CMM กับ การพัฒนาซอฟต์แวร์

ดร. ครรชิต มาลัยวงศ์

ถ้าจะถามว่าประเทศไทยเริ่มสนใจในการพัฒนาซอฟต์แวร์เมื่อใด ก็ต้องตอบว่าเมื่อนานไม่ต่ำกว่าสองทศวรรษแล้ว เมื่อแรกเริ่มที่ผมใช้เครื่อง IBM 1130 อันเป็นคอมพิวเตอร์ขนาดเล็กมากที่สถาบันเทคโนโลยีแห่งเอเชียเช่ามาจากไอบีเอ็มนั้น ซอฟต์แวร์ทุกอย่างทางไอบีเอ็มให้มาโดยไม่คิดเงินเลย นับตั้งแต่ ระบบควบคุม (Monitor system) คอมไพเลอร์ ซับรูทีนสำหรับงานวิทยาศาสตร์ และสถิติ ตลอดจนโปรแกรมประยุกต์ต่าง ๆ สุดแท้แต่จะต้องการ แต่พอใช้ไปได้ไม่กี่ปี ไอบีเอ็มก็แจ้งว่าต่อไปนี้ให้เปล่าไม่ได้แล้ว จะต้องคิดเงินค่าเช่าซอฟต์แวร์ด้วย ผมเองก็แปลกใจเพราะคิดเหมือน ๆ กับ ที่หลายคนคิดในขณะนี้ว่าซอฟต์แวร์น่าจะเป็นของได้เปล่า แต่นั่นเป็นช่วงที่ไอบีเอ็มกำลังมีปัญหากับทางรัฐบาลในเรื่องการผูกขาดธุรกิจ หากไอบีเอ็มไม่คิดเงินก็จะกลายเป็นว่าไอบีเอ็มตัดหนทางการค้าของบริษัทซอฟต์แวร์อื่น ๆ ซึ่งเริ่มจะเกิดขึ้นแล้ว ผมนั่งคิดต่อไปว่านักศึกษาของเอไอทีที่พัฒนาโปรแกรมเยี่ยม ๆ ทางด้านวิศวกรรมนั้นมีอยู่มาก แต่เมื่อทำโปรแกรมจนสำเร็จได้ผล และ นักศึกษาเองก็จบไปแล้ว โปรแกรมเหล่านี้ก็จะกระจัดกระจายหายไป ไม่เกิดประโยชน์อันใด ผมจึงนำเรื่องเสนอต่อสถาบันให้บังคับนักศึกษาทุกคนจะต้องส่งโปรแกรมที่พัฒนาแล้วมาเก็บไว้ที่ศูนย์คอมพิวเตอร์ โดยผมกะว่าในอนาคตศูนย์คอมพิวเตอร์ของผมจะเป็นศูนย์กลางซอฟต์แวร์ด้านวิศวกรรมที่ใหญ่ที่สุดในภูมิภาคเลยทีเดียว

ความคิดของผมไม่เป็นผล แม้ทางสถาบันจะยอมทำตามผมโดยไม่ทราบความสำคัญของโปรแกรมเหล่านี้มากนัก แต่ผมก็ไม่สามารถนำโปรแกรมเหล่านั้นมาตรวจสอบ แก้ไข จัดหมวดหมู่ และทำดัชนีเก็บไว้ใช้งานได้ เพราะไม่มีกำลังคนที่รู้ทั้งด้านคอมพิวเตอร์และงานวิศวกรรม จะใช้นักศึกษาก็ไม่ได้เพราะเขาก็จะต้องรีบเร่งทำวิทยานิพนธ์ของตนเองให้จบ ดังนั้นสุดท้ายผมเชื่อว่าป่านนี้โปรแกรมที่ผมได้รวบรวมไว้หลายพันหลายหมื่นโปรแกรมก็คงจะกระจัดกระจายสูญหายไปหมดแล้ว สาเหตุอีกข้อหนึ่งก็คือผมยังไม่ได้เรียนรู้หลักการทำโปรแกรมและซอฟต์แวร์ให้เป็นอุตสาหกรรม ยังไม่ทราบความสำคัญของการควบคุมคุณภาพซอฟต์แวร์ และที่สำคัญยังไม่ได้ศึกษาเรื่องของมาตรฐานต่าง ๆ ทางด้านซอฟต์แวร์

ในช่วงนั้นผมคิดแต่เพียงว่าแต่ละปีนักศึกษาแต่ละคนต้องเขียนโปรแกรมสำหรับใช้ในงานวิทยานิพนธ์ระดับก้าวหน้าออกมาคนละหลาย ๆ โปรแกรม ผมเชื่อจริง ๆ ว่า หากได้ทำจริงจังโปรแกรมเหล่านี้ก็จะเป็นรายได้ให้กับทางสถาบันได้ แต่ไม่เคยคิดหรอกครับว่าจะมีคนสร้างความร่ำรวยมหาศาลจากโปรแกรมได้เหมือนบริษัทซอฟต์แวร์เวลานี้

ต่อมา ทางกระทรวงวิทยาศาสตร์ฯ ก็เริ่มคิดฝันเหมือนผมเหมือนกันว่า ซอฟต์แวร์น่าจะเป็นอุตสาหกรรมสำหรับทำรายได้ให้แก่ประเทศได้ ไม่ได้คิดเฉย ๆ กระทรวงฯ ยังถึงกับตั้ง

กรรมการอุตสาหกรรมซอฟต์แวร์ขึ้นมาด้วย แต่คณะกรรมการก็นัดประชุมได้ไม่กี่ครั้งก็มีอันต้องเลิกลา สลายกันไปหมด

จนกระทั่งขณะนี้ ทางเนคเทคสามารถผลักดันโครงการซอฟต์แวร์พาร์คจนได้รับงบประมาณ และดำเนินการ ก้าวหน้าไปจนถึงขั้นซื้ออาคารมาเป็นสำนักงานโครงการได้แล้ว แต่งานด้านอุตสาหกรรมซอฟต์แวร์ของเราก็ยังคงล้มลุกคลุกคลาน ไม่ได้ตั้งไข่สักที

เรื่องที่น่าเจ็บปวดก็คือ โครงการนี้ได้รับความเห็นชอบเมื่อประเทศชาติกำลังหมดตัว ดังนั้นลำพังแต่จะทำให้เป็นโครงการที่จะสร้างความเชื่อมั่นให้แก่บริษัทซอฟต์แวร์ประเทศอื่นก็ยังไม่สามารถทำได้ ประเทศอื่นหลายประเทศได้ผลักดันโครงการอุตสาหกรรมซอฟต์แวร์นำหน้าเราไปไกลแล้ว ยกตัวอย่างเช่นอินเดียเริ่มก่อนหน้าเรานานถึงยี่สิบปี ไต้หวันที่นำหน้าเราอยู่แล้วทางด้านฮาร์ดแวร์ ก็เริ่มมีซอฟต์แวร์พาร์กสำหรับหนีเราไปอีกไกลสุดกู่ มาเลเซียที่ว่าย่อบแย่บก็ยังผลักดันโครงการ Multimedia Super Corridor ต่อไปในอาณาบริเวณที่มีเนื้อที่กว้างกว่าประเทศสิงคโปร์ แต่ของไทยเรานั้นลำพังจะหาอาคารสำหรับเป็นที่ทำการให้ได้สักหลังยังไม่ได้ เวลาไปพูดกับใครเขาบอกว่าได้มาแค่สี่ชั้นก็บุญแล้ว

การพัฒนาซอฟต์แวร์ให้เป็นอุตสาหกรรมสำคัญของไทยนั้นมีเรื่องที่จะต้องดำเนินการอีกมาก อย่าได้นึกเป็นอันขาดเชียวว่า เรียน Visual BASIC จากโครงการของกระทรวงแรงงานฯ แล้วจะทำให้สร้างอุตสาหกรรมซอฟต์แวร์ได้ หรือจะทำให้ร่ำรวยเหมือนบิลล์ เกตส์

เวลานี้การพัฒนาซอฟต์แวร์ของไทยนั้นยังล้าหลังประเทศอื่น ๆ อีกมาก ความรู้ในด้านการพัฒนาซอฟต์แวร์ของเรายังด้อยกว่าของประเทศอื่นมาก อาจจะเป็นเพราะเราไม่ได้พัฒนาซอฟต์แวร์อย่างเป็นล่ำเป็นสันมาก่อน อาจจะเป็นเพราะการเรียนการสอนด้านซอฟต์แวร์ของเรายังไม่ก้าวหน้า หรือ อาจจะเป็นเพราะไม่มีนักซอฟต์แวร์ที่มีประสบการณ์ต่อเนื่องเป็นเวลานาน ๆ ก็ได้ทั้งสิ้น

การที่จะส่งเสริมการพัฒนาซอฟต์แวร์ในเมืองไทยให้รุ่งเรืองจนเป็นอุตสาหกรรมได้ จึงจำเป็นที่จะต้องพัฒนาความรู้ความสามารถในด้านวิศวกรรมซอฟต์แวร์ให้ก้าวหน้าทัดเทียมกับประเทศอื่นได้ก่อน

ปัจจุบันนี้ความก้าวหน้าทางด้านซอฟต์แวร์ของต่างประเทศได้ล้ำหน้าประเทศไทยไปมาก การจะพัฒนาความรู้ขึ้นได้จึงจำเป็นจะต้องค่อย ๆ เก็บเนื้อหาที่น่าสนใจมาเล่าสู่กันฟังไปตามลำดับ หากใครสนใจต้องการศึกษาค้นคว้าต่อก็จะได้มีแหล่งอ้างอิงที่เป็นหลักเป็นฐาน

เรื่องแรกที่จะนำมาเสนอในที่นี้ก็คือเรื่อง Capability Maturity Model ทางด้านซอฟต์แวร์ แต่ก่อนจะไปถึงเรื่องนี้คงจะต้องเริ่มที่เรื่องของ SEI

สถาบันวิศวกรรมซอฟต์แวร์

การเป็นวิศวกรซอฟต์แวร์ ยังไม่ได้รับการรับรองให้เป็นวิชาชีพก็จริงอยู่ แต่งานทางด้านนี้มีความสำคัญอย่างยิ่ง ทุกวันนี้ซอฟต์แวร์มีขนาดใหญ่ขึ้นและซับซ้อนมากขึ้นจนสุดวิสัยที่จะมีใครพัฒนาระบบเหล่านี้ได้คนเดียว การพัฒนาซอฟต์แวร์ร่วมกันหลายคนจึงจำเป็นจะต้องมีกฎเกณฑ์และแนวทางในการทำงานอย่างรัดกุม ก็การพัฒนาซอฟต์แวร์หรือโปรแกรมในอดีตนั้นถือกันว่าเป็นศิลปะแบบหนึ่ง ดังที่ โดนัลด์ คนุธ ศาสตราจารย์ของมหาวิทยาลัยสแตนฟอร์ดยังเขียนตำราการเขียนโปรแกรมที่เลื่องชื่อออกมาชุดหนึ่งในชื่อว่า The Art of Programming เมื่อเป็นศิลปะจึงยากที่จะกำหนดกฎเกณฑ์ว่าใครจะทำอะไรก่อนอะไรหลัง จะทำอย่างไร หรือจะควบคุมตรวจสอบกันอย่างไร

ในเมื่อซอฟต์แวร์หรือโปรแกรมเป็นงานสร้างสรรค์ทางปัญญาที่ซับซ้อน การที่จะสื่อความกันระหว่างการพัฒนาโปรแกรมจึงเป็นเรื่องยาก งานซอฟต์แวร์ขนาดใหญ่จึงมักจะลงเอยด้วยความล้มเหลวยิ่งกว่าความสำเร็จ

ด้วยเหตุนี้เอง เมื่อยี่สิบปีมานี้บรรดาเกจิอาจารย์ทางด้านซอฟต์แวร์ทั้งหลายจึงเริ่มปรึกษาหาแนวทางที่จะทำให้การพัฒนาซอฟต์แวร์ประสบความสำเร็จมากขึ้น และหลังจากนั้นไม่นานก็เริ่มมีผู้คิดกระบวนการตลอดจนตัวแบบต่าง ๆ สำหรับการพัฒนาซอฟต์แวร์ออกมาเผยแพร่มากขึ้น ในกระบวนความคิดเหล่านี้ ความคิดที่ว่าการพัฒนาซอฟต์แวร์ควรจะอิงอาศัยแบบอย่างงานด้านวิศวกรรมที่มีระเบียบวิธีและขั้นตอนที่กำหนดไว้ชัดเจนเป็นอย่างดี ดูจะได้รับความสนใจมากที่สุด และทำให้เกิดแนวคิดเรื่อง **วิศวกรรมซอฟต์แวร์** ขึ้น

วิศวกรรมซอฟต์แวร์มุ่งที่จะใช้หลักการด้านวิศวกรรมในการตอบคำถาม หรือ หาทางดำเนินการในด้านต่อไปนี้

- เรากำลังแก้ปัญหาอะไร
- ปัญหานั้นเกี่ยวกับอะไรบ้าง
- ลักษณะของสิ่งที่จะนำมาใช้แก้ปัญหาได้เป็นอย่างไร
- แนวทางแก้ปัญหาเป็นเช่นใด
- เราจะสร้างคำตอบ หรือ สิ่งที่จะใช้แก้ปัญหาได้อย่างไร
- เราจะรู้ได้อย่างไรว่าคำตอบนั้นแก้ปัญหาได้
- เราจะขจัดข้อบกพร่องหรือผิดพลาดในสิ่งที่ใช้แก้ปัญหาได้อย่างไร
- เราจะควบคุมกระบวนการสร้างคำตอบหรือสิ่งที่จะใช้แก้ปัญหาได้อย่างไร
- เราจะรู้ได้อย่างไรว่าสิ่งที่เราใช้แก้ปัญหานั้น แก้ปัญหาได้จริง

ในเมื่อการพัฒนาซอฟต์แวร์นั้นอาจกล่าวได้ว่าเกี่ยวข้องกับการแก้ปัญหาอย่างใดอย่างหนึ่ง กระบวนการพัฒนาซอฟต์แวร์จึงนำขั้นตอนการแก้ปัญหามาใช้เป็นแบบอย่างในการพัฒนาซอฟต์แวร์ด้วย นั่นคือ [1]

- การกำหนดปัญหา เป็นการกำหนดว่าปัญหาที่ต้องการจัดทำซอฟต์แวร์ขึ้นคืออะไร มีลักษณะอย่างไร อะไรคือคำตอบหรือผลที่ต้องการ ซอฟต์แวร์นั้นจะต้องดำเนินการกับข้อมูลอะไรบ้าง ซอฟต์แวร์จะต้องทำหน้าที่อะไร จะต้องมีความสามารถมากขนาดใด จะต้องมีพฤติกรรมอย่างไร จะต้องจัดทำอินเทอร์เฟสแบบใด มีเงื่อนไขข้อกำหนดในการออกแบบซอฟต์แวร์อย่างไรบ้าง มีวิธีการตรวจสอบความถูกต้องอย่างไรบ้าง
- การพัฒนาซอฟต์แวร์ เป็นการเน้นว่าจะสร้างซอฟต์แวร์ได้อย่างไรและจะให้ซอฟต์แวร์ทำงานอย่างไร ในขั้นตอนนี้ผู้พัฒนาซอฟต์แวร์จะต้องพยายามกำหนดว่าจะสร้างข้อมูลอย่างไร จะกำหนดโครงสร้างของซอฟต์แวร์อย่างไร จะถ่ายทอดโครงสร้างลงเป็นตัวโปรแกรมได้อย่างไร และจะทดสอบโปรแกรมอย่างไร
- การบำรุงรักษาซอฟต์แวร์ เป็นการเน้นว่าจะเปลี่ยนแปลงซอฟต์แวร์ได้อย่างไร เช่น จะขยายซอฟต์แวร์ให้มีความสามารถมากขึ้น จะแก้ความบกพร่องผิดพลาดในโปรแกรม หรือ จะปรับเปลี่ยนอย่างไรในเมื่อสิ่งแวดล้อมของโปรแกรมเปลี่ยนไป

สถาบัน IEEE ได้ให้คำนิยามของวิศวกรรมซอฟต์แวร์ในปี 1993 ว่า

Software engineering: (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. (2) the study of approaches as in (1).

เมื่อปี 1984 กระทรวงกลาโหมสหรัฐฯ เล็งเห็นการพัฒนาซอฟต์แวร์มีความสำคัญอย่างยิ่งต่อการปฏิบัติการด้านต่าง ๆ ของกระทรวง หากการพัฒนาซอฟต์แวร์ยังมีปัญหาแล้วอาจส่งผลให้เกิดข้อบกพร่องร้ายแรงในซอฟต์แวร์ต่าง ๆ อันอาจจะเป็นผลร้ายต่อการดำเนินงานต่าง ๆได้ ดังนั้นกระทรวงกลาโหมฯ จึงตั้งโครงการจัดตั้งสถาบันวิศวกรรมซอฟต์แวร์ขึ้นและขอให้มหาวิทยาลัยต่าง ๆ ในสหรัฐฯ ที่ต้องการได้รับเงินทุนไปจัดตั้งสถาบันนี้ส่งข้อเสนอมาให้พิจารณา ผลปรากฏว่ามหาวิทยาลัยคาร์เนกี เมลลอน ได้รับเลือกให้เป็นผู้ดำเนินการสถาบันนี้

สถาบันวิศวกรรมซอฟต์แวร์ (Software Engineeging Institute) จึงเกิดขึ้นโดยมีเจ้าหน้าที่และผู้เชี่ยวชาญจากมหาวิทยาลัย จากภาครัฐ และ ภาคอุตสาหกรรม เข้ามาทำงานร่วมกัน โดยมีภารกิจสำคัญที่จะเป็นผู้นำในการผลักดันระดับความสามารถทางด้านวิศวกรรมซอฟต์แวร์ให้ก้าวหน้ามากขึ้น และ เพื่อปรับปรุงระบบต่าง ๆ ที่ต้องอาศัยซอฟต์แวร์เป็นพื้นฐานการทำงาน [2]

งานหลักของ SEI ก็คือการเปลี่ยนการพัฒนาซอฟต์แวร์จากรูปแบบการทำงานที่ใช้คนจำนวนมาก และ ทำงานอย่างไม่มีหลักเกณฑ์ให้กลายเป็นงานที่มีระบบและกฎเกณฑ์ อีกทั้งยังสามารถควบคุมและจัดการให้เป็นไปตามที่เหมาะที่ควรได้

ภารกิจของ SEI มีอยู่สองด้านคือ

- เพิ่มขีดความสามารถในการจัดการ งานส่วนนี้เน้นในด้านการทำให้องค์การสามารถพยากรณ์ และ ควบคุม คุณภาพ กำหนดเวลา ต้นทุน และ ผลผลิต ในการจัดหา สร้าง หรือปรับปรุงระบบซอฟต์แวร์
- พัฒนาแนวทางด้านเทคนิควิศวกรรม งานส่วนนี้เน้นในด้านความสามารถของวิศวกรในการวิเคราะห์ พยากรณ์ และ ควบคุม สมบัติบางประการของระบบซอฟต์แวร์ งานส่วนนี้เกี่ยวกับการกำหนดทางเลือกสำคัญ ๆ ระหว่างการจัดหา สร้าง หรือ ปรับปรุงระบบซอฟต์แวร์

กระบวนการซอฟต์แวร์

คำว่า กระบวนการซอฟต์แวร์ (software process) เป็นคำที่ใช้กันมากสำหรับระบุงานต่าง ๆ ที่เกี่ยวข้องกับกระบวนการพัฒนาซอฟต์แวร์ Pressman [1] กล่าวว่า กระบวนการซอฟต์แวร์นั้นหมายถึงกรอบของงานต่าง ๆ ที่จำเป็นสำหรับการพัฒนาซอฟต์แวร์ที่มีคุณภาพสูง คำว่ากระบวนการซอฟต์แวร์นี้มีความหมายทั้งเหมือนกันและแตกต่างกันจากคำว่าวิศวกรรมซอฟต์แวร์ คำว่ากระบวนการซอฟต์แวร์นั้นหมายถึงแนวทางที่ใช้ระหว่างการดำเนินการวิศวกรรมกับซอฟต์แวร์ แต่คำว่าวิศวกรรมซอฟต์แวร์นั้นรวมเทคโนโลยีต่าง ๆ ที่นำมาใช้ในกระบวนการซอฟต์แวร์ ตลอดจนเครื่องมือต่าง ๆ ทางด้านวิศวกรรมซอฟต์แวร์ด้วย

กระบวนการซอฟต์แวร์ของบริษัทและหน่วยงานต่าง ๆ นั้นมีความแตกต่างกันมาก สุดแท้แต่ความสามารถในการบริหารงานซอฟต์แวร์ และ ของบุคลากรผู้พัฒนาซอฟต์แวร์ บริษัทและหน่วยงานบางแห่งสามารถพัฒนาซอฟต์แวร์ได้ดี แต่บริษัทและหน่วยงานอื่นที่มีสมบัติคล้ายกันในแทบจะทุกด้านกลับไม่สามารถพัฒนาซอฟต์แวร์ได้ดีเท่า ด้วยเหตุนี้จึงเกิดคำถามว่าเหตุใดจึงเป็นเช่นนี้ มีปัจจัยอะไรหรือที่บงการให้เกิดความแตกต่างเช่นนี้

เพื่อตอบคำถามนี้ เมื่อเดือนพฤศจิกายน ปี 1986 สถาบัน SEI ร่วมกับ Mitre Corporation ได้เริ่มดำเนินการพัฒนาแบบจำลองสำหรับใช้ในการระบุว่าหน่วยงานต่าง ๆ มีความสามารถในการพัฒนาซอฟต์แวร์ถึงระดับไหนบ้าง [3] ในเดือนกันยายนปีต่อมา สถาบันได้ประกาศคำอธิบายสั้น ๆ เกี่ยวกับกรอบงานสำหรับวัดความเจริญก้าวหน้าในด้านการพัฒนาซอฟต์แวร์ออกมาให้ผู้สนใจรับทราบ การวัดนี้ใช้วิธีการสามแบบ แบบแรกเรียกว่าการประเมินกระบวนการซอฟต์แวร์ซึ่งเป็นการใช้ทีมงานซอฟต์แวร์มืออาชีพที่ได้รับการฝึกมาเป็นพิเศษในการกำหนดระดับความก้าวหน้าด้านซอฟต์แวร์ขององค์การ มาศึกษาประเด็นสำคัญต่าง ๆ เกี่ยวกับการ

พัฒนาซอฟต์แวร์ที่องค์การกำลังประสบอยู่ แบบที่สองคือ การประเมินความสามารถของซอฟต์แวร์ เป็นการใช้ทีมงานซอฟต์แวร์มืออาชีพในการประเมินหาผู้รับจ้างพัฒนาซอฟต์แวร์ที่มีความสามารถมากพอที่จะรับงานดูแลกระบวนการซอฟต์แวร์ที่องค์การกำลังดำเนินงานอยู่ และแบบที่สามเป็นการใช้แบบสอบถามที่มีคำตอบสำหรับประเมินค่าห้าระดับในการประเมินความเจริญก้าวหน้าของกระบวนการซอฟต์แวร์

หลังจากใช้กรอบงานข้างต้นในการวัดความเจริญก้าวหน้าของกระบวนการซอฟต์แวร์เป็นเวลานานถึงสี่ปี ในที่สุดสถาบันก็สามารถจัดทำแบบจำลองความเจริญก้าวหน้าทางด้านความสามารถทางด้านซอฟต์แวร์ขององค์การออกมาได้ในปี 1991 แบบจำลองนี้เรียกย่อ ๆ ว่า CMM (ย่อมาจาก Capability Maturity Model for Software)

CMM ช่วยให้หน่วยงานที่ทำงานด้านซอฟต์แวร์มีแนวทางสำหรับควบคุมกระบวนการในการพัฒนาและดูแลรักษาซอฟต์แวร์ และ แนวทางสำหรับทำให้หน่วยงานก้าวหน้าขึ้นไปถึงระดับชั้นที่มีความเป็นเลิศทางด้านวิศวกรรมซอฟต์แวร์ และ การจัดการซอฟต์แวร์ แบบจำลองนี้ช่วยให้หน่วยงานซอฟต์แวร์สามารถเลือกกลยุทธ์ในการปรับปรุงกระบวนการซอฟต์แวร์โดยการกำหนดว่าปัจจุบันหน่วยงานมีระดับความเจริญก้าวหน้าอยู่ขั้นใด แล้วจึงกำหนดประเด็นต่าง ๆ ที่มีความสำคัญต่อการปรับปรุงคุณภาพและกระบวนการซอฟต์แวร์ โดยวิธีให้ความสนใจเฉพาะในประเด็นที่สำคัญบางเรื่องจะช่วยให้หน่วยงานปรับปรุงกระบวนการซอฟต์แวร์ได้อย่างต่อเนื่องโดยตลอด และ สามารถยกระดับความเจริญก้าวหน้าได้อย่างยั่งยืน

หลักการพื้นฐานเกี่ยวกับความเจริญก้าวหน้าของกระบวนการ

ได้กล่าวไปแล้วว่า **กระบวนการซอฟต์แวร์** คือแนวทาง อันรวมทั้งกิจกรรม วิธีการ และวิธีปฏิบัติในการพัฒนาและบำรุงรักษาซอฟต์แวร์ ในข้อเสนอของ CMM ได้ให้รวมงานที่เกี่ยวข้องกับซอฟต์แวร์เอาไว้ในกระบวนการซอฟต์แวร์ด้วย [3] นั่นคือ แผนงานโครงการ เอกสารการออกแบบระบบ ตัวคำสั่งในโปรแกรม รายการทดสอบ และ คู่มือผู้ใช้ เมื่อหน่วยงานหรือองค์การเจริญก้าวหน้าขึ้นกระบวนการซอฟต์แวร์จะมีความชัดเจนมากขึ้น และ สามารถนำไปใช้งานได้อย่างสอดคล้องต้องกันทั้งหน่วยงาน

คำว่า **ความสามารถของกระบวนการซอฟต์แวร์** (Software process capability) หมายความถึงพิสัยของผลลัพธ์ที่อาจเกิดขึ้นได้จากการดำเนินงานตามกระบวนการซอฟต์แวร์นี้ ความสามารถของกระบวนการซอฟต์แวร์ของหน่วยงานแห่งหนึ่งสามารถใช้พยากรณ์ได้ว่าจะเกิดผลลัพธ์อย่างไรกับโครงการซอฟต์แวร์ต่อไปของหน่วยงานนั้น

คำว่า **ผลสำเร็จของกระบวนการซอฟต์แวร์** (Software process performance) หมายถึงผลที่เกิดขึ้นจริง ๆ จากการใช้กระบวนการซอฟต์แวร์ ดังนั้น ผลสำเร็จของกระบวนการซอฟต์แวร์จึง

หมายถึงผลลัพธ์ที่เกิดขึ้น ส่วนความสามารถของกระบวนการซอฟต์แวร์หมายถึงผลลัพธ์ที่คาดหมายได้ว่าจะเกิดขึ้น

คำว่า **ความเจริญก้าวหน้าของกระบวนการซอฟต์แวร์** (Software process maturity) หมายถึงระดับในการกำหนด จัดการ วัดขนาด ควบคุม และ ดำเนินการให้ได้ผลซึ่งกระบวนการซอฟต์แวร์หนึ่ง ๆ ความเจริญก้าวหน้านั้นหมายถึงศักยภาพในการเติบโตทางด้านความสามารถ และ เป็นเครื่องชี้วัดทั้งความร่ำรวยของกระบวนการซอฟต์แวร์ขององค์การ และ ความสม่ำเสมอในการใช้กระบวนการซอฟต์แวร์นี้ในโครงการซอฟต์แวร์ต่าง ๆ ทั้งองค์การ

ระดับความเจริญก้าวหน้าของกระบวนการซอฟต์แวร์

นักวิชาการของสถาบัน SEI เชื่อว่าความเจริญก้าวหน้าทางด้านซอฟต์แวร์นั้นไม่สามารถจะเนรมิตรให้เกิดขึ้นในพริบตา แต่จะเกิดขึ้นได้จากการปรับปรุงการทำงานไปทีละน้อย ๆ นักซอฟต์แวร์ของ SEI ใช้หลักการด้านคุณภาพของนักวิชาการด้านคุณภาพที่มีชื่อเสียงของโลก อาทิ วอลเตอร์ ชิวฮาร์ต, เอ็ดเวิรด เดมิง, โจเซฟ ฮูรัน, และ ฟิลิป ครอสบี ในการกำหนดขั้นตอนหรือระดับความเจริญก้าวหน้าของกระบวนการซอฟต์แวร์ โดยแบ่งออกเป็นห้าระดับดังนี้

ระดับที่ 1 ระดับตั้งต้น (Initial level) หน่วยงานที่อยู่ในระดับนี้ยังไม่สามารถสร้างสภาวะที่มั่นคงในการพัฒนาซอฟต์แวร์ได้ หน่วยงานยังมีความยากลำบากในการจัดกระบวนการซอฟต์แวร์ที่เป็นระบบ ส่งผลให้เกิดวิกฤติการณ์ในการทำงานอยู่เสมอ เมื่อเกิดวิกฤติการณ์ขึ้นผู้พัฒนาซอฟต์แวร์ก็จะเลิกทำงานตามที่วางแผนไว้ แล้วรีบด่วนเขียนคำสั่งและทดสอบแทน ความสำเร็จของการพัฒนาซอฟตแวร์ในระดับนี้ขึ้นอยู่กับประสบการณ์และความสามารถของหัวหน้าโครงการ และ ทีมงานที่มีประสิทธิผล หากได้หัวหน้าโครงการที่มีความสามารถ และกล้าคิดกล้าทำ มาปรับปรุงการพัฒนาซอฟต์แวร์ก็จะทำให้หน่วยงานมีกระบวนการซอฟต์แวร์ที่ดีขึ้น แต่ถ้าหากหัวหน้าโครงการเช่นนี้ลาออกไปหน่วยงานก็จะกลับไปสู่ระดับเดิม เพราะไม่ว่าจะใช้หลักการวิศวกรรมที่เยี่ยมขนาดไหน แต่ถ้าหากกระบวนการจัดการยังไม่ดีพอ ความพยายามที่จะปรับปรุงก็ไร้ประโยชน์ หน่วยงานที่อยู่ในระดับนี้อาจพัฒนาซอฟต์แวร์ได้สำเร็จ แต่มักจะประสบกับปัญหายุ่งเหยิง และการพัฒนาซอฟต์แวร์มักจะใช้เวลาและงบประมาณเกินที่กำหนดไว้ อาจกล่าวได้ว่าความสามารถที่ระดับนี้เป็นความสามารถของบุคคลมากกว่าขององค์การ

ระดับที่ 2 ระดับทำซ้ำได้ (Repeatable level) หน่วยงานที่อยู่ในระดับนี้เริ่มมีนโยบายในการบริหารจัดการโครงการซอฟต์แวร์และมีการกำหนดขั้นตอนการนำนโยบายไปใช้ การวางแผนและจัดการโครงการใหม่มักจะขึ้นกับประสบการณ์จากโครงการที่คล้ายกัน ความสามารถของกระบวนการนั้นเกิดจากการนำขั้นตอนพื้นฐานทางด้านการบริหารโครงการมาใช้ กระบวนการที่ได้ผลมีลักษณะเป็นงานที่มีการจัดทำเอกสาร ควบคุม มีการฝึกอบรม มีการวัดผล และ สามารถปรับปรุงให้ดีขึ้นได้ โครงการในระดับนี้มีการควบคุมและติดตามการทำงานตามภารกิจต่าง ๆ ที่

กำหนดในแผน มีการดูแลค่าใช้จ่ายต่าง ๆ ว่าเป็นไปตามที่กำหนดหรือไม่ เราอาจกล่าวได้ว่าลักษณะของระดับนี้ก็คือใช้ผลสำเร็จของโครงการที่ผ่านมาเป็นตัวอย่าง มีการกำหนดมาตรฐานโครงการ และมีการจัดรูปแบบองค์กรให้งานโครงการดำเนินไปได้ดี

ระดับที่ 3 ระดับชัดเจน (Defined Level) ในหน่วยงานที่อยู่ระดับนี้ จะมีการบันทึกทำเอกสารเกี่ยวกับกระบวนการมาตรฐานในการพัฒนาและบำรุงรักษาซอฟต์แวร์เอาไว้ อีกทั้งยังมีการทำเอกสารเกี่ยวกับงานวิศวกรรมซอฟต์แวร์และกระบวนการจัดการต่าง ๆ ด้วย ที่สำคัญคือก็คือกระบวนการเหล่านี้จะถูกนำมาผสมรวมกันเป็นเนื้อเดียว ในงาน CMM นั้น ถือว่ากระบวนการมาตรฐานนั้นเป็นเรื่องที่สำคัญมากของทั้งองค์การ กระบวนการในระดับ 3 นี้ช่วยให้หัวหน้าโครงการซอฟต์แวร์และลูกทีมทำงานได้อย่างมีประสิทธิผล หน่วยงานสามารถใช้แนวทางของวิศวกรรมซอฟต์แวร์ได้ผลขึ้นเมื่อมีการกำหนดมาตรฐานกระบวนการซอฟต์แวร์ นอกจากนี้หน่วยงานอาจจัดตั้งกลุ่มกระบวนการวิศวกรรมซอฟต์แวร์ขึ้นเพื่อรับผิดชอบงานต่าง ๆ ที่เกี่ยวกับกระบวนการซอฟต์แวร์ของหน่วยงานด้วย ที่สำคัญก็คือมีการจัดฝึกอบรมอย่างกว้างขวางเพื่อให้ผู้บริหารโครงการและลูกทีมมีความรู้และทักษะที่สามารถทำงานที่กำหนดได้ดี โครงการต่าง ๆ ที่ทำในระดับนี้จะช่วยให้หน่วยงานปรับเปลี่ยนกระบวนการซอฟต์แวร์ของตนตามลักษณะพิเศษของโครงการได้ กระบวนการซอฟต์แวร์ที่ปรับเปลี่ยนแล้วนี้ ทาง CMM เรียกว่ากระบวนการซอฟต์แวร์ที่ชัดเจน (Defined software process) (ความหมายของคำนี้คงจะต้องให้เทียบเคียงกับคำว่า Well-defined function ทางคณิตศาสตร์ ซึ่งราชบัณฑิตยสถานได้กำหนดให้ใช้ศัพท์ไทยว่าฟังก์ชันที่แจ่มชัด) กระบวนการซอฟต์แวร์ที่ชัดเจนจะต้องมีกระบวนการทางวิศวกรรมซอฟต์แวร์และกระบวนการจัดการที่แจ่มชัดด้วย และจะแสดงได้ด้วยการมีเงื่อนไขที่ชัดเจน มีมาตรฐานและวิธีการสำหรับทำงานต่าง ๆ มีกลไกในการตรวจสอบ มีการกำหนดผลลัพธ์ และ เงื่อนไขการจบโครงการ โดยที่โครงการพัฒนาซอฟต์แวร์มีความชัดเจนเช่นนี้ ฝ่ายบริหารจึงวางใจได้ว่าจะสามารถตรวจสอบความก้าวหน้าในการดำเนินงานโครงการได้ตลอดเวลา เราอาจกล่าวได้ว่าความสามารถในกระบวนการซอฟต์แวร์ระดับนี้ก็คือการมีมาตรฐานและความคล้องจองกันทั้งหน่วยงาน

ระดับที่ 4 ระดับจัดการ (Managed Level) หน่วยงานที่มีความสามารถอยู่ในระดับนี้สามารถกำหนดคุณภาพในเชิงจำนวนให้แก่ซอฟต์แวร์และกระบวนการซอฟต์แวร์ได้ หน่วยงานสามารถวัดคุณภาพและผลผลิตของกระบวนการซอฟต์แวร์สำคัญ ๆ ของทุกโครงการได้ โดยถือว่าเป็นส่วนหนึ่งของกระบวนการวัดผลงานขององค์การ มีการจัดทำฐานข้อมูลสำหรับบันทึกและวิเคราะห์ข้อมูลเกี่ยวกับการใช้กระบวนการซอฟต์แวร์ที่ชัดเจน การควบคุมโครงการต่าง ๆ ทำได้โดยการพยายามทำให้ผลการดำเนินงานมีความสม่ำเสมอมากขึ้น มีการกำหนดความเสี่ยงในการพัฒนาระบบและควบคุมความเสี่ยงให้ลดน้อยลง เราอาจกล่าวได้ว่าหน่วยงานที่มีการพัฒนาซอฟต์แวร์อยู่ในระดับนี้คือหน่วยงานที่สามารถวัดผลและพยากรณ์ผลที่จะเกิดในการทำงานโครง

การซอฟต์แวร์ได้อย่างแม่นยำ สามารถพยากรณ์แนวโน้มและคุณภาพของซอฟต์แวร์ได้ดี ในเมื่อสภาวะการทำงานค่อนข้างลงตัว เมื่อมีโครงการแปลก ๆ เข้ามาให้ทำ หน่วยงานก็สามารถปรับกระบวนการได้เป็นอย่างดี

ระดับที่ 5 ระดับเหมาะที่สุด (Optimizing Level) หน่วยงานที่อยู่ระดับนี้เป็นหน่วยงานที่เน้นในด้านการปรับปรุงกระบวนการอยู่ตลอดเวลา หน่วยงานมีวิธีการกำหนดจุดอ่อนและจุดแข็งของกระบวนการในเชิงรุก โดยมีเป้าหมายในการป้องกันไม่ให้เกิดข้อบกพร่องขึ้น หน่วยงานใช้ข้อมูลเกี่ยวกับประสิทธิผลของกระบวนการซอฟต์แวร์ในการวิเคราะห์ต้นทุนและกำไรของเทคโนโลยีใหม่ ๆ เพื่อใช้เสนอการเปลี่ยนแปลงกระบวนการซอฟต์แวร์ของหน่วยงาน มีการกำหนดว่านวัตกรรมใดเหมาะที่สุดสำหรับหน่วยงาน จากนั้นก็ถ่ายทอดไปใช้ทั้งองค์การ ทีมงานซอฟต์แวร์ในระดับนี้ทำหน้าที่วิเคราะห์ข้อบกพร่องเพื่อหาสาเหตุ มีการประเมินกระบวนการซอฟต์แวร์เพื่อป้องกันไม่ให้เกิดข้อบกพร่องซ้ำอีกและนำความรู้ที่ได้นั้นไปถ่ายทอดให้ทีมงานพัฒนาซอฟต์แวร์ทราบ นอกจากนั้นมีการวิเคราะห์ความสูญเสียในการดำเนินงานและพยายามลดไม่ให้เกิดความสูญเสียเหล่านั้นด้วย หน่วยงานที่มีความสามารถในระดับนี้คือหน่วยงานที่พยายามปรับปรุงตนอยู่ตลอดเวลา การปรับปรุงนี้มีทั้งที่ค่อยเป็นค่อยไป และ การปรับปรุงโดยการนำนวัตกรรมใหม่ ๆ มาใช้

กลุ่มกระบวนการหลัก

สถาบัน SEI ได้กำหนดกลุ่มกระบวนการหลัก (Key Process Area หรือ KPA) สำหรับระดับความสามารถแต่ละระดับเอาไว้ด้วย กลุ่มกระบวนการหลักเหล่านี้ใช้อธิบายฟังก์ชันต่าง ๆ ทางด้านวิศวกรรมซอฟต์แวร์ที่จะต้องมีในแต่ละระดับ การกำหนดกลุ่มกระบวนการหลักแต่ละรายการจะต้องจำแนกตามสมบัติต่อไปนี้

- *เป้าหมาย* วัตถุประสงค์หลักที่ KPA แต่ละรายการจะต้องบรรลุให้ได้
- *ข้อตกลง* ข้อกำหนดที่ระบุให้หน่วยงานต้องดำเนินงานให้บรรลุเป้าหมาย และเป็นการยืนยันว่าจะพยายามทำตามเป้าหมาย
- *ความสามารถ* สมบัติที่จำเป็นจะต้องมีทั้งทางด้านองค์กรและในเชิงเทคโนโลยีเพื่อให้หน่วยงานดำเนินงานตามข้อตกลง
- *กิจกรรม* ภารกิจที่จำต้องทำเพื่อให้บรรลุ KPA
- *วิธีการตรวจการดำเนินงาน* แนวทางในการตรวจกิจกรรมว่าดำเนินไปเช่นใด
- *วิธีการตรวจสอบผลการดำเนินงาน* แนวทางในการตรวจสอบการดำเนินงาน KPA ว่าดำเนินการได้ถูกต้องเหมาะสม

กลุ่มกระบวนการหลักที่ได้กำหนดขึ้นสำหรับการดำเนินงานเกี่ยวกับซอฟต์แวร์ในแต่ละระดับความสามารถมีดังต่อไปนี้

ระดับที่ 1

ไม่มี KPA

ระดับที่ 2

Software configuration management

Software quality assurance

Software subcontract management

Software project tracking and oversight

Software project planning

Requirements management

ระดับที่ 3

Peer reviews

Intergroup coordination

Software product engineering

Integrated software management

Training program

Organization process definition

Organization process focus

ระดับที่ 4

Software quality management

Quantitative process management

ระดับที่ 5

Process change management

Technology change management

Defect prevention

สรุป

ในที่นี้ผมคงไม่ต้องสรุปอะไรมาก ท่านผู้อ่านคงจะเห็นแล้วว่าการจะไต่ระดับขึ้นไปสู่ความเปนเลิศในด้านการพัฒนาซอฟต์แวร์นั้นไม่ใช่เรื่องง่ายเลย และเมื่อพิจารณากลุ่มกระบวนการหลักที่เราจะต้องดำเนินการแล้ว จะพบเห็นว่าประเด็นสำคัญของการพัฒนาซอฟต์แวร์นั้นไม่ได้อยู่

ที่ด้านเทคนิคแต่อย่างไร หากอยู่ที่กระบวนการจัดการมากกว่า หากหน่วยงานใดมีผู้บริหารที่เอาใจใส่ในเรื่องคอมพิวเตอร์ และสนใจต้องการพัฒนาซอฟต์แวร์ที่มีคุณภาพ ตลอดจนได้กำหนดวิธีการและขั้นตอนในการบริหารงานในส่วนนี้อย่างเป็นระบบได้แล้ว หน่วยงานนั้นก็มีโอกาสเลื่อนระดับชั้น CMM ขึ้นไปสู่ความเป็นเลิศได้ แต่หน่วยงานใดไม่มีผู้บริหารที่สนใจด้านนี้เลย คงมีแต่เพียงนักคอมพิวเตอร์เท่านั้น หน่วยงานเช่นนี้ก็คงจะไต่ระดับได้เพียงแค่ระดับหนึ่งหรือสองอย่างแน่นอน

ก่อนจบ มีข่าวดีที่ต้องการเรียนให้ท่านผู้อ่านทราบว่า โครงการอุทยานซอฟต์แวร์ได้เห็นความสำคัญของการพัฒนาความเป็นเลิศในด้านซอฟต์แวร์ของหน่วยงานต่าง ๆ ในไทย ดังนั้นจึงมีดำริที่จะเชิญให้บุคลากรจากสถาบัน SEI เดินทางเข้ามาถ่ายทอดความรู้และประสบการณ์ในด้านนี้แก่นักซอฟต์แวร์ไทย รูปแบบการจัดและเนื้อหาจะเป็นเช่นใด กรุณาติดต่อสอบถามได้ที่ นส. บุษบา โตวิวัฒน์ โครงการอุทยานซอฟต์แวร์ เนคเทค สวทช. โทรศัพท์……..โทรสาร……และ อีเมล์……

สวัสดีครับ

บรรณานุกรม

1. Pressman, R.S., Software Engineering: A practitioner's Approach, Fourth Edition, McGraw-Hill, Singapore, 1997.
2. SEI, About the SEI – Welcome, http:www.sei.cmu.edu:80/about/about.html
3. Paulk, M.C., et al, The Capability Maturity Model for Software, Version 1.1, Technical Report, CMU/SEI-93-TR-024, 1993.