ (Outputs) และเกณฑ์ในการบอกว่าเสร็จสมบูรณ์ (Completion Criteria) เนื่องจากกระบวนการซอฟต์แวร์ได้ถูกกำหนดไว้เป็นอย่างดี ผู้บริหารจะสามารถเข้าใจและเห็นภาพของขั้นตอนย่อยๆทางด้านเทคนิค ของทั้งโครงการได้อย่างชัดเจน

ความสามารถของกระบวนซอฟต์แวร์ในระดับที่ 3 นี้ องค์กรจะสรุปได้ว่ามีมาตรฐาน และ ไม่ค่อยเปลี่ยนแปลง เพราะว่าทั้งวิศวกรรมซอฟต์แวร์และกิจกรรมด้านการบริหารจะคงที่ และถูกปฏิบัติซ้ำตามที่เคยทำมา โดยที่การจัดทำสายของผลิตภัณฑ์ (Product Lines) ต้นทุน ตารางการทำงานและหน้าที่ต่างๆ อยู่ภายใต้การควบคุม และคุณภาพของซอฟต์แวร์จะถูกติดตามตรวจสอบ(Tracked) อยู่เสมอ ความสามารถของกระบวนการเหล่านี้อาศัยความเข้าใจในกิจกรรมต่างๆของทั้งองค์กร รวมทั้งเข้าใจในบทบาทและหน้าที่ความรับผิดชอบในกระบวนการซอฟต์แวร์ที่กำหนดขึ้น

3.4 ระดับที่ 4 - การจัดการ (Level 4 - The Managed Level)

ในระดับการจัดการนี้องค์กรได้จัดตั้งเป้าหมายทางคุณภาพเชิงปริมาณของผลิตภัณฑ์ซอฟต์แวร์และกระบวนการขึ้น โดยที่กิจกรรมกระบวนการซอฟต์แวร์ที่สำคัญของทั้งโครงการจะถูกวัดผลงานในด้านประสิทธิภาพและคุณภาพซึ่งเป็นส่วนหนึ่งของโปรแกรมการวัดผลงานขององค์กร มีการจัดทำฐานข้อมูลของกระบวนการซอฟต์แวร์ของทั้งองค์กรเพื่อใช้ในการสะสมรวบรวมและวิเคราะห์ข้อมูลที่ได้มาจากกระบวนการซอฟต์แวร์ที่ได้กำหนดขึ้นมาของโครงการ กระบวนการซอฟต์แวร์ในระดับที่ 4 นี้จะถูกกำหนดไว้เป็นอย่างดี และมีการวัดผลงานที่แน่นอน มาตรฐานในการวัดผลต่าง ๆได้ถูกจัดทำขึ้นเพื่อใช้ประเมินผลกระบวนการซอฟต์แวร์ของโครงการและผลิตภัณฑ์ต่าง ๆ

โครงการต่าง ๆ สามารถควบคุมผลิตภัณฑ์และกระบวนการต่าง ๆได้โดยการค่อย ๆปรับผลการปฏิบัติงานของกระบวนการให้เข้าไปยังขอบเขตที่ยอมรับได้ ซึ่งต้องอาศัยการเรียนรู้และการจัดการที่ระมัดระวัง

ความสามารถของกระบวนการซอฟต์แวร์ ในระดับที่ 4 นี้ องค์กรจะสรุปได้ว่า คาดการณ์ได้ เพราะว่า จะสามารถวัดผลงานของกระบวนการได้ และการดำเนินงาน จะกระทำภายใต้ขอบเขตที่จำกัดที่สามารถวัดผลได้ ความสามารถของกระบวนการในระดับนี้ทำให้ องค์กรสามารถคาดการณ์แนวโน้ม ทิศทางของกระบวนการ และคุณภาพของผลิตภัณฑ์ได้ภายใต้ขอบเขตของข้อจำกัดต่าง ๆ ถ้าข้อจำกัดเหล่านี้ถูกละเมิดจะมีการปฏิบัติเพื่อแก้ไขสถานการณ์ให้ถูกต้อง ผลิตภัณฑ์ซอฟต์แวร์ที่ได้จึงคาดการณ์ได้ว่าจะมีคุณภาพสูง

3.5 ระดับที่ 5 - การปรับปรุง (Level 5 - The Optimizing Level)

ในระดับการปรับปรุงนี้ ทั่วทั้งองค์กรจะมุ่งเน้นไปยังการปรับปรุง พัฒนากระบวนการอย่างต่อเนื่อง องค์กรจะมีวิธีการที่จะสามารถบ่งชี้จุดอ่อนและจุดแข็งของกระบวนการที่ดำเนินอยู่ โดยมีเป้าหมายในการป้องกันความเสียหายที่อาจเกิดขึ้นอีก ข้อมูลที่ได้จากกระบวนการซอฟต์แวร์ที่ประสบความสำเร็จ ใช้ในการวิเคราะห์ผลประโยชน์ด้านต้นทุนของเทคโนโลยีใหม่ และใช้ในการปรับปรุงเปลี่ยนแปลงกระบวนการซอฟต์แวร์ขององค์กร ความเปลี่ยนแปลงที่ใช้ประโยชน์จากการฝึกปฏิบัติทางด้านวิศวกรรมซอฟต์แวร์ (Software Engineering Practices) ที่ดีที่สุดได้ถูกกำหนดขึ้น และถ่ายทอดไปทั่วทั้งองค์กร

ทีมงานของโครงการซอฟต์แวร์ในระดับที่ 5 นี้ องค์กรจะวิเคราะห์ความเสียหายต่าง ๆที่เกิดขึ้นเพื่อหาสาเหตุ กระบวนการซอฟต์แวร์จะถูกประเมินผลงาน เพื่อป้องกันการเกิดซ้ำของความเสียหายในแบบที่รู้จัก และบทเรียนที่ได้รับจะถูกเผยแพร่ไปยังโครงการอื่น ๆ

ความสามารถของกระบวนการซอฟต์แวร์ในระดับที่ 5 นี้ องค์กรจะมีคุณลักษณะที่มีการปรับปรุงพัฒนาอย่างต่อเนื่อง เพราะว่าในระดับที่ 5 นี้ องค์กรจะพยายามอย่างต่อเนื่องที่จะปรับปรุงพัฒนาขีดความสามารถของกระบวนการ โดยการปรับปรุงพัฒนาประสิทธิภาพของกระบวนการของโครงการต่างๆ การปรับปรุงพัฒนาเกิดขึ้นจากทั้งการเพิ่มขึ้นของความก้าวหน้าในกระบวนการที่มีอยู่แล้วทีละน้อยๆ และจากการเปลี่ยนแปลงโดยใช้เทคโนโลยีและวิธีการใหม่ๆ

4 การทำความเข้าใจกับระดับของการเติบโต (Understanding the Maturity Levels)

ซีเอ็มเอ็ม เป็นแบบจำลองที่อธิบายองค์ประกอบที่จำเป็น ซึ่งบ่งชี้คุณลักษณะขององค์กรในแต่ละระดับของการเติบโต ซึ่งเป็นแบบจำลองพื้นฐานที่แสดงคุณลักษณะของพฤติกรรมโดยทั่วไป ที่จะพบได้ในองค์กรที่ดำเนินโครงการขนาดใหญ่ เช่น สัญญาทำงานกับหน่วยงานรัฐบาล ซีเอ็มเอ็มไม่ได้อธิบายว่าจะนำกระบวนการซอฟต์แวร์ไปใช้งานในองค์กรได้อย่างไร แต่จะอธิบายว่าอะไร คือองค์ประกอบที่จำเป็น ซึ่งมักพบโดยทั่วไปของกระบวนการซอฟต์แวร์

ซีเอ็มเอ็ม ไม่ได้อธิบายให้องค์กรทราบว่า จะปรับปรุงพัฒนาอย่างไร แต่ซีเอ็มเอ็ม อธิบายถึงลักษณะของระดับการเติบโตแต่ละขั้น โดยที่ไม่ได้บอกถึงแนวทางว่าจะต้องทำอย่างไรจึงจะสามารถไปถึงระดับต่างๆได้ ซึ่งอาจจะต้องใช้เวลาหลายปีในการพัฒนาจากระดับที่ 1 ไปสู่ระดับที่ 2 และการพัฒนาไปยังแต่ละระดับอาจต้องใช้เวลาอย่างน้อย 2 ปี

การปรับปรุงพัฒนากระบวนการซอฟต์แวร์เกิดขึ้นท่ามกลางสิ่งแวดล้อม ของการวางแผนกลยุทธ์ขององค์กร และวัตถุประสงค์ของธุรกิจ โครงสร้างขององค์กร เทคโนโลยีที่ใช้ วัฒนธรรมในสังคมองค์กร ระบบการบริหารขององค์กร ซีเอ็มเอ็มจะมุ่งเน้นไปยังกระบวนการต่างๆในแง่ของความพยายามในการจัดการด้านคุณภาพทั้งหมด การปรับปรุงคุณภาพของกระบวนการที่ประสบความสำเร็จ จะหมายความว่า สิ่งต่างๆที่อยู่นอกเหนือขอบเขตของกระบวนการซอฟต์แวร์จะต้องถูกจัดเตรียมไว้ด้วย ตัวอย่างเช่น ปัจจัยด้านบุคลากร ที่มีส่วนร่วมในการเปลี่ยนแปลงวัฒนธรรมขององค์กร ซึ่งจะทำให้การปรับปรุงพัฒนากระบวนการถูกนำไปใช้งานและเกิดขึ้นได้

4.1 ทำความเข้าใจกับระดับการเริ่มต้น (Understanding the Initial Level)

แม้ว่าในระดับที่ 1 องค์กรมักจะมีคุณลักษณะที่ไม่ได้วางแผนไว้ล่วงหน้า และกระบวนการต่างๆมักจะยุ่งเหยิงสับสน แต่บ่อยครั้งที่องค์กรพัฒนาผลิตภัณฑ์ได้สำเร็จ แม้ว่าอาจจะเกินงบประมาณและเกินตารางเวลาที่ได้กำหนดไว้ ความสำเร็จในระดับที่ 1 นี้ ต้องขึ้นอยู่กับความสามารถและความกล้าหาญของบุคลากรในองค์กร การคัดเลือก การว่าจ้าง การพัฒนา และ/หรือ การรักษาความสามารถของบุคลากรเป็นสิ่งสำคัญสำหรับองค์กรในทุกระดับของการเติบโต แต่สิ่งเหล่านี้จะอยู่นอกเหนือขอบเขตของซีเอ็มเอ็ม

4.2 ทำความเข้าใจกับระดับการทบทวนและระดับการกำหนด (Understanding the Repeatable and Defined Levels)

เมื่อโครงการเติบโตขึ้นในด้านขนาดและความซับซ้อน ความเอาใจใส่จะเปลี่ยนจากปัจจัยทางด้านเทคนิคไปเป็นปัจจัยทางด้านองค์กรและการจัดการ ซึ่งมุ่งเน้นไปยังการเติบโตของกระบวนการ กระบวนการทำให้บุคลากรสามารถทำงานได้อย่างมีประสิทธิภาพมากขึ้นโดยการมีส่วนร่วมในบทเรียนซึ่งเรียนรู้โดยพนักงานที่ดีที่สุดไปเป็นกระบวนการที่เป็นลายลักษณ์อักษร การเสริมสร้างทักษะต้องอาศัยการกระทำกระบวนการต่าง ๆเหล่านั้นอย่างมีประสิทธิภาพ (มักกระทำโดยการอบรม) และการปรับปรุงพัฒนาอย่างต่อเนื่อง โดยอาศัยการเรียนรู้จากบุคลากรที่ดำเนินงานนั้น ๆ

การจะไปถึงระดับที่ 2 ฝ่ายบริหารต้องมุ่งเน้นไปยังกระบวนการของตนเอง เพื่อที่จะบรรลุกระบวนการซอฟต์แวร์ที่มีระเบียบวินัย ระดับที่ 2 จะจัดเตรียมพื้นฐานสำหรับระดับที่ 3 เพราะว่าจุดมุ่งหมายก็คือ การกระทำทางด้านการจัดการเพื่อที่จะปรับปรุงพัฒนากระบวนการ ก่อนที่จะจัดการปัจจัยทางด้านเทคนิคและองค์กรในระดับที่ 3 ฝ่ายบริหารจะจัดตั้งตำแหน่งผู้นำเพื่อที่จะบรรลุยังระดับที่ 2 โดยการจัดทำเป็นเอกสารลายลักษณ์อักษร และปฏิบัติตามกระบวนการทางด้านการบริหารโครงการ

กระบวนการต่าง ๆ ของโครงการในระดับที่ 2 และ 3 จะแตกต่างกันไป โดยที่ ความต้องการขององค์กรในการบรรลุระดับที่ 2 ก็คือ การมีนโยบายที่จะเป็นแนวทางให้โครงการในการจัดตั้งกระบวนการทางด้านบริหารที่เหมาะสม มีขั้นตอนที่ถูกกำหนดขึ้นเป็นลายลักษณ์อักษร เพื่อจะเป็นพื้นฐานในการดำเนินไปอย่างต่อเนื่องของกระบวนการ ซึ่งสามารถนำไปใช้ได้ทั่วทั้งองค์กร โดยอาศัยการฝึกอบรมและความเชื่อถือในคุณภาพซอฟต์แวร์

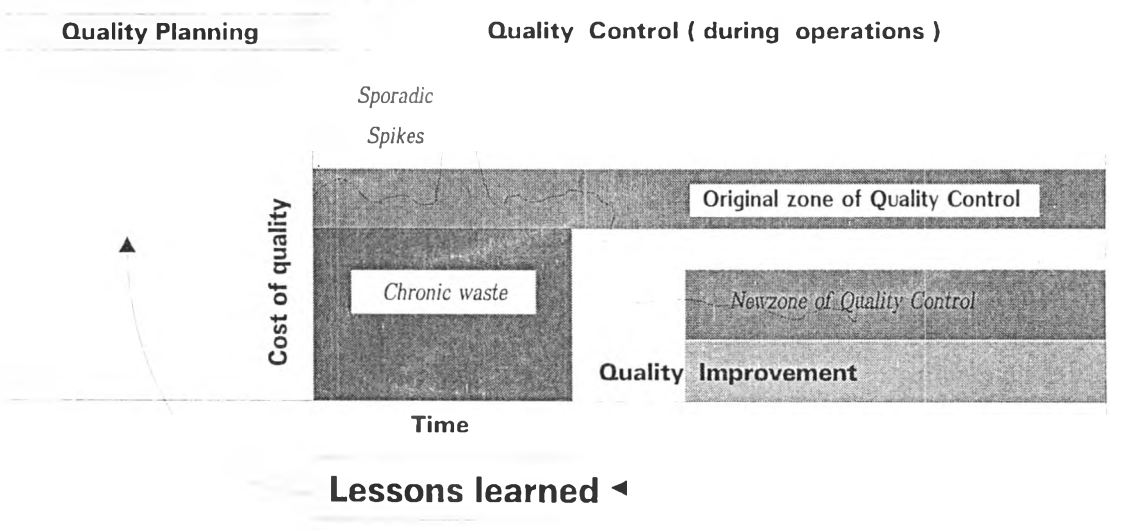
ในระดับที่ 3 ดำเนินตามการบริหารโครงการโดย การกำหนด(Defining) การรวมกันเป็นอันหนึ่งอันเดียวกัน(Integrating) และการจัดทำเป็นเอกสาร(Documenting)ของกระบวนการซอฟต์แวร์ทั้งหมด การรวมกันเป็นอันหนึ่งอันเดียวกัน ในที่นี้หมายถึง ผลลัพธ์ที่ได้จากงานหนึ่งสามารถนำเข้าสู่งานถัดไปได้อย่างต่อเนื่อง เมื่อมีการจับคู่ผิดพลาดระหว่างงาน จะสามารถทราบและระบุได้ในขั้นตอนการวางแผนของกระบวนการซอฟต์แวร์ มากกว่าที่จะรู้เมื่อเผชิญกับการทำงานจริงของกระบวนการ

4.3 ทำความเข้าใจกับระดับการจัดการและการปรับปรุง (Understanding the Managed and Optimizing Levels)

ระดับการเติบโต ระดับที่ 4 และ 5 ยังเป็นสิ่งที่ไม่ค่อยรู้จักกันเท่าไรนักเมื่อเทียบกับระดับอื่น ๆ ในอุตสาหกรรมซอฟต์แวร์ มีเพียงตัวอย่างแค่ 2-3 โครงการซอฟต์แวร์และองค์กรเท่านั้นที่อยู่ในระดับ 4 และ 5 ทำให้เป็นการยากที่จะสรุปคุณลักษณะโดยทั่วไปขององค์กรในระดับ 4 และ 5 คุณลักษณะ

ของระดับเหล่านี้ถูกกำหนดโดยอาศัยการเปรียบเทียบอุปมาอุปมัยกับอุตสาหกรรมอื่น และจากตัวอย่างเพียง 2-3 แห่งของอุตสาหกรรมซอฟต์แวร์ เพื่อแสดงถึงความสามารถของกระบวนการในระดับนี้

คุณลักษณะต่างๆของระดับที่ 4 และ 5 อาศัยพื้นฐานของการควบคุมกระบวนการในเชิงสถิติ (Statistical Process Control) ตามที่อธิบายในรูปภาพด้านล่าง แผนผังองค์ประกอบ 3 อย่างของจูราน (Juran Trilogy Diagram) แสดงให้เห็นถึงเป้าหมายพื้นฐานของการจัดการกระบวนการ



J.M. Juran Wilten,CT Used with the express permission of the Juran Institute ,Aug.1990

รูปภาพ 2.2 แสดงแผนผังองค์ประกอบ 3 อย่างของจูราน

จูราน แบ่งการจัดการที่มีคุณภาพออกเป็น 3 กระบวนการจัดการขั้นพื้นฐาน วัตถุประสงค์ของการวางแผนด้านคุณภาพ เพื่อที่จะจัดเตรียมอำนาจที่บังคับให้กระทำ (Operating Forces) ตัวอย่างเช่น ผู้ผลิตซอฟต์แวร์ จะหาวิธีการในการผลิตผลิตภัณฑ์นั้นให้สามารถตรงกับความต้องการของลูกค้าได้ แม้ว่าอำนาจที่บังคับให้กระทำจะช่วยในการผลิตผลิตภัณฑ์ แต่งานบางอย่างต้องทำใหม่เพราะว่าเกิดข้อบกพร่องในด้านคุณภาพ การสิ้นเปลือง(ความเสียหาย)นี้เกิดขึ้นอย่างเรื้อรัง เพราะว่ากระบวนการถูกวางแผนมาเช่นนั้น การควบคุมคุณภาพ (Quality Control) ได้ถูกนำมาใช้เพื่อป้องกันไม่ให้เกิดความเสียหายมากกว่าเดิม กราฟที่สูงขึ้นเป็นระยะๆ (Sporadic Spikes) ในกระบวนการดังที่แสดงในภาพ จะแสดงถึงกิจกรรมในการต่อต้านเป็นระยะๆ (Fire Fighting Activities) โดยที่การสิ้นเปลือง (ความเสียหาย) ที่เกิดขึ้นอย่างเรื้อรังจะนำมาซึ่งโอกาสในการปรับปรุงพัฒนา ซึ่งหมายถึงการปรับปรุงพัฒนาทางด้านคุณภาพ

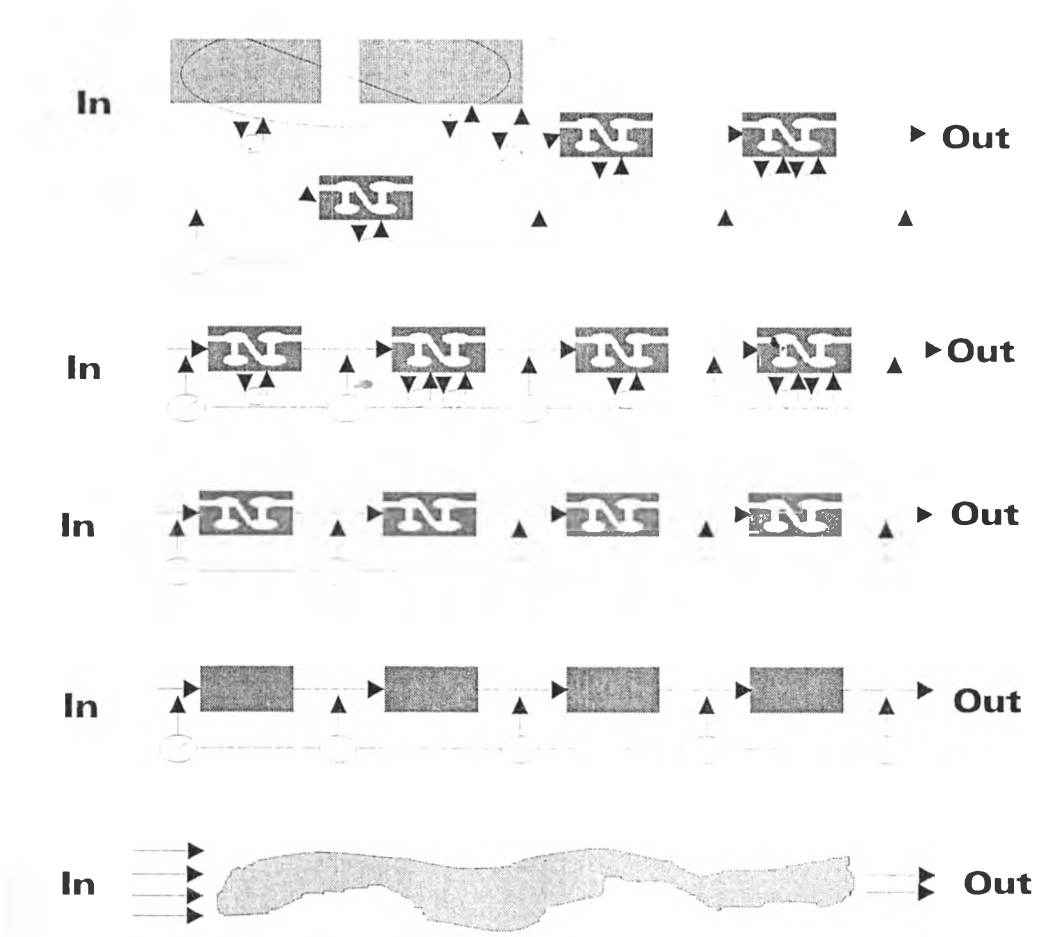
หน้าที่รับผิดชอบอย่างแรก และเป็นเป้าหมายของระดับที่ 4 คือ การควบคุมกระบวนการ (Process Control) กระบวนการซอฟต์แวร์ได้ถูกจัดการเพื่อให้ดำเนินการอย่างคงที่ในพื้นที่ของการควบคุมคุณภาพ (Zone of Quality Control) ซึ่งยังอาจมีการสิ้นเปลือง เสียหายที่เรื้อรัง เกิดขึ้นบ้าง และมีการพุ่งขึ้นของกราฟเป็นระยะๆ เมื่อมีการวัดผลงานซึ่งต้องการการควบคุม แต่ระบบโดยรวมจะอยู่ในสภาวะที่คงที่ ในดอนนี้พื้นฐานเกี่ยวกับการควบคุมความเปลี่ยนแปลงที่เป็นสาเหตุพิเศษ (Special Cause)จะเข้ามามีบทบาท ด้วยเหตุที่กระบวนการคงที่และวัดผลงานได้เมื่อมีเหตุการณ์พิเศษต่างๆเกิดขึ้น จึงทำให้สาเหตุพิเศษของความเปลี่ยนแปลงเหล่านั้นจะสามารถทราบและระบุได้

หน้าที่รับผิดชอบอย่างที่ 2 และเป็นเป้าหมายของระดับที่ 5 คือ การปรับปรุงคุณภาพกระบวนการอย่างต่อเนื่อง (Continuous Process Improvement) กระบวนการซอฟต์แวร์ได้ถูกเปลี่ยนแปลงโดยมีการปรับปรุงด้านคุณภาพ และพื้นที่ของการควบคุมคุณภาพได้เปลี่ยนไป พื้นฐานใหม่สำหรับการวัดประสิทธิภาพของงานได้ถูกสร้างขึ้นเพื่อลดปริมาณความเสียหายเรื้อรังที่เกิดขึ้น บทเรียนที่ใช้ในการเรียนรู้ (Lessons Learned) ในการปรับปรุงพัฒนากระบวนการได้ถูกนำประยุกต์ใช้ในแผนงานของกระบวนการในอนาคต พื้นฐานเกี่ยวกับการควบคุมความเปลี่ยนแปลงที่เป็นสาเหตุธรรมดา (Common Causes) จะเข้ามามีบทบาท ความเสียหาย สิ้นเปลืองจะกลายเป็นสิ่งที่ยอมรับไม่ได้ องค์กรพยายามที่จะกำจัดผลของความเสียหาย สิ้นเปลือง โดยการเปลี่ยนแปลงระบบ ตัวอย่างเช่น ปรับปรุงกระบวนการโดยการเปลี่ยนแปลงสาเหตุธรรมดาของการทำให้ไม่มีประสิทธิภาพ ไปเป็นการป้องกันไม่ให้เกิดความเสียหาย สิ้นเปลืองอีก

เราสามารถคาดได้ว่าองค์กรที่ไปถึงระดับการเติบโตสูงสุดของ ซีเอ็มเอ็ม จะมีกระบวนการที่มีความสามารถในการผลิตซอฟต์แวร์ที่เชื่อถือได้ในที่สุด โดยจะมีต้นทุนที่สามารถคาดการณ์ได้และตารางเวลาที่จำกัด ซีเอ็มเอ็มได้ความคิดเกี่ยวกับกระบวนการมาจากสิ่งที่เกิดขึ้นในกิจการผลิตทางด้านอุตสาหกรรม แต่ส่วนสำคัญของกระบวนด้านซอฟต์แวร์ไม่ได้ถอดแบบมาจากกระบวนการผลิตทางด้านอุตสาหกรรม แต่จะเน้นไปที่ปัจจัยด้านการออกแบบ (Design Issues) และ กิจกรรมทางด้านเพิ่มเติมความรู้ (Knowledge - Intensive Activity)

5. วิสัยทัศน์ของกระบวนการซอฟต์แวร์ (Visibility Into the Software Process)

รูปภาพด้านล่างแสดงถึง ระดับของวิสัยทัศน์ของสถานะโครงการและความสามารถในการจัดการ ของแต่ละระดับการเติบโตกระบวนการ ความสำเร็จของแต่ละขั้นการเติบโตที่เพิ่มขึ้นทีละน้อย จะนำมาซึ่งวิสัยทัศน์ที่ดีขึ้นเรื่อยๆของกระบวนการซอฟต์แวร์



รูปภาพ 2.3 แสดง วิสัยทัศน์ของกระบวนการซอฟต์แวร์ในแต่ละระดับการเติบโต

ในระดับที่ 1 กระบวนการซอฟต์แวร์ที่เกิดขึ้นมีรูปร่างไม่แน่นอน - เหมือนกับกล่องดำ วิสัยทัศน์ของกระบวนการของโครงการถูกจำกัด เมื่อระยะต่างๆของกิจกรรมถูกกำหนดขึ้นได้ยาก ผู้จัดการมักจะประสบกับความยากลำบากในการทราบถึงสถานะของความก้าวหน้าของโครงการและกิจกรรม ซึ่งนำไปสู่กฎ 90-90 (Ninety - Ninety Rule) ซึ่งกล่าวว่า 90 % ของโครงการจะสำเร็จก็ต่อเมื่อกินเวลาไป 90 % ของเวลาทั้งหมด การพัฒนาซอฟต์แวร์ส่วนมากเหมือนกับเวทย์มนต์ดำ (Black Magic) โดยเฉพาะกับผู้จัดการที่ไม่คุ้นเคยกับซอฟต์แวร์

ในระดับที่ 2 ความต้องการของลูกค้าและผลงานของผลิตภัณฑ์ สามารถควบคุมได้ และการฝึกฝนปฏิบัติด้านการจัดการโครงการขั้นพื้นฐานได้ถูกเริ่มต้น การควบคุมจัดการเหล่านี้ทำให้วิสัยทัศน์ของโครงการสามารถกำหนดได้เป็นครั้งคราว กระบวนการในการจัดทำซอฟต์แวร์เป็นเสมือนลำดับของกล่องดำ ซึ่งยอมให้ฝ่ายจัดการมองเห็นได้ในช่วงจุดต่อในขณะที่กิจกรรมดำเนินไประหว่างกล่อง

ต่าง ๆ (แต่ละขั้นของโครงการ) แม้ว่าฝ่ายจัดการบางทีไม่สามารถทราบรายละเอียดว่ามีอะไรเกิดขึ้นในกล่องนั้น ๆ แต่ผลผลิตของกระบวนการและจุดตรวจ (Checkpoints) เพื่อให้มั่นใจว่ากระบวนการนั้นดำเนินการอยู่ สามารถทราบและระบุได้ ฝ่ายจัดการจึงสามารถตอบโต้กับปัญหาได้ทันทีที่เกิดขึ้น

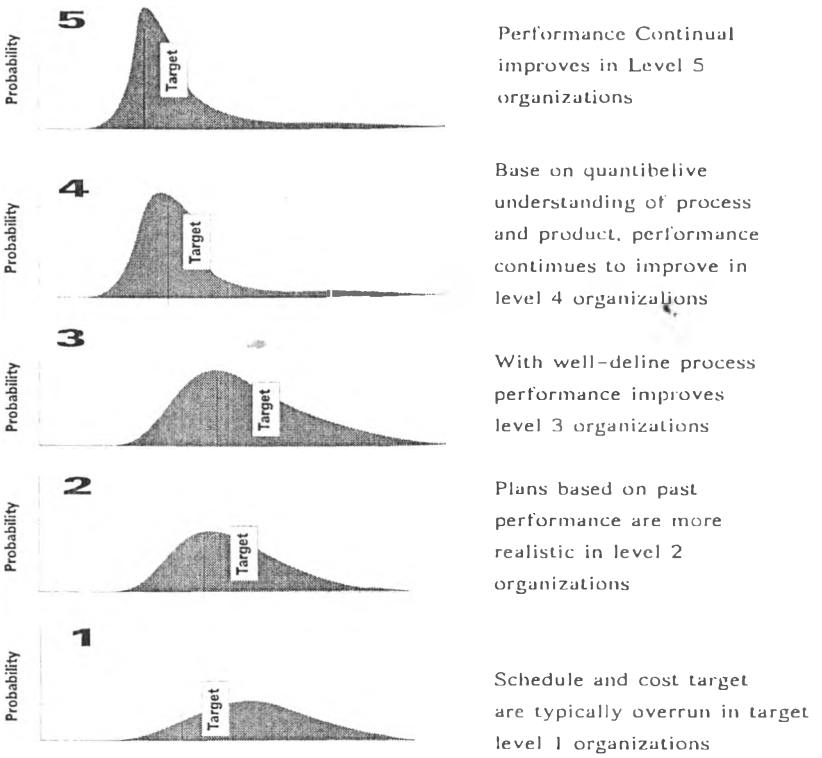
ในระดับที่ 3 โครงสร้างภายในของแต่ละกล่อง ตัวอย่างเช่น งานต่าง ๆในกระบวนการซอฟต์แวร์ที่ถูกกำหนดไว้ของโครงการจะสามารถมองเห็นได้ โครงสร้างภายในแสดงให้เห็นถึงวิธีการที่มาตรฐานกระบวนการซอฟต์แวร์ขององค์กรได้ถูกนำไปประยุกต์ใช้ในโครงการ ทั้งผู้จัดการและวิศวกร จะเข้าใจบทบาทและหน้าที่ความรับผิดชอบในกระบวนการ และวิธีการที่จะดำเนินการในรายละเอียดของแต่ละระดับให้เหมาะสม ฝ่ายจัดการจะต้องเตรียมการเกี่ยวกับความเสี่ยงที่อาจเกิดขึ้น บุคคลภายนอกโครงการสามารถได้รับทราบถึงสถานะของโครงการอย่างแม่นยำถูกต้องและรวดเร็ว เพราะว่า กระบวนการต่าง ๆที่ถูกกำหนดขึ้นทำให้เห็นภาพที่ชัดเจนในกิจกรรมต่าง ๆของโครงการ

ในระดับที่ 4 กระบวนการซอฟต์แวร์ต่าง ๆที่ถูกกำหนดขึ้น จะนำไปใช้เป็นเครื่องมือในการควบคุมเชิงปริมาณ ผู้จัดการจะสามารถประเมินความก้าวหน้าของกระบวนการและปัญหาต่าง ๆได้ พวกเขามีเป้าหมายพื้นฐานเชิงปริมาณสำหรับการตัดสินใจ และมีความสามารถในการคาดการณ์ผลลัพธ์ที่ได้จากการดำเนินไปของกระบวนการถูกต้องมากขึ้น เมื่อเทียบกับความเปลี่ยนแปลงในกระบวนการที่จะถูกพบน้อยลง

ในระดับที่ 5 วิธีการในการสร้างซอฟต์แวร์วิธีใหม่ที่ถูกปรับปรุงแล้ว ได้ถูกนำมาทดลองใช้อย่างต่อเนื่อง เพื่อควบคุมให้เกิดการพัฒนาทางด้านคุณภาพและประสิทธิภาพ กิจกรรมที่ไม่มีประสิทธิภาพหรือที่ก่อให้เกิดความเสียหายจะถูกแทนที่หรือถูกทบทวนใหม่ วิสัยทัศน์จะขยายจากกระบวนการที่เกิดขึ้นไปยังผลกระทบของการเปลี่ยนแปลงที่มีศักยภาพของกระบวนการ ผู้จัดการจะสามารถประเมินผลและติดตามตรวจสอบในเชิงปริมาณต่อผลกระทบและประสิทธิภาพของการเปลี่ยนแปลงได้

6. ความสามารถของกระบวนการและการคาดการณ์ผลการปฏิบัติงาน (Process Capability and the Prediction of Performance)

การเติบโตของกระบวนการซอฟต์แวร์ขององค์กร ช่วยในการคาดการณ์ความสามารถของโครงการในการบรรลุยังเป้าหมาย โครงการในระดับที่ 1 องค์กรจะประสบกับการเปลี่ยนแปลงอย่างมากในการที่จะบรรลุด้านต้นทุน ตารางการทำงาน หน้าที่รับผิดชอบต่าง ๆ และเป้าหมายในด้านคุณภาพ จากภาพด้านล่างจะแสดงถึงวิธีการปรับปรุง 3 ประการในการที่จะบรรลุเป้าหมาย ซึ่งก็คือการเติบโตเต็มที่ของกระบวนการซอฟต์แวร์ขององค์กร



รูปภาพ 2.4 แสดง ความสามารถของกระบวนการในแต่ละระดับการเติบโต

ประการแรก เมื่อการเติบโตเพิ่มขึ้น ความแตกต่างระหว่างผลลัพธ์ที่ตั้งเป้าไว้ และผลลัพธ์ที่เกิดขึ้นจริงจะค่อยๆ ลดน้อยลงตลอดทั่วทั้งโครงการ ตัวอย่างเช่น ถ้าโครงการ 10 โครงการที่มีขนาดเดียวกันตั้งเป้าว่า จะส่งมอบภายในวันที่ 1 พฤษภาคม ฉะนั้นวันที่ในการส่งมอบโดยเฉลี่ย ควรจะเลื่อนเข้าใกล้วันที่ 1 ไปเรื่อยๆ ตามการเติบโตขององค์กร องค์กรในระดับที่ 1 มักจะส่งมอบในวันที่ที่คาดเคลื่อนจากเดิมเป็นระยะห่างที่ค่อนข้างกว้าง ในขณะที่องค์กรในระดับ 5 จะสามารถบรรลุเป้าหมายตรงตามวันที่ที่ตั้งไว้อย่างเที่ยงตรง นั้นเป็นเพราะว่า องค์กรระดับ 5 ใช้การดำเนินการกระบวนการซอฟต์แวร์ ที่มีโครงสร้างอย่างระมัดระวังภายใต้ปัจจัยต่างๆ ที่คุ้นเคย และจะอาศัยข้อมูลที่มีอยู่เกี่ยวกับกระบวนการและความสามารถขององค์กรในการคัดเลือกวันส่งมอบ

ประการที่ 2 เมื่อการเติบโตเพิ่มขึ้นความเปลี่ยนแปลงของผลลัพธ์ที่เกิดขึ้นจริงเมื่อเทียบกับผลลัพธ์ที่ตั้งเป้าไว้ลดลง ตัวอย่างเช่นในระดับที่ 1 องค์กรไม่สามารถคาดการณ์ และมักจะเกิดช่วงห่างอย่างมากของวันที่ส่งมอบสำหรับโครงการที่มีขนาดใกล้เคียงกัน แต่สำหรับในองค์กรระดับที่ 5 โครงการที่คล้ายกัน จะถูกส่งมอบในช่วงเวลาที่ใกล้เคียงกัน (ช่วงห่างน้อยกว่า) ช่วงความเปลี่ยนแปลงที่

แคบลงนี้จะเกิดขึ้นในระดับการเติบโตสูงสุด เพราะว่าโครงการต่างๆ ดำเนินการภายใต้ปัจจัยที่ควบคุมเพื่อไปสู่ความสามารถขององค์กรในด้านต้นทุน ตารางการทำงาน หน้าที่รับผิดชอบ และคุณภาพ

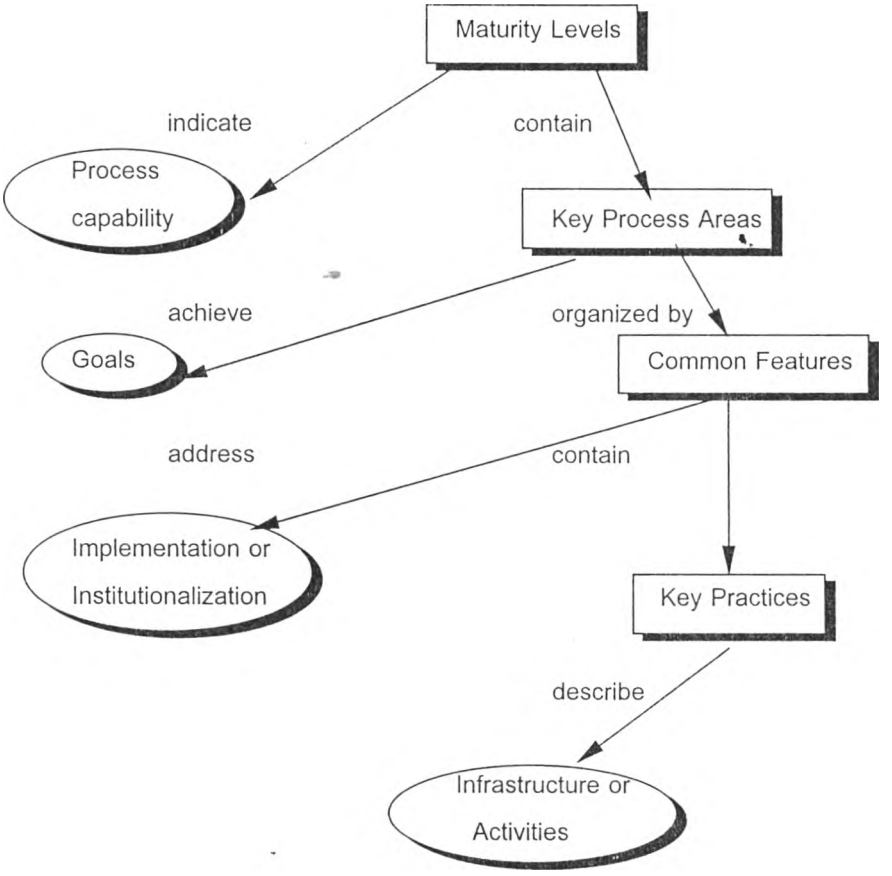
ประการที่ 3 ผลลัพธ์ตามเป้าหมายถูกปรับปรุงพัฒนาตามการเติบโตขององค์กรที่เพิ่มขึ้น นั่นคือเมื่อองค์กรซอฟต์แวร์เติบโตขึ้น ต้นทุนจะลดลง เวลาที่ใช้ในการพัฒนาจะสั้นลง ประสิทธิภาพและคุณภาพจะเพิ่มขึ้น ในระดับที่ 1 องค์กรจะใช้เวลาในการพัฒนาค่อนข้างนาน เพราะว่าต้องมีการทำงานซ้ำเพื่อแก้ไขข้อผิดพลาด ในทางตรงกันข้ามในระดับที่ 5 องค์กรจะใช้การปรับปรุงพัฒนากระบวนการที่ต่อเนื่อง และเทคนิคในการป้องกันความเสียหายผิดพลาดเพื่อเพิ่มประสิทธิภาพให้กระบวนการ และกำจัดต้นทุนในการทำงานซ้ำ ซึ่งเป็นผลทำให้เวลาที่ใช้ในการพัฒนาสั้นลง

<u>ประโยชน์ของ ซีเอ็มเอ็ม</u>

ใช้บ่งชี้จุดแข็ง จุดอ่อนในองค์กร ใช้ประเมินความเสี่ยงในการเลือกทำธุรกิจกับบริษัทต่าง ๆ และเพื่อที่จะคอยตรวจตราดูข้อสัญญาต่าง ๆ ผู้จัดการและฝ่ายเทคนิคจะใช้ซีเอ็มเอ็มเพื่อทำความเข้าใจกิจกรรมที่จำเป็นเพื่อวางแผนและบำรุงรักษาโครงการปรับปรุงพัฒนากระบวนการซอฟต์แวร์ขององค์กร และเป็นแนวทางช่วยในการปรับปรุงพัฒนากระบวนการซอฟต์แวร์ต่อไป

หอสมุดกลาง สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

โครงสร้างของซีเอ็มเอ็ม (THE CMM STRUCTURE)



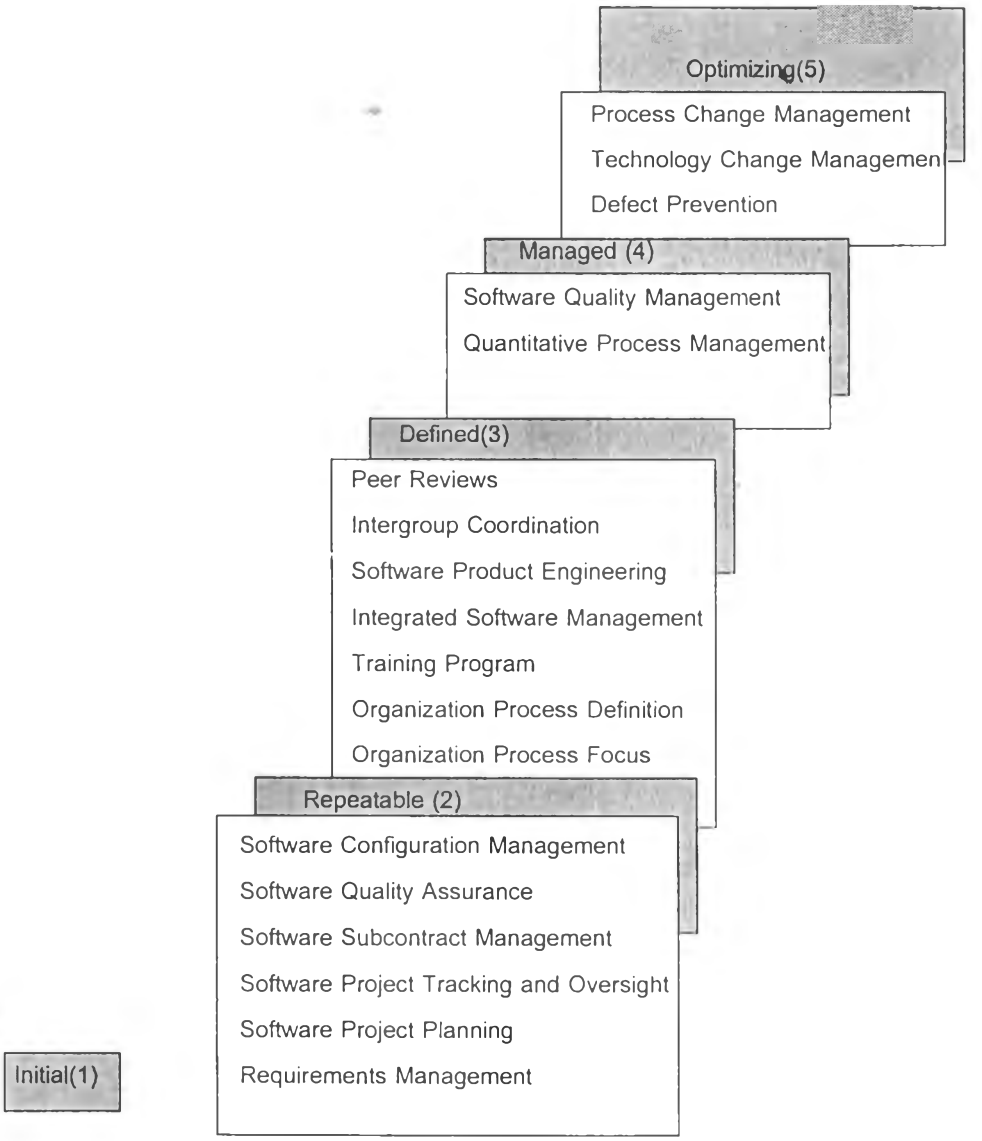
ระดับของการเติบโต (Maturity Levels) แต่ละระดับจะประกอบด้วยส่วนประกอบย่อยๆ ก็คือ วิธีปฏิบัติหลัก (Key Practices) ซึ่งแต่ละระดับการเติบโตจะประกอบด้วยหลายๆ กระบวนการหลัก (Key Process Area) ซึ่งสามารถแบ่งออกได้เป็นส่วนย่อยๆ อีก 5 ส่วนเรียกว่า คุณสมบัติพื้นฐาน (Common Features) ซึ่งจะบ่งชี้ถึงวิธีปฏิบัติหลักที่จะทำให้บรรลุไปยังเป้าหมายของแต่ละระดับการเติบโต

1. ระดับของการเติบโต (Maturity Levels)

เป็นระดับที่เปลี่ยนแปลงเพิ่มขึ้นทีละน้อย เพื่อที่จะบรรลุไปยังกระบวนการเติบโตของซอฟต์แวร์ ซึ่งจะบ่งชี้ถึงความสามารถของกระบวนการในแต่ละระดับ

2. กระบวนการหลัก (Key Process Areas)

ในแต่ละระดับของการเติบโต จะประกอบด้วยหลายๆกระบวนการหลัก (ยกเว้นระดับที่ 1) ซึ่งจะเป็นแนวทางให้กับองค์กรว่าควรจะมุ่งเน้นไปยังองค์ประกอบ ปัจจัยใดบ้าง ในการปรับปรุงพัฒนากระบวนการซอฟต์แวร์ของตนเพื่อที่จะบรรลุยังการเติบโตในแต่ละระดับ



รูปภาพ 2.5 แสดง กระบวนการหลักตามระดับของการเติบโต

(The Key Process Areas By Maturity Level)

กระบวนการหลักตามระดับของการเติบโตของซีเอ็มเอ็มอธิบายว่าองค์กรจะเติบโตอย่างไรซึ่งอาศัยจากประสบการณ์ด้านวิศวกรรมซอฟต์แวร์และด้านการจัดการและประสบการณ์นานกว่า 5 ปี เกี่ยวกับการประเมินความสามารถของซอฟต์แวร์และกระบวนการซอฟต์แวร์

กระบวนการหลักของการเติบโตระดับ 2 (The Key Process Areas at Level 2) มุ่งเน้นไปยังโครงการซอฟต์แวร์ที่เกี่ยวข้องกับการจัดทำการควบคุมบริหารโครงการขั้นพื้นฐาน คุณสมบัติของแต่ละองค์ประกอบเป็นดังนี้

- **การจัดการความต้องการ (Requirements Management)** คือต้องการสร้างความเข้าใจพื้นฐานระหว่างลูกค้ากับโครงการซอฟต์แวร์ซึ่งข้อตกลงกับลูกค้าจะเป็นพื้นฐานสำหรับการวางแผนและการจัดการโครงการซอฟต์แวร์
- **การจัดแผนงานโครงการซอฟต์แวร์ (Software Project Planning)** เพื่อสร้างแผนงานเพื่อติดตามควบคุมการทำงานในส่วนวิศวกรรมซอฟต์แวร์และเพื่อจัดการโครงการซอฟต์แวร์ ซึ่งแผนงานเหล่านี้มีความสำคัญต่อการจัดการโครงการ ถ้าไม่มีแผนงานที่สามารถปฏิบัติได้จริง การบริหารงานโครงการที่มีประสิทธิภาพจะไม่สามารถทำได้
- **การติดตามและการควบคุมดูแลโครงการซอฟต์แวร์ (Software Project Tracking and Oversight)** เพื่อสามารถติดตามผลงานในขั้นตอนการดำเนินงานจริง ทำให้การจัดการสามารถกระทำอย่างมีประสิทธิภาพ หากผลการดำเนินงานของโครงการซอฟต์แวร์เบี่ยงเบนไปจากแผนงานที่วางไว้
- **การบริหารผู้ร่วมดำเนินงานซอฟต์แวร์ (Software Subcontract Management)** เพื่อที่จะคัดเลือกผู้ร่วมดำเนินงานที่มีคุณภาพและสามารถจัดการควบคุมได้อย่างมีประสิทธิภาพ
- **ความเชื่อถือในคุณภาพของซอฟต์แวร์ (Software Quality Assurance)** เพื่อเตรียมแนวทางในการจัดการที่เหมาะสมให้กับกระบวนการ
- **การบริหารโครงร่างของซอฟต์แวร์ (Software Configuration Management)** เพื่อสร้างและบำรุงรักษาความครบถ้วนของผลิตภัณฑ์ของโครงการซอฟต์แวร์ตลอดทั้งวงจรชีวิตของโครงการ (Project's Software Life Cycle)

กระบวนการหลักของการเติบโตระดับ 3 (The Key Process Areas at Level 3) มุ่งเน้นไปยังโครงการและองค์กร เมื่อองค์กรได้เริ่มจัดทำการบริหารกระบวนการซอฟต์แวร์อย่างมีประสิทธิภาพตลอดทั่วทั้งโครงการ คุณสมบัติของแต่ละองค์ประกอบเป็นดังนี้

- **การมุ่งเน้นไปยังกระบวนการขององค์กร (Organization Process Focus)** เพื่อสร้างความรับผิดชอบของคนในองค์กรต่อกิจกรรมกระบวนการซอฟต์แวร์ ซึ่งจะปรับปรุงพัฒนา

ความสามารถของกระบวนการซอฟต์แวร์ทั้งหมดขององค์กร ผลลัพธ์ขั้นต้นของการกระทำนี้ คือ ชุดของกิจกรรมกระบวนการซอฟต์แวร์ (Set of Software Process Assets)

- **การนิยามกระบวนการขององค์กร (Organization Process Definition)** เพื่อพัฒนาและบำรุงรักษาชุดของกิจกรรมกระบวนการซอฟต์แวร์ที่ใช้ในการปรับปรุงพัฒนาประสิทธิภาพของกระบวนการตลอดทั้งโครงการ
- **การจัดโปรแกรมสำหรับฝึกอบรม (Training Program)** เพื่อปรับปรุงพัฒนาทักษะและความรู้ของแต่ละบุคคล เพื่อที่จะสามารถปฏิบัติงานตามบทบาทหน้าที่ได้อย่างมีประสิทธิภาพ
- **การบริหารซอฟต์แวร์ให้สมบูรณ์ (Integrated Software Management)** เพื่อรวมวิศวกรรมซอฟต์แวร์และกิจกรรมด้านบริหารเข้าเป็นอันหนึ่งอันเดียวกัน
- **วิศวกรรมผลิตภัณฑ์ซอฟต์แวร์ (Software Product Engineering)** เพื่อดำเนินการด้านวิศวกรรมซอฟต์แวร์อย่างต่อเนื่องต่อกระบวนการซึ่งจะรวบรวมกิจกรรมทางด้านวิศวกรรมซอฟต์แวร์ เพื่อผลิตผลิตภัณฑ์ซอฟต์แวร์ที่ถูกต้องและมีประสิทธิภาพอย่างสม่ำเสมอ กิจกรรมด้านวิศวกรรมซอฟต์แวร์จะเกี่ยวข้องกับกิจกรรมด้านเทคนิคของโครงการ เช่น การวิเคราะห์ความต้องการของลูกค้า การออกแบบ การเขียนโปรแกรมและการทดสอบ
- **การร่วมมือระหว่างกลุ่ม (Intergroup Coordination)** เพื่อจัดทำแนวทางสำหรับกลุ่มของวิศวกรรมซอฟต์แวร์ให้มีส่วนร่วมกับกลุ่มอื่น เพื่อให้โครงการมีความสามารถที่จะตอบสนองความต้องการของลูกค้าอย่างมีประสิทธิภาพเพิ่มมากขึ้น
- **การตรวจสอบทบทวนการทำงาน (Peer Reviews)** เพื่อขจัดข้อผิดพลาดจากการทำงานซอฟต์แวร์ได้อย่างทันทีและมีประสิทธิภาพ มีการรวบรวมผลกระทบที่สำคัญ เพื่อพัฒนาความเข้าใจในงานและความบกพร่องต่าง ๆ ที่สามารถป้องกันได้ให้ดีขึ้น

กระบวนการหลักของการเติบโตระดับ 4 (The Key Process Areas at Level 4) มุ่งเน้นไปยังการสร้างความเข้าใจกระบวนการซอฟต์แวร์และจัดทำงานด้านซอฟต์แวร์ คุณสมบัติของแต่ละองค์ประกอบเป็นดังนี้

- **การบริหารกระบวนการในเชิงปริมาณ (Quantitative Process Management)** เพื่อควบคุมผลการปฏิบัติงานของกระบวนการโครงการซอฟต์แวร์ในเชิงปริมาณ ผลการปฏิบัติงานของกระบวนการซอฟต์แวร์แสดงให้เห็นถึงผลลัพธ์จริงที่ได้รับจากกระบวนการซอฟต์แวร์
- **การบริหารคุณภาพซอฟต์แวร์ (Software Quality Management)** เพื่อพัฒนาความเข้าใจในคุณภาพของผลิตภัณฑ์ซอฟต์แวร์ของโครงการในเชิงปริมาณและเพื่อที่จะบรรลุเป้าหมายด้านคุณภาพที่ตั้งไว้

กระบวนการหลักของการเติบโตระดับ 5 (The Key Process Areas at Level 5) ครอบคลุมปัจจัยที่องค์กรและโครงการต่าง ๆ ต้องให้ความสนใจในการบำรุงรักษาการปรับปรุงกระบวนการซอฟต์แวร์ที่ดำเนินไปอย่างต่อเนื่องคุณสมบัติของแต่ละองค์ประกอบเป็นดังนี้

- **การป้องกันความเสียหาย (Defect Prevention)** เพื่อบ่งชี้สาเหตุของความเสียหายและป้องกันการเกิดซ้ำ
- **การบริหารความเปลี่ยนแปลงทางเทคโนโลยี (Technology Change Management)** เพื่อบ่งชี้คุณประโยชน์ของเทคโนโลยีใหม่ (เครื่องมือ วิธีการและกระบวนการ) และนำไปใช้ในองค์กร
- **การบริหารการเปลี่ยนแปลงกระบวนการ (Process Change Management)** เพื่อปรับปรุงพัฒนากระบวนการซอฟต์แวร์ที่ใช้ในองค์กรอย่างต่อเนื่อง โดยมีจุดประสงค์เพื่อปรับปรุงคุณภาพของซอฟต์แวร์ เพิ่มประสิทธิภาพและลดระยะเวลา ในการพัฒนาผลิตภัณฑ์

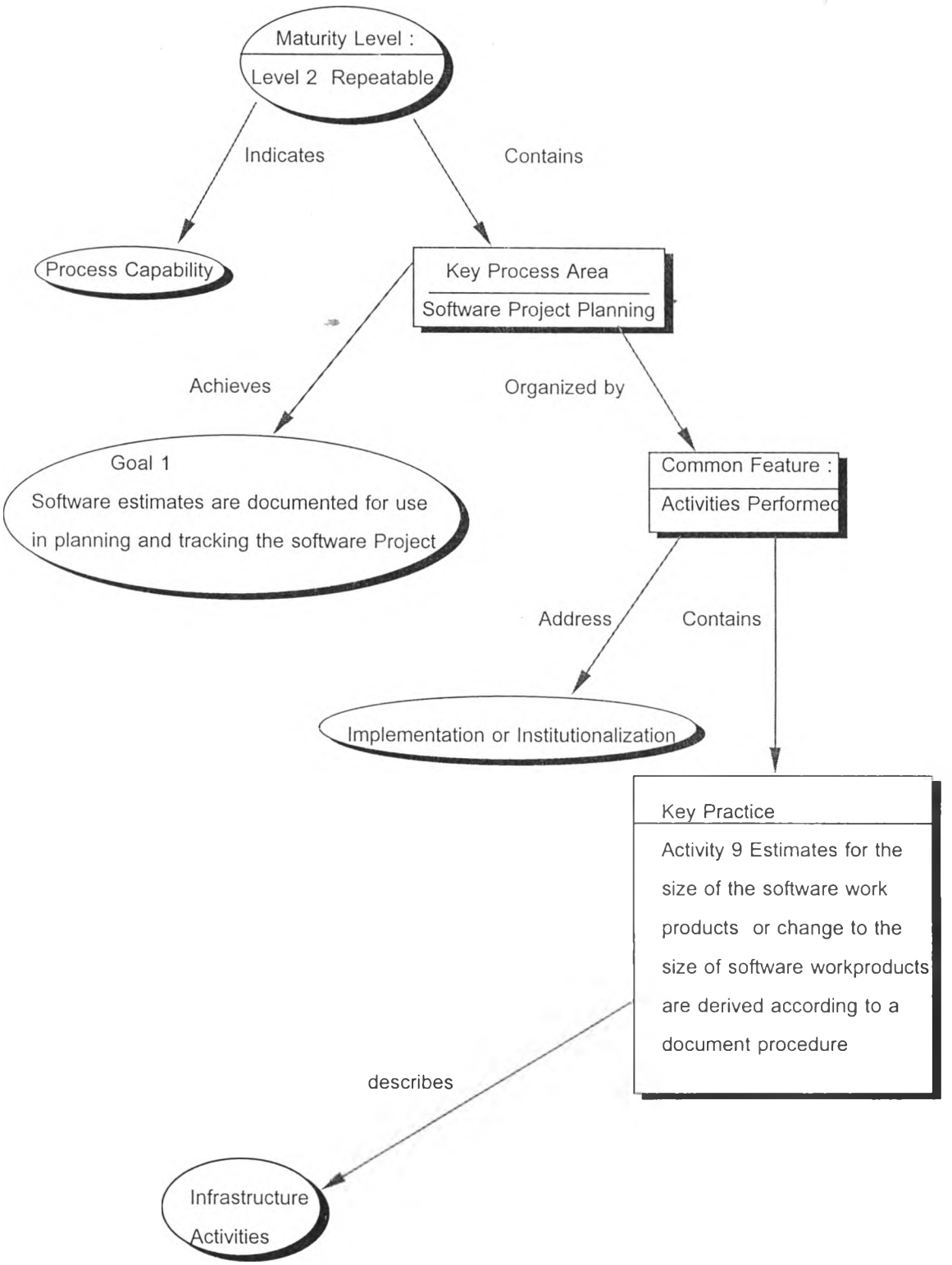
3. คุณสมบัติพื้นฐาน (Common Features)

กระบวนการหลัก ถูกจัดการโดยคุณสมบัติพื้นฐาน ซึ่งประกอบด้วย

- **สัญญาจะดำเนินการ (Commitment to Perform)** อธิบายถึงการกระทำที่องค์กรจะต้องทำ เพื่อให้มั่นใจว่ากระบวนการได้ถูกจัดทำขึ้นและสามารถปฏิบัติได้ ซึ่งจะรวมถึงการจัดทำนโยบายขององค์กร และการสนับสนุนของผู้บริหารระดับอาวุโส
- **ความสามารถในการดำเนินการ (Ability to Perform)** อธิบายถึงเงื่อนไขขั้นต้นที่จะเกิดขึ้นในโครงการหรือองค์กรเพื่อจะทำให้กระบวนการซอฟต์แวร์มีคุณสมบัติครบถ้วน ซึ่งได้แก่ ทรัพยากร โครงสร้างองค์กรและการฝึกอบรม
- **การดำเนินการ (Activities Performed)** อธิบายถึงบทบาทหน้าที่และขั้นตอนที่จำเป็นที่จะนำเอากระบวนการหลักไปใช้งาน จะรวมการจัดทำแผนงาน ขั้นตอน การดำเนินการติดตามตรวจสอบและกระทำการแก้ไขเท่าที่จำเป็น

4. วิธีปฏิบัติหลัก (Key Practices)

กระบวนการหลัก อธิบายในรูปแบบของวิธีปฏิบัติหลักที่จะทำให้บรรลุเป้าหมายซึ่งแต่ละวิธีปฏิบัติหลักประกอบด้วยประโยคเดียว และมักจะตามด้วยการอธิบายรายละเอียดซึ่งจะประกอบด้วยตัวอย่าง ที่จะกระทำในแต่ละระดับ



รูปภาพ 2.6 แสดง ตัวอย่างของวิธีปฏิบัติหลัก

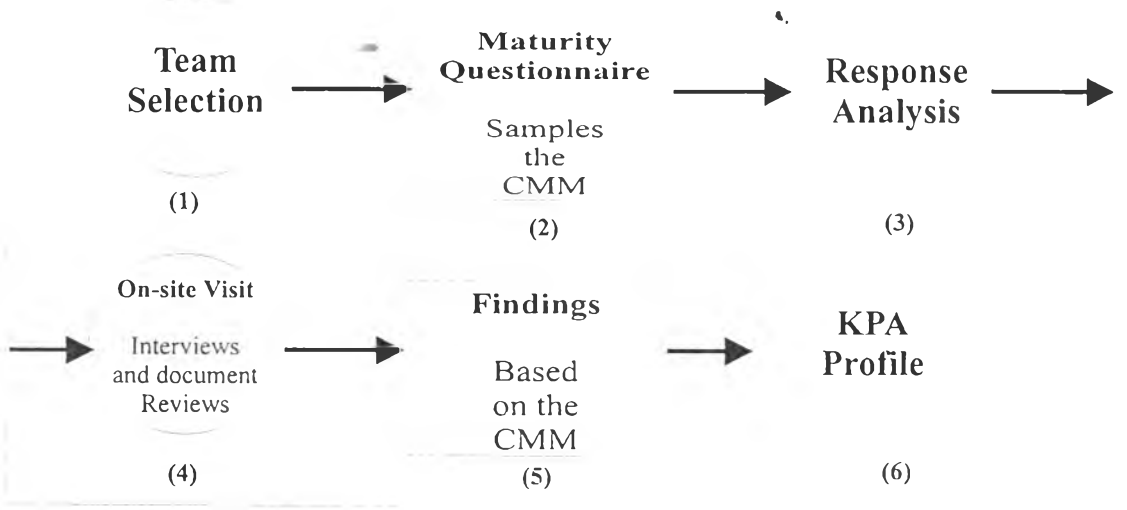
การนำซีเอ็มเอ็มไปประยุกต์ใช้ (Using the CMM)

ซีเอ็มเอ็มได้สร้างชุดของปัจจัยการเดิบโดทางซอฟต์แวร์ขององค์กรที่สามารถใช้กันโดยแพร่หลาย ซึ่งสามารถนำไปใช้ในกระบวนการพัฒนาและบำรุงรักษาซอฟต์แวร์ขององค์กร (Software Process Assessments) นอกจากนี้รัฐบาล หรือองค์กรธุรกิจยังสามารถนำไปใช้ในการประเมินความเสี่ยงในการทำสัญญากับบริษัทซอฟต์แวร์ในการจ้างทำโครงการซอฟต์แวร์ (Software Capability Evaluations) ได้อีกด้วย

<u>ขั้นตอนในการประเมินกระบวนการ และความสามารถขององค์กรซอฟต์แวร์</u> แบ่งออกเป็นขั้นตอนดังนี้ คือ

1. **การคัดเลือกทีมงาน (Team Selection)** ในขั้นตอนแรกจะต้องมีการคัดเลือกทีมงานที่จะมารับผิดชอบ เรื่องการทดสอบหรือประเมินกระบวนการ และความสามารถขององค์กรซอฟต์แวร์ ซึ่งทีมงานเหล่านี้ควรได้รับการฝึกอบรมเกี่ยวกับพื้นฐานแนวความคิดของซีเอ็มเอ็ม โดยเฉพาะเกี่ยวกับวิธีในการทดสอบ หรือประเมินกระบวนการมาอย่างดีรวมทั้งควรเป็นผู้ที่มีความรู้ความสามารถในด้านวิศวกรรมซอฟต์แวร์ และการจัดการ
2. **การตอบแบบทดสอบระดับการเติบโต (Maturity Questionnaire)** ในขั้นตอนนี้ตัวแทนจากองค์กรที่จะถูกประเมินจะต้องทำการตอบคำถามในแบบทดสอบระดับการเดิบโต
3. **วิเคราะห์คำตอบ (Response Analysis)** ในขั้นตอนนี้ทีมงานที่รับผิดชอบเรื่องการทดสอบและการประเมิน จะทำการวิเคราะห์คำตอบที่ได้จากขั้นตอนที่ 2 โดยการให้คะแนน และวิเคราะห์ผลรวมทั้งตั้งข้อสังเกตจุดที่ต้องการตรวจสอบเพิ่มเติม
4. **การไปตรวจสอบสถานที่ปฏิบัติงานจริง (On-Site Visit)** ในขั้นตอนนี้ ทีมงานที่รับผิดชอบ เรื่องการทดสอบและการประเมินจะไปตรวจสอบสถานที่ปฏิบัติงานจริงขององค์กรที่ถูกประเมิน ซึ่งจะมีการสัมภาษณ์ และการตรวจสอบ ทบทวนเอกสารต่างๆ ตามผลของการวิเคราะห์คำตอบโดยใช้กระบวนการหลักของการเดิบโดของซีเอ็มเอ็มเป็นแนวทางในการถามคำถามตรวจสอบวิเคราะห์ข้อมูล หลังจากนั้นทีมงานจะตัดสินว่าองค์กรอยู่ในระดับการเดิบโดใด และควรจะดำเนินการพัฒนาไปสู่เป้าหมายใด ซึ่งถ้าทีมงานผู้ทำการทดสอบ และองค์กรที่ถูกประเมินมีความคิดเห็นที่แตกต่างกัน ทีมงานผู้ทำการทดสอบจะต้องทำเอกสารชี้แจงรายละเอียด ผลการพิจารณา
5. **การรวบรวมข้อมูล (Finding)** ในขั้นตอนนี้ ทีมงานผู้ทำการทดสอบจะรวบรวมข้อมูลรายละเอียดที่ได้จากขั้นตอนที่ 4 จัดทำเป็นเอกสาร ซึ่งจะแสดงรายละเอียดจุดแข็ง และจุดอ่อนของกระบวนการซอฟต์แวร์ขององค์กร ซึ่งจะเป็นส่วนหนึ่งของพื้นฐานสำหรับคำแนะนำในการปรับปรุงกระบวนการ หรือกลายเป็นส่วนหนึ่งของการวิเคราะห์ความเสี่ยงซึ่งจัดทำโดยบริษัทที่ต้องการคัดเลือกจัดหาบริษัทผู้พัฒนาซอฟต์แวร์ภายนอกองค์กร

6. การรวบรวมข้อมูลกระบวนการหลักของการเติบโต (Key Process Area Profile) จะเป็นการจัดเตรียมข้อมูลเพื่อแสดงให้เห็นถึงรายละเอียดขององค์การที่ถูกประเมินว่ารายละเอียดใดที่มี หรือ ไม่มี และการบรรลุถึงเป้าหมายของกระบวนการหลักในระดับนั้นๆ เพราะว่าการรวบรวมข้อมูลในขั้นตอนที่ 5 ไม่ได้แสดงรายละเอียดปัญหาหลักที่ทำให้เป็นอุปสรรคในการที่จะบรรลุถึงเป้าหมายของแต่ละกระบวนการ



รูปภาพ 2.7 แสดงขั้นตอนในการประเมินกระบวนการ และความสามารถขององค์กรซอฟต์แวร์

<u>แบบทดสอบระดับการเติบโตขององค์กรซอฟต์แวร์</u>

แบบทดสอบด้านล่างเป็นแบบทดสอบระดับการเติบโตขององค์กรซอฟต์แวร์ตามเทคนิคในการประเมินกระบวนการของ SEI เพื่อให้องค์กรทราบว่าตนเองอยู่ในระดับใดของซีเอ็มเอ็ม

คำถามในการประเมินทั้งหมด มีหลักการพื้นฐานดังนี้ คือ

1. คุณภาพของซอฟต์แวร์ ส่วนมากจะเป็นผลเนื่องมาจากคุณภาพของกระบวนการที่พัฒนาซอฟต์แวร์นั้น
2. วิศวกรรมซอฟต์แวร์ เป็นกระบวนการซึ่งสามารถจัดการได้ วัดผลงาน และปรับปรุงพัฒนาอย่างต่อเนื่องได้
3. คุณภาพของกระบวนการซอฟต์แวร์ มีผลกระทบจากเทคโนโลยี ที่นำมาใช้สนับสนุนการดำเนินงาน
4. ระดับของเทคโนโลยีที่จะนำมาใช้ในวิศวกรรมซอฟต์แวร์ ควรจะต้องเหมาะสมกับการเติบโตของกระบวนการ

คำถามในการประเมินตามแบบทดสอบ จัดแบ่งออกเป็นกลุ่มต่าง ๆ ดังนี้ คือ

1. การจัดการองค์กร และทรัพยากร (Organization and Resource Management)

คำถามในส่วนนี้เกี่ยวกับความรับผิดชอบในหน้าที่งานต่างๆ บุคลากร ทรัพยากรที่ใช้ และสิ่งอำนวยความสะดวกต่างๆ จุดประสงค์เพื่อกำหนดขนาดความสำคัญ คุณภาพ และโครงสร้างขององค์กรด้านวิศวกรรมซอฟต์แวร์ คำถามต่างๆ จึงมุ่งเน้นในเรื่องความรับผิดชอบ คุณภาพ และปริมาณของทรัพยากรที่ใช้

1.1 โครงสร้างองค์กร (Organizational Structure)

1.1.1 ในแต่ละโครงการที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ มีการแต่งตั้งผู้จัดการซอฟต์แวร์ที่มีหน้าที่ความรับผิดชอบโครงการโดยเฉพาะ หรือไม่

1.1.2 ผู้จัดการโครงการซอฟต์แวร์รายงานขึ้นตรงกับผู้จัดการโครงการใหญ่ (ผู้จัดการพัฒนาโครงการ) หรือไม่

1.1.3 มีช่องทางในการรายงานผลการจัดการบริหารหน้าที่ในการสร้างความเชื่อถือในคุณภาพซอฟต์แวร์ (Software Quality Assurance - SQA) แยกจากการรายงานผลการจัดการบริหารโครงการพัฒนาซอฟต์แวร์หรือไม่

1.1.4 มีการแต่งตั้งบุคคลหรือทีมงานที่มีหน้าที่รับผิดชอบในการควบคุมการติดต่อเชื่อมโยงกับซอฟต์แวร์ (Software Interfaces) หรือไม่

1.1.5 งานทางด้านวิศวกรรมระบบซอฟต์แวร์ (Software System Engineering) อยู่ในความรับผิดชอบของทีมงานการออกแบบระบบ (System Design Team) หรือไม่

1.1.6 มีหน้าที่การควบคุมรายละเอียดขอบเขตโครงร่างของซอฟต์แวร์ (Software Configuration Control) สำหรับแต่ละโครงการรวมอยู่ในการพัฒนาซอฟต์แวร์ หรือไม่

1.1.7 มีการกำหนดกลุ่มวิศวกรรมซอฟต์แวร์ที่มีหน้าที่รับผิดชอบเกี่ยวกับกระบวนการที่ใช้ในการพัฒนาซอฟต์แวร์ (Process Group Software Engineering) หรือไม่

1.2 ทรัพยากรที่ใช้ บุคลากร และการฝึกอบรม (Resources Personnel and Training)

คำถามในส่วนนี้เกี่ยวข้องกับการฝึกอบรมทางด้านวิศวกรรมซอฟต์แวร์ การฝึกอบรมทางด้านกระบวนการต่างๆ และความเพียงพอของอุปกรณ์อำนวยความสะดวกที่ช่วยสนับสนุน

1.2.1 ผู้พัฒนาซอฟต์แวร์แต่ละคนมีคอมพิวเตอร์ส่วนตัวใช้เป็นเครื่องลูกของเครือข่าย (Workstation / Terminal) หรือไม่

1.2.2 มีการจัดโปรแกรมการฝึกอบรมสำหรับผู้จัดการพัฒนาโครงการที่ได้รับการแต่งตั้งใหม่ เพื่อให้มีความคุ้นเคยกับการบริหารโครงการซอฟต์แวร์ หรือไม่

1.2.3 มีการจัดโปรแกรมการฝึกอบรมด้านวิศวกรรมซอฟต์แวร์ที่จำเป็นสำหรับผู้พัฒนาซอฟต์แวร์ (Software Developers) หรือไม่

1.2.4 มีการจัดโปรแกรมการฝึกอบรมด้านวิศวกรรมซอฟต์แวร์ที่จำเป็นสำหรับหัวหน้าระดับต้นของทีมพัฒนาซอฟต์แวร์ (First Line Supervisors) หรือไม่

1.2.5 มีการจัดโปรแกรมการฝึกอบรมอย่างเป็นทางการสำหรับผู้ที่เป็นหัวหน้าในการตรวจสอบ ทบทวน การออกแบบ และการเขียนโปรแกรม (Design and Code Review Leaders) หรือไม่

1.3 การบริหารจัดการเทคโนโลยี (Technology Management)

คำถามในส่วนนี้เกี่ยวข้องกับวิธีการหรือเทคนิค ในการนำเอาเทคโนโลยีใหม่มาใช้ และการควบคุมการใช้เทคโนโลยีนั้นๆ

1.3.1 มีวิธีการหรือเทคนิคที่ใช้ในการบอกให้ทราบถึงรายละเอียดของเทคโนโลยีทางด้านวิศวกรรมซอฟต์แวร์ หรือไม่

1.3.2 มีวิธีการหรือเทคนิคใช้ในการประเมินเทคโนโลยีที่ใช้ในองค์กรเปรียบเทียบกับเทคโนโลยีที่มีอยู่ภายนอกหรือไม่

1.3.3 มีวิธีการ หรือเทคนิคที่ใช้เพื่อการตัดสินใจว่าเมื่อไรควรจะแทรกเทคโนโลยีใหม่ๆ เข้าไปยังกระบวนการพัฒนาซอฟต์แวร์

1.3.4 มีวิธีการ หรือเทคนิคที่ใช้ในการจัดการและสนับสนุนเทคโนโลยีใหม่ในเบื้องต้นหรือไม่

1.3.5 มีวิธีการหรือเทคนิคที่ใช้สำหรับบ่งบอก และเปลี่ยนแทนเทคโนโลยีที่ล้าสมัยหรือไม่

2. กระบวนการด้านวิศวกรรมซอฟต์แวร์ และการจัดการบริหาร กระบวนการ (Software Engineering Process and Its Management)

คำถามในส่วนนี้เกี่ยวกับขอบเขต รายละเอียด และความพร้อมของกระบวนการด้านวิศวกรรมซอฟต์แวร์ และวิธีการในการวัดผล การจัดการ และการปรับปรุงพัฒนากระบวนการ หัวข้อหลักได้แก่ มาตรฐาน และขั้นตอนการทำงาน การวัดผล การจัดการข้อมูล และการวิเคราะห์ และการควบคุมกระบวนการ

2.1 มาตรฐานของเอกสาร และขั้นตอนการทำงาน (Standards and Procedures)

คำถามในส่วนนี้จะเน้นถึงขอบเขต และการใช้งาน รูปแบบขั้นตอนการทำงาน และเอกสารต่างๆ ที่เกิดขึ้นระหว่างขั้นตอนการพัฒนาซอฟต์แวร์ เช่น ขั้นตอนการวิเคราะห์ความต้องการ การออกแบบ การเขียนโปรแกรม และการทดสอบ เป็นต้น

2.1.1 องค์กรที่ผลิตซอฟต์แวร์ ดำเนินการตามขั้นตอนกระบวนการพัฒนาซอฟต์แวร์ที่เป็นมาตรฐาน และเป็นเอกสารลายลักษณ์อักษรในแต่ละโครงการหรือไม่

2.1.2 เอกสารรายละเอียดขั้นตอนกระบวนการพัฒนาซอฟต์แวร์ที่เป็นมาตรฐาน อธิบายถึงรายละเอียดการใช้เครื่องมือ และเทคนิคต่างๆ เพื่อช่วยในการพัฒนาซอฟต์แวร์หรือไม่

2.1.3 มีขั้นตอนการทำงานที่เป็นทางการ ใช้ในการตรวจสอบการบริหารของแต่ละขั้นตอนในการพัฒนาซอฟต์แวร์ก่อนที่จะทำสัญญาข้อตกลงดำเนินการพัฒนาซอฟต์แวร์หรือไม่

2.1.4 มีขั้นตอนการทำงานที่เป็นทางการใช้ เพื่อให้เกิดความมั่นใจในการตรวจสอบสถานะของแต่ละโครงการพัฒนาซอฟต์แวร์ โดยผู้จัดการตามช่วงเวลาที่กำหนดหรือไม่

2.1.5 มีวิธีการ หรือเทคนิคที่ใช้สำหรับตรวจสอบบริษัทที่รับจ้างผลิตซอฟต์แวร์ (Subcontractors) เพื่อให้เกิดความมั่นใจเชื่อถือหรือไม่ ถ้ามีตรงตามหลักเกณฑ์กระบวนการของการพัฒนาซอฟต์แวร์หรือไม่

2.1.6 มีมาตรฐานที่ใช้ในการเก็บรวบรวมรายละเอียดของแฟ้มข้อมูลที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ (Software Development Files / Folders) หรือไม่

2.1.7 ในแต่ละโครงการ มีการดำเนินการตรวจสอบอิสระ (Independent Audits) ในแต่ละขั้นตอนของกระบวนการพัฒนาซอฟต์แวร์ หรือไม่

2.1.8 มีวิธีการหรือเทคนิคที่ใช้ในการประเมินการออกแบบ และประเมินรหัสโปรแกรมที่เคยมีอยู่แล้วเพื่อนำกลับมาใช้ในโปรแกรมระบบงานคอมพิวเตอร์ที่ทำขึ้นใหม่หรือไม่

2.1.9 มีการนำเอามาตรฐานของการเขียนรหัสโปรแกรม (Coding Standards) มาประยุกต์ใช้ในแต่ละโครงการพัฒนาซอฟต์แวร์หรือไม่

2.1.10 มีมาตรฐานที่ใช้ในการจัดเตรียมตัวอย่าง กรณีศึกษาต่างๆ ที่จะใช้ในการทดสอบระบบย่อยแต่ละส่วน (Unit Test Cases) หรือไม่

2.1.11 มีการประยุกต์ใช้ มาตรฐานในการบำรุงรักษารหัสโปรแกรม (Code Maintainability Standards) หรือไม่

2.1.12 มีการประยุกต์ใช้มาตรฐานในการตรวจสอบทบทวนการออกแบบภายในของซอฟต์แวร์ (Internal Design Review) หรือไม่

2.1.13 มีการประยุกต์ใช้มาตรฐานในการตรวจสอบทบทวนการเขียนรหัสโปรแกรม (Code Review) หรือไม่

2.1.14 มีขั้นตอนการทำงานที่เป็นทางการเพื่อใช้ในการประมาณขนาดของซอฟต์แวร์ (Estimates of Software Size) หรือไม่

2.1.15 มีขั้นตอนการทำงานที่เป็นทางการเพื่อใช้ในการกำหนดตารางเวลาการทำงานของการพัฒนาซอฟต์แวร์ (Software Development Schedules) หรือไม่

2.1.16 มีขั้นตอนการทำงานที่เป็นทางการนำมาประยุกต์ใช้ในการประมาณต้นทุนที่ใช้ในการพัฒนาซอฟต์แวร์ (Estimating Software Development Cost) หรือไม่

2.1.17 มีวิธีการหรือเทคนิคเพื่อให้แน่ใจว่าทีมออกแบบซอฟต์แวร์เข้าใจถึงความต้องการซอฟต์แวร์ แต่ละอย่าง (Software Requirement) หรือไม่

2.1.18 มีการประยุกต์ใช้มาตรฐานในการติดต่อเชื่อมโยงระหว่างผู้ใช้และคอมพิวเตอร์ (Man - Machine Interface Standards) เข้ากับแต่ละโครงการพัฒนาซอฟต์แวร์อย่างเหมาะสมหรือไม่

2.2 <u>การวัดผลกระบวนการ (Process Metrics)</u>

คำถามในส่วนนี้มุ่งเน้นถึงระดับของกระบวนการด้านวิศวกรรมซอฟต์แวร์ที่จะถูกวัดผล ซึ่งจะเกี่ยวข้องกับคุณภาพของซอฟต์แวร์ ปริมาณของจำนวนรหัสโปรแกรมที่พัฒนา ทรัพยากรที่ใช้ และสิ่งที่บ่งบองถึงความก้าวหน้าของกระบวนการ เช่น การตรวจสอบทบทวนผลการทดสอบ เป็นต้น

2.2.1 มีการเก็บข้อมูลพนักงานที่ใช้ไปในการพัฒนาซอฟต์แวร์จริงเปรียบเทียบกับข้อมูลจำนวนพนักงานตามแผนงาน หรือไม่

2.2.2 การประมาณขนาดของซอฟต์แวร์ ในแต่ละส่วนประกอบของโครงร่างซอฟต์แวร์ (Software Configuration Item) ใช้เวลาตามกำหนดที่ตั้งไว้หรือไม่

2.2.3 มีการเก็บรวบรวมสถิติ ของข้อผิดพลาดที่เกิดขึ้น ในการออกแบบซอฟต์แวร์ไว้หรือไม่

2.2.4 มีการเก็บรวบรวมสถิติของข้อผิดพลาดที่เกิดขึ้นในการเขียนโปรแกรม และการทดสอบไว้หรือไม่

2.2.5 มีการคาดการณ์ข้อผิดพลาดในการออกแบบที่อาจเกิดขึ้น และเปรียบเทียบกับสิ่งที่เกิดขึ้นจริงหรือไม่

2.2.6 มีการคาดการณ์ข้อผิดพลาดในการเขียนโปรแกรม และการทดสอบที่อาจเกิดขึ้นและเปรียบเทียบกับสิ่งที่เกิดขึ้นจริงหรือไม่

2.2.7 เวลาที่ใช้ไปจริงในการออกแบบแต่ละส่วนของซอฟต์แวร์ (Software Units Designed) เป็นไปตามแผนงานที่กำหนดไว้หรือไม่

2.2.8 การดำเนินการเพื่อให้แต่ละระบบย่อยของซอฟต์แวร์เสร็จสมบูรณ์ (Software Units Completing) การทดสอบระบบย่อย (Unit testing) ใช้เวลาจริงเป็นไปตามแผนงานที่กำหนดไว้หรือไม่

2.2.9 การรวบรวมระบบงานซอฟต์แวร์ในแต่ละส่วนย่อย (Software Units Integrated) ใช้เวลาจริงเป็นไปตามแผนงานที่กำหนดไว้หรือไม่

2.2.10 มีการติดตามตรวจสอบ การใช้ประโยชน์จากหน่วยความจำคอมพิวเตอร์ (Computer Memory Utilization) ทั้งที่ได้ตั้งเป้าไว้จากการประเมินและที่ใช้จริงหรือไม่

2.2.11 มีการติดตามตรวจสอบ การใช้ประโยชน์การนำเข้าผ่านคอมพิวเตอร์ (Computer Throughput Utilization) ทั้งที่ได้ตั้งเป้าไว้จากการประเมิน และที่ใช้จริงหรือไม่

2.2.12 มีการติดตามตรวจสอบ การใช้งานช่องทางการเข้าออกข้อมูลคอมพิวเตอร์

(Computer I/ O Channel Utilization) หรือไม่

2.2.13 มีการวัดระดับของการตรวจสอบ ทบทวนในขั้นตอนการออกแบบ และการเขียนโปรแกรม (Design And Code Review Coverage) และเก็บบันทึกรวบรวมไว้หรือไม่

2.2.14 มีการวัดผล และบันทึกจำนวนของรหัสโปรแกรมที่ถูกประมวลผลจริงในแต่ละขั้นของการทดสอบหน้าที่และคุณสมบัติของระบบงานหรือไม่

2.2.15 มีการติดตามรายละเอียดกิจกรรมต่างๆ ที่เป็นผลเนื่องมาจากการตรวจสอบทบทวนในขั้นตอนการออกแบบอย่างใกล้ชิดหรือไม่

2.2.16 มีการติดตามตรวจสอบรายงาน ปัญหาของซอฟต์แวร์ที่เกิดขึ้นจากการทดสอบระบบอย่างใกล้ชิดหรือไม่

2.2.17 มีการติดตามรายละเอียดกิจกรรมต่างๆ ที่เป็นผลเนื่องมาจากการตรวจสอบทบทวนในขั้นตอนการเขียนโปรแกรมอย่างใกล้ชิดหรือไม่

2.2.18 มีการติดตามความก้าวหน้าของการทดสอบ โดยหน่วยงานที่เกี่ยวข้องกับการส่งมอบซอฟต์แวร์ และมีการเปรียบเทียบกับแผนงานที่ตั้งไว้หรือไม่

2.2.19 มีการเก็บรวบรวมข้อมูลรายละเอียดการพัฒนาซอฟต์แวร์ และการเริ่มใช้ซอฟต์แวร์ เปรียบเทียบกับเวลาหรือไม่

2.3 การจัดการข้อมูล และการวิเคราะห์ข้อมูล (Data Management and Analysis)

การจัดการข้อมูลที่เกี่ยวข้องกับการจัดเก็บรวบรวม และการรักษาการวัดผลกระบวนการให้คงไว้ ซึ่งต้องการมาตรฐานของรายละเอียดคำจำกัดความของข้อมูล (Standardized Data Definitions) สิ่งอำนวยความสะดวกเพื่อช่วยในการจัดการข้อมูล และบุคลากร เพื่อให้มั่นใจว่าข้อมูลต่างๆ ได้รับมาถูกต้อง และมีการจัดการอย่างมีประสิทธิภาพ

2.3.1 มีการริเริ่มการจัดการ และควบคุมฐานข้อมูลของกระบวนการ เพื่อจัดเก็บข้อมูลการวัดผลกระบวนการที่ใช้ในทุกโครงการ

2.3.2 มีการวิเคราะห์ข้อมูลในการตรวจสอบทบทวน ที่เก็บรวบรวมได้ในระหว่างขั้นตอนการตรวจสอบทบทวนการออกแบบหรือไม่

2.3.3 มีการวิเคราะห์ข้อมูลความผิดพลาดต่างๆ จากการตรวจสอบทบทวนโปรแกรมและการทดสอบ เพื่อตัดสินว่ามีข้อผิดพลาดที่มีลักษณะคล้ายๆ กันนั้นหลงเหลืออยู่ในผลิตภัณฑ์ซอฟต์แวร์หรือไม่

2.3.4 มีการดำเนินการวิเคราะห์ความผิดพลาดต่างๆ เพื่อที่จะวิเคราะห์หาสาเหตุต่างๆ ที่เกี่ยวข้องกับกระบวนการซอฟต์แวร์ หรือไม่

2.3.5 มีวิธีการหรือเทคนิคที่ใช้สำหรับการวิเคราะห์หาสาเหตุของความผิดพลาดที่เกิดขึ้น (Error - Cause Analysis) หรือไม่

2.3.6 มีการตรวจสอบทบทวนสาเหตุของความผิดพลาดที่เกิดขึ้น เพื่อช่วยในการตัดสินว่าควรจะมีการเปลี่ยนแปลงกระบวนการหรือขั้นตอน เพื่อที่จะป้องกันไม่ให้เกิดความผิดพลาดขึ้นอีกหรือไม่

2.3.7 มีวิธีการหรือ เทคนิคที่ใช้สำหรับดำเนินการเพื่อป้องกันความผิดพลาดที่อาจเกิดขึ้นตั้งแต่เริ่มต้นหรือไม่

2.3.8 มีการวิเคราะห์ประสิทธิภาพของการตรวจสอบ ทบทวนในแต่ละโครงการหรือไม่

2.3.9 มีการวิเคราะห์ประสิทธิภาพของซอฟต์แวร์ ในแต่ละขั้นตอนของกระบวนการหรือไม่

2.4 การควบคุมกระบวนการ (Process Control)

คำถามในส่วนนี้เกี่ยวข้องกับการกำหนดรายละเอียดกระบวนการพัฒนาซอฟต์แวร์ และวิธีการหรือเทคนิคที่ใช้ในการระบุถึงปัญหาต่างๆ ของกระบวนการ การแก้ไขข้อมูลบกพร่องของกระบวนการ และการป้องกันไม่ให้เกิดปัญหาขึ้นอีก

2.4.1 มีวิธีการ หรือเทคนิคเพื่อให้ผู้จัดการอาวุโสตรวจสอบทบทวน สถานะของโครงการพัฒนาซอฟต์แวร์ อย่างสม่ำเสมอหรือไม่

2.4.2 มีวิธีการ หรือเทคนิคที่ใช้ในการประเมินตามช่วงระยะเวลาว่ากระบวนการวิศวกรรมซอฟต์แวร์ และการนำไปใช้งานมีการปรับปรุงพัฒนาหรือไม่

2.4.3 มีวิธีการ หรือเทคนิคที่ใช้ในการระบุและตัดสินปัจจัยต่างๆ ทางด้านวิศวกรรมซอฟต์แวร์ว่ามีผลกระทบกับซอฟต์แวร์หรือไม่

2.4.4 มีวิธีการหรือเทคนิคที่ใช้ในการนำเอาปัจจัยต่างๆ ที่ใช้ในการรวบรวมระบบงานซอฟต์แวร์ และการทดสอบที่เคยอยู่เป็นอิสระให้เข้ามาอยู่ในความสนใจดูแลของผู้จัดการโครงการ

2.4.5 มีวิธีการหรือเทคนิคที่ใช้สำหรับการแลกเปลี่ยนทางด้านเทคนิคระหว่างผู้พัฒนาซอฟต์แวร์กับผู้ใช้อย่างสม่ำเสมอหรือไม่

2.4.6 มีวิธีการหรือเทคนิคที่ใช้เพื่อให้มั่นใจว่าทุกคนเห็นด้วยกับมาตรฐานของวิศวกรรมซอฟต์แวร์หรือไม่

2.4.7 มีการตรวจสอบอนุมัติตารางเวลาการทำงาน และการประมาณต้นทุนของโครงการโดยผู้จัดการพัฒนาซอฟต์แวร์ ระดับต้นหรือไม่

2.4.8 มีวิธีการหรือเทคนิคที่ใช้ในการติดตามตรวจสอบการดำเนินการระหว่างขั้นตอนการวิเคราะห์ความต้องการซอฟต์แวร์ (Software Requirements) และการออกแบบขั้นต้น (Top Level Design) หรือไม่

2.4.9 มีวิธีการหรือเทคนิคที่ใช้ในการควบคุมการเปลี่ยนแปลงความต้องการซอฟต์แวร์หรือไม่

2.4.10 มีขั้นตอนการจัดการที่เป็นทางการ เพื่อใช้ในการตัดสินว่าแบบจำลองหน้าที่ต่างๆ ของซอฟต์แวร์ (Prototyping of Software Functions) เป็นส่วนที่เหมาะสมของขั้นตอนการออกแบบหรือไม่

2.4.11 มีวิธีการหรือเทคนิคที่ใช้ในการติดตามตรวจสอบการดำเนินงานระหว่างขั้นตอนการออกแบบขั้นต้น (Top Level Designs) และการออกแบบรายละเอียด (Detailed Designs) หรือไม่

2.4.12 มีการดำเนินการตรวจสอบทบทวนการออกแบบภายในซอฟต์แวร์ (Internal Software Design) หรือไม่

2.4.13 มีวิธีการหรือเทคนิคที่ใช้ในการควบคุมการเปลี่ยนแปลงการออกแบบซอฟต์แวร์หรือไม่

2.4.14 มีวิธีการหรือเทคนิคที่ใช้ในการติดตามตรวจสอบการดำเนินงานระหว่างขั้นตอนการออกแบบรายละเอียดซอฟต์แวร์ (Software Detailed Design) และการเขียนโปรแกรม (Coding) หรือไม่

2.4.15 มีการเก็บบันทึกรายละเอียดความก้าวหน้าของการพัฒนาระบบงานซอฟต์แวร์ในแต่ละส่วนย่อยๆ ไว้อย่างเป็นทางการหรือไม่

2.4.16 มีการดำเนินการตรวจสอบทบทวนการเขียนโปรแกรมซอฟต์แวร์หรือไม่

2.4.17 มีวิธีการหรือเทคนิคที่ใช้ในการควบคุมการเปลี่ยนแปลงแก้ไขโปรแกรมหรือไม่ (ใครสามารถทำการเปลี่ยนแปลงแก้ไขโปรแกรมได้ และภายใต้เงื่อนไขใดๆ)

2.4.18 มีวิธีการหรือเทคนิคที่ใช้ในการกำหนดคุณสมบัติหรือรายละเอียดของเครื่องมือทางด้านซอฟต์แวร์ (Software Tools) เพื่อใช้ในขั้นตอนการพัฒนาซอฟต์แวร์หรือไม่

2.4.19 มีวิธีการหรือเทคนิคที่ใช้ในการตรวจสอบว่าตัวอย่างต่างๆ ที่กลุ่มควบคุมคุณภาพซอฟต์แวร์ได้ทำการทดสอบเป็นตัวแทนที่แท้จริงของงานที่ดำเนินการอยู่ หรือไม่

2.4.20 มีวิธีการหรือเทคนิคที่ใช้เพื่อให้มั่นใจว่ามีการทดสอบหาข้อผิดพลาดของระบบ (Regression Testing) อยู่เป็นประจำ หรือไม่

2.4.21 มีวิธีการหรือเทคนิคที่ใช้เพื่อให้มั่นใจว่า การทดสอบหาข้อผิดพลาดของระบบได้ดำเนินการอย่างพอเพียง หรือไม่

2.4.22 มีการดำเนินการตรวจสอบทบทวน กรณีศึกษาต่างๆ ที่ใช้ในการทดสอบอย่างเป็นทางการ (Formal Test Case) หรือไม่

3. เครื่องมือ และเทคโนโลยีที่ใช้ (Tools and Technology)

คำถามในส่วนนั้นเกี่ยวข้องกับเครื่องมือ และเทคโนโลยีที่ใช้ในกระบวนการวิศวกรรมซอฟต์แวร์ จะมุ่งเน้นเพื่อให้เกิดความมั่นใจว่ากระบวนการต่างๆ ได้ถูกจัดเตรียมเครื่องมือ และวิธีการพื้นฐานที่จะใช้ไว้เรียบร้อยแล้ว ซึ่งคำถามในส่วนนี้จะไม่คิดรวมคะแนน ในการวัดผลระดับการเติบโตขององค์กร

I 17986564

3.1 มีเครื่องมือที่ใช้ในการควบคุม และติดตามกิจกรรมการเปลี่ยนแปลงต่างๆ ในทุกขั้นตอนการพัฒนาซอฟต์แวร์ หรือไม่

3.2 มีเครื่องมือทางด้านคอมพิวเตอร์ที่ใช้ช่วยในการติดตามการดำเนินงาน ในขั้นตอนการวิเคราะห์ความต้องการไปยังขั้นตอนการออกแบบซอฟต์แวร์หรือไม่

3.3 มีการใช้ภาษาที่ใช้ในการออกแบบ (Program Design Language - PDL) ใช้ในขั้นตอนการออกแบบหรือไม่

3.4 มีเครื่องมือทางด้านคอมพิวเตอร์ที่ใช้ช่วยในการติดตามการดำเนินงานในขั้นตอนการออกแบบซอฟต์แวร์ไปยังขั้นตอนการเขียนโปรแกรมหรือไม่

3.5 มีการใช้ภาษาระดับสูง (High - Order Language) ในการพัฒนาผลิตภัณฑ์ซอฟต์แวร์หรือไม่

3.6 มีเครื่องมือในการสร้างข้อมูลนำเข้าเพื่อการทดสอบโดยอัตโนมัติหรือไม่

3.7 มีเครื่องมือทางด้านคอมพิวเตอร์ที่ใช้ในการวัดผลการทดสอบหรือไม่

3.8 มีเครื่องมือทางด้านคอมพิวเตอร์ที่ใช้ในการติดตามตรวจสอบทุกหน้าที่ที่ต้องการ และมั่นใจว่าหน้าที่งานต่างๆ เหล่านั้นได้มีการทดสอบแล้วหรือไม่

3.9 มีเครื่องมือที่ใช้ในการวิเคราะห์ขนาด และกิจกรรมการเปลี่ยนแปลงต่างๆ ในองค์ประกอบของซอฟต์แวร์หรือไม่

3.10 มีเครื่องมือที่ใช้ในการวิเคราะห์ความซับซ้อนของซอฟต์แวร์ (Software Complexity) หรือไม่

3.11 มีเครื่องมือที่ใช้ในการวิเคราะห์การเชื่อมโยงระหว่างระบบย่อยต่างๆ (Cross Reference Between Modules) หรือไม่

3.12 มีการใช้เครื่องมือในการตรวจสอบหาความผิดพลาดของรหัสโปรแกรมโดยทันทีหรือไม่

3.13 มีอุปกรณ์อำนวยความสะดวกในด้านเอกสารต่างๆ แบบทันทีทันใด (Interactive Documentation Facilities) ให้กับผู้พัฒนาซอฟต์แวร์ และผู้บำรุงรักษาซอฟต์แวร์หรือไม่

3.14 มีเครื่องมือทางด้านคอมพิวเตอร์ที่ใช้ในการติดตามตรวจสอบและรายงานผลสถานะของซอฟต์แวร์เก็บอยู่ในไลบรารี่ของการพัฒนาซอฟต์แวร์ หรือไม่

3.15 มีการใช้แบบจำลองในการออกแบบองค์ประกอบ ที่มีความสำคัญของซอฟต์แวร์หรือไม่

3.16 มีการใช้แบบจำลองในการออกแบบองค์ประกอบ ที่มีความสำคัญในการติดต่อเชื่อมโยงระบบงานซอฟต์แวร์หรือไม่

<u>คำแนะนำในการประเมินผลแบบทดสอบระดับการเติบโต</u>

คำถามแต่ละข้อในแต่ละระดับ ผู้ทำการทดสอบจะต้องตอบคำถามเป็น " ผ่าน " หรือ " ไม่ผ่าน " เท่านั้น

ผ่าน *กรณีองค์กรมีการดำเนินงานเป็นไปตามคำถามที่ตั้งไว้*

ไม่ผ่าน *กรณีองค์กรไม่มีการดำเนินงานเป็นไปตามคำถามที่ตั้งไว้*

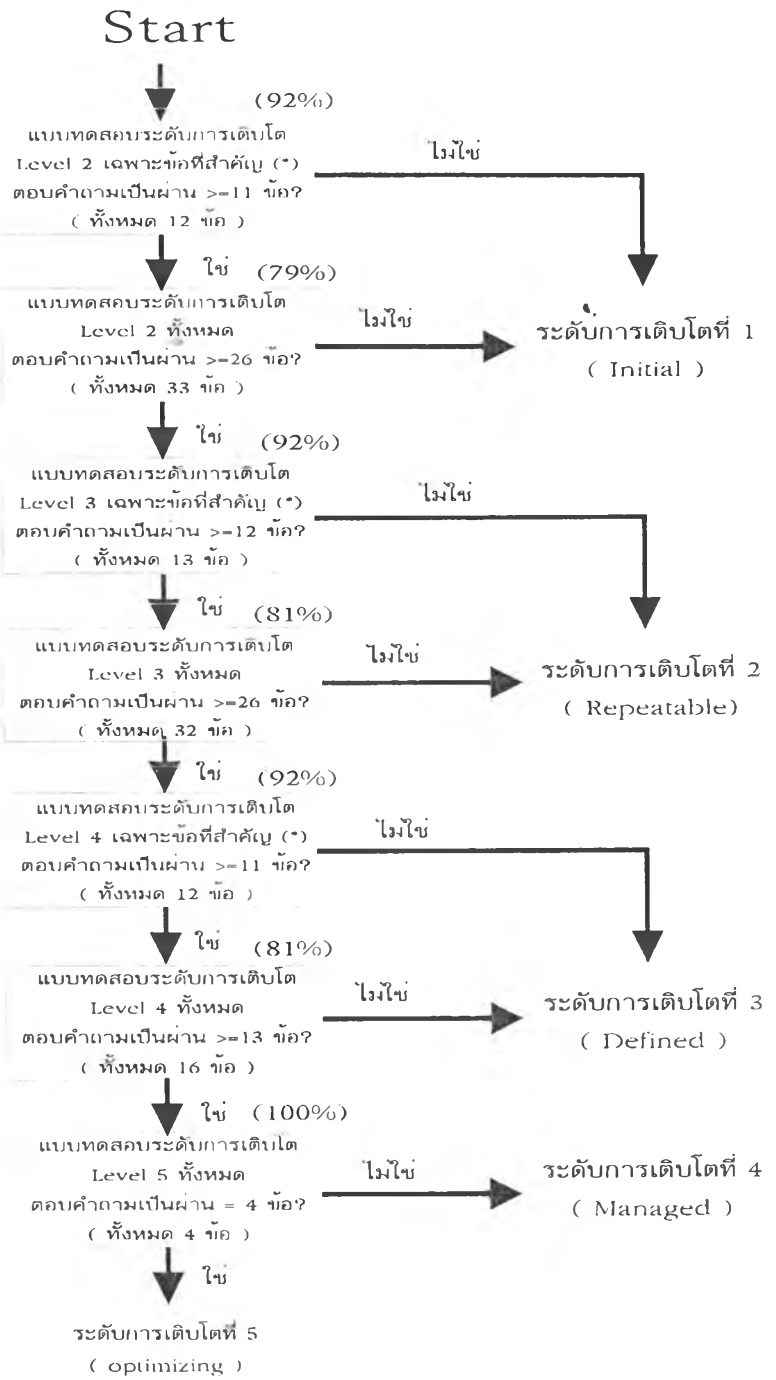
ในแต่ละระดับจะมีเกณฑ์การตัดสิน เพื่อให้ทราบว่าองค์กรอยู่ในระดับการเติบโตใด ดังนี้คือ

1. **การทดสอบในระดับที่ 2 (Repeatable)** แบ่งออกเป็น
   - ถ้าองค์กรตอบคำถามเฉพาะข้อที่สำคัญ (*) เป็นผ่านมากกว่า 90% (>11 ข้อ) ให้ทำแบบทดสอบของระดับนี้ทั้งหมด ถ้าไม่ใช่จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 1
   - ถ้าองค์กรตอบคำถามทั้งหมดของระดับนี้เป็นผ่าน มากกว่า 80% (> 26 ข้อ) ให้ทำแบบทดสอบระดับต่อไป ถ้าไม่ใช่จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 1

2. **การทดสอบในระดับที่ 3 (Defined)** แบ่งออกเป็น
   - ถ้าองค์กรตอบคำถามเฉพาะที่สำคัญ (*) เป็นผ่านมากกว่า 90% (> 12 ข้อ) ให้ทำแบบทดสอบของระดับนั้นทั้งหมด ถ้าไม่ใช่ จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 2
   - ถ้าองค์กรตอบคำถามทั้งหมดของระดับนี้เป็นผ่านมากกว่า 80% (> 26 ข้อ) ให้ทำแบบทดสอบระดับต่อไป ถ้าไม่ใช่จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 2

3. **การทดสอบในระดับที่ 4 (Managed)** แบ่งออกเป็น
   - ถ้าองค์กรตอบคำถามเฉพาะข้อที่สำคัญ (*) เป็นผ่านมากกว่า 90 % (> 11 ข้อ) ให้ทำแบบทดสอบของระดับนี้ทั้งหมด ถ้าไม่ใช่จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 3
   - ถ้าองค์กรตอบคำถามทั้งหมดของระดับนี้ เป็นผ่านมากกว่า 80% (> 13 ข้อ) ให้ทำแบบทดสอบระดับต่อไป ถ้าไม่ใช่จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 3

4. **การทดสอบในระดับที่ 5 (Optimizing)** ถ้าองค์กรตอบคำถามเป็นผ่านเท่ากับ 100% ของคำถามทั้งหมด (4 ข้อ) จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 5 ถ้าไม่ใช่จะถือว่าองค์กรนั้นอยู่ในระดับการเติบโตที่ 4



รูปภาพ 2.8 แสดงเกณฑ์การตัดสินระดับการเติบโตจากแบบทดสอบ

แบบทดสอบแต่ละระดับประกอบด้วย

1. คำถาม เป็นรายละเอียดต่าง ๆ ที่ผู้จะทำการทดสอบจะต้องศึกษารายละเอียดในแต่ละหัวข้อให้เข้าใจ และใช้ดุลยพินิจในการตรวจสอบว่าองค์กรของตนเองมีคุณสมบัติหรือมีรายละเอียดต่าง ๆ ในแต่ละขั้นตอนการทำงาน ครบถ้วนหรือไม่
2. ผ่าน เป็นช่องที่ให้ผู้ทำการทดสอบใส่เครื่องหมาย 4 ถ้าองค์กรมีคุณสมบัติครบถ้วน หรือมีการดำเนินงานเป็นไปตามคำถาม
3. ไม่ผ่าน เป็นช่องที่ให้ผู้ทำการทดสอบใส่เครื่องหมาย 4 ถ้าองค์กรมีคุณสมบัติไม่ครบถ้วนหรือไม่มีการดำเนินงานเป็นไปตามคำถาม

ระดับที่ 1 การเริ่มต้น (Initial)

องค์กรในระดับการเริ่มต้นนี้ จะมีลักษณะที่ไม่มีการกำหนดขั้นตอนการทำงาน ในการพัฒนาซอฟต์แวร์และการควบคุมการดำเนินงานอย่างชัดเจน นอกจากนี้องค์กรยังไม่สามารถประยุกต์การจัดการด้านวิศวกรรมซอฟต์แวร์ รวมทั้งเครื่องมือ และเทคโนโลยีที่ทันสมัย เข้ามาช่วยในการดำเนินงานอย่างต่อเนื่องอีกด้วย

ระดับที่ 2 การทบทวน (Repeatable)

องค์กรในระดับการเติบโตระดับที่ 2 นี้ องค์กรจะใช้วิธีการและการปฏิบัติงานมาตรฐานในการจัดการบริหารกิจกรรมการพัฒนาซอฟต์แวร์ ได้แก่ การประเมินต้นทุน ตารางเวลาการทำงาน การเปลี่ยนแปลงความต้องการ การเปลี่ยนแปลงรหัสโปรแกรม และการตรวจสอบทบทวนสถานะของกิจกรรม

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
1.1.1	ในแต่ละโครงการที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ มีการแต่งตั้งผู้จัดการซอฟต์แวร์ที่มีหน้าที่ความรับผิดชอบโครงการโดยเฉพาะ หรือไม่		
1.1.2	ผู้จัดการโครงการซอฟต์แวร์รายงานขึ้นตรงกับผู้จัดการโครงการใหญ่ (ผู้จัดการพัฒนาโครงการ) หรือไม่		
*1.1.3	มีช่องทางในการรายงานผลการจัดการบริหารหน้าที่ในการสร้างความเชื่อถือในคุณภาพซอฟต์แวร์ (Software Quality Assurance - SQA) แยกจากการรายงานผลการจัดการบริหารโครงการพัฒนาซอฟต์แวร์หรือไม่		
*1.1.6	มีหน้าที่ในการควบคุมรายละเอียดขอบเขตโครงร่างของ		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	ซอฟต์แวร์ (Software Configuration Control) สำหรับแต่ละโครงการรวมอยู่ในการพัฒนาซอฟต์แวร์ หรือไม่		
1.2.2	มีการจัดโปรแกรมการฝึกอบรมสำหรับผู้จัดการพัฒนาโครงการที่ได้รับการแต่งตั้งใหม่ เพื่อให้มีความคุ้นเคยกับการบริหารโครงการซอฟต์แวร์ หรือไม่		
1.3.1	มีวิธีการหรือเทคนิคที่ใช้ในการบอกให้ทราบถึงรายละเอียดของเทคโนโลยีทางด้านวิศวกรรมซอฟต์แวร์ หรือไม่		
*2.1.3	มีขั้นตอนการทำงานที่เป็นทางการ ใช้ในการตรวจสอบการบริหารของแต่ละขั้นตอนในการพัฒนาซอฟต์แวร์ก่อนที่จะทำสัญญาข้อตกลงดำเนินการพัฒนาซอฟต์แวร์หรือไม่		
2.1.4	มีขั้นตอนการทำงานที่เป็นทางการ ใช้เพื่อให้เกิดความมั่นใจในการตรวจสอบสถานะของแต่ละโครงการพัฒนาซอฟต์แวร์ โดยผู้จัดการตามช่วงเวลาที่กำหนดหรือไม่		
2.1.5	มีวิธีการหรือเทคนิคที่ใช้สำหรับตรวจสอบบริษัทที่รับจ้างผลิตซอฟต์แวร์ (Subcontractors) เพื่อให้เกิดความมั่นใจเชื่อถือหรือไม่ ถ้ามีตรงตามหลักเกณฑ์กระบวนการของการพัฒนาซอฟต์แวร์หรือไม่		
2.1.7	ในแต่ละโครงการ มีการดำเนินการตรวจสอบอิสระ (Independent Audits) ในแต่ละขั้นตอนของกระบวนการพัฒนาซอฟต์แวร์ หรือไม่		
2.1.9	มีการนำเอามาตรฐานของการเขียนรหัสโปรแกรม (Coding Standards) มาประยุกต์ใช้ในแต่ละโครงการพัฒนาซอฟต์แวร์ หรือไม่		
*2.1.14	มีขั้นตอนการทำงานที่เป็นทางการเพื่อใช้ในการประมาณขนาดของซอฟต์แวร์ (Estimates Of Software Size) หรือไม่		
*2.1.15	มีขั้นตอนการทำงานที่เป็นทางการเพื่อใช้ในการกำหนดตารางเวลาการทำงานของการพัฒนาซอฟต์แวร์ (Software Development Schedules) หรือไม่		
*2.1.16	มีขั้นตอนการทำงานที่เป็นทางการนำมาประยุกต์ใช้ในการประมาณต้นทุนที่ใช้ในการพัฒนาซอฟต์แวร์ (Estimating Software Development Cost) หรือไม่		
2.1.17	มีวิธีการหรือเทคนิคเพื่อให้แน่ใจว่าทีมออกแบบซอฟต์แวร์		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	เข้าใจถึงความต้องการซอฟต์แวร์ (Software Requirement) หรือไม่		
2.2.1	มีการเก็บข้อมูลพนักงานที่ใช้ไปในการพัฒนาซอฟต์แวร์จริง เปรียบเทียบกับข้อมูลจำนวนพนักงานตามแผนงาน หรือไม่		
*2.2.2	การประมาณขนาดของซอฟต์แวร์ในแต่ละส่วนประกอบของโครงร่างซอฟต์แวร์ (Software Configuration Item) ใช้เวลาตามกำหนดที่ตั้งไว้หรือไม่		
*2.2.4	มีการเก็บรวบรวมสถิติของข้อผิดพลาดที่เกิดขึ้นในการเขียนโปรแกรม และการทดสอบไว้หรือไม่		
2.2.7	เวลาที่ใช้ไปจริงในการออกแบบแต่ละส่วนของซอฟต์แวร์ (Software Units Designed) เป็นไปตามแผนงานที่กำหนดไว้หรือไม่		
2.2.8	การดำเนินการเพื่อให้แต่ละระบบย่อยของซอฟต์แวร์เสร็จสมบูรณ์ (Software Units Completing) การทดสอบระบบย่อย (Unit Testing) ใช้เวลาจริงเป็นไปตามแผนงานที่กำหนดไว้หรือไม่		
2.2.9	การรวบรวมระบบงานซอฟต์แวร์ในแต่ละส่วนย่อย (Software Units Integrated) ใช้เวลาจริงเป็นไปตามแผนงานที่กำหนดไว้หรือไม่		
2.2.10	มีการติดตามตรวจสอบ การใช้ประโยชน์จากหน่วยความจำคอมพิวเตอร์ (Computer Memory Utilization) ทั้งที่ได้ตั้งเป้าไว้จากการประเมินและที่ใช้จริงหรือไม่		
2.2.11	มีการติดตามตรวจสอบการใช้ประโยชน์การนำเข้าผ่านคอมพิวเตอร์ (Computer Throughput Utilization) ทั้งที่ได้ตั้งเป้าไว้จากการประเมิน และที่ใช้จริงหรือไม่		
2.2.12	มีการติดตามตรวจสอบการใช้งานช่องทางการเข้าออกข้อมูลคอมพิวเตอร์ (Computer I/O Channel Utilization) หรือไม่		
2.2.16	มีการติดตามตรวจสอบรายงานปัญหาของซอฟต์แวร์ที่เกิดขึ้นจากการทดสอบระบบอย่างใกล้ชิดหรือไม่		
2.2.18	มีการติดตามความก้าวหน้าของการทดสอบ (โดยเฉพาะส่วนของซอฟต์แวร์ที่ต้องส่งมอบ) เปรียบเทียบกับแผนงานที่ตั้งไว้หรือไม่		
2.2.19	มีการเก็บรวบรวมข้อมูลรายละเอียดการพัฒนาซอฟต์แวร์ และ		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	การเริ่มใช้ซอฟต์แวร์ เปรียบเทียบกับเวลาหรือไม่		
*2.4.1	มีวิธีการ หรือเทคนิคเพื่อให้ผู้จัดการอาวุโสใช้ในการตรวจสอบทบทวน สถานะของโครงการพัฒนาซอฟต์แวร์ อย่างสม่ำเสมอหรือไม่		
2.4.5	มีวิธีการหรือเทคนิคที่ใช้สำหรับการแลกเปลี่ยนทางด้านเทคนิคระหว่างผู้พัฒนาซอฟต์แวร์กับผู้ใช้อย่างสม่ำเสมอหรือไม่		
*2.4.7	มีการตรวจสอบอนุมัติตารางเวลาการทำงาน และการประมาณต้นทุนของโครงการโดยผู้จัดการพัฒนาซอฟต์แวร์ ระดับต้นหรือไม่		
*2.4.9	มีวิธีการหรือเทคนิคที่ใช้ในการควบคุมการเปลี่ยนแปลงความต้องการซอฟต์แวร์ หรือไม่		
*2.4.17	มีวิธีการหรือเทคนิคที่ใช้ในการควบคุมการปลี่ยนแปลงแก้ไขโปรแกรมหรือไม่ (ใครสามารถทำการเปลี่ยนแปลงแก้ไขโปรแกรมได้ และภายใต้เงื่อนไขใดๆ)		
2.4.20	มีวิธีการหรือเทคนิคที่ใช้เพื่อให้มั่นใจว่ามีการทดสอบหาข้อผิดพลาดของระบบ (Regression Testing) อยู่เป็นประจำ หรือไม่		

ระดับที่ 3 การกำหนด (Defined)

องค์กรในระดับการเติบโตระดับที่ 3 นี้ องค์กรไม่เพียงแต่มีการกำหนดกระบวนการทำงานซอฟต์แวร์ ในรูปแบบมาตรฐานของวิธีการทางวิศวกรรมซอฟต์แวร์เท่านั้น แต่ยังมีการจัดทำกลยุทธ์ในการปรับปรุงพัฒนาองค์กรด้วย ซึ่งกระบวนการปรับปรุงพัฒนานี้จะรวมทั้ง การตรวจสอบทบทวนการออกแบบ การตรวจสอบทบทวนการเขียนโปรแกรม การจัดโปรแกรมฝึกอบรมให้กับโปรแกรมเมอร์และทีมงานตรวจสอบ และการเพิ่มเป้าหมายขององค์กร เน้นไปทางด้านวิศวกรรมซอฟต์แวร์ การปรับปรุงพัฒนาที่สำคัญในขั้นตอนนี้ คือ มีการจัดตั้งกลุ่มกระบวนการวิศวกรรมซอฟต์แวร์ (Software engineering process group) ซึ่งประกอบด้วยผู้เชี่ยวชาญที่เกี่ยวข้องกับแต่ละขั้นตอนของกระบวนการพัฒนาซอฟต์แวร์ ซึ่งมุ่งเน้นไปยังกระบวนการทางด้านวิศวกรรมซอฟต์แวร์และความถูกต้องเที่ยงตรงในการนำกระบวนการนั้นไปใช้งาน

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
1.1.4	มีการแต่งตั้งบุคคลหรือทีมงานที่มีหน้าที่รับผิดชอบในการควบคุมการติดต่อเชื่อมโยงกับซอฟต์แวร์ (Software Interfaces) หรือไม่		
1.1.5	งานทางด้านวิศวกรรมระบบซอฟต์แวร์ (Software System Engineering) อยู่ในความรับผิดชอบของทีมงานการออกแบบระบบ (System Design Team) หรือไม่		
*1.1.7	มีการกำหนดกลุ่มวิศวกรรมซอฟต์แวร์ที่มีหน้าที่รับผิดชอบเกี่ยวกับกระบวนการที่ใช้ในการพัฒนาซอฟต์แวร์ (Process Group Software Engineering) หรือไม่		
1.2.1	ผู้พัฒนาซอฟต์แวร์แต่ละคนมีคอมพิวเตอร์ส่วนตัวใช้เป็นเครื่องลูกของเครือข่าย (Workstation / Terminal) หรือไม่		
*1.2.3	มีการจัดโปรแกรมการฝึกอบรมด้านวิศวกรรมซอฟต์แวร์ที่จำเป็น สำหรับผู้พัฒนาซอฟต์แวร์ (Software Developers) หรือไม่		
1.2.4	มีการจัดโปรแกรมการฝึกอบรมด้านวิศวกรรมซอฟต์แวร์ที่จำเป็นสำหรับหัวหน้าระดับต้นของทีมพัฒนาซอฟต์แวร์ (First Line Supervisors) หรือไม่		
*1.2.5	มีการจัดโปรแกรมการฝึกอบรมอย่างเป็นทางการสำหรับผู้ที่เป็นหัวหน้าในการตรวจสอบทบทวน การออกแบบ และการเขียนโปรแกรม (Design and Code Review Leaders) หรือไม่		
1.3.2	มีวิธีการหรือเทคนิคใช้ในการประเมินเทคโนโลยีที่ใช้ในองค์กรเปรียบเทียบ กับเทคโนโลยีที่มีอยู่ภายนอกหรือไม่		
*2.1.1	องค์กรที่ผลิตซอฟต์แวร์ ดำเนินการตามขั้นตอนกระบวนการพัฒนาซอฟต์แวร์ที่เป็นมาตรฐาน และเป็นเอกสารลายลักษณ์อักษรในแต่ละโครงการหรือไม่		
2.1.2	เอกสารรายละเอียดขั้นตอนกระบวนการพัฒนาซอฟต์แวร์ที่เป็นมาตรฐานอธิบายถึงรายละเอียดการใช้เครื่องมือ และเทคนิคต่างๆ เพื่อช่วยในการพัฒนาซอฟต์แวร์ หรือไม่		
2.1.6	มีมาตรฐานที่ใช้ในการเก็บรวบรวมรายละเอียดของแฟ้มข้อมูลที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ (Software Development Files / Folders) หรือไม่		
2.1.8	มีวิธีการหรือเทคนิคที่ใช้ในการประเมินการออกแบบ และ		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	ประเมินรหัสโปรแกรมที่เคยมีอยู่แล้วเพื่อนำกลับมาใช้ในโปรแกรมระบบงานคอมพิวเตอร์ที่ทำขึ้นใหม่หรือไม่		
2.1.10	มีมาตรฐานที่ใช้ในการจัดเตรียมตัวอย่างกรณีศึกษาต่าง ๆ ที่จะใช้ในการทดสอบระบบย่อยแต่ละส่วน (Unit Test Cases) หรือไม่		
2.1.11	มีการประยุกต์ใช้มาตรฐานในการบำรุงรักษารหัสโปรแกรม (Code Maintainability Standards) หรือไม่		
2.1.18	มีการประยุกต์ใช้มาตรฐานในการติดต่อเชื่อมโยงระหว่างผู้ใช้และคอมพิวเตอร์ (Man - Machine Interface Standards) เข้ากับแต่ละโครงการพัฒนาซอฟต์แวร์อย่างเหมาะสมหรือไม่		
*2.2.3	มีการเก็บรวบรวมสถิติของข้อผิดพลาดที่เกิดขึ้นในการออกแบบซอฟต์แวร์ไว้หรือไม่		
2.2.15	มีการติดตามรายละเอียดกิจกรรมต่าง ๆ ที่เป็นผลเนื่องมาจากการตรวจสอบทบทวนในขั้นตอนการออกแบบอย่างใกล้ชิดหรือไม่		
2.2.17	มีการติดตามรายละเอียดกิจกรรมต่าง ๆ ที่เป็นผลเนื่องมาจากการตรวจสอบทบทวนในขั้นตอนการเขียนโปรแกรมอย่างใกล้ชิดหรือไม่		
2.4.3	มีวิธีการหรือเทคนิคที่ใช้ในการระบุและตัดสินปัจจัยต่าง ๆ ทางด้านวิศวกรรมซอฟต์แวร์ว่ามีผลกระทบกับซอฟต์แวร์หรือไม่		
2.4.4	มีวิธีการหรือเทคนิคที่ใช้ในการนำเอาปัจจัยต่าง ๆ ที่ใช้ในการรวบรวมระบบงานซอฟต์แวร์ และการทดสอบที่เคยอยู่เป็นอิสระให้เข้ามาอยู่ในความสนใจดูแลของผู้จัดการโครงการ		
2.4.6	มีวิธีการหรือเทคนิคที่ใช้เพื่อให้มั่นใจว่าทุกคนเห็นด้วยกับมาตรฐานของวิศวกรรมซอฟต์แวร์หรือไม่		
2.4.8	มีวิธีการหรือเทคนิคที่ใช้ในการติดตามตรวจสอบการดำเนินการระหว่างขั้นตอนการวิเคราะห์ความต้องการซอฟต์แวร์ (Software Requirements) และการออกแบบขั้นต้น (Top Level Design) หรือไม่		
2.4.11	มีวิธีการหรือเทคนิคที่ใช้ในการติดตามตรวจสอบการดำเนินงานระหว่างขั้นตอนการออกแบบขั้นต้น (Top Level		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	Designs) และการออกแบบรายละเอียด (Detailed Designs) หรือไม่		
*2.4.12	มีการดำเนินการตรวจสอบทบทวนการออกแบบภายในซอฟต์แวร์ (Internal Software Design) หรือไม่		
*2.4.13	มีวิธีการหรือเทคนิคที่ใช้ในการควบคุมการเปลี่ยนแปลงการออกแบบซอฟต์แวร์หรือไม่		
2.4.14	มีวิธีการหรือเทคนิคที่ใช้ในการติดตามตรวจสอบการดำเนินงานระหว่างขั้นตอนการออกแบบรายละเอียดซอฟต์แวร์ (Software Detailed Design) และการเขียนโปรแกรม (Coding) หรือไม่		
2.4.15	มีการเก็บบันทึกรายละเอียดความก้าวหน้าของการพัฒนาระบบงานซอฟต์แวร์ในแต่ละส่วนย่อยๆ ไว้อย่างเป็นทางการหรือไม่		
*2.4.16	มีการดำเนินการตรวจสอบทบทวนการเขียนโปรแกรมซอฟต์แวร์หรือไม่		
2.4.18	มีวิธีการหรือเทคนิคที่ใช้ในการกำหนดคุณสมบัติหรือรายละเอียดของเครื่องมือทางด้านซอฟต์แวร์ (Software Tools) เพื่อใช้ในขั้นดอนการพัฒนาซอฟต์แวร์หรือไม่		
*2.4.19	มีวิธีการหรือเทคนิคที่ใช้ในการตรวจสอบว่าตัวอย่างต่างๆ ที่กลุ่มควบคุมคุณภาพซอฟต์แวร์ได้ทำการทดสอบุเป็นตัวแทนที่แท้จริงของงานที่ดำเนินการอยู่ หรือไม่		
*2.4.21	มีวิธีการหรือเทคนิคที่ใช้เพื่อให้มั่นใจว่า การทดสอบหาข้อผิดพลาดของระบบได้ดำเนินการอย่างพอเพียง หรือไม่		
2.4.22	มีการดำเนินการตรวจสอบทบทวน กรณีศึกษาต่างๆ ที่ใช้ในการทดสอบอย่างเป็นทางการ (Formal Test Case) หรือไม่		

ระดับที่ 4 การจัดการ (Managed)

องค์กรในระดับการ เติบโตระดับที่ 4 นี้ การตัดสินใจในการดำเนินงานขององค์กรจะมีพื้นฐานมาจากข้อมูลจำนวนมาก และการวิเคราะห์รายละเอียดของข้อมูลที่เก็บรวบรวมได้จากขั้นตอนการตรวจสอบทบทวนและการทดสอบจะมีการใช้เครื่องมือในการจัดการควบคุม ในขั้นดอนการออกแบบ

มากขึ้น เพื่อสนับสนุนการรวบรวมข้อมูลและการวิเคราะห์องค์กรเริ่มเรียนรู้ที่จะคาดการณ์ความผิดพลาดต่าง ๆ ที่อาจเกิดขึ้นต่อโครงการได้อย่างถูกต้องแม่นยำและสมเหตุสมผล

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
1.3.3	มีวิธีการ หรือเทคนิคที่ใช้เพื่อการตัดสินใจว่าเมื่อไรควรจะแทรกเทคโนโลยีใหม่ๆ เข้าไปยังกระบวนการพัฒนาซอฟต์แวร์		
*1.3.4	มีวิธีการ หรือเทคนิคที่ใช้ในการจัดการและสนับสนุนเทคโนโลยีใหม่ในเบื้องต้นหรือไม่		
2.1.12	มีการประยุกต์ใช้มาตรฐานในการตรวจสอบทบทวนการออกแบบภายในของซอฟต์แวร์ (internal design review) หรือไม่		
*2.1.13	มีการประยุกต์ใช้มาตรฐานในการตรวจสอบทบทวนการเขียนรหัสโปรแกรม (Code Review) หรือไม่		
*2.2.5	มีการคาดการณ์ข้อผิดพลาดในการออกแบบที่อาจเกิดขึ้นและเปรียบเทียบกับสิ่งที่เกิดขึ้นจริงหรือไม่		
*2.2.6	มีการคาดการณ์ข้อผิดพลาดในการเขียนโปรแกรม และการทดสอบที่อาจเกิดขึ้น และเปรียบเทียบกับสิ่งที่เกิดขึ้นจริงหรือไม่		
*2.2.13	มีการวัดระดับของการตรวจสอบทบทวนในขั้นตอนการออกแบบ และการเขียนโปรแกรม (Design And Code Review Coverage) และเก็บบันทึกรวบรวมไว้หรือไม่		
*2.2.14	มีการวัดผล และบันทึกจำนวนของรหัสโปรแกรมที่ถูกประมวลผลจริงในแต่ละขั้นของการทดสอบหน้าที่และคุณสมบัติของระบบงาน		
*2.3.1	มีการริเริ่มการจัดการและควบคุมฐานข้อมูลของกระบวนการเพื่อจัดเก็บข้อมูลการวัดผลกระบวนการที่ใช้ในทุกโครงการ		
*2.3.2	มีการวิเคราะห์ข้อมูลในการตรวจสอบทบทวน ที่เก็บรวบรวมได้ในระหว่างขั้นตอนการตรวจสอบทบทวนการออกแบบหรือไม่		
*2.3.3	มีการวิเคราะห์ข้อมูลความผิดพลาดต่างๆ จากการตรวจสอบทบทวนโปรแกรม และการทดสอบ เพื่อตัดสินว่ามีข้อผิดพลาดที่มีลักษณะคล้ายๆ กันนั้นหลงเหลืออยู่ใน		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	ผลิตภัณฑ์ซอฟต์แวร์หรือไม่		
*2.3.4	มีการดำเนินการวิเคราะห์ความผิดพลาดต่างๆ เพื่อที่จะวิเคราะห์หาสาเหตุต่างๆ ที่เกี่ยวข้องกับกระบวนการซอฟต์แวร์ หรือไม่		
*2.3.8	มีการวิเคราะห์ประสิทธิภาพของการตรวจสอบทบทวนในแต่ละโครงการหรือไม่		
2.3.9	มีการวิเคราะห์ประสิทธิภาพของซอฟต์แวร์ในแต่ละขั้นตอนของกระบวนการหรือไม่		
*2.4.2	มีวิธีการหรือเทคนิคที่ใช้ในการประเมินตามช่วงระยะเวลาว่ากระบวนการวิศวกรรมซอฟต์แวร์ และการนำไปใช้งานมีการปรับปรุงพัฒนาหรือไม่		
2.4.10	มีขั้นตอนการจัดการที่เป็นทางการ เพื่อใช้ในการตัดสินว่าแบบจำลองหน้าที่ต่างๆ ของซอฟต์แวร์ (Prototyping of Software Functions) เป็นส่วนที่เหมาะสมของขั้นตอนการออกแบบหรือไม่		

ระดับที่ 5 การปรับปรุง (Optimized)

องค์กรในระดับการเติบโตระดับที่ 5 นี้ องค์กรไม่เพียงแต่มีเป้าหมายในการควบคุมกระบวนการอย่างรัดกุม แต่ยังมีเป้าหมายหลักในการปรับปรุงพัฒนาการทำงานให้เกิดประโยชน์สูงสุด ซึ่งจะรวมทั้งการวิเคราะห์ความผิดพลาดอย่างละเอียด และการวิเคราะห์ข้อมูลต้นทุนที่เกิดขึ้นในระหว่างขั้นตอนของกระบวนการซอฟต์แวร์ และยังเริ่มต้นในการวิเคราะห์เพื่อให้เข้าใจถึงสาเหตุของความผิดพลาดที่เกิดขึ้น เพื่อศึกษาถึงแนวทางในการป้องกันไม่ให้เกิดขึ้น

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
*1.3.5	มีวิธีการหรือเทคนิคที่ใช้สำหรับบ่งบอก และเปลี่ยนแทนเทคโนโลยีที่ล้าสมัยหรือไม่		
*2.3.5	มีวิธีการหรือเทคนิคที่ใช้สำหรับการวิเคราะห์หาสาเหตุของความผิดพลาดที่เกิดขึ้น (Error - Cause Analysis) หรือไม่		
*2.3.6	มีการตรวจสอบทบทวนสาเหตุของความผิดพลาด ที่เกิดขึ้นเพื่อช่วยในการตัดสินว่าควรจะมีการเปลี่ยนแปลงกระบวนการหรือขั้นตอน เพื่อที่จะป้องกันไม่ให้เกิดความผิดพลาดขึ้นอีกหรือไม่		
*2.3.7	มีวิธีการหรือเทคนิคที่ใช้สำหรับดำเนินการเพื่อป้องกันความผิด		

ข้อ	คำถาม	ผ่าน	ไม่ผ่าน
	พลาดที่อาจเกิดขึ้นตั้งแต่เริ่มต้นหรือไม่		

อนาคตของทิศทางซีเอ็มเอ็ม (Future Directions of the CMM)

การจะก้าวไปสู่ระดับที่สูงขึ้นของการเติบโตของกระบวนการซอฟต์แวร์ (Software Process Maturity) จะเพิ่มขึ้นทีละน้อย และอาศัยการดำเนินการพัฒนากระบวนการอย่างต่อเนื่องในระยะยาว องค์กรซอฟต์แวร์อาจใช้เวลา 10 ปี หรือมากกว่านั้น เพื่อจะก่อให้เกิดการปรับปรุงพัฒนากระบวนการที่ต่อเนื่อง แม้นว่าปัจจุบันอาจยังไม่เป็นที่คุ้นเคยมากนัก แต่ในปัจจุบันก็เริ่มเป็นที่รู้จักกันในบริษัทของสหรัฐอเมริกา และคาดว่าคงจะแพร่หลายกันในอนาคต

สิ่งที่ซีเอ็มเอ็มไม่ได้ครอบคลุม (What the CMM Does Not Cover)

ซีเอ็มเอ็มไม่ได้อธิบายปัจจัยทุกอย่างที่สำคัญต่อความสำเร็จของโครงการ ตัวอย่างเช่น ซีเอ็มเอ็มไม่ได้อธิบายองค์ประกอบที่สำคัญของระบบงานคอมพิวเตอร์ เทคโนโลยีซอฟต์แวร์ หรือแนะนำว่าจะเลือกจ้างพนักงานที่มีคุณสมบัติอย่างไร แม้ว่าองค์ประกอบบางอย่างสำคัญต่อความสำเร็จขององค์กรแต่ก็ไม่ได้รวมอยู่ในซีเอ็มเอ็ม

ซีเอ็มเอ็มอธิบายเกี่ยวกับการพัฒนาอย่างเป็นขั้นตอน โครงร่างของงานที่ควรจะเป็น ซึ่งเกี่ยวข้องกับการบริหารงานซอฟต์แวร์ (Software Management) และกระบวนการ ทางวิศวกรรม (Engineering Process) เท่านั้น

ซีเอ็มเอ็มเสนอโครงสร้างของแนวคิดเพื่อการปรับปรุงการจัดการและการพัฒนาผลิตภัณฑ์ซอฟต์แวร์ แต่ไม่ได้รับประกันว่า ผลิตภัณฑ์ซอฟต์แวร์นั้นจะถูกสร้างขึ้นได้สำเร็จ หรือปัญหาทุกอย่างในด้านวิศวกรรมซอฟต์แวร์จะต้องถูกแก้ไขได้ ซีเอ็มเอ็มเพียงเสนอแนวปฏิบัติสำหรับการพัฒนาการเติบโตของซอฟต์แวร์อย่างมีประสิทธิภาพเท่านั้น