การพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัวตามหลักการความเป็นส่วนตัวโดยใช้ยูเอ็มแอล

นางสาวธีราพร ศุภกุล

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2559

Development of Privacy Design Patterns Based on Privacy Principles Using UML

Miss Theeraporn Suphakul

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Software Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2016

หัวข้อวิทยานิพนธ์	การพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัวตามหลักการความเป็นส่วนตัวโดยใช้ยูเอ็มแอล
โดย	นางสาวธีราพร ศุภกุล
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(รองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(รองศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา)

.....กรรมการ
(รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์)

.....กรรมการภายนอกมหาวิทยาลัย
(ผู้ช่วยศาสตราจารย์ ดร.มฑุปายาส ทองมาก)

ธีราพร ศุภกุล : การพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัวตามหลักการความเป็นส่วนตัวโดยใช้ยูเอ็มแอล (Development of Privacy Design Patterns Based on Privacy Principles Using UML) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา, 280 หน้า.

ความเป็นส่วนตัวเป็นคุณลักษณะเชิงคุณภาพที่สำคัญของซอฟต์แวร์ เนื่องจากข้อมูลส่วนบุคคลของผู้ใช้ซอฟต์แวร์จะถูกรวบรวม จัดเก็บ ประมวลผล และถ่ายโอนโดยซอฟต์แวร์ ผู้ใช้จึงควรได้รับการรับประกันว่าข้อมูลส่วนบุคคลของตนได้รับการปกป้องดูแลอย่างเหมาะสม จากการที่หลักการความเป็นส่วนตัวควรจะถูกนำไปพิจารณารวมเข้าไปในการออกแบบแอปพลิเคชัน งานวิจัยนี้จึงมีเป้าหมายที่จะสนับสนุนการออกแบบที่มีการคำนึงถึงเรื่องความเป็นส่วนตัว และได้นำเสนอการพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัว ซึ่งเป็นไปตามหลักการความเป็นส่วนตัวขององค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา โดยได้อธิบายถึงรายละเอียดของหลักการ และการประยุกต์ใช้แบบรูปการออกแบบและพัฒนาแอปพลิเคชัน งานวิจัยได้เสนอแบบจำลองการออกแบบซอฟต์แวร์ตามหลักการความเป็นส่วนตัว โดยใช้ภาษายูเอ็มแอล เพื่อให้สามารถนำไปใช้ในการออกแบบแอปพลิเคชันที่คำนึงถึงความเป็นส่วนตัวได้ และได้ใช้แอปพลิเคชันร้านค้าออนไลน์เป็นกรณีศึกษาในการประยุกต์ใช้แบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่เสนอ

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2559

5870175821 : MAJOR SOFTWARE ENGINEERING
KEYWORDS: DESIGN PATTERNS / UML / OECD PRIVACY PRINCIPLES

THEERAPORN SUPHAKUL: Development of Privacy Design Patterns Based on Privacy Principles Using UML. ADVISOR: ASSOC. PROF. DR.TWITTIE SENIVONGSE, 280 pp.

Privacy is a major quality attribute of any software. Since personal data of users are collected, stored, processed, and transferred by the applications they use, they need to be assured that proper data protection is in place. Since privacy principles should be taken into account and incorporated into application design, this research aims to promote privacy by design and presents a development of privacy design patterns. The patterns follow the privacy principles of the Organisation for Economic Co-operation and Development (OECD) and describe details of the privacy principles and how to apply them to the design and implementation of the applications. Software design models realizing the privacy principles are also proposed, using UML notations, so as to enable reuse of the design in privacy-aware applications. An online shop application is used as a case study to apply the proposed privacy design patterns.

Department: Computer Engineering Student's Signature .....
Field of Study: Software Engineering Advisor's Signature .....
Academic Year: 2016

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้เนื่องจากได้รับความกรุณาจาก รองศาสตร์จารย์ ดร. ทวิตีย์ เสนีวงศ์ ณ อยุธยา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้คำแนะนำ และแนวทางในการทำวิทยานิพนธ์ ตลอดจนการปรับปรุงแก้ไขข้อบกพร่องต่าง ๆ ในงานวิทยานิพนธ์ด้วยความเอาใจใส่ ทำให้ข้าพเจ้าสามารถดำเนินงานวิทยานิพนธ์นี้จนกระทั่งประสบผลสำเร็จ และได้งานวิทยานิพนธ์ที่มีคุณภาพ

ขอกราบขอบพระคุณคณะกรรมการสอบวิทยานิพนธ์ทุกท่าน ได้แก่ รองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี ประธานกรรมการ รศ. ดร.ธาราทิพย์ สุวรรณศาสตร์ กรรมการ และผู้ช่วยศาสตราจารย์ ดร.มฑุปายาส ทองมาก กรรมการภายนอกมหาวิทยาลัย ที่ได้กรุณาสละเวลาให้คำแนะนำเกี่ยวกับการทำวิทยานิพนธ์ และพิจารณาเนื้อหาให้ถูกต้องและครบถ้วน เพื่อให้วิทยานิพนธ์ฉบับนี้มีคุณภาพและมีความสมบูรณ์ยิ่งขึ้น

ขอกราบขอบพระคุณคณาจารย์ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย ที่ได้ให้ความรู้ทั้งทางด้านวิชาการ การใช้ชีวิตในสังคม และด้านคุณธรรม

สุดท้ายขอขอบคุณเพื่อน ๆ สาขาวิศวกรรมซอฟต์แวร์ รวมถึงพี่ ๆ และเพื่อนของข้าพเจ้าที่คอยช่วยเหลือ เป็นกำลังใจ ให้คำแนะนำแก่ข้าพเจ้า และได้สละเวลามาช่วยประเมินงานวิทยานิพนธ์ทำให้งานวิทยานิพนธ์นี้สำเร็จลุล่วงไปได้

สารบัญ

หน้า

หน้า

หน้า

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การพัฒนาซอฟต์แวร์โดยทั่วไปจะเกี่ยวข้องกับข้อมูลส่วนบุคคลของผู้ใช้ เช่น ชื่อ เลขประจำตัวประชาชน วันเกิด เบอร์โทรศัพท์ อีเมล ตำแหน่งที่ตั้ง ข้อมูลทางการเงิน เป็นต้น ซึ่งหากข้อมูลส่วนบุคคลเหล่านี้ถูกละเมิด หรือนำไปใช้ในทางที่ผิดจะทำให้เกิดปัญหากับผู้เป็นเจ้าของข้อมูลหรือผู้ที่เกี่ยวข้องได้ ดังนั้นจึงต้องมีการคำนึงถึงการคุ้มครองความเป็นส่วนตัว (Privacy) สำหรับข้อมูลส่วนบุคคลของผู้ใช้ โดยหลักการความเป็นส่วนตัว (Privacy Principles) ที่ได้รับความนิยมในระดับสากลหลักการหนึ่ง คือ หลักการความเป็นส่วนตัวขององค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา หรือ โออีซีดี (Organization for Economic Co-Operation and Development: OECD) โดยมีหลักการ 8 ประการคือ หลักข้อจำกัดในการเก็บรวบรวมข้อมูล หลักคุณภาพของข้อมูล หลักการกำหนดวัตถุประสงค์ หลักข้อจำกัดในการใช้ข้อมูล หลักการรักษาความมั่นคงของข้อมูล หลักการเปิดเผย หลักการมีส่วนร่วมของบุคคล และหลักความรับผิดชอบ [1] หากการพัฒนาซอฟต์แวร์มีการคำนึงถึงหลักการความเป็นส่วนตัว จะเป็นการสร้างความเชื่อมั่นให้กับผู้ใช้ว่าข้อมูลส่วนบุคคลได้รับการคุ้มครอง และดูแลโดยซอฟต์แวร์ที่ใช้งาน

เนื่องจากหลักการความเป็นส่วนตัวที่อยู่ในรูปแบบของเอกสาร หรือภาษาธรรมชาตินั้นยากที่นักพัฒนาซอฟต์แวร์จะทำความเข้าใจ จึงอาจทำให้การพัฒนาถูกละเลยไปในส่วนของการคุ้มครองความเป็นส่วนตัวของข้อมูล งานวิทยานิพนธ์นี้จึงนำเสนอแบบรูปการออกแบบสำหรับความเป็นส่วนตัว (Privacy Design Patterns) เพื่อให้ง่ายต่อการพัฒนา และช่วยสนับสนุนให้การพัฒนาซอฟต์แวร์มีการคุ้มครองความเป็นส่วนตัวตามหลักการความเป็นส่วนตัวขององค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา

ในปัจจุบันมีงานวิจัยจำนวนหนึ่งที่นำเสนอแบบรูปสำหรับความเป็นส่วนตัว เช่น งานวิจัย Privacy Patterns for Online Interaction ที่นำเสนอโดย S. Romanosky และคณะ ได้นำเสนอแบบรูปสำหรับความเป็นส่วนตัว 3 แบบรูป [2] งานวิจัย Towards Organizational Privacy Patterns นำเสนอโดย J. Porekar ได้นำเสนอแบบรูปสำหรับความเป็นส่วนตัว 2 แบบรูป [3] เป็นต้น แต่งานวิจัยเหล่านี้ไม่ได้นำเสนอแบบรูปในรูปแบบของแบบจำลองซอฟต์แวร์สำหรับให้นักพัฒนาซอฟต์แวร์นำไปประยุกต์ใช้ได้โดยสะดวก ในงานวิทยานิพนธ์นี้ผู้วิจัยจะทำการศึกษาหลักการความ

เป็นส่วนตัว ปัญหาที่เกิดขึ้นในการคุ้มครองความเป็นส่วนตัว และผลเฉลย (Solution) ที่แก้ปัญหาเหล่านั้นจากงานวิจัยอื่น ๆ ที่มีอยู่ เพื่อนำมากำหนดรายการความต้องการสำหรับแต่ละแบบรูปการออกแบบ และออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่ครอบคลุมหลักการความเป็นส่วนตัวขององค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา โดยนำเสนอในรูปแบบของแบบจำลองซอฟต์แวร์ยูเอ็มแอล (Unified Modeling Language Diagram: UML Diagram) โดยใช้แผนภาพคลาส (Class Diagram) แผนภาพลำดับ (Sequence Diagram) และแผนภาพกิจกรรม (Activity Diagram) และนำเสนอตัวอย่างการนำไปใช้ ซึ่งแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่เสนอนี้จะช่วยให้นักพัฒนาซอฟต์แวร์นำไปประยุกต์ใช้ได้สะดวกขึ้น อันจะเป็นการสนับสนุนให้การพัฒนาซอฟต์แวร์มีการคำนึงถึงความเป็นส่วนตัวของผู้ใช้งานมากยิ่งขึ้น

1.2 วัตถุประสงค์ของงานวิจัย

นำเสนอแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่ครอบคลุมหลักการความเป็นส่วนตัวของโออีซีดี

1.3 ขอบเขตงานวิจัย

1. หลักการความเป็นส่วนตัวที่ใช้ในงานวิจัยคือ หลักการความเป็นส่วนตัวของโออีซีดี
2. กำหนดรายการความต้องการของแต่ละแบบรูปการออกแบบสำหรับความเป็นส่วนตัวจากหลักการความเป็นส่วนตัว เพื่อนำมาใช้ในการออกแบบแบบจำลองยูเอ็มแอล
3. ออกแบบแบบจำลองยูเอ็มแอลสำหรับแบบรูปการออกแบบสำหรับความเป็นส่วนตัว โดยครอบคลุมเฉพาะแผนภาพคลาส แผนภาพลำดับ และแผนภาพกิจกรรม
4. นำเสนอแบบรูปการออกแบบสำหรับความเป็นส่วนตัว ซึ่งประกอบด้วยคำอธิบายแบบรูป และแบบจำลองยูเอ็มแอล
5. การตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับความเป็นส่วนตัวจะใช้รายการตรวจสอบ โดยให้ผู้ที่มีประสบการณ์ในการพัฒนาซอฟต์แวร์ หรือวิเคราะห์และออกแบบระบบที่เกี่ยวข้องกับข้อมูลส่วนบุคคลอย่างน้อย 2 ปี และเป็นจำนวนอย่างน้อย 5 คนเป็นผู้ทำการตรวจสอบ
6. แสดงตัวอย่างการนำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวไปใช้โดยพัฒนาแอปพลิเคชันที่มีการนำแบบรูปการออกแบบที่นำเสนอไปใช้ในการพัฒนา จำนวน 1 ระบบ

1.4 ขั้นตอนการดำเนินงาน

1) ศึกษาองค์ความรู้และทฤษฎีที่เกี่ยวข้องกับงานวิจัย
2) วิเคราะห์หลักการความเป็นส่วนตัว
3) กำหนดรายการความต้องการของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว
4) ออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัว
5) ตรวจสอบความครบถ้วน และปรับปรุงแบบรูปการออกแบบสำหรับความเป็นส่วนตัว
6) แสดงตัวอย่างการนำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอไปใช้
7) สรุปผลการวิจัย
8) จัดทำบทความวิจัย
9) จัดทำรายงานวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ได้แบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่ครอบคลุมหลักการความเป็นส่วนตัวของโออีซีดี ซึ่งช่วยให้ง่ายต่อการพัฒนา และช่วยสนับสนุนให้การพัฒนาซอฟต์แวร์คำนึงถึงการคุ้มครองความเป็นส่วนตัวตามหลักการความเป็นส่วนตัว

1.6 โครงสร้างของเนื้อหาในวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้แบ่งเนื้อหาออกเป็น 6 บทคือ บทที่ 1 บทนำ กล่าวถึงความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ ขอบเขต ขั้นตอนการดำเนินงาน และประโยชน์ที่คาดว่าจะได้รับ บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง กล่าวถึงองค์ความรู้ ทฤษฎี และงานวิจัยที่มีอยู่ที่เกี่ยวข้อง บทที่ 3 แนวคิดและวิธีการดำเนินงานวิจัย กล่าวถึงวิธีการ และขั้นตอนในการออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัว บทที่ 4 การตรวจสอบความครบถ้วนของแบบรูปการออกแบบ กล่าวถึงวิธีการในการตรวจสอบความครบถ้วนของแบบรูปการออกแบบที่นำเสนอ และผลลัพธ์ที่ได้จากการตรวจสอบ บทที่ 5 การนำแบบรูปการออกแบบไปประยุกต์ใช้ กล่าวถึงตัวอย่างการนำแบบรูปการออกแบบที่นำเสนอไปประยุกต์ใช้ และบทที่ 6 สรุปผลการวิจัย กล่าวถึงผลสรุปของงานวิจัย ข้อจำกัด แนวคิดในการนำไปพัฒนาต่อ และผลงานการตีพิมพ์

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 หลักการความเป็นส่วนตัวขององค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา

องค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา หรือ โออีซีดี (Organization for Economic Co-Operation and Development : OECD) เป็นองค์การที่ส่งเสริมนโยบายในการปรับปรุงเศรษฐกิจและสังคมความเป็นอยู่ของผู้คนทั่วโลก โออีซีดีได้เสนอหลักการความเป็นส่วนตัว (Privacy Principles) ของข้อมูลส่วนบุคคลโดยระบุความหมายของคำศัพท์ต่าง ๆ ที่เกี่ยวข้องกับหลักการความเป็นส่วนตัวไว้ดังนี้ [1]

1) **“Data controller”** means a party who, according to national law, is competent to decide about the contents and use of personal data regardless of whether or not such data are collected, stored, processed or disseminated by that party or by an agent on its behalf.
2) **“Personal data”** means any information relating to an identified or identifiable individual (data subject).
3) **“Laws protecting privacy”** means national laws or regulations, the enforcement of which has the effect of protecting personal data consistent with these Guidelines.
4) **“Privacy enforcement authority”** means any public body, as determined by each Member country, that is responsible for enforcing laws protecting privacy, and that has powers to conduct investigations or pursue enforcement proceedings.
5) **“Transborder flows of personal data”** means movements of personal data across national borders.

สำหรับหลักการความเป็นส่วนตัวของข้อมูลส่วนบุคคล โออีซีดีได้เสนอหลักการไว้ 8 ประการ ดังนี้ [1]

1) Collection Limitation Principle

There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject.

2) Data Quality Principle

Personal data should be relevant to the purposes for which they are to be used, and, to the extent necessary for those purposes, should be accurate, complete and kept up-to-date.

3) Purpose Specification Principle

The purposes for which personal data are collected should be specified not later than at the time of data collection and the subsequent use limited to the fulfilment of those purposes or such others as are not incompatible with those purposes and as are specified on each occasion of change of purpose.

4) Use Limitation Principle

Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with Purpose Specification Principle except:

a) with the consent of the data subject; or

b) by the authority of law.

5) Security Safeguards Principle

Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorized access, destruction, use, modification or disclosure of data.

6) Openness Principle

There should be a general policy of openness about developments, practices and policies with respect to personal data. Means should be readily available of establishing the existence and nature of personal data, and the main purposes of their use, as well as the identity and usual residence of the data controller.

7) Individual Participation Principle

Individuals should have the right:

b) to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to them;

c) to have communicated to them, data relating to them

I. within a reasonable time;

II. at a charge, if any, that is not excessive;

III. in a reasonable manner; and

IV. in a form that is readily intelligible to them;

d) to be given reasons if a request made under subparagraphs (a) and (b) is denied, and to be able to challenge such denial; and

e) to challenge data relating to them and, if the challenge is successful

8) Accountability Principle

A data controller should be accountable for complying with measures which give effect to the principles stated above.

2.1.3 แบบรูปการออกแบบ

แบบรูปการออกแบบ (Design Pattern) เป็นการแสดงโครงสร้างความสัมพันธ์ของระบบย่อย หรือส่วนประกอบของซอฟต์แวร์ที่อธิบายแนวทางในการแก้ปัญหาที่เกิดขึ้นบ่อยครั้งในการออกแบบ เพื่อนำมาใช้ในการแก้ปัญหาใหม่ที่เกิดขึ้นในลักษณะเดิม สำหรับรูปแบบของแบบรูปการออกแบบที่จะนำมาใช้ในงานวิทยานิพนธ์นี้มาจากหนังสือ “Design Patterns Elements of Reusable Object-Oriented Software” โดยมีองค์ประกอบของแบบรูปการออกแบบดังนี้ [4]

1) **ชื่อแบบรูป และการจำแนกประเภทของแบบรูป (Pattern Name and Classification)** ชื่อของแบบรูปที่แสดงถึงสาระสำคัญของแบบรูป ซึ่งการตั้งชื่อแบบรูปที่ดีช่วยให้การสื่อสารเป็นไปได้ง่าย ในส่วนการจำแนกประเภทของแบบรูปนั้นในงานวิทยานิพนธ์นี้จะจำแนกประเภทของแบบรูปตามหลักการความเป็นส่วนตัว
2) **จุดประสงค์ (Intent)** คำอธิบายสั้น ๆ เพื่อตอบคำถามว่า แบบรูปที่ออกแบบนั้นทำอะไร เหตุผลและจุดประสงค์ของแบบรูปคืออะไร และการออกแบบนั้นแก้ปัญหาอะไร
3) **ชื่ออื่น ๆ (Also Known As)** ชื่ออื่น ๆ ที่เป็นที่รู้จักของแบบรูป (ถ้ามี)

4) **แรงจูงใจ (Motivation)** สถานการณ์ที่แสดงให้เห็นถึงปัญหาของการออกแบบ และแบบรูปจะแก้ปัญหานั้นได้อย่างไร ซึ่งสถานการณ์นี้จะช่วยให้เข้าใจแบบรูปมากขึ้น
5) **การนำไปใช้ (Applicability)** สถานการณ์ที่แบบรูปนั้นสามารถนำไปปรับใช้ได้ วิธีการออกแบบที่ไม่ดีที่แบบรูปนั้นสามารถนำมาแก้ปัญหาได้
6) **โครงสร้าง (Structure)** อธิบายรายละเอียดโครงสร้างของแบบรูป โดยในงานวิทยานิพนธ์จะนำเสนอในรูปแบบของแบบจำลองซอฟต์แวร์ยูเอ็มแอลคือ แผนภาพแสดงโครงสร้างโดยใช้แผนภาพคลาส และแผนภาพแสดงพฤติกรรมโดยใช้แผนภาพกิจกรรม
7) **ผู้มีส่วนร่วม (Participants)** คลาสหรืออ็อบเจกต์ที่มีส่วนร่วมในการออกแบบแบบรูป
8) **การร่วมมือ (Collaboration)** วิธีการในการทำงานร่วมกันของคลาสหรืออ็อบเจกต์ที่มีส่วนร่วม และในงานวิทยานิพนธ์จะนำเสนอแผนภาพแสดงการโต้ตอบร่วมด้วย โดยใช้แผนภาพลำดับ
9) **ผลลัพธ์ (Consequences)** ผลลัพธ์ที่ได้ และสิ่งที่ต้องแลกเปลี่ยน (Trade-off) จากการประยุกต์ใช้แบบรูป เพื่อช่วยในการประเมินทางเลือกสำหรับการออกแบบ ต้นทุนและประโยชน์ที่ได้รับจากการประยุกต์ใช้แบบรูป
10) **การทำให้เกิดผล (Implementation)** อธิบายการทำให้เกิดผลของแบบรูป คำแนะนำหรือเทคนิคการนำแบบรูปไปประยุกต์ใช้ ซึ่งสามารถนำไปปรับใช้ให้ตรงกับความต้องการของผู้ใช้ได้
11) **ตัวอย่างโค้ด (Sample Code)** โค้ดแสดงการนำแบบรูปไปทำให้เกิดผล ในงานวิทยานิพนธ์จะนำเสนอตัวอย่างโค้ดในรูปแบบของภาษาจาวา (Java)
12) **การใช้ที่เป็นที่รู้จัก (Known Uses)** ตัวอย่างการใช้แบบรูปในระบบที่มีอยู่
13) **รูปแบบที่เกี่ยวข้อง (Related Patterns)** แบบรูปที่เกี่ยวข้องหรือใกล้เคียงกับแบบรูปที่ออกแบบ

2.1.4 แบบจำลองซอฟต์แวร์ยูเอ็มแอล

แบบจำลองซอฟต์แวร์ยูเอ็มแอล [5] เป็นเครื่องมือที่ใช้ในการสื่อสารระหว่างผู้ใช้งานกับผู้พัฒนา รวมถึงผู้พัฒนาด้วยกันเอง โดยการสร้างแบบจำลองซอฟต์แวร์ยูเอ็มแอลนั้นต้องการตอบสนองการออกแบบเชิงวัตถุ (Object-Oriented Design Principle) ซึ่งจะถูกนำเสนอในรูปแบบแผนภาพ โดยแบบจำลองซอฟต์แวร์ยูเอ็มแอล แบ่งออกเป็น 3 ประเภทด้วยกันคือ 1) แผนภาพแสดงโครงสร้าง (Structure Diagrams) ประกอบด้วย แผนภาพคลาส (Class Diagram) แผนภาพอ็อบเจกต์ (Object Diagram) แผนภาพส่วนโปรแกรม (Component Diagram) แผนภาพโครงสร้าง

ส่วนประกอบ (Composite Structure Diagram) แผนภาพแพคเกจ (Package Diagram) และแผนภาพการใช้งาน (Deployment Diagram) 2) แผนภาพแสดงพฤติกรรม (Behavior Diagrams) ประกอบด้วย แผนภาพยูสเคส (Use Case Diagram) แผนภาพกิจกรรม (Activity Diagram) และแผนภาพสถานะ (State Machine Diagram) และ 3) แผนภาพแสดงการโต้ตอบ (Interaction Diagrams) ประกอบด้วย แผนภาพลำดับ (Sequence Diagram) แผนภาพการสื่อสาร (Communication Diagram) แผนภาพเวลา (Timing Diagram) และแผนภาพแสดงภาพรวมการโต้ตอบ (Interaction Overview Diagram) ซึ่งในงานวิทยานิพนธ์นี้จะนำเสนอแบบรูปการออกแบบสำหรับความเป็นส่วนตัวโดยใช้แผนภาพคลาส แผนภาพลำดับ และแผนภาพกิจกรรม

1) แผนภาพคลาส (Class Diagram)

แผนภาพคลาสคือ แผนภาพที่ใช้แสดงความสัมพันธ์ระหว่างคลาส ซึ่งจะแสดงให้เห็นถึงโครงสร้างของระบบและข้อมูลที่ระบบควรจัดเก็บ โดยแผนภาพคลาสจะประกอบด้วยคลาส และกลุ่มความสัมพันธ์ระหว่างคลาส ในส่วนของคลาสจะแบ่งออกเป็น 3 ส่วนโดยแต่ละส่วนนั้นแสดงถึง ชื่อคลาส (Name) คุณสมบัติ (Attribute) และการทำงาน (Operations)

2) แผนภาพลำดับ (Sequence Diagram)

แผนภาพลำดับคือ แผนภาพที่แสดงให้เห็นปฏิสัมพันธ์ระหว่างอ็อบเจกต์ของคลาสตามลำดับเวลา โดยประกอบด้วย คลาสหรืออ็อบเจกต์ เส้นเพื่อใช้แสดงลำดับเวลา และเส้นเพื่อแสดงกิจกรรมที่เกิดขึ้นจากคลาสหรืออ็อบเจกต์

3) แผนภาพกิจกรรม (Activity Diagram)

แผนภาพกิจกรรมคือ แผนภาพที่ใช้อธิบายกิจกรรมที่เกิดขึ้นในลักษณะกระแสการไหลของการทำงาน (Workflow) ซึ่งจะมีลักษณะคล้ายกับผังงาน (Flowchart) โดยขั้นตอนในการทำงานแต่ละขั้นจะเรียกว่า กิจกรรม (Activity) โดยในแผนภาพกิจกรรมจะมีจุดเริ่มต้นกับจุดสิ้นสุด และในระหว่างจุดเริ่มต้นกับจุดสิ้นสุดจะมีขั้นตอนหรือกิจกรรมต่าง ๆ ในระบบ

2.2 งานวิจัยที่เกี่ยวข้อง

ในส่วนของงานวิจัยที่เกี่ยวข้องจะแบ่งเป็น 3 ส่วนด้วยกันคือ งานวิจัยที่แนะนำแนวทางสำหรับความเป็นส่วนตัว งานวิจัยที่เกี่ยวข้องกับแบบรูปความเป็นส่วนตัว งานวิจัยที่เกี่ยวข้องกับแบบจำลองซอฟต์แวร์สำหรับความเป็นส่วนตัว

2.2.1 งานวิจัยที่แนะนำแนวทางสำหรับความเป็นส่วนตัว

1) Assessing Privacy Protection in Alumni Service [6]

งานวิจัยนี้นำเสนอแนวทางสำหรับความเป็นส่วนตัวของระบบบริการสมาชิกศิษย์เก่า เรียกแนวทางนี้ว่า เซเว่นซี (7C) โดยมีพื้นฐานบนระบบบริการสมาชิกศิษย์เก่า แบบรูปความเป็นส่วนตัว ข้อบังคับการคุ้มครองความเป็นส่วนตัว และเทคโนโลยีที่ช่วยคุ้มครองความเป็นส่วนตัว ซึ่งแนวทางเซเว่นซีนี้จะประกอบไปด้วย 1) เนื้อหา (Content) เป็นขั้นตอนในการรวบรวม และทำความเข้าใจเนื้อหาของระบบ 2) ความลับ (Confidentiality) เป็นการระบุว่าเนื้อหาใดในระบบเป็นข้อมูลส่วนบุคคลที่ระบบจะต้องคุ้มครอง 3) การเชื่อมโยง (Connectivity) แสดงการเชื่อมโยงระหว่างฟังก์ชันการทำงานกับข้อมูลส่วนบุคคล 4) การยินยอม (Consent) เป็นการระบุว่าผู้ใช้สามารถควบคุมการเปิดเผยข้อมูลส่วนตัวได้อย่างไรบ้าง 5) ข้อบังคับ (Constraint) ข้อบังคับความเป็นส่วนตัวที่ผู้พัฒนาระบบจะต้องคำนึงถึง 6) ตัวควบคุม (Control) จะทำอย่างไรเพื่อรักษาข้อมูลส่วนบุคคลให้เป็นไปตามข้อกำหนด และระเบียบข้อบังคับ และ 7) วิธีการ (Construct) การเลือกเทคโนโลยีที่นำมาใช้ เพื่อทำให้การควบคุมนั้นเกิดขึ้นได้ โดยผู้วิจัยหวังว่าแนวทางเซเว่นซีจะทำให้นักพัฒนาระบบสามารถเข้าใจความเป็นส่วนตัวได้มากขึ้น

ในงานวิทยานิพนธ์นี้จะนำแนวทางเซเว่นซีในส่วนของตัวควบคุมที่อธิบายว่าจะใช้วิธีอย่างไรในการควบคุมเพื่อรักษาข้อมูลส่วนบุคคลให้เป็นไปตามข้อกำหนด และระเบียบข้อบังคับมาเป็นแนวทางในการสร้างแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

2) Privacy Design Strategies [7]

งานวิจัยนี้นำเสนอกลยุทธ์ในการออกแบบสำหรับความเป็นส่วนตัวโดยอยู่บนพื้นฐานของกฎหมายคุ้มครองข้อมูล นโยบายความเป็นส่วนตัวของโออีซีดี (OECD) และมาตรฐานไอเอสโอ 29100 (ISO/IEC 29100) โดยนำเสนอ 8 กลยุทธ์คือ 1) MINIMISE ปริมาณของข้อมูลส่วนบุคคลที่นำไปประมวลผลควรจะจำกัดให้มีจำนวนน้อยที่สุดที่เป็นไปได้ 2) HIDE ข้อมูลส่วนบุคคลใด ๆ และความสัมพันธ์ของข้อมูลควรจะถูกซ่อนจากมุมมองปกติ 3) SEPARATE การแยกการประมวลผลหรือการจัดเก็บข้อมูล 4) AGGREGATE ข้อมูลส่วนบุคคลที่นำมาใช้ในการประมวลผลควรจะให้มีความละเอียดน้อยที่สุด 5) INFORM เมื่อใดก็ตามที่ข้อมูลถูกนำไปประมวลผลควรมีการแจ้งไปยังเจ้าของข้อมูล 6) CONTROL การควบคุมการใช้งานข้อมูลส่วนบุคคล 7) ENFORCE การบังคับใช้นโยบายสำหรับการคุ้มครองความเป็นส่วนตัว และ 8) DEMONSTRATE การแสดงให้เห็นถึงนโยบายสำหรับการคุ้มครองความเป็นส่วนตัวและข้อกำหนดทางกฎหมายที่ใช้บังคับ นอกจากนี้ยังได้มีการแนะนำ

แบบรูปสำหรับความเป็นส่วนตัวที่เกี่ยวข้องสำหรับแต่ละกลยุทธ์ไว้ เช่น สำหรับการ HIDE แบบรูปสำหรับความเป็นส่วนตัวที่แนะนำคือ การเข้ารหัสข้อมูล (Encryption) สำหรับการ INFORM แบบรูปสำหรับความเป็นส่วนตัวที่แนะนำคือ การแจ้งเตือนการละเมิดข้อมูล (Data Breach Notification) เป็นต้น

ในงานวิทยานิพนธ์นี้จะนำกลยุทธ์ต่าง ๆ ในการออกแบบสำหรับความเป็นส่วนตัว และแบบรูปสำหรับความเป็นส่วนตัวที่ได้มีการแนะนำไว้ในงานวิจัยมาเป็นแนวทางในการสร้างแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

2.2.2 งานวิจัยที่เกี่ยวข้องกับแบบรูปความเป็นส่วนตัว

1) Privacy Patterns for Online Interactions [2]

งานวิจัยนี้นำเสนอแบบรูปความเป็นส่วนตัว 3 แบบรูปสำหรับการโต้ตอบออนไลน์ ได้แก่ 1) Inform consent for Web-Based Transactions เป็นแบบรูปที่อธิบายถึงเว็บไซต์ต้องมีการแจ้งให้กับผู้ใช้งานทราบเมื่อมีการเก็บรวบรวมและใช้งานข้อมูลส่วนบุคคล 2) Masked Online Traffic แบบรูปนี้นำเสนอผลเฉลยเพื่อคุ้มครองความเป็นส่วนตัวของผู้ใช้โดยการลดปริมาณของข้อมูลที่มีการเปิดเผยในขณะที่การโต้ตอบออนไลน์ และ 3) Minimal Information Asymmetry เป็นแบบรูปที่อธิบายว่าจะทำอย่างไรในการคุ้มครองความเป็นส่วนตัวของผู้ใช้โดยการรวบรวมข้อมูลเพิ่มเติมเกี่ยวกับบุคคลที่จะทำธุรกรรมด้วย เพื่อนำมาตัดสินใจและทำธุรกรรมเฉพาะกับบุคคลที่น่าไว้ใจ

2) Towards Organizational Privacy Patterns [3]

งานวิจัยนี้นำเสนอแบบรูปความเป็นส่วนตัว 2 แบบรูปคือ 1) Obtaining Explicit Consent อธิบายถึงวิธีการแก้ปัญหาสำหรับองค์กรที่ต้องการส่งมอบบริการที่จำเป็นต้องเก็บรวบรวมข้อมูลที่มีความอ่อนไหว ต้องมีการแจ้งให้กับผู้ใช้ทราบและได้รับอนุญาตจากผู้ใช้งานในการที่จะเก็บรวบรวมและใช้งานข้อมูลที่มีความอ่อนไหวเหล่านั้น และ 2) Access Control to Sensitive Data Based on Purpose อธิบายถึงวิธีการแก้ปัญหาข้อมูลส่วนบุคคลที่มีการเก็บรวบรวมจะต้องนำไปใช้ให้ตรงกับวัตถุประสงค์ที่ได้กำหนดไว้ และไม่ควรนำไปใช้สำหรับวัตถุประสงค์อื่น

3) A Pattern Collection for Privacy Enhancing Technology [8]

งานวิจัยนี้นำเสนอแบบรูปสำหรับส่วนต่อประสานผู้ใช้ (User Interfaces) ในการคุ้มครองความเป็นส่วนตัวโดยนำเสนอทั้งหมด 12 แบบรูป ซึ่งแบ่งออกเป็น 3 กลุ่มคือ PET Interaction จะกล่าวถึงแบบรูปที่เกี่ยวข้องกับการมีปฏิสัมพันธ์กับผู้ใช้งาน PET Patterns for Privacy Policies จะ

กล่าวถึงแบบรูปที่เกี่ยวข้องกับการประกาศนโยบายความเป็นส่วนตัวให้กับผู้ใช้ทราบ และ PET Visualization จะกล่าวถึงแบบรูปที่เกี่ยวข้องกับการแสดงนโยบายความเป็นส่วนตัวให้ผู้ใช้เข้าใจได้ง่ายขึ้น ตัวอย่างแบบรูปที่นำเสนอ เช่น Secure Passwords แบบรูปนี้จะช่วยให้ผู้ใช้สามารถสร้างและเลือกรหัสผ่านที่มั่นคงโดยการให้ข้อเสนอแนะที่เหมาะสม, Informed Consent แบบรูปนี้จะช่วยให้ผู้ใช้ทราบว่าข้อมูลส่วนบุคคลที่ถูกเก็บรวบรวมนั้นถูกนำไปใช้อย่างไร และ Privacy Aware Wording แบบรูปนี้จะกล่าวถึงการที่ผู้ใช้จะต้องเข้าใจเนื้อหาและเงื่อนไขของนโยบายความเป็นส่วนตัวได้อย่างชัดเจน เป็นต้น

ในงานวิทยานิพนธ์นี้จะศึกษาแบบรูปสำหรับความเป็นส่วนตัวที่มีอยู่ในปัจจุบัน เพื่อสกัดรายการความต้องการสำหรับผลเฉลยของแต่ละแบบรูป และนำรายการความต้องการที่ได้มาออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัวต่อไป ซึ่งจากงานวิจัยที่เกี่ยวข้องกับแบบรูปสำหรับความเป็นส่วนตัวที่ได้ศึกษามา แต่ละแบบรูปที่มีการนำเสนอนั้นจะไม่ได้มีการแสดงในส่วนของแบบจำลองซอฟต์แวร์ หรืออธิบายรายละเอียดโครงสร้างของแบบรูป ในงานวิทยานิพนธ์นี้จึงมีการนำเสนอแบบรูปการออกแบบสำหรับความเป็นส่วนตัว โดยนำเสนอในรูปแบบของแบบจำลองซอฟต์แวร์ยูเอ็มแอล โดยใช้แผนภาพคลาส แผนภาพลำดับ และแผนภาพกิจกรรม

2.2.3 งานวิจัยที่เกี่ยวข้องกับแบบจำลองซอฟต์แวร์สำหรับความเป็นส่วนตัว

1) A Conceptual Model for Privacy Policies [9]

งานวิจัยนี้นำเสนอแบบจำลองเชิงแนวคิด (Conceptual Model) สำหรับการนิยาม และการบังคับใช้นโยบายการคุ้มครองความเป็นส่วนตัว โดยนำเสนอในรูปแบบของแบบจำลองซอฟต์แวร์ยูเอ็มแอลคือ แผนภาพคลาส และแผนภาพลำดับ ผู้วิจัยแบ่งโครงสร้างของแบบจำลองตามคุณลักษณะของนโยบายความเป็นส่วนตัว (Privacy Policy) โดยแบ่งออกเป็น 3 คลาสได้แก่ คลาสผู้ใช้ (User) คือคลาสของผู้ที่สนใจการประมวลผลข้อมูล หรือผู้มีส่วนร่วมในการประมวลผล ซึ่งจะแบ่งตามบทบาทที่แตกต่างกันได้แก่ เจ้าของข้อมูล (Subject) ผู้ดำเนินการ (Processor) และ ผู้ควบคุม (Controller), คลาสข้อมูล (Data) คือคลาสของสารสนเทศที่ถูกนำไปประมวลผล ซึ่งจะแบ่งเป็นข้อมูลที่ระบุตัวตน (Identifiable Data) และข้อมูลที่มีความอ่อนไหว (Sensible Data), และคลาสการกระทำ (Action) คือการดำเนินการใด ๆ ที่กระทำโดยผู้ใช้ ซึ่งจะแบ่งเป็น หน้าที่ (Obligation) การประมวลผล (Processing) และวัตถุประสงค์ (Purpose) โดยแบบจำลองเชิงแนวคิดที่ได้สามารถนำมาปรับใช้ได้กับทุกนโยบายการคุ้มครองความเป็นส่วนตัว

ในงานวิจัยนี้นำเสนอเพียงภาพรวมของแบบจำลองเชิงแนวคิดสำหรับการนิยาม และการบังคับใช้นโยบายการคุ้มครองความเป็นส่วนตัว ซึ่งหากจะนำมาใช้เพื่อความเป็นส่วนตัวจะต้องมีการนำมาเพิ่มคลาสบางส่วนเข้าไปในแต่ละการใช้งาน และในงานวิทยานิพนธ์นี้จะนำแนวทางการสร้างแบบจำลองเชิงแนวคิดสำหรับความเป็นส่วนตัว ในส่วนของการแบ่งคุณลักษณะของคลาสต่าง ๆ ออกเป็นคลาสย่อยจากงานวิจัยนี้มาเป็นแนวทางในการสร้างแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

2) Towards a UML Profile for Privacy Aware Application [10]

งานวิจัยนี้นำเสนอยูเอ็มแอลโปรไฟล์ (UML Profile) สำหรับการตระหนักถึงความเป็นส่วนตัวของโปรแกรมประยุกต์ (Applications) ซึ่งโปรไฟล์ที่นำเสนอจะอธิบายถึงนโยบายความเป็นส่วนตัวที่นำมาใช้ในโปรแกรมประยุกต์ เพื่อแสดงให้เห็นถึงองค์ประกอบของนโยบายความเป็นส่วนตัวที่จะนำมาช่วยในการจัดทำเอกสารนโยบายความเป็นส่วนตัวได้ โดยทำการสร้างแบบจำลองเชิงแนวคิดที่มีพื้นฐานบนนโยบายความเป็นส่วนตัวของโออีซีดี (OECD) และมาตรฐานการคุ้มครองความเป็นส่วนตัว (The Global Privacy Standard) และแบบจำลองอ้างอิง (Reference Model) ของมาตรฐานไอเอสโอ 29100 (ISO/IEC 29100) มาตรฐานไอเอสโอ 29101 (ISO/IEC 29101) และมาตรฐานองค์กรพัฒนาโครงสร้างสารสนเทศ (Organization for the Advancement of Structured Information Standards : OASIS) จากนั้นจึงนำแบบจำลองเชิงแนวคิดที่ได้มาแปลงเป็นยูเอ็มแอลโปรไฟล์

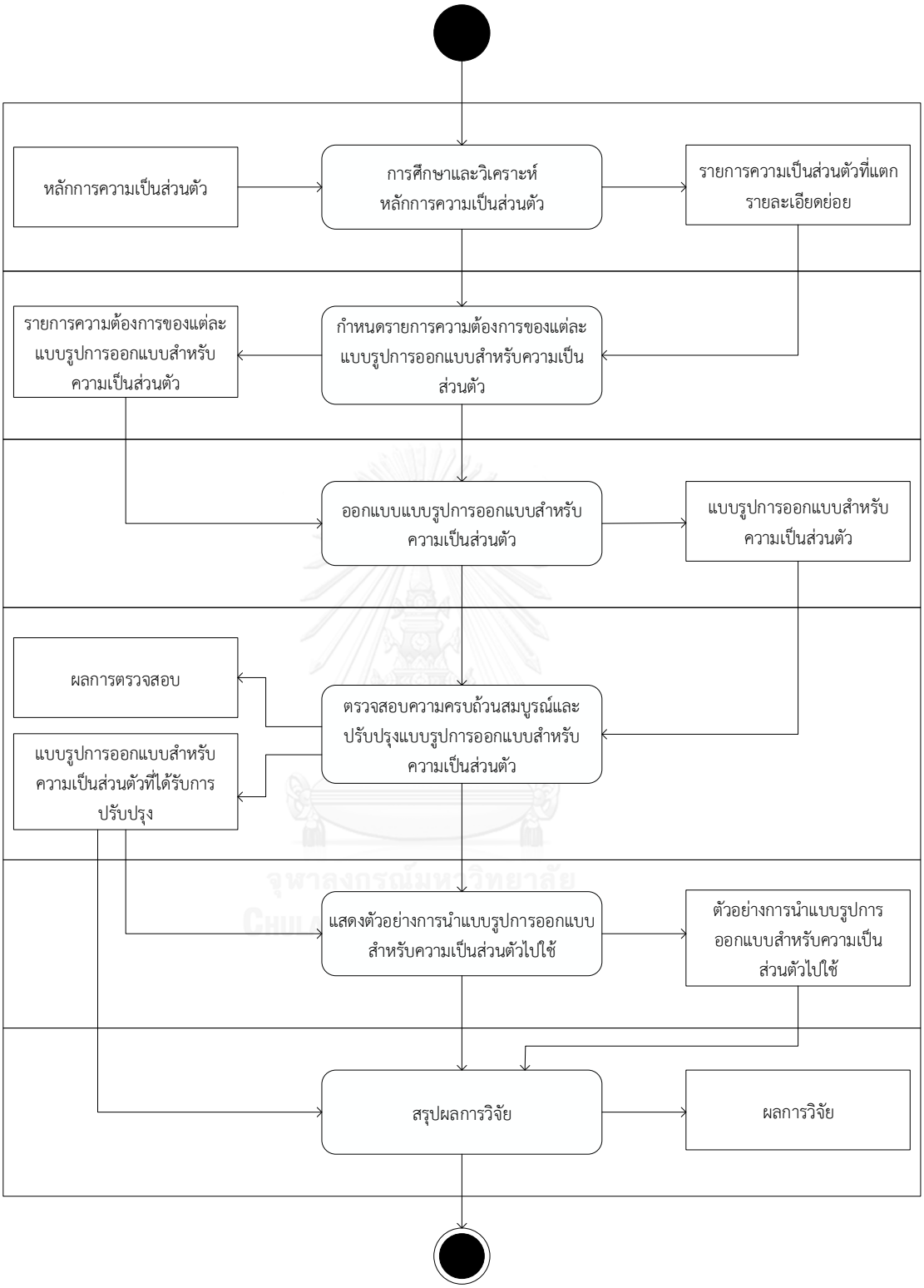
ในงานวิทยานิพนธ์นี้จะนำแนวทางการสร้างแบบจำลองเชิงแนวคิดสำหรับความเป็นส่วนตัวในส่วนของการแบ่งคุณลักษณะของคลาสต่าง ๆ ออกเป็นคลาสย่อยจากงานวิจัยนี้มาเป็นแนวทางในการสร้างแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

บทที่ 3
การพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

ในส่วนของการพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัวตามหลักการความเป็นส่วนตัวโดยใช้ยูเอ็มแอล มีขั้นตอนการดำเนินงานตามแผนภาพกิจกรรมในรูปที่ 3.1 ขั้นตอนที่ 1 เป็นการศึกษาและวิเคราะห์หลักการความเป็นส่วนตัวโดยเริ่มจากการศึกษาหลักการความเป็นส่วนตัว โดยหลักการความเป็นส่วนตัวที่นำมาใช้ในงานวิทยานิพนธ์นี้คือหลักการความเป็นส่วนตัวขององค์การเพื่อความร่วมมือทางเศรษฐกิจและการพัฒนา หรือโออีซีดี ซึ่งได้กล่าวไว้ในบทที่ 2 หัวข้อ 2.1.1 เนื่องจากเป็นหลักการหนึ่งที่ได้รับความนิยมในระดับสากล แล้วจึงนำหลักการที่ได้มาวิเคราะห์ แล้วแตกรายละเอียดย่อยสำหรับแต่ละหลักการความเป็นส่วนตัวในขั้นตอนที่ 2 เพื่อกำหนดรายการความต้องการสำหรับนำมาใช้ในการออกแบบแบบจำลองยูเอ็มแอลสำหรับแบบรูปการออกแบบในขั้นตอนที่ 3 ซึ่งจะแสดงอยู่ในรูปแบบของแผนภาพคลาสเพื่อแสดงโครงสร้างของคลาสและความสัมพันธ์ระหว่างคลาส แผนภาพลำดับเพื่อแสดงลำดับการทำงานและอ็อบเจกต์ที่เกี่ยวข้อง และแผนภาพกิจกรรมเพื่อแสดงลำดับกิจกรรมของการทำงานและทางเลือกที่เกิดขึ้น ขั้นตอนที่ 4 จะทำการตรวจสอบความครบถ้วนของแบบรูปการออกแบบ และทำการปรับปรุงแบบรูปการออกแบบ ขั้นตอนที่ 5 เป็นการนำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอมาแสดงตัวอย่างการนำแบบรูปการออกแบบไปใช้ และขั้นตอนที่ 6 เป็นการสรุปผลการวิจัย

3.1 การศึกษาและวิเคราะห์หลักการความเป็นส่วนตัว

ขั้นตอนนี้จะศึกษาหลักการความเป็นส่วนตัวของโออีซีดี ซึ่งหลักการความเป็นส่วนตัวของโออีซีดีนั้นได้กล่าวถึงในบทที่ 2 หัวข้อ 2.1.1 จากนั้นผู้วิจัยจะทำการแตกรายละเอียดของแต่ละหลักการความเป็นส่วนตัวออกเป็นข้อย่อย ซึ่งสามารถแสดงตัวอย่างการแตกรายละเอียดย่อยของหลักการความเป็นส่วนตัวในหัวข้อหลักข้อจำกัดในการเก็บรวบรวมข้อมูลได้ดังตารางที่ 3.1 สำหรับรายละเอียดย่อยของหลักการความเป็นส่วนตัวทั้งหมดแสดงไว้ในภาคผนวก ข



รูปที่ 3.1 กระบวนการสร้างแบบรูปการออกแบบสำหรับความเป็นส่วนตัวตามหลักการความเป็นส่วนตัว

ตารางที่ 3.1 ตัวอย่างการแตกรายละเอียดย่อยของหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject. (รายละเอียดเพิ่มเติมดูที่ภาคผนวก ก)	1. There should be limits to the collection of personal data, where limits may relate to the following. 1.1 The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly. 1.2 The data collection should be restricted to the minimum necessary to fulfil the specified purpose. 1.3 Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection. 1.4 There should be limits to data collection activities of the data controller such that the data should not be collected by the means that is unknown or deceiving to the data subject. 1.5 Civil rights of the data subject should be of concern in the data collection.

ตารางที่ 3.1 ตัวอย่างการแตกรายละเอียดย่อยของหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject. (รายละเอียดเพิ่มเติมดูที่ภาคผนวก ก)	2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.
		3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.
		4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.

3.2 การกำหนดรายการความต้องการของแต่ละแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

จากรายละเอียดย่อยของหลักการความเป็นส่วนตัว ผู้วิจัยจะนำรายละเอียดย่อยดังกล่าวมาใช้ในการกำหนดรายการความต้องการของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว โดยที่บางรายละเอียดย่อยจะสามารถนำมาสร้างเป็นแบบจำลองยูเอ็มแอล เช่น รายละเอียดย่อยข้อที่ 2 ในตารางที่ 3.1 “The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.” ในขณะที่บางรายละเอียดย่อยซึ่งไม่สามารถนำมาสร้างเป็นแบบจำลองยูเอ็มแอลได้ จะปรากฏอยู่ในส่วนคำอธิบายของแบบรูปการออกแบบแทน เช่น รายละเอียดย่อยข้อที่ 1.3 ในตารางที่ 3.1 “Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection.” ขั้นตอนการกำหนดรายการความต้องการในหัวข้อนี้จะเป็นการกำหนดรายการความต้องการเฉพาะที่สามารถนำมาออกแบบแบบจำลองยูเอ็มแอลได้ และในหนึ่งแบบรูปการออกแบบสำหรับความเป็นส่วนตัวนั้นจะตอบสนองหลักการความเป็นส่วนตัวหนึ่งหลักการ ตัวอย่างรายการความต้องการของแบบรูปการออกแบบสำหรับความเป็นส่วนตัวในหัวข้อหลัก

ข้อจำกัดในการเก็บรวบรวมข้อมูลแสดงได้ดังตารางที่ 3.2 สำหรับรายการความต้องการเพื่อใช้ในการออกแบบแบบจำลองยูเอ็มแอลสำหรับแบบรูปการออกแบบทั้งหมดแสดงไว้ในภาคผนวก ค

ตารางที่ 3.2 ตัวอย่างรายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

Pattern Name	**Collection Limitation Pattern**
Privacy Principle	Collection Limitation Principle
Pattern Requirement	
ID	**Requirement**
PT-CL01	The system should be able to collect personal data that are necessary for use.
PT-CL02	The system should be able to record the information about the source of personal data collection.
PT-CL03	The system should be able to record the information about how long personal data will be retained.
PT-CL04	The system should let the data subject know about personal data collection.
PT-CL05	The system should allow the data subject to give or refuse consent to data collection.
PT-CL06	The system should be able to record the information about the party who gives consent to data collection on behalf of the data subject.

รายการความต้องการเหล่านี้สัมพันธ์กับบางรายละเอียดย่อยของหลักการความเป็นส่วนตัว และเมื่อนำรายการความต้องการเหล่านี้ไปออกแบบแบบจำลองยูเอ็มแอลแล้ว จะทำให้แบบจำลองสามารถสนับสนุนหลักการความเป็นส่วนตัวตามรายละเอียดย่อยนั้นได้ ความสัมพันธ์ระหว่างรายการความต้องการและรายละเอียดย่อยของแต่ละหลักการความเป็นส่วนตัวแสดงไว้ในตารางที่ 3.3

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ

รายละเอียดย่อย		**รายการความต้องการของแบบรูปการออกแบบ**	
		รหัส	**รายการความต้องการ**
Collection Limitation Principle			
1. There should be limits to the collection of personal data.			
1.1	The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly.		ไม่มี
1.2	The data collection should be restricted to the minimum necessary to fulfil the specified purpose.	PT-CL01	The system should be able to collect personal data that are necessary for use.
		PT-CL02	The system should be able to record the information about how long personal data will be retained.
1.3	Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection.		ไม่มี
1.4	There should be limits to data collection activities of the data controller such that the data should not be collected by the means that is unknown or deceiving to the data subject.	PT-CL03	The system should be able to record the information about the source of personal data collection.
1.5	Civil rights of the data subject should be of concern in the data collection.		ไม่มี
2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.		PT-CL04	The system should let the data subject know about personal data collection.

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Collection Limitation Principle		
3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.	PT-CL05	The system should allow the data subject to give or refuse consent to data collection.
4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.	PT-CL06	The system should be able to record the information about the party who gives consent to data collection on behalf of the data subject.
Data Quality Principle		
1. Personal data should be relevant to the purposes for which they are to be used.		ไม่มี
2. Personal data, to the extent necessary for their purposes, should be accurate.	PT-DQ02	The system should be able to validate the collected personal data for accuracy.
	PT-DQ03	The system should be able to check integrity of the collected personal data for accuracy.
3. Personal data, to the extent necessary for their purposes, should be complete.	PT-DQ01	The system should be able to check, for data completeness, that all required personal data are collected.
4. Personal data, to the extent necessary for their purposes, should be kept up-to-date.	PT-DQ04	The system should be able to collect current personal data for data up-to-dateness.
Purpose Specification Principle		
1. The purposes for which personal data are collected should be specified not later than at the time of data collection.	PT-PS01	The system should be able to inform the information about the purpose for which the collected data are to be used.

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
2. The subsequent use of personal data should be limited to the fulfilment of the purposes specified prior to or at the time of collection.		ไม่มี
3. The subsequent use of personal data may be for purposes other than the original purposes, but such purposes should be specified as change.		ไม่มี
4. Later changes of purposes should be specified.		ไม่มี
5. On each occasion of change of purpose, the new purposes should not be incompatible with the original purposes.		ไม่มี
6. Specification of purpose should be made by public declarations or information to the data subject.		ไม่มี
7. When data no longer serve a purpose, and if it is practicable, it may be necessary to have them destroyed (erased) or given an anonymous form.	PT-PS02	The system should be able to check the retention period or expiration date of personal data.
	PT-PS03	The system should be able to delete or anonymize personal data after their retention has expired.
8. The Purpose Specification Principle is closely associated with the Data Quality Principle and the Use Limitation Principle.		ไม่มี

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Use Limitation Principle		
1. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except with the consent of the data subject or his representative.	PT-UL01	The system should let the data subject know about the use of personal data which deviates from the specified purposes.
2. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except by the authority of law or licences granted by supervisory bodies.	PT-UL02	The system should allow the data subject to give or refuse consent to the use of personal data which deviates from the specified purposes.
Security Safeguards Principle		
1. Personal data should be protected by reasonable security safeguards against unauthorized disclosure of data, based on informational measures such as enciphering.	PT-SS01	The system should be able to encrypt the personal data to ensure confidentiality.
2. Personal data should be protected by reasonable security safeguards against unauthorized access, destruction, use, and modification of data, based on organizational measures such as authority levels with regard to access to data.	PT-SS02	The system should be able to authenticate users of the system before accessing personal data.
	PT-SS03	The system should be able to define access rights to create, destroy, view, and modify personal data based on authority levels or roles within the organization.

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Security Safeguards Principle		
2. Personal data should be protected by reasonable security safeguards against unauthorized access, destruction, use, and modification of data, based on organizational measures such as authority levels with regard to access to data.	PT-SS04	The system should be able to control access to the personal data by different access types including create, destroy, view, and modify, based on authority levels or roles within the organization.
3. Personal data should be protected by reasonable security safeguards against loss or destruction and modification of data, both accidentally and deliberately, where data integrity would be affected.	PT-SS05	The system should be able to provide a mechanism to ensure data integrity.
4. Personal data should be protected by reasonable security safeguards against loss of access by which data availability would be affected.	PT-SS06	The system should be able to provide a mechanism to ensure data availability.
5. Personal data should be protected by reasonable security safeguards against threats of data breach, based on informational measures such as monitoring of unusual activities and responses to them.	PT-SS07	The system should be able to monitor and record access to personal data.
	PT-SS08	The system should be able to check for suspicious data breach and notify the data controller and data subject.
6. Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees' security training, up-to-date security safeguards).		ไม่มี

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Openness Principle		
1. There should be a general policy of openness about developments, practices and policies with respect to personal data.	PT-OP01	The system should be able to record the information about the privacy policy, including the existence and nature of personal data, main purpose of collection and use, and identity and usual residence of the data controller.
2. Means should be readily available of establishing the existence and nature of personal data.	PT-OP02	The system should let the data subject know about the privacy policy.
	PT-OP03	The system should let the data subject know about change of the privacy policy.
	PT-OP04	The system should provide the data subject with access to the privacy policy.
3. Means should be readily available of establishing the main purposes of the use of personal data.	PT-OP02	The system should let the data subject know about the privacy policy.
	PT-OP03	The system should let the data subject know about change of the privacy policy.
	PT-OP04	The system should provide the data subject with access to the privacy policy.
4. Means should be readily available of establishing the identity and usual residence of the data controller.	PT-OP02	The system should let the data subject know about the privacy policy.

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Openness Principle		
4. Means should be readily available of establishing the identity and usual residence of the data controller.	PT-OP03	The system should let the data subject know about change of the privacy policy.
	PT-OP04	The system should provide the data subject with access to the privacy policy.
5. “Readily available” implies that data subjects should be able to obtain information without unreasonable effort as to time, advance knowledge, travelling, and so forth, and without unreasonable cost.	PT-OP02	The system should let the data subject know about the privacy policy.
	PT-OP03	The system should let the data subject know about change of the privacy policy.
	PT-OP04	The system should provide the data subject with access to the privacy policy.
6. Regular information from the data controller regarding the privacy policy should be published.	PT-OP01	The system should be able to record the information about the privacy policy, including the existence and nature of personal data, main purpose of collection and use, and identity and usual residence of the data controller.
7. The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle.		ไม่มี

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Individual Participation Principle		
1. A data subject can obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him/her (or his/her personal data).		ไม่มี
2. A data subject should have the right to access to his/her personal data within a reasonable time when a request to access is made.	PT-IP02	The system should allow the data subject to access his/her personal data.
3. A data subject should have the right to access to his/her personal data at a reasonable cost.	PT-IP02	The system should allow the data subject to access his/her personal data.
4. A data subject should have the right to access to his/her personal data in a reasonable manner.	PT-IP02	The system should allow the data subject to access his/her personal data.
5. A data subject should have the right to access to his/her personal data in a form that is readily intelligible.	PT-IP02	The system should allow the data subject to access his/her personal data.
6. A data subject should have the right to reasons for, and challenge, adverse decisions when requests for confirmation or access are denied.	PT-IP01	The system should allow inquiry of and response to issues related to personal data.
7. A data subject should have the right to challenge his/her personal data.	PT-IP01	The system should allow inquiry of and response to issues related to personal data.
8. A data controller should erase, rectify, complete, or amend the data subject's personal data if the challenge is successful.	PT-IP03	The system should allow the data controller to erase and update the data subject's personal data.

ตารางที่ 3.3 ความสัมพันธ์ระหว่างรายละเอียดย่อยของหลักการความเป็นส่วนตัว และรายการความต้องการของแบบรูปการออกแบบ (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**	
	รหัส	**รายการความต้องการ**
Accountability Principle		
1. A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles.		ไม่มี
2. A data controller should have in place a privacy management program that is tailored to its operations and integrated into its governance structure.		ไม่มี
3. A data controller should provide ongoing monitoring, periodic assessment of data breach, and response to incidents.	PT-AC02	The system should be able to perform periodic assessment of data breach.
	PT-AC03	The system should be able to handle data breach.
	PT-AC05	The system should be able to report on data breach checking.
4. There should be a method for the data subject to inquire about issues related to privacy of personal data.	PT-AC04	The system should allow inquiry of and response to issues related to personal data.
5. A data controller should provide evidence of its privacy management program to the privacy enforcement authorities.	PT-AC01	The system should be able to record events that occur in the system.
6. A data controller should provide notice to the privacy enforcement authorities and data subject in the event of significant data breach.	PT-AC06	The system should be able to notify relevant parties, i.e. privacy enforcement authorities and data subject, in the event of significant data breach.

3.3 การออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

ในขั้นตอนนี้จะทำการออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัว โดยนำรายละเอียดจากหลักการความเป็นส่วนตัวของโออีซีดีมาใช้กำหนดคำอธิบายภายในแบบรูป และนำรายการความต้องการของแต่ละแบบรูปจากหัวข้อที่ 3.2 มาทำการออกแบบในส่วนแบบจำลองของซอฟต์แวร์เพื่อตอบสนองหลักการความเป็นส่วนตัว โดยที่องค์ประกอบของแบบรูปจะนำมาจาก [4] ตามที่กล่าวไว้ในหัวข้อที่ 2.1.3 แต่มีการปรับเปลี่ยนดังนี้

1) ในส่วนของโครงสร้าง (Structure) ของแบบรูป ซึ่งอธิบายโดยใช้เฉพาะแผนภาพคลาส ได้มีการปรับให้เป็นส่วนของพฤติกรรมและโครงสร้าง (Behavior and Structure) โดยจะแสดงแผนภาพกิจกรรมซึ่งแสดงพฤติกรรมของแบบรูปเป็นข้อมูลเพิ่มเติมด้วย นอกเหนือไปจากโครงสร้างของแบบรูปตามแผนภาพคลาส
2) ในส่วนของการร่วมมือกัน (Collaboration) จะแสดงแผนภาพลำดับเพื่อแสดงการทำงานร่วมกันของคลาสต่าง ๆ จากแผนภาพคลาสด้วย แทนการใช้เฉพาะคำบรรยาย
3) ในส่วนของตัวอย่างโค้ด (Sample Code) ได้มีการปรับให้เป็นส่วนของตัวอย่างการออกแบบและโค้ด (Sample Design and Code) โดยแสดงตัวอย่างแผนภาพคลาสซึ่งประยุกต์ใช้แบบรูปการออกแบบ และตัวอย่างโค้ดซึ่งพัฒนาตามตัวอย่างแผนภาพคลาส แทนการแสดงเฉพาะโค้ดในส่วนตัวอย่างโค้ด (Sample Code) และแสดงตัวอย่างการออกแบบไว้ในส่วนแรงจูงใจ (Motivation) แยกจากกัน

ตัวอย่างแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูลแสดงได้ดังตารางที่ 3.4 สำหรับแบบรูปการออกแบบทั้งหมดแสดงไว้ในภาคผนวก ง

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

Pattern Name	**Collection Limitation Pattern**
Privacy Principle	Collection Limitation Principle
Intent	This pattern assures that the personal data that the system of the data controller will collect should be limited and are collected with the knowledge or consent of the data subject. As personal data are attributes of the data subject, it is necessary to assure that he/she is still in control of the personal data and the collection does not violate his/her civil right as the owner of the collected data.

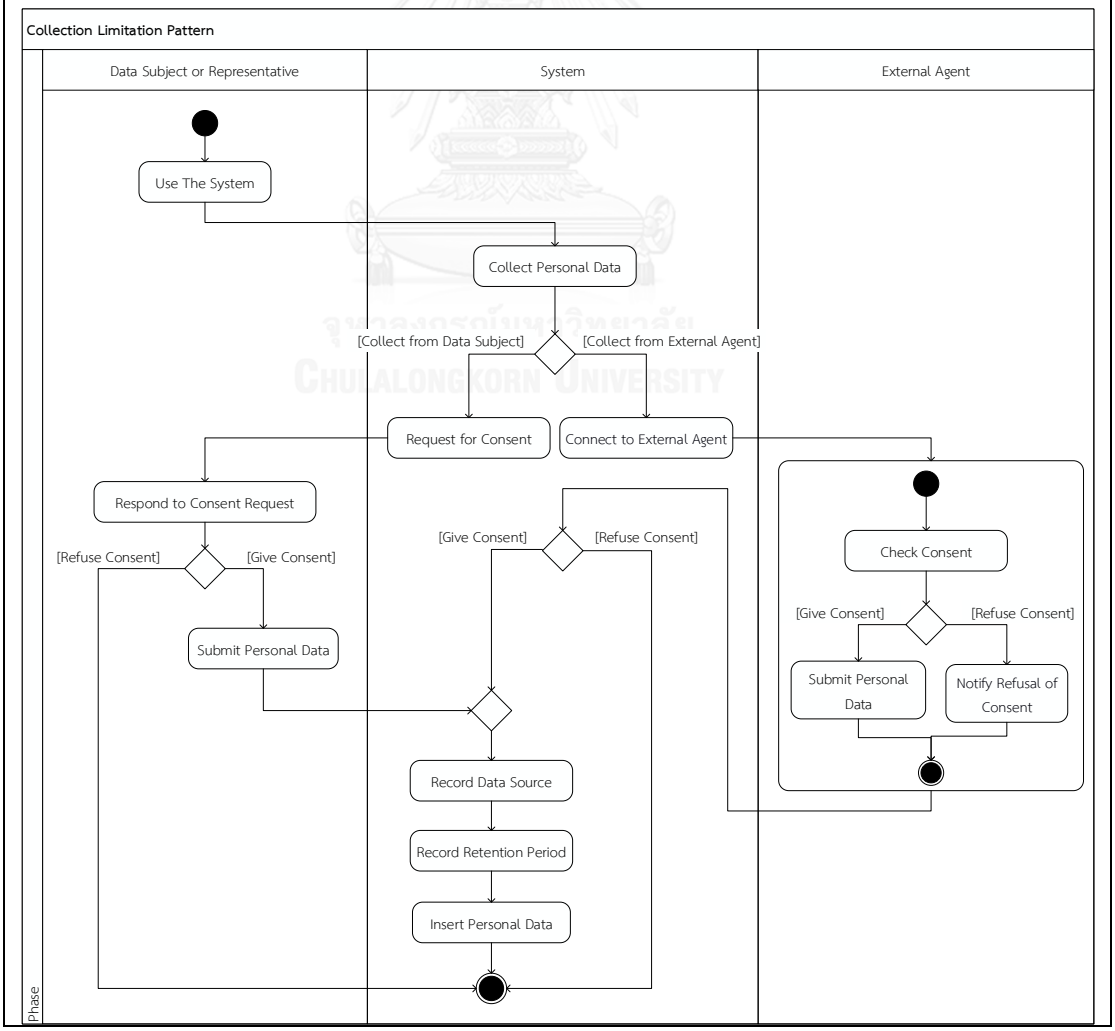
ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Motivation	When a data subject (or user) is about to use a service of a data controller on the Internet, the service may need to collect certain personal data of the data subject for the processing of the service. For example, a cosmetic online shopping web site needs to collect a customer's personal data at registration time. Among the collected personal data (such as name, email, and phone number), the web site also collects national ID and does not ask the customer to give consent to the collection. In this scenario, sensitive information (i.e. national ID) that is not used in the processing of the online shop is unnecessarily disclosed. If the online shop neither asks the customer to give consent to the collection nor handles data privacy well, the customer's sensitive information may leak and be misused in a way that causes harm to the customer. Thus, the data subject should be aware of what, when, and how personal data are collected. The system of the data controller should collect only the personal data that are necessary for use, and let the data subject know about personal data collection and give consent. Sometimes it may be the case that consent is given by a representative on behalf of the data subject.
Applicability	Use the Collection Limitation pattern whenever the system is about to collect personal data, i.e. any information relating to an identified or identifiable individual (data subject) [1]. According to [11], the system should use the Collection Limitation pattern when collecting any of the following categories of personal data: • Preference data i.e. Preferences on books, music, cities, friends, wish list, and so forth. • Communication record i.e. SMS text, phone records, address book, and so forth. • Web footprint i.e. Visited websites, search keywords, social comment logs and social graph, and so forth • Personal profile i.e. Height, weight, published papers, education/career experiences, exam performance, and so forth. • Consumption service record i.e. Bank account and transaction records, flight and hotel orders, car rental orders, supermarket records, e-commerce transaction record, and so forth.

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

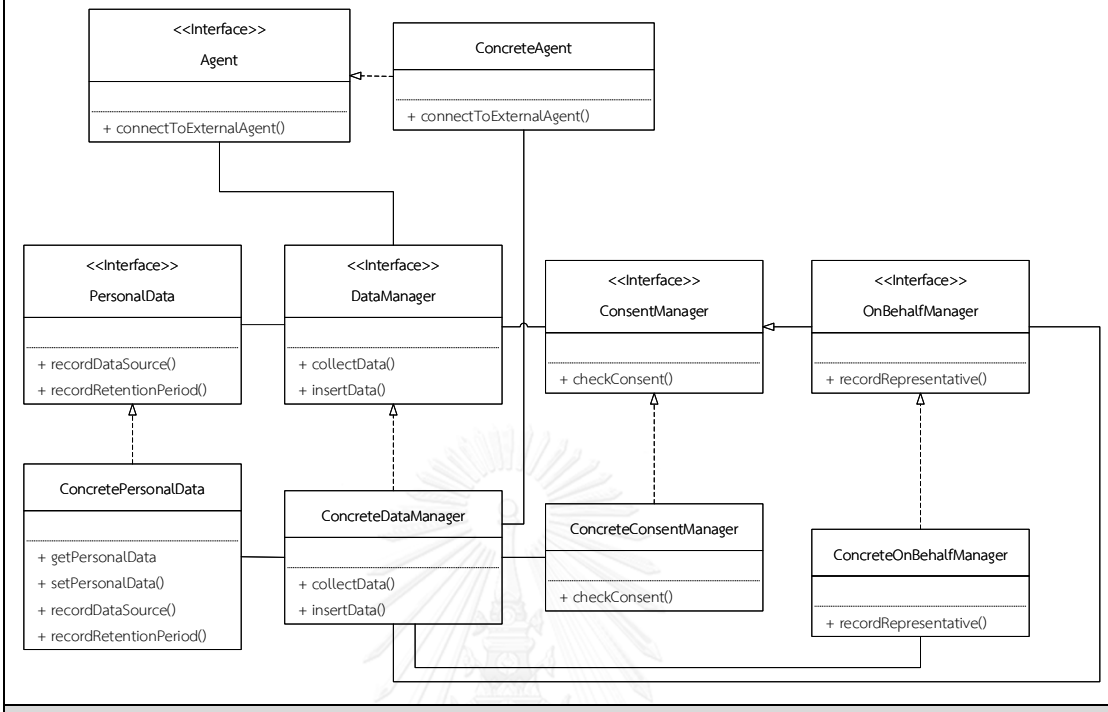
<table>
<tr><td>Applicability</td><td>
Public service record Personal salary records, household energy record, personal credit, and so forth.
Also, the system should use the Collection Limitation pattern no matter where such personal data are collected from [11]:

Personal device i.e. PC, smart phones, mobile devices, tablet, and so forth.
Services i.e. Web applications, and so forth.
Social network i.e. Facebook, Twitter, blogs, and so forth.
Sensors i.e. GPS, thermometer, wearable devices, and so forth.

The data subject itself i.e. Email, work schedule, documents, pictures, video, audio.
</td></tr>
<tr><td colspan="2">Behavior and Structure</td></tr>
<tr><td colspan="2">Activity Diagram
 
</td></tr>
</table>

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Class Diagram



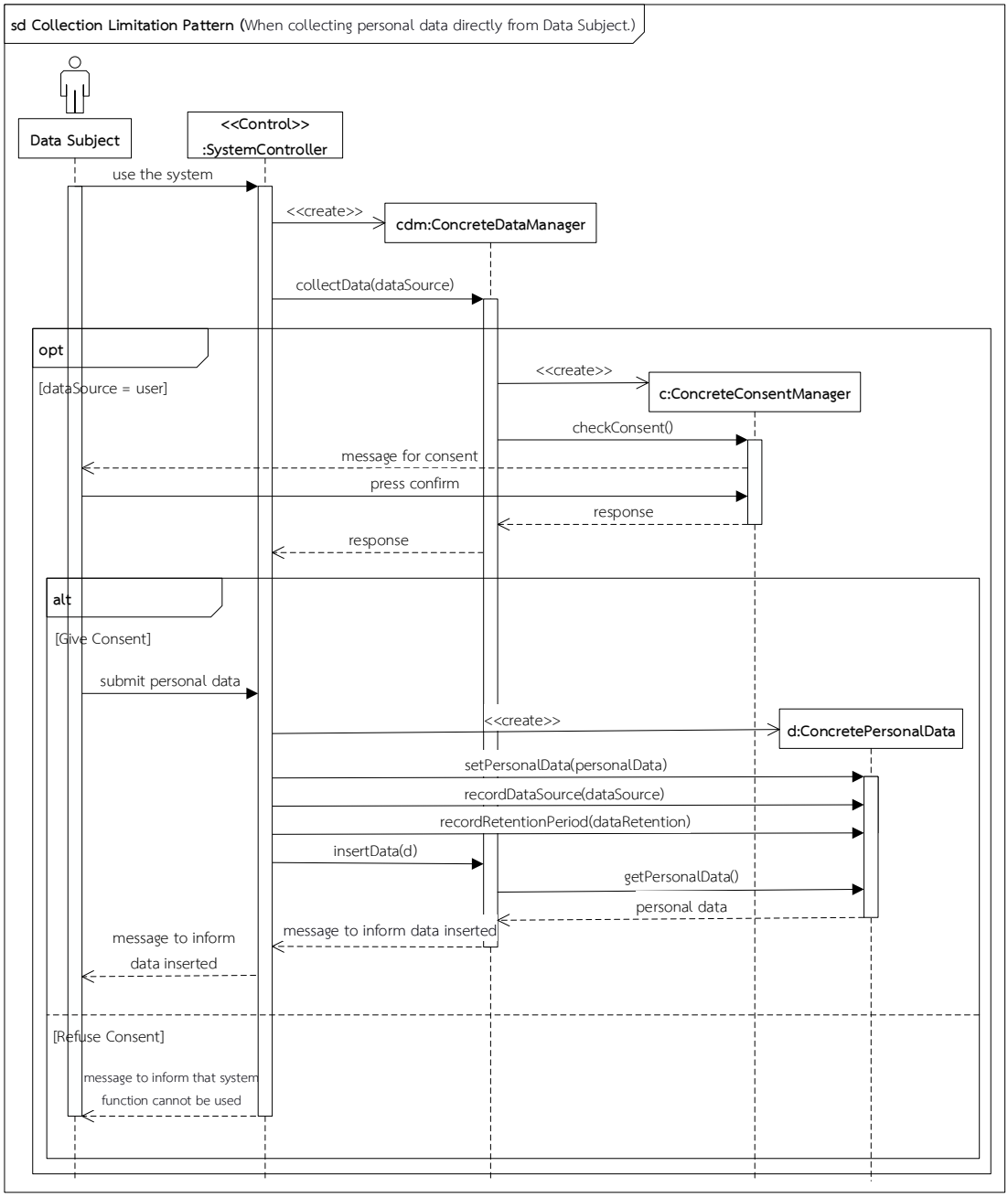
Participants

Interface Name/Class Name	Detail
Interface DataManager	An interface that has the responsibilities to coordinate checking of consent and collection of personal data from the data subject. It inserts the collected data into the system if the data subject gives consent.
Class ConcreteDataManager	A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. coordinating update of personal data when there are changes etc.
Interface ConsentManager	An interface that has the responsibilities to inform the data subject that the system needs to collect personal data and check whether the data subject gives or refuses consent.
Class ConcreteConsentManager	A concrete class that implements the interface ConsentManager.
Interface OnBehalfManager	An interface that inherits from the interface ConsentManager. It has the responsibility to record the information about who, on behalf of the data subject, has given the consent to collect the personal data.

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

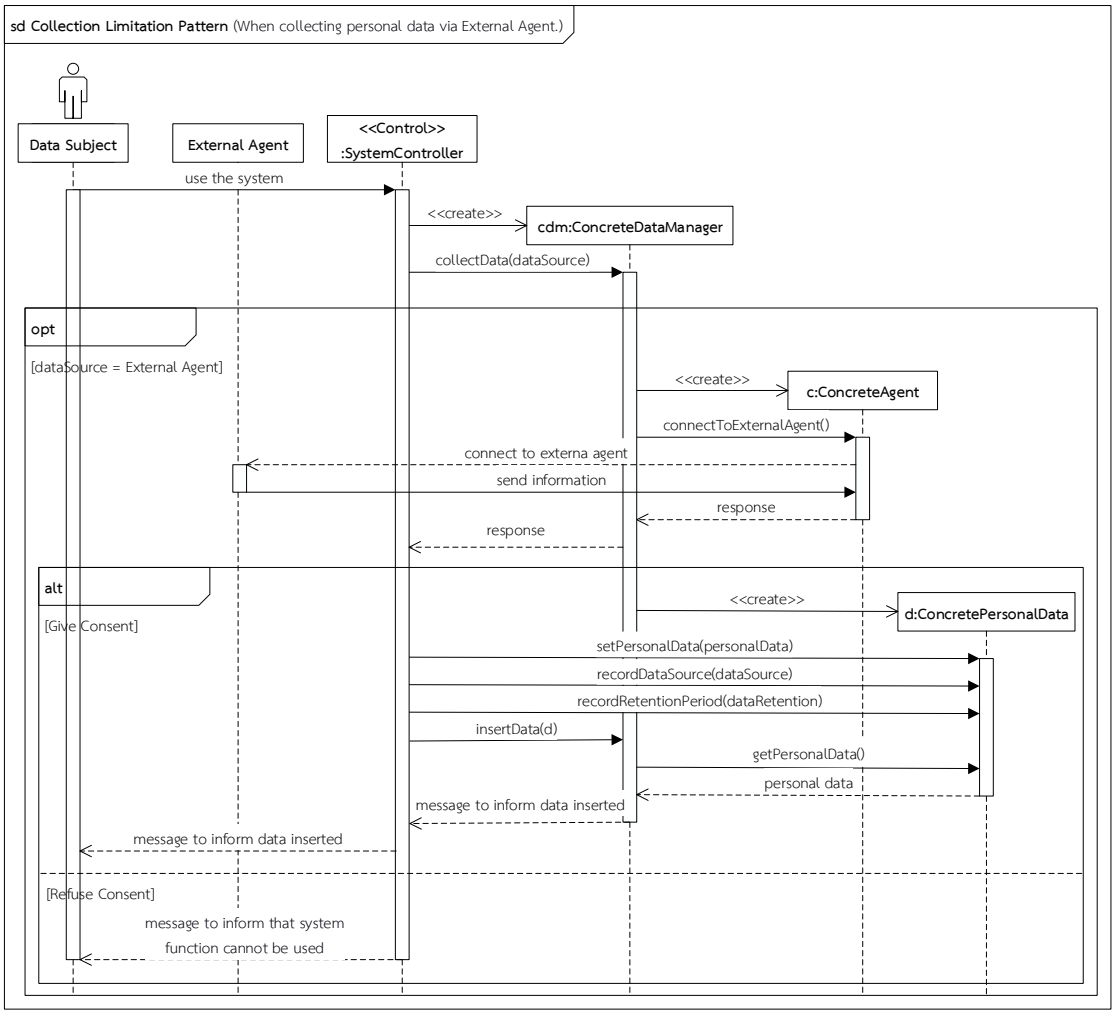
Participants	
Interface Name/Class Name	**Detail**
Class ConcreteOnBehalfManager	A concrete class that implements the interface OnBehalfManager.
Interface Agent	An interface that has the responsibilities to connect to an external agent to collect personal data of the data subject on behalf of the system, and obtain personal data via the external agent if consent is given.
Class ConcreteAgent	A concrete class that implements the interface Agent.
Interface PersonalData	An interface that represents the collected personal data.
Class ConcretePersonalData	A concrete class with the data attributes that are necessary for the purpose of the system.
Collaboration	
1. When the data subject requests to use a system function, the system controller creates a ConcreteDataManager object to collect personal data. The data source where personal data will be collected (i.e. data subject or external agent) and data retention detail (e.g. retention time, format) should be determined by the data controller. 2. If personal data will be collected directly from the data subject, 2.1. The ConcreteDataManager object creates a ConcreteConsentManager object to check whether the data subject gives or refuse consent to personal data collection. 2.2. The ConcreteDataManager object notifies the system controller of the consent checking result. 2.3. If consent is given, the system controller creates a ConcretePersonalData object and calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store. 3. If personal data will be collected via an external agent, 3.1. The ConcreteDataManager object creates a ConcreteAgent object to connect to an external agent. 3.2. If the external agent is given the consent to collect personal data, the ConcreteAgent object creates a ConcretePersonalData object and calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store.	

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Collaboration
4. In the case that consent is given by a representative on behalf of the data subject and the system needs to record such a fact, the ConcreteDataManager can create an OnBehalfManager object, instead of a ConcreteConsentManager object, to check consent and also record the representative information.
Sequence Diagram (When collecting personal data directly from Data Subject.) 

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data via External Agent.)



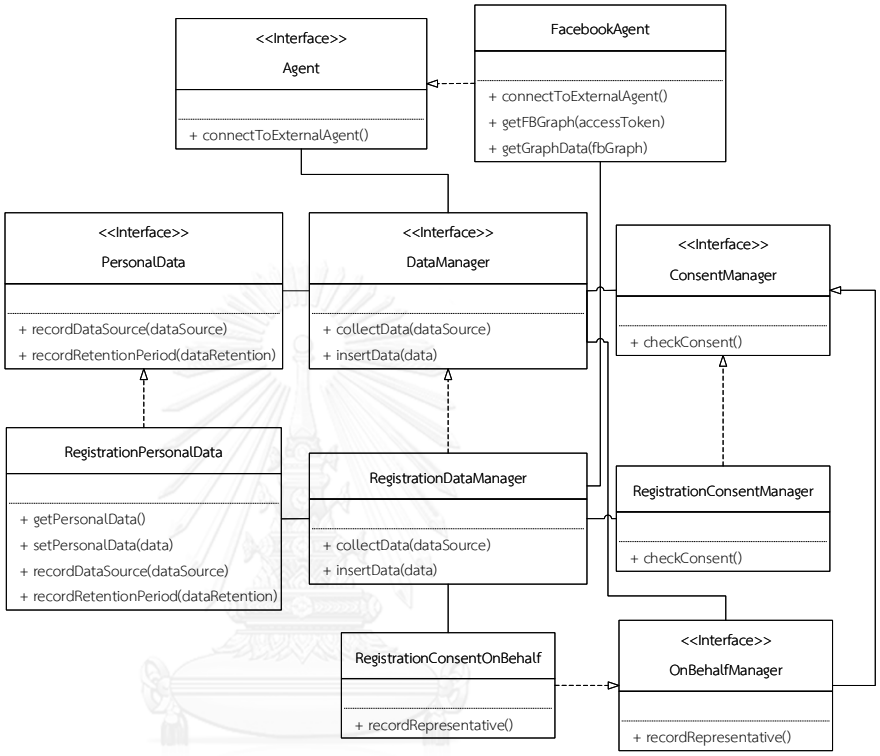
ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Consequences	1. Using this pattern, the system of the data controller can handle collection of personal data as recommended by OECD. The system can handle important tasks that are addressed by the Collection Limitation principle. That is, the personal data management, consent and on-behalf consent management, and connection to external agents tasks are specifically accommodated by corresponding interfaces (and their concrete classes). Thus, cohesion of each individual class in the design pattern structure can be enhanced. 2. The data subject can be assured that the personal data that will be used by the system are collected with the knowledge or consent of his/hers. The data subject can be informed of what data are about to be collected and can decide whether such data are necessary for the processing of the system. The data subject has the right to reject the collection and, as a result, the service, if he/she does not want to disclose such requested personal information.
Implementation	Issues regarding the implementation of the Collection Limitation pattern are as follows. 1. The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly. Such a limit imposed on personal data collection entails a data quality requirement and the data controller who implements the system should refer to the Data Quality pattern also. 2. Since data collection should be restricted to the minimum necessary to fulfil the specified purpose [1], the data controller who implements the system should decide on which personal data are really needed. Therefore, the class ConcretePersonalData should be implemented to have only the data attributes that are necessary for the purpose of the system. 3. Even though OECD defines that there should be limits to the collection of personal data, the detail about which data are prohibited from collection are left out because it depends on the traditions and attitudes in each country where the system operates. For example [1], personal data such as race, religious beliefs, criminal records, health, sex life, and so forth are sensitive and the collection of which should be restricted or even prohibited by European legislation. On the other hand, for the United States privacy

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

	legislation, no data are specified as being more private or sensitive than others as sensitivity may depend on the context and use of the data. Therefore, the data controller who implements the system should decide on which personal data attributes should be included in the class ConcretePersonalData while not violating the privacy legislation and traditions that apply to the system. 4. OECD defines that there should be limits to data collection activities of the data controller. For example [1], hidden devices should not be used and the data subject should not be deceived to provide personal information. However, the control of such limits cannot be captured completely by the structural model of this design pattern. To help address this issue, the data controller should record the source of the data collection, i.e. whether the data are collected from the data subject directly or an external agent. The class ConcretePersonalData should be implemented to record the source of the collected personal data. This could help to deal with data collection misconduct if such a case arises. In addition, the data controller should be open with the data subject about what data are collected and for what purposes. The data controller should refer to the Openness pattern also. 5. OECD defines that there should be limits to the collection of personal data in such a way that the civil rights of the data subject should be of concern and the data collection should not violate civil rights of the data subject. Since the data subject should be entitled to his/her civil rights including the right to his/her collected personal data, the data controller should also refer to the Individual Participation pattern. 6. It is possible that the data subject may be represented by another party who gives consent to the collection of the data subject's personal data, e.g. in the case of minors, mentally disabled person etc. If the data controller needs to record this fact, e.g. for legal purpose, the interface OnBehalfManager that inherits from the interface ConsentManager can be implemented (i.e. class ConcreteOnBehalfManager) to also record the information about who, on behalf of the data subject, has given the consent to collect the personal data.

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	An example of a user registration system that applies the Collection Limitation pattern is shown below. The system collects personal data of any user who requests to use the system and the collection can be done either by the system or via Facebook.  When a user (data subject) is required to register with the system to use its service, the system uses the class RegistrationDataManager that implements the interface DataManager to collect personal data (name, email, tel) as shown in the sample code below. If the data source of collection is “User”, the data are collected directly from the user. Otherwise they are collected via Facebook. `interface DataManager{` `  public Map<String, String> collectData(String dataSource);` `  public void insertData(RegistrationPersonalData pd);` `}`

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	

```
public class RegistrationDataManager implements DataManager{

    @Override
    public Map<String, String> collectData(String dataSource) {
        Map<String,String> data = new HashMap<String, String>();
        if(dataSource.equalsIgnoreCase("User")){
            RegistrationConsentManager rcm = new RegistrationConsentManager();
            if(rcm.checkConsent()){
                data.put("consent", "accept");
            }else{
                data.put("consent", "refuse");
            }
        }else{
            FacebookAgent fa = new  FacebookAgent();
            data.putAll(fa.connectToExternalAgent());
        }
        return data;
    }
    @Override
    public void insertData(RegistrationPersonalData data){
        //TODO Add your code to insert personal data into data store.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String insert = "INSERT INTO personaldata "
                    + "(name, email, tel, dataSource, dataRetention) "
                    + "VALUES (?, ?, ?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert) ;
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, data.getName());
            stmt.setString(parameterIndex++, data.getEmail());
            stmt.setString(parameterIndex++, data.getTel());
            stmt.setString(parameterIndex++, data.getDataSource());
            stmt.setString(parameterIndex++, data.getDataRetention());

            stmt.execute();
            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
    }
}
```

If personal data are collected directly from the user, the class RegistrationConsentManager that implements the interface ConsentManager is used to check whether the user gives or refuse consent to personal data collection as shown in the sample code below.

```
public interface ConsentManager {
    public boolean checkConsent();
}
public class RegistrationConsentManager implements ConsentManager{
    private Scanner scan;

    @Override
    public boolean checkConsent() {
        scan = new Scanner(System.in);
        System.out.println("Give Consent Please Enter 'Y' "
                + "and Refuse Consent Please Enter 'N'");
        String accept = scan.next();
        if(accept.equalsIgnoreCase("Y"))
            return true;
        else
            return false;
    }
}
```

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	The class RegistrationPersonalData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of the system as shown in the sample code below. The system requires only the attributes name, email, and telephone number to use its service. In addition, it records the source of collection of these data as well as the end of the retention period which varies by different types of data that the system collects during the use of its service, i.e. registration data, location data, and others.

```
public interface PersonalData {
    public String recordDatasource(String dataSource);
    public String recordRetentionPeriod(String dataRetention);
}
public class RegistrationPersonalData implements PersonalData{
    private String name;
    private String email;
    private String tel;
    private String dataSource;
    private String dataRetention;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getTel() {
        return tel;
    }
    public void setTel(String tel) {
        this.tel = tel;
    }
    public String getDataSource() {
        return dataSource;
    }
    public String getDataRetention() {
        return dataRetention;
    }
    @Override
    public String recordDatasource(String dSource) {
        dataSource = dSource;
        return dataSource;
    }
    @Override
    public String recordRetentionPeriod(String dRetention) {
        if(dRetention.equalsIgnoreCase("Registration"))
            dataRetention = "When deleting account";
        else if(dRetention.equalsIgnoreCase("Location"))
            dataRetention = "When ending program";
        else
            dataRetention = "When deleting data";
        return dataRetention;
    }
}
```

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	In the case that the user registers via Facebook, the class FacebookAgent that implements the interface Agent is used to collect personal data as shown in the sample code below.

```
public interface Agent {
    public Map<String, String> connectToExternalAgent();
}
public class FacebookAgent implements Agent{

    @Override
    public Map<String, String> connectToExternalAgent() {
        String accessToken = "";
        /* TODO Add your code to connect to external agent,
         * i.e. Facebook, to obtain an access token here */
        String graph = getFBGraph(accessToken);
        Map<String, String> fbProfileData = getGraphData(graph);
        return fbProfileData;
    }

    public String getFBGraph(String accessToken) {
        String graph = null;
        try {

            String g = "https://graph.facebook.com/me?" + accessToken;
            URL u = new URL(g);
            URLConnection c = u.openConnection();
            BufferedReader in = new BufferedReader(new InputStreamReader(
                    c.getInputStream()));
            String inputLine;
            StringBuffer b = new StringBuffer();
            while ((inputLine = in.readLine()) != null)
                b.append(inputLine + "\n");
            in.close();
            graph = b.toString();
            System.out.println(graph);
        } catch (Exception e) {
            e.printStackTrace();
            throw new RuntimeException("ERROR in getting FB graph data. " + e);
        }
        return graph;
    }
    public Map<String, String> getGraphData(String fbGraph) {
        Map<String, String> fbProfile = new HashMap<String, String>();
        try {
            JSONObject json = new JSONObject(fbGraph);
            fbProfile.put("id", json.getString("id"));
            fbProfile.put("name", json.getString("name"));
            if (json.has("email"))
                fbProfile.put("email", json.getString("email"));
            if (json.has("gender"))
                fbProfile.put("gender", json.getString("gender"));
        } catch (JSONException e) {
            e.printStackTrace();
            throw new RuntimeException("ERROR in parsing FB graph data. " + e);
        }
        return fbProfile;
    }
}
```

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

<table>
<tr><td>Sample Design and Code</td><td>In the case that that the system needs to record, e.g. for legal purpose, who gives consent on behalf of the user, the class RegistrationConsentOnBehalf that implements the interface OnBehalfManager as shown in the sample code below would be called by the class RegistrationDataManager instead of the class RegistrationConsentManager.

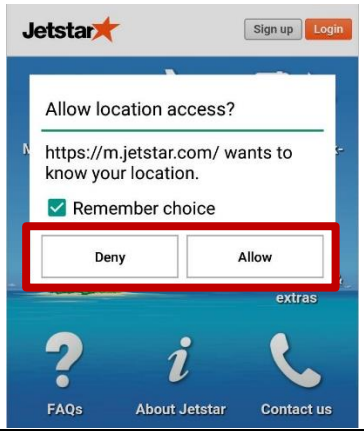
<pre>
public interface OnBehalfManager extends ConsentManager {
    public String recordRepresentative();
}
public class RegistrationConsentOnBehalf implements OnBehalfManager{

    private Scanner scan;

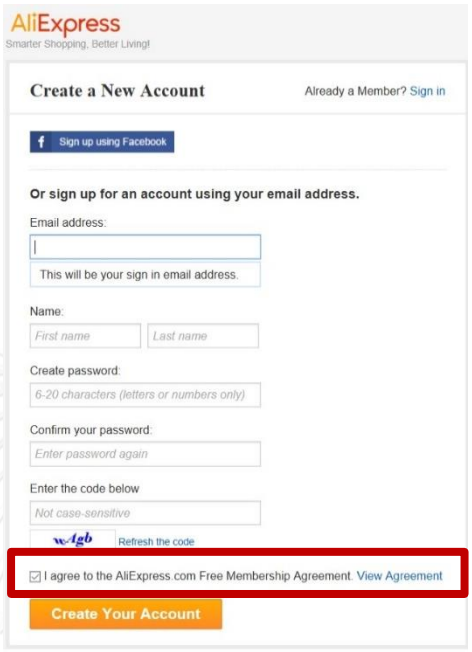
    @Override
    public boolean checkConsent() {
        scan = new Scanner(System.in);
        System.out.println("Give Consent Please Enter 'Y' "
                + "and Refuse Consent Please Enter 'N'");
        String accept = scan.next();
        if(accept.equalsIgnoreCase("Y"))
            return true;
        else
            return false;
    }

    @Override
    public String recordRepresentative() {
        scan = new Scanner(System.in);
        System.out.println("Enter name of representative");
        String representative = scan.next();
        return representative;
</pre></td></tr>
<tr><td>Known use</td><td>Examples of existing systems that take the Collection Limitation principle into account include:

- JetStar (Website): While a user is using a web service on a mobile phone, the system requires further access to the user location and will ask for user consent. The user can then give or refuse consent to the collection of his/her location data.


</td></tr>
</table>

ตารางที่ 3.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Known use	● AliExpress (Website): When a user creates an account, the system will collect personal data. If the user gives consent to the system to collect and use personal data, the user should check the agreement checkbox. 
Related Patterns	Data Quality The Data Quality pattern can be used with the Collection Limitation pattern to ensure that the personal data that are collected are accurate, complete, and up-to-date for the processing of the system. Openness The Openness pattern can be used with the Collection Limitation pattern to ensure that the collection of personal data is for the real benefit of the data subject in the use of the system service. The data subject should be informed of what personal data are collected and for what purposes. Individual Participation In addition to the Collection Limitation pattern, the Individual Participation pattern can be used so that the data subject remains entitled to the right to participate in the use of the collected personal data by the system and be informed of the issues related to the collected personal data.

จากตัวอย่างการพัฒนาแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูลดังข้างต้น ผู้วิจัยได้ดำเนินการพัฒนาแบบรูปการออกแบบสำหรับหลักความเป็นส่วนตัวของโออีซีดีทั้งหมด 8 แบบรูปดังนี้

1. แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Pattern)
2. แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (Data Quality Pattern)
3. แบบรูปการออกแบบสำหรับหลักการกำหนดวัตถุประสงค์ (Purpose Specification Pattern)
4. แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (Use Limitation Pattern)
5. แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Pattern)
6. แบบรูปการออกแบบสำหรับหลักการเปิดเผย (Openness Pattern)
7. แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (Individual Participation Pattern)
8. แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (Accountability Pattern)

โดยที่ความสัมพันธ์ของแบบรูปการออกแบบสำหรับความเป็นส่วนตัวทั้งหมดแสดงได้ดังรูปที่ 3.2 โดยความสัมพันธ์ระหว่างแบบรูปสามารถแบ่งได้เป็น 2 กลุ่มด้วยกันคือ กลุ่มที่ต้องปฏิบัติตามแบบรูปอื่นที่มีความสัมพันธ์กันจึงจะบรรลุเป้าหมายของแบบรูปนั้น และกลุ่มที่เลือกได้ว่าจะปฏิบัติตามแบบรูปอื่นที่มีความสัมพันธ์กันหรือไม่ ซึ่งสามารถอธิบายความสัมพันธ์ระหว่างแบบรูปได้ดังนี้

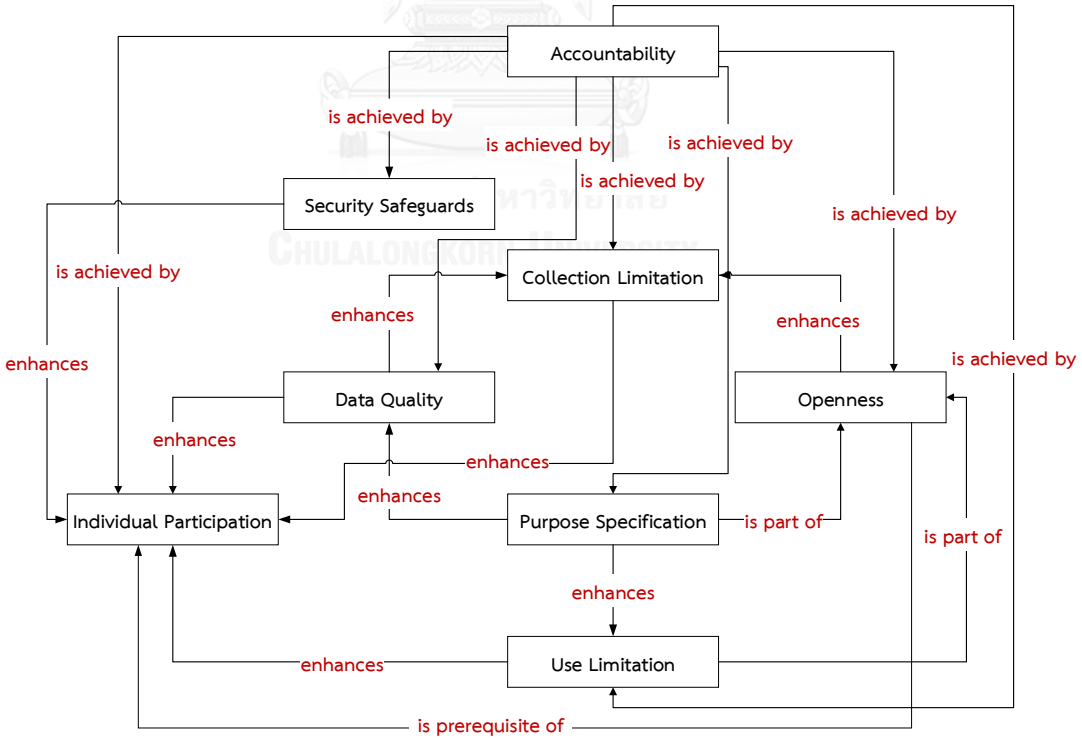
1. กลุ่มที่ต้องปฏิบัติตามแบบรูปอื่นที่มีความสัมพันธ์กันจึงจะบรรลุเป้าหมายของแบบรูปนั้น
    1.1 is achieved by การบรรลุเป้าหมายของแบบรูปการออกแบบทำโดยการปฏิบัติตามแบบรูปการออกแบบอื่น ๆ ที่มีความสัมพันธ์กัน จากรูปที่ 3.2 จะเห็นว่า การบรรลุเป้าหมายของแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (Accountability Pattern) นั้น จะทำได้โดยการปฏิบัติตามแบบรูปการออกแบบอื่น ๆ ให้ครบทั้ง 7 แบบรูป
    1.2 is prerequisite of ควรจะทำแบบรูปการออกแบบนั้นก่อนที่จะทำแบบรูปการออกแบบอื่น จากรูปที่ 3.2 จะเห็นว่าแบบรูปการออกแบบสำหรับหลักการเปิดเผย (Openness Pattern) ควรทำก่อนแบบรูปการออกแบบสำหรับการมีส่วนร่วมของบุคคล (Individual Participation Pattern) เนื่องจากก่อนที่เจ้าของข้อมูลจะมีส่วน

ร่วมในการจัดการข้อมูลส่วนบุคคลใด ๆ ของตน ระบบควรต้องแจ้งให้ทราบก่อนว่ามีข้อมูลใดเกี่ยวข้องบ้างและใช้งานข้อมูลนั้นอย่างไร

1.3 is part of แบบรูปการออกแบบนั้นเป็นส่วนหนึ่งหรือเป็นส่วนย่อยของแบบรูปการออกแบบอื่น เช่น แบบรูปการออกแบบสำหรับข้อกำหนดวัตถุประสงค์ (Purpose Specification Pattern) เป็นส่วนหนึ่งของแบบรูปการออกแบบสำหรับหลักการเปิดเผย (Openness Pattern) เนื่องจากในหลักการเปิดเผยนั้นต้องมีการกำหนดวัตถุประสงค์ด้วย

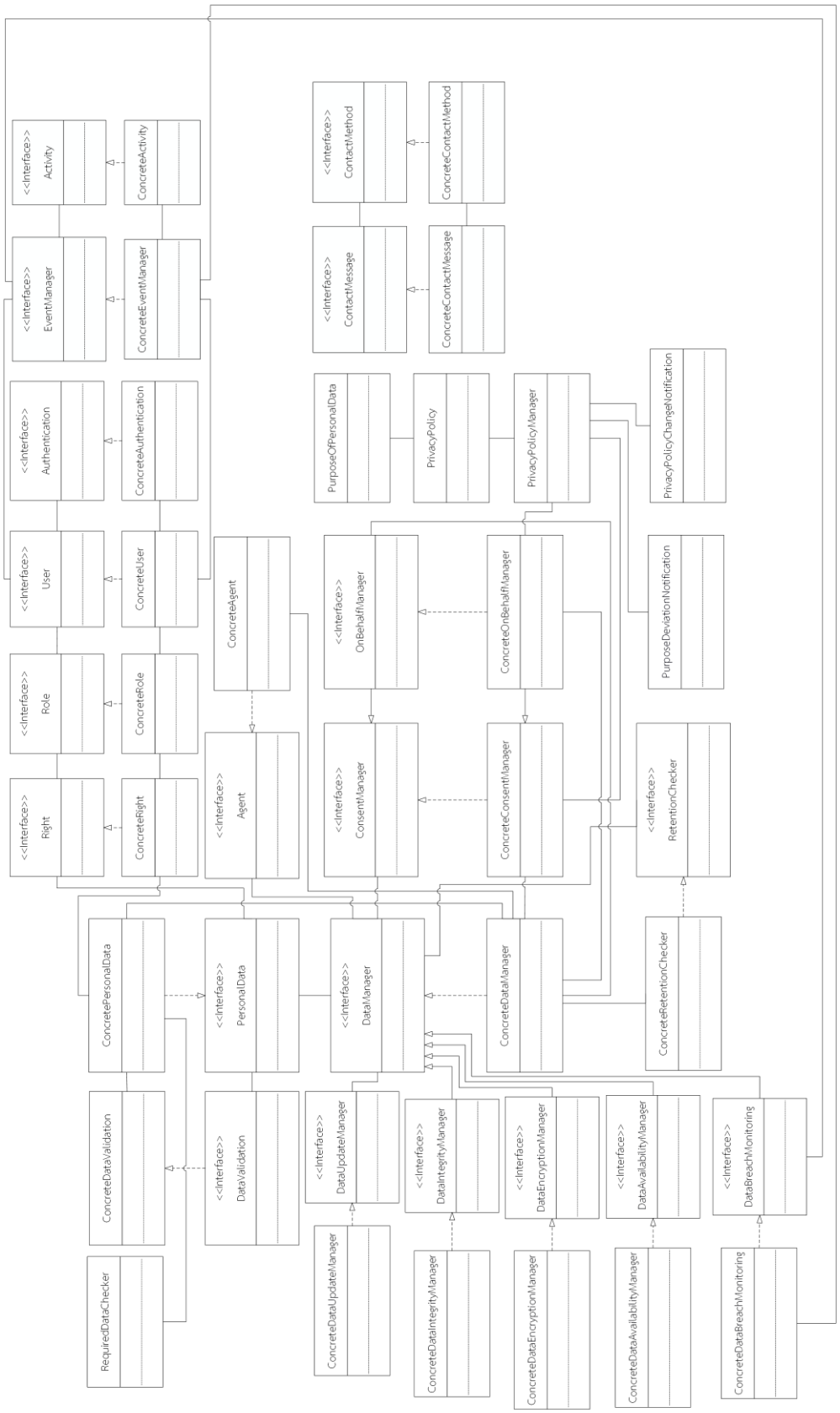
2. กลุ่มที่เลือกได้ว่าจะปฏิบัติตามแบบรูปอื่นที่มีความสัมพันธ์กันหรือไม่

2.1 enhances แบบรูปการออกแบบสามารถช่วยให้แบบรูปการออกแบบที่เกี่ยวข้องมีความสมบูรณ์ยิ่งขึ้น เช่น แบบรูปการณ์ออกแบบสำหรับหลักการเปิดเผย (Openness Pattern) จะช่วยให้แบบรูปการออกแบบสำหรับหลักข้อจำกัดการเก็บรวบรวมข้อมูล (Collection Limitation Pattern) มีความสมบูรณ์ยิ่งขึ้น เนื่องจากจะมีการเปิดเผยข้อมูลเพิ่มเติมให้ผู้เป็นเจ้าของข้อมูลทราบเกี่ยวกับนโยบายความเป็นส่วนตัวของระบบด้วยว่า ข้อมูลที่จะเก็บรวบรวมนั้นมีอะไรบ้าง เก็บเมื่อไร และจะถูกนำไปใช้งานอย่างไรก่อนที่ระบบจะร้องขอความยินยอมในการเก็บรวบรวมข้อมูล

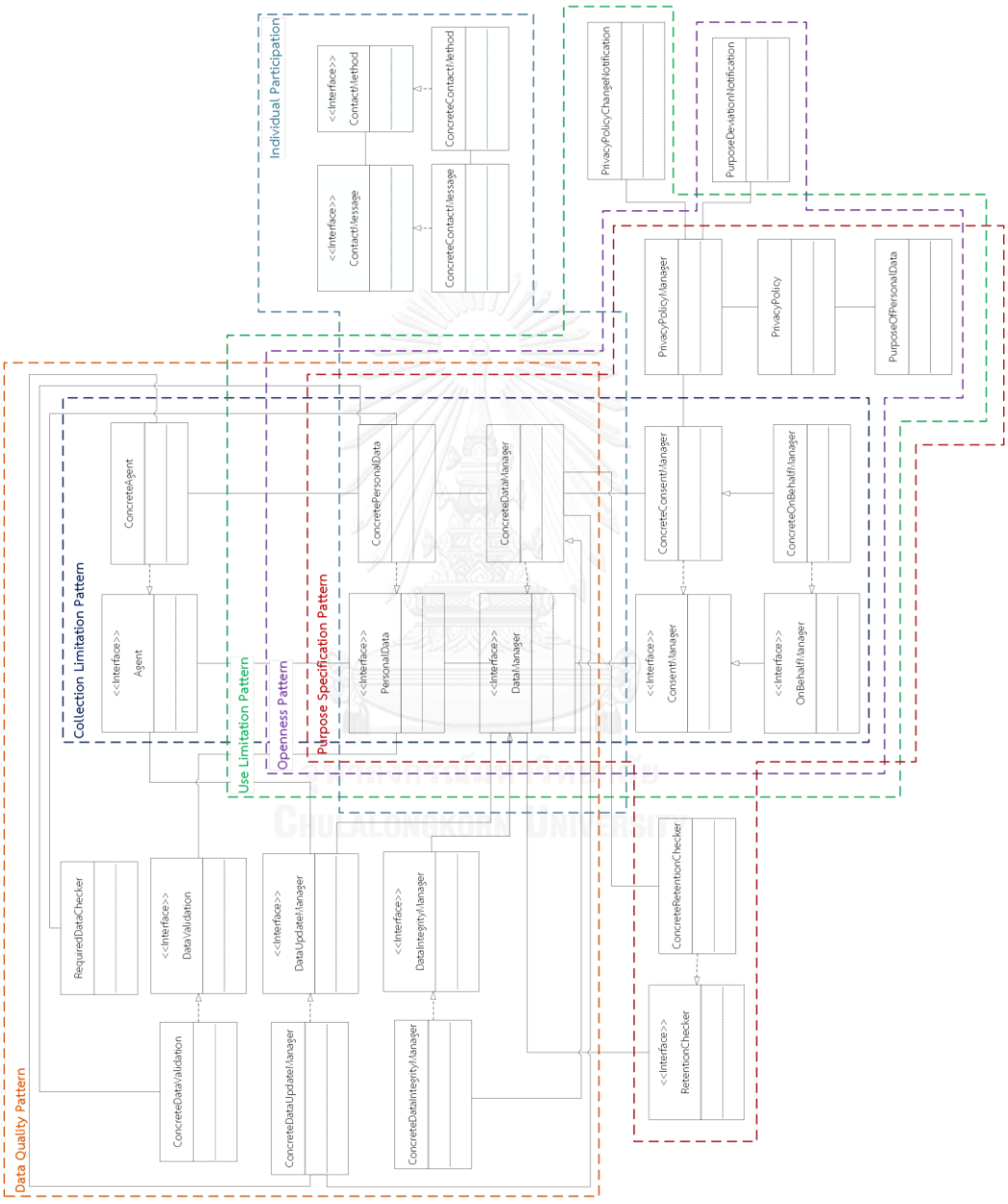


รูปที่ 3.2 ความสัมพันธ์ของแต่ละแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

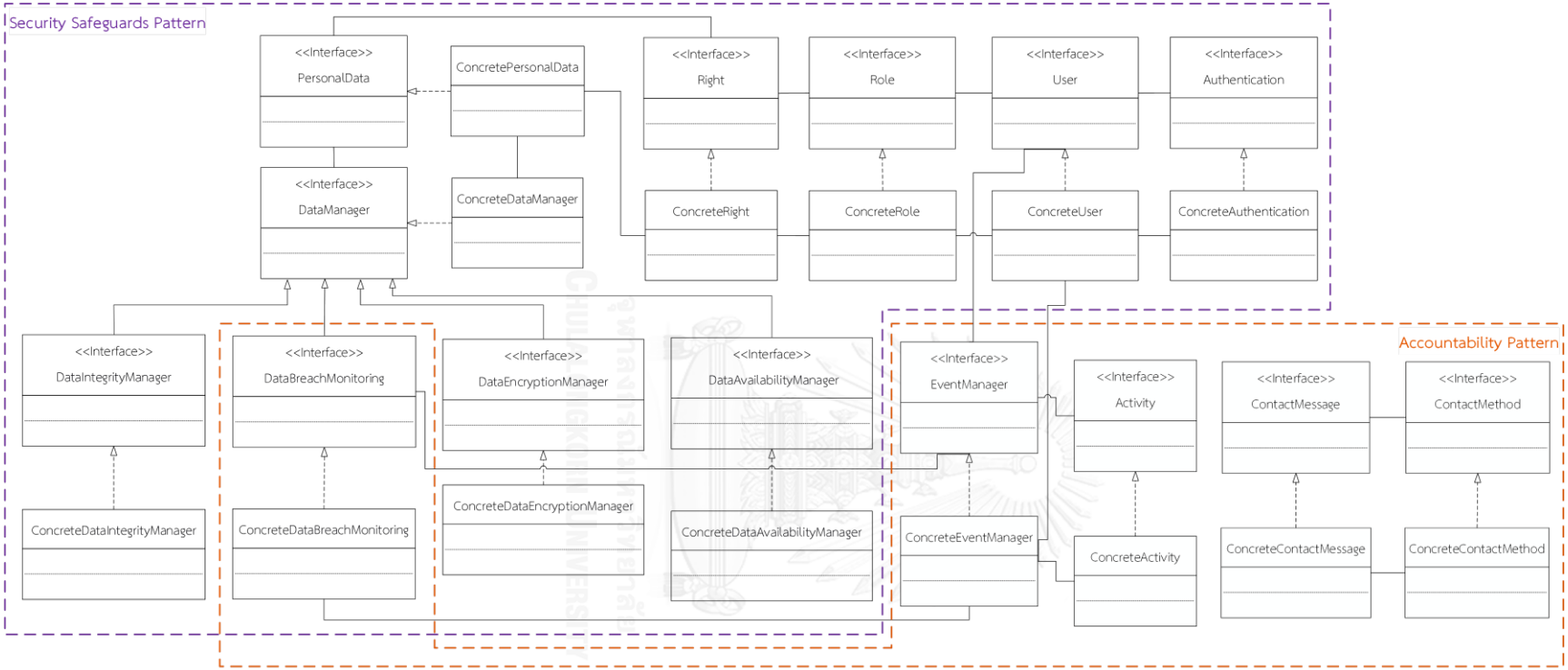
รูปที่ 3.3 แสดงภาพรวมของแผนภาพคลาสของแบบรูปการออกแบบสำหรับความเป็นส่วนตัวทั้งหมด ส่วนแผนภาพคลาสทั้งหมดโดยแบ่งส่วนตามแต่ละแบบรูปการออกแบบแสดงได้ดังรูปที่ 3.4 และ 3.5



รูปที่ 3.3 ภาพรวมของแผนภาพคลาสของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว



รูปที่ 3.4 แผนภาพคลาสของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว (ครอบคลุม 6 หลักการความเป็นส่วนตัว)



รูปที่ 3.5 แผนภาพคลาสของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว (ครอบคลุม 2 หลักการความเป็นส่วนตัว)

เมื่อออกแบบแบบรูปการออกแบบสำหรับความเป็นส่วนตัว ดังแสดงตัวอย่างในตารางที่ 3.4 เรียบร้อยแล้ว ผู้วิจัยได้สรุปว่าสาระสำคัญของหลักการความเป็นส่วนตัวในแต่ละรายละเอียดย่อยนั้นมีปรากฏอยู่ในส่วนใดของแบบรูปการออกแบบสาหรับความเป็นส่วนตัว สามารถแสดงได้ดังตารางที่ 3.5

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว

รายละเอียดย่อย		**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
		รหัส	**รายการความต้องการ**	
Collection Limitation Principle				
1. There should be limits to the collection of personal data.				
1.1	The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly.		ไม่มี	**Implementation** (ข้อ 1), **Related Patterns** (Data Quality)
1.2	The data collection should be restricted to the minimum necessary to fulfil the specified purpose.	PT-CL01	The system should be able to collect personal data that are necessary for use.	**Behavior and Structure** (Interface PersonalData), **Implementation** (ข้อ 2)
		PT-CL02	The system should be able to record the information about how long personal data will be retained.	
1.3	Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection.		ไม่มี	**Implementation** (ข้อ 3)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย		**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
		รหัส	**รายการความต้องการ**	
Collection Limitation Principle				
1.4	There should be limits to data collection activities of the data controller such that the data should not be collected by the means that is unknown or deceiving to the data subject.	PT-CL03	The system should be able to record the information about the source of personal data collection.	**Behavior and Structure** (Interface PersonalData, Interface Agent), **Implementation** (ข้อ 4), **Related Patterns** (Openness)
1.5	Civil rights of the data subject should be of concern in the data collection.		ไม่มี	**Implementation** (ข้อ 5), **Related Patterns** (Individual Participation)
2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.		PT-CL04	The system should let the data subject know about personal data collection.	**Behavior and Structure** (Interface ConsentManager)
3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.		PT-CL05	The system should allow the data subject to give or refuse consent to data collection.	**Behavior and Structure** (Interface ConsentManager)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Collection Limitation Principle			
4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.	PT-CL06	The system should be able to record the information about the party who gives consent to data collection on behalf of the data subject.	**Behavior and Structure** (Interface OnBehalfManager), **Implementation** (ข้อ 6)
Data Quality Principle			
1. Personal data should be relevant to the purposes for which they are to be used.		ไม่มี	**Implementation** (ข้อ 1), **Related Patterns** (Purpose Specification)
2. Personal data, to the extent necessary for their purposes, should be accurate.	PT-DQ02	The system should be able to validate the collected personal data for accuracy.	**Behavior and Structure** (Interface DataValidation, Interface DataIntegrityManager), **Implementation** (ข้อ 2)
	PT-DQ03	The system should be able to check integrity of the collected personal data for accuracy.	
3. Personal data, to the extent necessary for their purposes, should be complete.	PT-DQ01	The system should be able to check, for data completeness, that all required personal data are collected.	**Behavior and Structure** (Class RequiredDataChecker)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Data Quality Principle			
4. Personal data, to the extent necessary for their purposes, should be kept up-to-date.	PT-DQ04	The system should be able to collect current personal data for data up-to-dateness.	**Behavior and Structure** (Interface DataUpdateManager), **Implementation** (ข้อ 2), **Related Patterns** (Individual Participation)
Purpose Specification Principle			
1. The purposes for which personal data are collected should be specified not later than at the time of data collection.	PT-PS01	The system should be able to inform the information about the purpose for which the collected data are to be used.	**Behavior and Structure** (Class PurposeOfPersonalData), Implementation (ข้อ 2)
2. The subsequent use of personal data should be limited to the fulfilment of the purposes specified prior to or at the time of collection.		ไม่มี	**Implementation** (ข้อ 4)
3. The subsequent use of personal data may be for purposes other than the original purposes, but such purposes should be specified as change.		ไม่มี	**Implementation** (ข้อ 4)
4. Later changes of purposes should be specified.		ไม่มี	**Implementation** (ข้อ 4)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Purpose Specification Principle			
5. On each occasion of change of purpose, the new purposes should not be incompatible with the original purposes.		ไม่มี	**Implementation** (ข้อ 4)
6. Specification of purpose should be made by public declarations or information to the data subject.		ไม่มี	**Implementation** (ข้อ 3), **Related Patterns** (Openness)
7. When data no longer serve a purpose, and if it is practicable, it may be necessary to have them destroyed (erased) or given an anonymous form.	PT-PS02	The system should be able to check the retention period or expiration date of personal data.	**Behavior and Structure** (Interface RetentionChecker), **Implementation** (ข้อ 5)
	PT-PS03	The system should be able to delete or anonymize personal data after their retention has expired.	
8. The Purpose Specification Principle is closely associated with the Data Quality Principle and the Use Limitation Principle.		ไม่มี	**Implementation** (ข้อ 1) **Related Patterns** (Data Quality, Use Limitation)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Use Limitation Principle			
1. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except with the consent of the data subject or his representative.	PT-UL01	The system should let the data subject know about the use of personal data which deviates from the specified purposes.	**Behavior and Structure** (Interface ConsentManager), **Implementation** (ข้อ1, 2), **Related Patterns** (Openness)
2. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except by the authority of law or licences granted by supervisory bodies.	PT-UL02	The system should allow the data subject to give or refuse consent to the use of personal data which deviates from the specified purposes.	**Behavior and Structure** (Interface PurposeDeviation Notification), **Implementation** (ข้อ1), **Related Patterns** (Openness)
Security Safeguards Principle			
1. Personal data should be protected by reasonable security safeguards against unauthorized disclosure of data, based on informational measures such as enciphering.	PT-SS01	The system should be able to encrypt the personal data to ensure confidentiality.	**Behavior and Structure** (Interface DataEncryption Maneger), **Implementation** (ข้อ 3)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Security Safeguards Principle			
2. Personal data should be protected by reasonable security safeguards against unauthorized access, destruction, use, and modification of data, based on organizational measures such as authority levels with regard to access to data.	PT-SS02	The system should be able to authenticate users of the system before accessing personal data.	**Behavior and Structure** (Interface User, Interface Role, Interface Right, Interface Authentication), **Implementation** (ข้อ 2)
	PT-SS03	The system should be able to define access rights to create, destroy, view, and modify personal data based on authority levels or roles within the organization.	
	PT-SS04	The system should be able to control access to the personal data by different access types including create, destroy, view, and modify, based on authority levels or roles within the organization.	

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Security Safeguards Principle			
3. Personal data should be protected by reasonable security safeguards against loss or destruction and modification of data, both accidentally and deliberately, where data integrity would be affected.	PT-SS05	The system should be able to provide a mechanism to ensure data integrity.	**Behavior and Structure** (Interface DataIntegrity Manager), **Implementation** (ข้อ 4), **Related Patterns** (Data Quality)
4. Personal data should be protected by reasonable security safeguards against loss of access by which data availability would be affected.	PT-SS06	The system should be able to provide a mechanism to ensure data availability.	**Behavior and Structure** (Interface DataAvailability Manager), **Implementation** (ข้อ 5)
5. Personal data should be protected by reasonable security safeguards against threats of data breach, based on informational measures such as monitoring of unusual activities and responses to them.	PT-SS07	The system should be able to monitor and record access to personal data.	**Behavior and Structure** (Interface DataBreach Mornitoring), **Implementation**, **Related Patterns** (Data Quality)
	PT-SS08	The system should be able to check for suspicious data breach and notify the data controller and data subject.	

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Security Safeguards Principle			
6. Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees' security training, up-to-date security safeguards).		ไม่มี	**Implementation** (ข้อ 7)
Openness Principle			
1. There should be a general policy of openness about developments, practices and policies with respect to personal data.	PT-OP01	The system should be able to record the information about the privacy policy, including the existence and nature of personal data, main purpose of collection and use, and identity and usual residence of the data controller.	**Behavior and Structure** (Interface PrivacyPolicyManager, Interface PrivacyPolicy, Interface PurposeOf PersonalData)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Openness Principle			
2. Means should be readily available of establishing the existence and nature of personal data.	PT-OP02	The system should let the data subject know about the privacy policy.	**Behavior and Structure** (Interface PrivacyPolicyManager, Interface PrivacyPolicyChange Notification)
	PT-OP03	The system should let the data subject know about change of the privacy policy.	
	PT-OP04	The system should provide the data subject with access to the privacy policy.	
3. Means should be readily available of establishing the main purposes of the use of personal data.	PT-OP02	The system should let the data subject know about the privacy policy.	**Behavior and Structure** (Interface PrivacyPolicyManager, Interface PrivacyPolicyChange Notification)
	PT-OP03	The system should let the data subject know about change of the privacy policy.	
	PT-OP04	The system should provide the data subject with access to the privacy policy.	

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Openness Principle			
4. Means should be readily available of establishing the identity and usual residence of the data controller.	PT-OP02	The system should let the data subject know about the privacy policy.	**Behavior and Structure** (Interface PrivacyPolicyManager, Interface PrivacyPolicyChange Notification)
	PT-OP03	The system should let the data subject know about change of the privacy policy.	
	PT-OP04	The system should provide the data subject with access to the privacy policy.	
5. “Readily available” implies that data subjects should be able to obtain information without unreasonable effort as to time, advance knowledge, travelling, and so forth, and without unreasonable cost.	PT-OP02	The system should let the data subject know about the privacy policy.	**Behavior and Structure** (Interface PrivacyPolicyManager, Interface PrivacyPolicyChange Notification), **Implementation** (ข้อ 2)
	PT-OP03	The system should let the data subject know about change of the privacy policy.	
	PT-OP04	The system should provide the data subject with access to the privacy policy.	

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Openness Principle			
6. Regular information from the data controller regarding the privacy policy should be published.	PT-OP01	The system should be able to record the information about the privacy policy, including the existence and nature of personal data, main purpose of collection and use, and identity and usual residence of the data controller.	**Behavior and Structure** (Interface PrivacyPolicyManager, Interface PrivacyPolicy, Interface PurposeOf PersonalData), **Implementation**
7. The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle.		ไม่มี	**Implementation** (ข้อ 3), **Related Patterns** (Individual Participation
Individual Participation Principle			
1. A data subject can obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him/her (or his/her personal data).		ไม่มี	**Implementation** (ข้อ 1), **Related Patterns** (Openness, Collection Limitation, Use Limitation)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Individual Participation Principle			
2. A data subject should have the right to access to his/her personal data within a reasonable time when a request to access is made.	PT-IP02	The system should allow the data subject to access his/her personal data.	**Behavior and Structure** (Interface DataManager, Interface PersonalData), **Implementation** (ข้อ 3), **Related Patterns** (Security Safeguards)
3. A data subject should have the right to access to his/her personal data at a reasonable cost.	PT-IP02	The system should allow the data subject to access his/her personal data.	**Behavior and Structure** (Interface DataManager, Interface PersonalData), **Implementation** (ข้อ 3), **Related Patterns** (Security Safeguards)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Individual Participation Principle			
4. A data subject should have the right to access to his/her personal data in a reasonable manner.	PT-IP02	The system should allow the data subject to access his/her personal data.	**Behavior and Structure** (Interface DataManager, Interface PersonalData), **Implementation** (ข้อ 3), **Related Patterns** (Security Safeguards)
5. A data subject should have the right to access to his/her personal data in a form that is readily intelligible.	PT-IP02	The system should allow the data subject to access his/her personal data.	**Behavior and Structure** (Interface DataManager, Interface PersonalData), **Implementation** (ข้อ 3), **Related Patterns** (Security Safeguards)
6. A data subject should have the right to reasons for, and challenge, adverse decisions when requests for confirmation or access are denied.	PT-IP01	The system should allow inquiry of and response to issues related to personal data.	**Behavior and Structure** (Interface ContactMessage, Interface ContactMethod), **Implementation** (ข้อ 3)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
	รหัส	**รายการความต้องการ**	
Individual Participation Principle			
7. A data subject should have the right to challenge his/her personal data.	PT-IP01	The system should allow inquiry of and response to issues related to personal data.	**Behavior and Structure** (Interface ContactMessage, Interface ContactMethod), **Implementation** (ข้อ 4)
8. A data controller should erase, rectify, complete, or amend the data subject's personal data if the challenge is successful.	PT-IP03	The system should allow the data controller to erase and update the data subject's personal data.	**Behavior and Structure** (Interface DataManager, Interface PersonalData)
Accountability Principle			
1. A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles.		ไม่มี	**Implementation** (ข้อ 1), **Related Patterns** (All Privacy Pattern)
2. A data controller should have in place a privacy management program that is tailored to its operations and integrated into its governance structure.		ไม่มี	**Implementation** (ข้อ 1)

ตารางที่ 3.5 การปรากฏของรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปความเป็นส่วนตัว (ต่อ)

รายละเอียดย่อย	**รายการความต้องการของแบบรูปการออกแบบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
Accountability Principle	**รหัส**	**รายการความต้องการ**	
3. A data controller should provide ongoing monitoring, periodic assessment of data breach, and response to incidents.	PT-AC02	The system should be able to perform periodic assessment of data breach.	**Behavior and Structure** (Interface DataBreach Monitoring)
	PT-AC03	The system should be able to handle data breach.	
	PT-AC05	The system should be able to report on data breach checking.	
4. There should be a method for the data subject to inquire about issues related to privacy of personal data.	PT-AC04	The system should allow inquiry of and response to issues related to personal data.	**Behavior and Structure** (Interface ContactMessage, Interface ContactMethod
5. A data controller should provide evidence of its privacy management program to the privacy enforcement authorities.	PT-AC01	The system should be able to record events that occur in the system.	**Implementation** (ข้อ 2), **Behavior and Structure** (Interface User, Interface EventManager, Interface Activity)
6. A data controller should provide notice to the privacy enforcement authorities and data subject in the event of significant data breach.	PT-AC06	The system should be able to notify relevant parties, i.e. privacy enforcement authorities and data subject, in the event of significant data breach.	**Implementation** (ข้อ 2), **Behavior and Structure** (Interface DataBreach Monitoring)

บทที่ 4
การตรวจสอบความครบถ้วนของแบบรูปการออกแบบ

ในบทนี้จะกล่าวถึงการทดลองในการตรวจสอบความครบถ้วนของแบบรูปการออกแบบที่นำเสนอโดยใช้รายการตรวจสอบ (Checklist) เพื่อให้ผู้มีประสบการณ์ด้านการวิเคราะห์ และออกแบบระบบที่เกี่ยวข้องกับข้อมูลส่วนบุคคลทำการตรวจสอบว่าแบบรูปการออกแบบที่นำเสนอนั้นมีความครบถ้วนตามหลักการความเป็นส่วนตัวหรือไม่ และเพื่อรับทราบข้อคิดเห็นจากผู้ตรวจสอบ โดยรายการตรวจสอบนี้จะตรวจสอบว่าสาระสำคัญของหลักการความเป็นส่วนตัวมีปรากฏอยู่ในแบบรูปการออกแบบหรือไม่ ซึ่งสามารถแสดงตัวอย่างรายการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูลได้ดังตารางที่ 4.1 โดยขั้นตอนการตรวจสอบความครบถ้วนของแบบรูปการออกแบบประกอบด้วย 1) การกำหนดวัตถุประสงค์การตรวจสอบ 2) การวางแผนการตรวจสอบ 3) การให้ข้อมูลกับผู้ตรวจสอบ 4) การดำเนินการตรวจสอบ 5) การรวบรวมและสรุปผลการตรวจสอบ ซึ่งแต่ละขั้นตอนมีรายละเอียดดังนี้

4.1 การกำหนดวัตถุประสงค์การตรวจสอบ

การตรวจสอบความครบถ้วนของแบบรูปการออกแบบที่นำเสนอโดยใช้รายการตรวจสอบ มีวัตถุประสงค์เพื่อตรวจสอบว่าแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอนั้นมีความครบถ้วนตามหลักการความเป็นส่วนตัวของโออีซีดีหรือไม่ หากแบบรูปการออกแบบที่นำเสนอยังขาดสาระสำคัญของหลักการความเป็นส่วนตัวในส่วนใด ผู้วิจัยจะทำการปรับปรุงแบบรูปการออกแบบที่นำเสนอให้มีความครบถ้วนตามหลักการความเป็นส่วนตัว

4.2 การวางแผนการตรวจสอบ

การตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับความเป็นส่วนตัวจะใช้รายการตรวจสอบรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่กล่าวถึงในบทที่ 3 หัวข้อที่ 3.1 ว่ามีปรากฏอยู่ในแบบรูปการออกแบบหรือไม่ ซึ่งสามารถแสดงตัวอย่างรายการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูลได้ดังตารางที่ 5 โดยกำหนดให้ผู้ที่มีประสบการณ์ในการพัฒนาซอฟต์แวร์ หรือวิเคราะห์และออกแบบระบบที่เกี่ยวข้องกับข้อมูลส่วนบุคคลอย่างน้อย 2 ปี เป็นจำนวน 5 คนเป็นผู้ทำการตรวจสอบในแต่ละแบบรูปการออกแบบ

ตารางที่ 4.1 ตัวอย่างรายการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

รายละเอียดย่อย		**รายการความต้องการของแบบรูปการออกแบบ**		**การตรวจสอบ**		**ปรากฏในส่วนใดของแบบรูปการออกแบบ**
		รหัส	**รายการความต้องการ**	**พบ**	**ไม่พบ**	
Collection Limitation Principle						
1. There should be limits to the collection of personal data.						
1.1	The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly.		ไม่มี			
1.2	The data collection should be restricted to the minimum necessary to fulfil the specified purpose.	PT-CL01	The system should be able to collect personal data that are necessary for use.			
		PT-CL02	The system should be able to record the information about how long personal data will be retained.			
1.3	Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection.		ไม่มี			

ตารางที่ 4.1 รายการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

รายละเอียดย่อย		รายการความต้องการของแบบรูปการออกแบบ		การตรวจสอบ		ปรากฏในส่วนใดของแบบรูปการออกแบบ
		รหัส	รายการความต้องการ	พบ	ไม่พบ	
Collection Limitation Principle						
1.4	There should be limits to data collection activities of the data controller such that the data should not be collected by the means that is unknown or deceiving to the data subject.	PT-CL03	The system should be able to record the information about the source of personal data collection.			
1.5	Civil rights of the data subject should be of concern in the data collection.		ไม่มี			
2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.		PT-CL04	The system should let the data subject know about personal data collection.			
3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.		PT-CL05	The system should allow the data subject to give or refuse consent to data collection.			
4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.		PT-CL06	The system should be able to record the information about the party who gives consent to data collection on behalf of the data subject.			

4.3 การให้ข้อมูลกับผู้ตรวจสอบ

การให้ข้อมูลกับผู้ตรวจสอบคือ การอธิบายให้กับผู้ตรวจสอบทราบถึงวัตถุประสงค์ของการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว หลักความเป็นส่วนตัวที่นำมาใช้ในงานวิจัย แนวทางในการพัฒนาแบบรูปการออกแบบ และแนะนำแนวทางในการตรวจสอบความครบถ้วนของแบบรูปการออกแบบ

4.4 การดำเนินการตรวจสอบ

การตรวจสอบความครบถ้วนของแบบรูปการออกแบบดำเนินการโดยใช้รายการตรวจสอบดังแสดงตัวอย่างรายการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูลได้ดังตารางที่ 5 และนำแบบรูปการออกแบบที่นำเสนอให้ผู้ตรวจสอบได้อ่านและศึกษา เพื่อนำมาตรวจสอบว่าสาระสำคัญของหลักการความเป็นส่วนตัวมีปรากฏอยู่ในแบบรูปการออกแบบหรือไม่

4.5 การรวบรวมและสรุปผลการตรวจสอบ

ผู้วิจัยได้ทำการรวบรวมผล และสรุปผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบที่นำเสนอ ซึ่งตัวอย่างผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูลแสดงดังตารางที่ 4.2 สำหรับผลการตรวจสอบแบบรูปการออกแบบทั้งหมดแสดงไว้ในภาคผนวก จ.

ตารางที่ 4.2 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

รายละเอียดย่อย		คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
		ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Collection Limitation Principle						
1. There should be limits to the collection of personal data.						
1.1	The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly.	Implementation	Implementation	Implement	Implementation	Implementation
1.2	The data collection should be restricted to the minimum necessary to fulfil the specified purpose.	Implementation, Behavior and Structure	Behavior and Structure	Behavior and Structure, Sample Design and Code	Implementation, Sample Design and Code	Behavior and Structure, Implementation
1.3	Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection.	Implementation	Implementation	Implementation	Implementation	Implementation
1.4	There should be limits to data collection activities of the data controller such that the data should not be collected by the means that is unknown or deceiving to the data subject.	Implementation, Behavior and Structure	Behavior and Structure	Implementation, Behavior and Structure, Sample Design and Code	Implementation, Sample Design and Code	Behavior and Structure

ตารางที่ 4.2 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Collection Limitation Principle					
1.5 Civil rights of the data subject should be of concern in the data collection.	Implementation	Implementation	Implementation	Implementation	Implementation
2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.	Intent, Behavior and Structure	Behavior and Structure	Behavior and Structure, Sample Design and Code	Intent, Sample Design and Code	Behavior and Structure
3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.	Intent, Behavior and Structure	Behavior and Structure	Behavior and Structure, Sample Design and Code	Intent, Sample Design and Code	Behavior and Structure
4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.	Implementation, Behavior and Structure	Behavior and Structure	Implementation, Behavior and Structure, Sample Design and Code	Implementation, Sample Design and Code	Behavior and Structure, Implementation

จากผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว ผลลัพธ์ที่ได้มีความใกล้เคียงกับรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ปรากฏในแบบรูปการออกแบบตามที่ผู้วิจัยได้สรุปและกล่าวไว้ในตารางที่ 3.5 อย่างไรก็ตาม มีบางรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบพบว่ามีปรากฏในแบบรูปการออกแบบ แต่เป็นการพบผิดตำแหน่งหรือไม่มีความใกล้เคียงกับที่กล่าวไว้ในตารางที่ 3.5 ซึ่งสามารถแสดงผลการตรวจพบผิดตำแหน่งได้ดังตารางที่ 4.3

ตารางที่ 4.3 รายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบพบแต่ผิดตำแหน่ง

หลักการความเป็นส่วนตัว	รายละเอียดย่อยที่ผู้ตรวจสอบพบผิดตำแหน่ง	จำนวนผู้พบผิด	ตำแหน่งที่พบผิด	ตำแหน่งที่ปรากฏจริง
1. หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	-	-	-	-
2. หลักคุณภาพของข้อมูล (Data Quality Principle)	รายละเอียดย่อยข้อที่ 1 “Personal data should be relevant to the purposes for which they are to be used.”	2	Motivation	Implementation, Related Patterns
3. หลักข้อกำหนดวัตถุประสงค์ (Purpose Specification Principle)	รายละเอียดย่อยข้อที่ 2 “The subsequent use of personal data should be limited to the fulfilment of the purposes specified prior to or at the time of collection.”	1	Motivation	Implementation
4. หลักข้อจำกัดในการใช้ข้อมูล (Use Limitation Pattern)	-	-	-	-

ตารางที่ 4.3 รายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบพบแต่ผิดตำแหน่ง (ต่อ)

หลักการความเป็นส่วนตัว	รายละเอียดย่อยที่ผู้ตรวจสอบพบผิดตำแหน่ง	จำนวนผู้พบผิด	ตำแหน่งที่พบผิด	ตำแหน่งที่ปรากฏจริง
5. หลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Principle)	รายละเอียดย่อยข้อที่ 6 “Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees’ security training, up-to-date security safeguards).”	1	Consequences	Implementation
6. หลักการเปิดเผย (Openness Principle)	รายละเอียดย่อยข้อที่ 7 “The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle.”	2	Behavior and Structure	Implementation, Related Patterns
7. หลักการมีส่วนร่วมของบุคคล (Individual Participation Principle)	-	-	-	-

ตารางที่ 4.3 รายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบพบแต่ผิดตำแหน่ง (ต่อ)

หลักการความเป็นส่วนตัว	รายละเอียดย่อยที่ผู้ตรวจสอบพบผิดตำแหน่ง	จำนวนผู้พบผิด	ตำแหน่งที่พบผิด	ตำแหน่งที่ปรากฏจริง
8. หลักความรับผิดชอบ (Accountability Principle)	รายละเอียดย่อยข้อที่ 1 “A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles.”	2	Intent, Behavior and Structure	Implementation, Related Patterns

จากตารางที่ 4.3 แสดงรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบพบแต่ผิดตำแหน่ง ผู้วิจัยได้จัดทำสรุปจำนวนการตรวจพบรายละเอียดย่อยของหลักการความเป็นส่วนตัวผิดตำแหน่ง ดังแสดงในตารางที่ 4.4

ตารางที่ 4. 4 สรุปจำนวนการตรวจพบรายละเอียดย่อยของหลักการความเป็นส่วนตัวผิดตำแหน่ง

หลักการความเป็นส่วนตัว	จำนวนรายการที่พบผิด	จำนวนผู้พบผิด
1. หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	-	-
2. หลักคุณภาพของข้อมูล (Data Quality Principle)	1	2
3. หลักข้อกำหนดวัตถุประสงค์ (Purpose Specification Principle)	1	1
4. หลักข้อจำกัดในการใช้ข้อมูล (Use Limitation Pattern)	-	-
5. หลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Principle)	1	1
6. หลักการเปิดเผย (Openness Principle)	1	2
7. หลักการมีส่วนร่วมของบุคคล (Individual Participation Principle)	-	-
8. หลักความรับผิดชอบ (Accountability Principle)	1	2

เนื่องจากมีผู้ตรวจสอบพบผิดตำแหน่งเป็นจำนวนน้อย ผู้วิจัยได้สอบถามและอธิบายเพิ่มเติม ซึ่งผู้ตรวจสอบเหล่านั้นเห็นด้วยกับคำชี้แจงและตำแหน่งที่ปรากฏจริง

นอกจากนี้ยังมีกรณีที่บางรายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบไม่พบว่ามีปรากฏในแบบรูปการออกแบบ แสดงดังตารางที่ 4.5

ตารางที่ 4.5 รายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้ตรวจสอบไม่พบว่ามีปรากฏในแบบรูปการออกแบบ

หลักการความเป็นส่วนตัว	รายละเอียดย่อยที่ผู้ตรวจสอบไม่พบว่ามีปรากฏในแบบรูปการออกแบบ	จำนวนคนที่ไม่พบ
1. หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	-	-
2. หลักคุณภาพของข้อมูล (Data Quality Principle)	-	-
3. หลักข้อกำหนดวัตถุประสงค์ (Purpose Specification Principle)	รายละเอียดย่อยข้อที่ 5 “On each occasion of change of purpose, the new purposes should not be incompatible with the original purposes.”	1
4. หลักข้อจำกัดในการใช้ข้อมูล (Use Limitation Principle)	-	-
5. หลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Principle)	รายละเอียดย่อยข้อที่ 6 “Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees’ security training, up-to-date security safeguards).”	2
6. หลักการเปิดเผย (Openness Principle)	-	-
7. หลักการมีส่วนร่วมของบุคคล (Individual Participation Principle)	รายละเอียดย่อยข้อที่ 3 “A data subject should have the right to access to his/her personal data at a reasonable cost.”	1

ตารางที่ 4.5 รายละเอียดย่อยของหลักการความเป็นส่วนตัวที่ผู้เชี่ยวชาญไม่พบว่ามีปรากฏในแบบรูปการออกแบบ (ต่อ)

หลักการความเป็นส่วนตัว	รายละเอียดย่อยที่ผู้ตรวจสอบไม่พบว่ามีปรากฏในแบบรูปการออกแบบ	จำนวนที่ไม่พบ
8. หลักความรับผิดชอบ (Accountability Principle)	รายละเอียดย่อยข้อที่ 1 “A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles”	1
	รายละเอียดย่อยข้อที่ 2 “A data controller should have in place a privacy management program that is tailored to its operations and integrated into its governance structure.”	2

จากตารางที่ 4.5 ผู้วิจัยได้ทำการตรวจสอบในแต่ละรายละเอียดย่อยที่ผู้ตรวจสอบไม่พบปรากฏในแบบรูปการออกแบบสรุปว่า

1. หลักข้อกำหนดวัตถุประสงค์ รายละเอียดย่อยข้อที่ 5 “On each occasion of change of purpose, the new purposes should not be incompatible with the original purposes.” มีปรากฏในแบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ในหัวข้อ Implementation ข้อย่อยที่ 4 แล้ว

2. หลักการรักษาความมั่นคงของข้อมูล รายละเอียดย่อยข้อที่ 6 “Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees’ security training, up-to-date security safeguards).” มีปรากฏในแบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูลในหัวข้อ Implementation ข้อย่อยที่ 7 แล้ว

3. หลักการมีส่วนร่วมของบุคคล รายละเอียดย่อยข้อที่ 3 “A data subject should have the right to access to his/her personal data at a reasonable cost.” มีปรากฏในแบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคลในหัวข้อ Implementation ข้อย่อยที่ 3 แล้ว

4. หลักความรับผิดชอบ รายละเอียดย่อยข้อที่ 2 “A data controller should have in place a privacy management program that is tailored to its operations and integrated into its governance structure.” มีปรากฏในแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ ในหัวข้อ Implementation ข้อย่อยที่ 1 แล้ว

5. หลักความรับผิดชอบ รายละเอียดย่อยข้อที่ 1 “A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles.” มีปรากฏในแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ ในหัวข้อ Related Patterns

จากผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับความเป็นส่วนตัว ประเด็นปัญหาที่พบในการตรวจสอบคือ แบบรูปการออกแบบเป็นภาษาอังกฤษคำศัพท์บางคำค่อนข้างยากผู้ตรวจสอบอ่านแล้วไม่เข้าใจ และรายละเอียดย่อยที่ไม่มีปรากฏในหัวข้อ “Behavior and Structure” ผู้ตรวจสอบจะเข้าใจว่าไม่มีปรากฏในแบบรูป ต้องอธิบาย และยกตัวอย่างให้ผู้เชี่ยวชาญทราบว่าได้มีการเขียนคำอธิบายไว้ในหัวข้อ “Implementation” หรือ “Related Pattern” เนื่องจากในบางรายละเอียดย่อยนั้นไม่สามารถเขียนออกมาในรูปแบบของแผนภาพยูเอ็มแอลได้โดยตรง เช่น รายละเอียดย่อยของหลักการมีส่วนร่วมของบุคคล รายละเอียดย่อยข้อที่ 3 “A data subject should have the right to access to his/her personal data at a reasonable cost.” เป็นต้น และผู้ตรวจสอบได้ให้คำแนะนำ และข้อควรปรับปรุงไว้ดังตารางที่ 4.6

ตารางที่ 4.6 รายการคำแนะนำ และข้อควรปรับปรุง

หลักการความเป็นส่วนตัว	รายการคำแนะนำ และข้อควรปรับปรุง	การปรับปรุง
1. หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	-	-
2. หลักคุณภาพของข้อมูล (Data Quality Principle)	ควรมีการคำนึงถึงกรณีที่ข้อมูลในระบบมีความทันสมัยกว่าแหล่งที่มาของข้อมูล ซึ่งอาจทำให้การอัพเดทอาจทำให้ข้อมูลในระบบมีความทันสมัยลดลงได้	การรวบรวมข้อมูลต้องทำจากเจ้าของข้อมูลหรือต้นทางของข้อมูลหรือแหล่งที่เชื่อถือได้ หากเจ้าของข้อมูลหรือแหล่งข้อมูลที่เชื่อถือได้ให้ข้อมูลไม่ทันสมัย จะถือว่าเกินขอบเขตของงานวิจัย
	การตรวจสอบความถูกต้อง และครบถ้วนของข้อมูลที่เก็บรวบรวมข้อมูล ควรจะตรวจตั้งแต่ตอนที่กรอกค่าจากหน้าจอเลย ไม่ต้องสร้างเป็นคลาสมาตรวจสอบเมื่อส่งค่าเข้ามาแล้ว	ขึ้นอยู่กับการออกแบบ ภาษาที่ใช้ และระบบที่พัฒนา เช่น ผู้พัฒนาระบบสามารถประยุกต์ใช้คลาสการตรวจสอบข้อมูลในการพัฒนาส่วนหน้า (Font End) ส่วนคลาสอื่น ๆ ประยุกต์ในส่วนหลัง (Back End) ของแอปพลิเคชันได้
3. หลักข้อกำหนดวัตถุประสงค์ (Purpose Specification Principle)	-	-
4. หลักข้อจำกัดในการใช้ข้อมูล (Use Limitation Principle)	-	-
5. หลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Principle)	-	-
6. หลักการเปิดเผย (Openness Principle)	การเก็บข้อมูลนโยบายความเป็นส่วนตัวในรูปแบบควรเก็บในรูปแบบของไฟล์ ไม่จำเป็นต้องทำเป็นคลาส	ขึ้นอยู่กับการพัฒนา แต่ควรสามารถเก็บเป็นไฟล์ได้ แต่ควรมีคลาสเพื่อใช้ในการเก็บข้อมูลแอตทริบิวต์ของไฟล์นโยบาย (เช่น รหัส เวอร์ชัน วันที่สร้าง ผู้สร้าง การเปลี่ยนแปลงจากเวอร์ชันก่อนหน้า)

ตารางที่ 4.6 รายการคำแนะนำ และข้อควรปรับปรุง (ต่อ)

หลักการความเป็นส่วนตัว	รายการคำแนะนำ และข้อควรปรับปรุง	การปรับปรุง
7. หลักการมีส่วนร่วมของบุคคล (Individual Participation Principle)	-	-
8. หลักความรับผิดชอบ (Accountability Principle)	การเก็บรายละเอียดการใช้งาน (User Detail และ Activity Detail) ในรูปแบบของไฟล์ล็อก (Log File) ไม่ควรจะต้องมีคลาสที่เก็บโดยเฉพาะ ถ้าจะเก็บตรงไหนก็ไปเรียกใช้งาน Logger Library ตรงนั้นเลย แล้วไปกำหนดให้เขียนลงไฟล์ตรงการตั้งค่าโปรแกรม	ในขั้นตอนการพัฒนาสามารถใช้ไลบรารีคลาส Logger ในรูปแบบของไฟล์ล็อก เพื่อเก็บล็อกได้

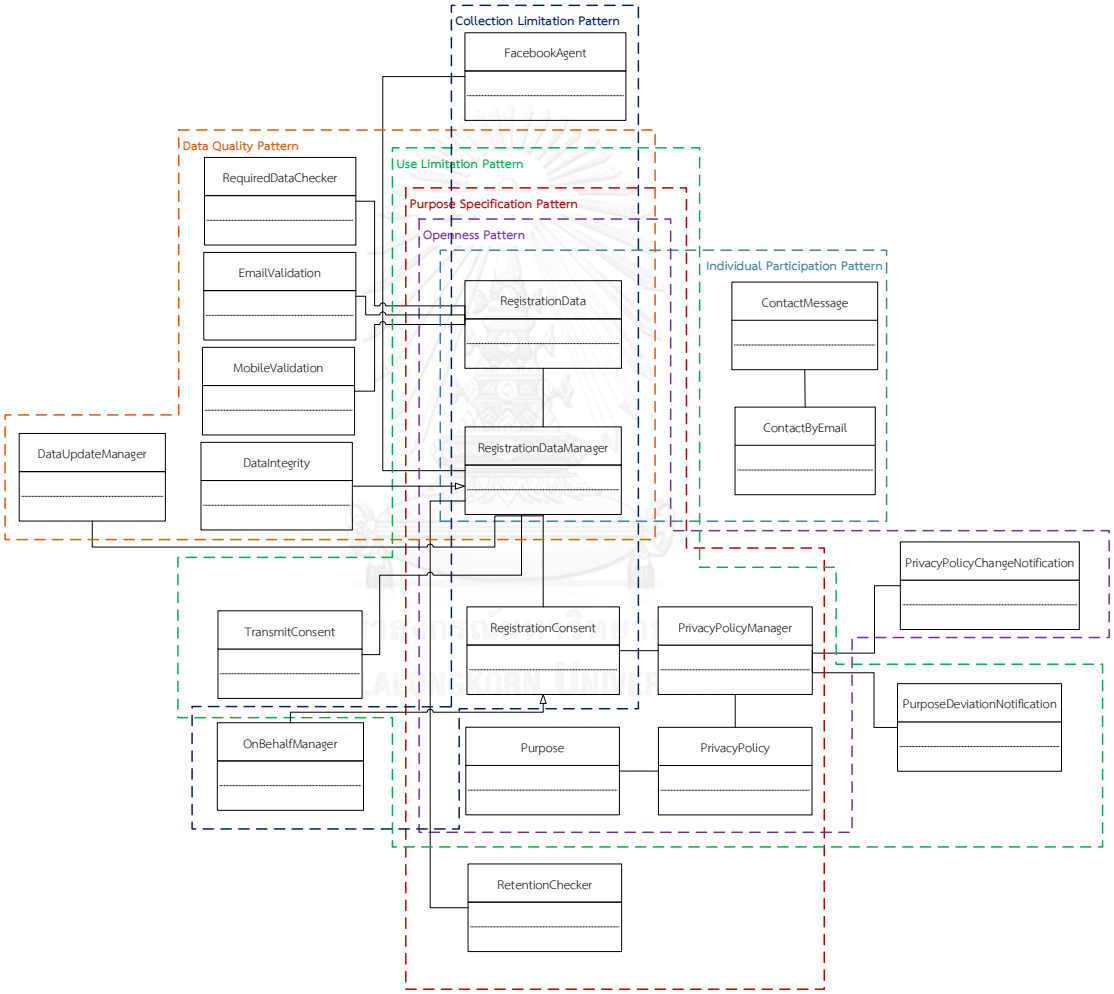
บทที่ 5
การนำแบบรูปการออกแบบไปประยุกต์ใช้

ในบทนี้จะแสดงตัวอย่างการนำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวไปใช้ เพื่อแสดงให้เห็นว่าแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอสามารถนำไปประยุกต์ใช้ได้อย่างครบถ้วนตามหลักการความเป็นส่วนตัว โดยผู้วิจัยได้ทำการพัฒนาระบบร้านค้าออนไลน์ ที่นำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอไปประยุกต์ใช้ โดยรายการความต้องการของระบบร้านค้าออนไลน์ที่พัฒนาแสดงดังตารางที่ 5.1 ดังนี้

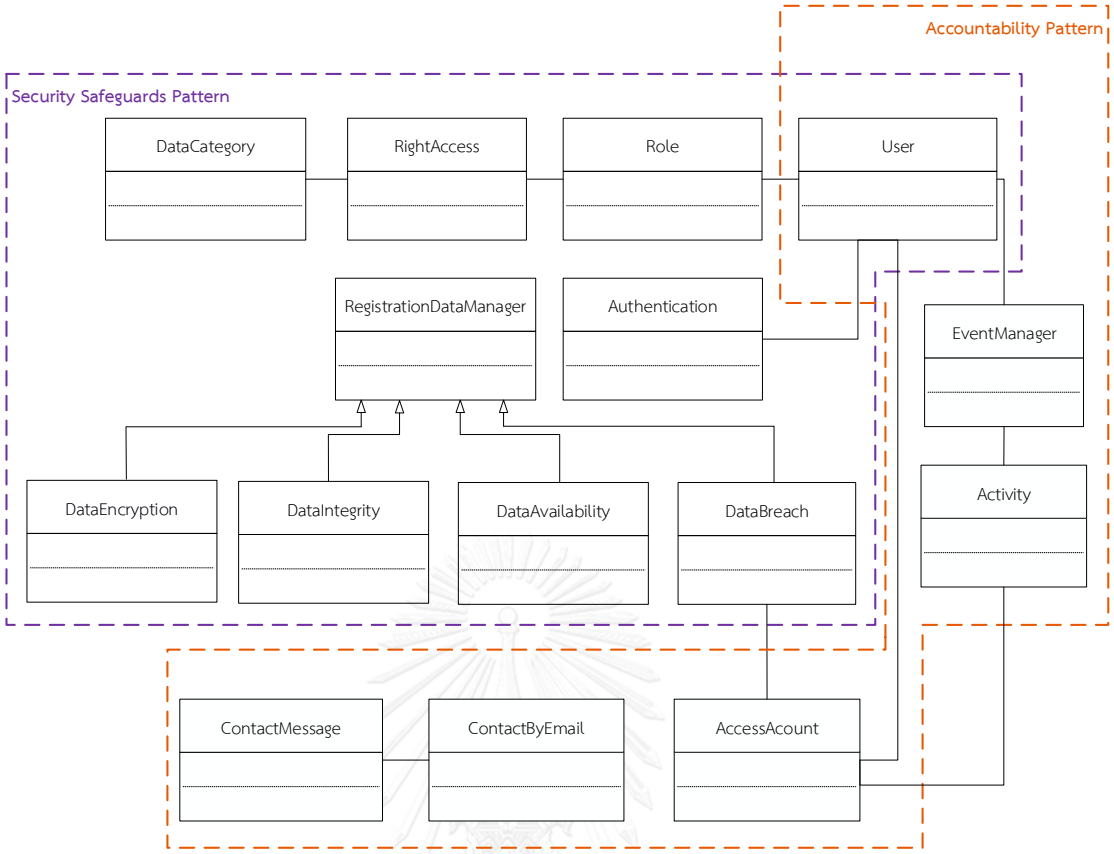
ตารางที่ 5.1 รายการความต้องการของระบบร้านค้าออนไลน์

รายการความต้องการของระบบร้านค้าออนไลน์	
รหัส	**รายการความต้องการ**
FR_01	เจ้าของระบบสามารถสร้างนโยบายความเป็นส่วนตัวของระบบได้
FR_02	เจ้าของระบบสามารถแก้ไขหรือเปลี่ยนแปลงนโยบายความเป็นส่วนตัวได้
FR_03	ระบบสามารถแจ้งให้ผู้ใช้งานระบบ (หรือ เจ้าของข้อมูล) ทราบ หากมีการแก้ไขหรือเปลี่ยนแปลงนโยบายความเป็นส่วนตัว
FR_04	ผู้ใช้งานสามารถสร้างบัญชีผู้ใช้ของระบบได้
FR_05	ผู้ใช้งานสามารถเลือกได้ว่ายินยอมหรือไม่ที่จะให้ระบบเก็บรวบรวมข้อมูลส่วนบุคคล
FR_06	ผู้ใช้งานสามารถแก้ไขข้อมูลส่วนบุคคลของตนเองได้
FR_07	ผู้ใช้งานสามารถซื้อขายสินค้าได้
FR_08	ระบบสามารถส่งข้อมูลส่วนบุคคลของผู้ใช้ไปยังบุคคลที่สามได้
FR_09	ผู้ใช้งานสามารถเลือกได้ว่าจะยินยอมหรือไม่ที่จะให้ระบบส่งข้อมูลของผู้ใช้ไปยังบุคคลที่สาม
FR_10	ผู้ใช้งานสามารถติดต่อมายังผู้ดูแลระบบได้ และผู้ดูแลระบบสามารถติดต่อกลับไปยังผู้ใช้งานได้
FR_11	ระบบสามารถบันทึกประวัติการเข้าถึงบัญชีผู้ใช้งาน (IP Address, OS, Location และ Browser)
FR_12	ระบบสามารถแจ้งให้ผู้ใช้งานที่เป็นเจ้าของบัญชีผู้ใช้ทราบ หากมีการเข้าถึงบัญชีผู้ใช้ที่สงสัยว่าอาจเป็นการละเมิดข้อมูลส่วนบุคคล
FR_13	ผู้ดูแลระบบสามารถกำหนดสิทธิการเข้าถึงข้อมูลในระบบได้ตามบทบาท (Role) ของผู้ใช้งาน
FR_14	ระบบสามารถบันทึกการกระทำที่เกิดขึ้นกับข้อมูลส่วนบุคคลของผู้ใช้งานได้ โดยบันทึกในรูปแบบของไฟล์ล็อก
FR_15	ผู้ดูแลระบบสามารถดูไฟล์ล็อกได้
FR_16	ผู้ดูแลระบบสามารถแจ้งให้ผู้ใช้งานทราบได้ว่าจะทำการตรวจสอบข้อมูลส่วนบุคคลของผู้ใช้งาน
FR_17	ผู้ดูแลระบบสามารถเพิ่ม แก้ไขหรือเปลี่ยนแปลงข้อมูลสินค้าในร้านได้

ในการพัฒนาระบบร้านค้าออนไลน์ ผู้วิจัยได้พัฒนาในรูปแบบของเว็บแอปพลิเคชัน โดยแบ่งการทำงานเป็นสองส่วนคือ ส่วนหน้าหรือส่วนติดต่อผู้ใช้ (Front-end) พัฒนาโดยใช้ ReactJS และส่วนหลังหรือระบบจัดการเว็บไซต์ (Back-end) พัฒนาโดยใช้ Python โดยในส่วนของการนำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวไปใช้นั้นจะอยู่ที่ส่วนหลัง ทำให้การนำแบบรูปมาประยุกต์ใช้จะไม่ได้นำส่วนของ Interface เข้ามาใช้ เนื่องจากในภาษา Python นั้นไม่มี Interface แผนภาพคลาสของระบบร้านค้าออนไลน์ที่นำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวไปใช้แสดงดังรูปที่ 5.1 และ 5.2



รูปที่ 5.1 แผนภาพคลาสของระบบร้านค้าออนไลน์ (ครอบคลุม 6 หลักการความเป็นส่วนตัว)

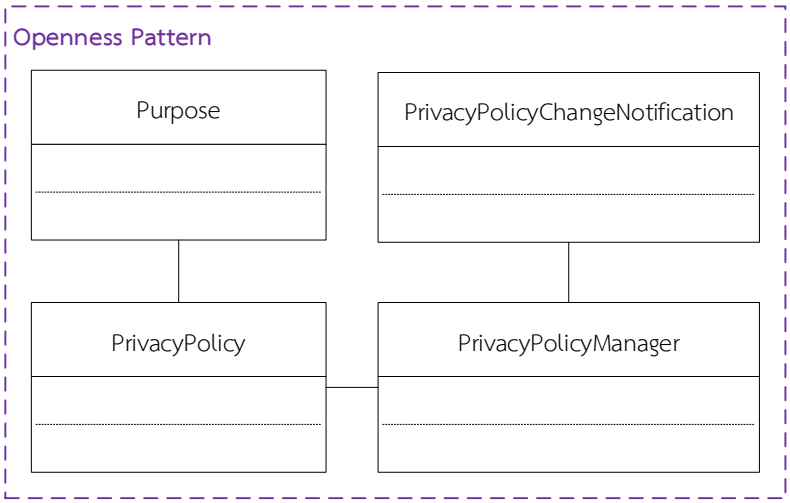


รูปที่ 5.2 แผนภาพคลาสของระบบร้านค้าออนไลน์ (ครอบคลุม 2 หลักการความเป็นส่วนตัว)

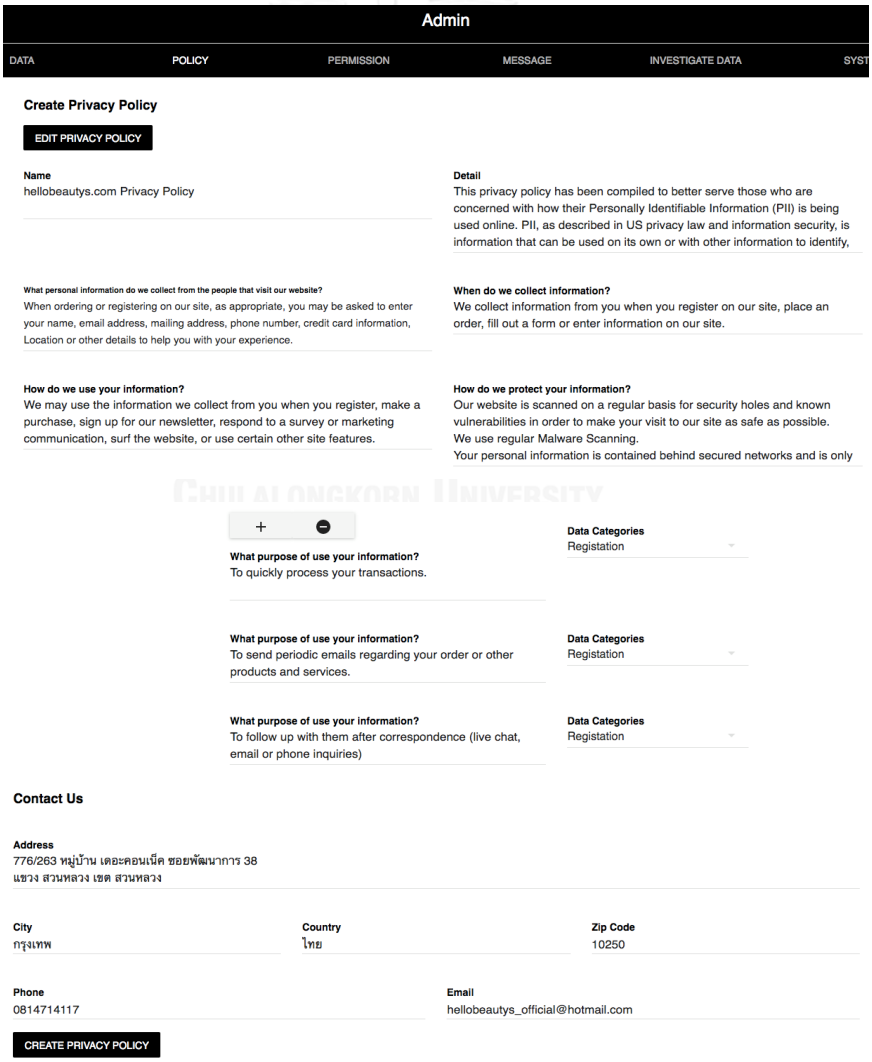
โดยตัวอย่างการทำงานของระบบกรณีศึกษาร้านค้าออนไลน์ (เว็บไซต์ HELLOBEAUTYS) ที่มีการนำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวไปใช้มีดังนี้

5.1 การสร้างนโยบายความเป็นส่วนตัว

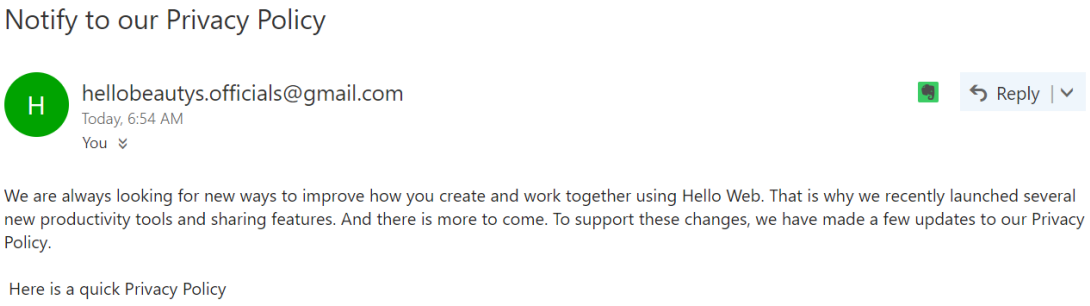
เมื่อทำการพัฒนาระบบผู้พัฒนา ผู้พัฒนาจะต้องมีการสร้างนโยบายความเป็นส่วนตัวขึ้นมาเพื่อแจ้งให้ผู้ใช้งานระบบ หรือเจ้าของข้อมูลทราบว่าระบบจะทำการเก็บรวบรวมข้อมูลอะไรบ้าง และข้อมูลเหล่านี้จะถูกนำไปใช้อย่างไร โดยในการสร้างนโยบายความเป็นส่วนตัวจะใช้แบบรูปการออกแบบสำหรับหลักการเปิดเผย แผนภาพคลาสของการสร้างนโยบายความเป็นส่วนตัวแสดงดังรูปที่ 5.3 ตัวอย่างโปรแกรมสำหรับการสร้างนโยบายความเป็นส่วนตัว แสดงดังรูปที่ 5.4 และเมื่อมีการเปลี่ยนแปลงนโยบายความเป็นส่วนตัวระบบจะต้องมีการแจ้งการเปลี่ยนแปลงนี้ให้กับผู้เป็นเจ้าของข้อมูลทราบแสดงดังรูปที่ 5.5



รูปที่ 5.3 แผนภาพคลาสของการสร้างนโยบายความเป็นส่วนตัว



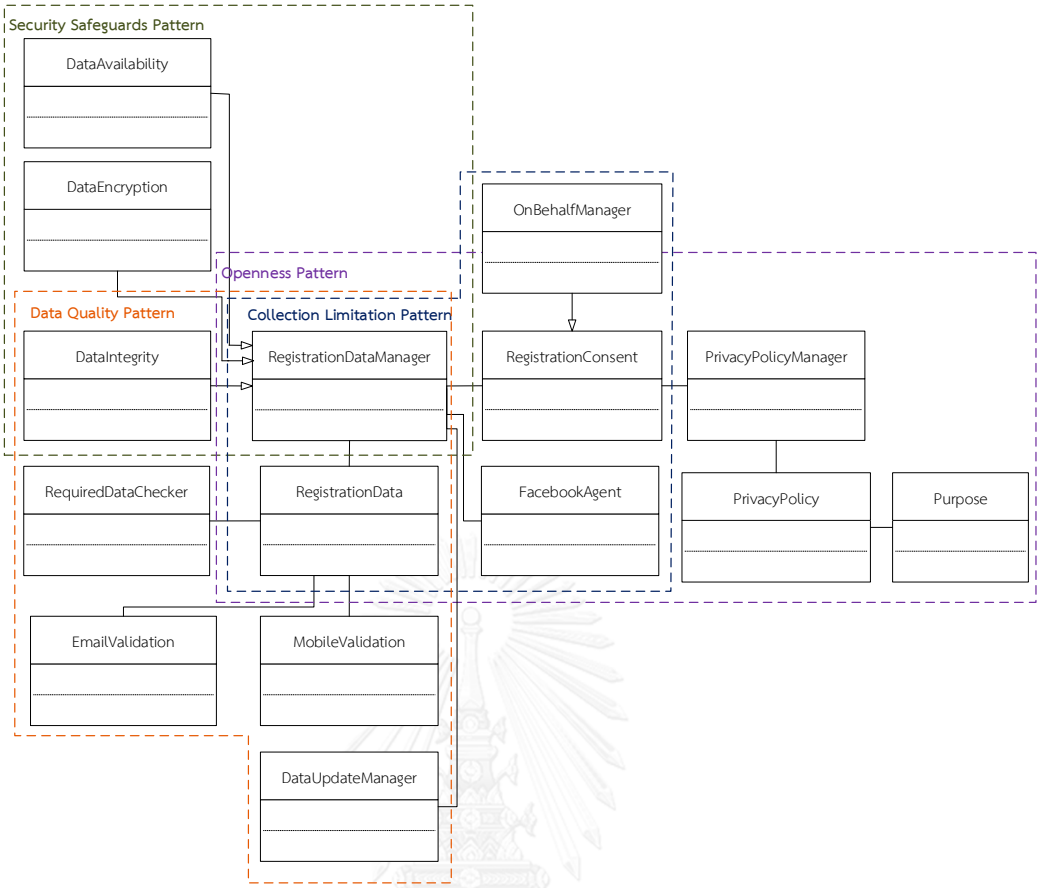
รูปที่ 5.4 หน้าสำหรับสร้าง และเปลี่ยนแปลงนโยบายความเป็นส่วนตัว



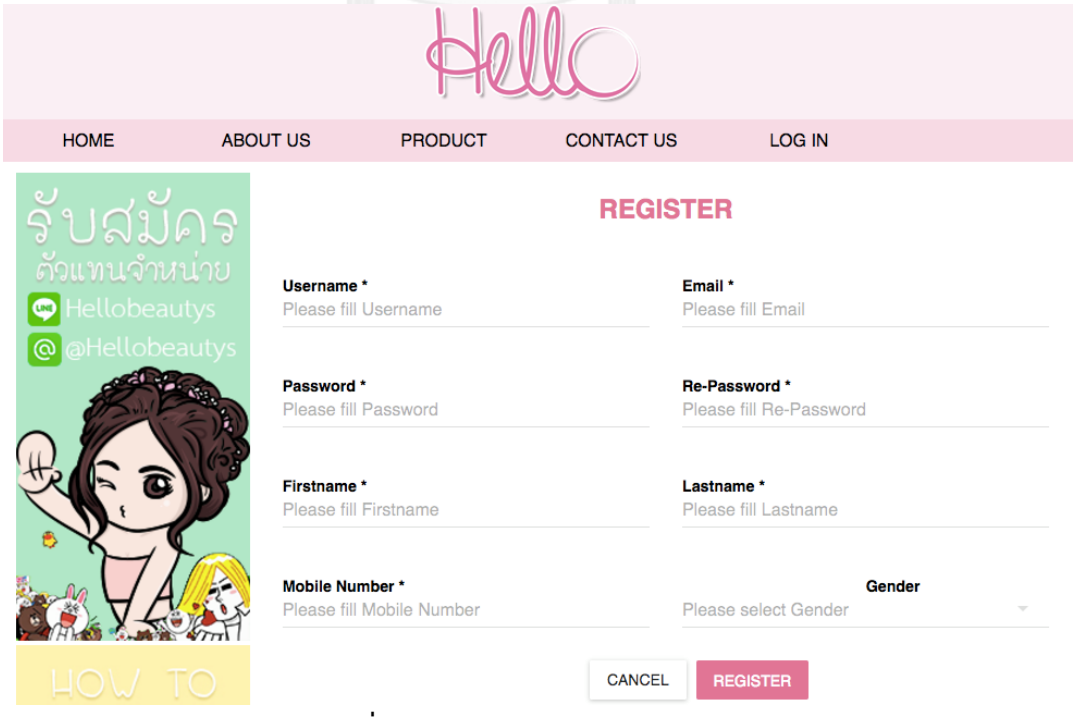
รูปที่ 5.5 เมื่อมีการเปลี่ยนแปลงนโยบายความเป็นส่วนตัวต้องมีการแจ้งให้ผู้เป็นเจ้าของข้อมูลทราบ

5.2 การสมัครสมาชิก

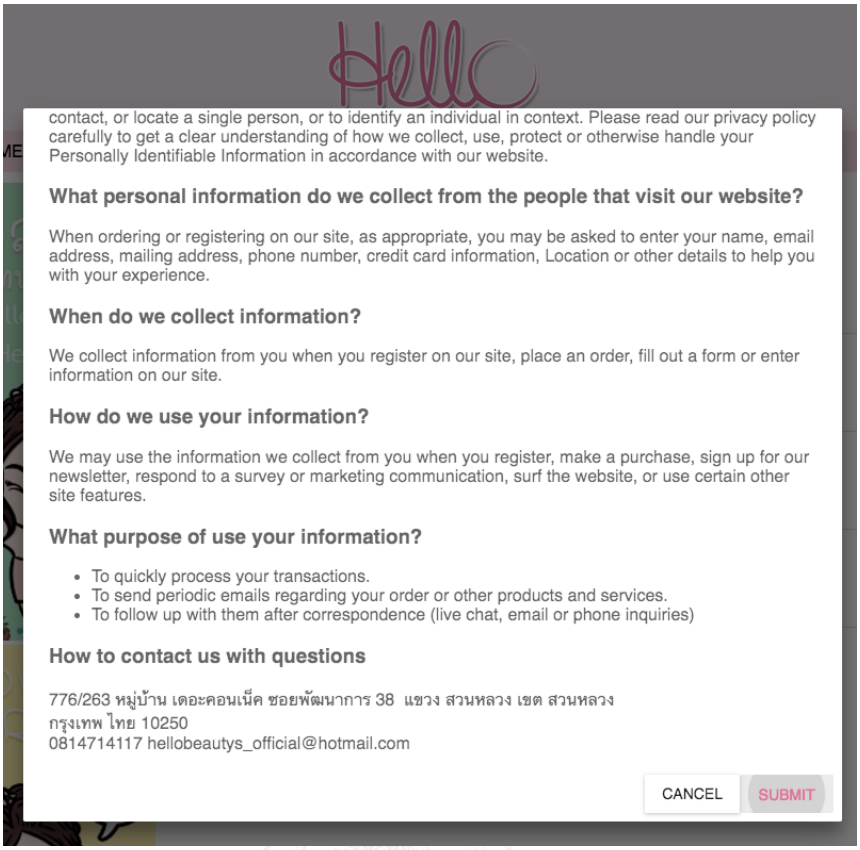
การพัฒนาในส่วนของการสมัครสมาชิกนั้น ผู้พัฒนาจะต้องนำแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล หลักคุณภาพของข้อมูล และหลักการเปิดเผยมาใช้ เพื่อให้ข้อมูลที่เก็บนั้นมีความถูกต้อง ครบถ้วน และเพื่อแจ้งให้ผู้ใช้ทราบว่าระบบจะทำการเก็บรวบรวมข้อมูลของผู้ใช้ แผนภาพคลาสของการสมัครสมาชิกแสดงดังรูปที่ 5.6 และตัวอย่างโปรแกรมสำหรับการสมัครสมาชิก แสดงดังรูปที่ 5.7 โดยเมื่อผู้ใช้กดสมัครสมาชิกระบบจะแสดงนโยบายความเป็นส่วนตัวให้ผู้ใช้ทราบว่าข้อมูลที่ถูกเก็บรวบรวมนั้นจะถูกนำไปใช้อย่างไร และหากผู้ใช้ยินยอมข้อมูลของผู้ใช้จะถูกบันทึกลงในระบบ หากไม่ยินยอมผู้ใช้จะไม่สามารถเป็นสมาชิกของระบบได้แสดงตัวอย่างดังรูปที่ 5.8 โดยข้อมูลที่มีความอ่อนไหวระบบจะทำการเข้ารหัสข้อมูลก่อนที่จะบันทึกข้อมูลลงในระบบแสดงตัวอย่างดังรูปที่ 5.9 และเมื่อข้อมูลถูกเก็บเข้าลงฐานข้อมูลของระบบ ระบบจะมีการเตรียมข้อมูลไว้สำหรับตรวจสอบความความคงสภาพของข้อมูล (Data Integrity) โดยในตัวอย่างนี้จะใช้วิธีการ Checksumming ซึ่งแสดงตัวอย่างดังรูปที่ 5.10 และมีการสำรองข้อมูลไว้เพื่อป้องกันกรณีข้อมูลถูกทำลาย หรือสูญหาย และเพื่อให้ข้อมูลมีความพร้อมในการใช้งานอยู่เสมอ (Data Availability) โดยในตัวอย่างนี้จะใช้วิธีการบันทึกข้อมูลเป็นไฟล์ เมื่อมีข้อมูลเข้ามาในระบบเพื่อนำไฟล์นี้มากู้คืนข้อมูลหากเกิดปัญหาขึ้น ซึ่งแสดงตัวอย่างดังรูปที่ 5.11



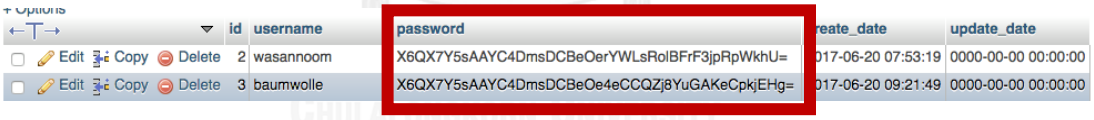
รูปที่ 5.6 แผนภาพคลาสของการสมัครสมาชิก



รูปที่ 5.7 หน้าสำหรับการสมัครสมาชิก



รูปที่ 5.8 เมื่อผู้ใช้กดสมัครสมาชิกระบบจะทำการแจ้งนโยบายความเป็นส่วนตัวให้ผู้ใช้ทราบ

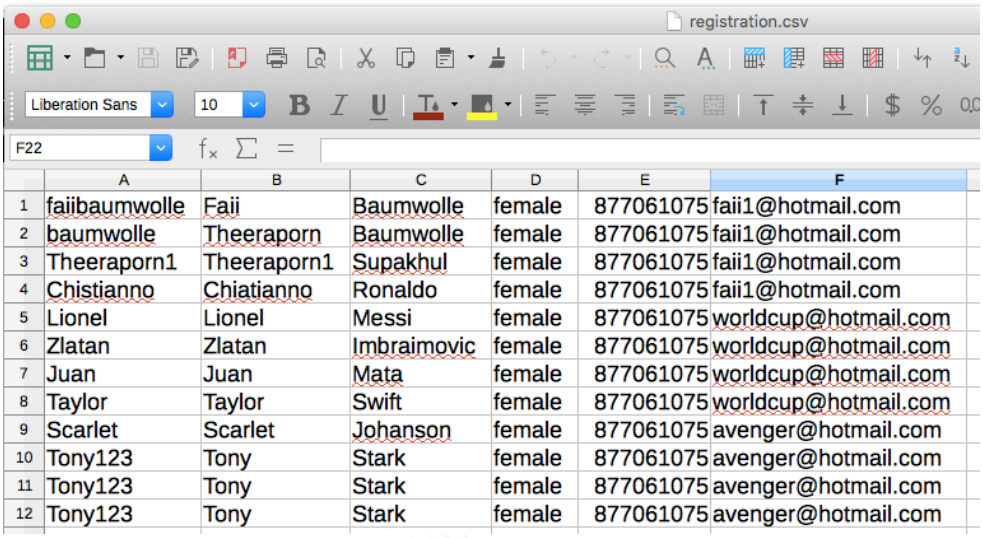


	id	username	password	reate_date	update_date
Edit Copy Delete	2	wasannoom	X6QX7Y5sAAYC4DmsDCBeOerYWLsRolBFrF3jpRpWkhU=	017-06-20 07:53:19	0000-00-00 00:00:00
Edit Copy Delete	3	baumwolle	X6QX7Y5sAAYC4DmsDCBeOe4eCCQZj8YuGAKeCpkjEHg=	017-06-20 09:21:49	0000-00-00 00:00:00

รูปที่ 5.9 การเข้ารหัสข้อมูลที่มีความอ่อนไหว

id	firstname	lastname	gender	email	tel	email_cs	tel_cs
1	Wasan	Chubuatong	male	wasannoom@hotmail.com	087706107	c274fec470ea7e14767df0c221671a8f	c9b3e73bc65ebeae26cd79048917ed18
3	Theeraporn	Suphakul	female	baumwolle.faii@gmail.com	098689884	8bb83866080a15b6c6b6068c01e27c37	efd9905ec8391eae675bb159b9dcdb8e

รูปที่ 5.10 การเตรียมข้อมูลไว้สำหรับตรวจสอบความความคงสภาพของข้อมูล



registration.csv

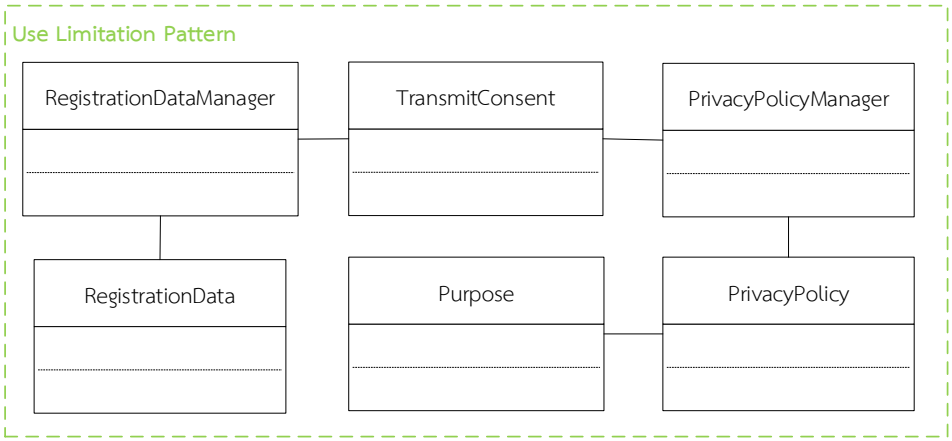
Liberation Sans | 10 | F22

	A	B	C	D	E	F
1	faiibaumwolle	Faii	Baumwolle	female	877061075	faii1@hotmail.com
2	baumwolle	Theeraporn	Baumwolle	female	877061075	faii1@hotmail.com
3	Theeraporn1	Theeraporn1	Supakhul	female	877061075	faii1@hotmail.com
4	Chistianno	Chiatianno	Ronaldo	female	877061075	faii1@hotmail.com
5	Lionel	Lionel	Messi	female	877061075	worldcup@hotmail.com
6	Zlatan	Zlatan	Imbraimovic	female	877061075	worldcup@hotmail.com
7	Juan	Juan	Mata	female	877061075	worldcup@hotmail.com
8	Taylor	Taylor	Swift	female	877061075	worldcup@hotmail.com
9	Scarlet	Scarlet	Johanson	female	877061075	avenger@hotmail.com
10	Tony123	Tony	Stark	female	877061075	avenger@hotmail.com
11	Tony123	Tony	Stark	female	877061075	avenger@hotmail.com
12	Tony123	Tony	Stark	female	877061075	avenger@hotmail.com

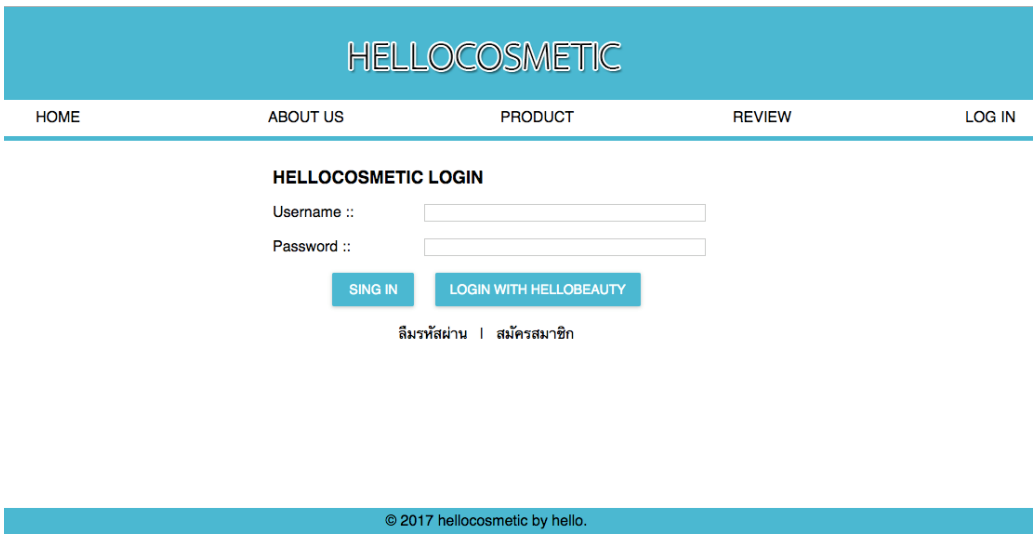
รูปที่ 5.11 การเตรียมข้อมูลให้ข้อมูลมีความพร้อมในการใช้งานอยู่เสมอ

5.3 การส่งข้อมูลไปยังบุคคลที่สาม

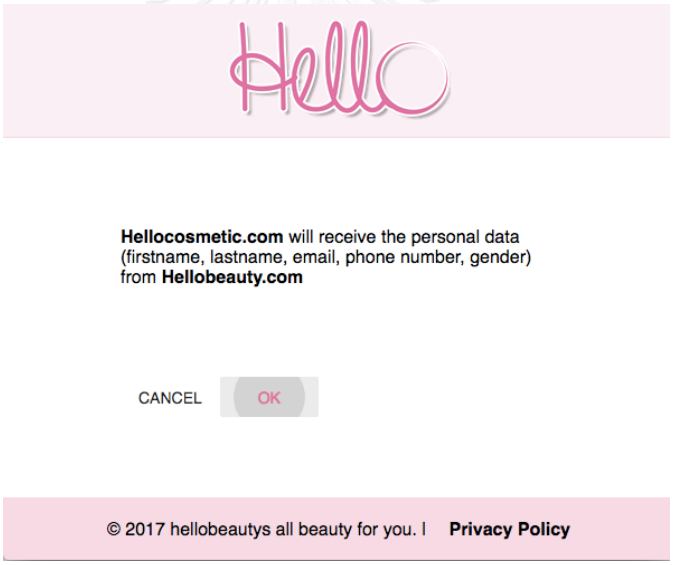
การพัฒนาในส่วนของการส่งข้อมูลไปยังบุคคลที่สาม เมื่อมีบุคคลที่สามมาขอใช้ข้อมูลส่วนบุคคลในระบบ ผู้พัฒนาจะต้องมีการแจ้งให้กับเจ้าของข้อมูลทราบ และได้รับอนุญาตจากเจ้าของข้อมูลก่อนจึงจะสามารถส่งข้อมูลไปยังบุคคลที่สามได้ โดยจะประยุกต์ใช้แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้งาน แผนภาพคลาสของการส่งข้อมูลไปยังบุคคลที่สามแสดงดังรูปที่ 5.12 ตัวอย่างหน้าเว็บไซต์ของบุคคลที่สามแสดงดังรูปที่ 5.13 และเมื่อผู้ใช้ต้องการสมัครสมาชิกของเว็บบุคคลที่สาม (HELLOCOSMETIC) โดยการ “LOGIN WITH HELLOBEAUTYS” คือสมัครสมาชิกโดยใช้ข้อมูลจากเว็บไซต์ HELLOBEAUTYS เว็บไซต์ HELLOBEAUTYS จะทำการแจ้งให้ผู้ใช้ทราบว่าข้อมูลส่วนบุคคลของผู้ใช้จะถูกบันทึกลงไปยังเว็บไซต์ HELLOCOSMETIC แสดงดังรูปที่ 5.14 หากผู้ใช้ยินยอมเว็บไซต์ HELLOBEAUTYS จะส่งข้อมูลของผู้ใช้ให้กับ HELLOCOSMETIC



รูปที่ 5.12 แผนภาพคลาสของการส่งข้อมูลไปยังบุคคลที่สาม



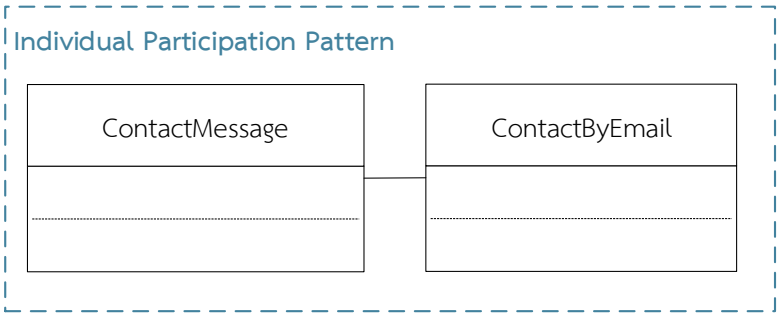
รูปที่ 5.13 หน้าเว็บไซต์ของบุคคลที่สาม (HELLOCOSMETIC) ที่จะดึงข้อมูลจากระบบไปใช้



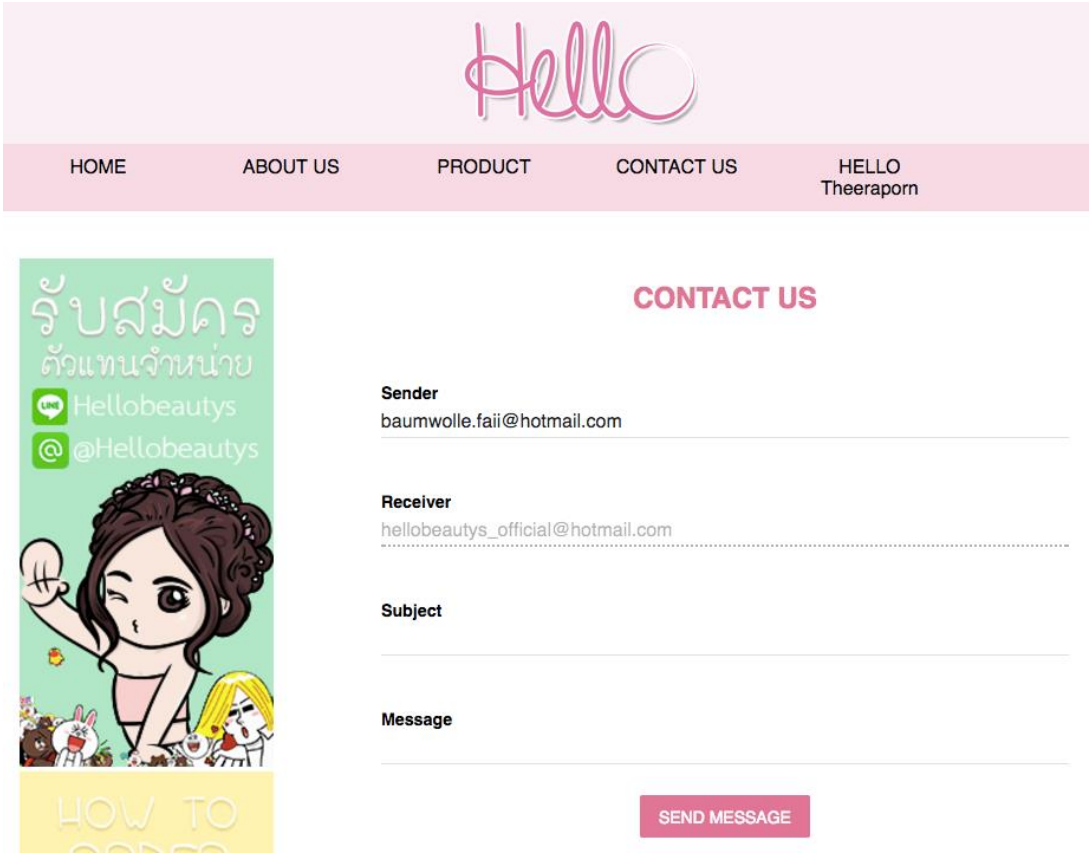
รูปที่ 5.14 ระบบต้องแจ้งให้ผู้ใช้ทราบว่าจะส่งข้อมูลส่วนบุคคลของผู้ใช้ไปยังบุคคลที่สาม

5.4 การติดต่อไปยังผู้ดูแลระบบ

การพัฒนาการติดต่อไปยังผู้ดูแลระบบเป็นการประยุกต์ใช้แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล เพื่อให้ผู้ใช้งานหรือเจ้าของข้อมูลสอบถามเกี่ยวกับข้อมูลส่วนบุคคลที่เกี่ยวข้องกับตนเอง หรือสอบถามข้อมูลความเป็นส่วนตัวกับผู้ดูแลระบบได้ แผนภาพคลาสของการติดต่อไปยังผู้ดูแลระบบแสดงดังรูปที่ 5.15 ตัวอย่างโปรแกรมการติดต่อไปยังผู้ดูและระบบแสดงดังรูปที่ 5.16



รูปที่ 5.15 แผนภาพคลาสของการติดต่อไปยังผู้ดูแลระบบ

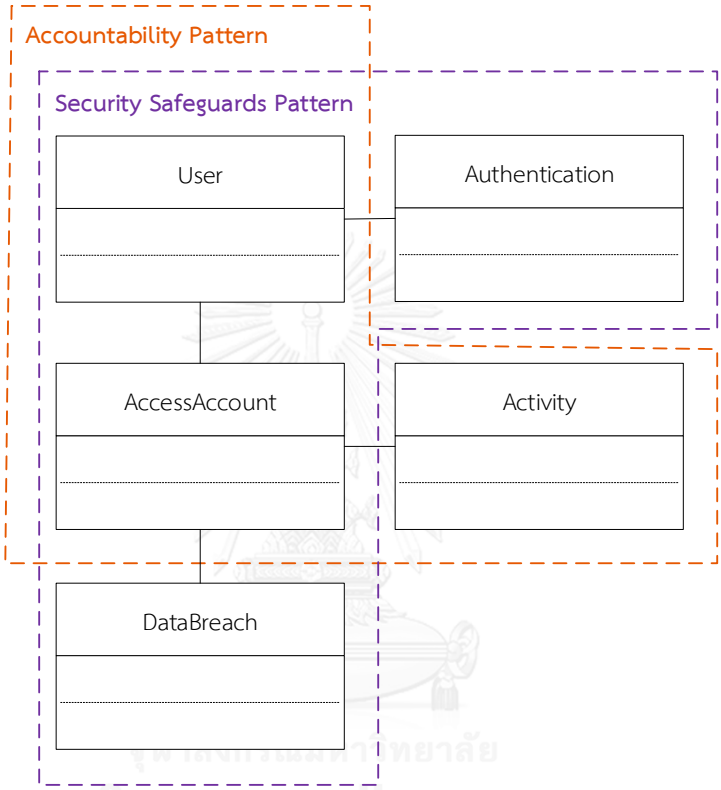


รูปที่ 5.16 หน้าแสดงการติดต่อไปยังผู้ดูแลระบบ

5.5 การบันทึกประวัติการเข้าถึงบัญชีผู้ใช้

การบันทึกประวัติการเข้าถึงบัญชีผู้ใช้ เพื่อทำการตรวจสอบว่ามีพฤติกรรมการเข้าถึงบัญชีผู้ใช้ที่น่าสงสัย หรืออาจจะเป็นการละเมิดข้อมูลส่วนบุคคลหรือไม่ โดยจะประยุกต์ใช้แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล และหลักความรับผิดชอบ แผนภาพคลาสของการเก็บประวัติการเข้าถึงบัญชีผู้ใช้แสดงดังรูปที่ 5.17 ตัวอย่างการบันทึกประวัติการเข้าถึงบัญชีผู้ใช้แสดงดัง

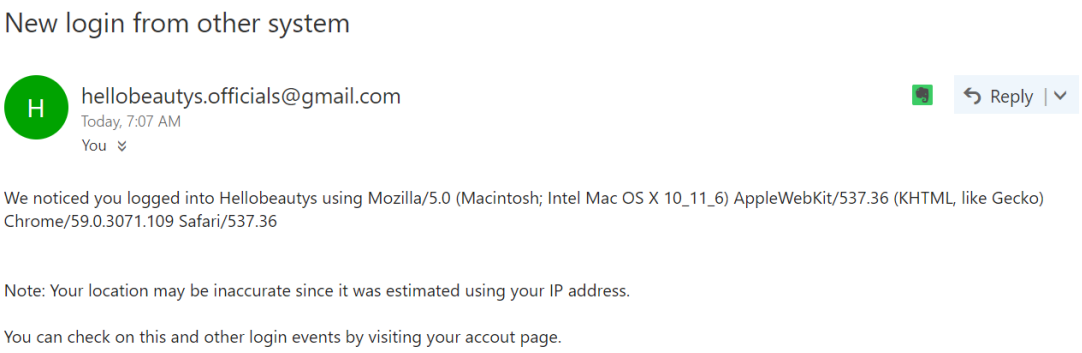
รูปที่ 5.18 โดยจะเป็นการบันทึกลงฐานข้อมูล เพื่อนำมาใช้ตรวจสอบว่าการเข้าถึงนี้มีพฤติกรรมที่น่าจะเป็นการละเมิดข้อมูลส่วนบุคคลหรือไม่ หากมีพฤติกรรมที่อาจจะเป็นการละเมิดข้อมูลส่วนบุคคลระบบจะทำการส่งเมล์แจ้งให้กับผู้เป็นเจ้าของข้อมูลทราบ แสดงดังรูปที่ 5.19 โดยเป็นการแจ้งเตือนเมื่อเกิดการเข้าถึงบัญชีผู้ใช้จากเครื่องคอมพิวเตอร์ที่ต่างไปจากเดิม



รูปที่ 5.17 แผนภาพคลาสของการเก็บประวัติการเข้าถึงบัญชีผู้ใช้

id	ipaddress	location	browser	os	firstaccess	lastaccess	username
56	localhost:5002	Bangkok	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) Ap...	Windows 7	2017-07-05 22:14:44	2017-07-05 22:55:03	faiibaumwolle
57	localhost:5002	Bangkok	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) Ap...	MacIntel	2017-07-05 22:15:44	2017-07-05 22:55:03	faiibaumwolle

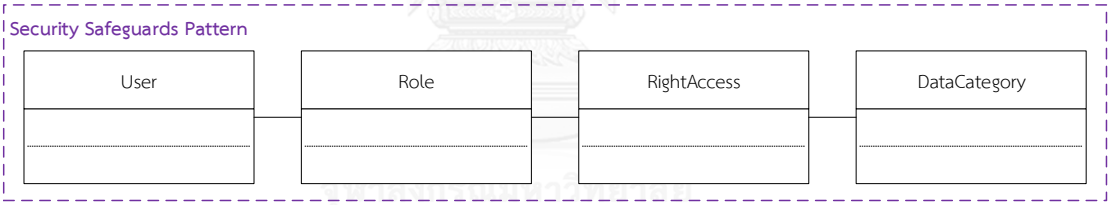
รูปที่ 5.18 การเก็บข้อมูลประวัติการเข้าถึงบัญชีผู้ใช้



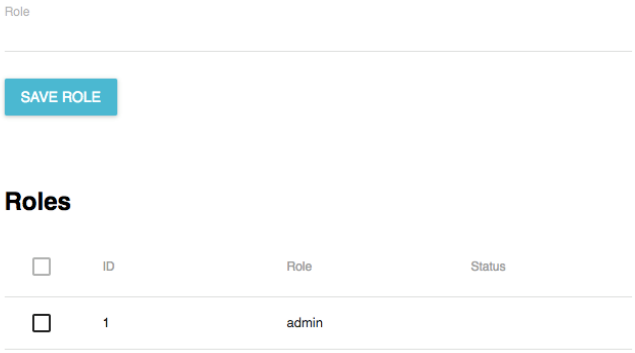
รูปที่ 5.19 การแจ้งเตือนเมื่อพบว่ามีพฤติกรรมที่อาจจะเป็นการละเมิดข้อมูลส่วนบุคคล

5.6 การกำหนดสิทธิการเข้าถึง

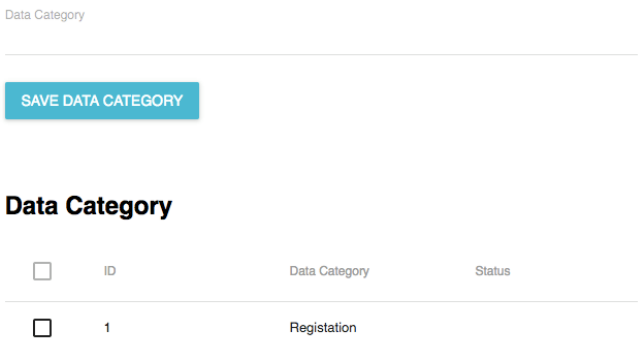
การกำหนดสิทธิการเข้าถึง เพื่อระบุว่าในระบบนี้ผู้ใช้ที่มีบทบาทนี้ สามารถเข้าถึงข้อมูล เพื่อทำอะไรได้บ้าง แผนภาพคลาสของการกำหนดสิทธิการเข้าถึงข้อมูลแสดงดังรูปที่ 5.20 โดยในตัวอย่างนี้กำหนดให้ผู้ดูแลระบบจะต้องมากำหนดว่าในระบบนี้มีบทบาทอะไรบ้างแสดงดังรูปที่ 5.21 และมีข้อมูลประเภทใดบ้างในระบบแสดงดังรูปที่ 5.22 และสามารถทำอะไรกับข้อมูลประเภทนั้นได้บ้างแสดงดังรูปที่ 5.23 จากนั้นจึงจะทำการกำหนดสิทธิการเข้าถึงข้อมูลแสดงดังรูปที่ 5.24



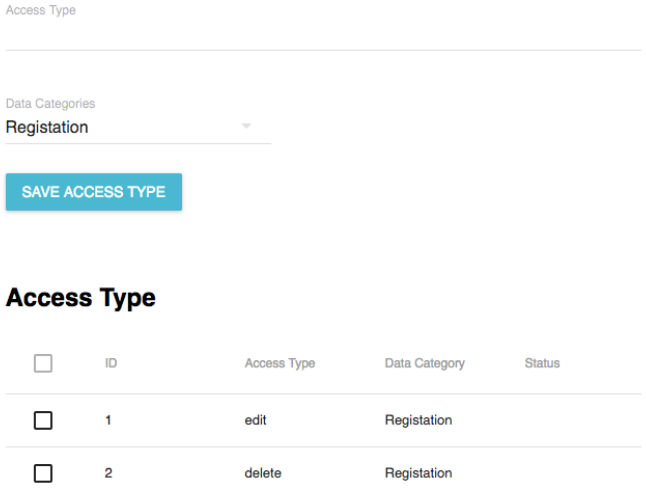
รูปที่ 5.20 แผนภาพคลาสของการกำหนดสิทธิการเข้าถึงข้อมูล



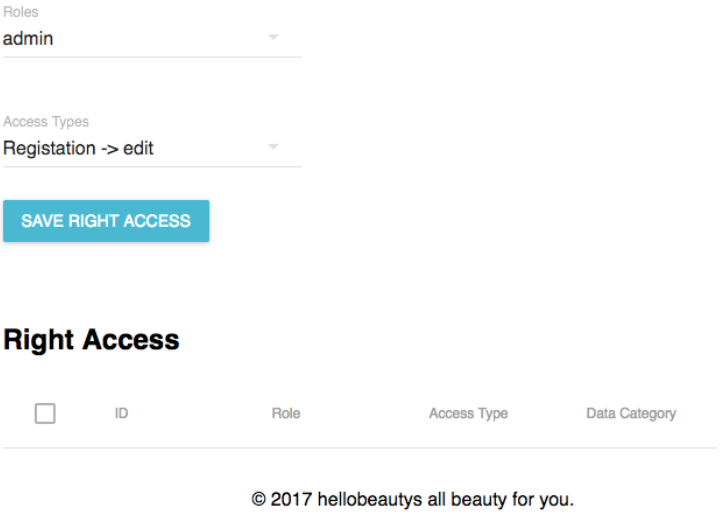
รูปที่ 5.21 การกำหนดบทบาทของผู้ใช้ในระบบ



รูปที่ 5.22 การกำหนดประเภทข้อมูลในระบบ



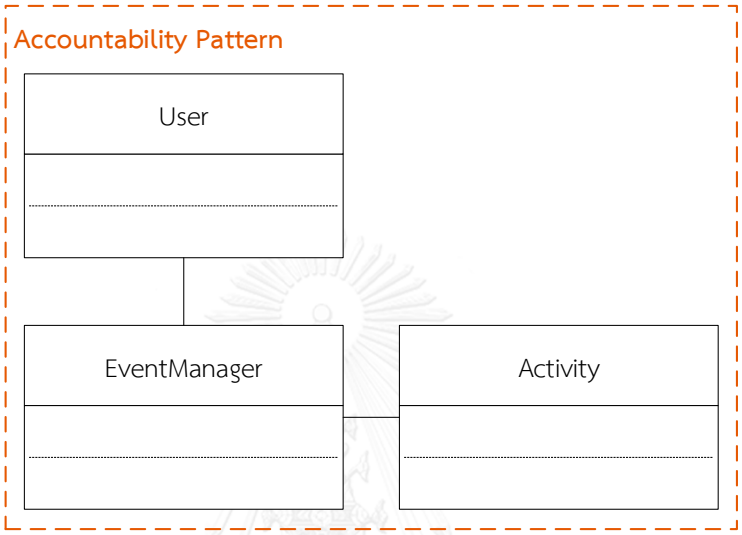
รูปที่ 5.23 การกำหนดการประเภทการเข้าถึงข้อมูล



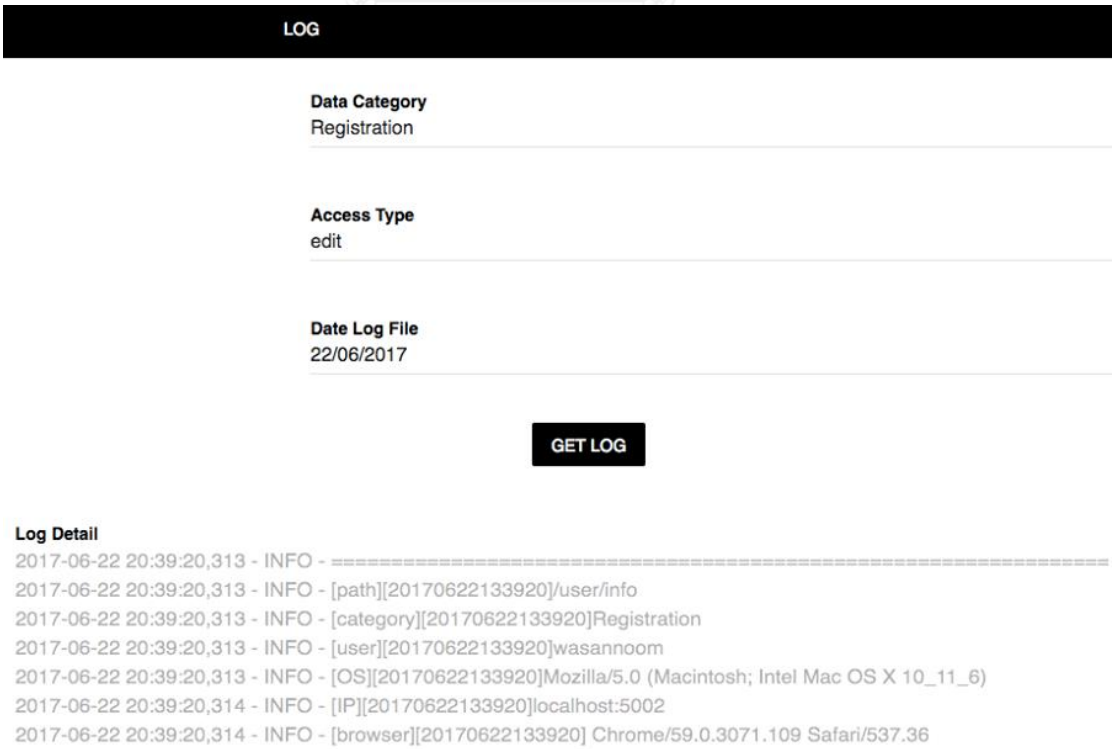
รูปที่ 5.24 การกำหนดสิทธิการเข้าถึงข้อมูล

5.7 การเก็บไฟล์ล็อก

การเก็บไฟล์ล็อก ทำเพื่อนำไฟล์มาใช้ตรวจสอบในภายหลังว่าเกิดกิจกรรมใดขึ้นบ้างในระบบ แผนภาพคลาสของการเก็บไฟล์ล็อกแสดงดังรูปที่ 5.25 และตัวอย่างไฟล์ล็อกที่เก็บแสดงดังรูปที่ 5.26 โดยผู้ดูแลระบบสามารถเข้ามาเลือกได้ว่าจะดูไฟล์ล็อกที่เกิดจากกิจกรรมใด ข้อมูลประเภทใด และวันที่เท่าไร



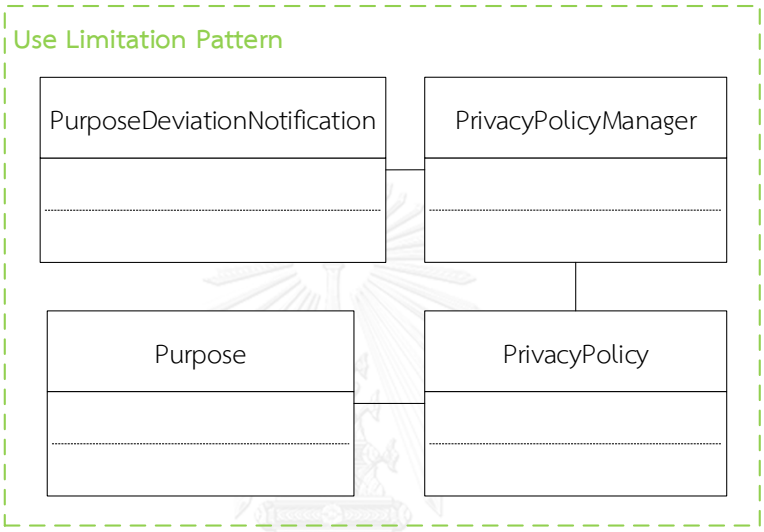
รูปที่ 5.25 แผนภาพคลาสของการเก็บไฟล์ล็อก



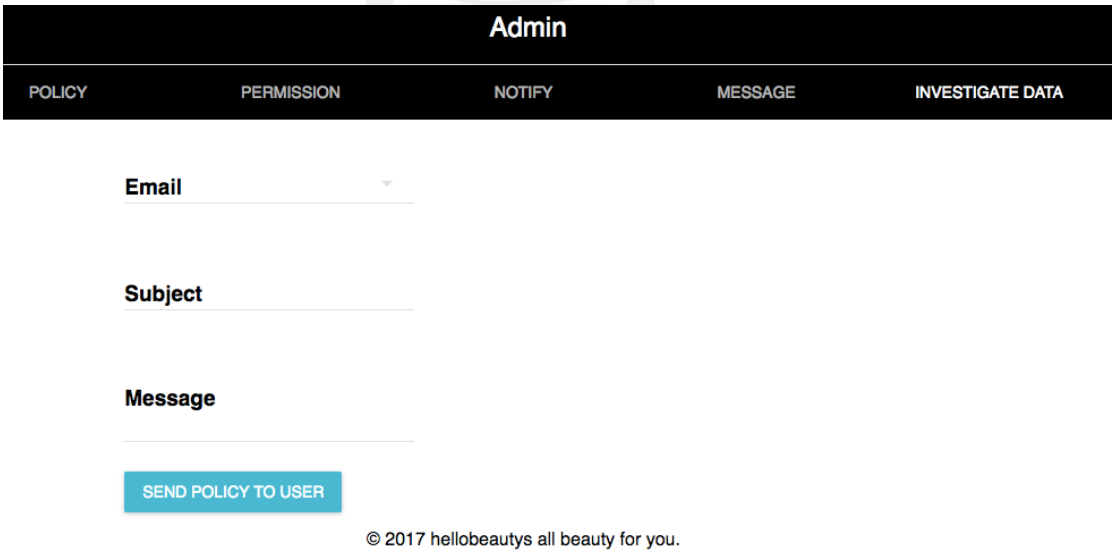
รูปที่ 5.26 ไฟล์ล็อกที่เก็บแสดงกิจกรรมที่เกิดขึ้นในระบบ

5.8 การขอเข้าถึงข้อมูลส่วนบุคคลของผู้ใช้เพื่อตรวจสอบ

การขอเข้าถึงข้อมูลส่วนบุคคลของผู้ใช้เพื่อตรวจสอบ ในกรณีมีการร้องขอเพื่อตรวจสอบข้อเท็จจริงเกี่ยวกับข้อมูลส่วนบุคคลของผู้ใช้งาน ผู้ดูแลระบบจะต้องมีการแจ้งให้เจ้าของข้อมูลทราบว่าจะมีการตรวจสอบข้อมูลเกิดขึ้น แผนภาพคลาสของการขอเข้าถึงข้อมูลส่วนบุคคลของผู้ใช้เพื่อตรวจสอบแสดงดังรูปที่ 5.27 และหน้าจอการส่งข้อความแจ้งผู้ใช้เพื่อขอเข้าถึงข้อมูลส่วนบุคคลแสดงดังรูปที่ 5.28



รูปที่ 5.27 แผนภาพคลาสของการขอเข้าถึงข้อมูลส่วนบุคคลของผู้ใช้



รูปที่ 5.28 หน้าจอการส่งข้อความแจ้งผู้ใช้เพื่อขอเข้าถึงข้อมูลส่วนบุคคล

บทที่ 6
สรุปผลการวิจัย

6.1 ผลสรุปของงานวิจัย

งานวิจัยนี้นำเสนอแบบรูปการออกแบบสำหรับความเป็นส่วนตัว ตามหลักการความเป็นส่วนตัวของโออีซีดี โดยที่องค์ประกอบของแบบรูปจะปรับมาจากหนังสือ “Design patterns: elements of reusable object-oriented software” [4] ตามที่กล่าวไว้ในหัวข้อที่ 2.1.3 และมีการแสดงแบบจำลองซอฟต์แวร์ยูเอ็มแอลคือ แผนภาพกิจกรรม แผนภาพคลาส และแผนภาพลำดับ เพื่อให้ง่ายต่อการนำแบบรูปไปประยุกต์ใช้ แบบรูปการออกแบบสำหรับความเป็นส่วนตัวมีดังนี้

1) แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล
2) แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล
3) แบบรูปการออกแบบสำหรับหลักการกำหนดวัตถุประสงค์
4) แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล
5) แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล
6) แบบรูปการออกแบบสำหรับหลักการเปิดเผย
7) แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล
8) แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ

เมื่อพัฒนาแบบรูปการออกแบบสำหรับความเป็นส่วนตัวแล้วผู้วิจัยได้ทำการตรวจสอบความครบถ้วนของแบบรูปการออกแบบที่นำเสนอ โดยผลลัพธ์ที่ได้จากการตรวจสอบผู้วิจัยจะนำมาพิจารณาว่ายังมีสาระสำคัญของหลักการความเป็นส่วนตัวที่ไม่ปรากฏอยู่ในแบบรูปการออกแบบหรือไม่ ซึ่งผลลัพธ์ที่ได้จะแสดงให้เห็นว่าสาระสำคัญของหลักการความเป็นส่วนตัวใดบ้างที่ยังไม่ถูกนำมาพิจารณาในแบบรูปการออกแบบ ซึ่งบางสาระสำคัญของหลักการความเป็นส่วนตัวที่ไม่มีปรากฏในหัวข้อ “Behavior and Structure” ผู้ตรวจสอบจะเข้าใจว่าไม่มีปรากฏในแบบรูป จึงต้องอธิบายให้ผู้ตรวจสอบทราบว่าสาระสำคัญนี้อยู่ในส่วนใดของแบบรูป

ต่อมาผู้วิจัยได้นำแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอมาประยุกต์ใช้กับระบบร้านค้าออนไลน์ เพื่อแสดงให้เห็นว่าแบบรูปการออกแบบสำหรับความเป็นส่วนตัวที่นำเสนอสามารถนำไปประยุกต์ใช้ได้อย่างครบถ้วนตามหลักการความเป็นส่วนตัว

6.2 ข้อจำกัดของงานวิจัย

1) แบบรูปการออกแบบสำหรับความเป็นส่วนตัวครอบคลุมหลักการความเป็นส่วนตัวของโออีซีดี แต่มีรายละเอียดบางส่วนของหลักการความเป็นส่วนตัวที่ไม่สามารถนำมาออกแบบในรูปแบบของแบบจำลองยูเอ็มแอลได้โดยตรง ดังนั้นผู้วิจัยจึงกำหนดรายการความต้องการที่สนับสนุนหลักการนั้น แล้วออกแบบแบบจำลองยูเอ็มแอลตามรายการความต้องการนั้นร่วมกับการเสนอการแนะนำวิธีการปฏิบัติ (Implementation) ซึ่งจะช่วยสนับสนุนการพัฒนาซอฟต์แวร์ให้ครอบคลุมหลักการความเป็นส่วนตัวของโออีซีดี
2) การแสดงตัวอย่างการประยุกต์ใช้แบบรูปการออกแบบสำหรับความเป็นส่วนตัว เป็นการแสดงตัวอย่างระบบเพียงระบบเดียวการออกแบบจึงยังไม่ครอบคลุมทุกกรณีที่แบบรูปสามารถครอบคลุมได้ เช่น จากกรณีตัวอย่างระบบร้านค้าออนไลน์ไม่มีความจำเป็นต้องเก็บข้อมูลผู้ให้ความยินยอมแทนตัวเจ้าของข้อมูล จึงไม่ได้ใช้งาน Interface OnBehalfManager และไม่มีงานที่ต้องดึงข้อมูลส่วนบุคคลแบบอัตโนมัติเป็นระยะจึงไม่ได้ใช้งาน Interface DataUpdateManager

6.3 งานวิจัยในอนาคต

1. เพิ่มเติมการแปลงจากแบบจำลองยูเอ็มแอล ให้เป็นโค้ดที่สามารถนำมาพัฒนาเพิ่มเติมและนำไปประยุกต์ใช้งานจริงได้
2. เพิ่มการตรวจสอบแบบรูปการออกแบบสำหรับความเป็นส่วนตัวโดยให้ผู้ตรวจสอบนำแบบรูปที่นำเสนอมาออกแบบตามโจทย์ที่กำหนด เพื่อตรวจสอบว่าแบบรูปที่นำเสนอนั้นยังมีส่วนใดที่ต้องพัฒนา หรือปรับปรุงอีกในการนำไปใช้งานจริง
3. เพิ่มการประเมินคุณภาพของแบบรูปการออกแบบสำหรับความเป็นส่วนตัวในด้านอื่น เช่น ทำการทดลองว่าการมีแบบรูปนี้ช่วยให้การพัฒนาแอปพลิเคชันที่ต้องการรักษาความเป็นส่วนตัวของข้อมูลส่วนบุคคล สามารถทำได้สะดวกรวดเร็วขึ้น และได้ซอฟต์แวร์ที่สมบูรณ์ขึ้นเพียงใดในแง่ความสามารถในการรักษาความเป็นส่วนตัว เมื่อเทียบกับการพัฒนาโดยไม่ใช้แบบรูป
4. เพิ่มตัวอย่างการประยุกต์ใช้แบบรูปการออกแบบสำหรับความเป็นส่วนตัว เพื่อให้เห็นการนำแบบรูปไปประยุกต์ใช้ในโดเมนของระบบที่แตกต่างกันออกไป

6.4 ผลงานตีพิมพ์จากงานวิทยานิพนธ์

งานวิทยานิพนธ์นี้มีผลงานการตีพิมพ์เรื่อง "Development of Privacy Design Patterns Based on Privacy Principles and UML" ซึ่งได้รับการนำเสนอและตีพิมพ์ในงาน "18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2017)" ซึ่งจัดขึ้นที่เมืองคานาซาว่า จังหวัดอิชิคาว่า ประเทศญี่ปุ่น ระหว่างวันที่ 26 - 28 มิถุนายน 2560

รายการอ้างอิง

[1] The Organization for Economic Co-operation and Development (OECD). *The OECD Privacy Framework*. OECD Publishing.

[2] Sasha Romanosky, et al., *Privacy Patterns for Online Interactions*, in *Proceedings of the 2006 conference on Pattern languages of programs*. 2006, ACM: Portland, Oregon, USA. p. 21-23.

[3] Porekar, J., A. Jerman-Blazic, and T. Klobucar. *Towards Organizational Privacy Patterns*. in *Second International Conference on the Digital Society*. 2008.

[4] Gamma, E., et al., *Design patterns: elements of reusable object-oriented software*. 1995: Addison-Wesley Longman Publishing Co., Inc. 395.

[5] Object Management Group. *What is UML [Online]*. [cited 2016 October 20]; Available from: http://www.omg.org/UML/what-is-uml.htm.

[6] Sa-Adaem, K. and Y. Teng-Amnuay, *Assessing Privacy Protection in Alumni Service.* International Journal of Computer and Electrical Engineering, 2013.

[7] Hoepman, J.-H., *Privacy Design Strategies*, in *ICT Systems Security and Privacy Protection: 29th IFIP TC 11 International Conference, SEC 2014, Marrakech, Morocco, June 2-4, 2014. Proceedings*, N. Cuppens-Boulahia, et al., Editors. 2014, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 446-459.

[8] Graf, C., et al., *A Pattern Collection for Privacy Enhancing Technology*, in *PATTERNS 2010 : The Second International Conferences on Pervasive Patterns and Applications*. January 2010. p. 72-77.

[9] Coen-Porisini, A., et al., *A conceptual model for privacy policies*, in *Proceedings of the 11th IASTED International Conference on Software Engineering and Applications*. 2007, ACTA Press: Cambridge, Massachusetts. p. 570-577.

[10] Basso, T., et al., *Towards a UML Profile for Privacy-Aware Applications*, in *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*. 2015. p. 371-378.

[11] Wang, J. and Z. Wang, *A Survey on Personal Data Cloud.* The Scientific World Journal 2014. 2014: p. 1-13.

[12] Gopalan Sivathanu, C.P.W., and Erez Zadok. *Ensuring Data Integrity in Storage: Techniques and Applications*. [cited 2017 June 20]; Available from: https://www.fsl.cs.sunysb.edu/docs/integrity-storagess05/integrity.html.

[13] Schumacher, M., et al., *Security Patterns Integrating Security and Systems Engineering*. 2006: Library of Congress Cataloging-in-Publication Data. 565.

[14] Stretch, J. *Symmetric Encryption, Asymmetric Encryption, and Hashing*. [cited 2017 June 20]; Available from: http://packetlife.net/blog/2010/nov/23/symmetric-asymmetric-encryption-hashing/.

ภาคผนวก

ภาคผนวก ก.
รายละเอียดหลักการความเป็นส่วนตัวของโออีซีดี

ก. 1 หลักข้อจำกัดในการเก็บรวบรวมข้อมูล

Paragraph 7: Collection Limitation Principle

As an introductory comment on the principles set out in Paragraphs 7 to 14 of the Guidelines it should be pointed out that these principles are interrelated and partly overlapping. Thus, the distinctions between different activities and stages involved in the processing of data which are assumed in the principles, are somewhat artificial and it is essential that the principles are treated together and studied as a whole. Paragraph 7 deals with two issues, viz. (a) limits to the collection of data which, because of the manner in which they are to be processed, their nature, the context in which they are to be used or other circumstances, are regarded as specially sensitive; and (b) requirements concerning data collection methods. Different views are frequently put forward with respect to the first issue. It could be argued that it is both possible and desirable to enumerate types or categories of data which are per se sensitive and the collection of which should be restricted or even prohibited. There are precedents in European legislation to this effect (race, religious beliefs, criminal records, for instance). On the other hand, it may be held that no data are intrinsically "private" or "sensitive" but may become so in view of their context and use. This view is reflected, for example, in the privacy legislation of the United States.

The Expert Group discussed a number of sensitivity criteria, such as the risk of discrimination, but has not found it possible to define any set of data which are universally regarded as sensitive. Consequently, Paragraph 7 merely contains a general statement that there should be limits to the collection of personal data. For one thing, this represents an affirmative recommendation to lawmakers to decide on limits which would put an end to the indiscriminate collection of personal data. The nature of the limits is not spelt out but it is understood that the limits may relate to:

- data quality aspects (i.e. that it should be possible to derive information of sufficiently high quality from the data collected, that data should be collected in a proper information framework, etc.);
- limits associated with the purpose of the processing of data (i.e. that only certain categories of data ought to be collected and, possibly, that data collection should be restricted to the minimum necessary to fulfil the specified purpose);
- "earmarking" of specially sensitive data according to traditions and attitudes in each Member country;

- limits to data collection activities of certain data controllers;
- civil rights concerns.

The second part of Paragraph 7 (data collection methods) is directed against practices which involve, for instance, the use of hidden data registration devices such as tape recorders, or deceiving data subjects to make them supply information. The knowledge or consent of the data subject is as a rule essential, knowledge being the minimum requirement. On the other hand, consent cannot always be imposed, for practical reasons. In addition, Paragraph 7 contains a reminder ("where appropriate") that there are situations where for practical or policy reasons the data subject's knowledge or consent cannot be considered necessary. Criminal investigation activities and the routine up-dating of mailing lists may be mentioned as examples. Finally, Paragraph 7 does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.

ก. 2 หลักคุณภาพของข้อมูล

Paragraph 8: Data Quality Principle

Requirements that data be relevant can be viewed in different ways. In fact, some members of the Expert Group hesitated as to whether such requirements actually fitted into the framework of privacy protection. The conclusion of the Group was to the effect, however, that data should be related to the purpose for which they are to be used. For instance, data concerning opinions may easily be misleading if they are used for purposes to which they bear no relation, and the same is true of evaluative data. Paragraph 8 also deals with accuracy, completeness and up-to-dateness which are all important elements of the data quality concept. The requirements in this respect are linked to the purposes of data, i.e. they are not intended to be more far-reaching than is necessary for the purposes for which the data are used. Thus, historical data may often have to be collected or retained; cases in point are social research, involving so-called longitudinal studies of developments in society, historical research, and the activities of archives. The "purpose test" will often involve the problem of whether or not harm can be caused to data subjects because of lack of accuracy, completeness and up-dating.

ก. 3 หลักการกำหนดวัตถุประสงค์

Paragraph 9: Purpose Specification Principle

The Purpose Specification Principle is closely associated with the two surrounding principles, i.e. the Data Quality Principle and the Use Limitation Principle. Basically, Paragraph 9 implies that before, and in any case not later than at the time data collection it should be possible to identify the purposes for which these data are to be used, and that later changes of purposes should likewise be specified. Such specification of purposes can be made in a number of alternative or complementary ways, e.g. by public declarations, information to data subjects, legislation, administrative decrees, and licences provided by supervisory bodies. According to Paragraphs 9 and 10, new purposes should not be introduced arbitrarily; freedom to make changes should imply compatibility with the original purposes. Finally, when data no longer serve a purpose, and if it is practicable, it may be necessary to have them destroyed (erased) or given an anonymous form. The reason is that control over data may be lost when data are no longer of interest; this may lead to risks of theft, unauthorised copying or the like.

ก. 4 หลักข้อจำกัดในการใช้ข้อมูล

Paragraph 10: Use Limitation Principle

This paragraph deals with uses of different kinds, including disclosure, which involve deviations from specified purposes. For instance, data may be transmitted from one computer to another where they can be used for unauthorised purposes without being inspected and thus disclosed in the proper sense of the word. As a rule the initially or subsequently specified purposes should be decisive for the uses to which data can be put. Paragraph 10 foresees two general exceptions to this principle: the consent of the data subject (or his representative – see Paragraph 52 above) and the authority of law (including, for example, licences granted by supervisory bodies). For instance, it may be provided that data which have been collected for purposes of administrative decision-making may be made available for research, statistics and social planning.

ก. 5 หลักการรักษาความมั่นคงของข้อมูล

Paragraph 11: Security Safeguards Principle

Security and privacy issues are not identical. However, limitations on data use and disclosure should be reinforced by security safeguards. Such safeguards include physical measures (locked doors and identification cards, for instance), organisational measures (such as authority levels with regard to access to data) and, particularly in computer systems, informational measures (such as enciphering and threat monitoring of unusual activities and responses to them). It should be emphasised that the category of organisational measures includes obligations for data processing personnel to maintain confidentiality. Paragraph 11 has a broad coverage. The cases mentioned in the provision are to some extent overlapping (e.g. access/ disclosure). "Loss" of data encompasses such cases as accidental erasure of data, destruction of data storage media (and thus destruction of data) and theft of data storage media. "Modified" should be construed to cover unauthorised input of data, and "use" to cover unauthorised copying.

ก. 6 หลักการเปิดเผย

Paragraph 12: Openness Principle

The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle (Paragraph 13); for the latter principle to be effective, it must be possible in practice to acquire information about the collection, storage or use of personal data. Regular information from data controllers on a voluntary basis, publication in official registers of descriptions of activities concerned with the processing of personal data, and registration with public bodies are some, though not all, of the ways by which this may be brought about. The reference to means which are "readily available" implies that individuals should be able to obtain information without unreasonable effort as to time, advance knowledge, travelling, and so forth, and without unreasonable cost.

ก. 7 หลักการมีส่วนร่วมของบุคคล

Paragraph 13: Individual Participation Principle

The right of individuals to access and challenge personal data is generally regarded as perhaps the most important privacy protection safeguard. This view is shared by the Expert Group which, although aware that the right to access and challenge cannot be absolute, has chosen to express it in clear and fairly specific language. With respect to the individual sub-paragraphs, the following explanations are called for:

The right to access should as a rule be simple to exercise. This may mean, among other things, that it should be part of the day-to-day activities of the data controller or his representative and should not involve any legal process or similar measures. In some cases it may be appropriate to provide for intermediate access to data; for example, in the medical area a medical practitioner can serve as a go-between. In some countries supervisory organs, such as data inspection authorities, may provide similar services. The requirement that data be communicated within reasonable time may be satisfied in different ways. For instance, a data controller who provides information to data subjects at regular intervals may be exempted from obligations to respond at once to individual requests. Normally, the time is to be counted from the receipt of a request. Its length may vary to some extent from one situation to another depending on circumstances such as the nature of the data processing activity. Communication of such data "in a reasonable manner" means, among other things, that problems of geographical distance should be given due attention. Moreover, if intervals are prescribed between the times when requests for access must be met, such intervals should be reasonable. The extent to which data subjects should be able to obtain copies of data relating to them is a matter of implementation which must be left to the decision of each Member country.

The right to reasons in Paragraph 13(c) is narrow in the sense that it is limited to situations where requests for information have been refused. A broadening of this right to include reasons for adverse decisions in general, based on the use of personal data, met with sympathy in the Expert Group. However, on final consideration a right of this kind was thought to be too broad for insertion in the privacy framework constituted by the Guidelines. This is not to say that a right to reasons for adverse decisions may not be appropriate, e.g. in order to inform and alert a subject to his rights so that he can exercise them effectively.

The right to challenge in 13(c) and (d) is broad in scope and includes first instance challenges to data controllers as well as subsequent challenges in courts, administrative bodies, professional organs or other institutions according to domestic rules of procedure (compare with Paragraph 19 of the Guidelines). The right to challenge does not imply that the data subject can decide what remedy or relief is available (rectification, annotation that data are in dispute, etc.): such matters will be decided by domestic law and legal procedures. Generally speaking, the criteria which decide the outcome of a challenge are those which are stated elsewhere in the Guidelines.

ก. 8 หลักความรับผิดชอบ

Paragraph 14: Accountability Principle

The data controller decides about data and data processing activities. It is for his benefit that the processing of data is carried out. Accordingly. it is essential that under domestic law accountability for complying with privacy protection rules and decisions should be placed on the data controller who should not be relieved of this obligation merely because the processing of data is carried out on his behalf by another party, such as a service bureau. On the other hand, nothing in the Guidelines prevents service bureaux personnel, "dependent users" (see paragraph 40) and others from also being held accountable. For instance, sanctions against breaches of confidentiality obligations may be directed against all parties entrusted with the handling of personal information (*cf.* Paragraph 19 of the Guidelines). Accountability under Paragraph 14 refers to accountability supported by legal sanctions, as well as to accountability established by codes of conduct, for instance.

ภาคผนวก ข.
รายละเอียดย่อยของหลักการความเป็นส่วนตัวของโออีซีดี

ข. 1 รายละเอียดย่อยของหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

ตารางที่ ข.1 รายละเอียดย่อยของหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)	There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject. (รายละเอียดเพิ่มเติมดูที่ภาคผนวก)	1. There should be limits to the collection of personal data, where limits may relate to the following. 1.1 The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly. 1.2 The data collection should be restricted to the minimum necessary to fulfil the specified purpose. 1.3 Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection. 1.4 There should be limits to data collection activities of the data controller. 1.5 Civil rights of the data subject should be of concern in the data collection.

ตารางที่ ข.1 รายละเอียดย่อยของหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักข้อจำกัดในการเก็บรวบรวมข้อมูล (Collection Limitation Principle)		2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.
		3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.
		4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.

ข. 2 รายละเอียดย่อยของหลักคุณภาพของข้อมูล

ตารางที่ ข.2 รายละเอียดย่อยของหลักคุณภาพของข้อมูล

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักคุณภาพของข้อมูล (Data Quality Principle)	Personal data should be relevant to the purposes for which they are to be used, and, to the extent necessary for those purposes, should be accurate, complete and kept up-to-date.	1. Personal data should be relevant to the purposes for which they are to be used.
		2. Personal data, to the extent necessary for their purposes, should be accurate.
		3. Personal data, to the extent necessary for their purposes, should be complete.
		4. Personal data, to the extent necessary for their purposes, should be kept up-to-date.

ข. 3 รายละเอียดย่อยของหลักข้อกำหนดวัตถุประสงค์

ตารางที่ ข.3 รายละเอียดย่อยของหลักข้อกำหนดวัตถุประสงค์

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักข้อกำหนดวัตถุประสงค์ (Purpose Specification Principle)	The purposes for which personal data are collected should be specified not later than at the time of data collection and the subsequent use limited to the fulfilment of those purposes or such others as are not incompatible with those purposes and as are specified on each occasion of change of purpose.	1. The purposes for which personal data are collected should be specified not later than at the time of data collection.
		2. The subsequent use of personal data should be limited to the fulfilment of the purposes specified prior to or at the time of collection.
		3. The subsequent use of personal data may be for purposes other than the original purposes, but such purposes should be specified as change.
		4. Later changes of purposes should be specified.
		5. On each occasion of change of purpose, the new purposes should not be incompatible with the original purposes.
		6. Specification of purpose should be made by public declarations or information to the data subject.
		7. When data no longer serve a purpose, and if it is practicable, it may be necessary to have them destroyed (erased) or given an anonymous form.
		8. The Purpose Specification Principle is closely associated with the Data Quality Principle and the Use Limitation Principle.

ข. 4 รายละเอียดย่อยของหลักข้อจำกัดในการใช้ข้อมูล

ตารางที่ ข.4 รายละเอียดย่อยของหลักข้อจำกัดในการใช้ข้อมูล

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักข้อจำกัดในการใช้ข้อมูล (Use Limitation Principle)	Personal data should not be disclosed, made available, or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle except: a) with the consent of the data subject; or b) by the authority of law.	1. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except with the consent of the data subject or his representative.
		2. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except by the authority of law or licences granted by supervisory bodies.

ข. 5 รายละเอียดย่อยของหลักการรักษาความมั่นคงของข้อมูล

ตารางที่ ข.5 รายละเอียดย่อยของหลักการรักษาความมั่นคงของข้อมูล

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Principle)	Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorized access, destruction, use, modification, or disclosure of data.	1. Personal data should be protected by reasonable security safeguards against unauthorized disclosure of data, based on informational measures such as enciphering.
		2. Personal data should be protected by reasonable security safeguards against unauthorized access, destruction, use, and modification of data, based on organizational measures such as authority levels with regard to access to data.

ตารางที่ ข.5 รายละเอียดย่อยของหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักการรักษาความมั่นคงของข้อมูล (Security Safeguards Principle)		3. Personal data should be protected by reasonable security safeguards against loss or destruction and modification of data, both accidentally and deliberately, where data integrity would be affected.
		4. Personal data should be protected by reasonable security safeguards against loss of access by which data availability would be affected.
		5. Personal data should be protected by reasonable security safeguards against threats of data breach, based on informational measures such as monitoring of unusual activities and responses to them.
		6. Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees' security training, up-to-date security safeguards).

ข. 6 รายละเอียดย่อยของหลักการเปิดเผย

ตารางที่ ข.6 รายละเอียดย่อยของหลักการเปิดเผย

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักการเปิดเผย (Openness Principle)	There should be a general policy of openness about developments, practices and policies with respect to personal data. Means should be readily available of establishing the existence and nature of personal data, and the main purposes of their use, as well as the identity and usual residence of the data controller.	1. There should be a general policy of openness about developments, practices and policies with respect to personal data.
		2. Means should be readily available of establishing the existence and nature of personal data.
		3. Means should be readily available of establishing the main purposes of the use of personal data.
		4. Means should be readily available of establishing the identity and usual residence of the data controller.
		5. “Readily available” implies that data subjects should be able to obtain information without unreasonable effort as to time, advance knowledge, travelling, and so forth, and without unreasonable cost.
		6. Regular information from the data controller regarding the privacy policy should be published.
		7. The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle.

ข. 7 รายละเอียดย่อยของหลักการมีส่วนร่วมของบุคคล

ตารางที่ ข.7 รายละเอียดย่อยของหลักการมีส่วนร่วมของบุคคล

<table>
<tr><th>หลักการความเป็นส่วนตัว</th><th>รายละเอียด</th><th>รายละเอียดย่อย</th></tr>
<tr><td rowspan="9">หลักการมีส่วนร่วมของบุคคล (Individual Participation Principle)</td><td rowspan="9">An individual should have the right:
a) to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him;
b) to have communicated to him, data relating to him
i. within a reasonable time;
ii. at a charge, if any, that is not excessive;
iii. in a reasonable manner; and
iv. in a form that is readily intelligible to him;
c) to be given reasons if a request made under subparagraphs (a) and (b) is denied, and to be able to challenge such denial; and
d) to challenge data relating to him and, if the challenge is successful, to have the data erased, rectified, completed or amended.</td><td>1. A data subject can obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him/her (or his/her personal data).</td></tr>
<tr><td>2. A data subject should have the right to access to his/her personal data within a reasonable time when a request to access is made.</td></tr>
<tr><td>3. A data subject should have the right to access to his/her personal data at a reasonable cost.</td></tr>
<tr><td>4. A data subject should have the right to access to his/her personal data in a reasonable manner.</td></tr>
<tr><td>5. A data subject should have the right to access to his/her personal data in a form that is readily intelligible.</td></tr>
<tr><td>6. A data subject should have the right to reasons for, and challenge, adverse decisions when requests for confirmation or access are denied.</td></tr>
<tr><td>7. A data subject should have the right to challenge his/her personal data.</td></tr>
<tr><td>8. A data controller should erase, rectify, complete, or amend the data subject’s personal data if the challenge is successful.</td></tr>
</table>

ข. 8 รายละเอียดย่อยของหลักความรับผิดชอบ

ตารางที่ ข.8 รายละเอียดย่อยของหลักความรับผิดชอบ

หลักการความเป็นส่วนตัว	รายละเอียด	รายละเอียดย่อย
หลักความรับผิดชอบ (Accountability Principle)	A data controller should be accountable for complying with measures which give effect to the principles stated above.	1. A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles.
		2. A data controller should have in place a privacy management program that is tailored to its operations and integrated into its governance structure.
		3. A data controller should provide ongoing monitoring, periodic assessment of data breach, and response to incidents.
		4. There should be a method for the data subject to inquire about issues related to privacy of personal data.
		5. A data controller should provide evidence of its privacy management program to the privacy enforcement authorities.
		6. A data controller should provide notice to the privacy enforcement authorities and data subject in the event of significant data breach.

ภาคผนวก ค.

รายการความต้องการของแต่ละแบบรูปการออกแบบสำหรับความเป็นส่วนตัว

ค. 1 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

ตารางที่ ค.1 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

Pattern Name	**Collection Limitation Pattern**
Privacy Principle	Collection Limitation Principle
Pattern Requirement	
ID	**Requirement**
PT-CL01	The system should be able to collect personal data that are necessary for use.
PT-CL02	The system should be able to record the information about the source of personal data collection.
PT-CL03	The system should be able to record the information about how long personal data will be retained.
PT-CL04	The system should let the data subject know about personal data collection.
PT-CL05	The system should allow the data subject to give or refuse consent to data collection.
PT-CL06	The system should be able to record the information about the party who gives consent to data collection on behalf of the data subject.

ค. 2 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล

ตารางที่ ค.2 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล

Pattern Name	**Data Quality Pattern**
Privacy Principle	Data Quality Principle
Pattern Requirement	
ID	**Requirement**
PT-DQ01	The system should be able to check, for data completeness, that all required personal data are collected.
PT-DQ02	The system should be able to validate the collected personal data for accuracy.
PT-DQ03	The system should be able to check integrity of the collected personal data for accuracy.
PT-DQ04	The system should be able to collect current personal data for data up-to-dateness.

ค. 3 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์

ตารางที่ ค.3 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์

Pattern Name	**Purpose Specification Pattern**
Privacy Principle	Purpose Specification Principle
Pattern Requirement	
ID	**Requirement**
PT-PS01	The system should be able to inform the information about the purpose for which the collected data are to be used.
PT-PS02	The system should be able to check the retention period or expiration date of personal data.
PT-PS03	The system should be able to delete or anonymize personal data after their retention has expired.

ค. 4 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล

ตารางที่ ค.4 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล

Pattern Name	**Use Limitation Pattern**
Privacy Principle	Use Limitation Principle
Pattern Requirement	
ID	**Requirement**
PT-UL01	The system should let the data subject know about the use of personal data which deviates from the specified purposes.
PT-UL02	The system should allow the data subject to give or refuse consent to the use of personal data which deviates from the specified purposes.

ค. 5 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล

ตารางที่ ค.5 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล

Pattern Name	**Security Safeguards Pattern**
Privacy Principle	Security Safeguards Principle
Pattern Requirement	
ID	**Requirement**
PT-SS01	The system should be able to encrypt the personal data to ensure confidentiality.
PT-SS02	The system should be able to authenticate users of the system before accessing personal data.
PT-SS03	The system should be able to define access rights to create, destroy, view, and modify personal data based on authority levels or roles within the organization.
PT-SS04	The system should be able to control access to the personal data by different access types including create, destroy, view, and modify, based on authority levels or roles within the organization.
PT-SS05	The system should be able to provide a mechanism to ensure data integrity.
PT-SS06	The system should be able to provide a mechanism to ensure data availability.
PT-SS07	The system should be able to monitor and record access to personal data.
PT-SS08	The system should be able to check for suspicious data breach and notify the data controller and data subject.

ค. 6 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักการเปิดเผย

ตารางที่ ค.6 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักการเปิดเผย

Pattern Name	**Openness Pattern**
Privacy Principle	Openness Principle
Pattern Requirement	
ID	**Requirement**
PT-OP01	The system should be able to record the information about the privacy policy, including the existence and nature of personal data, main purpose of collection and use, and identity and usual residence of the data controller.
PT-OP02	The system should let the data subject know about the privacy policy.
PT-OP03	The system should let the data subject know about change of the privacy policy.
PT-OP04	The system should provide the data subject with access to the privacy policy.

ค. 7 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล

ตารางที่ ค.7 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล

Pattern Name	**Individual Participation Pattern**
Privacy Principle	Individual Participation Principle
Pattern Requirement	
ID	**Requirement**
PT-IP01	The system should allow inquiry of and response to issues related to personal data.
PT-IP02	The system should allow the data subject to access his/her personal data.
PT-IP03	The system should allow the data controller to erase and update the data subject's personal data.

ค. 8 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ

ตารางที่ ค.8 รายการความต้องการของแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ

Pattern Name	**Accountability Pattern**
Privacy Principle	Accountability Principle
Pattern Requirement	
ID	**Requirement**
PT-AC01	The system should be able to record events that occur in the system.
PT-AC02	The system should be able to perform periodic assessment of data breach.
PT-AC03	The system should be able to handle data breach.
PT-AC04	The system should allow inquiry of and response to issues related to personal data.
PT-AC05	The system should be able to report on data breach checking.
PT-AC06	The system should be able to notify relevant parties, i.e. privacy enforcement authorities and data subject, in the event of significant data breach.

ภาคผนวก ง.

แบบรูปการออกแบบสำหรับความเป็นส่วนตัว

ง. 1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

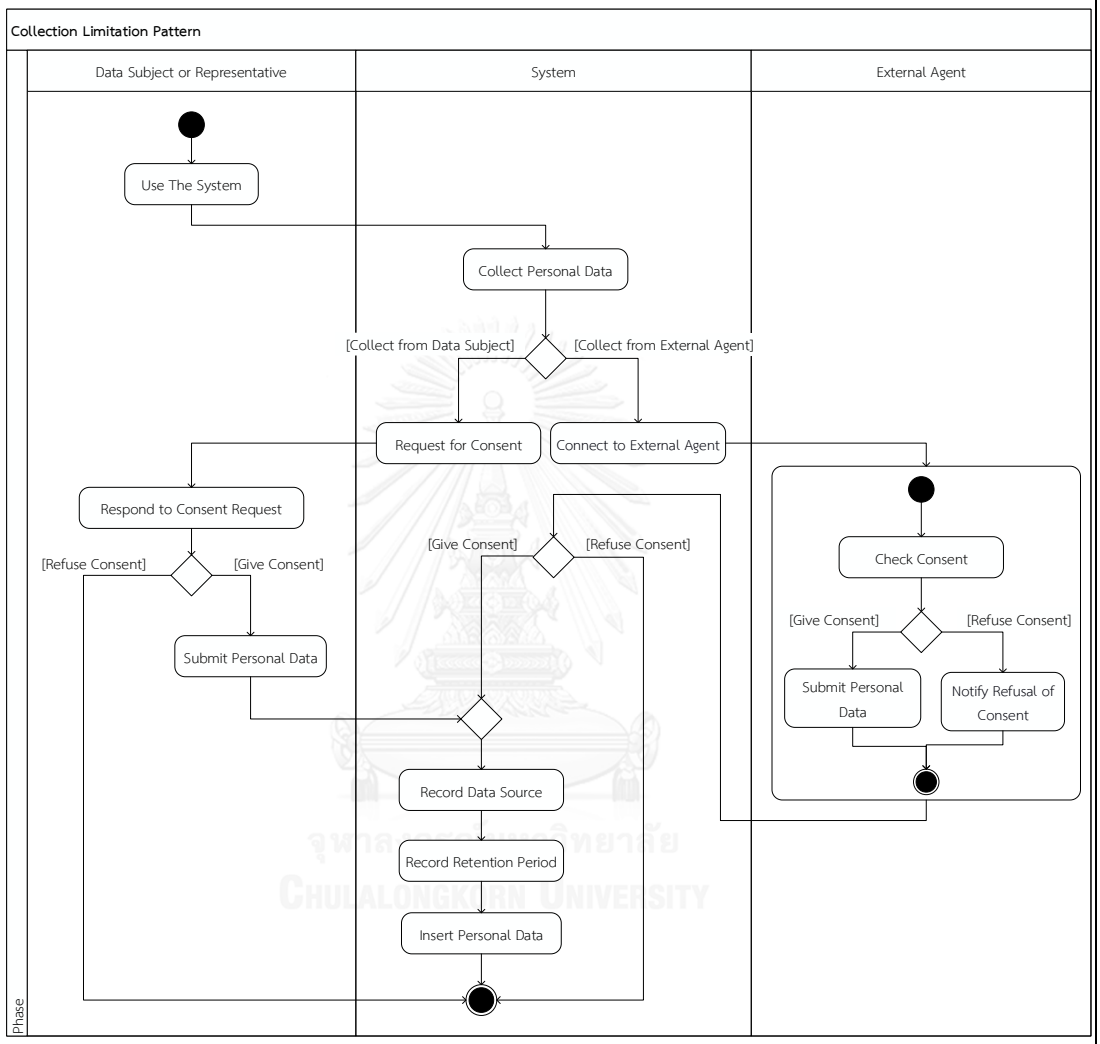
Pattern Name	**Collection Limitation Pattern**
Privacy Principle	Collection Limitation Principle
Intent	This pattern assures that the personal data that the system of the data controller will collect should be limited and are collected with the knowledge or consent of the data subject. As personal data are attributes of the data subject, it is necessary to assure that he/she is still in control of the personal data and the collection does not violate his/her civil right as the owner of the collected data.
Motivation	When a data subject (or user) is about to use a service of a data controller on the Internet, the service may need to collect certain personal data of the data subject for the processing of the service. For example, a cosmetic online shopping web site needs to collect a customer's personal data at registration time. Among the collected personal data (such as name, email, and phone number), the web site also collects national ID and does not ask the customer to give consent to the collection. In this scenario, sensitive information (i.e. national ID) that is not used in the processing of the online shop is unnecessarily disclosed. If the online shop neither asks the customer to give consent to the collection nor handles data privacy well, the customer's sensitive information may leak and be misused in a way that causes harm to the customer. Thus, the data subject should be aware of what, when, and how personal data are collected. The system of the data controller should collect only the personal data that are necessary for use, and let the data subject know about personal data collection and give consent. Sometimes it may be the case that consent is given by a representative on behalf of the data subject.

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Applicability	Use the Collection Limitation pattern whenever the system is about to collect personal data, i.e. any information relating to an identified or identifiable individual (data subject) [1]. According to [11], the system should use the Collection Limitation pattern when collecting any of the following categories of personal data: • Preference data i.e. Preferences on books, music, cities, friends, wish list, and so forth. • Communication record i.e. SMS text, phone records, address book, and so forth. • Web footprint i.e. Visited websites, search keywords, social comment logs and social graph, and so forth • Personal profile i.e. Height, weight, published papers, education/career experiences, exam performance, and so forth. • Consumption service record i.e. Bank account and transaction records, flight and hotel orders, car rental orders, supermarket records, e-commerce transaction record, and so forth. • Public service record Personal salary records, household energy record, personal credit, and so forth. Also, the system should use the Collection Limitation pattern no matter where such personal data are collected from [11]: • Personal device i.e. PC, smart phones, mobile devices, tablet, and so forth. • Services i.e. Web applications, and so forth. • Social network i.e. Facebook, Twitter, blogs, and so forth. • Sensors i.e. GPS, thermometer, wearable devices, and so forth. • The data subject itself i.e. Email, work schedule, documents, pictures, video, audio.

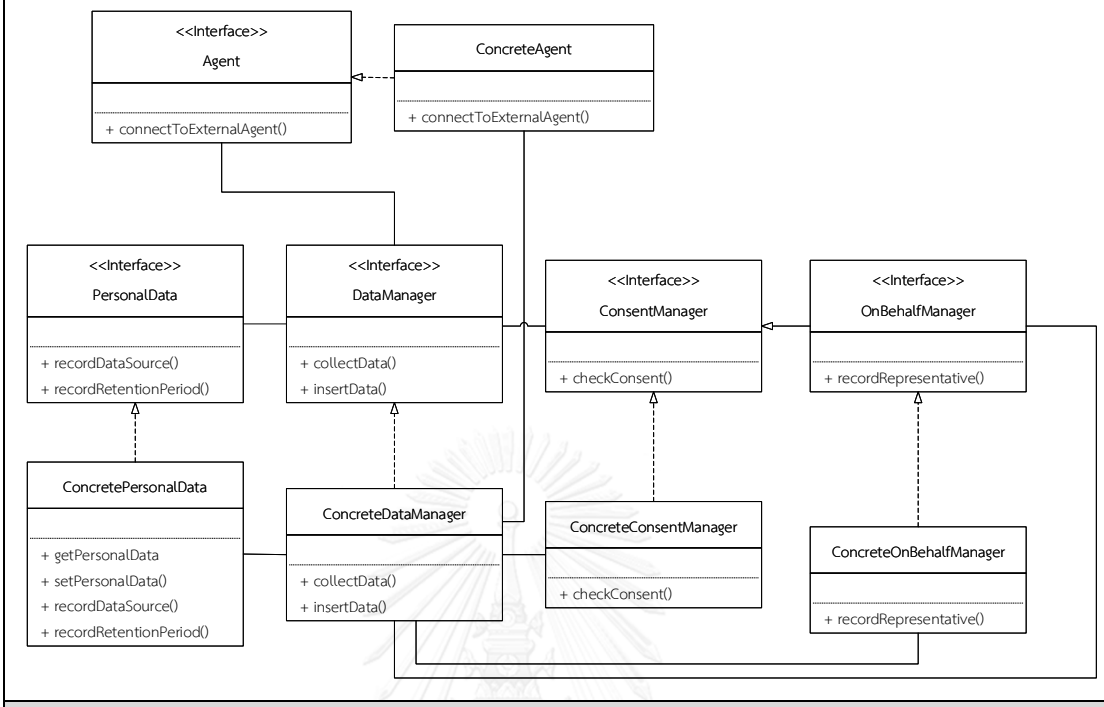
ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Behavior and Structure
Activity Diagram



ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Class Diagram



Participants	
Interface Name/Class Name	**Detail**
Interface DataManager	An interface that has the responsibilities to coordinate checking of consent and collection of personal data from the data subject. It inserts the collected data into the system if the data subject gives consent.
Class ConcreteDataManager	A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. coordinating update of personal data when there are changes etc.
Interface ConsentManager	An interface that has the responsibilities to inform the data subject that the system needs to collect personal data and check whether the data subject gives or refuses consent.
Class ConcreteConsentManager	A concrete class that implements the interface ConsentManager.
Interface OnBehalfManager	An interface that inherits from the interface ConsentManager. It has the responsibility to record the information about who, on behalf of the data subject, has given the consent to collect the personal data.

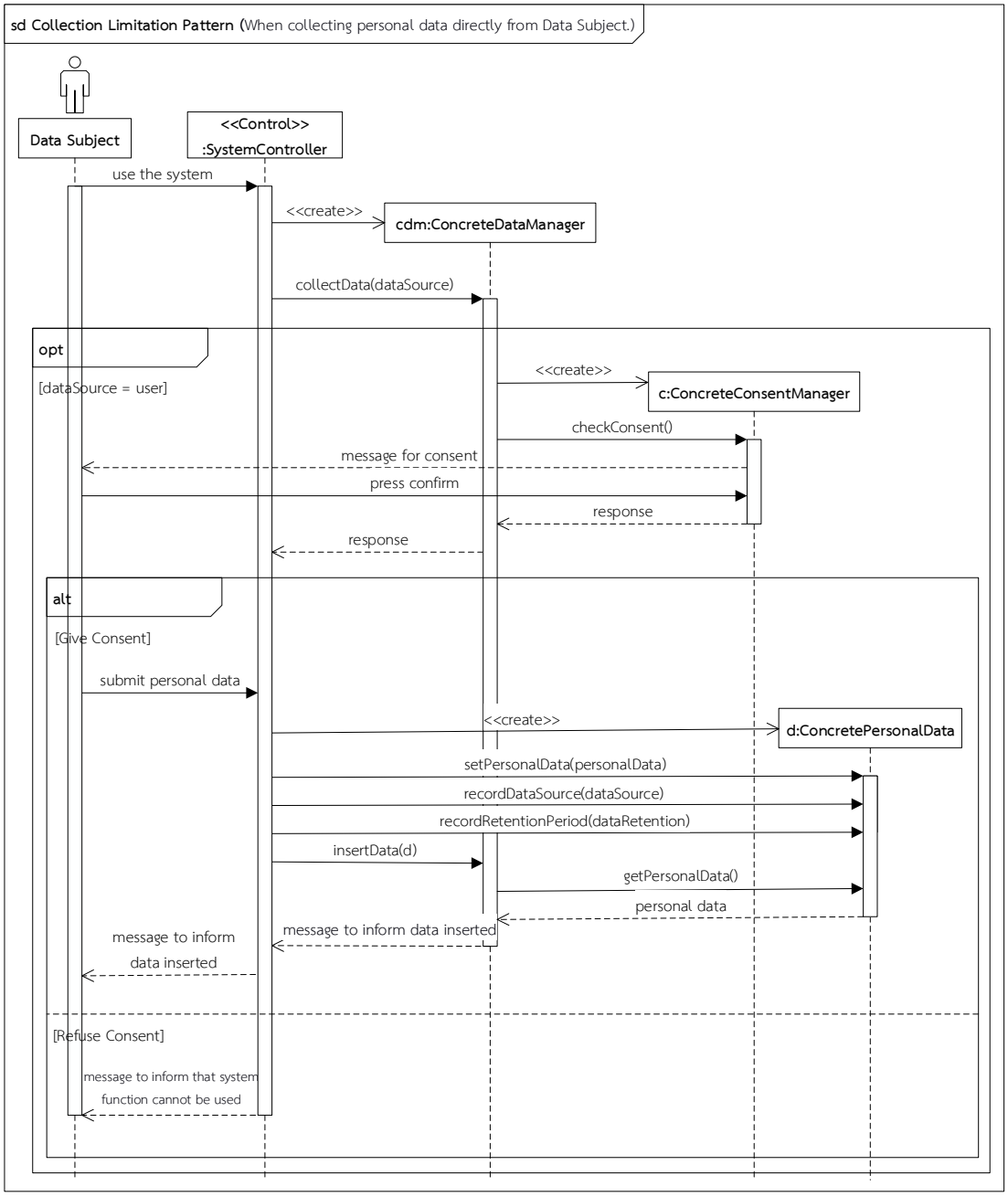
ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Participants	
Interface Name/Class Name	**Detail**
Class ConcreteOnBehalfManager	A concrete class that implements the interface OnBehalfManager.
Interface Agent	An interface that has the responsibilities to connect to an external agent to collect personal data of the data subject on behalf of the system, and obtain personal data via the external agent if consent is given.
Class ConcreteAgent	A concrete class that implements the interface Agent.
Interface PersonalData	An interface that represents the collected personal data.
Class ConcretePersonalData	A concrete class with the data attributes that are necessary for the purpose of the system.
Collaboration	
1. When the data subject requests to use a system function, the system controller creates a ConcreteDataManager object to collect personal data. The data source where personal data will be collected (i.e. data subject or external agent) and data retention detail (e.g. retention time, format) should be determined by the data controller. 2. If personal data will be collected directly from the data subject, 2.1 The ConcreteDataManager object creates a ConcreteConsentManager object to check whether the data subject gives or refuse consent to personal data collection. 2.2 The ConcreteDataManager object notifies the system controller of the consent checking result. 2.3 If consent is given, the system controller creates a ConcretePersonalData object and calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store. 3 If personal data will be collected via an external agent, 3.1 The ConcreteDataManager object creates a ConcreteAgent object to connect to an external agent. 3.2 If the external agent is given the consent to collect personal data, the ConcreteAgent object creates a ConcretePersonalData object and calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store.	

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

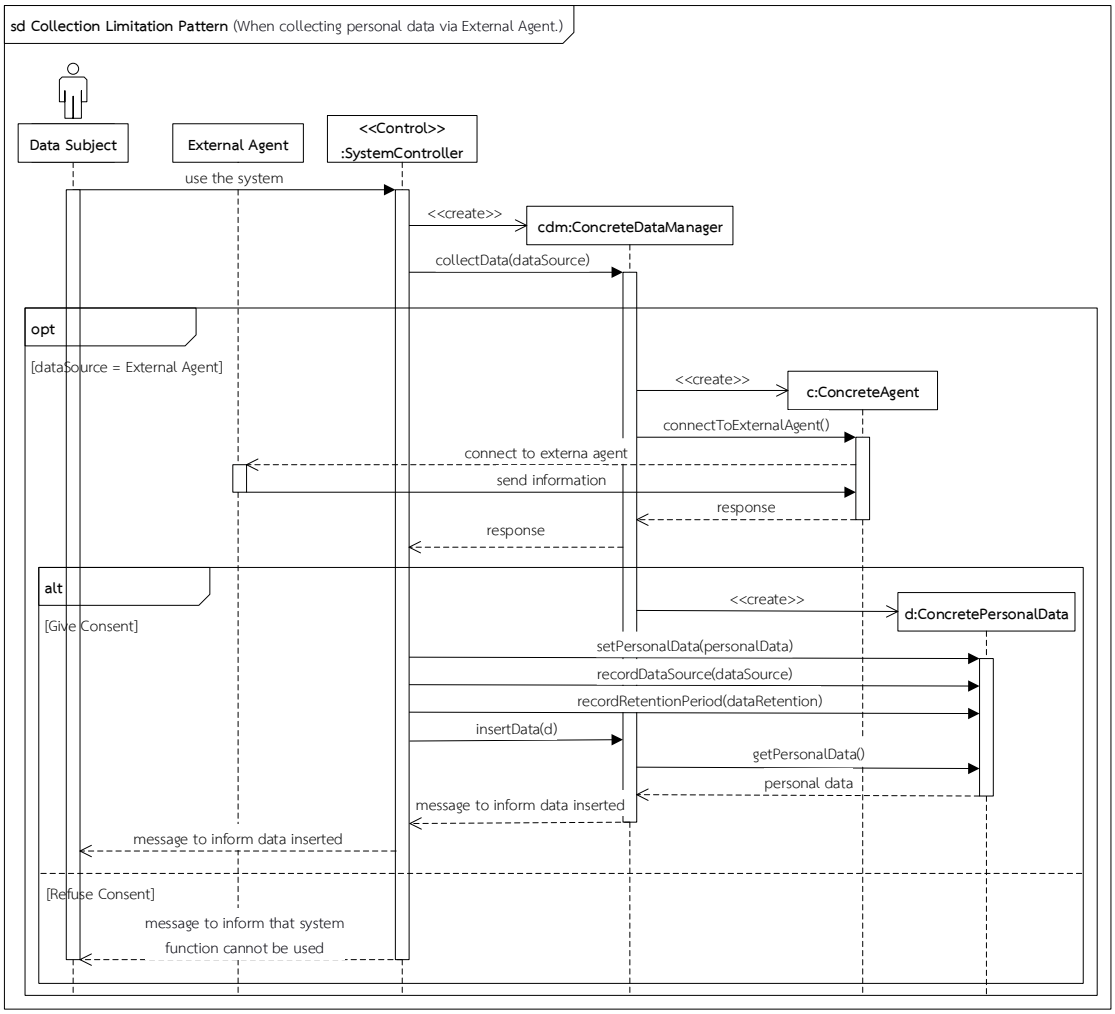
Collaboration
4 In the case that consent is given by a representative on behalf of the data subject and the system needs to record such a fact, the ConcreteDataManager can create an OnBehalfManager object, instead of a ConcreteConsentManager object, to check consent and also record the representative information.

Sequence Diagram (When collecting personal data directly from Data Subject.)



ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data via External Agent.)



ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

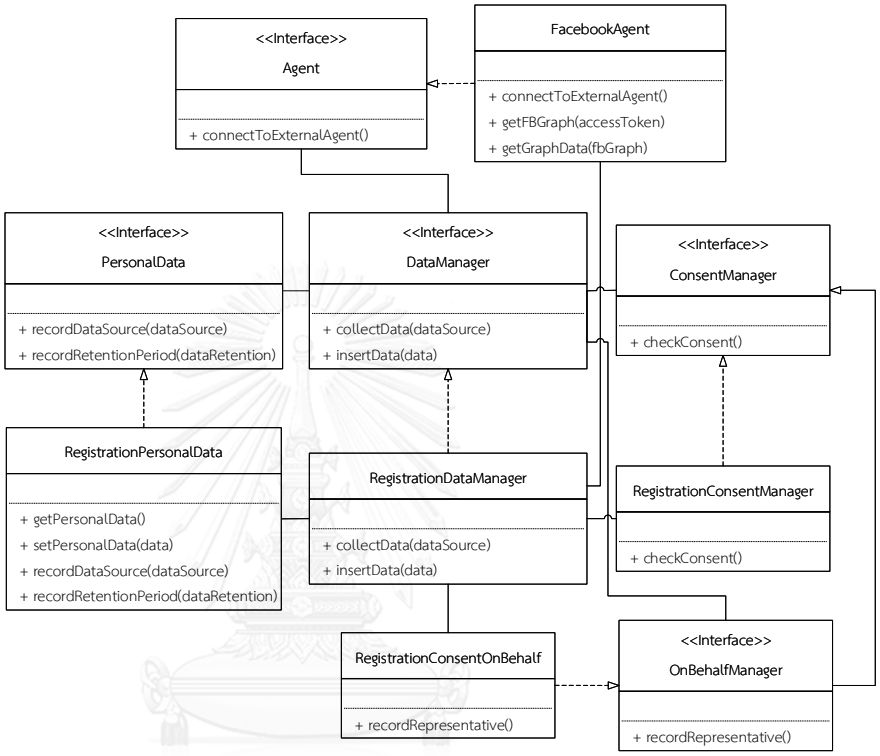
Consequences	1. Using this pattern, the system of the data controller can handle collection of personal data as recommended by OECD. The system can handle important tasks that are addressed by the Collection Limitation principle. That is, the personal data management, consent and on-behalf consent management, and connection to external agents tasks are specifically accommodated by corresponding interfaces (and their concrete classes). Thus, cohesion of each individual class in the design pattern structure can be enhanced. 2. The data subject can be assured that the personal data that will be used by the system are collected with the knowledge or consent of his/hers. The data subject can be informed of what data are about to be collected and can decide whether such data are necessary for the processing of the system. The data subject has the right to reject the collection and, as a result, the service, if he/she does not want to disclose such requested personal information.
Implementation	Issues regarding the implementation of the Collection Limitation pattern are as follows. 1. The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly. Such a limit imposed on personal data collection entails a data quality requirement and the data controller who implements the system should refer to the Data Quality pattern also. 2. Since data collection should be restricted to the minimum necessary to fulfil the specified purpose [1], the data controller who implements the system should decide on which personal data are really needed. Therefore, the class ConcretePersonalData should be implemented to have only the data attributes that are necessary for the purpose of the system. 3. Even though OECD defines that there should be limits to the collection of personal data, the detail about which data are prohibited from collection are left out because it depends on the traditions and attitudes in each country where the system operates. For example [1], personal data such as race, religious beliefs, criminal records, health, sex life, and so forth are sensitive and the collection of which should be restricted or even prohibited by European legislation. On the other hand, for the United States privacy

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

	legislation, no data are specified as being more private or sensitive than others as sensitivity may depend on the context and use of the data. Therefore, the data controller who implements the system should decide on which personal data attributes should be included in the class ConcretePersonalData while not violating the privacy legislation and traditions that apply to the system. 4. OECD defines that there should be limits to data collection activities of the data controller. For example [1], hidden devices should not be used and the data subject should not be deceived to provide personal information. However, the control of such limits cannot be captured completely by the structural model of this design pattern. To help address this issue, the data controller should record the source of the data collection, i.e. whether the data are collected from the data subject directly or an external agent. The class ConcretePersonalData should be implemented to record the source of the collected personal data. This could help to deal with data collection misconduct if such a case arises. In addition, the data controller should be open with the data subject about what data are collected and for what purposes. The data controller should refer to the Openness pattern also. 5. OECD defines that there should be limits to the collection of personal data in such a way that the civil rights of the data subject should be of concern and the data collection should not violate civil rights of the data subject. Since the data subject should be entitled to his/her civil rights including the right to his/her collected personal data, the data controller should also refer to the Individual Participation pattern. 6. It is possible that the data subject may be represented by another party who gives consent to the collection of the data subject's personal data, e.g. in the case of minors, mentally disabled person etc. If the data controller needs to record this fact, e.g. for legal purpose, the interface OnBehalfManager that inherits from the interface ConsentManager can be implemented (i.e. class ConcreteOnBehalfManager) to also record the information about who, on behalf of the data subject, has given the consent to collect the personal data.

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	An example of a user registration system that applies the Collection Limitation pattern is shown below. The system collects personal data of any user who requests to use the system and the collection can be done either by the system or via Facebook.



When a user (data subject) is required to register with the system to use its service, the system uses the class RegistrationDataManager that implements the interface DataManager to collect personal data (name, email, tel) as shown in the sample code below. If the data source of collection is “User”, the data are collected directly from the user. Otherwise they are collected via Facebook.

```
interface DataManager{
    public Map<String, String> collectData(String dataSource);
    public void insertData(RegistrationPersonalData pd);
}
```

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	

```
public class RegistrationDataManager implements DataManager{

    @Override
    public Map<String, String> collectData(String dataSource) {
        Map<String,String> data = new HashMap<String, String>();
        if(dataSource.equalsIgnoreCase("User")){
            RegistrationConsentManager rcm = new RegistrationConsentManager();
            if(rcm.checkConsent()){
                data.put("consent", "accept");
            }else{
                data.put("consent", "refuse");
            }
        }else{
            FacebookAgent fa = new  FacebookAgent();
            data.putAll(fa.connectToExternalAgent());
        }
        return data;
    }
    @Override
    public void insertData(RegistrationPersonalData data){
        //TODO Add your code to insert personal data into data store.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String insert = "INSERT INTO personaldata "
                    + "(name, email, tel, dataSource, dataRetention) "
                    + "VALUES (?, ?, ?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert) ;
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, data.getName());
            stmt.setString(parameterIndex++, data.getEmail());
            stmt.setString(parameterIndex++, data.getTel());
            stmt.setString(parameterIndex++, data.getDataSource());
            stmt.setString(parameterIndex++, data.getDataRetention());

            stmt.execute();
            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
    }
}
```

If personal data are collected directly from the user, the class RegistrationConsentManager that implements the interface ConsentManager is used to check whether the user gives or refuse consent to personal data collection as shown in the sample code below.

```
public interface ConsentManager {
    public boolean checkConsent();
}
public class RegistrationConsentManager implements ConsentManager{
    private Scanner scan;

    @Override
    public boolean checkConsent() {
        scan = new Scanner(System.in);
        System.out.println("Give Consent Please Enter 'Y' "
                + "and Refuse Consent Please Enter 'N'");
        String accept = scan.next();
        if(accept.equalsIgnoreCase("Y"))
            return true;
        else
            return false;
    }
}
```

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	The class RegistrationPersonalData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of the system as shown in the sample code below. The system requires only the attributes name, email, and telephone number to use its service. In addition, it records the source of collection of these data as well as the end of the retention period which varies by different types of data that the system collects during the use of its service, i.e. registration data, location data, and others.

```
public interface PersonalData {
    public String recordDatasource(String dataSource);
    public String recordRetentionPeriod(String dataRetention);
}
public class RegistrationPersonalData implements PersonalData{
    private String name;
    private String email;
    private String tel;
    private String dataSource;
    private String dataRetention;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getTel() {
        return tel;
    }
    public void setTel(String tel) {
        this.tel = tel;
    }
    public String getDataSource() {
        return dataSource;
    }
    public String getDataRetention() {
        return dataRetention;
    }
    @Override
    public String recordDatasource(String dSource) {
        dataSource = dSource;
        return dataSource;
    }
    @Override
    public String recordRetentionPeriod(String dRetention) {
        if(dRetention.equalsIgnoreCase("Registration"))
            dataRetention = "When deleting account";
        else if(dRetention.equalsIgnoreCase("Location"))
            dataRetention = "When ending program";
        else
            dataRetention = "When deleting data";
        return dataRetention;
    }
}
```

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

Sample Design and Code	In the case that the user registers via Facebook, the class FacebookAgent that implements the interface Agent is used to collect personal data as shown in the sample code below.

```
public interface Agent {
    public Map<String, String> connectToExternalAgent();
}
public class FacebookAgent implements Agent{

    @Override
    public Map<String, String> connectToExternalAgent() {
        String accessToken = "";
        /* TODO Add your code to connect to external agent,
         * i.e. Facebook, to obtain an access token here */
        String graph = getFBGraph(accessToken);
        Map<String, String> fbProfileData = getGraphData(graph);
        return fbProfileData;
    }

    public String getFBGraph(String accessToken) {
        String graph = null;
        try {

            String g = "https://graph.facebook.com/me?" + accessToken;
            URL u = new URL(g);
            URLConnection c = u.openConnection();
            BufferedReader in = new BufferedReader(new InputStreamReader(
                    c.getInputStream()));
            String inputLine;
            StringBuffer b = new StringBuffer();
            while ((inputLine = in.readLine()) != null)
                b.append(inputLine + "\n");
            in.close();
            graph = b.toString();
            System.out.println(graph);
        } catch (Exception e) {
            e.printStackTrace();
            throw new RuntimeException("ERROR in getting FB graph data. " + e);
        }
        return graph;
    }
    public Map<String, String> getGraphData(String fbGraph) {
        Map<String, String> fbProfile = new HashMap<String, String>();
        try {
            JSONObject json = new JSONObject(fbGraph);
            fbProfile.put("id", json.getString("id"));
            fbProfile.put("name", json.getString("name"));
            if (json.has("email"))
                fbProfile.put("email", json.getString("email"));
            if (json.has("gender"))
                fbProfile.put("gender", json.getString("gender"));
        } catch (JSONException e) {
            e.printStackTrace();
            throw new RuntimeException("ERROR in parsing FB graph data. " + e);
        }
        return fbProfile;
    }
}
```

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

<table>
<tr><td>Sample Design and Code</td><td>In the case that that the system needs to record, e.g. for legal purpose, who gives consent on behalf of the user, the class RegistrationConsentOnBehalf that implements the interface OnBehalfManager as shown in the sample code below would be called by the class RegistrationDataManager instead of the class RegistrationConsentManager.

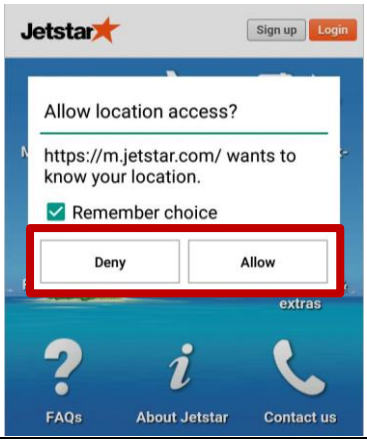
<pre>
public interface OnBehalfManager extends ConsentManager {
    public String recordRepresentative();
}
public class RegistrationConsentOnBehalf implements OnBehalfManager{

    private Scanner scan;

    @Override
    public boolean checkConsent() {
        scan = new Scanner(System.in);
        System.out.println("Give Consent Please Enter 'Y' "
                + "and Refuse Consent Please Enter 'N'");
        String accept = scan.next();
        if(accept.equalsIgnoreCase("Y"))
            return true;
        else
            return false;
    }

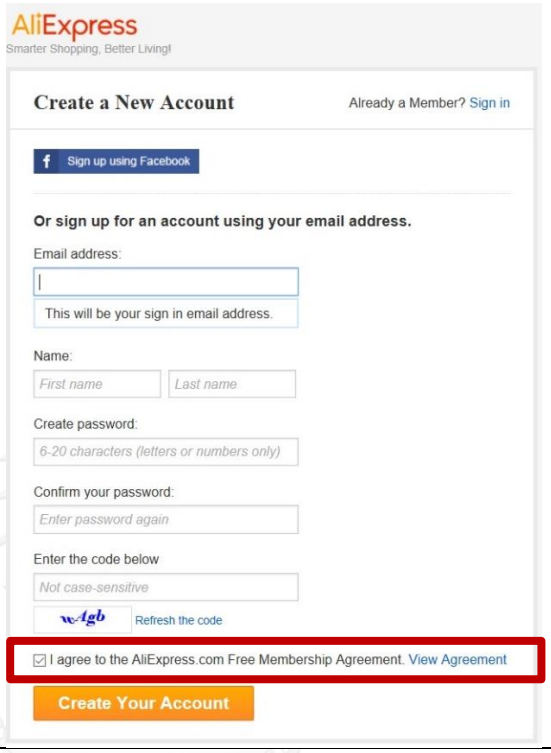
    @Override
    public String recordRepresentative() {
        scan = new Scanner(System.in);
        System.out.println("Enter name of representative");
        String representative = scan.next();
        return representative;
</pre>
</td></tr>
<tr><td>Known use</td><td>Examples of existing systems that take the Collection Limitation principle into account include:

- JetStar (Website): While a user is using a web service on a mobile phone, the system requires further access to the user location and will ask for user consent. The user can then give or refuse consent to the collection of his/her location data.



</td></tr>
</table>

ตารางที่ ง.1 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

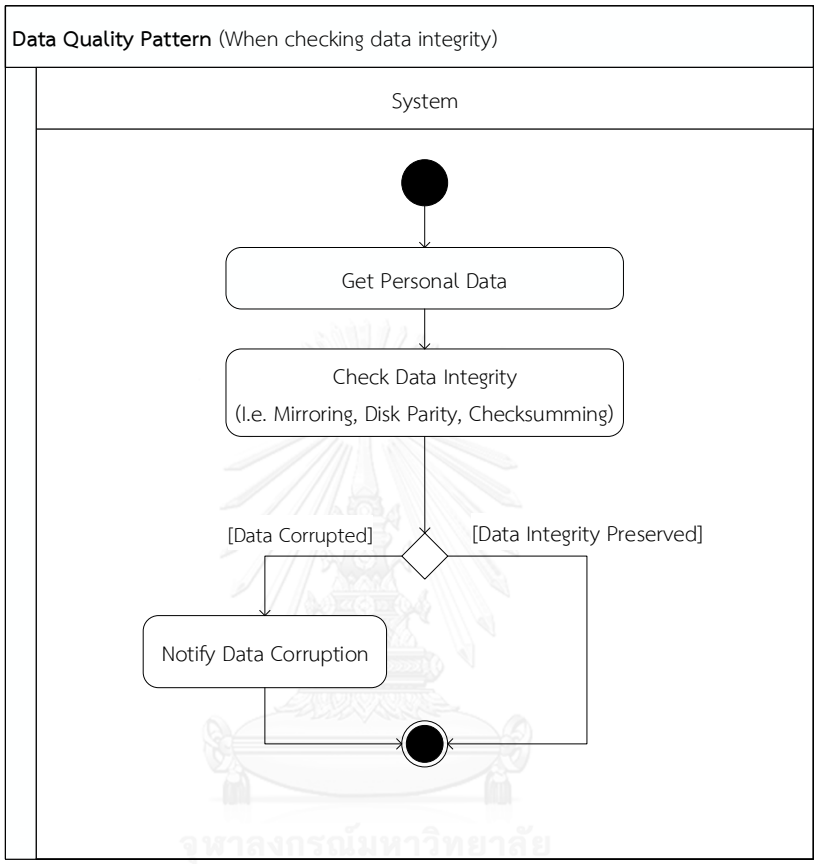
Known use	● AliExpress (Website): When a user creates an account, the system will collect personal data. If the user gives consent to the system to collect and use personal data, the user should check the agreement checkbox. 
Related Patterns	Data Quality The Data Quality pattern can be used with the Collection Limitation pattern to ensure that the personal data that are collected are accurate, complete, and up-to-date for the processing of the system. Openness The Openness pattern can be used with the Collection Limitation pattern to ensure that the collection of personal data is for the real benefit of the data subject in the use of the system service. The data subject should be informed of what personal data are collected and for what purposes. Individual Participation In addition to the Collection Limitation pattern, the Individual Participation pattern can be used so that the data subject remains entitled to the right to participate in the use of the collected personal data by the system and be informed of the issues related to the collected personal data.

ง. 2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล

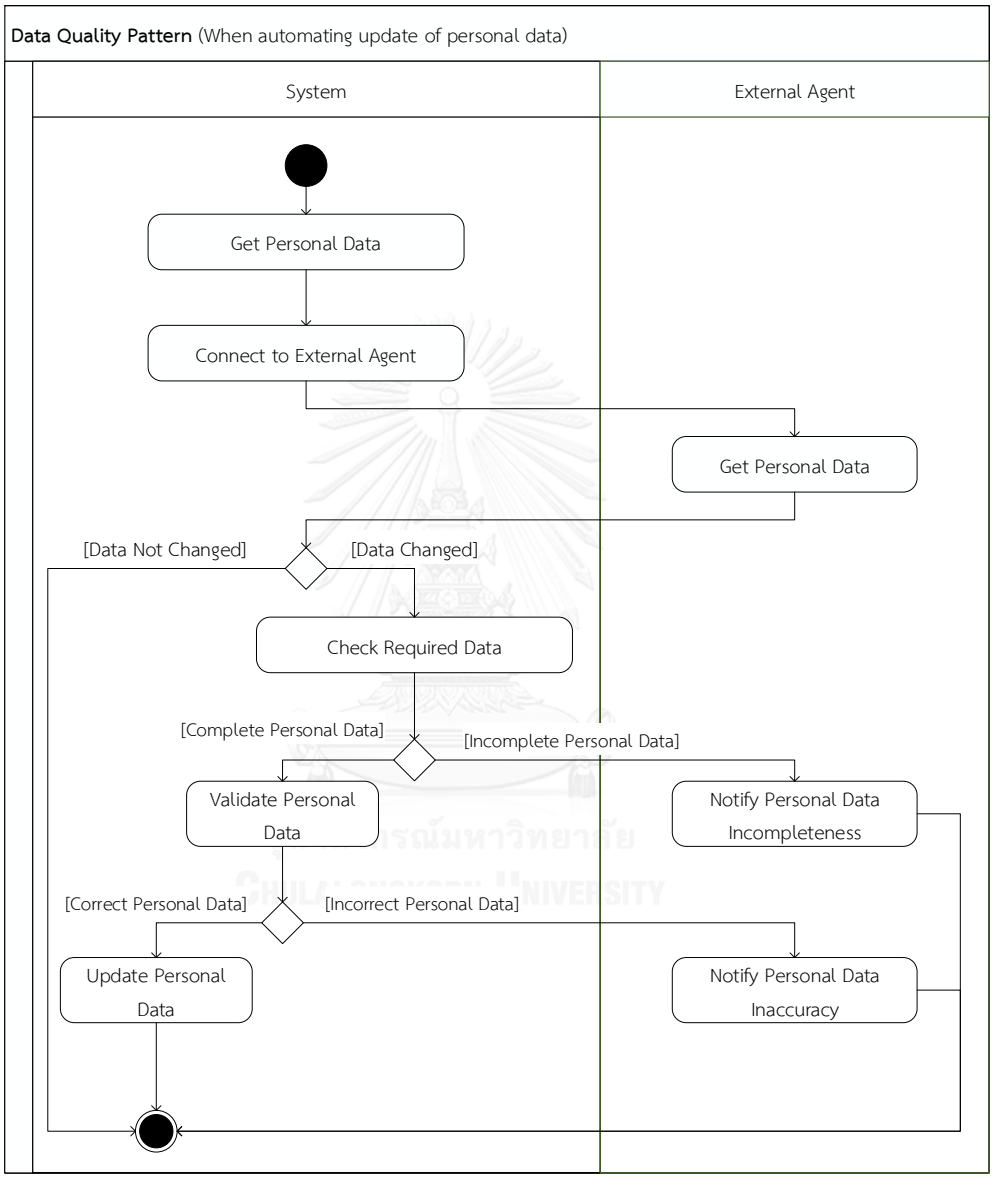
Pattern Name	**Data Quality Pattern**
Privacy Principle	Data Quality Principle
Intent	This pattern assures that the collected personal data are relevant to the purposes for which they are to be used. At the same time, the pattern assures accuracy, completeness, and up-to-dateness of the collected data, since poor quality data, as a result of any mistakes or passage of time, may lead to poor quality service of the system.
Motivation	The data controller should consider different quality aspects of the personal data that are collected, processed, and stored. First, the collected data should be relevant to the purpose of use, otherwise the collection would be wasteful. Furthermore, even though the data are relevant, their inaccuracy, incompleteness, and out-of-dateness may lead to processing mistakes and misinterpretations of data, and harm can be caused to the data subject. Thus, the system should collect only the data that are relevant to the purpose of use, and provide mechanisms to promote completeness, accuracy, and up-to-dateness of the collected personal data.
Applicability	Use the Data Quality pattern to maintain quality of the personal data of a data subject throughout their life cycle within the system, i.e. from collection to storage and processing: • When collecting data, the data controller should ensure that the collected data are relevant to the purposes. This pattern can be used to check also that all necessary data are collected and those collected are validated. • When storing data, this pattern can be used to allow a data integrity mechanism to be incorporated into the system to ensure integrity of the personal data maintained by the system. • When processing data, this pattern can be used to ensure that the personal data to be processed are kept up-to-date.

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

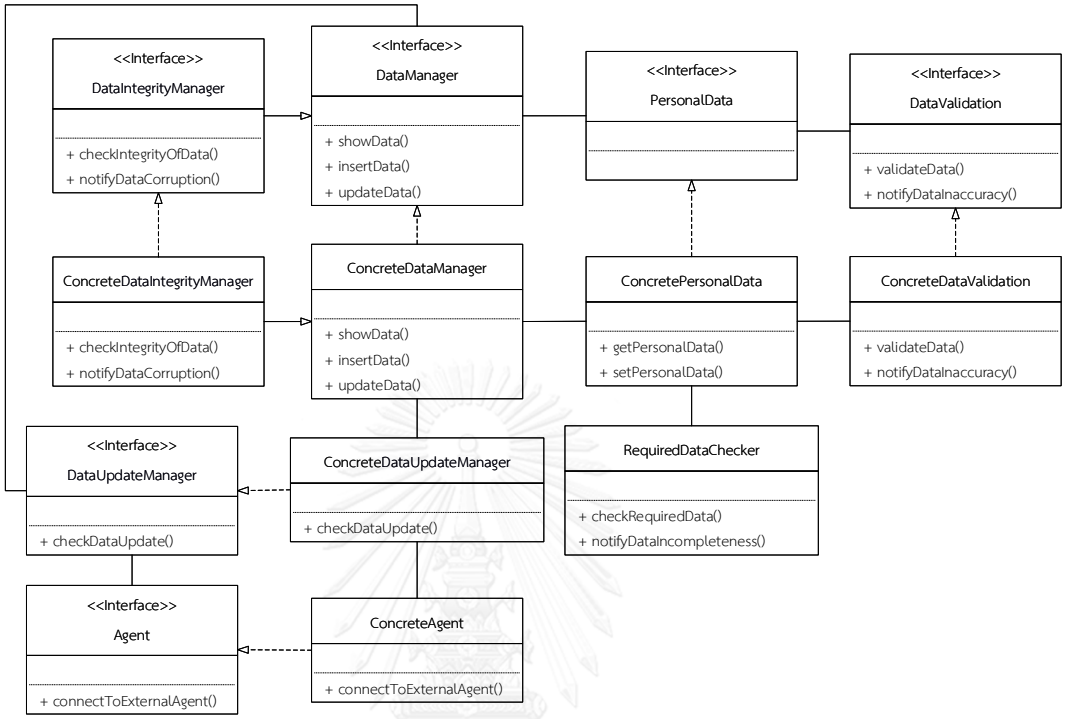
Behavior and Structure
Activity Diagram (When checking data integrity) 

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Behavior and Structure
Activity Diagram (When automating update of personal data)



ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Behavior and Structure	
Class Diagram	
	
Participants	
Interface Name/Class Name	**Detail**
Interface DataManager	An interface that has the responsibilities to coordinate checking of consent and collection of personal data from the data subject. It inserts the collected data into the system if the data subject gives consent.
Class ConcreteDataManager	A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. coordinating update of personal data when there are changes etc.
Interface PersonalData	An interface that represents the collected personal data.
Class ConcretePersonalData	A concrete class with the data attributes that are necessary for the purpose of the system.
Interface DataValidation	An interface that has the responsibilities to validate personal data for accuracy, and notify the data subject or data controller if the collected data are not valid.

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Participants	
Interface Name/Class Name	**Detail**
Class ConcreteDataValidation	A concrete class that implements the interface DataValidation. Its behavior varies by the validation criteria, e.g. validation of data values and format.
Class RequiredDataChecker	A concrete class that has the responsibilities to check completeness of the data that are required by the system, and notify the data subject or data controller if the collected personal data are incomplete.
Interface DataIntegrityManager	An interface that inherits from the interface DataManager. It has the responsibilities to implement a data integrity mechanism to assure data integrity.
Class ConcreteDataIntegrityManager	A concrete class that implements the interface DataIntegrityManager. Other responsibilities related to checking personal data integrity can be added.
Interface DataUpdateManager	An interface that has the responsibilities to automate collection of the updates of personal data when needed.
Class ConcreteDataUpdateManager	A concrete class that implements the interface DataUpdateManager. Other responsibilities related to automating the collection or update of personal data can be added.
Interface Agent	An interface that has the responsibilities to connect to an external agent to collect personal data.
Class ConcreteAgent	A concrete class that implements the interface Agent.
Collaboration	
1. When checking data integrity 1.1. The system controller creates a ConcreteDataIntegrityManager object to check personal data integrity. 1.2. The ConcreteDataIntegrityManager object get personal data. The ConcreteDataIntegrityManager object then creates a ConcretePersonalData object, gets the personal data from a data store, and sets the personal data into the ConcretePersonalData object which in turn will be sent to the ConcreteDataIntegrityChecker object.	

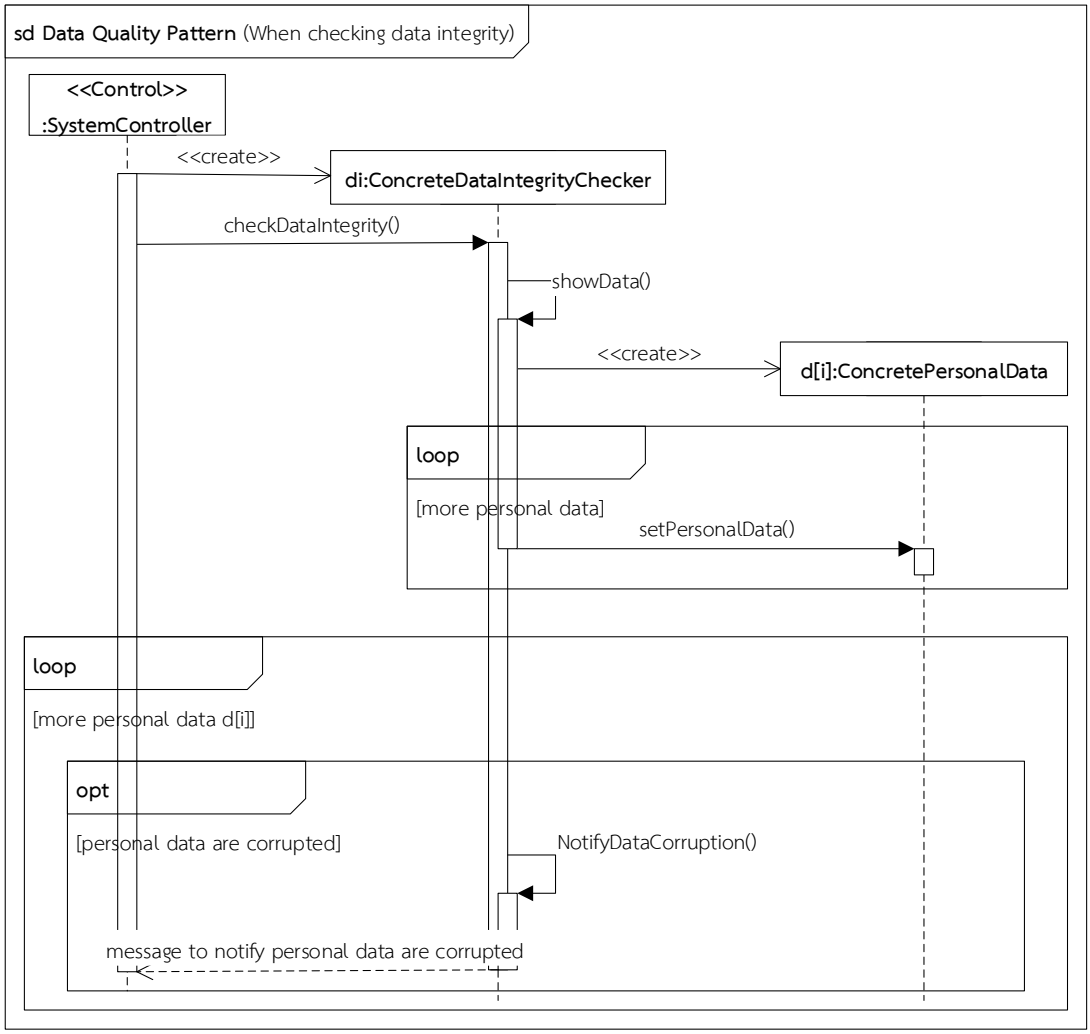
ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

1.3. If personal data are corrupted, the ConcreteDataIntegrityManager object notifies the system controller of data corruption.

2. When automating update of personal data
   2.1. The system controller creates a ConcreteDataUpdateManager object to check personal data updates.
   2.2. The ConcreteDataUpdateManager object creates a ConcreteDataManager object which in turn creates a ConcretePersonalData object, gets the personal data from a data store, and sets the personal data into the ConcretePersonalData object. Then the ConcretePersonalData object will be sent to the ConcreteDataUpdateManager object.
   2.3. The ConcreteDataUpdateManager object creates a ConcreteAgent object to connect to an external agent in order to get the current personal data from the external agent.
   2.4. If the ConcreteDataUpdateManager finds that there are updates of personal data from the external agent, the ConcreteDataUpdateManager object will set the updates of personal data into the ConcretePersonalData object.
   2.5. The ConcretePersonalData object creates a RequiredDataChecker object to check the updated personal data items that are required and notify the system controller if updated personal data are incomplete.
   2.6. The ConcretePersonalData object creates a ConcreteDataValidation object to validate the updated personal data and notify the system controller if the updated data are inaccurate.
   2.7. If the updated personal data are complete and accurate, the ConcreteDataUpdateManager object calls the ConcreteDataManager object to update the ConcretePersonalData object into a data store.

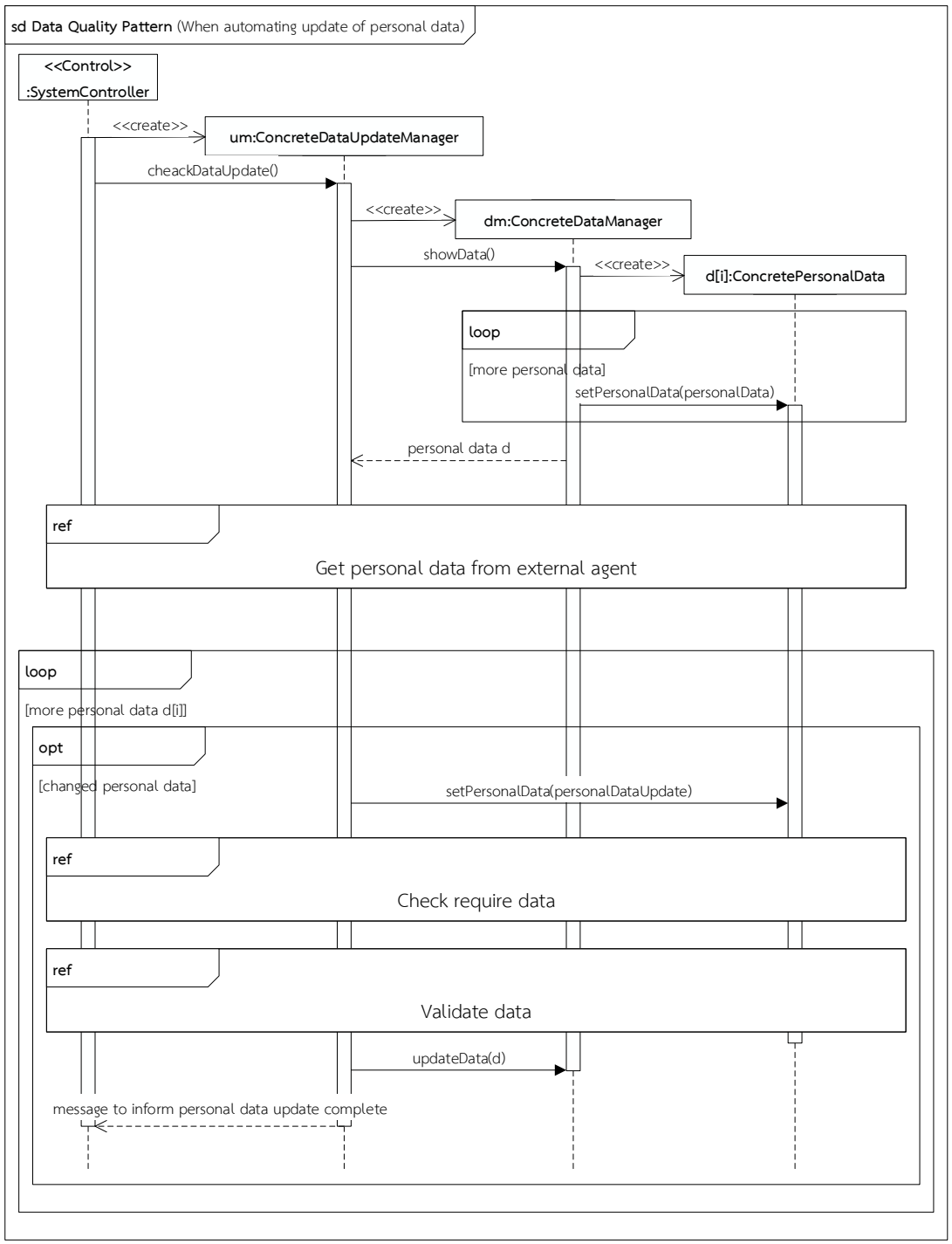
ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sequence Diagram (When checking data integrity)



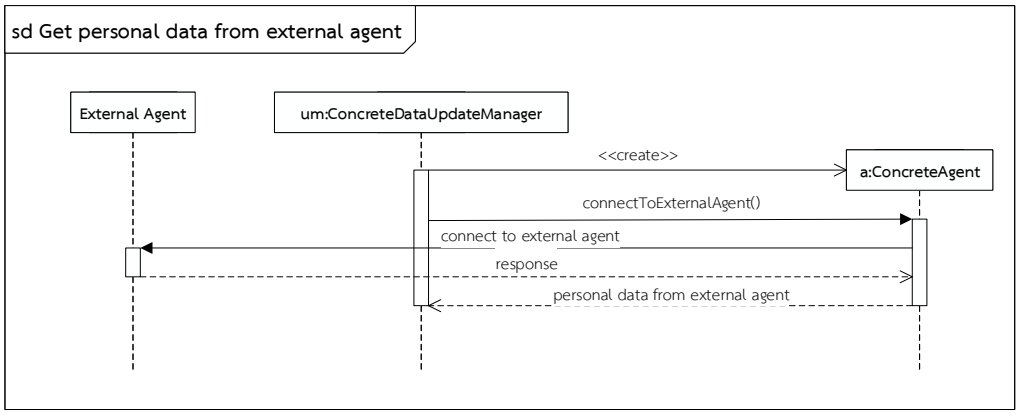
ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

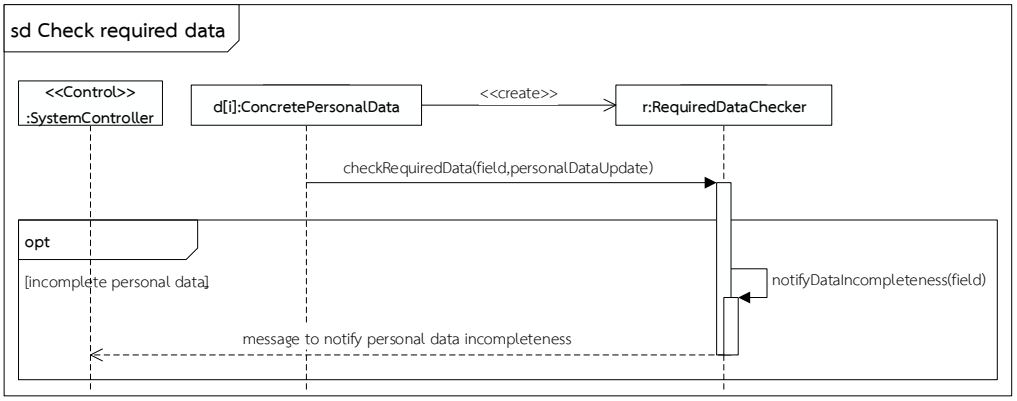
Sequence Diagram (When automating update of personal data)

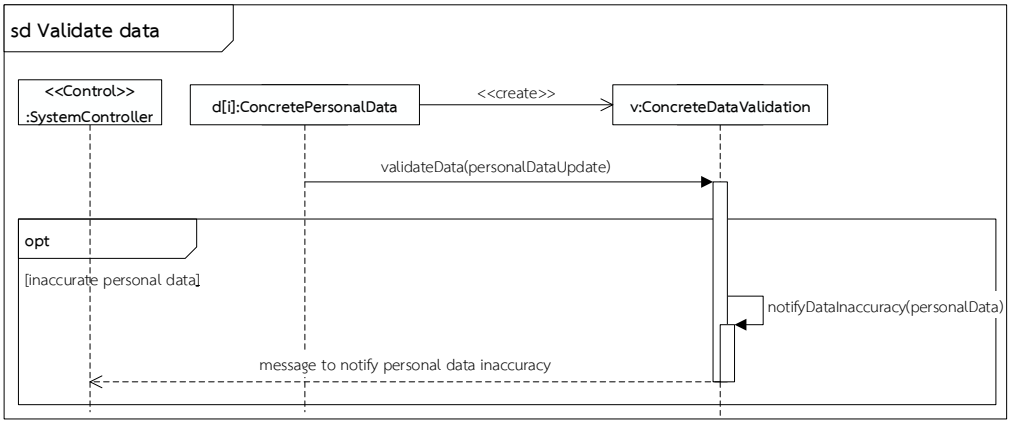


ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sequence Diagram (When automating update of personal data - continued)







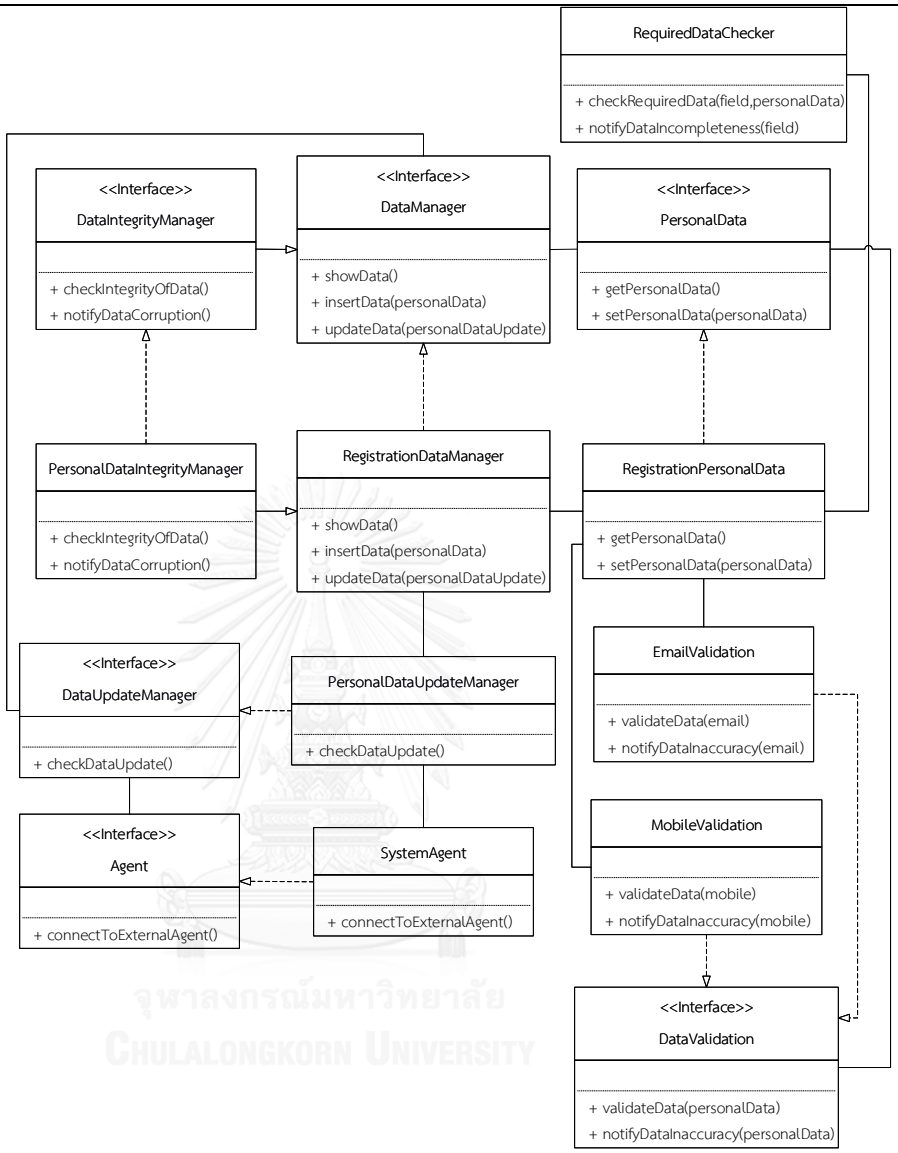
ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Consequences	1. Using this pattern, the system of the data controller can maintain the quality of the personal data as recommended by OECD. The data Subject can be assured that the personal data that are collected by the system are relevant to the purposes of use, accurate, complete, and kept up-to-date. Given the data of good quality, the system should perform its service as intended. 2. There are costs in terms of time and space associated with checking for accuracy, completeness, and up-to-dateness of the personal data. For example, the system has to perform data validation any time personal data are collected or updated. Regular checks may be performed to keep personal data updated. Also, extra storage space is required for any data integrity mechanisms that are used by the system. 3. In the case that mirroring is used as a data integrity mechanism, the system architecture employs a replication model. Consistency across replicas has to be maintained when there are updates to the personal data.
Implementation	1. The OECD defines that the collected personal data should be relevant to the purpose for which they are to be used since the data that are used in a manner different from the intended purpose could be misleading or misinterpreted. The data controller who implements the system should decide on which personal data are relevant to the purpose of use and clearly specify the purpose. The data controller should refer to the Purpose Specification pattern also. 2. To assure accuracy of the collected personal data, data integrity should be maintained over the entire lifecycle of the data. That is, the system should prevent or at least detect unintended changes made to the data as a result of storage, retrieval, or processing of data and human errors, as well as intended changes made by malicious intent. The class ConcreteDataIntegrityChecker should implement a data integrity mechanism to assure data integrity. Common techniques include mirroring (or replicating data to compare different copies of the data), disk parity, and checksumming [12].

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Implementation	3. To assure data up-to-dateness, the system may collect the update of the personal data from time to time. The storage of the collected data may vary depending on the purpose of use. For example, the class ConcreteDataManager should be implemented to either maintain only the most recent data or retain them as an archive. For automated collection, the class ConcreteDataUpdateManager can be configured to collect the updates when needed. 4. To assure up-to-dateness of personal data that are collected directly from the data subject, the system should allow the data subject to keep their personal data up-to-date. The data controller who implements the system should refer to the Individual Participation pattern also.
Sample Design and Code	An example of a system which collect personal data from external source and the system should update personal data to currently that applies the Data Quality pattern is shown below.

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sample Design and Code	 An example shows the system that collect personal (i.e. name, email, tel) data from an external agent and the system should update personal data from time to time. The class PersonalDataUpdateManager that implements the interface DataUpdateManager should be implemented to collect the updated personal data. `interface DataUpdateManager { public void checkDataUpdate(); }`

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sample Design and Code	

```
public class PersonalDataUpdateManager implements DataUpdateManager{
    @Override
    public void checkDataUpdate() {
        // TODO Add your code to check check update personal data
        RegistrationDataManager rdm = new RegistrationDataManager();
        ArrayList<RegistrationPersonalData> personalData =
                new ArrayList<RegistrationPersonalData>();
        personalData = rdm.showData();

        SystemAgent agent = new SystemAgent();
        JSONArray personalDataFromAgent = new JSONArray();
        personalDataFromAgent = agent.connectToExternalAgent();

        for(int i=0; i<personalData.size(); i++){
            RegistrationPersonalData updData = new RegistrationPersonalData();
            boolean chk = false;
            try {
                for(int j=0; j<personalDataFromAgent.length(); j++){
                    if(personalDataFromAgent.getJSONObject(j).getString("id")
                            .equalsIgnoreCase(personalData.get(i).getExid())){
                        if(!personalDataFromAgent.getJSONObject(j).getString("email").
                                equalsIgnoreCase(personalData.get(i).getEmail())){
                            updData.setEmail(personalDataFromAgent.getJSONObject(j).getString("email"));
                            chk = true;
                        }

                        if(!personalDataFromAgent.getJSONObject(j).getString("tel").
                                equalsIgnoreCase(personalData.get(i).getTel())){
                            updData.setTel(personalDataFromAgent.getJSONObject(j).getString("tel"));
                            chk = true;
                        }
                    }
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
             if(chk){
                 System.out.println("update personal data");
                 updData.setId(personalData.get(i).getId());
                 rdm.updateData(updData);
             }
        }
    }
}
```

The class SystemAgent that implements the interface Agent is used to collect personal data from an external agent as shown in the sample code below.

```
public interface Agent {
    public JSONArray connectToExternalAgent();
}
```

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sample Design and Code	

```java
public class SystemAgent implements Agent{

    @Override
    public JSONArray connectToExternalAgent() {
        // TODO Add your code to connect to external agent
        JSONArray jsonArr = new JSONArray();
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getExternalDBConnection();
        try {
            String query = "SELECT * FROM personaldata" ;
            PreparedStatement stmt = connection.prepareStatement(query) ;
            ResultSet result = stmt.executeQuery();
            while(result.next()){
                JSONObject json = new JSONObject();
                json.put("id", result.getString("id"));
                json.put("name", result.getString("name"));
                json.put("email", result.getString("email"));
                json.put("tel", result.getString("tel"));
                jsonArr.put(json);
            }
            System.out.println(jsonArr.toString());
            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
         catch (JSONException e) {e.printStackTrace();}
        return jsonArr;
    }
}
```

The class RegistrationDataManager that implements the interface DataManager to show, insert, and update personal data (name, email, tel) is shown in the sample code below. Note that the data inserted or updated will have their checksum values generated for integrity checking later.

```java
public interface DataManager {
    public ArrayList<RegistrationPersonalData> showData();
    public void insertData(RegistrationPersonalData data);
    public void updateData(RegistrationPersonalData data);
}
```

```java
    @Override
    public void insertData(RegistrationPersonalData data){
        //TODO Add your code to insert personal data into data store.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        MessageToMD5 msgmd5 = new MessageToMD5();
        try {
            String insert = "INSERT INTO personaldata "
                    + "(name, email, email_cs, tel, tel_cs) "
                    + "VALUES (?, ?, ?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert) ;
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, data.getName());
            stmt.setString(parameterIndex++, data.getEmail());
            stmt.setString(parameterIndex++, msgmd5.messageToMD5(data.getEmail()));
            stmt.setString(parameterIndex++, data.getTel());
            stmt.setString(parameterIndex++, msgmd5.messageToMD5(data.getTel()));

            stmt.execute();
            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
    }
```

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sample Design and Code	

```
    @Override
    public void updateData(RegistrationPersonalData data) {
        // TODO Add your code to update personal data
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        MessageToMD5 msgmd5 = new MessageToMD5();
        try {
            StringBuffer upd = new StringBuffer("UPDATE personaldata SET ");
            if (data.getName() != null) {
                upd.append("name = '"+data.getName()+"', ");
            }
            if (data.getEmail() != null) {
                upd.append("email = '"+data.getEmail()+"', ");
                upd.append("email_cs = '"+msgmd5.messageToMD5(data.getEmail())+"', ");
            }
            if (data.getTel() != null) {
                upd.append("tel = '"+data.getTel()+"', ");
                upd.append("tel_cs = '"+msgmd5.messageToMD5(data.getTel()+"', "));
            }

            upd.append("updatedate = '"+sdf.format(now)+"' ");
            upd.append("WHERE id = '"+data.getId()+"' ");
            System.out.println(upd.toString()) ;
            PreparedStatement stmt = connection.prepareStatement(upd.toString()) ;
            stmt.execute();
            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
    }
}
```

The class RegistrationPersonalData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of the system as shown in the sample code below.

```
public interface PersonalData {

}
public class RegistrationPersonalData implements PersonalData{
    private String id;
    private String exid;
    private String name;
    private String email;
    private String emailChecksum;
    private String tel;
    private String telChecksum;
    private DataValidation valid;
    private RequiredDataChecker rdc;

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getExid() {
        return exid;
    }
    public void setExid(String exid) {
        this.exid = exid;
    }
```

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sample Design and Code	

```
    public String getName() {
        return name;
    }
    public void setName(String name) {
        rdc = new RequiredDataChecker();
        rdc.checkRequireData("name", name);
        this.name = name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        rdc = new RequiredDataChecker();
        rdc.checkRequireData("email", email);
        valid = new EmailValidation();
        if(valid.validateData(email)){
            this.email = email;
        }
    }
    public String getEmailChecksum() {
        return emailChecksum;
    }
    public void setEmailChecksum(String emailChecksum) {
        this.emailChecksum = emailChecksum;
    }
    public String getTel() {
        return tel;
    }
    public void setTel(String tel) {
        rdc = new RequiredDataChecker();
        rdc.checkRequireData("mobile", tel);
        valid = new MobileValidation();
        if(valid.validateData(tel)){
            this.tel = tel;
        }
    }
    public String getTelChecksum() {
        return telChecksum;
    }
    public void setTelChecksum(String telChecksum) {
        this.telChecksum = telChecksum;
    }
}
```


The class RequiredDataChecker have responsibilities to check personal data that are required by the system and notify the data subject if the collected personal data are incomplete.

```
public class RequiredDataChecker {
    public void checkRequireData(String field,String data){
        if(data == null || data.equals("")){
            notifyDataIncompleteness(field);
            System.exit(1);
        }
    }

    public void notifyDataIncompleteness(String field) {
        System.out.println("Please insert "+field+" feild");
    }
}
```

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Sample Design and Code	The class EmailValidation and class MobileValidation that implements the interface DataValidation is implemented to validate the accuracy of personal data. In this example, the system validates email and mobile number.

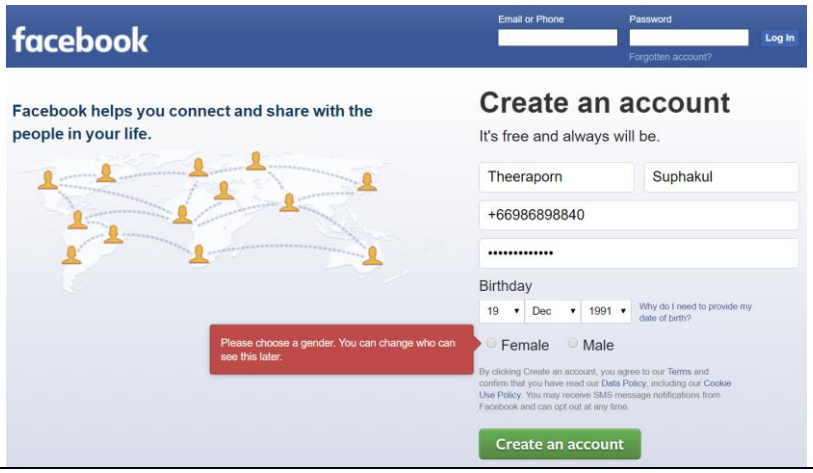
```
public interface DataValidation {
    public boolean validateData(String data);
    public void notifyDataInaccuracy(String data);
}
public class EmailValidation implements DataValidation{
    @Override
    public boolean validateData(String email) {
        boolean result = false;
        try {
            InternetAddress emailAddress = new InternetAddress(email);
            emailAddress.validate();
            result = true;
        } catch (AddressException ex) {
            notifyDataInaccuracy(email);
            System.exit(1);
        }
        return result;
    }

    @Override
    public void notifyDataInaccuracy(String email) {
        System.out.println(email+" is not valid");
        System.out.println("Please insert valid email");
    }
}
public class MobileValidation implements DataValidation{
    @Override
    public boolean validateData(String mobile) {
        boolean result = false;
        if(mobile.length() < 10){
            notifyDataInaccuracy(mobile);
            System.exit(1);
        }else{
            result = true;
        }
        return result;
    }

    @Override
    public void notifyDataInaccuracy(String mobile) {
        System.out.println(mobile+" not valid");
        System.out.println("Please insert valid mobile");
    }

}
```

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

<table>
<tr><td>Sample Design and Code</td><td>To assure accuracy of the collected personal data, the system uses the class PersonalDataIntegrityManager that implements the interface DataIntegrityManager to check integrity of the personal data in a data store, i.e. using the checksum method in this case, and notify the system if personal data are corrupted.

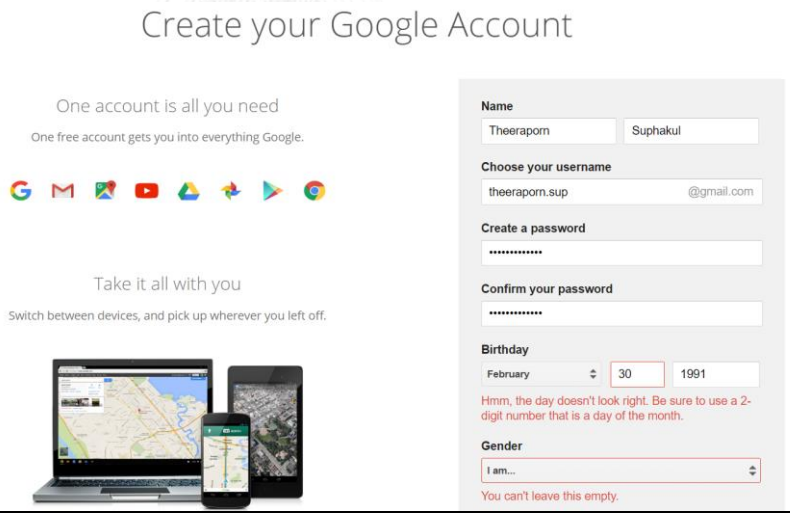
```
public interface DataIntegrityManager extends DataManager{
    public void checkDataIntegrity();
    public void notifyDataCorruption();
}
public class RegistrationDataIntegrityManager
        extends RegistrationDataManager implements DataIntegrityManager{

    private MessageToMD5 msgmd5;

    @Override
    public void checkDataIntegrity() {
        // TODO Add your code to check data integrity
        ArrayList<RegistrationPersonalData> personalData =
                new ArrayList<RegistrationPersonalData>();
        personalData = showData();
        for(int i=0; i<personalData.size(); i++){
            System.out.println("Check data integrity... "+i);
            msgmd5 = new MessageToMD5();
            String email = msgmd5.messageToMD5(personalData.get(i).getEmail());
            String emailCheckSum = personalData.get(i).getEmailChecksum();
            if(!email.equalsIgnoreCase(emailCheckSum))
                notifyDataCorruption();
        }
    }

    @Override
    public void notifyDataCorruption() {
        // TODO Add your code to notify data are corrupted
        System.out.println("Personal data are corrupted");
    }
}
```
</td></tr>
<tr><td>Known use</td><td>Examples of existing systems that take the Data Quality principle into account include:

- Facebook (Website): When a user creates an account and does not specify all the required data or specifies incorrect data, the system will notify the user to correct the input.



</td></tr>
</table>

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Known use	 • Google (Website): When a user creates an account and does not specify all the required data or specifies incorrect data, the system will notify the user to correct the input. 
Related Patterns	Collection Limitation The Collection Limitation pattern requires the Data Quality pattern to ensure that the personal data that are collected are accurate, complete, and up-to-date for the processing of the system.

ตารางที่ ง.2 แบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล (ต่อ)

Related Patterns	Purpose Specification As the collected personal data will be considered of good quality if they are useful with regard to the purposes of use as specified by the data controller, the Data Quality pattern should be used with the Purpose Specification pattern. Individual Participation The Individual Participation pattern can be used with the Data Quality pattern to ensure that the data subject can participate in keeping his/her personal data up-to-date. Security Safeguards The Security Safeguards pattern requires the Data Quality pattern as preserving to ensure that the personal data that are collected are protected by reasonable security safeguards against loss or destruction and modification of data.

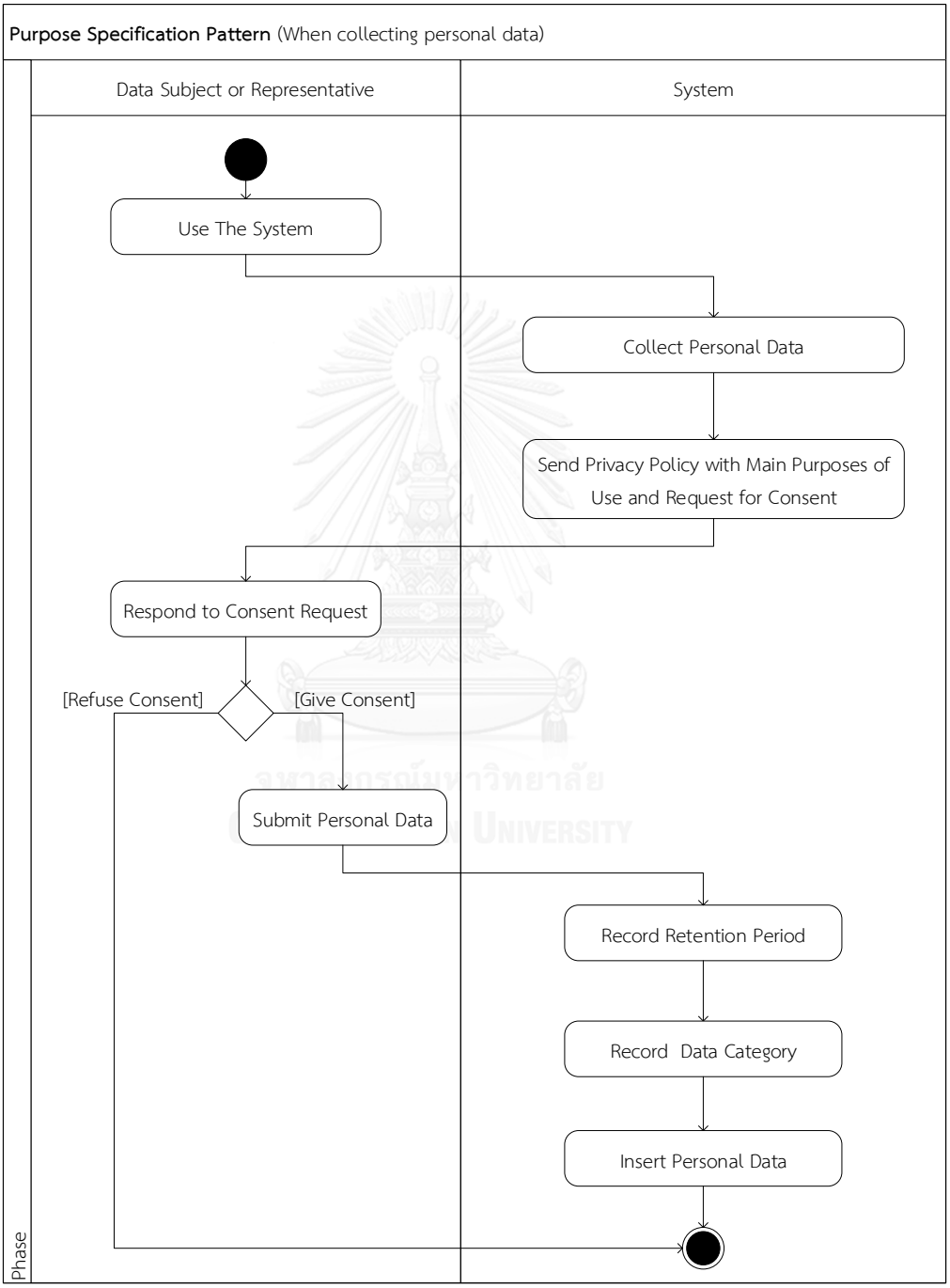
ง. 3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์

Pattern Name	**Purpose Specification Pattern**
Privacy Principle	Purpose Specification Principle
Intent	This pattern assures that the personal data for which the system are to be collected and used should have their purposes specified. Any change of purposes should also be specified.
Motivation	The data controller should bear in mind the purposes of the collected personal data and always use them in a way that corresponds to the purposes. Therefore, the purposes of collection and use should be specified and made known to the data subject no later than the collection time as well as during use, in the case of any changes of purposes. This also implies that, when the collected data are no longer serve any purposes, they should be removed, or at least anonymized to lower the risk of theft or unauthorized access when the data that are no longer of interest are still retained.
Applicability	Use the Purpose Specification pattern when the system is about to collect personal data and to change and end their use.

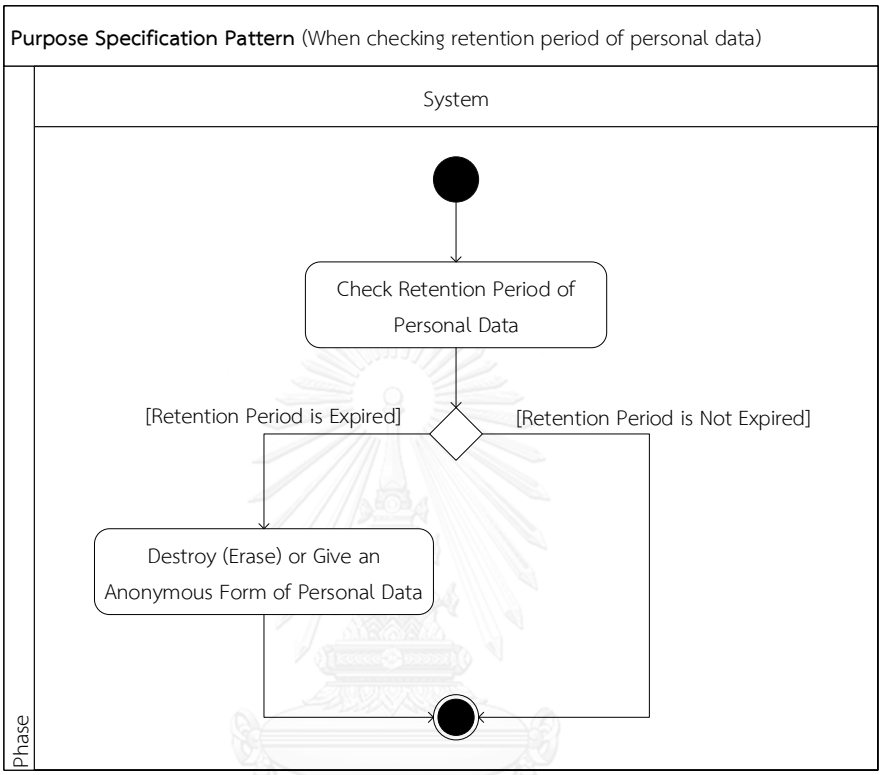
ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Behavior and Structure
Activity Diagram (When collecting personal data)

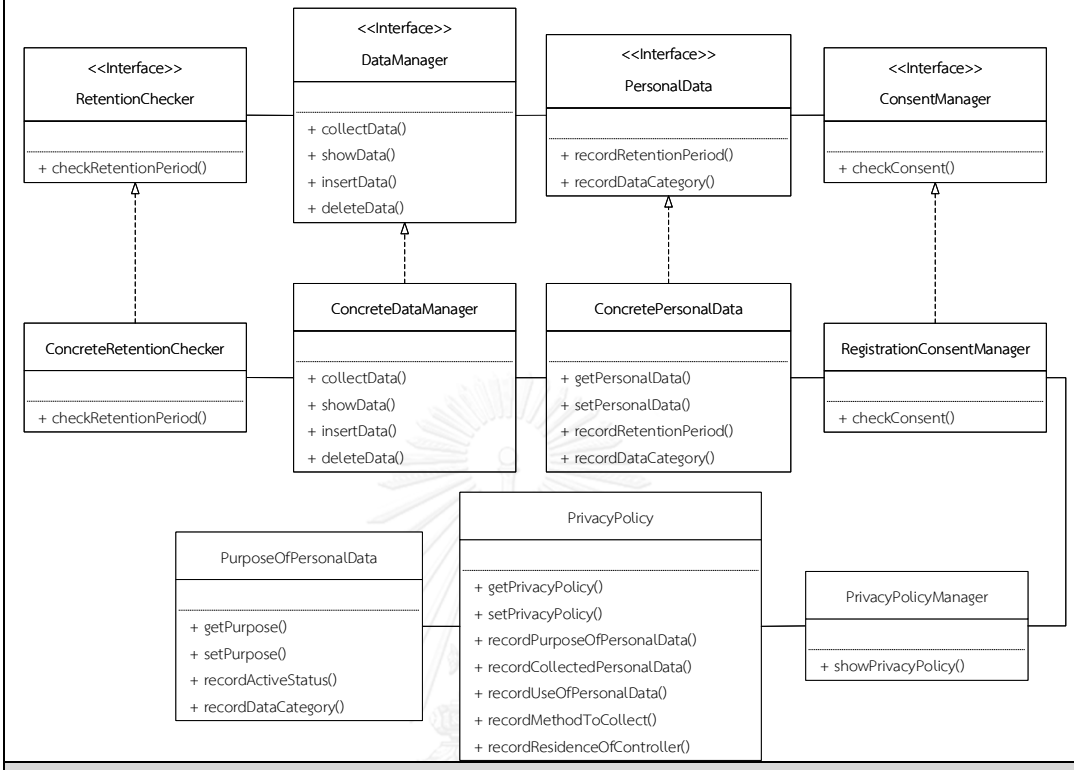


ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Behavior and Structure
Activity Diagram (When checking retention period of personal data)



ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

<table>
<tr><th colspan="2">Behavior and Structure</th></tr>
<tr><td colspan="2">Class Diagram
 
</td></tr>
<tr><th colspan="2">Participants</th></tr>
<tr><th>Interface Name/Class Name</th><th>Detail</th></tr>
<tr><td>Interface DataManager</td><td>An interface that has the responsibilities to coordinate checking of consent and collection of personal data from the data subject. It inserts the collected data into the system if the data subject gives consent.</td></tr>
<tr><td>Class ConcreteDataManager</td><td>A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. coordinating update of personal data when there are changes etc.</td></tr>
<tr><td>Interface PersonalData</td><td>An interface that represents the collected personal data.</td></tr>
<tr><td>Class ConcretePersonalData</td><td>A concrete class with the data attributes that are necessary for the purpose of the system.</td></tr>
</table>

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

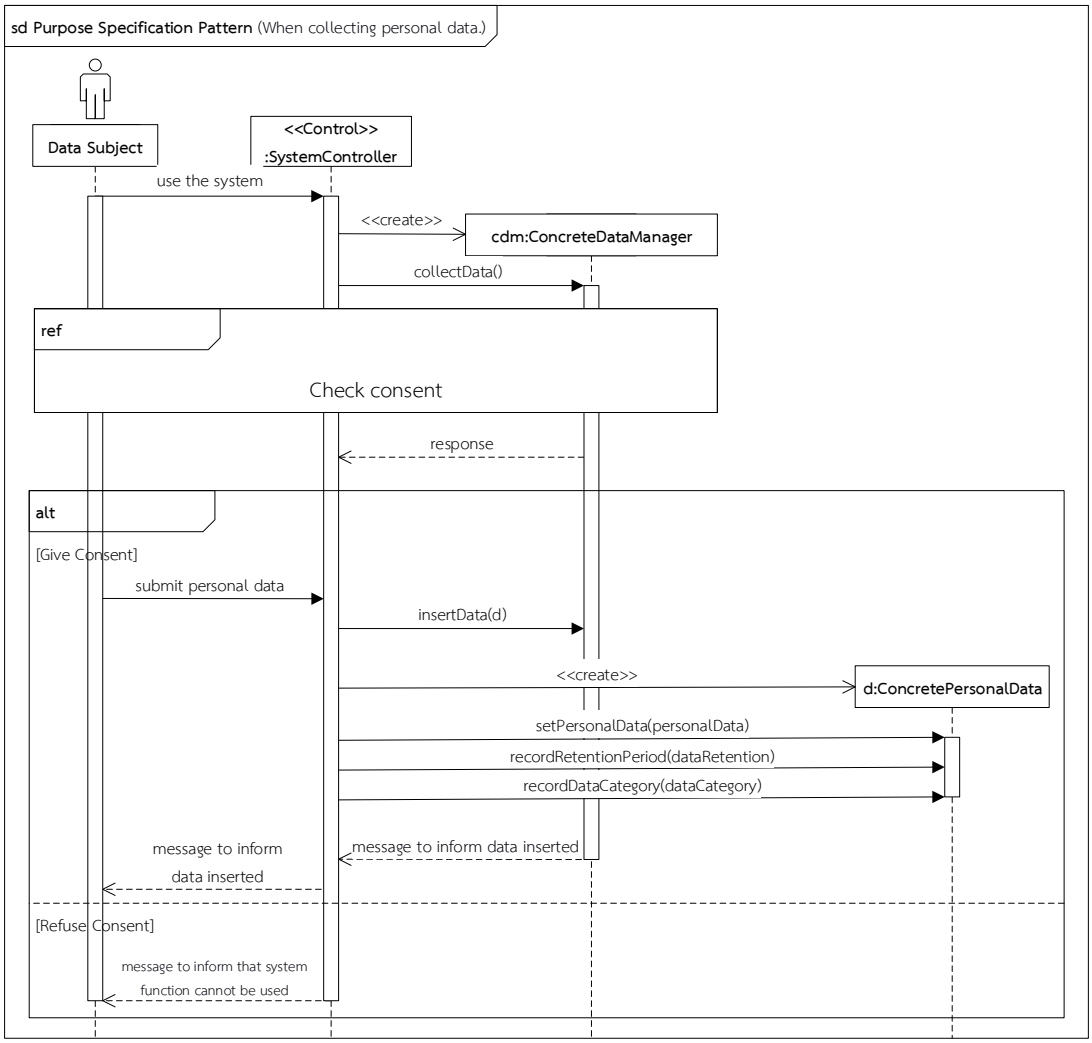
Participants	
Interface Name/Class Name	**Detail**
Interface ConsentManager	An interface that has the responsibilities to inform the data subject that the system needs to collect personal data and check whether the data subject gives consent.
Class ConcreteConsentManager	A concrete class that implements the interface ConsentManager.
Class PrivacyPolicyManager	A concrete class that has the responsibilities to show and update the privacy policy.
Class PrivacyPolicy	A concrete class that represents the privacy policy of the system with the data attributes that record the policy content, including the collection method, purpose of use, and data controller information.
Class PurposeOfPersonalData	A concrete class that represents the purpose of personal data with the data attributes that record the purpose of use of personal data.
Interface RetentionChecker	An interface that has the responsibilities to check the retention period of personal data.
Class ConcreteRetentionChecker	A concrete class that implements the interface RetentionChecker.
Collaboration	
1. When collecting personal data 1.1 The system controller creates a ConcreteDataManager object to collect personal data. 1.2 The ConcreteDataManager object checks for consent of the data subject: 1.2.1 The ConcreteDataManager object creates a ConcreteConsentManager object to check whether the data subject gives or refuse consent to collect his/her personal data. 1.2.2 The ConcreteConsentManager object creates a PrivacyPolicyManager object to view the privacy policy. The PrivacyPolicyManager object then creates a PrivacyPolicy object, gets the privacy policy from a data store, and sets the policy into the PrivacyPolicy object. 1.2.3 The PrivacyPolicy objects creates PurposeOfPersonalData objects and sets the purposes of use into them.	

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Collaboration
1.2.4 The PrivacyPolicyManager object sends the privacy policy to the ConcreteConsentManager object which in turn will pass on the policy to the data subject. 1.3 If the data subject gives consent, the system controller creates a ConcretePersonalData object with its retention period and data category, and then calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store. 2. When checking retention period of personal data 2.1 The system controller creates a ConcreteRetentionChecker object to check retention period of personal data. 2.2 The ConcreteRetentionChecker object creates a ConcreteDataManager object to get personal data from a data store. 2.3 The ConcreteDataManager object creates a ConcretePersonalData object and sets the personal data into the ConcretePersonalData object. 2.4 The ConcreteRetentionChecker object gets the retention period of personal data from the ConcretePersonalData object and check if it has expired. 2.5 If the retention period is expired, the ConcreteRetentionChecker object calls the ConcreteDataManager object to destroy (erase) personal data.

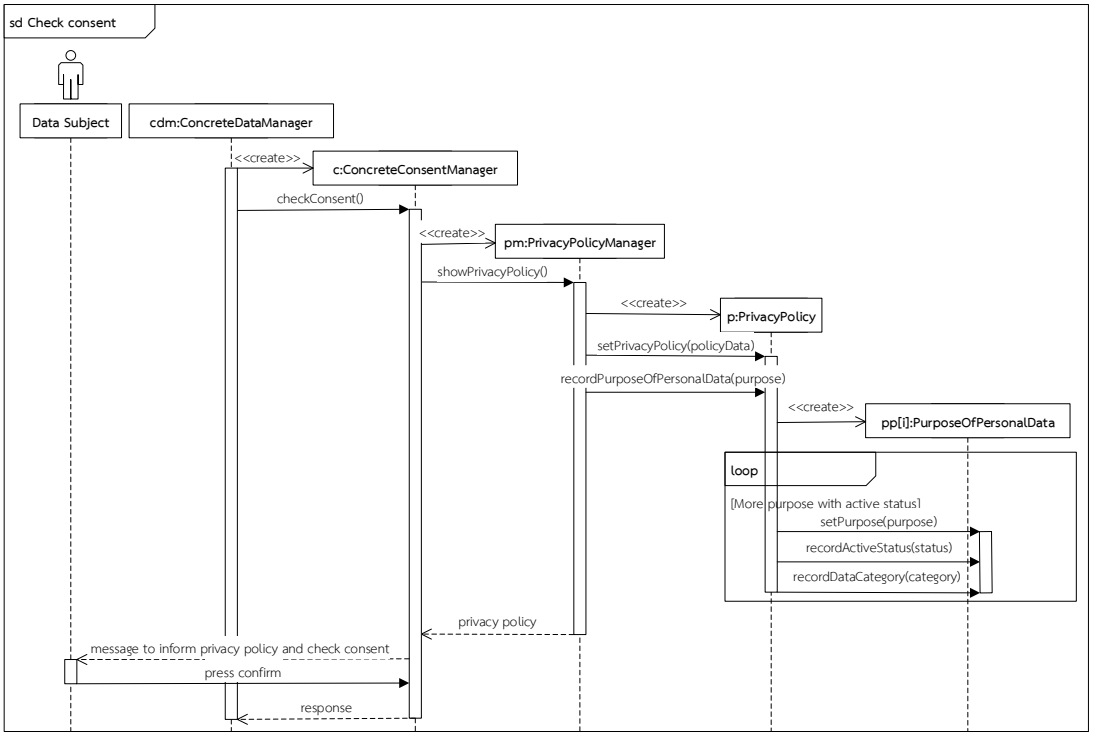
ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data)

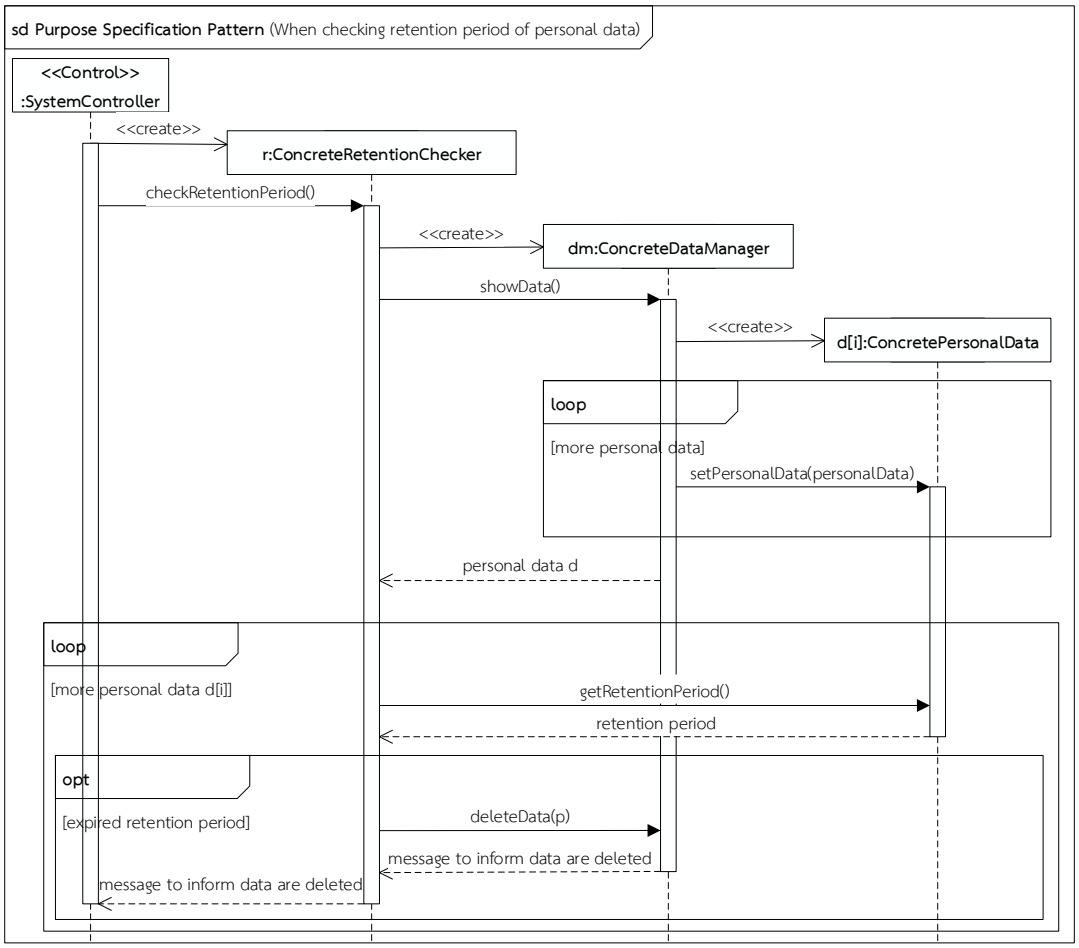


ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data - continued)



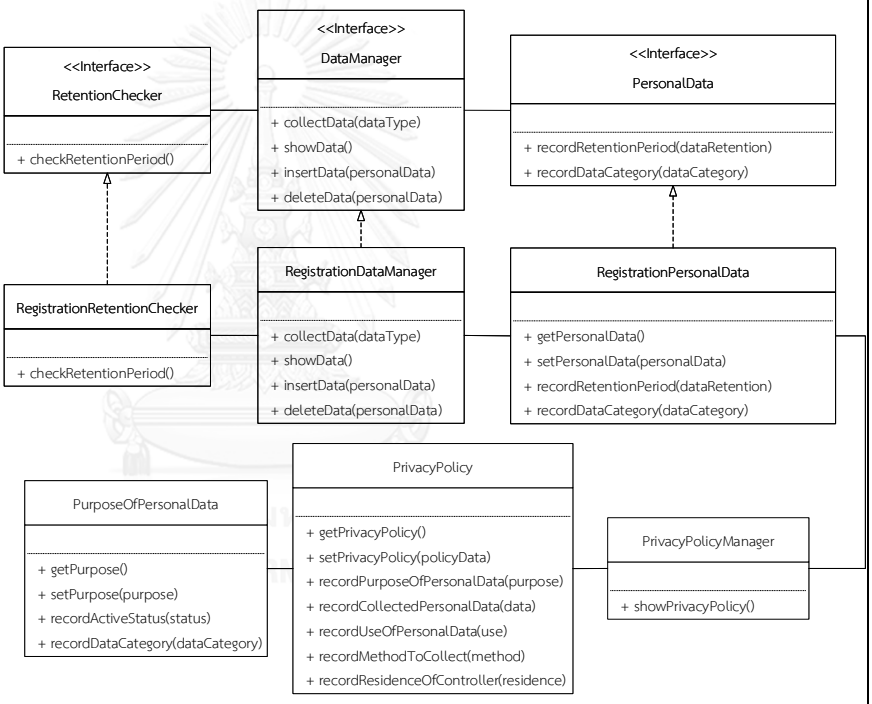
ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Collaboration	
Sequence Diagram (When checking retention period of personal data)  	
Consequences	Using this pattern, The personal data for which the system is to be collected and used have their purposes specified and any change of purposes be specified also.

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Implementation	Issues regarding the implementation of the Purpose Specification pattern are as follows. 1. The Purpose Specification principle is closely associated with Data Quality principle and the Use Limitation principle since the collected data should be relevant to the purpose so as to be considered useful and their use should be constrained by the purpose. Thus, the data controller who implements the system should refer to the Data Quality pattern and Use Limitation pattern also. 2. The OECD defines that the purposes of the collected personal data should be specified not later than at the time of data collection. The data controller who implements the system should inform the purpose when collecting personal data. The class ConcretePersonalData should record the category of the collected personal data, and the class PurposeOfPersonalData should record the purpose for which such collected category of data is used. This is to allow checking if the collected category of personal data is relevant to the purpose. 3. The OECD defines that the specification of purposes can be made in a number of alternative or complementary ways, e.g. by public declarations, information to data subjects, legislation, administrative decrees, and licenses provided by supervisory bodies. To address this issue, the data controller who implements the system should refer to the Openness pattern and specify the purposes in the published privacy policy. 4. The new purposes for which the personal data are subsequently used should be introduced as change of purposes while the change should not be in an arbitrary way. The data controller who implements the system should record the change of purposes while always maintaining compatibility between the original purposes and the changed purposes. In addition, the data controller should maintain a history of purposes. That is, the class PurposeOfPersonalData should record the status of purpose such as “inactive” for the purposes that were canceled or changed, and “active” for the latest purposes that are in use. The data controller should also refer to the Openness pattern to make the change of purposes known to the data subject.

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Implementation	5. When personal data no longer serve the purposes, the data controller who implements the system should make sure that the personal data are destroyed (erased) or given an anonymous form. The class ConcreteRetentionChecker can be called by the system whenever the data controller needs to check retention expiration. Otherwise, the system can be configured to call the class ConcreteRetentionChecker automatically to check retention expiration at a regular time interval.
Sample Design and Code	An example of a user registration system that applies the Purpose Specification pattern is shown below.  When a user (or data subject) is required to register with the system to use its service, the system uses the class RegistrationDataManager that implements the interface DataManager to collect personal data (name, email, tel) as shown in the sample code below.

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Sample Design and Code	

```
public interface DataManager {
    public boolean collectData(String dataCategory);
    public ArrayList<RegistrationPersonalData> showData();
    public void insertData(RegistrationPersonalData data);
    public void deleteData(RegistrationPersonalData data);
}

public class RegistrationDataManager implements DataManager{
    @Override
    public boolean collectData(String dataCategory){
        //TODO Add your code to check whatever user give or refuse consent
        boolean checker = false;
        RegistrationConsentManager consent = new RegistrationConsentManager();
        if(consent.checkConsent())
            checker = true;
        return checker;
    }
    @Override
    public ArrayList<RegistrationPersonalData> showData() {
        // TODO Add your code to get personal data from data store
        ArrayList<RegistrationPersonalData> data = new ArrayList<RegistrationPersonalData>();
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String query = "SELECT * FROM personaldata" ;
            PreparedStatement stmt = connection.prepareStatement(query) ;
            ResultSet result = stmt.executeQuery();
            while(result.next()){
                RegistrationPersonalData d = new RegistrationPersonalData();

                d.setName(result.getString("name"));
                d.setEmail(result.getString("email"));
                d.setTel(result.getString("tel"));
                d.setRetentionPeriod(result.getString("retention"));

                data.add(d);
            }
            stmt.close();
        } catch (SQLException e) {}
        return data;
    }
    @Override
    public void insertData(RegistrationPersonalData data){
        //TODO Add your code to insert personal data into data store.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String insert = "INSERT INTO personaldata "
                    + "(name, email, tel, retention, data_category) "
                    + "VALUES (?, ?, ?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert) ;
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, data.getName());
            stmt.setString(parameterIndex++, data.getEmail());
            stmt.setString(parameterIndex++, data.getTel());
            stmt.setString(parameterIndex++, data.getRetentionPeriod());
            stmt.setString(parameterIndex++, data.getDataCategory());

            stmt.execute();
            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
    }
```

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Sample Design and Code	``` @Override public void deleteData(RegistrationPersonalData data) { //TODO Add your code to delete personal data when data no longer serve a purpose. DatabaseConnection dbConnection = new DatabaseConnection(); Connection connection = dbConnection.getDBConnection(); try{ String insert = "DELETE FROM personaldata " + "where name = ? and email = ?" ; PreparedStatement stmt = connection.prepareStatement(insert) ; int parameterIndex = 1; stmt.setString(parameterIndex++, data.getName()); stmt.setString(parameterIndex++, data.getEmail()); stmt.execute(); stmt.close(); }catch(SQLException e) {} } } ``` When data are collected directly from the user, the class RegistrationConsentManager that implements the interface ConsentManager acquires the privacy policy with main purpose of use to inform the user. It also checks whether the user gives or refuse consent to personal data collection. The sample code is shown below. ``` public interface ConsentManager { public boolean checkConsent(); } public class RegistrationConsentManager implements ConsentManager{ private Scanner scan; @Override public boolean checkConsent() { /* TODO Add your code to show policy to user and check whatever * user give or refuse consent */ scan = new Scanner(System.in); PrivacyPolicyManager policyManager = new PrivacyPolicyManager(); PrivacyPolicy policy = policyManager.showPrivacyPolicy(); System.out.println("Name :: "+policy.getName()); System.out.println("Collected Data :: "+policy.getDataCollected()); System.out.println("Use Data :: "+policy.getDataUse()); System.out.println("Collected Method :: "+policy.getCollectionMethod()); for(int i=0; i<policy.getPurpose().size(); i++){ System.out.println("Purpose "+i+" :: " +policy.getPurpose().get(i).getDetail()); } System.out.println("Residence :: "+policy.getResidenceOfController()); System.out.println("Give Consent Please Enter 'Y' " + "and Refuse Consent Please Enter 'N'"); String accept = scan.next(); if(accept.equalsIgnoreCase("Y")) return true; else return false; } } ``` When a data controller wants to show or update (or change) the privacy policy, the system uses the class PrivacyPolicyManager to show or update the policy of the system.

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Sample Design and Code

```
public class PrivacyPolicyManager {

    public PrivacyPolicy showPrivacyPolicy() {
        // TODO Add your code to show privacy policy
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        PrivacyPolicy pp = new PrivacyPolicy();
        PreparedStatement stmt;
        try {
            String query1 = "SELECT * FROM privacypolicy ORDER BY id DESC LIMIT 1"
            stmt = connection.prepareStatement(query1) ;
            ResultSet  result1 = stmt.executeQuery();
            while(result1.next()){
                pp.setId(result1.getString("id"));
                pp.setName(result1.getString("name"));
                pp.setDataCollected(result1.getString("collecteddata"));
                pp.setDataUse(result1.getString("usedata"));
                pp.setCollectionMethod(result1.getString("collectedmethod"));
                pp.setResidenceOfController(result1.getString("residence"));
            }

           JSONArray jsonArr = new JSONArray();
           String query2 = "SELECT * FROM purpose where active_status = 'Yes' " ;
           stmt = connection.prepareStatement(query2) ;
           ResultSet result2 = stmt.executeQuery();
           while(result2.next()){
               JSONObject json = new JSONObject();
               json.put("id", result2.getString("id"));
               json.put("detail", result2.getString("detail"));
               json.put("data_category", result2.getString("data_category"));
               json.put("active_status", result2.getString("active_status"));
               jsonArr.put(json);
           }
           pp.recordPurposeOfPersonalData(jsonArr);
           stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
          catch (JSONException e) {e.printStackTrace();}
        return pp;
    }
}
```

The class PrivacyPolicy is implemented to have the data attributes that record the policy content, including the purpose of collection, what personal data are collected, use of personal data, collection method, and data controller information as shown in the sample code below.

```
public class PrivacyPolicy {
    private String id;
    private String name;
    private String detail;
    private ArrayList<PurposeOfPersonalData> purpose;
    private String dataCollected;
    private String dataUse;
    private String collectionMethod;
    private String residenceOfController;
    private Date dateCreation;
    private Date dateUpdate;

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getDetail() {
        return detail;
    }
    public void setDetail(String detail) {
        this.detail = detail;
    }
```

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Sample Design and Code	``` public ArrayList<PurposeOfPersonalData> getPurpose() { return purpose; } public void setPurpose(ArrayList<PurposeOfPersonalData> purpose) { this.purpose = purpose; } public String getDataCollected() { return dataCollected; } public void setDataCollected(String dataCollected) { this.dataCollected = dataCollected; } public String getDataUse() { return dataUse; } public void setDataUse(String dataUse) { this.dataUse = dataUse; } public String getCollectionMethod() { return collectionMethod; } public void setCollectionMethod(String collectionMethod) { this.collectionMethod = collectionMethod; } public String getResidenceOfController() { return residenceOfController; } public void setResidenceOfController(String residenceOfController) { this.residenceOfController = residenceOfController; } public Date getDateCreation() { return dateCreation; } public void setDateCreation(Date dateCreation) { this.dateCreation = dateCreation; } public Date getDateUpdate() { return dateUpdate; } public void setDateUpdate(Date dateUpdate) { this.dateUpdate = dateUpdate; } public void recordPurposeOfPersonalData(JSONArray purpose) { //TODO Add your code to record purpose this.purpose = new ArrayList<PurposeOfPersonalData>(); for(int i=0; i<purpose.length(); i++){ PurposeOfPersonalData p = new PurposeOfPersonalData(); try { p.setId(purpose.getJSONObject(i).getString("id")); p.setDetail(purpose.getJSONObject(i).getString("detail")); p.recordDataCategory(purpose.getJSONObject(i).getString("data_category")); p.recordActiveStatus(purpose.getJSONObject(i).getString("active_status")); } catch (JSONException e) {e.printStackTrace();} this.purpose.add(p); } } public void recordCollectedPersonalData(String data) { //TODO Add your code to record collected personal data this.dataCollected = data; } public void recordUseOfPersonalData(String use) { //TODO Add your code to record use of personal data this.dataUse = use; } public void recordMethodToCollect(String method) { //TODO Add your code to record method to collect personal data this.collectionMethod = method; } public void recordResidenceOfController(String residence) { //TODO Add your code to record residence of data controller this.residenceOfController = residence; } } ``` The class PurposeForPersonalData is implemented to have the data attributes that store the purpose of the collection and use of personal data. The sample code is shown below.

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Sample Design and Code

```
public class PurposeOfPersonalData {
    private String id;
    private String detail;
    private String dataCategory;
    private String activeStatus;
    private Date dateCreation;
    private Date dateUpdate;

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getDetail() {
        return detail;
    }
    public void setDetail(String detail) {
        this.detail = detail;
    }
    public String getDataCategory() {
        return dataCategory;
    }
    public String getActiveStatus() {
        return activeStatus;
    }
    public Date getDateCreation() {
        return dateCreation;
    }
    public void setDateCreation(Date dateCreation) {
        this.dateCreation = dateCreation;
    }
    public Date getDateUpdate() {
        return dateUpdate;
    }
    public void setDateUpdate(Date dateUpdate) {
        this.dateUpdate = dateUpdate;
    }
    public void recordActiveStatus(String activeStatus) {
        // // TODO Add your code to record purpose status
        this.activeStatus = activeStatus;
    }
    public void recordDataCategory(String dataCategory) {
        // TODO Add your code to record category of collected personal data
        this.dataCategory = dataCategory;
    }
}
```

The class RegistrationPersonalData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of the system as shown in the sample code below.

```
public interface PersonalData {
    public void recordRetentionPeriod(String dataRetention);
    public void recordDataCategory(String dataCategory);
}
public class RegistrationPersonalData implements PersonalData{
    private String name;
    private String email;
    private String tel;
    private String retentionPeriod;
    private String dataCategory;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
```

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

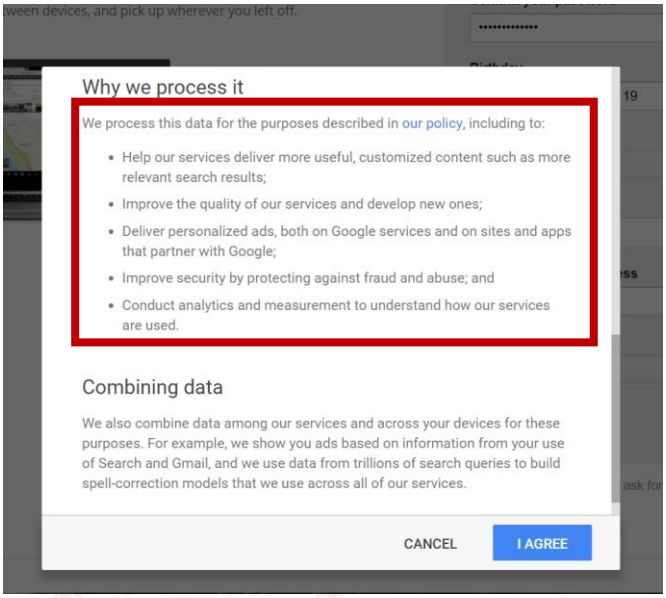
Sample Design and Code	

```
    public String getTel() {
        return tel;
    }
    public void setTel(String tel) {
        this.tel = tel;
    }
    public String getRetentionPeriod() {
        return retentionPeriod;
    }
    public void setRetentionPeriod(String retentionPeriod) {
        this.retentionPeriod = retentionPeriod;
    }
    public String getDataCategory() {
        return dataCategory;
    }
    @Override
    public void recordRetentionPeriod(String dRetention) {
        // TODO Add your code to record retention period of collected personal data
        if(dRetention.equalsIgnoreCase("Registration"))
            this.retentionPeriod = "Until delete account";
        else if(dRetention.equalsIgnoreCase("Location"))
            this.retentionPeriod = "Until end program";
        else
            this.retentionPeriod = "Until deleting data";
    }

    @Override
    public void recordDataCategory(String dataCategory) {
        // TODO Add your code to record type of collected personal data
        this.dataCategory = dataCategory;
    }
}
```

The class RegistrationRetentionChecker that implements the interface RetentionChecker is implemented to check retention period. If retention period of personal data is expired, the system should destroy (erase) or give an anonymous form of personal data. An example shows erase personal data when not serve the purpose.

```
public interface RetentionChecker {
    public void checkRetentionPeriod();
}


public class RegistrationRetentionChecker implements RetentionChecker{

    @Override
    public void checkRetentionPeriod() {
        // TODO Add your code to check retention period
        RegistrationDataManager dataManager = new RegistrationDataManager();
        ArrayList<RegistrationPersonalData> data = dataManager.showData();
        for(int i=0; i<data.size(); i++){
            String retention = data.get(i).getRetentionPeriod();
            if(retention.equals("An expired retention")){
                dataManager.deleteData(data.get(i));
            }
        }
    }
}
```

ตารางที่ ง.3 แบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

Known use	Examples of existing systems that take the Purpose Specification principle into account include: ● Google (Website): When a user creates an account, the system will collect personal data and inform the user of the privacy policy that describes how the system process his/her personal data. If the user gives consent to the system to collect and use personal data, the user should click the agreement button. 
Related Patterns	Data Quality The Data Quality pattern requires the Purpose Specification pattern to ensure that the collected personal data will be considered of good quality if they are useful with regard to the purposes of use as specified by the data controller. Use Limitation The Use Limitation pattern requires the Purpose Specification pattern to ensure that the use and disclosure of the personal data should always be limited to the purpose specification. Openness. The Openness pattern can be used with the Purpose Specification pattern to ensure that the information about purpose of use should always be open to the data subject.

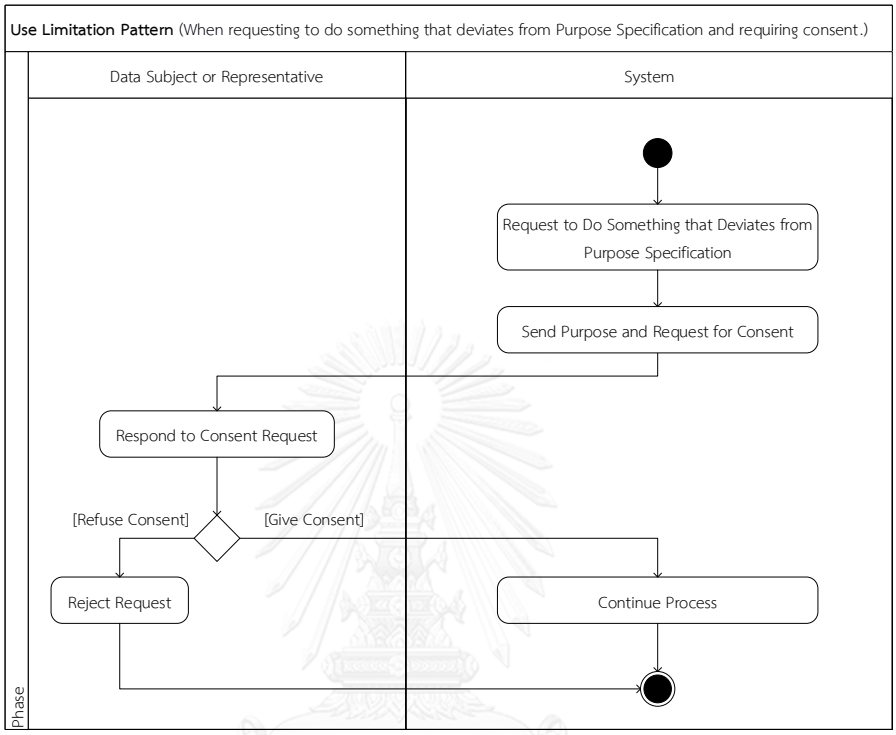
ง. 4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล

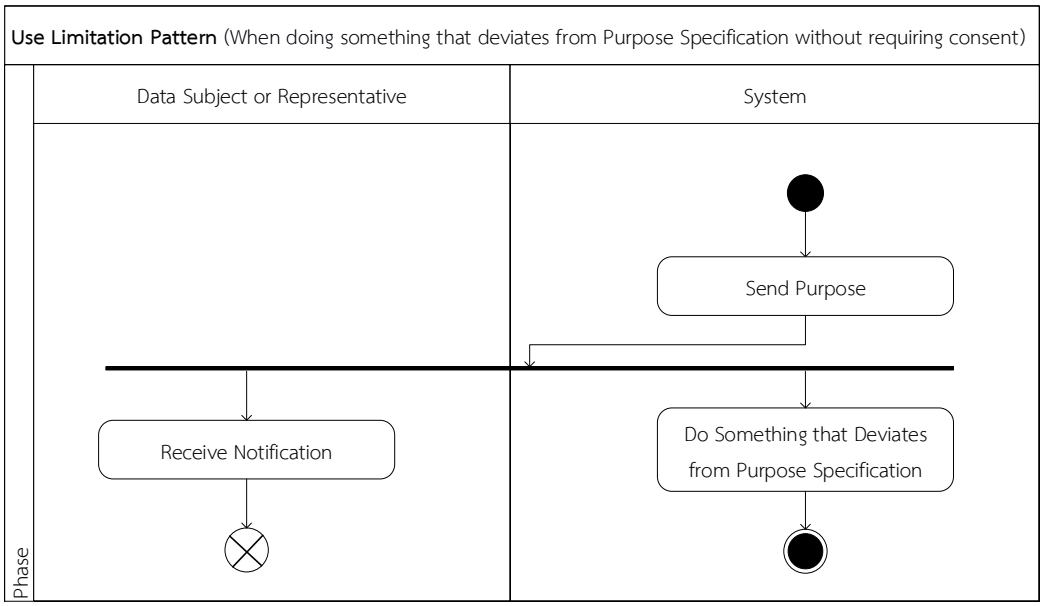
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล

Pattern Name	**Use Limitation Pattern**
Privacy Principle	Use Limitation Principle
Intent	This pattern assures that the collected personal data should not be disclosed, made available, or otherwise used for purposes other than the initially or subsequently specified purposes. Deviations from the specified purposes are possible but should be in a limited way.
Motivation	In normal circumstances, the collected personal data should be put to use for the specified purposes, but sometimes the disclosure and use of collected personal data may deviate from the specified purposes. For example, the data may be transmitted from one computer to another and therefore disclosed and used for unauthorised purposes without knowledge of the data subject. In some cases, the data collected initially for operational or administrative purposes may be required for further research, statistics, and social planning. It is also often the case that the collected data are disclosed for investigation by the authority of law. Such deviations may be allowed but consent from the data subject or its representative should be given prior to personal data disclosure. In the case of the use by the authority of law or supervisory bodies where consent is not required, the data controller may inform the data subject.
Applicability	Use the Use Limitation pattern when the collected personal data may be used in a way that deviates from the specified purposes.

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

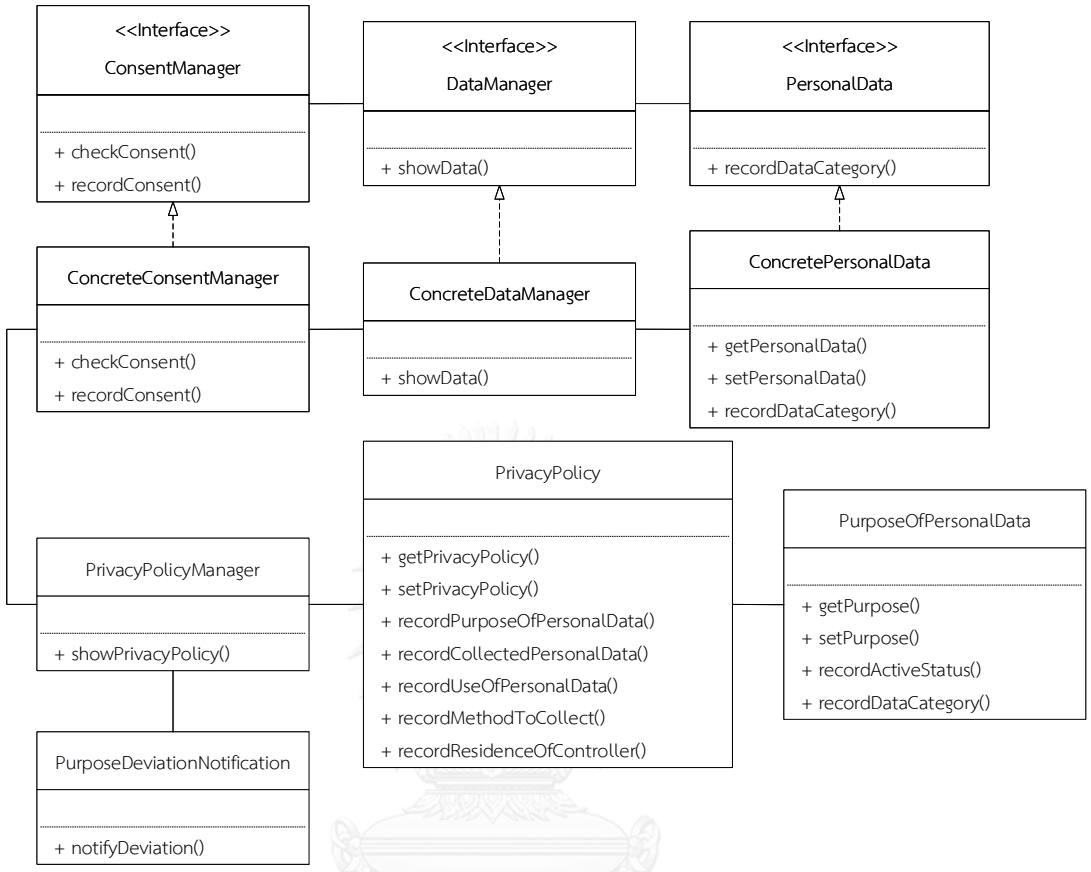
Behavior and Structure
Activity Diagram (When requesting to do something that deviates from Purpose Specification and requiring consent.)
Activity Diagram (When doing something that deviates from Purpose Specification without requiring consent)





ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Class Diagram



Participants	
Interface Name/Class Name	**Detail**
Interface DataManager	An interface that has the responsibilities to coordinate checking of consent and manage personal data.
Class ConcreteDataManager	A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. collect or update of personal data.
Interface ConsentManager	An interface that has the responsibilities to inform the data subject that the system needs to transmit personal data and check whether the data subject gives or refuses consent.
Class ConcreteConsentManager	A concrete class that implements the interface ConsentManager.

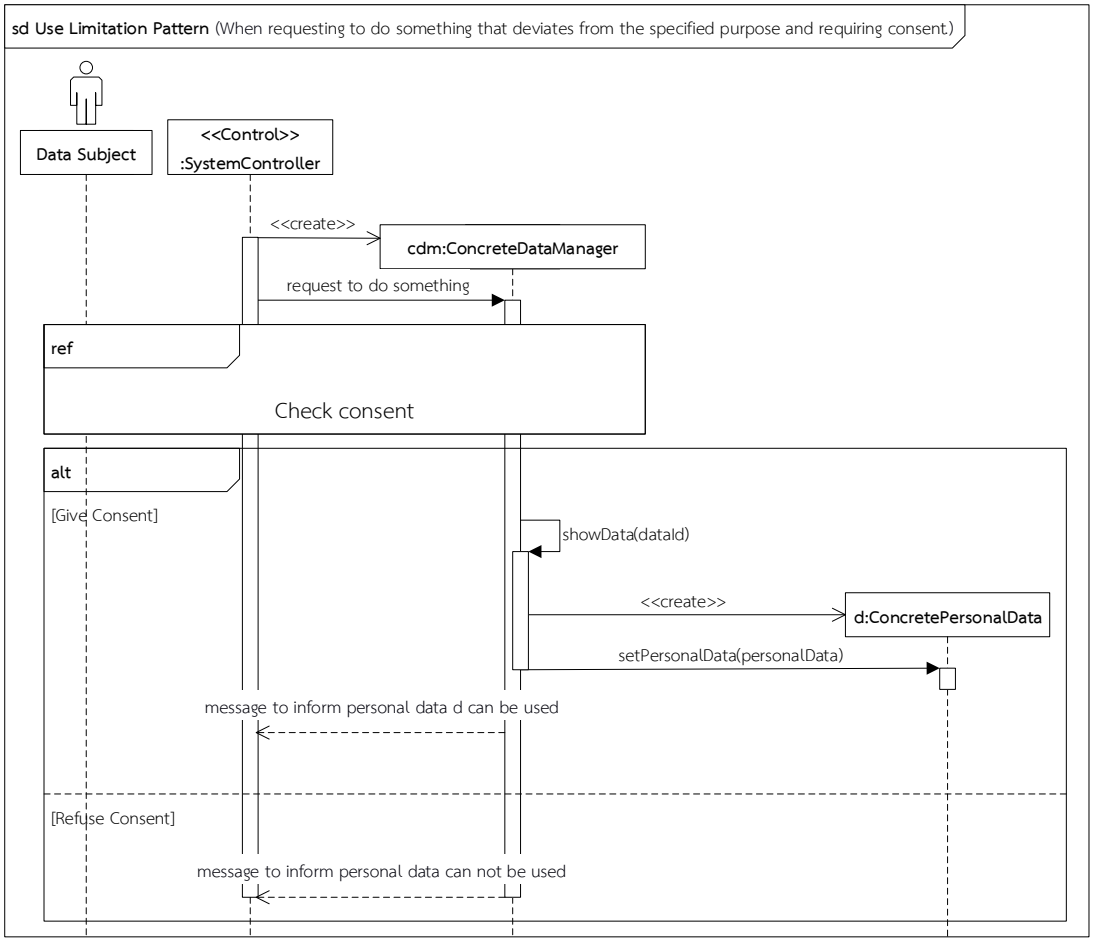
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Participants	
Interface Name/Class Name	**Detail**
Interface PersonalData	A concrete that represents the collected personal data.
Class ConcretePersonalData	An object class with the data attributes that are necessary for the purpose of the system.
Class PrivacyPolicyManager	A concrete class that has the responsibilities to show and update the privacy policy.
Class PrivacyPolicy	A concrete class that represents the privacy policy of the system with the data attributes that record the policy content, including the collection method, purpose of use, and data controller information.
Class PurposeOfPersonalData	A concrete class that represents the purpose of personal data with the data attributes that record the purpose of use of personal data.
Class PurposeDeviationNotification	A concrete class that has the responsibilities to notify the data subject when the data controller requires to do something that deviates from the specified purpose without requiring consent.

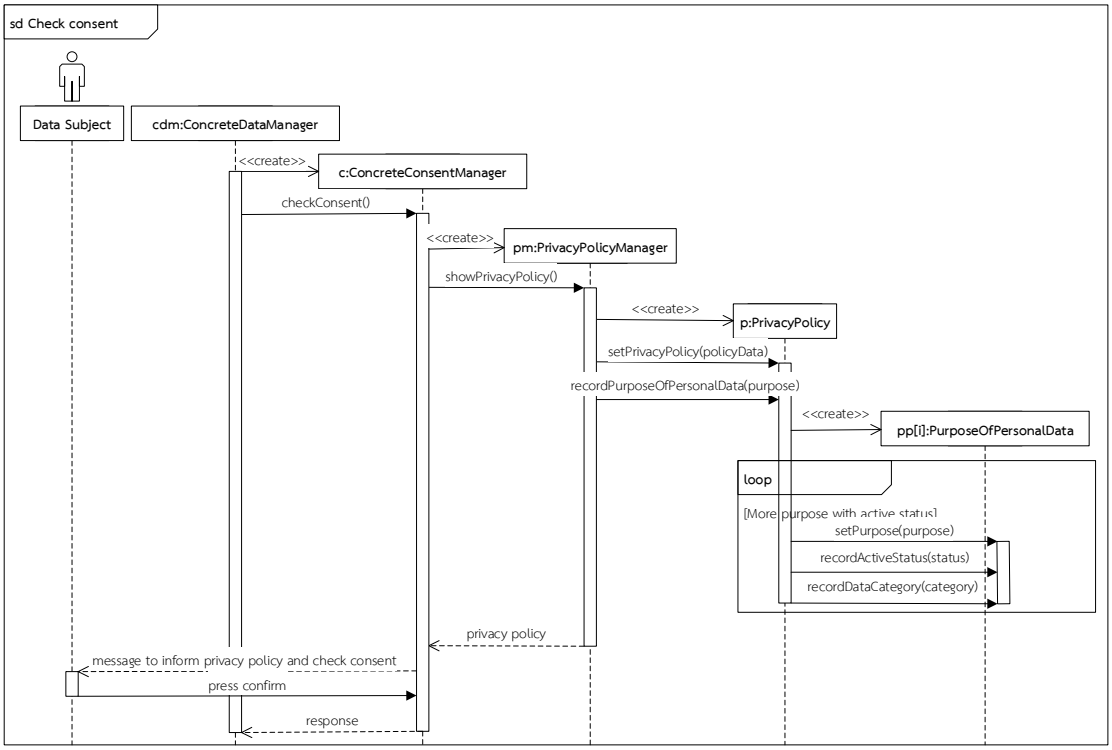
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Collaboration
1. When requesting to do something that deviates from the specified purpose and requiring consent, 1.1. The system controller creates a ConcreteDataManager object to use personal data. 1.2. The ConcreteDataManager object checks for consent of the data subject: 1.2.1. The ConcreteDataManager object creates a ConcreteConsentManager object to check whether the data subject gives or refuse consent to use his/her personal data. 1.2.2. The ConcreteConsentManager object creates a PrivacyPolicyManager object to view the privacy policy. The PrivacyPolicyManager object then creates a PrivacyPolicy object, gets the privacy policy from a data store, and sets the policy into the PrivacyPolicy object. 1.2.3. The PrivacyPolicy objects creates a PurposeOfPersonalData objects and sets the purposes of use into them. 1.2.4. The PrivacyPolicyManager object sends the privacy policy to the ConcreteConsentManager object which in turn will pass on the policy to the data subject. 1.2.5. If consent is given by the data subject, the ConcreteConsentManager object records the consent information. 1.3. If the data subject gives consent, 1.3.1. The ConcreteDataManager object then creates a ConcretePersonalData object, get the personal data from a data store, sets the personal data into the ConcretePersonalData object which in turn will be sent to the system controller. 2. When doing something that deviates from the specified purpose without requiring consent. 2.1. The system controller creates a PurposeDeviationNotification object to notify the data subject about the use of personal data which deviates from the specified purpose. 2.2. The PurposeDeviationNotification object shows the privacy policy: 2.2.1. The PurposeDeviationNotification object creates a PrivacyPolicyManager object to view the privacy policy. The PrivacyPolicyManager object then creates a PrivacyPolicy object, gets the privacy policy from a data store, and sets the policy into the PrivacyPolicy object. 2.2.2. The PrivacyPolicy objects creates PurposeOfPersonalData objects and sets the purposes of use into them.

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

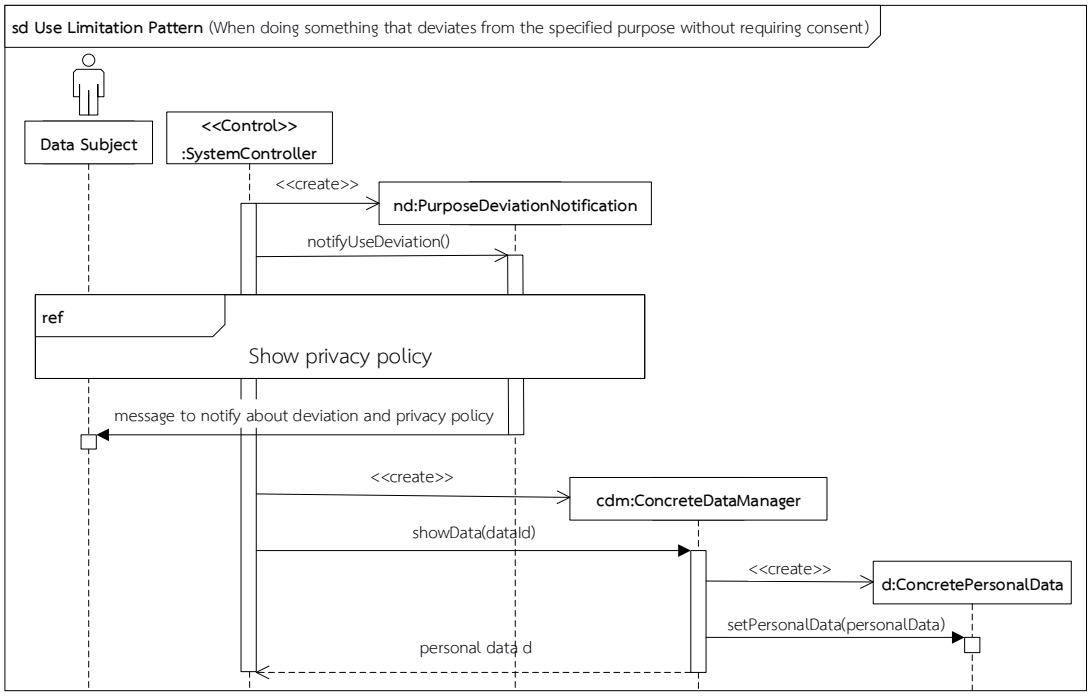
Collaboration
2.2.3 The PrivacyPolicyManager object sends the privacy policy to the PurposeDeviationNotification object. 2.3 The PurposeDeviationNotification object sends the privacy policy to the data subject. 2.4 The system controller creates a ConcreteDataManager object to view personal data. The ConcreteDataManager object then creates a ConcretePersonalData object, get the personal data from a data store, sets the personal data into the ConcretePersonalData object which in turn will be sent to the system controller.
Sequence Diagram (When requesting to do something that deviates from the specified purpose and requiring consent.) 

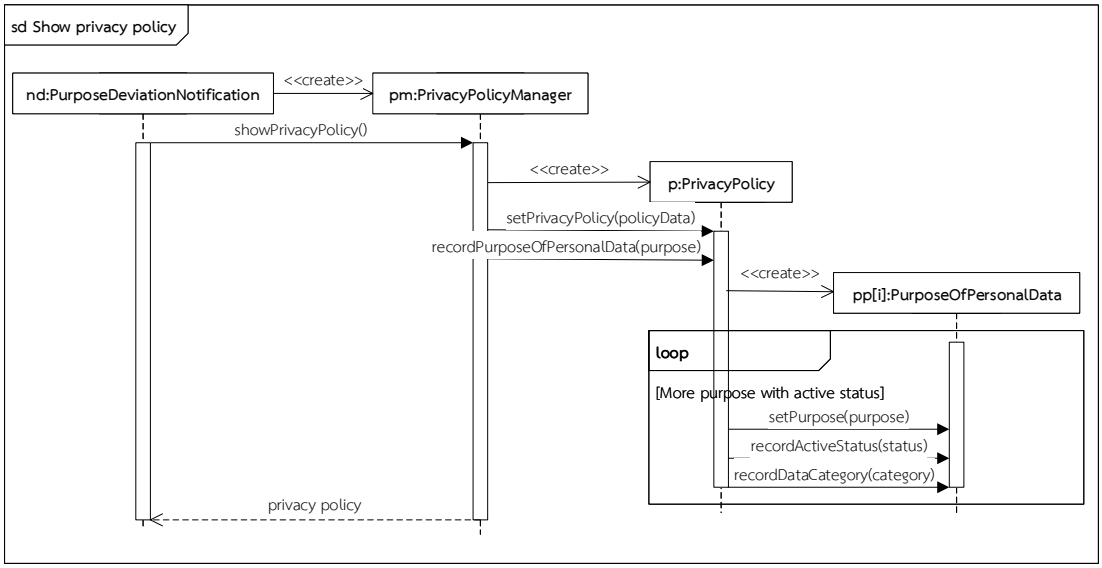
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Collaboration
Sequence Diagram (When requesting to do something that deviates from the specified purpose and requiring consent. - continued) 

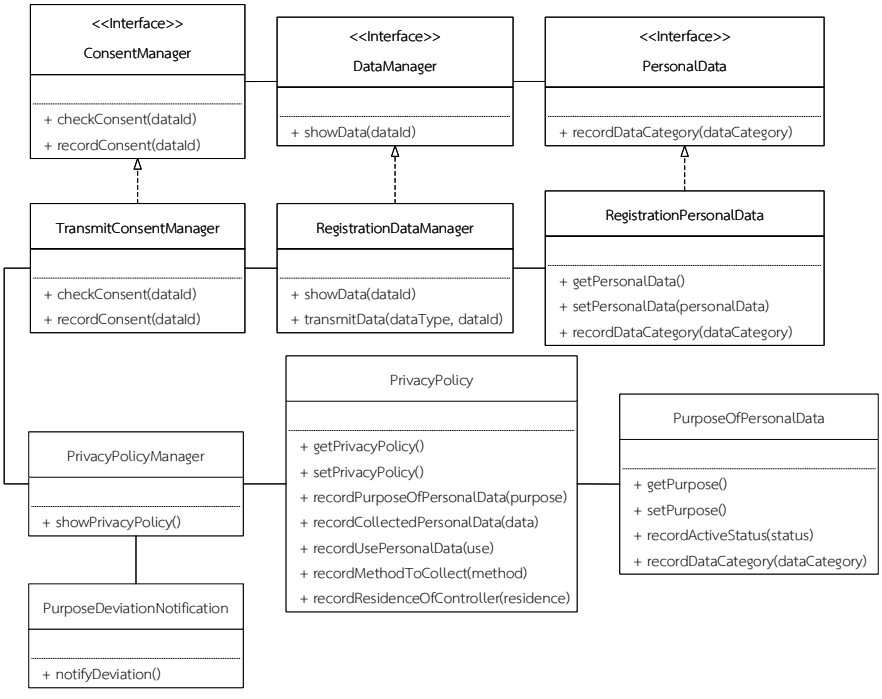
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Collaboration
Sequence Diagram (When doing something that deviates from the specified purpose without requiring consent)





ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Consequences	Using this pattern, the system of the data controller can ensure that the use of the data subject's personal data is well-controlled within the limit of the specified purposes as recommended by OECD. Even in the case that the personal data are to be used in certain exception situations that are not intended by the initial purposes, the data subject is further requested for consent or at least informed.
Implementation	Issues regarding the implementation of the Use Limitation pattern are as follows. 1. The OECD defines that the collected personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with Purpose Specification Principle. To address this issue, the data controller should refer to the Purpose Specification pattern and Openness pattern to maintain the purposes in a privacy policy. 2. The data controller should always consult the privacy policy prior to the use of the collected personal data to determine whether such use will deviate from the specified purposes and, if so, whether consent will be needed.
Sample Design and Code	An example below shows the case of a system that requires to transmit personal data for use in another computer where such use is not specified in the initial purpose of use. 

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Sample Design and Code	When requiring to transmit personal registration data of the data subject to another system, the system uses the class RegistrationDataManager that implements the interface DataManager to transmit personal data as shown in the sample code below.

```
public interface DataManager {
    public RegistrationPersonalData showData(String dataId);
}
public class RegistrationDataManager implements DataManager{

    @Override
    public RegistrationPersonalData showData(String dataId) {
        // TODO Add your code to get personal data from data store
        RegistrationPersonalData data = new RegistrationPersonalData();
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String query = "SELECT * FROM personaldata WHERE id = ?" ;
            PreparedStatement stmt = connection.prepareStatement(query) ;
            stmt.setInt(1, Integer.parseInt(dataId));
            ResultSet result = stmt.executeQuery();
            while(result.next()){
                data.setName(result.getString("name"));
                data.setEmail(result.getString("email"));
                data.setTel(result.getString("tel"));
                data.recordDataCategory(result.getString("data_category"));
            }
            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
        return data;
    }

    public RegistrationPersonalData transmitData(String dataType, String dataId) {
        /* TODO Add your code to check whatever user give or refuse consent to
         * transmit personal data from one computer to another.
         */
        TransmissionConsentManager consent = new TransmissionConsentManager();
        if(consent.checkConsent(dataId)){
            RegistrationPersonalData data = showData(dataId);
            return data;
        }
        return null;
    }
}
```

The system is required to transmit personal data from one computer to another, the class TransmissionConsentManager that implements the interface ConsentManager is used to check whether the user gives or refuses consent to transmit personal data as shown in the sample code below.

```
public interface ConsentManager {
    public boolean checkConsent(String dataId);
    public void recordConsent(String dataId);
}
```

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Sample Design and Code	``` public class TransmissionConsentManager implements ConsentManager{ private Scanner scan; private SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd"); private Date date = new Date(); @Override public boolean checkConsent(String dataId) { /* TODO Add your code to inform user about personal data are * transmitted from one computer to another i.e. personal data * are transmitted to external agent. */ scan = new Scanner(System.in); System.out.println("The system will transmit your personal data."); SystemPrivacyPolicyManager policyManager = new SystemPrivacyPolicyManager(); SystemPrivacyPolicy policy = policyManager.showPrivacyPolicy(); System.out.println("Name :: "+policy.getName()); System.out.println("Collected Data :: "+policy.getDataCollected()); System.out.println("Use Data :: "+policy.getDataUse()); System.out.println("Collected Method :: "+policy.getCollectionMethod()); for(int i=0; i<policy.getPurpose().size(); i++) System.out.println("Purpose "+i+" :: " + policy.getPurpose().get(i).getDetail()); System.out.println("Residence :: "+policy.getResidenceOfController()); System.out.println("Give Consent Please Enter 'Y' " + "and Refuse Consent Please Enter 'N'"); String accept = scan.next(); if(accept.equalsIgnoreCase("Y")){ recordConsent(dataId); return true; } else return false; } @Override public void recordConsent(String dataId) { // TODO Add your code to record about consent. Logger logger = Logger.getLogger(TransmissionConsentManager.class.getName()); FileHandler fh; try { fh = new FileHandler("D:/log/event-transmission-" + "consent"+sdf.format(date)+".log",true); logger.addHandler(fh); logger.info("Data ID "+dataId+" are consent to transmit"); }catch (IOException e) { e.printStackTrace(); } } } ```
	When a data controller wants to show the privacy policy, the system uses the class PrivacyPolicyManager to show the policy of the system.

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Sample Design and Code	

```
public class PrivacyPolicyManager {

    public PrivacyPolicy showPrivacyPolicy() {
        // TODO Add your code to show privacy policy
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        PrivacyPolicy pp = new PrivacyPolicy();
        PreparedStatement stmt;
        try {
            String query1 = "SELECT * FROM privacypolicy ORDER BY id DESC LIMIT 1"
            stmt = connection.prepareStatement(query1) ;
            ResultSet  result1 = stmt.executeQuery();
            while(result1.next()){
                pp.setId(result1.getString("id"));
                pp.setName(result1.getString("name"));
                pp.setDataCollected(result1.getString("collecteddata"));
                pp.setDataUse(result1.getString("usedata"));
                pp.setCollectionMethod(result1.getString("collectedmethod"));
                pp.setResidenceOfController(result1.getString("residence"));
            }

           JSONArray jsonArr = new JSONArray();
           String query2 = "SELECT * FROM purpose where active_status = 'Yes' " ;
           stmt = connection.prepareStatement(query2) ;
           ResultSet result2 = stmt.executeQuery();
           while(result2.next()){
               JSONObject json = new JSONObject();
               json.put("id", result2.getString("id"));
               json.put("detail", result2.getString("detail"));
               json.put("data_category", result2.getString("data_category"));
               json.put("active_status", result2.getString("active_status"));
               jsonArr.put(json);
           }
           pp.recordPurposeOfPersonalData(jsonArr);
           stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
          catch (JSONException e) {e.printStackTrace();}
        return pp;
    }
}
```

The class PrivacyPolicy is implemented to have the data attributes that record the policy content, including the purpose of collection, what personal data are collected, use of personal data, collection method, and data controller information as shown in the sample code below.

```
public class PrivacyPolicy {
    private String id;
    private String name;
    private String detail;
    private ArrayList<PurposeOfPersonalData> purpose;
    private String dataCollected;
    private String dataUse;
    private String collectionMethod;
    private String residenceOfController;
    private Date dateCreation;
    private Date dateUpdate;
```

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Sample Design and Code	

```
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getDetail() {
    return detail;
}
public void setDetail(String detail) {
    this.detail = detail;
}
public ArrayList<PurposeOfPersonalData> getPurpose() {
    return purpose;
}
public void setPurpose(ArrayList<PurposeOfPersonalData> purpose) {
    this.purpose = purpose;
}
public String getDataCollected() {
    return dataCollected;
}
public void setDataCollected(String dataCollected) {
    this.dataCollected = dataCollected;
}
public String getDataUse() {
    return dataUse;
}
public void setDataUse(String dataUse) {
    this.dataUse = dataUse;
}
public String getCollectionMethod() {
    return collectionMethod;
}
public void setCollectionMethod(String collectionMethod) {
    this.collectionMethod = collectionMethod;
}
public String getResidenceOfController() {
    return residenceOfController;
}
public void setResidenceOfController(String residenceOfController) {
    this.residenceOfController = residenceOfController;
}
public Date getDateCreation() {
    return dateCreation;
}
public void setDateCreation(Date dateCreation) {
    this.dateCreation = dateCreation;
}
public Date getDateUpdate() {
    return dateUpdate;
}
public void setDateUpdate(Date dateUpdate) {
    this.dateUpdate = dateUpdate;
}
public void recordPurposeOfPersonalData(JSONArray purpose) {
    //TODO Add your code to record purpose
    this.purpose = new ArrayList<PurposeOfPersonalData>();
    for(int i=0; i<purpose.length(); i++){
        PurposeOfPersonalData p = new PurposeOfPersonalData();
        try {
            p.setId(purpose.getJSONObject(i).getString("id"));
            p.setDetail(purpose.getJSONObject(i).getString("detail"));
            p.recordDataCategory(purpose.getJSONObject(i).getString("data_category"));
            p.recordActiveStatus(purpose.getJSONObject(i).getString("active_status"));
        } catch (JSONException e) {e.printStackTrace();}
        this.purpose.add(p);
    }
}
```

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Sample Design and Code	The class PurposeForPersonalData is implemented to have the data attributes that store the purpose of the collection and use of personal data. The sample code is shown below.

```
    public void recordCollectedPersonalData(String data) {
        //TODO Add your code to record collected personal data
        this.dataCollected = data;
    }

    public void recordUseOfPersonalData(String use) {
        //TODO Add your code to record use of personal data
        this.dataUse = use;
    }

    public void recordMethodToCollect(String method) {
        //TODO Add your code to record method to collect personal data
        this.collectionMethod = method;
    }

    public void recordResidenceOfController(String residence) {
        //TODO Add your code to record residence of data controller
        this.residenceOfController = residence;
    }
}
```

```
public class PurposeOfPersonalData {
    private String id;
    private String detail;
    private String dataCategory;
    private String activeStatus;
    private Date dateCreation;
    private Date dateUpdate;

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getDetail() {
        return detail;
    }
    public void setDetail(String detail) {
        this.detail = detail;
    }
    public String getDataCategory() {
        return dataCategory;
    }
    public String getActiveStatus() {
        return activeStatus;
    }
    public Date getDateCreation() {
        return dateCreation;
    }
    public void setDateCreation(Date dateCreation) {
        this.dateCreation = dateCreation;
    }
    public Date getDateUpdate() {
        return dateUpdate;
    }
    public void setDateUpdate(Date dateUpdate) {
        this.dateUpdate = dateUpdate;
    }
    public void recordActiveStatus(String activeStatus) {
        // // TODO Add your code to record purpose status
        this.activeStatus = activeStatus;
    }
    public void recordDataCategory(String dataCategory) {
        // TODO Add your code to record category of collected personal data
        this.dataCategory = dataCategory;
    }
}
```

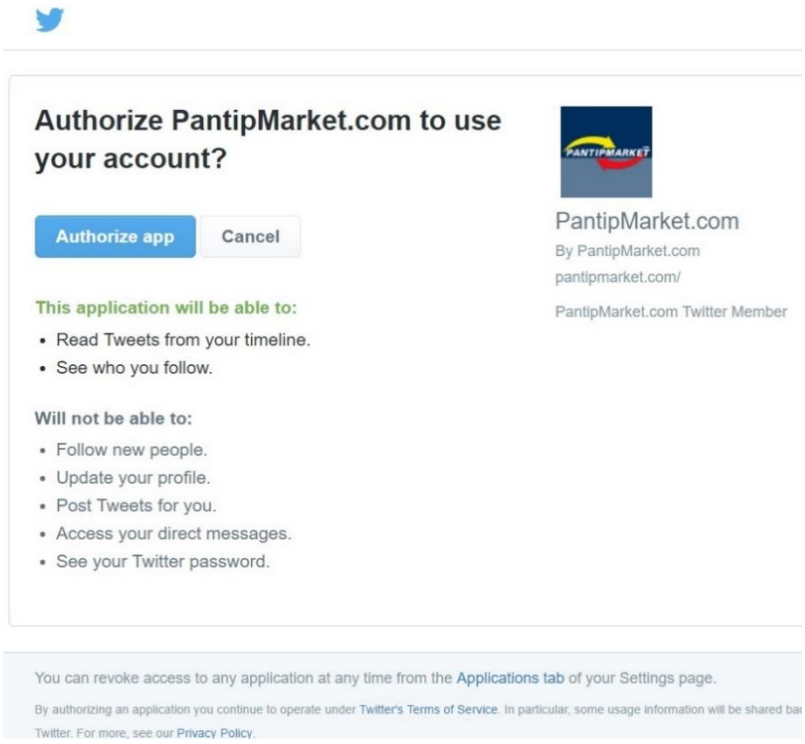
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Sample Design and Code	The class RegistrationPersonalData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of the system as shown in the sample code below. `public interface PersonalData {` `  public void recordDataCategory(String dataCategory);` `}` `public class RegistrationPersonalData implements PersonalData{` `  private String name;` `  private String email;` `  private String tel;` `  private String dataCategory;` `  public String getName() {` `  return name;` `  }` `  public void setName(String name) {` `  this.name = name;` `  }` `  public String getEmail() {` `  return email;` `  }` `  public void setEmail(String email) {` `  this.email = email;` `  }` `  public String getTel() {` `  return tel;` `  }` `  public void setTel(String tel) {` `  this.tel = tel;` `  }` `  public String getDataCategory() {` `  return dataCategory;` `  }` `  @Override` `  public void recordDataCategory(String dataCategory) {` `  // TODO Add your code to record category of collected personal data` `  this.dataCategory = dataCategory;` `  }` `}` When the data controller requires to do something that deviates from the specified purpose without requiring consent, the class PurposeDeviationNotification should be implemented to notify the data subject.

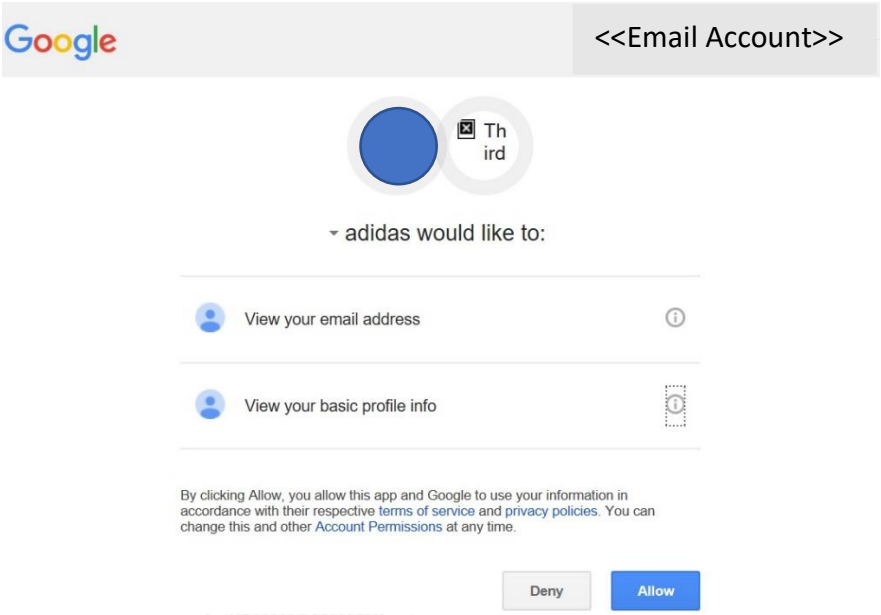
ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

<table>
<tr><td>Sample Design and Code</td><td>

```
public class PurposeDeviationNotification {
    public void notifyDeviation(){
        /* TODO Add your code to notify data subject when the data controller requires
         * to do something that deviates from the specified purpose
         */
        System.out.print("We notice you about...");

        PrivacyPolicyManager policyManager = new PrivacyPolicyManager();
        PrivacyPolicy policy = policyManager.showPrivacyPolicy();
        System.out.println("Name :: "+policy.getName());
        System.out.println("Collected Data :: "+policy.getDataCollected());
        System.out.println("Use Data :: "+policy.getDataUse());
        System.out.println("Collected Method :: "+policy.getCollectionMethod());
        for(int i=0; i<policy.getPurpose().size(); i++)
            System.out.println("Purpose "+i+" :: "+policy.getPurpose().get(i).getDetail()
        System.out.println("Residence :: "+policy.getResidenceOfController());

        System.out.println("Should you have any inquiries, please feel free to contact us'
    }
}
```

</td></tr>
<tr><td>Known use</td><td>Examples of existing systems that take the Use Limitation principle into account include:
• Twitter (Website): When PantipMarket.com requires users' personal data from Twitter, Twitter informs the user (or data subject) about the personal data that will be transmitted to PantipMarket.com. If the user gives consent, personal data are transmitted.
 
</td></tr>
</table>

ตารางที่ ง.4 แบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล (ต่อ)

Known use	• Google (Website): When the Adidas website require users' personal data from Google, Google informs the user (or data subject) about the personal data that will be transmitted to the Adidas website. If the user gives consent, personal data are transmitted. 
Related Patterns	Purpose Specification The Purpose Specification pattern can be used with the Use Limitation pattern to ensure that the use and disclosure of the personal data should always be limited to the purpose specification. Openness The Openness pattern can be used with the Use Limitation pattern to ensure that the information about how the personal data are used and disclosed should always be open to the data subject. Even in the case that the system has to do something that deviates from the specified purpose, that should be made known to the data subject. Individual Participation The Individual Participation pattern requires the Use Limitation pattern to ensure that the data subject can participate in the use of his/her personal data by being informed of the use of his/her personal data which deviates from the specified purposes.

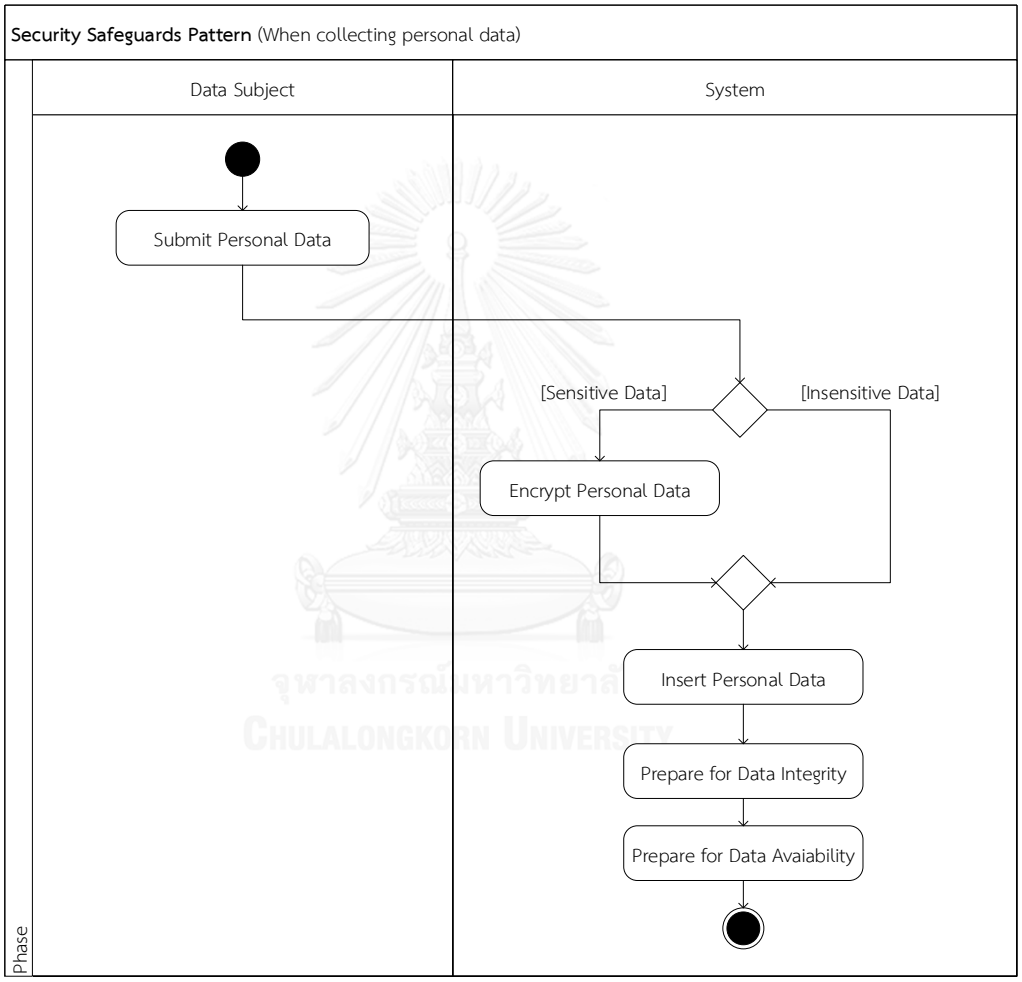
ง. 5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล

Pattern Name	**Security Safeguards Pattern**
Privacy Principle	Security Safeguards Principle
Intent	This pattern assures that the system should be able to protect personal data from risks such as loss or unauthorized access, destruction, use, modification or disclosure of data.
Motivation	The collected personal data are an important asset. Personal data breach may be caused by careless employees who do not follow security procedures, hackers who access inadequately protected databases, or thieves who steal unsecured devices etc. Personal data breach can do harm to both the data controller and data subject in terms of financial loss, reputational impact, and loss of trust and confidence. The data controller needs to have security safeguards in place to protect the three principal components of information security, i.e. confidentiality, integrity, availability.
Applicability	Use the Security Safeguards pattern to maintain security of the personal data of a data subject throughout their life cycle within the system: • On storage or during transmission, the collected personal data should be encrypted. • On creation, use, and destruction of the personal data, access control has to be defined and enforced. • On access to the personal data, the system should privide an authentication mechanism to authenticate users who are either the data subject or other users of the system who can access personal data. • On data collection and storage, the system should provide a data integrity mechanism so that the collected personal data can be checked for integrity, and in some cases of failure, the corrupted data can be fixed or replaced. • On data collection and storage, the system should provide a mechanism to prevent disruption of access to personal data, and in the case of failure, the service or data can be replaced.

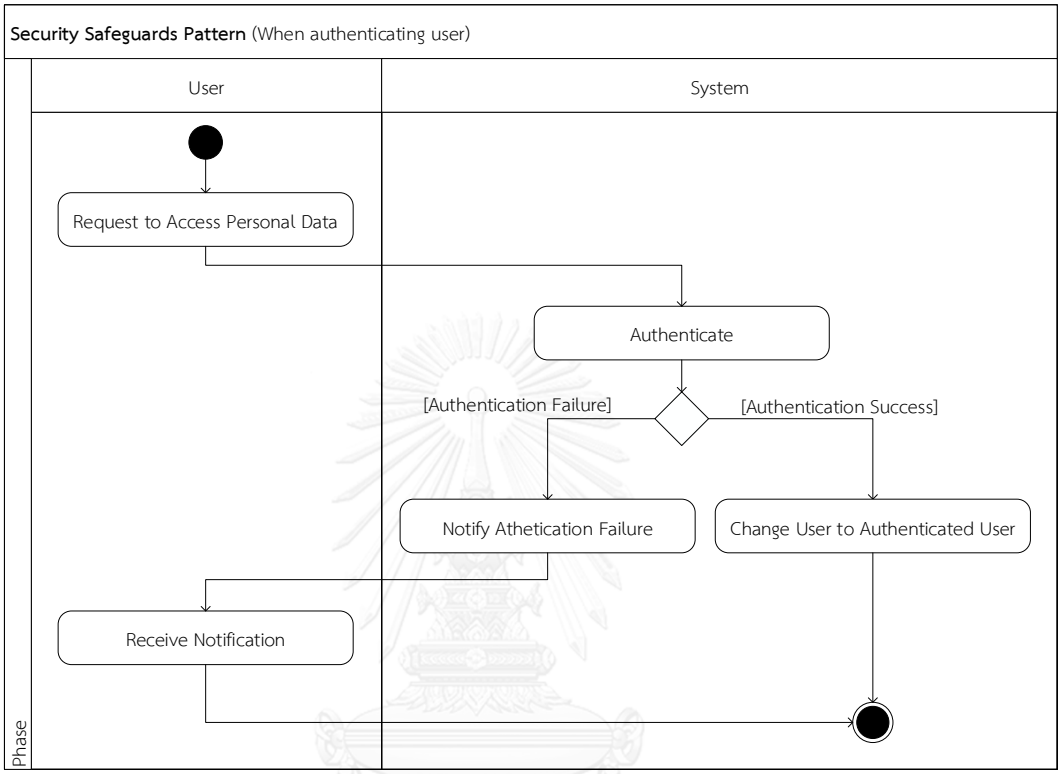
ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Applicability	• On use of the personal data, data breach monitoring is in place and relevant parties are notified of suspicious circumstances. The system should handle further the suspicious data breach.
Behavior and Structure	
Activity Diagram (When collecting personal data.)	



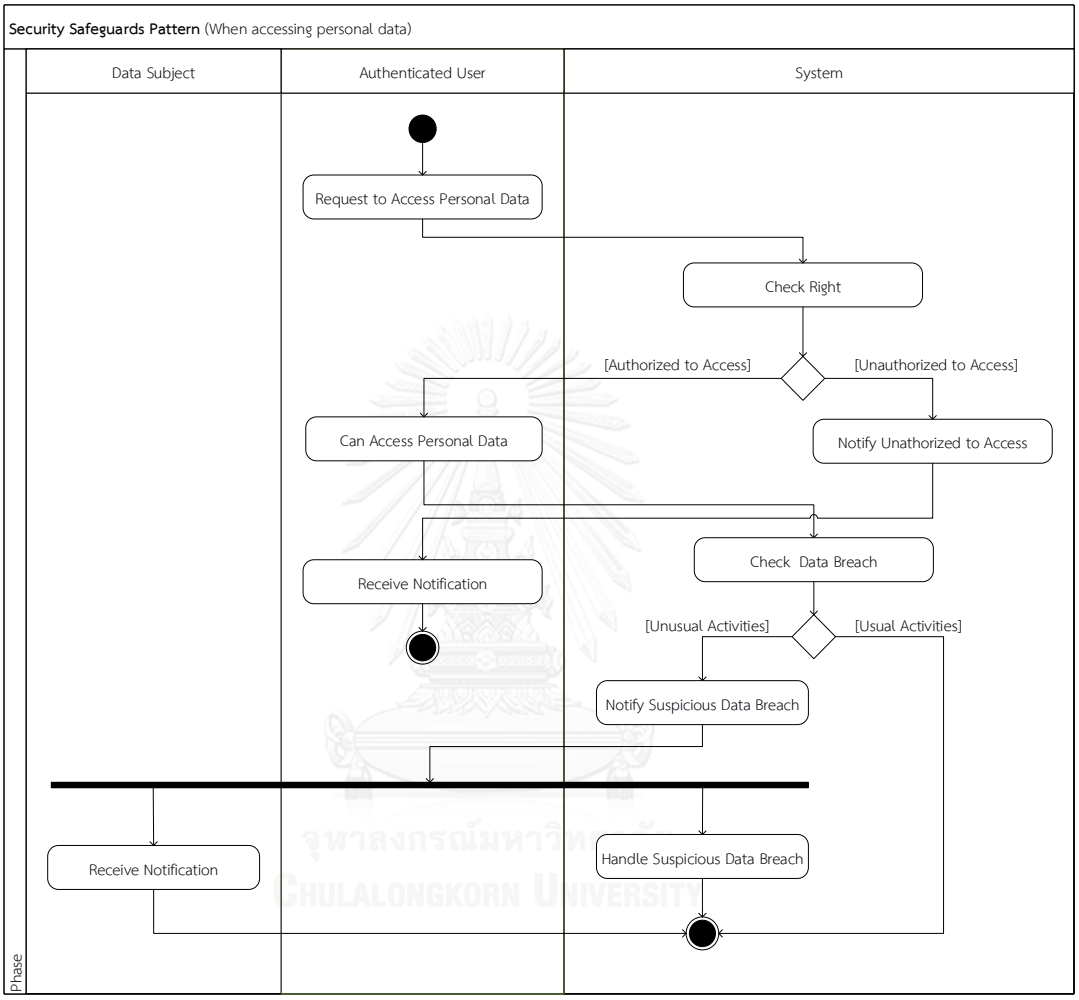
ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Behavior and Structure
Activity Diagram (When authenticating user)



ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

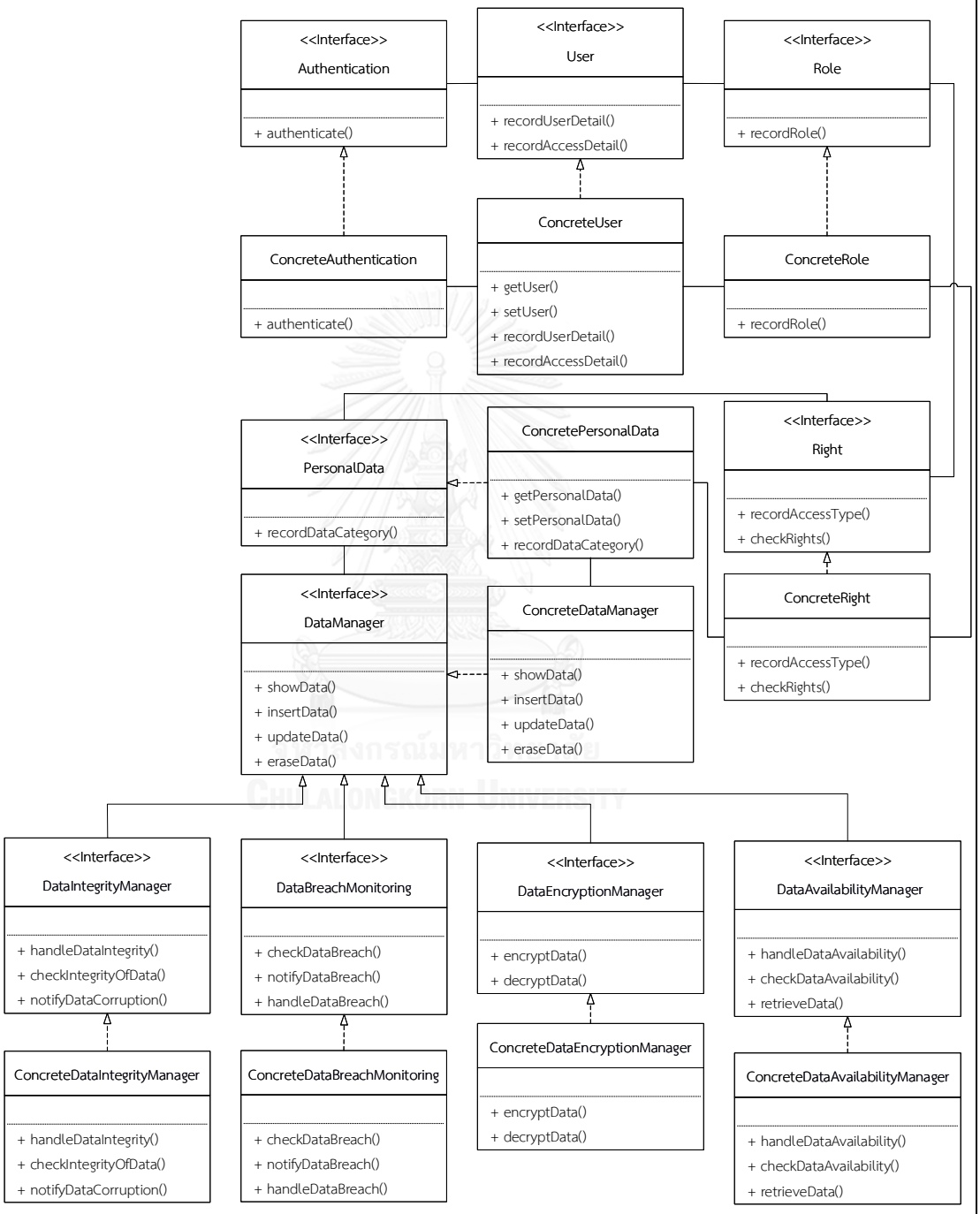
Behavior and Structure
Activity Diagram (When accessing personal data)



ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Behavior and Structure

Class Diagram



ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Participants	
Interface Name/Class Name	**Detail**
Interface DataManager	An interface that has the responsibilities to manage personal data, including showing the personal data to an authorized user.
Class ConcreteDataManager	A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. insert, erase or update of personal data etc.
Interface PersonalData	An interface that represents the collected personal data.
Class ConcretePersonalData	A concrete class that implement PersonalData with the data attributes that are necessary for the purpose of the system.
Interface User	An interface that represents the user who accesses the system (and may not be the data subject) and has responsibilities to record user detail and access detail.
Class ConcreteUser	A concrete class that implements the interface User with the data attributes to identify who accesses the system as well as access detail.
Interface Authentication	An interface that has responsibilities to authenticate users before accessing the personal data.
Class ConcreteAuthentication	A concrete class that implements the interface Authentication.
Interface Role	An interface that represents the role of the user who accesses the system (e.g. member, admin, guest).
Class ConcreteRole	A concrete class that implements the interface Role with the data attributes to identify the role of the user.
Interface Right	An interface that has responsibilities to record and check the access right of a user role over protected personal data based on the access type (e.g. create, destroy, view, modify).
Class ConcreteRight	A concrete class that implements the interface Right.
Interface DataBreachMonitoring	An interface that has responsibilities to monitor access to personal data, notify relevant parties (e.g. data subject and data controller) about unauthorized or suspicious access behavior, and further handle the breach.

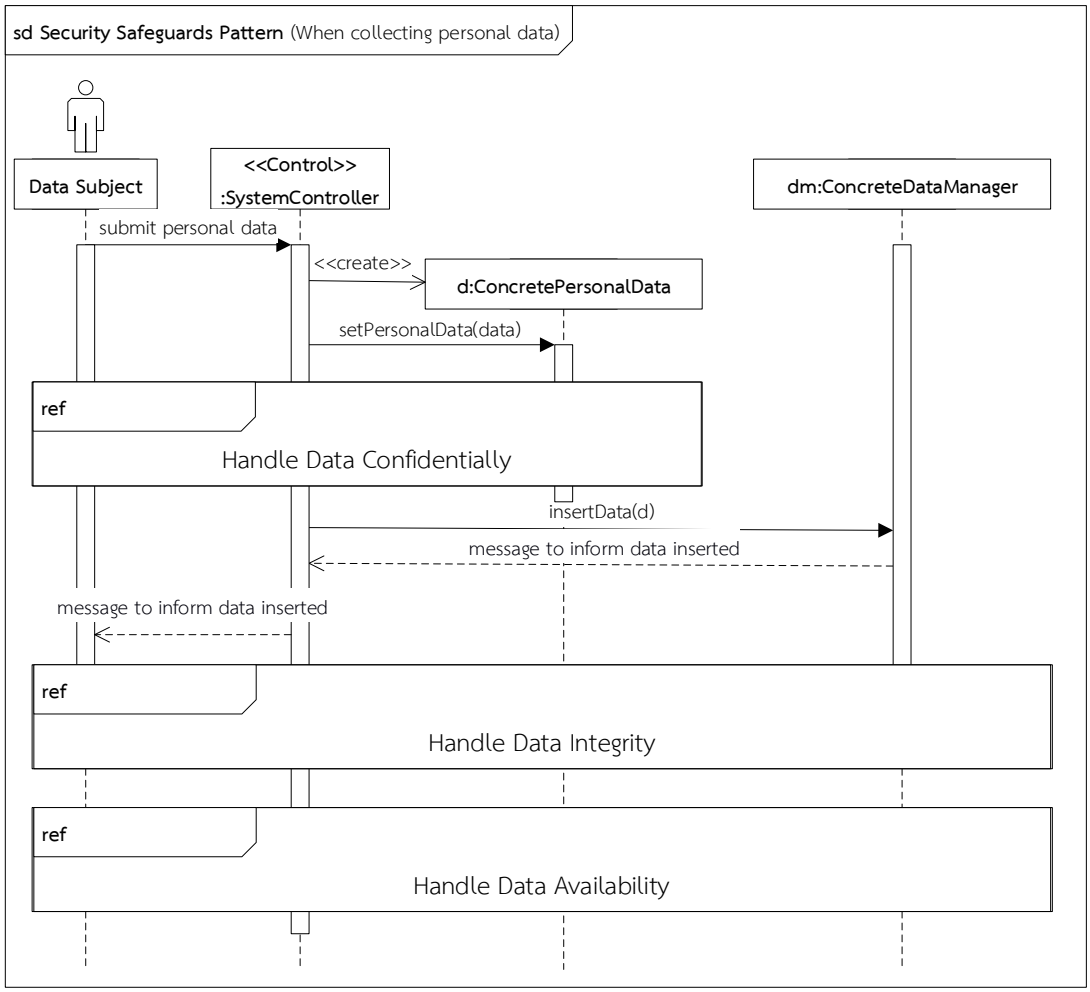
ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Participants	
Interface Name/Class Name	**Detail**
Class ConcreteDataBreachMonitoring	A concrete class that implements the interface DataBreachMonitoring.
Interface DataEncryptionManager	An interface that has the responsibilities to encrypt personal data to hide them from plain view for confidentiality as well as decrypt them.
Class ConcreteDataEncryptionManager	A concrete class that implements the interface DataEncryptionManager.
Interface DataIntegrityManager	An interface that has the responsibilities to provide a mechanism and check for integrity of personal data.
Class ConcreteDataIntegrityManager	A concrete class that implements the interface DataIntegrityManager.
Interface DataAvailabilityManager	An interface that has the responsibilities to provide a mechanism and check for availability of personal data.
Class ConcreteDataAvailabilityManager	A concrete class that implements the interface DataAvailabilityManager.
Collaboration	
1. When collecting personal data 1.1 The system controller creates a ConcretePersonalData object to set the collected personal data into the object. 1.2 The system controller handles data confidentiality: 1.2.1 If the collected personal data are sensitive, the ConcretePersonalData creates a ConcreteDataEncryptionManager object to encrypt personal data. 1.3 The system controller calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store. 1.4 The system controller handles data integrity: 1.4.1 The system controller creates a ConcreteDataIntegrityManager object to provide a mechanism for data integrity.	

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

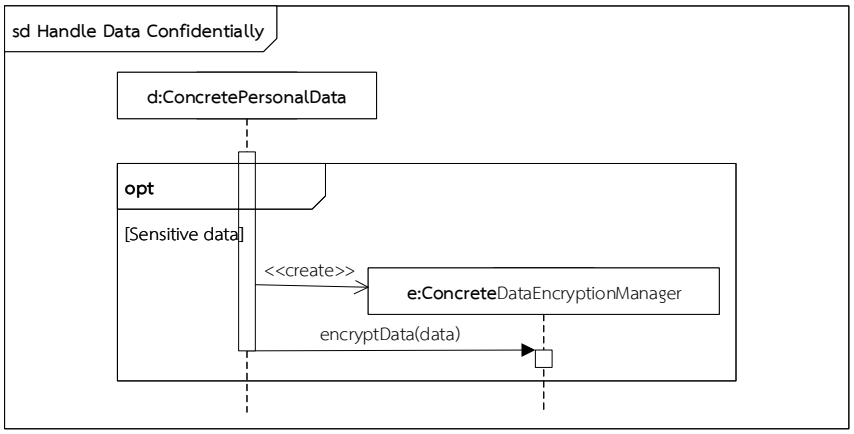
Collaboration
1.4.2 The ConcreteDataIntegrityManager object calls the ConcretePersonalData object to get the collected personal data and provide an integrity mechanism for those data. 1.5 The system controller handles data availability: 1.5.1 The system controller creates a ConcreteDataAvailabilityManager object to provide a mechanism for data availability. 1.5.2 The ConcreteDataAvailabilityManager object calls the ConcretePersonalData object to get the collected personal data and provide an availability mechanism for those data. 2. When accessing personal data 2.1 A user is authenticated: 2.1.1 A user requests to access personal data. 2.1.2 The system controller creates a ConcreteAuthentication object which then gets the user detail from a ConcreteUser object for authentication. 2.1.3 If authentication fails, the system controller notifies the user. 2.2 The authenticated user requests to access personal data. 2.3 The system controller calls a ConcreteUser object to get a ConcreteRole object that identifies the role of the authenticated user. 2.4 The system controller calls the ConcreteRole object to get a set of ConcreteRight objects for that role. 2.5 The system controller calls the ConcreteRight objects to get the associated ConcretePersonalData objects and checks if the authenticated user is authorized to access the personal data with regard to its role and the access type. 2.6 If the authenticated user is authorized to access, the system controller creates a ConcreteDataManager to access the personal data. 2.7 If the authenticated user is not authorized to access, the system controller notifies the authenticated user. 2.8 The system controller creates a ConcreteDataBreachMonitoring object to monitor access to personal data. 2.8.1 The system controller calls the ConcreteDataBreachMonitoring to check for suspicious access behavior.

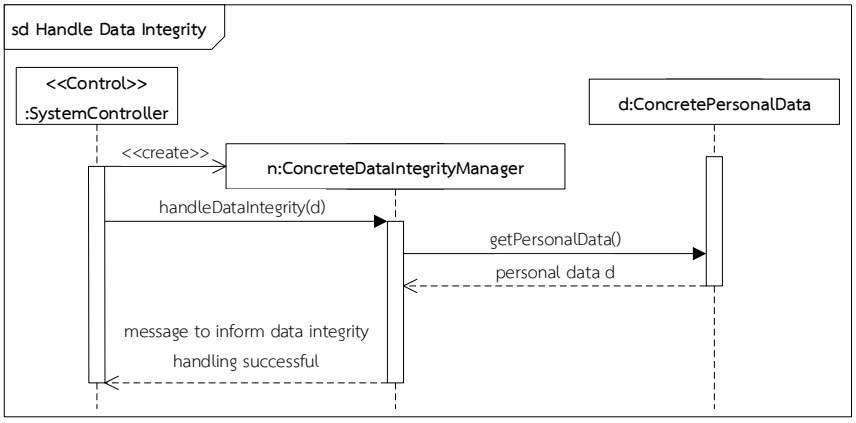
ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

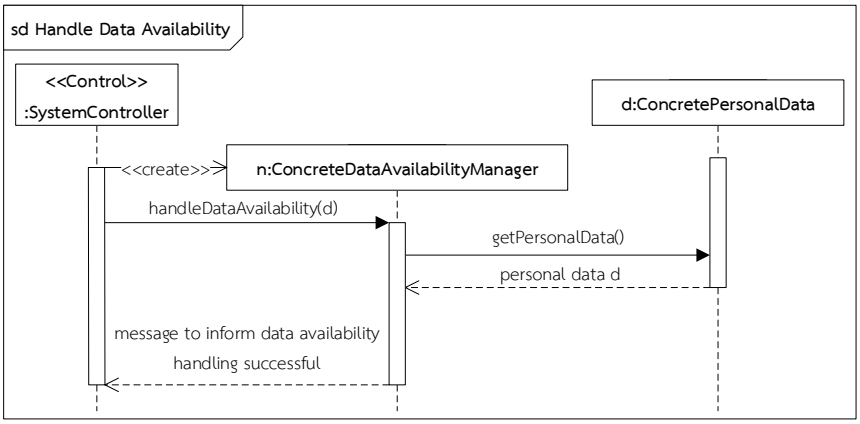
Collaboration
2.8.2 If access behavior is unusual, the ConcreteDataBreachMonitoring object notifies the data subject about suspicious access behavior, and handles the data breach.
Sequence Diagram (When collecting personal data.) 

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data. – continued)

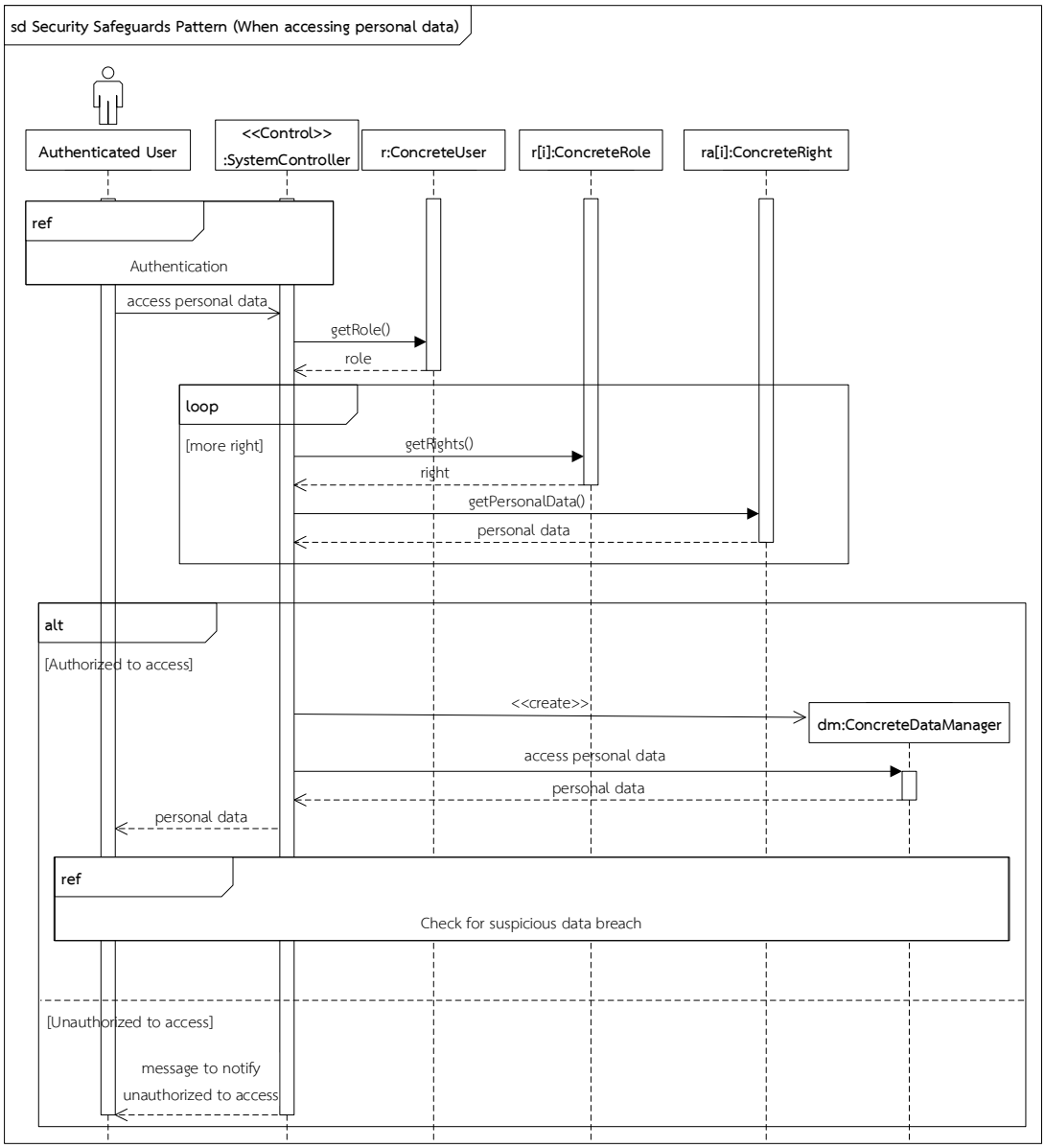






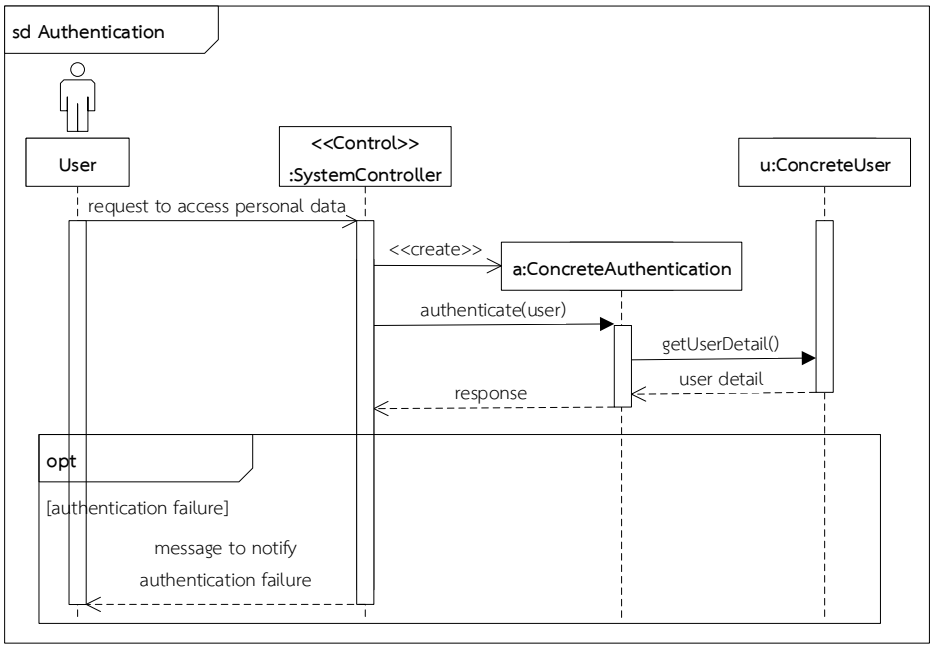
ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

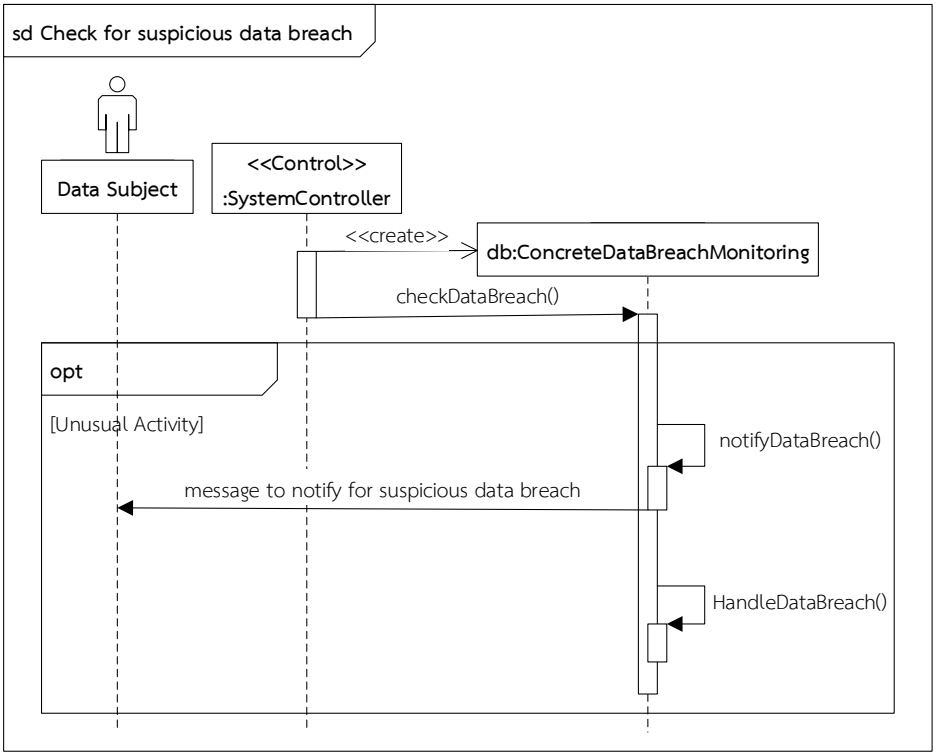
Collaboration
Sequence Diagram (When accessing personal data.)



ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Collaboration
Sequence Diagram (When accessing personal data. - continued)





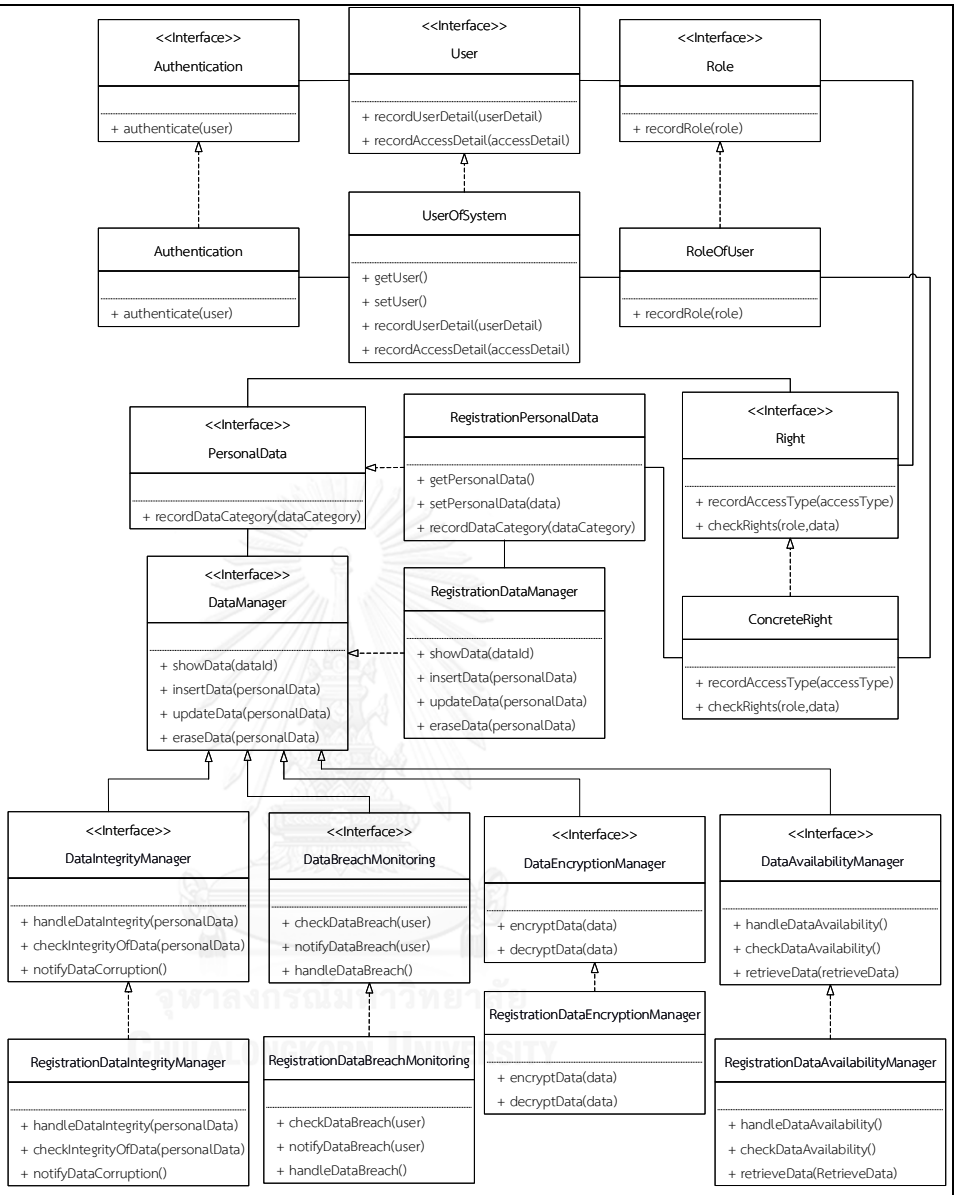
ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Consequences	1. Using this pattern, the system of the data controller can maintain the security of the personal data as recommended by OECD. The data Subject can be assured that the system will protect personal data by reasonable security safeguards from risks such as loss or unauthorized access, destruction, use, modification or disclosure of data. 2. There can be a performance impact related to encryption, depending on the encryption process that is implemented. In general, the actual encryption is not a significant factor. On the contrary, the overhead from network connection to fetch an encryption key from a key server, storage of the encrypted data, and event logging related to every key request and authentication with the key server are time-consuming processes. 2. There are costs in terms of time and space associated with providing for data integrity and data availability. Extra storage space is required for any data integrity mechanisms that are used by the system and regular checking has to be in place. 3. In the case that mirroring is used as a data integrity and data availability mechanism, the system architecture employs a replication model. Consistency across replicas has to be maintained when there are updates to the personal data.
Implementation	Issues regarding the implementation of the Security Safeguards pattern are as follows. 1. To implement security safeguards, the data controller should additionally refer to existing security patterns, such as those in the book titled Security Patterns Integrating Security and Systems Engineering [13]. 2. The class ConcreteAuthentication should implement an authentication mechanism to validate the identity of the user who is about to access personal data. Common mechanisms include username-password, smart card authentication, fingerprint scanning. 3. The class ConcreteDataEncryptionManager should implement a data encryption method to assure data confidentiality. Common methods include hashing, symmetric method (private-key cryptography), and asymmetric method (public-key cryptography) [14].

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Implementation	4. The class ConcreteDataIntegrityManager should implement a data integrity mechanism to assure data integrity. Common techniques include mirroring (or replicating data to compare different copies of the data), disk parity, and checksumming [12]. 5. The class ConcreteDataAvailabilityManager should implement a data availability mechanism to assure data availability. Common techniques include mirroring (or replicating data) and backup with log rerun. 6. The class ConcreteDataBreachMonitoring should make use of the user detail and the history of access detail that is maintained as a log of events in the system in order to check for data breach. The data controller should refer to the Accountability pattern also. To handle the data breach, the system may choose to notify the data subject only or block the user from accessing personal data until further arrangement is made by the data subject. 7. In addition to the security safeguards implemented by the system, the data controller should also take proper physical measures, i.e. technical, administrative, and physical controls, to guard against security threats to the system and personal data [14]. Technical controls include access control systems, intrusion detection systems, and auditing systems. Administrative controls include site location, facility design, emergency plan. Examples of physical controls are building materials, locks, guards.
Sample Design and Code	An example below shows the case of a system that collects personal data for use in the system where sensitive data should be encrypted. The system should provide a data integrity mechanism, and corrupted data can be fixed. Also, the system should provide a data availability mechanism to prevent disruption of access to personal data.

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	 When a user (data subject) submit personal data to the system, the system uses the class RegistrationPersonalData that implements the interface PersonalData to set the collected personal data into the object. `public interface PersonalData {` `  public void recordDataCategory(String dataCategory);` `}`

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	``` public class RegistrationPersonalData implements PersonalData{ private String id; private String name; private String email; private String emailChecksum; private String tel; private String telChecksum; private String dataCategory; public String getId() { return id; } public void setId(String id) { this.id = id; } public String getName() { return name; } public void setName(String name) { this.name = name; } public String getEmail() { return email; } public void setEmail(String email) { this.email = email; } public String getEmailChecksum() { return emailChecksum; } public void setEmailChecksum(String emailChecksum) { this.emailChecksum = emailChecksum; } public String getTel() { return tel; } public void setTel(String tel) { this.tel = tel; } public String getTelChecksum() { return telChecksum; } public void setTelChecksum(String telChecksum) { this.telChecksum = telChecksum; } public String getDataCategory() { return dataCategory; } @Override public void recordDataCategory(String dataCategory) { // TODO Auto-generated method stub this.dataCategory = dataCategory; } } ``` To hide the sensitive data from plain view for confidentiality, the system uses the class RegistrationDataEncryptionManager that implements the interface DataEncryptionManager to encrypt and decrypt personal data (e.g. using AES encryption in this example). ``` interface DataEncryptionManager extends DataManager{ public String encryptData(String data); public String decryptData(String data); } ```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	

```
public class RegistrationDataEncryptionManager
           extends RegistrationDataManager implements DataEncryptionManager{

    private String key = "Bar12345Bar12345"; // 128 bit key
    private String initVector = "RandomInitVector"; // 16 bytes IV

    @Override
    public String encryptData(String data) {
        try {
            IvParameterSpec iv = new IvParameterSpec(initVector.getBytes("UTF-8"));
            SecretKeySpec skeySpec = new SecretKeySpec(key.getBytes("UTF-8"), "AES");

            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec, iv);

            byte[] encrypted = cipher.doFinal(data.getBytes());
            System.out.println("encrypted string: "
                    + Base64.encodeBase64String(encrypted));

            return Base64.encodeBase64String(encrypted);
        } catch (Exception ex) { ex.printStackTrace();}
        return null;
    }
    @Override
    public String decryptData(String data) {
         try {
             IvParameterSpec iv = new IvParameterSpec(initVector.getBytes("UTF-8"));
             SecretKeySpec skeySpec = new SecretKeySpec(key.getBytes("UTF-8"), "AES");

             Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
             cipher.init(Cipher.DECRYPT_MODE, skeySpec, iv);

             byte[] original = cipher.doFinal(Base64.decodeBase64(data));

             return new String(original);
         } catch (Exception ex) { ex.printStackTrace();}
         return null;
    }
}
```

The class RegistrationDataManager implements the interface DataManager to show, insert, update or erase personal data

```
public interface DataManager {
    public PersonalData showData(String dataId);
    public void insertData(RegistrationPersonalData data);
    public void updateData(RegistrationPersonalData data);
    public void eraseData(RegistrationPersonalData data);
}
public class RegistrationDataManager implements DataManager{

    protected Date now = new Date();
    protected SimpleDateFormat sdf =
                        new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

    @Override
    public PersonalData showData(String dataId) {
        // TODO Add your code to get personal data from a data store
        RegistrationPersonalData data = new RegistrationPersonalData();
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String query = "SELECT * FROM personaldata WHERE id = ?" ;
            PreparedStatement stmt = connection.prepareStatement(query);
            stmt.setInt(1, Integer.parseInt(dataId));
            ResultSet result = stmt.executeQuery();
            while(result.next()){
                data.setId(result.getString("id"));
                data.setName(result.getString("name"));
                data.setEmail(result.getString("email"));
                data.setEmailChecksum(result.getString("email_cs"));
                data.setTel(result.getString("tel"));
                data.setTelChecksum(result.getString("tel_cs"));
            }
            stmt.close();
        } catch (SQLException e) {}
        return data;
    }
```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	

```
    @Override
    public void insertData(RegistrationPersonalData data){
        //TODO Add your code to insert personal data into a data store.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String insert = "INSERT INTO personaldata "
                    + "(name, email, tel) "
                    + "VALUES (?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert)
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, data.getName());
            stmt.setString(parameterIndex++, data.getEmail());
            stmt.setString(parameterIndex++, data.getTel());

            stmt.execute();
            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
    }
    @Override
    public void updateData(RegistrationPersonalData data) {
        // TODO Add your code to update personal data
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            StringBuffer upd = new StringBuffer("UPDATE personaldata SET ");
            if (data.getName() != null) {
                upd.append("name = '"+data.getName()+"', ");
            }
            if (data.getEmail() != null) {
                upd.append("email = '"+data.getEmail()+"', ");
            }
            if (data.getTel() != null) {
                upd.append("tel = '"+data.getTel()+"', ");
            }

            upd.append("updatedate = '"+sdf.format(now)+"' ");
            upd.append("WHERE id = '"+data.getId()+"' ");
            System.out.println(upd.toString()) ;
            PreparedStatement stmt =
                    connection.prepareStatement(upd.toString()) ;
            stmt.execute();
            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
    }
    @Override
    public void eraseData(RegistrationPersonalData data) {
        // TODO Add your code to erase personal data
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try{
            String insert = "DELETE FROM personaldata "
                    + "where id = ? " ;

            PreparedStatement stmt = connection.prepareStatement(insert)
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, data.getId());

            stmt.execute();
            stmt.close();
        }catch(SQLException e) {}
    }
}
```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	To provide a mechanism to assure data integrity, the system uses the class RegistrationDataIntegrityManager that implements the interface DataIntegrityManager to provide and check integrity of the personal data (e.g. using bcrypt checksum in this example), and notify the system if personal data are corrupted.

```
interface DataIntegrityManager extends DataManager{
    public void handleDataIntegrity(RegistrationPersonalData data);
    public void checkDataIntegrity(RegistrationPersonalData data);
    public void notifyDataCorruption();
}
public class RegistrationDataIntegrityManager
            extends RegistrationDataManager implements DataIntegrityManager{
    private static int workload = 12;

    @Override
    public void handleDataIntegrity(RegistrationPersonalData data) {
        // TODO Add your code to prepare personal data for check data integrity
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        String salt = BCrypt.gensalt(workload);
        try {
            StringBuffer upd = new StringBuffer("UPDATE personaldata SET ");

            if (data.getEmail() != null) {
                upd.append("email_cs = '"+BCrypt.hashpw(data.getEmail(),salt)+"', ");
            }
            if (data.getTel() != null) {
                upd.append("tel_cs = '"+BCrypt.hashpw(data.getTel(),salt)+"', ");
            }
            upd.append(" updatedate = '"+sdf.format(now)+"' ");
            upd.append("WHERE name = '"+data.getName()+"' ");
            upd.append("AND email = '"+data.getEmail()+"' ");
            System.out.println(upd.toString()) ;
            PreparedStatement stmt = connection.prepareStatement(upd.toString()) ;
            stmt.execute();
            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
    }
     @Override
     public void checkDataIntegrity(RegistrationPersonalData data) {
         // TODO Add your code to check data integrity
         RegistrationPersonalData personalData = new RegistrationPersonalData(
         personalData = (RegistrationPersonalData) showData(data.getId());

         System.out.println("Check data integrity...");

         String email = personalData.getEmail();
         String emailCheckSum = personalData.getEmailChecksum();
         if(!BCrypt.checkpw(email,emailCheckSum))
             notifyDataCorruption();

         String tel = personalData.getTel();
         String telCheckSum = personalData.getTelChecksum();
         if(!BCrypt.checkpw(tel,telCheckSum))
             notifyDataCorruption();
     }

     @Override
     public void notifyDataCorruption() {
         // TODO Add your code to notify data are corrupted
                 System.out.println("Personal data are corrupted");
     }
}
```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	To provide a mechanism to assure data availability, the system uses the class RegistrationDataAvailabilityManager that implements the interface DataAvailabilityManager to provide availability of personal data (e.g. using replication in this example), and check to recover if personal data are lost.

```
interface DataAvailabilityManager extends DataManager{
    public void handleDataAvailability();
    public void checkDataAvailability();
    public void retrieveData(String date);
}
public class RegistrationDataAvailabilityManager
            extends RegistrationDataManager
            implements DataAvailabilityManager{
    private SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
    private Date datenow = new Date();

    @Override
    public void handleDataAvailability() {
        // TODO Add your code to prepare personal data for data availability
        PrintWriter pw;
        try {
            pw = new PrintWriter(new File("D:/log/registrationdata-"+
                    sdf.format(datenow)+".csv"));
            StringBuilder sb = new StringBuilder();

            JSONArray data = showData();
            for(int i=0; i<data.length(); i++){
                sb.append(data.getJSONObject(i).getString("id")); sb.append(',');
                sb.append(data.getJSONObject(i).getString("name")); sb.append(',');
                sb.append(data.getJSONObject(i).getString("email")); sb.append(',');
                sb.append(data.getJSONObject(i).getString("email_cs")); sb.append(',')
                sb.append(data.getJSONObject(i).getString("tel")); sb.append(',');
                sb.append(data.getJSONObject(i).getString("tel_cs")); sb.append('\n');
            }
            pw.write(sb.toString());
            pw.close();
            System.out.println("prepare personal data done!");
        } catch (FileNotFoundException e) {e.printStackTrace();}
          catch (JSONException e) {e.printStackTrace();}
    }
     @Override
     public void checkDataAvailability() {
         // TODO Add your code to check for data availability
         JSONArray data = showData();
         for(int i=0; i<data.length(); i++){
             System.out.println("can access personal data");
         }
     }
    @Override
    public void retrieveData(String date) {
        // TODO Add your code to retrive data
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        String filePath = "D:/log/registrationdata-"+date+".csv";
        try {
            String insert = "INSERT INTO personaldata "
                    + "(name, email, email_cs, tel, tel_cs) "
                    + "VALUES (?, ?, ?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert)

            FileReader fl = new FileReader(filePath);
            BufferedReader readFile = new BufferedReader(fl);
            String line = "";
            while ((line = readFile.readLine()) != null) {
                String[] data = line.split(",");
                for(int i = 1; i<data.length; i++)
                    stmt.setString(i, data[i]);

                stmt.execute();
            }
            stmt.close();
            readFile.close();
        } catch (FileNotFoundException e) {e.printStackTrace();}
          catch (IOException e) {e.printStackTrace();}
          catch (SQLException e) {e.printStackTrace();}
    }
```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	The system uses the class UserAuthentication that implements the interface Authentication to authenticate users before accessing the personal data and notify the user, if authentication fails. `interface Authentication {` `  public boolean authenticate(UserOfSystem user);` `}` `public class UserAuthentication implements Authentication{` `  @Override` `  public boolean authenticate(UserOfSystem user) {` `  DatabaseConnection dbConnection = new DatabaseConnection();` `  Connection connection = dbConnection.getDBConnection();` `  try {` `  String query = "SELECT * FROM useraccount "` `  + " WHERE username = ? AND password = ?" ;` `  PreparedStatement stmt = connection.prepareStatement(query)` `  stmt.setString(1, user.getUsername());` `  stmt.setString(2, user.getPassword());` `  ResultSet result = stmt.executeQuery();` `  while(result.next()){` `  return true;` `  }` `  stmt.close();` `  } catch (SQLException e) {e.printStackTrace();}` `  return false;` `  }` `}` The class UserOfSystem that implements the interface User can record who accesses the system, access detail, and user role in the system. `interface User {` `  public void recordUsername(String username);` `  public void recordPassword(String password);` `  public void recordLocation();` `  public void recordOS();` `  public void recordAccessDate();` `}` `public class UserOfSystem implements User{` `  private String username;` `  private String password;` `  private String location;` `  private String os;` `  private Date accessDate;` `  private RoleOfUser role;` `  RegistrationDataEncryptionManager encrypt =` `  new RegistrationDataEncryptionManager();`

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code
```
    public String getUsername() {
        return username;
    }
    public String getPassword() {
        return password;
    }
    public String getLocation() {
        return location;
    }
    public void setLocation(String location) {
        this.location = location;
    }
    public String getOs() {
        return os;
    }
    public void setOs(String os) {
        this.os = os;
    }
    public Date getAccessDate() {
        return accessDate;
    }
    public void setAccessDate(Date accessDate)
        this.accessDate = accessDate;
    }
    public RoleOfUser getRole() {
        return role;
    }
    public void setRole(RoleOfUser role) {
        this.role = role;
    }
    @Override
    public void recordUsername(String username) {
        /* TODO Add your code to record username
         */
        this.username = username;
    }
    @Override
    public void recordPassword(String password) {
        /* TODO Add your code to record password
         */
        this.password = encrypt.encryptData(password);
    }
    @Override
    public void recordLocation() {
        /* TODO Add your code to record location that user live
         * when access personal data.
         */
        Locale currentLocale = Locale.getDefault();
        this.location = currentLocale.getDisplayLanguage();
    }
    @Override
    public void recordOS() {
        /* TODO Add your code to recode operating system that user
         * uses to access personal data.
         */
        String operate = System.getProperty("os.name");
        this.os = operate;
    }
    @Override
    public void recordAccessDate() {
        /* TODO Add your code to recode date that user access
         * personal data.
         */
        Date datenow = new Date();
        this.accessDate = datenow;
    }
```


The class RoleOfUser that implements the interface Role can record a role (e.g. member, admin, guest) and different rights associated with the role.


```
interface Role {
    public void recordRole(String role);
}
```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	

```
public class RoleOfUser implements Role{
    private String role;
    private ArrayList<AccessRight> right;

    public String getRole() {
        return role;
    }
    public void setRole(String role) {
        this.role = role;
    }
    public ArrayList<AccessRight> getRight() {
        return right;
    }
    public void setRight(ArrayList<AccessRight> right) {
        this.right = right;
    }
    @Override
    public void recordRole(String role) {
        // TODO Add your code to record role of user
        this.role = role;
    }

}
```

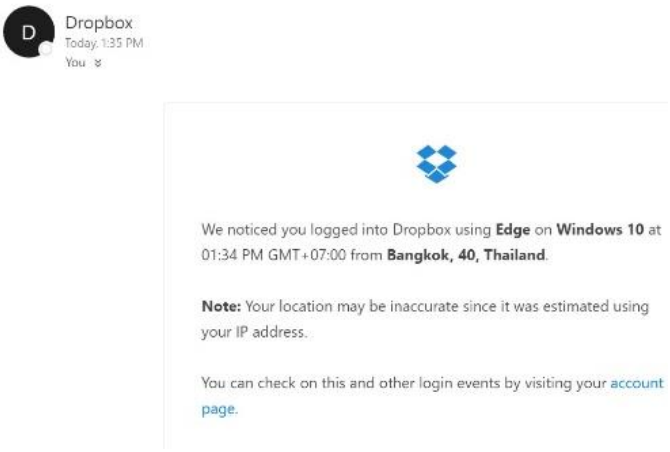
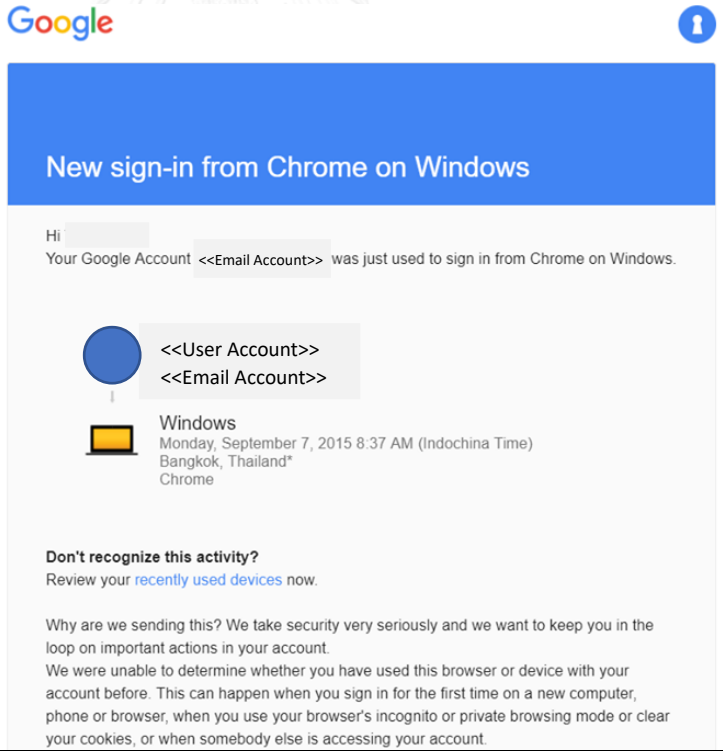
The class AccessRight that implements the interface Right can record the accesstype (e.g. create, destroy, view, modify) associated with a RegistrationPersonalData object. It also records which RoleOfUser object has this right.

```
interface Right {
    public void recordAccessType(String accessType);
    public boolean checkRights(String accessType,String role,
                    String dataCategory);
}
public class AccessRight implements Right{
    private String accessType;
    private RoleOfUser role;
    private RegistrationPersonalData data;

    public String getAccessType() {
        return accessType;
    }
    public void setAccessType(String accessType) {
        this.accessType = accessType;
    }
    public RoleOfUser getRole() {
        return role;
    }
    public void setRole(RoleOfUser role) {
        this.role = role;
    }
    public RegistrationPersonalData getData() {
        return data;
    }
    public void setData(RegistrationPersonalData data) {
        this.data = data;
    }
    @Override
    public void recordAccessType(String accessType) {
        // TODO Add your cord to record access type
        this.accessType = accessType;
    }
    @Override
    public boolean checkRights(String accessType,String role,String dataCategory)
        // TODO Add your code to check right to access personal data
        if(this.accessType.equalsIgnoreCase(accessType)
                && this.role.getRole().equalsIgnoreCase(role)
                && this.data.getDataCategory().equalsIgnoreCase(dataCategory))
            return true;
        return false;
```

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Sample Design and Code	To monitor access to personal data, the system uses the class RegistrationDataBreachMonitoring to check for suspicious access behavior, notify the data subject if access behavior is unusual, and handle the data breach. `interface DataBreachMonitoring extends DataManager{` `  public boolean checkDataBreach(UserOfSystem user,String date);` `  public void notifyDataBreach(UserOfSystem user);` `  public void HandleDataBreach();` `}` `public class RegistrationDataBreachMonitoring` `  extends RegistrationDataManager implements DataBreachMonitoring{` `  @Override` `  public boolean checkDataBreach(UserOfSystem user,String date) {` `  /* TODO Add your code to check for suspicious access behavior.` `  * if this access are suspicious the system should notify user` `  */` `  boolean checker = false;` `  SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");` `  Date dateCheck;` `  try {` `  dateCheck = sdf.parse(date);` `  UserEventManager uEvent = new UserEventManager();` `  uEvent.showEvent("Update", "Registration", dateCheck);` `  //Add your code to check for suspicious access behavior` `  } catch (ParseException e) {e.printStackTrace();}` `  if(!checker)` `  notifyDataBreach(user);` `  return checker;` `  }` `  @Override` `  public void notifyDataBreach(UserOfSystem user) {` `  /* TODO Add your code to notify data subject that access are` `  * suspicious i.e. notify email, send SMS and so on` `  */` `  SimpleDateFormat ft =` `  new SimpleDateFormat ("E yyyy.MM.dd 'at' hh:mm:ss a");` `  System.out.println("Send notice to email!!");` `  System.out.println("We noticed you logged into Account using");` `  System.out.println(user.getOs());` `  System.out.println("from "+user.getLocation());` `  System.out.println("date "+ft.format(user.getAccessDate()));` `  }` `  @Override` `  public void HandleDataBreach() {` `  // TODO Add your code to handle data breach` `  }` `}`
Known use	Examples of existing systems that take the Security Safeguards principle into account include: • Dropbox (Website): When a user logs in to Dropbox with a user name and password, the user is authenticated. In addition, if the user uses a new browser, a new operating system or a new location to log in, Dropbox will notify the data subject (or representative) to inform about a potential data breach.

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Known use	 • Google (Website): When a user logs in to Google with a user name and password, the user is authenticated. In addition, if the user uses a new browser, a new operating system or a new location to log in, Facebook will notify the data subject (or representative) to inform about a potential data breach. 
Related Patterns	Data Quality The Data Quality pattern can be used with the Security Safeguards pattern as preserving integrity of the collected personal data can ensure accuracy of the data as well as that a security mechanism is in place.

ตารางที่ ง.5 แบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

Related Patterns	Accountability The Accountability pattern can be used with the Security Safeguards pattern as a system needs to log events when monitoring to check for unusual activities so that proper actions can be taken by the system and data controller. Individual Participation The Individual Participation requires the Security Safeguards to ensure that the data subject can participate in the handling of his/her personal data with regard to access types.

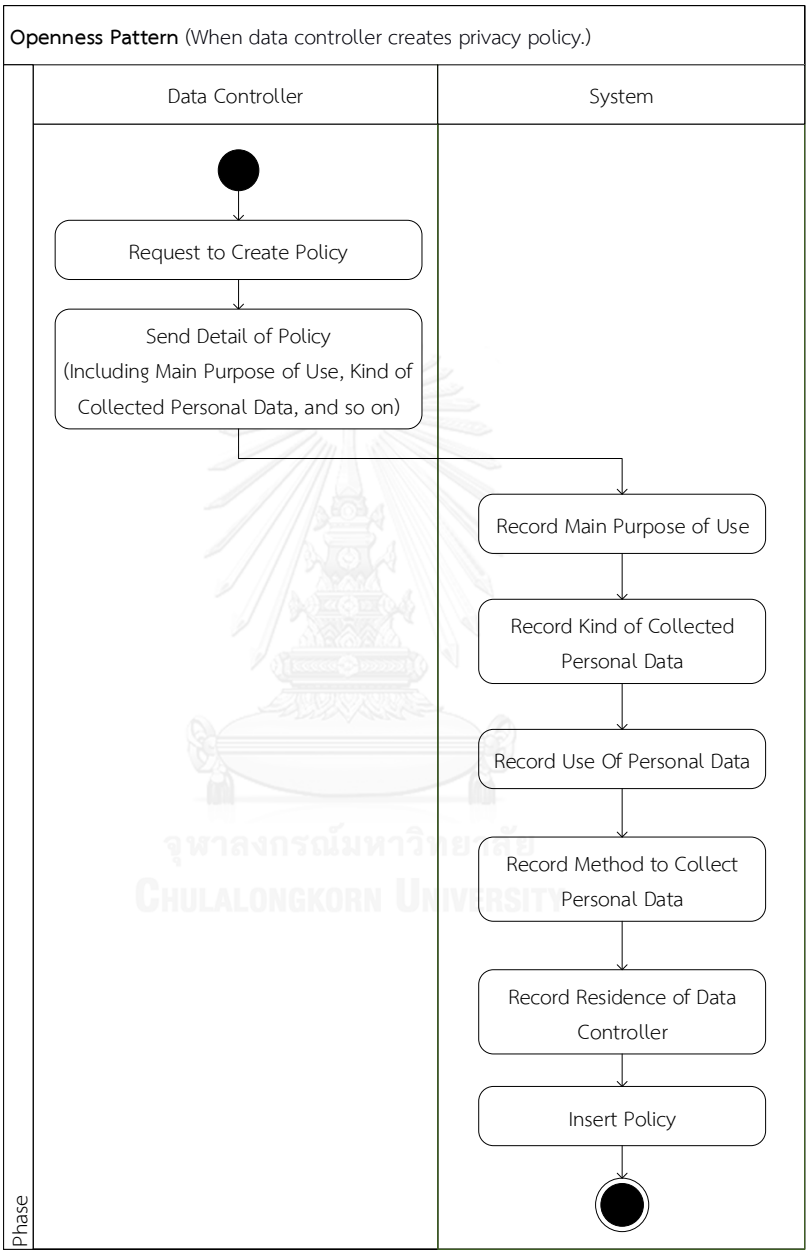
ง. 6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย

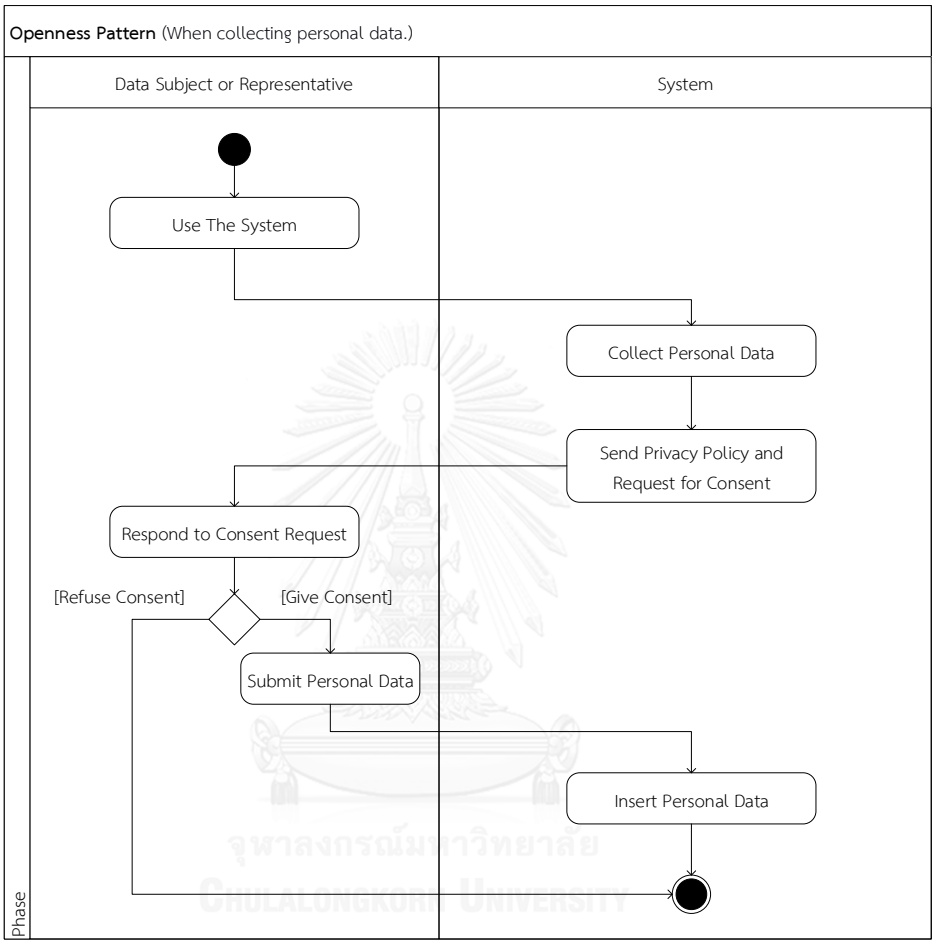
Pattern Name	**Openness Pattern**
Privacy Principle	Openness Pattern
Intent	This pattern assures that the general developments, practices, and policies with respect to the collected personal data should be open to a data subject. Such information should be accessible without unreasonable effort from the data subject and without unreasonable cost, since failure to do so may result in the data subject losing trust in using the system.
Motivation	When the system needs to collect certain personal data of the data subject for the processing of the system, the data subject should be informed of the general policy of the collection, storage, and use of personal data by the system. This is to allow the data subject to further control the use of his/her data throughout the use of the system, and, in the case of problems, to make requests to the right person in charge.
Applicability	Use the Openness pattern when a system of a data controller needs to • Create a privacy policy • Update a privacy policy be creating a new version of the policy • Inform the data subject when collecting personal data • Inform the data subject when the policy is changed and the new version of the policy becomes effective • Allow the data subject to access the current privacy policy.

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Behavior and Structure
Activity Diagram (When data controller creates privacy policy)

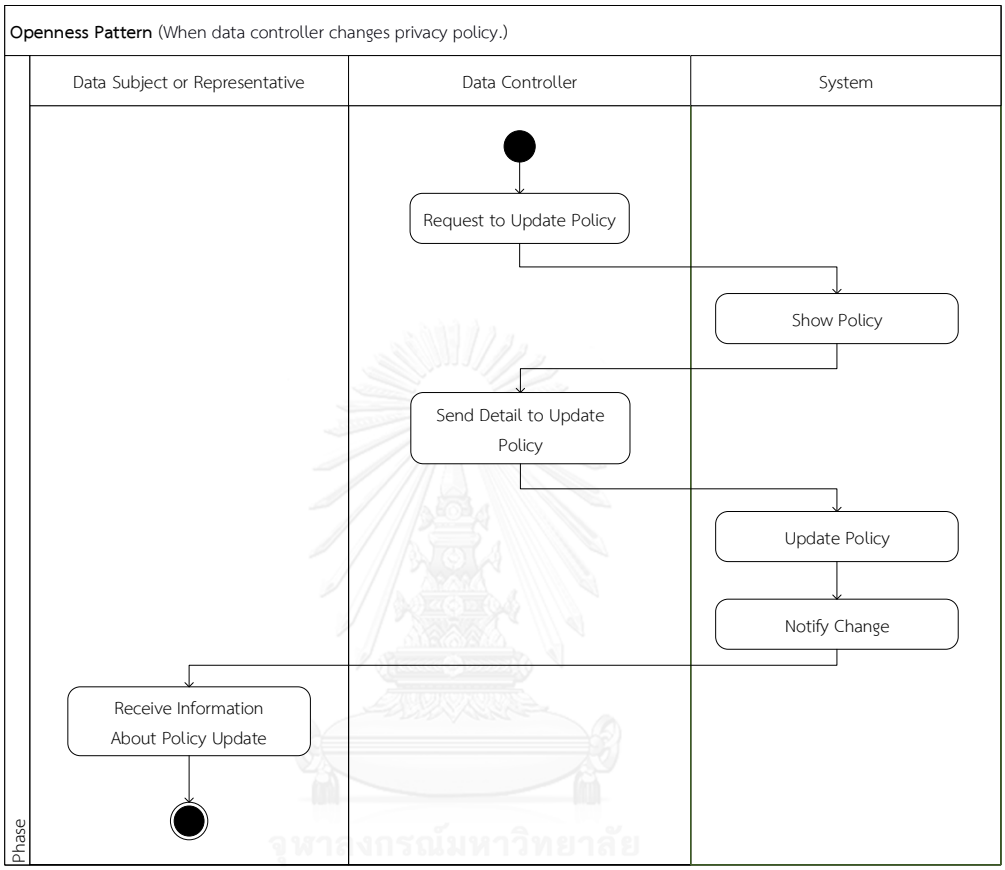


ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

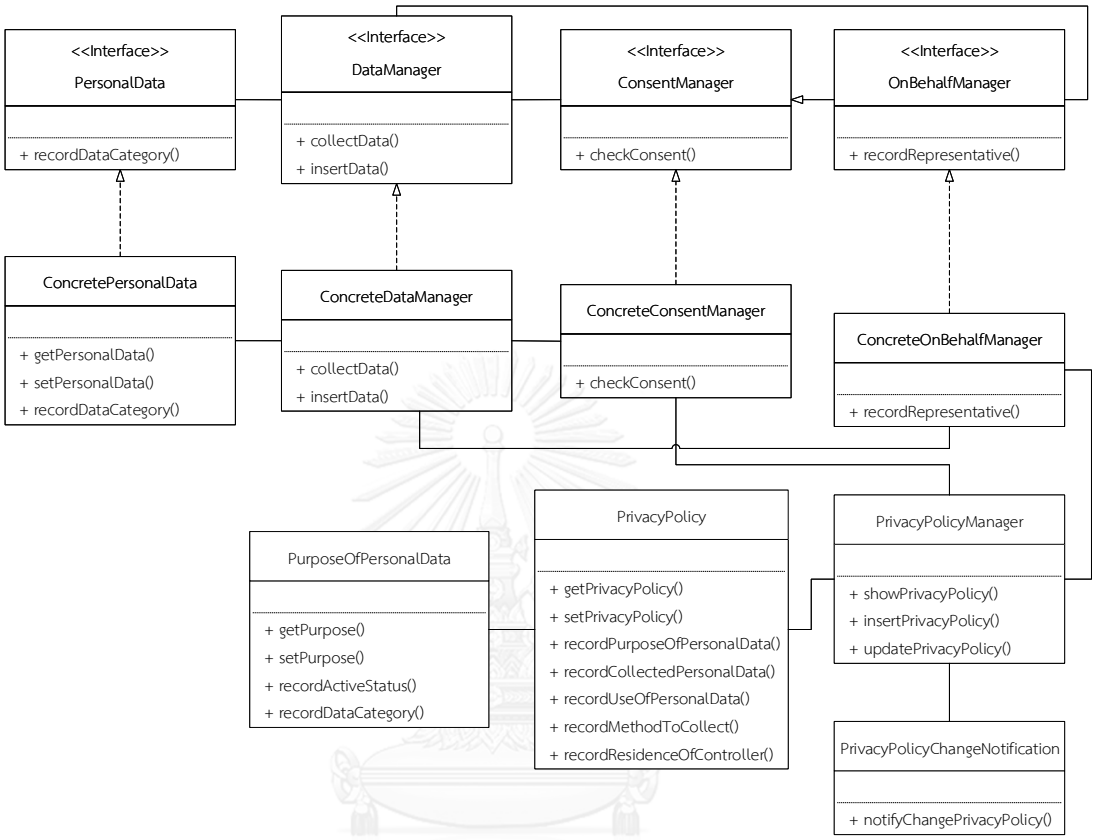
Behavior and Structure
Activity Diagram (When collecting personal data) 

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Behavior and Structure
Activity Diagram (When data controller changes privacy policy)



ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Behavior and Structure	
Class Diagram  	
Participants	
Interface Name/Class Name	**Detail**
Interface DataManager	An interface that has the responsibilities to coordinate checking of consent and collection of personal data from the data subject. It inserts the collected data into the system if the data subject gives consent.
Class ConcreteDataManager	A concrete class that implements the interface DataManager. Other responsibilities related to managing personal data can be added, i.e. coordinating update of personal data when there are changes etc.
Interface ConsentManager	An interface that has the responsibilities to inform the data subject that the system needs to collect personal data and check whether the data subject gives consent.

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

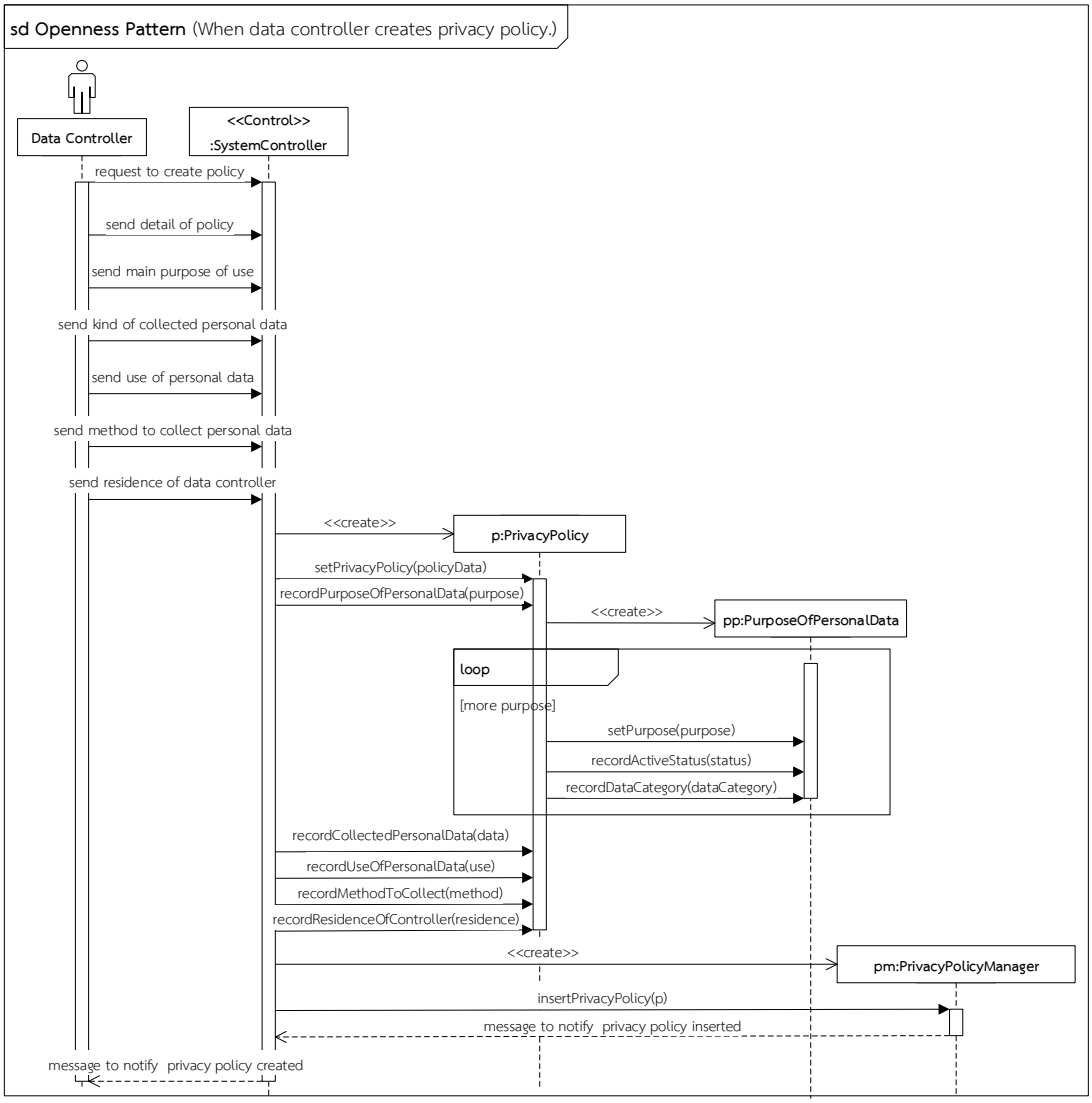
Participants	
Interface Name/Class Name	**Detail**
Class ConcreteConsentManager	A concrete class that implements the interface ConsentManager.
Interface OnBehalfManager	An interface that inherits from the interface ConsentManager. It has the responsibility to record the information about who, on behalf of the data subject, has given the consent to collect the personal data.
Class ConcreteOnBehalfManager	A concrete class that implements the interface OnBehalfManager.
Interface PersonalData	An interface that represents the collected personal data.
Class ConcretePersonalData	A concrete class with the data attributes that are necessary for the purpose of the system.
Class PrivacyPolicyManager	A concrete class that has the responsibilities to show and update the privacy policy.
Class PrivacyPolicy	A concrete class that represents the privacy policy of the system with the data attributes that record the policy content, including the collection method, purpose of use, and data controller information.
Class PurposeOfPersonalData	A concrete class that represents the purpose of use of personal data with the data attributes that record the purpose of use of personal data.
Class PrivacyPolicyChangeNotification	A concrete class that has the responsibilities to notify the data subject when the privacy policy is changed.
Collaboration	
1. When data controller creates privacy policy, 1.1. The system controller creates a PrivacyPolicy object to set the privacy policy from the data controller (with the detail of the policy, purpose of use, kind of collected personal data, use of personal data, method to collect personal data, and residence of data controller) into the PrivacyPolicy object. 1.2. To set the purpose of use, the PrivacyPolicy object creates a PurposeOfPersonalData object to set the purpose (with the active status and category of data that are collected for this purpose) into the PurposeOfPersonalData object.	

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Collaboration
1.3 The system controller creates the PrivacyPolicyManager object to insert the PrivacyPolicy object into a data store. 2 When collecting personal data. 2.1 The system controller creates a ConcreteDataManager object to collect personal data. 2.2 The ConcreteDataManager object checks for consent of the data subject: 2.2.1 The ConcreteDataManager object creates a ConcreteConsentManager object to check whether the data subject gives or refuse consent to collect his/her personal data. 2.2.2 The ConcreteConsentManager object creates a PrivacyPolicyManager object to view the privacy policy. The PrivacyPolicyManager object then creates a PrivacyPolicy object, gets the privacy policy from a data store, and sets the policy into the PrivacyPolicy object. 2.2.3 The PrivacyPolicy objects creates PurposeOfPersonalData objects and sets the purposes of use into them. 2.2.4 The PrivacyPolicyManager object sends the privacy policy to the ConcreteConsentManager object which in turn will pass on the policy to the data subject. 2.3 If the data subject gives consent, the system controller creates a ConcretePersonalData object and calls the ConcreteDataManager to insert the ConcretePersonalData object into a data store. 3 When data controller changes privacy policy, 3.1 The system controller creates a PrivacyPolicyManager object to view the privacy policy. The PrivacyPolicyManager object then creates a PrivacyPolicy object, gets the privacy policy from a data store, and sets the policy into the PrivacyPolicy object which in turn will be sent to the data controller. 3.2 If the data controller updates the privacy policy, the system controller creates a new PrivacyPolicy object to set the policy updates into the PrivacyPolicy object. 3.3 If the purpose in the policy are change or update, the PrivacyPolicy object creates a PurposeOfPersonalData object to sets the purpose updates into the PurposeOfPersonalData object 3.4 The system controller calls the PrivacyPolicyManager object to update the PrivacyPolicy object into a data store.

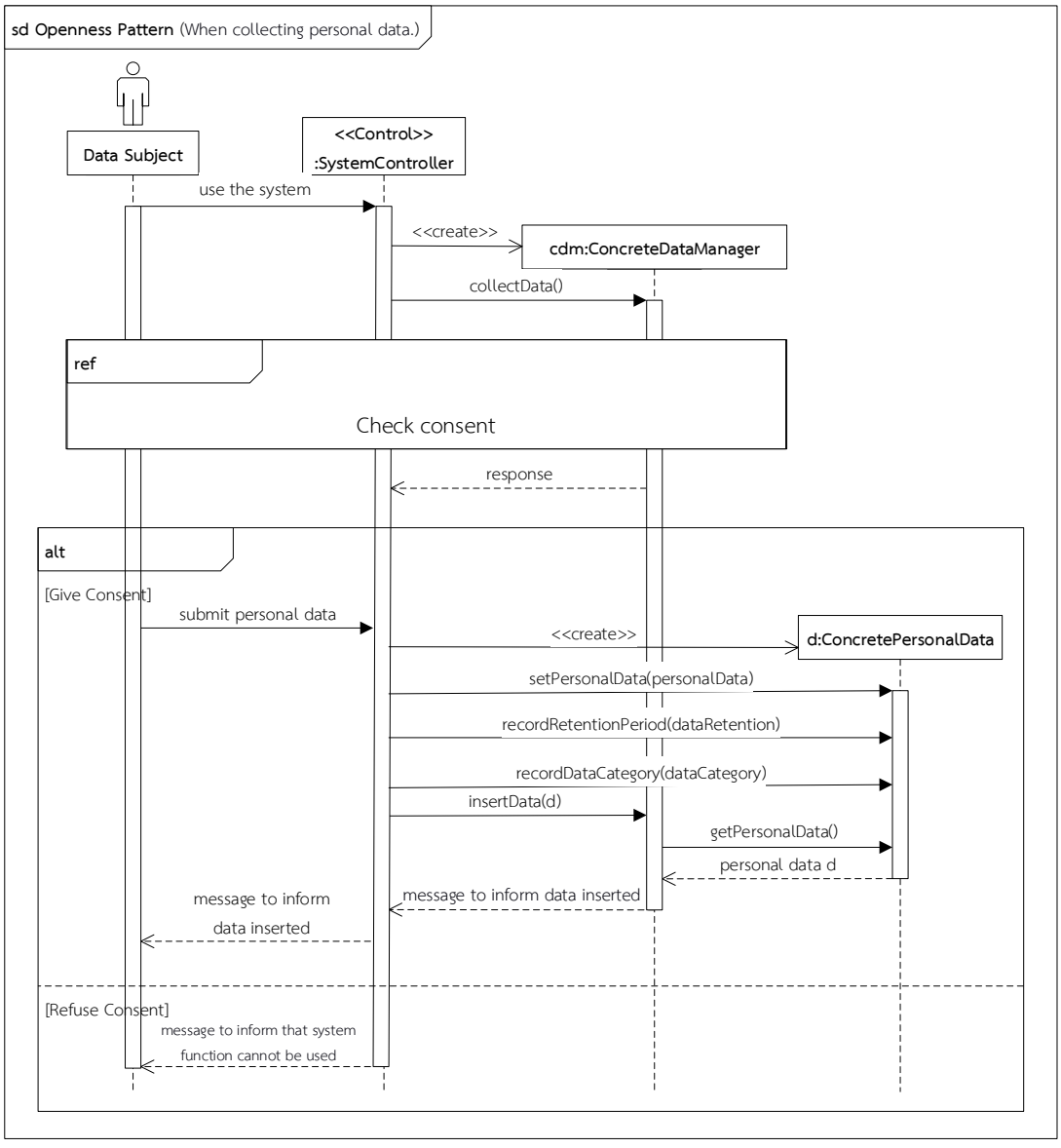
ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Collaboration
3.5 If the update is successful, the PrivacyPolicyManager object creates a PrivacyPolicyChangeNotification object to inform the data subject about privacy policy change.
Sequence Diagram (When data controller creates privacy policy)

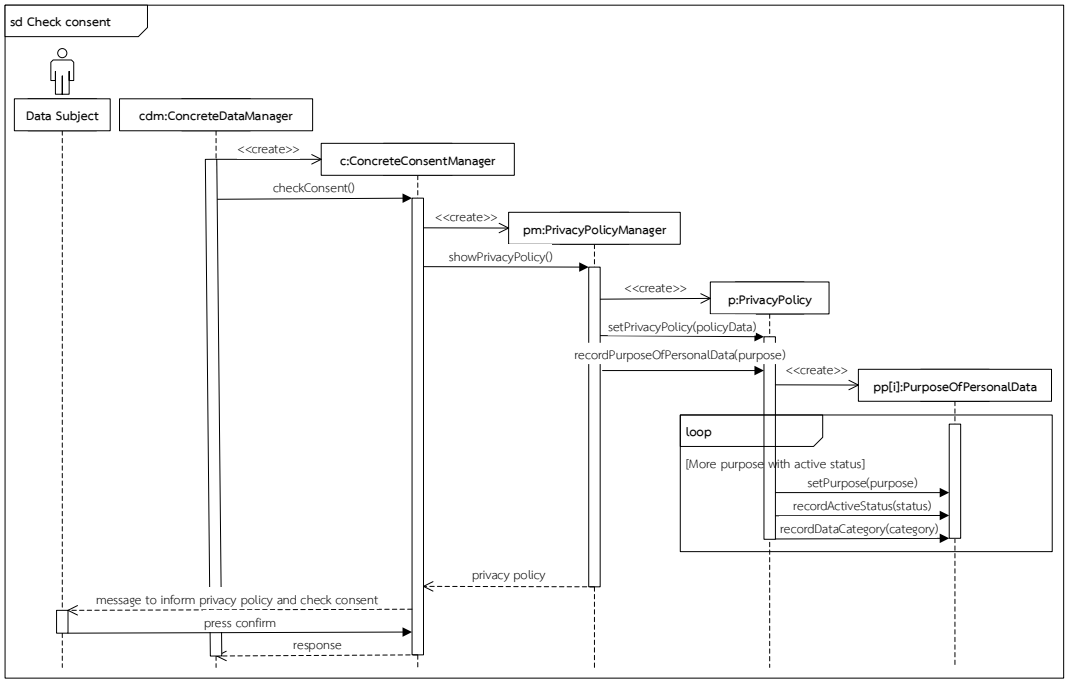


ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data.)

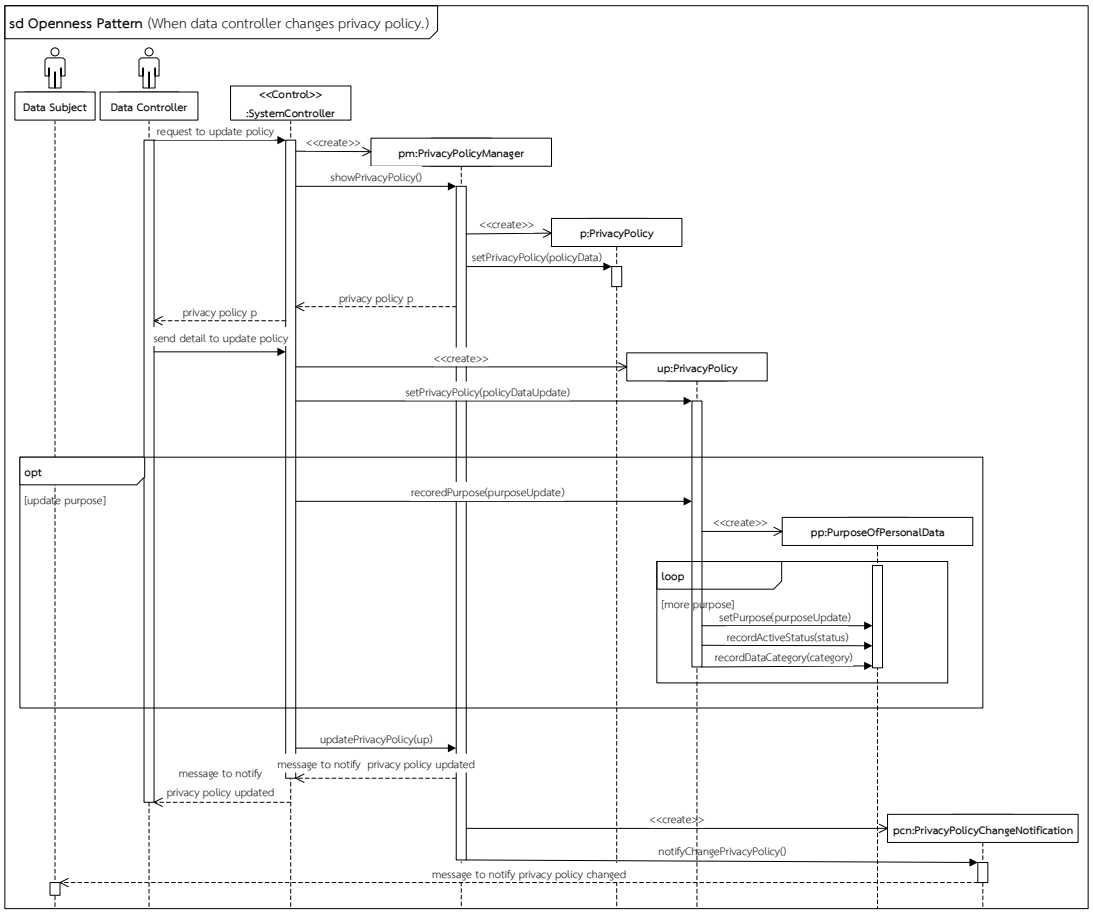


ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Collaboration
Sequence Diagram (When collecting personal data. - continued) 

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sequence Diagram (When data controller changes privacy policy)



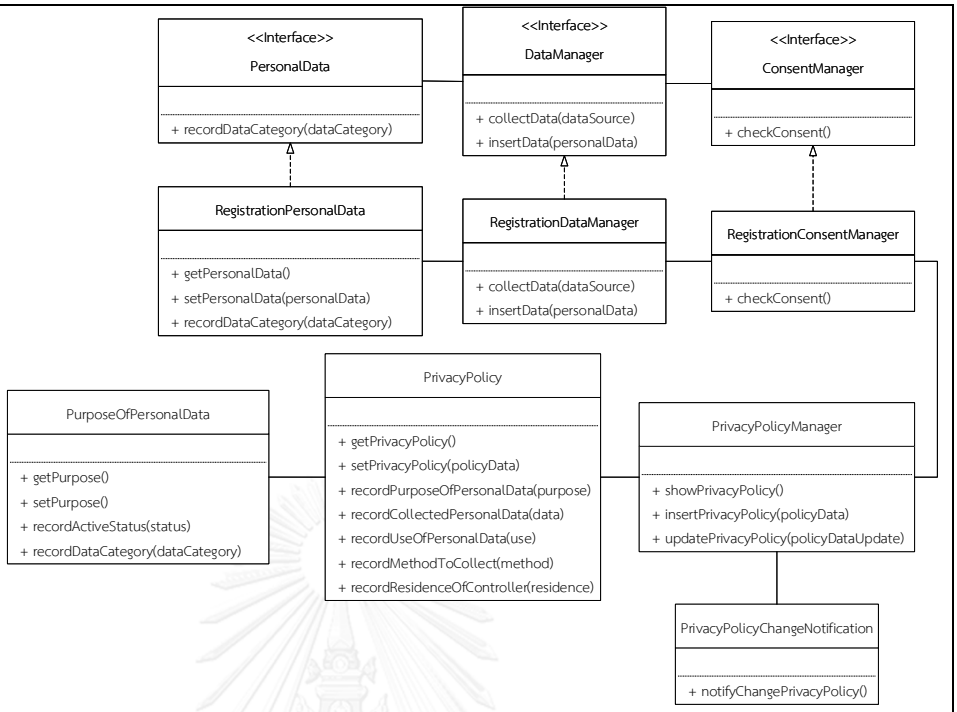
Consequences	Using this pattern, the openness of the system with regard to personal data can be achieved as recommended by OECD. The system can create a current version of the privacy policy while maintaining the policy history. The data subject can be assured that he/she is informed of the current policy that controls the collection, storage, and use of personal data by which trust in the system can be built.

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Implementation	Issues regarding the implementation of the Openness pattern are as follows. 1. The OECD defines that the data subject should be able to obtain privacy policy information without unreasonable effort and unreasonable cost, and regular information from the data controller regarding the privacy policy should be published. To address this issue, the class PrivacyPolicyManager should implement the method showPrivacyPolicy for another class to obtain privacy policy information. The system 2. The system may retain only the latest version of the PrivacyPolicy object, specifying the creation date and latest update date of the policy. In certain cases, the system may be implemented to retain a history of policy changes as a collection of different versions of the PrivacyPolicy objects. 3. The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle since the data subject should first acquire the information about what, when, why, and how his/her personal data are collected and used before he/she can take part in the processing of personal data. The Openness pattern should already be in place when the Individual Participation pattern is applied.
Sample Design and Code	An example of a user registration system that applies the Openness pattern is shown below. Any user who requests to use the system is informed of the privacy policy at the time personal data are collected.

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code



When a user (data subject) is required to register with the system to use its service, the system uses the class RegistrationDataManager that implements the interface DataManager to collect personal data (name, email, tel) as shown in the sample code below.

```
public interface DataManager {
    public boolean collectData();
    public void insertData(RegistrationPersonalData data);
}
```

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	**Sample Code** `public class RegistrationDataManager implements DataManager{` `@Override` `public boolean collectData(){` `//TODO Add your code to check whatever user give or refuse consent` `boolean checker = false;` `RegistrationConsentManager consent = new RegistrationConsentManager();` `if(consent.checkConsent())` `checker = true;` `return checker;` `}` `@Override` `public void insertData(RegistrationPersonalData data){` `//TODO Add your code to insert personal data into data store.` `DatabaseConnection dbConnection = new DatabaseConnection();` `Connection connection = dbConnection.getDBConnection();` `try {` `String insert = "INSERT INTO personaldata "` `+ "(name, email, tel, data_category) "` `+ "VALUES (?, ?, ?, ?)" ;` `PreparedStatement stmt = connection.prepareStatement(insert) ;` `int parameterIndex = 1;` `stmt.setString(parameterIndex++, data.getName());` `stmt.setString(parameterIndex++, data.getEmail());` `stmt.setString(parameterIndex++, data.getTel());` `stmt.setString(parameterIndex++, data.getDataCategory());` `stmt.execute();` `stmt.close();` `} catch (SQLException e) {}` `}` `}` When data are collected directly from the user, the class RegistrationConsentManager that implements the interface ConsentManager acquires the privacy policy of the system to inform the user. It also checks whether the user gives or refuse consent to personal data collection. The sample code is shown below. `public interface ConsentManager {` `public boolean checkConsent();` `}`

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	

```
public class RegistrationConsentManager implements ConsentManager{
    private Scanner scan;
    @Override
    public boolean checkConsent() {
        /* TODO Add your code to show policy to user and check whatever
         * user give or refuse consent
         */
        scan = new Scanner(System.in);
        PrivacyPolicyManager policyManager = new PrivacyPolicyManager();

        PrivacyPolicy policy = policyManager.showPrivacyPolicy();
        System.out.println("Name :: "+policy.getName());
        System.out.println("Collected Data :: "+policy.getDataCollected());
        System.out.println("Use Data :: "+policy.getDataUse());
        System.out.println("Collected Method :: "+policy.getCollectionMethod())
        for(int i=0; i<policy.getPurpose().size(); i++){
            System.out.println("Purpose "+i+" :: "
                    +policy.getPurpose().get(i).getDetail());
        }
        System.out.println("Residence :: "+policy.getResidenceOfController());

        System.out.println("Give Consent Please Enter 'Y' "
                + "and Refuse Consent Please Enter 'N'");
        String accept = scan.next();
        if(accept.equalsIgnoreCase("Y"))
            return true;
        else
            return false;
    }
}
```

When a data controller wants to show or update (or change) the privacy policy, the system uses the class SystemPrivacyPolicyManager that implements the interface PrivacyPolicyManager to show or update the policy of the system.

```
public interface PrivacyPolicyManager {
    public SystemPrivacyPolicy showPrivacyPolicy();
    public void insertPrivacyPolicy(SystemPrivacyPolicy policy);
    public void updatePrivacyPolicy(SystemPrivacyPolicy policy);
}
public class SystemPrivacyPolicyManager implements PrivacyPolicyManager{

    Date now = new Date();
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    Calendar cal = Calendar.getInstance();

    @Override
    public SystemPrivacyPolicy showPrivacyPolicy() {
        // TODO Add your code to show privacy policy
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        SystemPrivacyPolicy pp = new SystemPrivacyPolicy();
        PreparedStatement stmt;
        try {
            String query1 = "SELECT * FROM privacypolicy ORDER BY id DESC LIMIT 1" ;
            stmt = connection.prepareStatement(query1) ;
            ResultSet  result1 = stmt.executeQuery();
            while(result1.next()){
                pp.setId(result1.getString("id"));
                pp.setName(result1.getString("name"));
                pp.setDataCollected(result1.getString("collecteddata"));
                pp.setDataUse(result1.getString("usedata"));
                pp.setCollectionMethod(result1.getString("collectedmethod"));
                pp.setResidenceOfController(result1.getString("residence"));
            }
```

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	

```
            JSONArray jsonArr = new JSONArray();
            String query2 = "SELECT * FROM purpose where active_status = 'Active' " ;
            stmt = connection.prepareStatement(query2) ;
            ResultSet result2 = stmt.executeQuery();
            while(result2.next()){
                JSONObject json = new JSONObject();
                json.put("id", result2.getString("id"));
                json.put("detail", result2.getString("detail"));
                json.put("data_category", result2.getString("data_category"));
                json.put("active_status", result2.getString("active_status"));
                jsonArr.put(json);
            }
            pp.recordPurposeOfPersonalData(jsonArr);

            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
          catch (JSONException e) {e.printStackTrace();}
        return pp;
    }
    @Override
    public void insertPrivacyPolicy(SystemPrivacyPolicy policy) {
        // TODO Add your code to insert privacy policy
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        PreparedStatement stmt;
        try{
            String insert1 = "INSERT INTO privacypolicy "
                    + "(name, detail, collecteddata, collectedmethod, usedata,"
                    + "residence, createdate, updatedate) "
                    + "VALUES (?, ?, ?, ?, ?, ?, ?, ?)" ;

            stmt = connection.prepareStatement(insert1) ;
            int parameterIndex1 = 1;

            stmt.setString(parameterIndex1++, policy.getName());
            stmt.setString(parameterIndex1++, policy.getDetail());
            stmt.setString(parameterIndex1++, policy.getDataCollected());
            stmt.setString(parameterIndex1++, policy.getCollectionMethod());
            stmt.setString(parameterIndex1++, policy.getDataUse());
            stmt.setString(parameterIndex1++, policy.getResidenceOfController());
            stmt.setTimestamp(parameterIndex1++, new Timestamp(now.getTime()));
            stmt.setTimestamp(parameterIndex1++, new Timestamp(now.getTime()));
            stmt.execute();
            for(int i=0; i<policy.getPurpose().size(); i++){
                String insert2 = "INSERT INTO purpose "
                        + "(detail, data_category, active_status, "
                        + "createdate, updatedate) "
                        + "VALUES (?, ?, ?, ?, ?)" ;

                stmt = connection.prepareStatement(insert2) ;
                int parameterIndex2 = 1;
                stmt.setString(parameterIndex2++, policy.getPurpose().get(i).getDetail());
                stmt.setString(parameterIndex2++, policy.getPurpose().get(i).getDataCategory());
                stmt.setString(parameterIndex2++, policy.getPurpose().get(i).getActiveStatus());
                stmt.setTimestamp(parameterIndex2++, new Timestamp(now.getTime()));
                stmt.setTimestamp(parameterIndex2++, new Timestamp(now.getTime()));
                stmt.execute();
            }

            stmt.close();

        }catch(SQLException e) {e.printStackTrace();}
    }
    @Override
    public void updatePrivacyPolicy(SystemPrivacyPolicy policy) {
        // TODO Add your code to update privacy policy
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        PreparedStatement stmt;
        try {
            String insert1 = "INSERT INTO privacypolicy "
                    + "(name, detail, collecteddata, collectedmethod, usedata,"
                    + "residence, createdate, updatedate) "
                    + "VALUES (?, ?, ?, ?, ?, ?, ?, ?)" ;

            stmt = connection.prepareStatement(insert1) ;
            int parameterIndex1 = 1;

            stmt.setString(parameterIndex1++, policy.getName());
            stmt.setString(parameterIndex1++, policy.getDetail());
            stmt.setString(parameterIndex1++, policy.getDataCollected());
            stmt.setString(parameterIndex1++, policy.getCollectionMethod());
            stmt.setString(parameterIndex1++, policy.getDataUse());
            stmt.setString(parameterIndex1++, policy.getResidenceOfController());
            stmt.setTimestamp(parameterIndex1++, new Timestamp(now.getTime()));
            stmt.setTimestamp(parameterIndex1++, new Timestamp(now.getTime()));
            stmt.execute();
```

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	

```
            if(policy.getPurpose() != null){
                for(int i=0; i<policy.getPurpose().size(); i++){
                    if(policy.getPurpose().get(i).getActiveStatus().equals("Yes")){
                        String insert2 = "INSERT INTO purpose "
                                + "(detail, data_category, active_status, "
                                + "createdate, updatedate) "
                                + "VALUES (?, ?, ?, ?, ?)" ;

                        stmt = connection.prepareStatement(insert2) ;
                        int parameterIndex2 = 1;
                        stmt.setString(parameterIndex2++,
                                policy.getPurpose().get(i).getDetail());
                        stmt.setString(parameterIndex2++,
                                policy.getPurpose().get(i).getDataCategory());
                        stmt.setString(parameterIndex2++,
                                policy.getPurpose().get(i).getActiveStatus());
                        stmt.setTimestamp(parameterIndex2++, new Timestamp(now.getTime()));
                        stmt.setTimestamp(parameterIndex2++, new Timestamp(now.getTime()));
                        stmt.execute();
                    } else{
                        StringBuffer upd = new StringBuffer("UPDATE purpose SET ");
                        upd.append("active_status = '"+policy.getPurpose().get(i).getActiveStatus()+"', ");
                        upd.append("updatedate = '"+sdf.format(now)+"' ");
                        upd.append("WHERE id = '"+policy.getPurpose().get(i).getId()+"' ");
                        stmt = connection.prepareStatement(upd.toString()) ;
                        stmt.execute();
                    }
                }
            }

            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
        //TODO Add your code to notify user
        PrivacyPolicyChangeNotification notice = new PrivacyPolicyChangeNotification();
        notice.notifyChangePrivacyPolicy();
    }
}
```

The class PrivacyPolicy is implemented to have the data attributes that record the policy content, including the purpose of collection, what personal data are collected, use of personal data, collection method, and data controller information as shown in the sample code below.

```
public class PrivacyPolicy {
    private String id;
    private String name;
    private String detail;
    private ArrayList<PurposeOfPersonalData> purpose;
    private String dataCollected;
    private String dataUse;
    private String collectionMethod;
    private String residenceOfController;
    private Date dateCreation;
    private Date dateUpdate;

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getDetail() {
        return detail;
    }
    public void setDetail(String detail) {
        this.detail = detail;
    }
```

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	

```
public ArrayList<PurposeOfPersonalData> getPurpose() {
    return purpose;
}
public void setPurpose(ArrayList<PurposeOfPersonalData> purpose) {
    this.purpose = purpose;
}
public String getDataCollected() {
    return dataCollected;
}
public void setDataCollected(String dataCollected) {
    this.dataCollected = dataCollected;
}
public String getDataUse() {
    return dataUse;
}
public void setDataUse(String dataUse) {
    this.dataUse = dataUse;
}
public String getCollectionMethod() {
    return collectionMethod;
}
public void setCollectionMethod(String collectionMethod) {
    this.collectionMethod = collectionMethod;
}
public String getResidenceOfController() {
    return residenceOfController;
}
public void setResidenceOfController(String residenceOfController) {
    this.residenceOfController = residenceOfController;
}
public Date getDateCreation() {
    return dateCreation;
}
public void setDateCreation(Date dateCreation) {
    this.dateCreation = dateCreation;
}
public Date getDateUpdate() {
    return dateUpdate;
}
public void setDateUpdate(Date dateUpdate) {
    this.dateUpdate = dateUpdate;
}
@Override
public void recordPurposeOfPersonalData(JSONArray purpose) {
    //TODO Add your code to record purpose
    this.purpose = new ArrayList<SystemPurposeOfPersonalData>();
    for(int i=0; i<purpose.length(); i++){
        SystemPurposeOfPersonalData p = new SystemPurposeOfPersonalData();
        try {
            p.setId(purpose.getJSONObject(i).getString("id"));
            p.setDetail(purpose.getJSONObject(i).getString("detail"));
            p.recordDataCategory(purpose.getJSONObject(i).getString("data_category"));
            p.recordActiveStatus(purpose.getJSONObject(i).getString("active_status"));
        } catch (JSONException e) {e.printStackTrace();}
        this.purpose.add(p);
    }
}

@Override
public void recordCollectedPersonalData(String data) {
    //TODO Add your code to record collected personal data
    this.dataCollected = data;
}

@Override
public void recordUseOfPersonalData(String use) {
    //TODO Add your code to record use of personal data
    this.dataUse = use;
}

@Override
public void recordMethodToCollect(String method) {
    //TODO Add your code to record method to collect personal data
    this.collectionMethod = method;
}

@Override
public void recordResidenceOfController(String residence) {
    //TODO Add your code to record residence of data controller
    this.residenceOfController = residence;
}
```

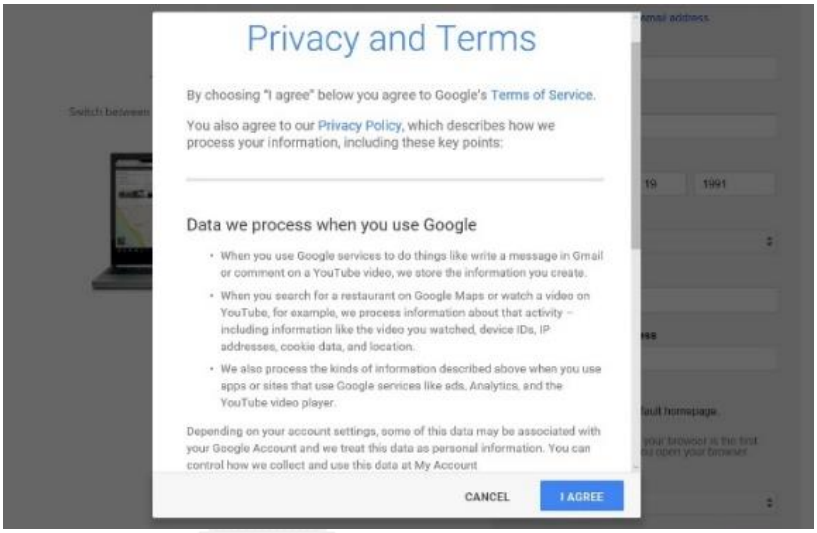
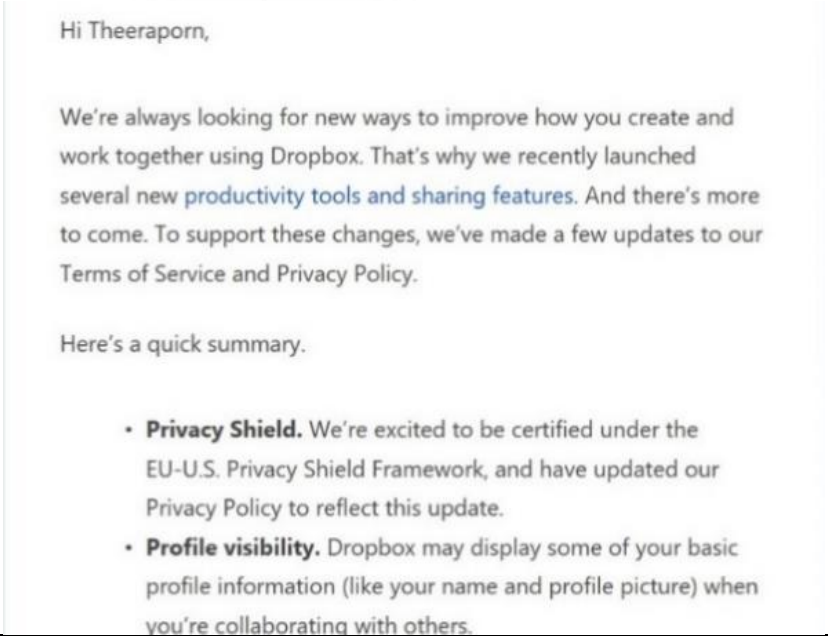
ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	The class PurposeForPersonalData is implemented to have the data attributes that store the purpose of the collection and use of personal data. The sample code is shown below. `public class PurposeOfPersonalData {` `  private String id;` `  private String detail;` `  private String dataCategory;` `  private String activeStatus;` `  private Date dateCreation;` `  private Date dateUpdate;` `  public String getId() {` `  return id;` `  }` `  public void setId(String id) {` `  this.id = id;` `  }` `  public String getDetail() {` `  return detail;` `  }` `  public void setDetail(String detail) {` `  this.detail = detail;` `  }` `  public String getDataCategory() {` `  return dataCategory;` `  }` `  public String getActiveStatus() {` `  return activeStatus;` `  }` `  public Date getDateCreation() {` `  return dateCreation;` `  }` `  public void setDateCreation(Date dateCreation)` `  this.dateCreation = dateCreation;` `  }` `  public Date getDateUpdate() {` `  return dateUpdate;` `  }` `  public void setDateUpdate(Date dateUpdate) {` `  this.dateUpdate = dateUpdate;` `  }` `  public void recordActiveStatus(String activeStatus) {` `  // // TODO Add your code to record purpose status` `  this.activeStatus = activeStatus;` `  }` `  public void recordDataCategory(String dataCategory) {` `  // TODO Add your code to record type of collected personal data` `  this.dataCategory = dataCategory;` `  }` `}` The class RegistrationPersonalData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of the system as shown in the sample code below.

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Sample Design and Code	**Code:** `public interface PersonalData {` `  public void recordDataCategory(String dataCategory);` `}` `public class RegistrationPersonalData implements PersonalData{` `  private String name;` `  private String email;` `  private String tel;` `  private String dataCategory;` `  public String getName() {` `  return name;` `  }` `  public void setName(String name) {` `  this.name = name;` `  }` `  public String getEmail() {` `  return email;` `  }` `  public void setEmail(String email) {` `  this.email = email;` `  }` `  public String getTel() {` `  return tel;` `  }` `  public void setTel(String tel) {` `  this.tel = tel;` `  }` `  public String getDataCategory() {` `  return dataCategory;` `  }` `  public void setDataCategory(String dataCategory) {` `  this.dataCategory = dataCategory;` `  }` `  @Override` `  public void recordDataCategory(String dataCategory) {` `  // TODO Add your code to record category of collected personal data` `  this.dataCategory = dataCategory;` `  }` `}` In the case that the data controller updates the privacy policy of the system, the system uses the class PrivacyPolicyChangeNotification to notify the user about change. `public class PrivacyPolicyChangeNotification {` `  public void notifyChangePrivacyPolicy(){` `  //TODO Add your code to notify change of privacy policy to data subject` `  System.out.println("We are updating the Privacy Policy."` `  + "We want to notify you about these updates.");` `  }` `}`

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Known use	Examples of existing systems that take the Openness principle into account include: ● Google (Website): When a user creates an account, the system will collect personal data and inform the user of the privacy policy that describes how the system process his/her personal data. If the user gives consent to the system to collect and use personal data, the user should click the agreement button.  ● Dropbox (Website): When a system changes the privacy policy, the system will inform the user of the change.  Hi Theeraporn, We're always looking for new ways to improve how you create and work together using Dropbox. That's why we recently launched several new productivity tools and sharing features. And there's more to come. To support these changes, we've made a few updates to our Terms of Service and Privacy Policy. Here's a quick summary. • **Privacy Shield.** We're excited to be certified under the EU-U.S. Privacy Shield Framework, and have updated our Privacy Policy to reflect this update. • **Profile visibility.** Dropbox may display some of your basic profile information (like your name and profile picture) when you're collaborating with others.

ตารางที่ ง.6 แบบรูปการออกแบบสำหรับหลักการเปิดเผย (ต่อ)

Related Patterns	Collection Limitation The Collection Limitation pattern requires the Openness pattern to ensure that the data subject is informed of the privacy policy related to the collection of the personal data, where the policy includes the detail on what personal data are collected and for what purposes. Individual Participation The Individual Participation pattern requires the Openness pattern to establish the privacy policy and inform the data about what, when, why, and how his/her personal data are collected and used before he/she can take part in the processing of personal data.

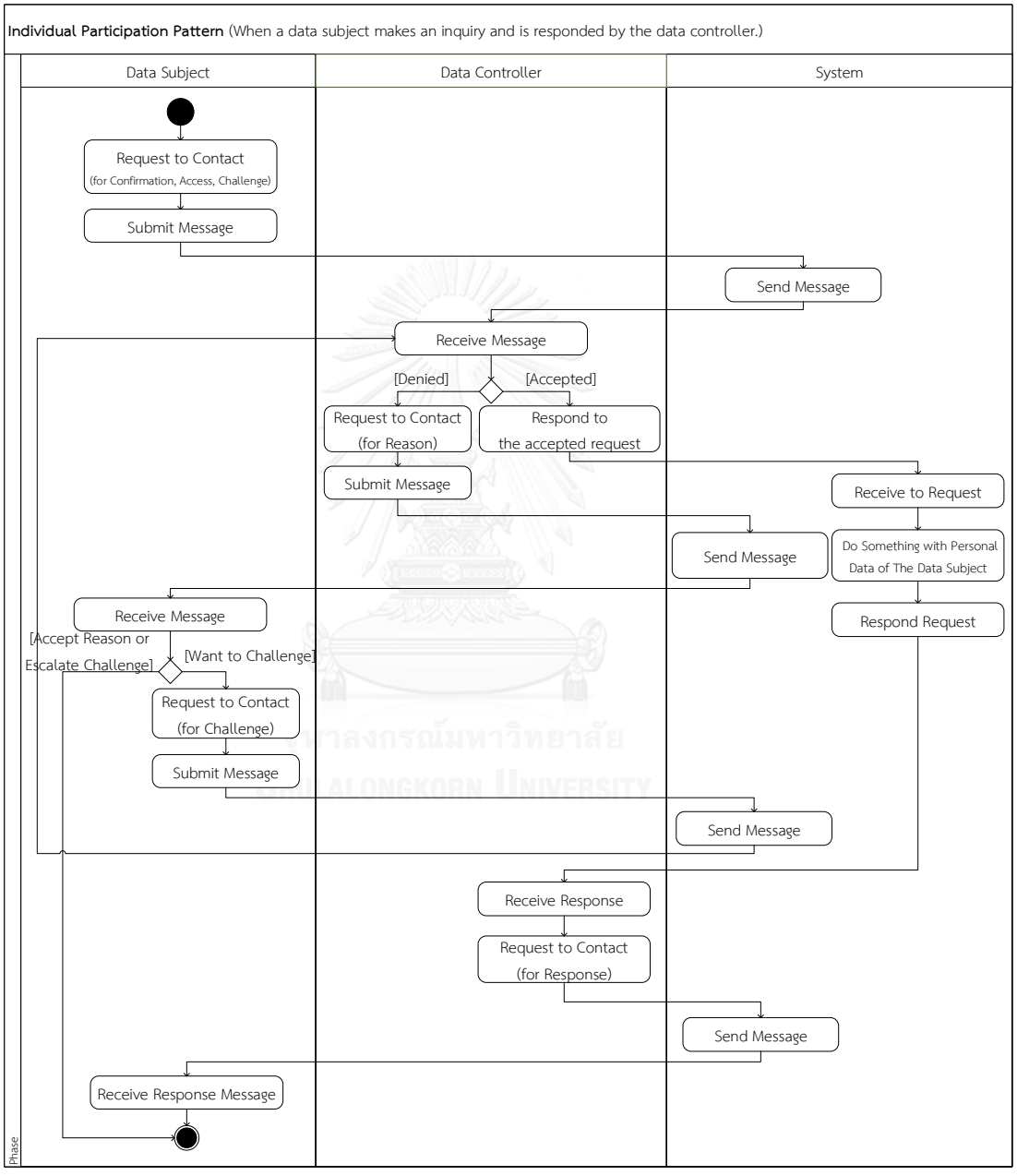
ง. 7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล

Pattern Name	**Individual Participation Pattern**
Privacy Principle	Individual Participation Principle
Intent	This pattern assures that a data subject has the rights to participate in his/her personal data that are handled by the data controller and other parties, such as a data processor (if any). Participation includes request for information, access, be given reasons, and challenge, with regard to personal data.
Motivation	Throughout the use of a service by a data subject, several categories of personal data are collected, used, and stored. For example, user profile information may be first collected during registration. Details of access and activities of the data subject may be recorded during the use of the service. Certain data are readily available for access while other data are not. To preserve the data subject's right to control his/her personal data, the data controller should provide a convenient and efficient way for the data subject to access and make a request about personal data when needed. Where there is a dispute, the data subject's right to control personal data should be protected. That is, reasons or explanations should be given by the data controller and further challenges can be made by the data subject.
Applicability	Use the Individual Participation pattern throughout the life cycle of personal data within the system. That is, use this pattern when • The data subject needs to inquire about his/her personal data. • The data subject requests to access (i.e. view, update, erase) his/her personal data. • The data subject is to be given reasons for adverse decisions when a request is denied. • The data subject needs to challenge adverse decisions when a request is denied. • The data subject needs to challenge the use of his/her personal data by the data controller. • The data controller responds to a successful challenge by erasing, rectifying, completing, or amending personal data.

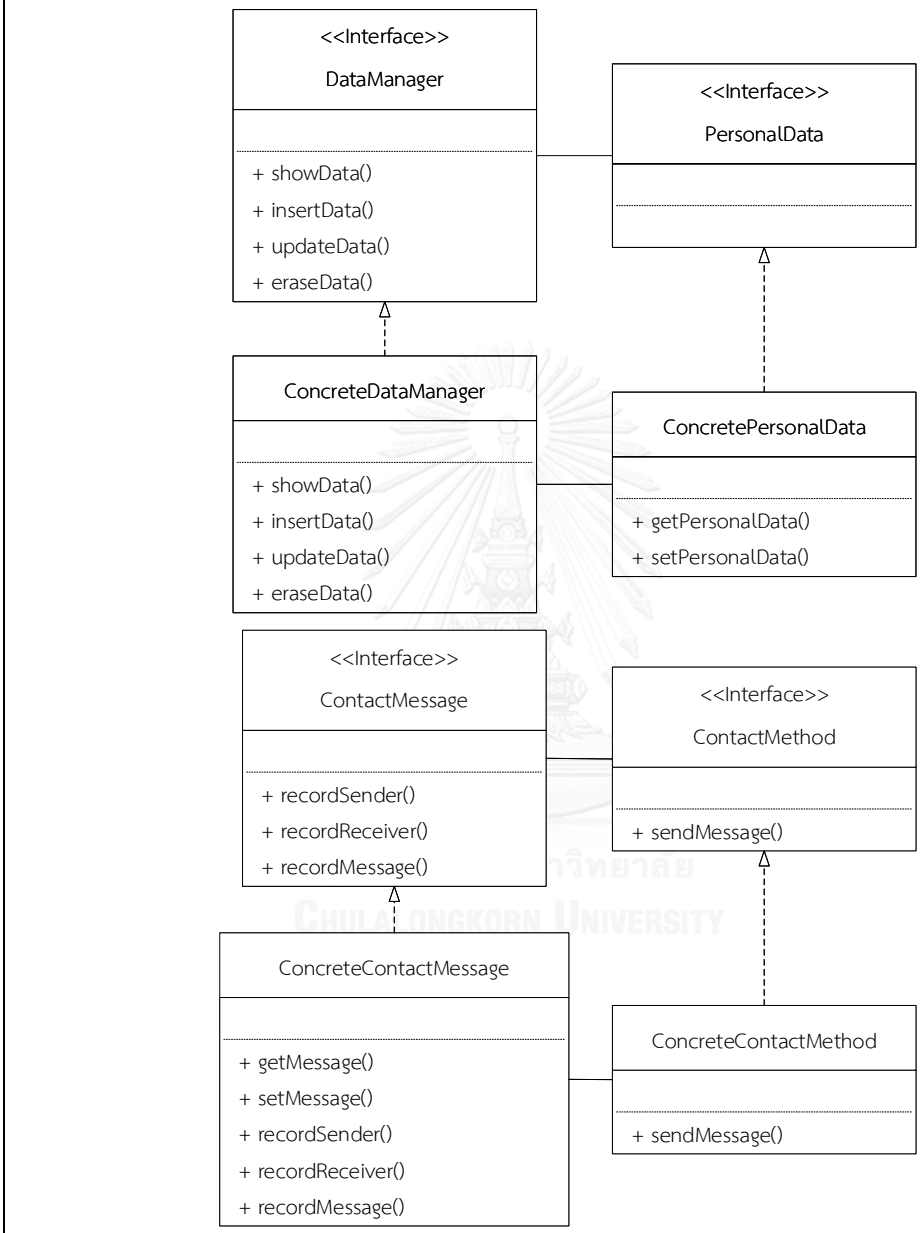
ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Behavior and Structure
Activity Diagram (When a data subject makes an inquiry and is responded by the data controller.)



ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Behavior and Structure
Class Diagram



ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

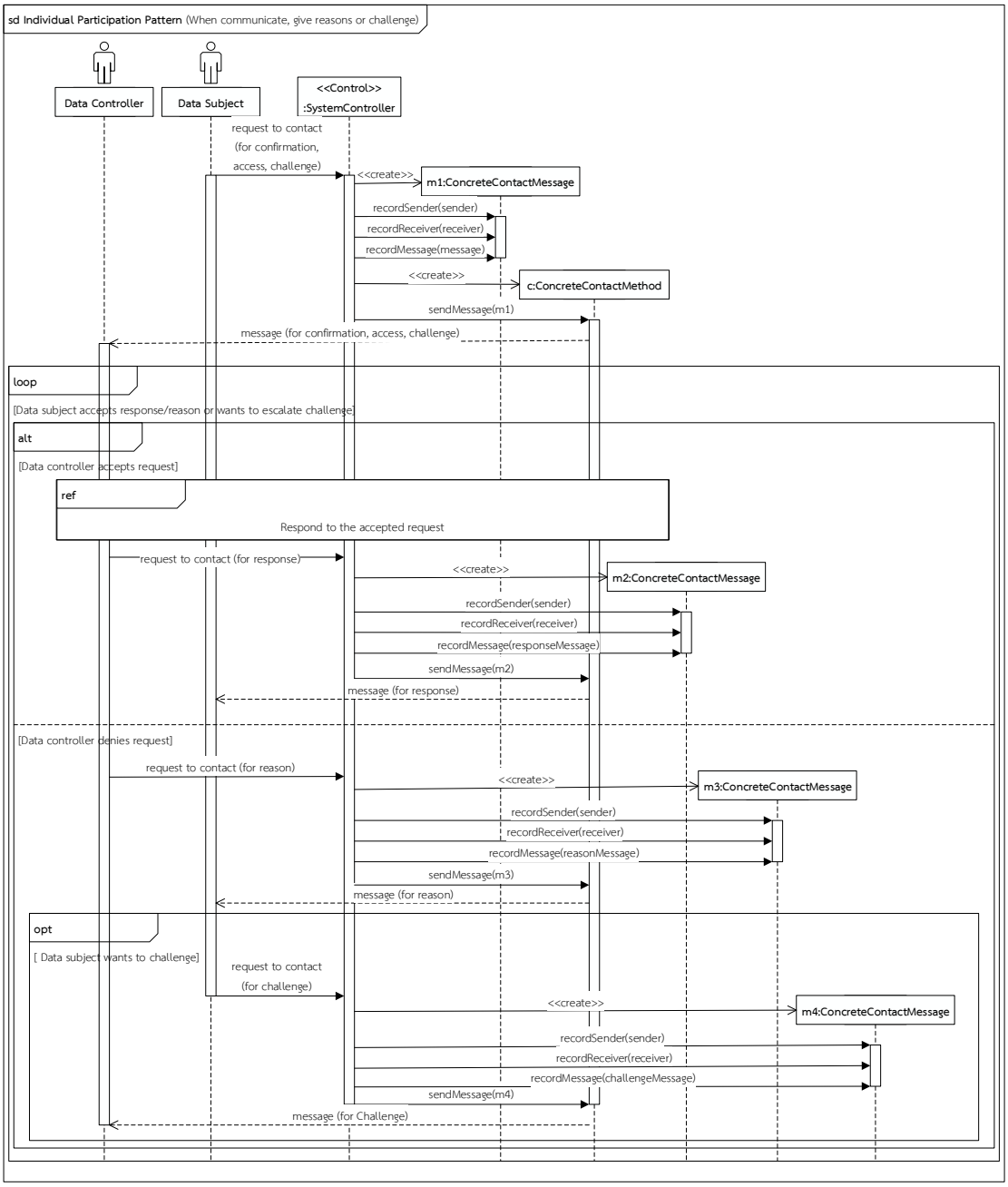
Participants	
Interface Name/Class Name	**Detail**
Interface DataManager	An interface that has the responsibilities to manage personal data, including show, insert, update, and erase personal data.
Class ConcreteDataManager	A concrete class that implements the interface DataManager.
Interface PersonalData	An interface that represents the collected personal data.
Class ConcretePersonalData	A concrete class that implements the interface PersonalData with the data attributes that are necessary for the purpose of the system.
Interface ContactMessage	An interface that represents the inquiry (or response) message of the user (or data controller) with regard to a privacy issue, and has the responsibilities to record the sender, receiver, and content of the message.
Class ConcreteContactMessage	A concrete class that implements the interface ContactMessage. Other responsibilities related to the message can be added.
Interface ContactMethod	An interface that has responsibilities to send a message from a sender to a receiver.
Class ConcreteContactMethod	A concrete class that implements the interface ContactMethod.
Collaboration	
1. When the data subject requests to contact the data controller, e.g. for confirmation information, access to personal data, or challenge 1.1 The system controller creates a ConcreteContactMessage object to set the sender, receiver and message content (e.g. for confirmation information, access to personal data, or challenge) into the object. 1.2 The system controller creates a ConcreteContactMethod object to send the ConcreteContactMessage to the receiver. 2. If the request is accepted 2.1 The system controller creates a ConcreteDataManager object to do something with the personal data as requested. 2.2 The system controller creates a ConcreteContactMessage object to set the sender, receiver and message content (e.g. for response) into the object.	

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

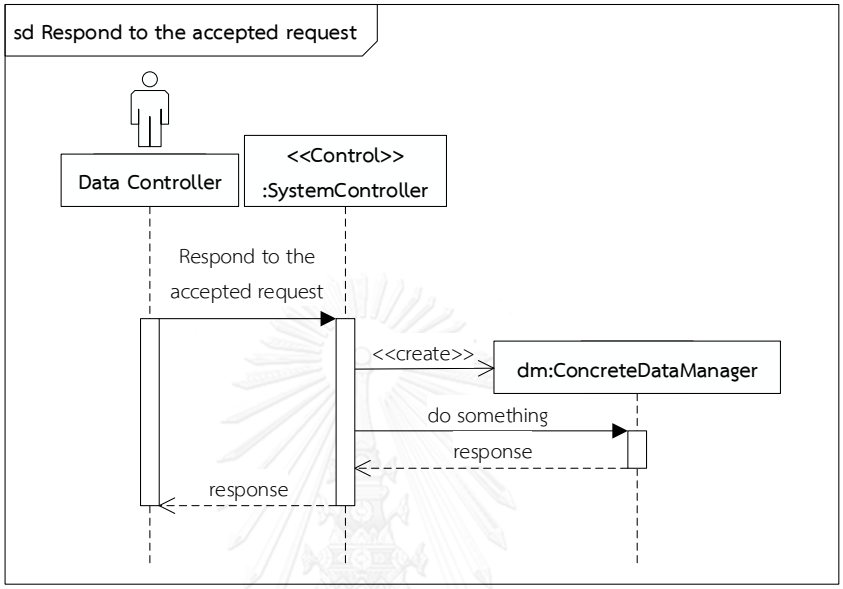
Collaboration
2.3 The system controller creates a ConcreteContactMethod object to send the ConcreteContactMessage to the receiver. 3. If the request is denied 3.1 The system controller creates a ConcreteContactMessage object to set the sender, receiver and message content (e.g. for reason) into the object. 3.2 The system controller creates a ConcreteContactMethod object to send the ConcreteContactMessage to the receiver. 3.3 If the data subject wants to challenge 3.3.1 The system controller creates a ConcreteContactMessage object to set the sender, receiver and message content (e.g. for challenge request) into the object. 3.3.2 The system controller creates a ConcreteContactMethod object to send the ConcreteContactMessage to the receiver. 4. Repeat steps 2 and 3 until the data controller accepts the request or the data subject accepts the reason or wants to escalate challenge.

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Collaboration
Sequence Diagram (When a data subject makes an inquiry and is responded by the data controller.)



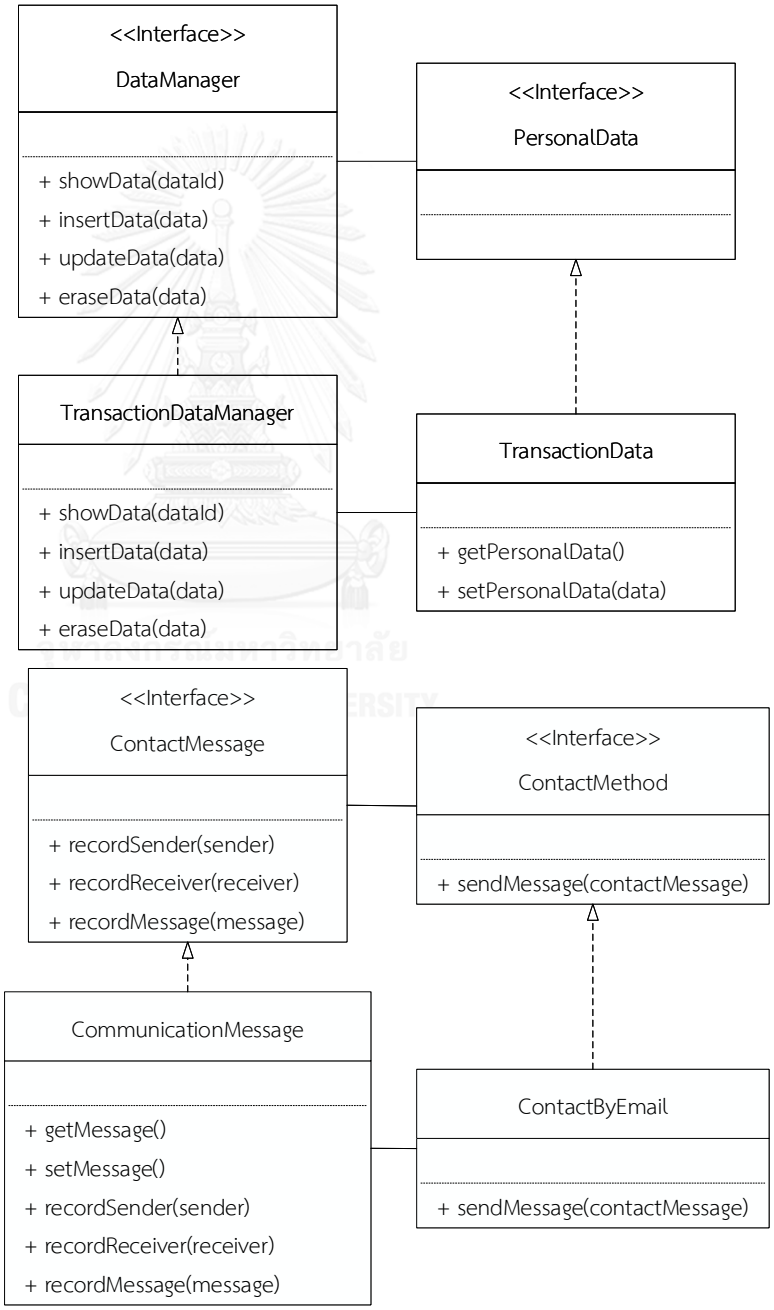
ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Collaboration
Sequence Diagram (When a data subject makes an inquiry and is responded by the data controller. -continued) 

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Consequences	Using this pattern, the system of the data controller can assure that the data subject can exercise the right to personal data as recommended by OECD.
Implementation	Issues regarding the implementation of the Individual Participation pattern are as follows. 1. Several categories of personal data may be collected by the system. Some are not readily accessible to the data subject whereas some are available for access. For example, details of access and activities of the data subject may not be readily accessible and, if needed, a request has to be made. The data controller is then obliged to confirm whether or not such data are available or can be prepared and provided. For other personal data that the system collects or makes available for access, e.g. user profile data at registration, the system effectively confirms the data subject in the handling of such data through the privacy policy and request for consent. The data controller should refer to the Openness, Collection Limitation, and Use Limitation patterns. 2. The system should allow the data subject to participate in his/her personal data with regard to different access types (e.g. create, destroy, view modify). The data controller should refer to the Security Safeguards pattern. 3. For personal data that are not readily accessible but can be requested and those that are made available for the data subject to access via the system, the data subject should be able to access within a reasonable time, at a reasonable cost (if that is the case), in a reasonable manner, and in a form that is readily intelligible. This implies that, access or request to access personal data should be incorporated as a normal operation into the system and does not involve any legal or complex process. Nevertheless, the time it takes to access the personal data once a request is made may depend on the varied length of the data processing activity. Also, providing access to personal data electronically via the system would be a reasonable manner and it could help reduce potential costs, e.g. when compared to a trip to the data controller for an access to a copy of personal data. The data controller should present the data in a usable and easy to understand manner.

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Implementation	4. Even though the system allows the data subject, in the first step, to challenge the adverse decisions to the data controller, the dispute may not be settled. The data subject may decide to escalate to subsequent challenges in courts or administrative bodies.
Sample Design and Code	An example below shows the case of a data subject who inquires about the detail of a payment transaction. 

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Sample Design and Code	When a data subject communicates to the data controller about the payment transaction, the system uses the class CommunicationMessage that implements the interface ContactMessage to record the sender, receiver, and content of the message as shown in the sample code below. `interface ContactMessage {` `  public void recordSender(String sender);` `  public void recordReceiver(String receiver);` `  public void recordMessage(String message);` `}` `public class CommunicationMessage implements ContactMessage{` `  private String sender;` `  private String receiver;` `  private String message;` `  public String getSender() {` `  return sender;` `  }` `  public String getReceiver() {` `  return receiver;` `  }` `  public String getMessage() {` `  return message;` `  }` `  @Override` `  public void recordSender(String sender) {` `  // TODO Add your code to record who sends message.` `  this.sender = sender;` `  }` `  @Override` `  public void recordReceiver(String receiver) {` `  // TODO Add your code to record who receives message.` `  this.receiver = receiver;` `  }` `  @Override` `  public void recordMessage(String message) {` `  // TODO Add your code to record the message.` `  this.message = message;` `  }` `}` The class ContactByEmail that implements the interface ContactMethod is implemented to send a message from a sender to a receiver. `interface ContactMethod {` `  public void sendMessage(CommunicationMessage msg);` `}`

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Sample Design and Code	

```
public class ContactByEmail implements ContactMethod{
    @Override
    public void sendMessage(CommunicationMessage msg) {
        // TODO Add your code to send a message from a sender to a receiver.
        String to = msg.getReceiver();
        String from = msg.getSender();
        String host = "localhost";

        //Get the session object
        Properties properties = System.getProperties();
        properties.setProperty("mail.smtp.host", host);
        Session session = Session.getDefaultInstance(properties);

        //Compose the message
        try{
                MimeMessage message = new MimeMessage(session);
                message.setFrom(new InternetAddress(from));
                message.addRecipient
                    (Message.RecipientType.TO,new InternetAddress(to));
                message.setSubject("Ping");
                message.setText(msg.getMessage());

                //Send message
                Transport.send(message);
                System.out.println("message sent successfully....");
        }catch (MessagingException mex) {mex.printStackTrace();}
    }
}
```

The data controller accepts the request from the data subject. The system uses the class TransactionDataManager that implements the interface DataManager to show personal data as shown in the sample code below.

```
public interface DataManager {
    public PersonalData showData(String dataId);
    public void insertData(PaymentTransactionData trans);
    public void updateData(PaymentTransactionData trans);
    public void eraseData(PaymentTransactionData trans);
}
public class TransactionDataManager implements DataManager{

    protected Date now = new Date();
    protected SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    protected Timestamp timestamp = new java.sql.Timestamp(now.getTime());

    @Override
    public PersonalData showData(String dataId) {
        // TODO Add your code to get data from data store.
        PaymentTransactionData data = new PaymentTransactionData();
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try{
            String query = "SELECT * FROM transaction WHERE id = ?" ;
            PreparedStatement stmt = connection.prepareStatement(query)
            stmt.setInt(1, Integer.parseInt(dataId));
            ResultSet result = stmt.executeQuery();
            while(result.next()){
                data.setId(result.getString("id"));
                data.setTransactionType(result.getString("type"));
                data.setTransactionDetail(result.getString("detail"));
                data.setTransactionResult(result.getString("result"));
                data.setTransactionDate(result.getDate("date"));
            }
            stmt.close();
        } catch (SQLException e) {}
        return data;
    }
```

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Sample Design and Code	

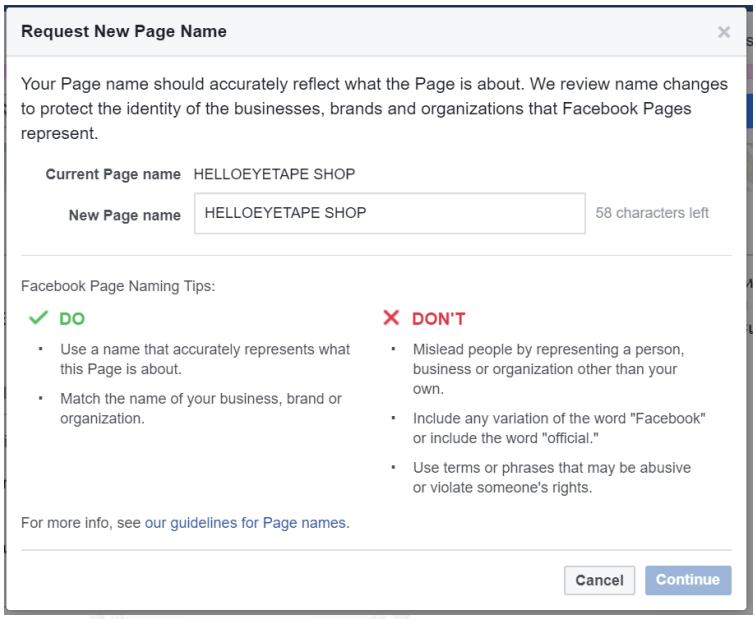
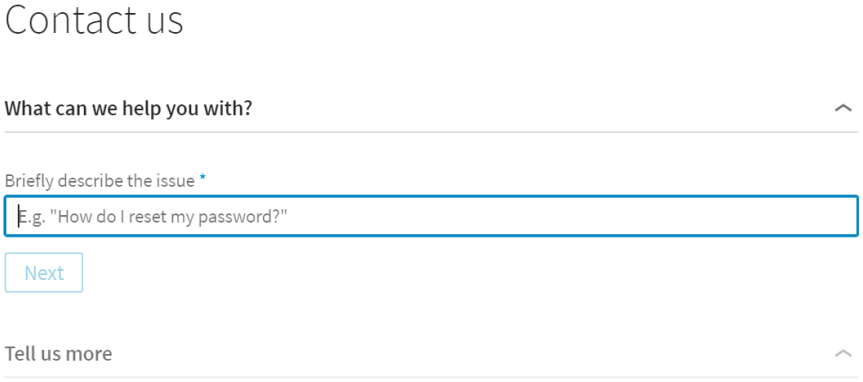
```
    @Override
    public void insertData(PaymentTransactionData trans) {
        //TODO Add your code to insert data into data store.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            String insert = "INSERT INTO transaction "
                    + "(type, detail, result,date ) "
                    + "VALUES (?, ?, ?, ?)" ;
            PreparedStatement stmt = connection.prepareStatement(insert) ;
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, trans.getTransactionType());
            stmt.setString(parameterIndex++, trans.getTransactionDetail());
            stmt.setString(parameterIndex++, trans.getTransactionResult());
            stmt.setTimestamp(parameterIndex++, timestamp);

            stmt.execute();
            stmt.close();

        } catch (SQLException e) {e.printStackTrace();}
    }
    @Override
    public void updateData(PaymentTransactionData trans) {
        // TODO Add your code to update data.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try {
            StringBuffer upd = new StringBuffer("UPDATE transaction SET ");
            if (trans.getTransactionType() != null) {
                upd.append("typr = '"+trans.getTransactionType()+"', ");
            }
            if (trans.getTransactionDetail() != null) {
                upd.append("email = '"+trans.getTransactionDetail()+"', ");
            }
            if (trans.getTransactionResult() != null) {
                upd.append("tel = '"+trans.getTransactionResult()+"', ");
            }

            upd.append("updatedate = '"+sdf.format(now)+"' ");
            upd.append("WHERE id = '"+trans.getId()+"' ");
            System.out.println(upd.toString()) ;
            PreparedStatement stmt = connection.prepareStatement(upd.toString()
            stmt.execute();
            stmt.close();
        } catch (SQLException e) {e.printStackTrace();}
    }
    @Override
    public void eraseData(PaymentTransactionData trans) {
    //TODO Add your code to erase data.
        DatabaseConnection dbConnection = new DatabaseConnection();
        Connection connection = dbConnection.getDBConnection();
        try{
            String insert = "DELETE FROM transaction "
                    + "where id = ? " ;

            PreparedStatement stmt = connection.prepareStatement(insert)
            int parameterIndex = 1;

            stmt.setString(parameterIndex++, trans.getId());

            stmt.execute();
            stmt.close();
        }catch(SQLException e) {}
    }
}
```

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

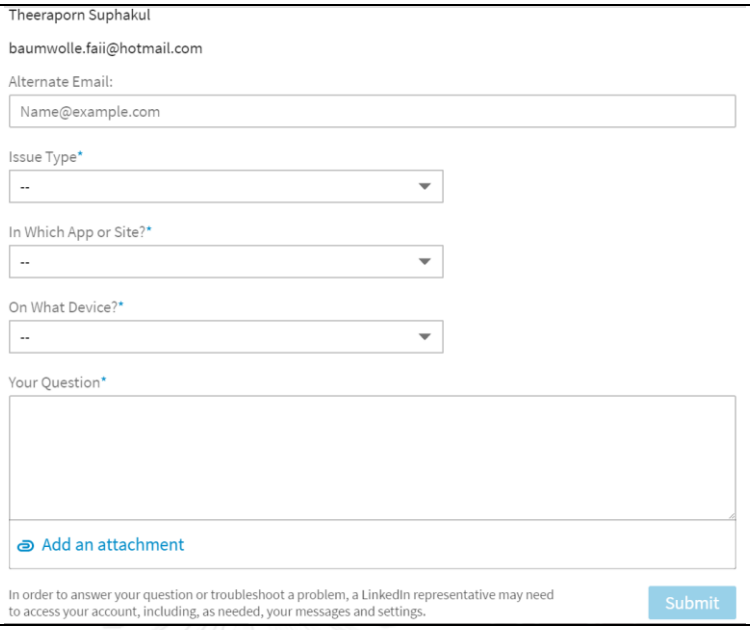
Sample Design and Code	The class PaymentTransactionData that implements the interface PersonalData is implemented to have only the data attributes that are necessary for the purpose of payment transaction as shown in the sample code below.

```
public interface PersonalData {
}
public class PaymentTransactionData implements PersonalData{
    private String id;
    private String transactionType;
    private String transactionResult;
    private String transactionDetail;
    private Date transactionDate;
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getTransactionType() {
        return transactionType;
    }
    public void setTransactionType(String transactionType) {
        this.transactionType = transactionType;
    }
    public String getTransactionResult() {
        return transactionResult;
    }
    public void setTransactionResult(String transactionResult) {
        this.transactionResult = transactionResult;
    }
    public String getTransactionDetail() {
        return transactionDetail;
    }
    public void setTransactionDetail(String transactionDetail) {
        this.transactionDetail = transactionDetail;
    }
    public Date getTransactionDate() {
        return transactionDate;
    }
    public void setTransactionDate(Date transactionDate) {
        this.transactionDate = transactionDate;
    }
```

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

Known use	Examples of existing systems that take the Individual Participation principle into account include: • Facebook Page (Website): When a data subject wants to access the page name to have it changed, the data subject has to make a request to the data controller via the system.  • LinkedIn (Website): The website provides a way to contact the data controller via the system to inquire about issues related to personal data. 

ตารางที่ ง.7 แบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

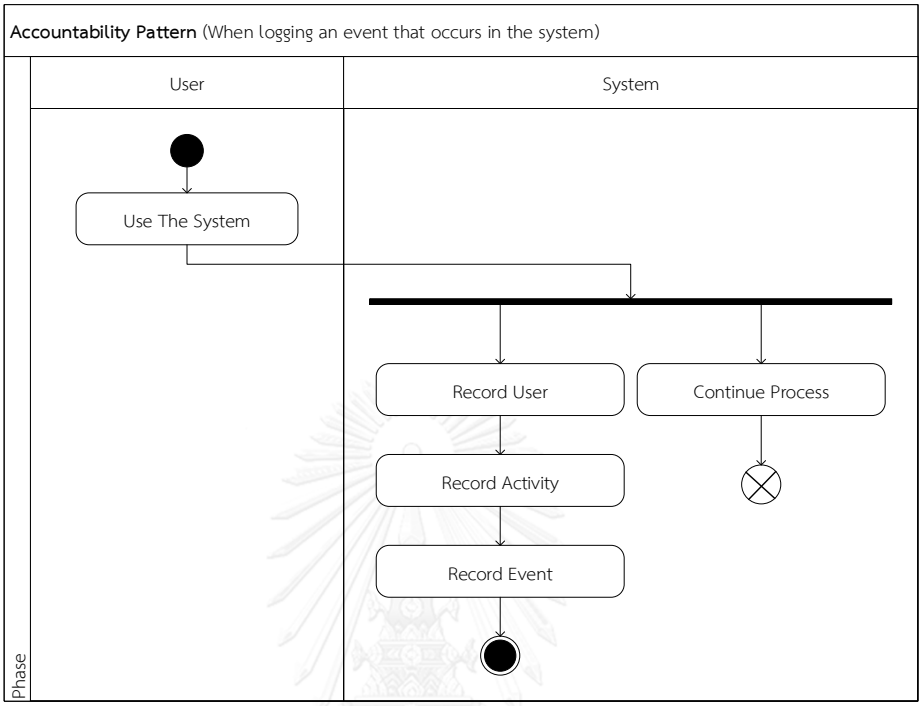
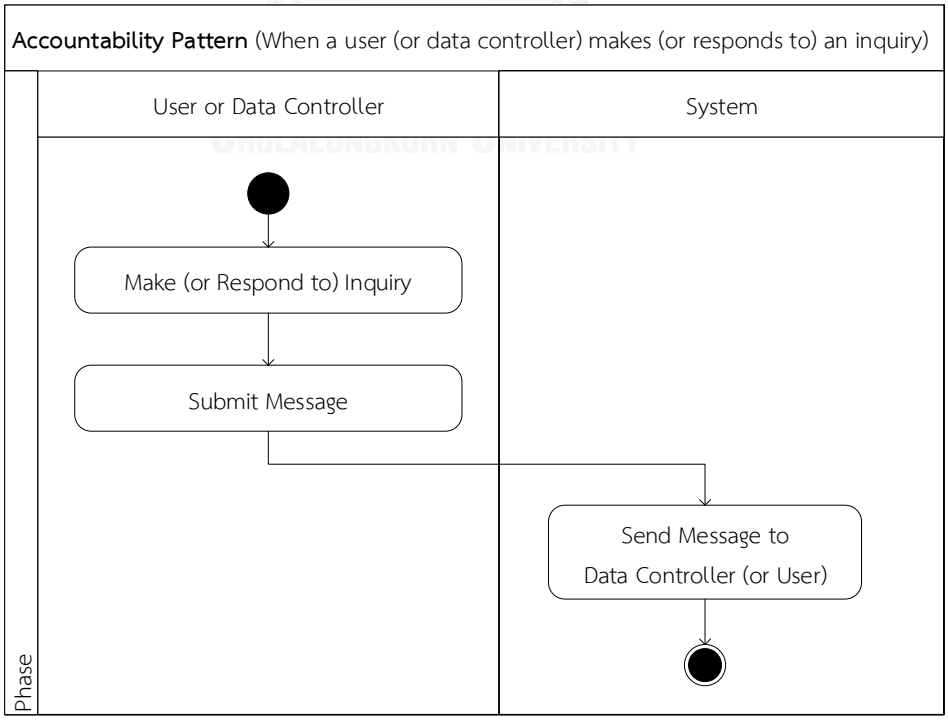
	
Related Patterns	Openness The Openness pattern can be used with the Individual Participation pattern to ensure that the data subject can participate in the collection and use of his/her personal data by being informed of the privacy policy and what personal data are collected for what purposes. Collection Limitation The Collection Limitation pattern can be used with the Individual Participation pattern to ensure that the data subject can participate in giving or refusing consent to his/her personal data collection. Use Limitation The Use Limitation pattern can be used with the Individual Participation pattern to ensure that the data subject can participate in the collection and use of his/her personal data by being informed of the use of his/her personal data which deviates from the specified purposes. Security Safeguards The Security Safeguards can be used with the Individual Participation pattern to ensure that the data subject can participate in the handling of his/her personal data with regard to access types.

ง. 8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ

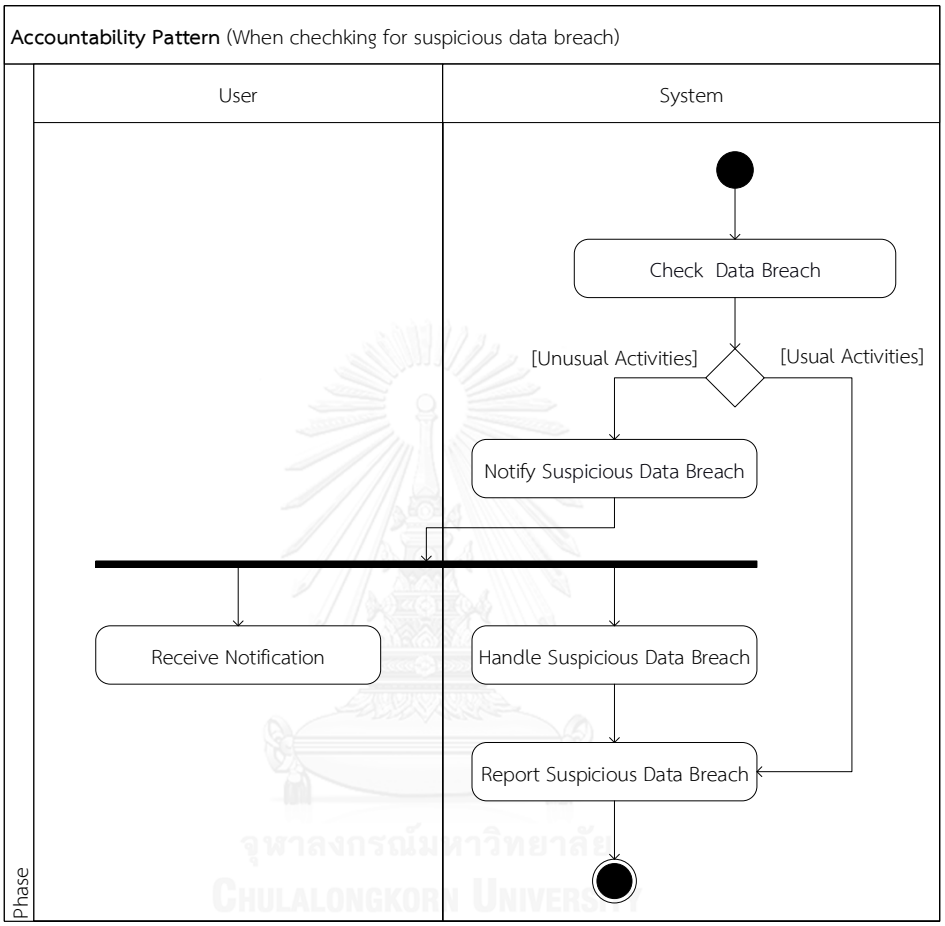
Pattern Name	**Accountability Pattern**
Privacy Principle	Accountability Principle
Intent	This pattern assures that the data controller and the data processor who carries out the processing of personal data on behalf of the data controller (if any) comply with the privacy principles of the OECD, and provide necessary measures for accountability.
Motivation	The data subject entrusts the data controller with the handling of his/her personal data. The data controller is therefore held accountable for demonstrating to the data subject, as well as the privacy enforcement authorities, its compliance with privacy principles. To do so, the data controller and the data processor have to ensure that necessary privacy-enabling mechanisms are taken into account in order to protect, detect, and handle privacy issues.
Applicability	Use the Accountability pattern throughout the life cycle of personal data within the system. That is, use all other patterns in order to comply with the OECD principles. In addition, use this pattern to • Record and view a log of events which enables monitoring of the system. • Assess data breach using the log of events or history of access. • Handle and report on data breach incidents and notify relevant parties. • Enable inquiry of privacy issues.

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Behavior and Structure
Activity Diagram (When logging an event that occurs in the system) 
Activity Diagram (When a user (or data controller) makes (or responds to) an inquiry) 

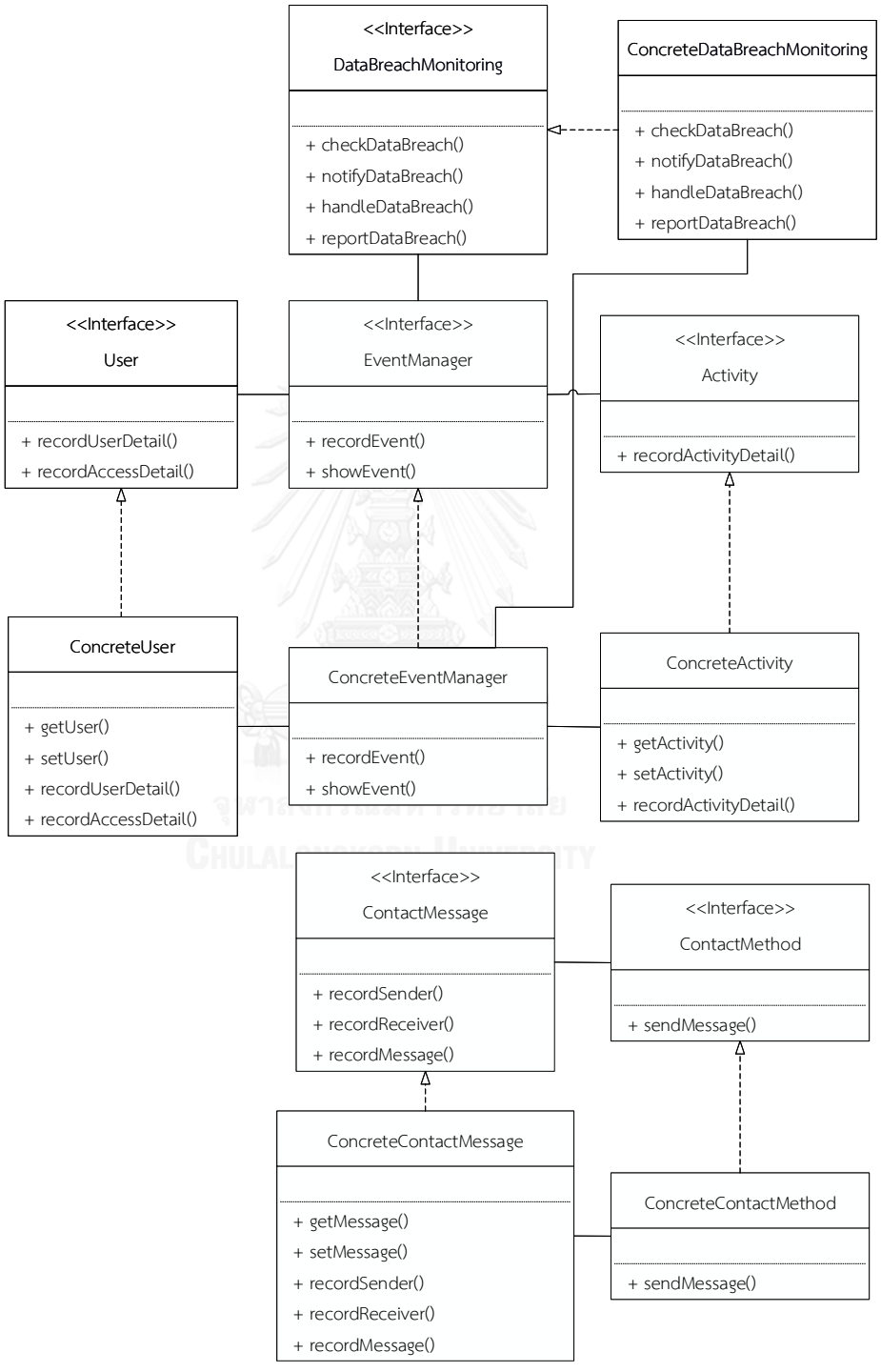
ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Behavior and Structure
Activity Diagram (When checking for suspicious data breach)



ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Behavior and Structure
Class Diagram



ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

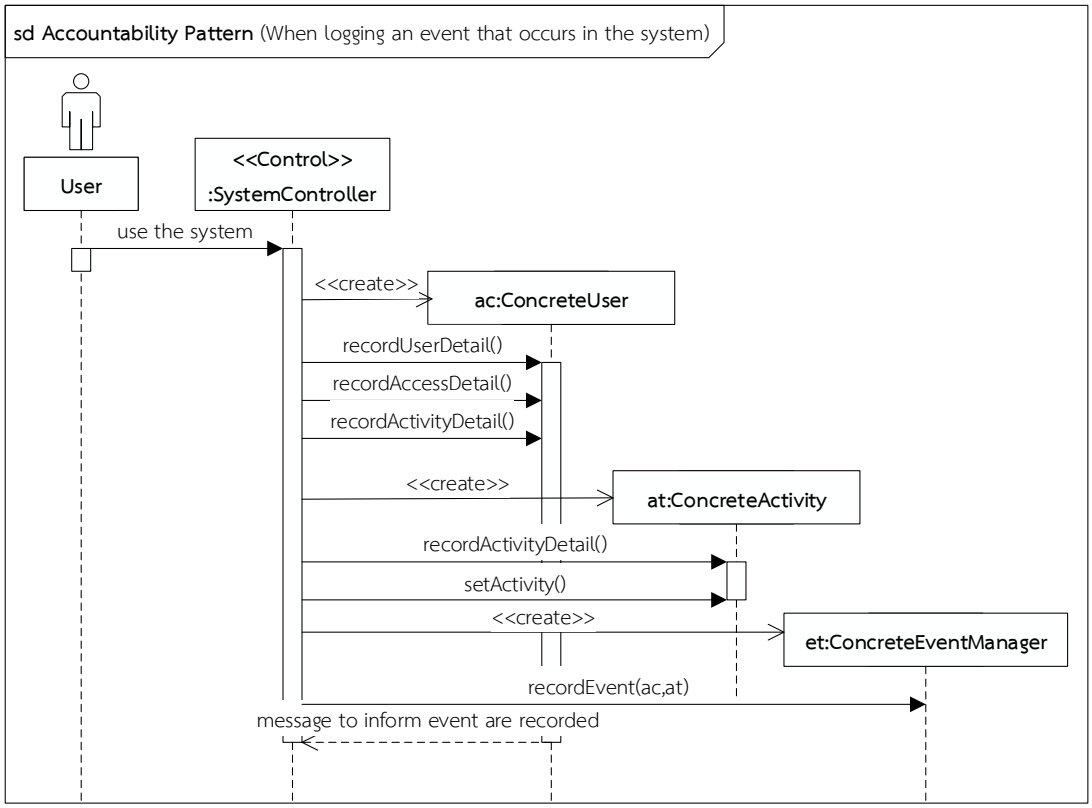
Participants	
Interface Name/Class Name	**Detail**
Interface User	An interface that represents the user who accesses the system (and may not be the data subject) and has responsibilities to record user detail and access detail.
Class ConcreteUser	A concrete class that implements the interface User with the data attributes to identify who accesses the system as well as access detail.
Interface Activity	An interface that represents the user activity within the system and has responsibilities to record activity detail, e.g. what personal data are accessed and access type.
Class ConcreteActivity	A concrete class that implements the Interface Activity with the data attributes to identify the user activity.
Interface EventManager	An interface that has responsibilities to record and show events that occur in the system.
Class ConcreteEventManager	A concrete class that implements the Interface EventManager.
Interface DataBreachMonitoring	An interface that has responsibilities to monitor access to personal data, notify relevant parties (e.g. data subject and data controller) about unauthorized or suspicious access behavior, further handle the breach, and report data breach.
Class ConcreteDataBreachMonitoring	A concrete class that implements the interface DataBreachMonitoring.
Interface ContactMessage	An interface that represents the inquiry (or response) message of the user (or data controller) with regard to a privacy issue, and has the responsibilities to record the sender, receiver, and content of the message.
Class ConcreteContactMessage	A concrete class that implements the interface ContactMessage. Other responsibilities related to the message can be added.
Interface ContactMethod	An interface that has responsibilities to send a message from a sender to a receiver.
Class ConcreteContactMethod	A concrete class that implements the interface ContactMethod.

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Collaboration
1. When logging an event that occurs in the system 1.1 When a user uses the system, the system controller creates a ConcreteUser object to record the user detail (e.g. username, role) and access detail (e.g. location, browser, operating system, access date). 1.2 The system controller creates a ConcreteActivity object to record the user activity detail (e.g. access type and accessed data). 1.3 The system controller creates a ConcreteEventManager object to record the ConcreteUser and ConcreteActivity objects as an event in a data store. 2. When a user (or data controller) makes (or responds to) an inquiry 2.1 The system controller creates a ConcreteContactMessage object to set the sender, receiver and message content into the object. 2.2 The system controller creates a ConcreteContactMethod object to send the ConcreteContactMessage to the receiver. 3. When checking for suspicious data breach 3.1 The system controller creates a ConcreteDataBreachMonitoring object to monitor access to personal data 3.2 The system controller calls the ConcreteDataBreachMonitoring to check for suspicious access behavior. 3.3 The ConcreteDataBreachMonitoring object calls ConcreteEventManager to view a log of events that occur in the system. 3.4 If access behavior is unusual, the ConcreteDataBreachMonitoring object notifies the data subject about suspicious access behavior, and handles data breach. 3.5 The ConcreteDataBreachMonitoring object generates a data breach checking report.

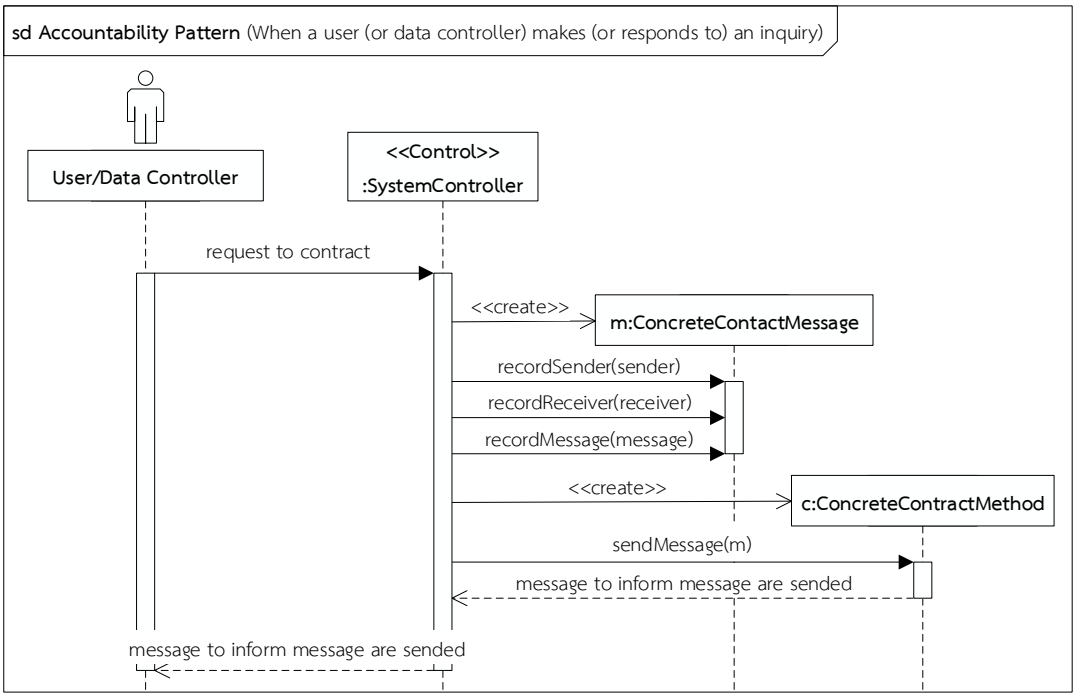
ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Collaboration
Sequence Diagram (When logging an event that occurs in the system)



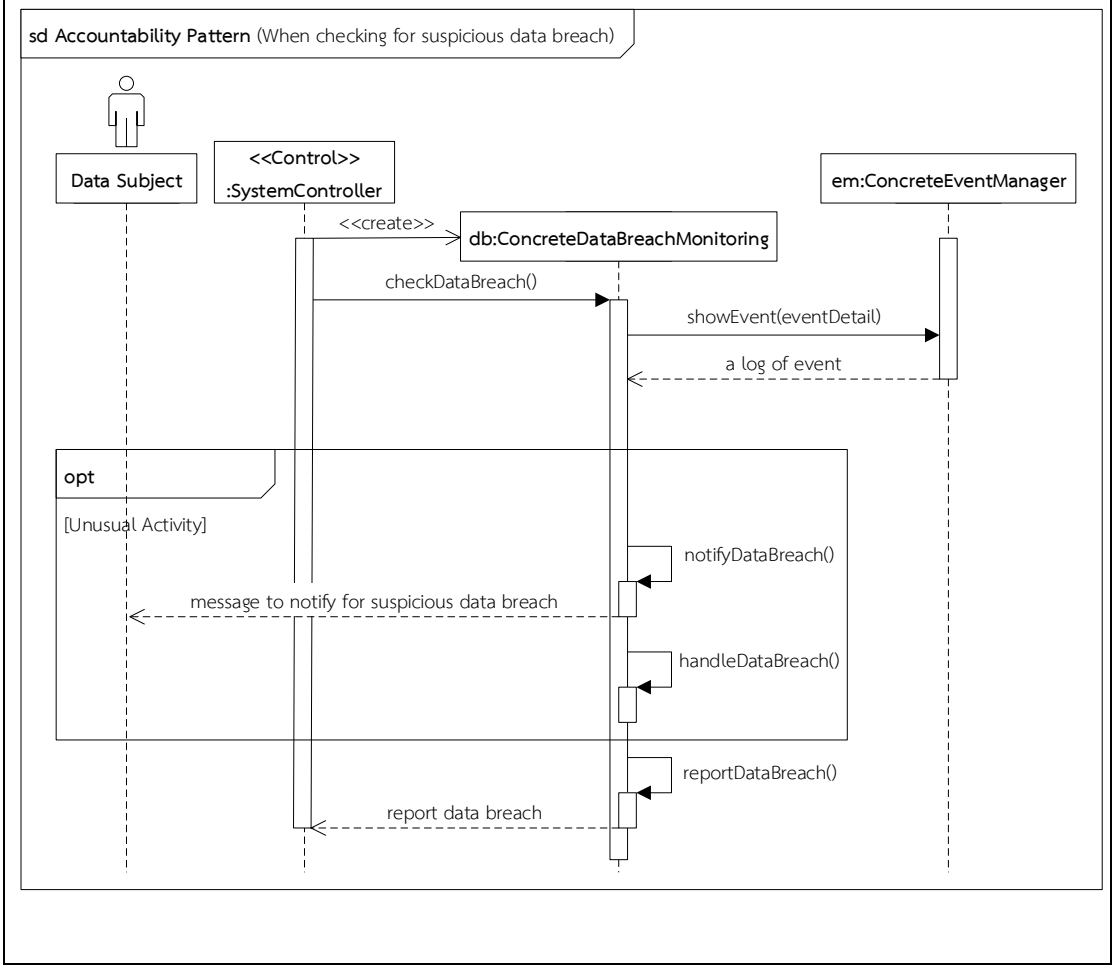
ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Collaboration
Sequence Diagram (When a user (or data controller) makes (or responds to) an inquiry)



ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

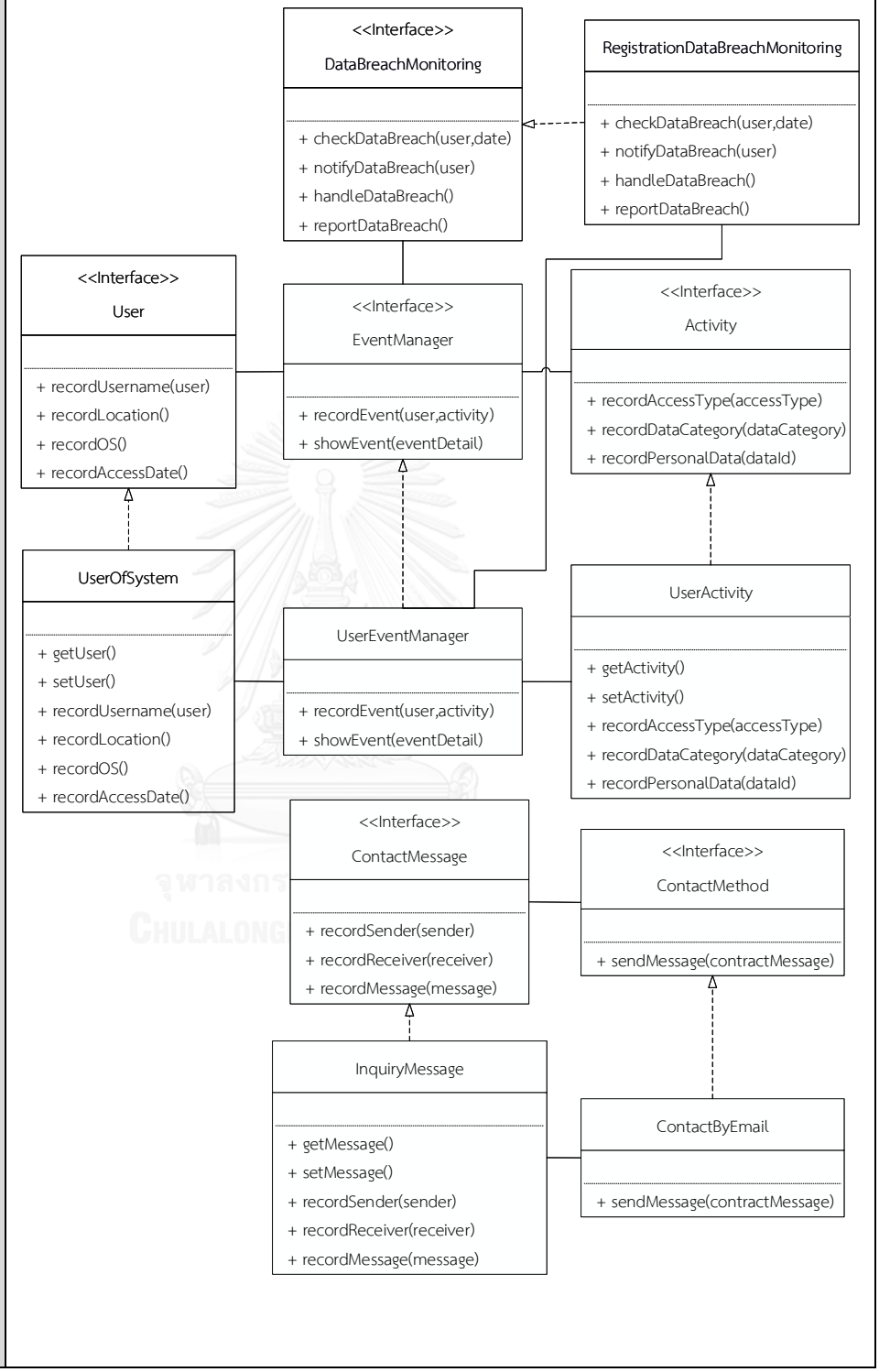
Collaboration
Sequence Diagram (When checking for suspicious data breach)



ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Consequences	1. Using this pattern, the system of the data controller can comply with the principles as recommended by OECD. The data controller and data processor are prepared to demonstrate accountability to the data subject. 2. There are costs in terms of time and space associated with managing a log of events. Extra storage space is required for logging user activities and periodic assessment of data breach on the log may consume time and may require an algorithm to identify suspicious behavior or loopholes in the system.
Implementation	Issues regarding the implementation of the Accountability pattern are as follows. 1. All privacy-enabling mechanisms recommended by OECD privacy principles should be tailored to the operations of the data controller itself and of the data processor who performs data processing on behalf of the data controller. All mechanisms should be recorded in the privacy management program of the data controller and data processor. 2. The data controller may choose to notify privacy enforcement authorities only when there is a significant data breach. The data controller should report to the authorized bodies an intrusion by an attacker that accesses all personal data, rather than failed attempts to log in to a user account. However, the latter might be a threat to an individual and should be reported the relevant data subject.
Sample Design and Code	An example below shows the case of a system that records details of events of access and activities of a user within the system. The system allows the user to make inquiries and can respond to the user. The system operation is constantly monitored and regularly checked for data breach incidents. In the presence of such incidents, relevant parties are notified. The system generates a report as evidence of data breach monitoring and checking.

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Sample Design and Code	

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Sample Design and Code	When a user uses the system, the system uses the class UserOfSystem that implements the interface User to record user detail (e.g. username) and access detail (e.g. location, os, and access date) as shown in the sample code below.

```
interface User {
    public void recordUsername(String username);
    public void recordLocation();
    public void recordOS();
    public void recordAccessDate();
}
public class UserOfSystem implements User{
    private String username;
    private String location;
    private String os;
    private Date accessDate;


    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getLocation() {
        return location;
    }
    public void setLocation(String location) {
        this.location = location;
    }
    public String getOs() {
        return os;
    }
    public void setOs(String os) {
        this.os = os;
    }
    public Date getAccessDate() {
        return accessDate;
    }
    public void setAccessDate(Date accessDate) {
        this.accessDate = accessDate;
    }
    @Override
    public void recordUsername(String username) {
        /* TODO Add your code to record user
         */
        this.username = username;
    }
    @Override
    public void recordLocation() {
        /* TODO Add your code to record location that user live
         * when access personal data.
         */
        Locale currentLocale = Locale.getDefault();
        this.location = currentLocale.getDisplayLanguage();
    }
    @Override
    public void recordOS() {
        /* TODO Add your code to recode operating system that user
         * uses to access personal data.
         */
        String operate = System.getProperty("os.name");
        this.os = operate;
    }
    @Override
    public void recordAccessDate() {
        /* TODO Add your code to recode date that user access
         * personal data.
         */
        Date datenow = new Date();
        this.accessDate = datenow;
    }
}
```

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Sample Design and Code	The system uses the class UserActivity that implements the interface Activity to record activity detail (e.g. access type, data category, id of personal data) `public interface Activity {` `  public void recordAccessType(String accessType);` `  public void recordDataCategory(String dataCategory);` `  public void recordPersonalData(String dataId);` `}` `public class UserActivity implements Activity{` `  private String accessType;` `  private String dataCategory;` `  private String dataId;` `  public String getAccessType() {` `  return accessType;` `  }` `  public void setAccessType(String accessType) {` `  this.accessType = accessType;` `  }` `  public String getDataCategory() {` `  return dataCategory;` `  }` `  public void setDataCategory(String dataCategory) {` `  this.dataCategory = dataCategory;` `  }` `  public String getDataId() {` `  return dataId;` `  }` `  public void setDataId(String dataId) {` `  this.dataId = dataId;` `  }` `  @Override` `  public void recordAccessType(String accessType) {` `  // TODO Add your code to record activity` `  this.accessType = accessType;` `  }` `  @Override` `  public void recordDataCategory(String dataCategory) {` `  // TODO Add your code to record data category` `  this.dataCategory = dataCategory;` `  }` `  @Override` `  public void recordPersonalData(String dataId) {` `  // TODO  Add your code to record personal data` `  this.dataId = dataId;` `  }` `}`

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Sample Design and Code	The system uses the class UserEventManager that implements the interface EventManager to record and show events that occur in the system. An example records events into a log file as shown in the sample code below.

```
public interface EventManager {
    public void recordEvent(UserOfSystem user,UserActivity activity);
    public void showEvent(String activity,String dataCategory,Date date);
}
public class UserEventManager implements EventManager{
    private final static Logger logger =
            Logger.getLogger(UserEventManager.class.getName());
    private SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
    @Override
    public void recordEvent(UserOfSystem user,UserActivity activity) {
        // TODO Add your code to record event into a data store.
        FileHandler fh;
        try {
            String accessType = activity.getAccessType();
            String dataCategory = activity.getDataCategory();
            fh = new FileHandler("D:/log/"+accessType+"-"+dataCategory+"-"
                    +sdf.format(user.getAccessDate())+".log",true);
            logger.addHandler(fh);

            JSONObject juser = new JSONObject();
            juser.put("username", user.getUsername());
            juser.put("location", user.getLocation());
            juser.put("os", user.getOs());
            juser.put("date", user.getAccessDate());
            String accessorXml = XML.toString(juser);
            logger.info(accessorXml);

            JSONObject jactivity = new JSONObject();
            jactivity.put("accesstype", activity.getAccessType());
            jactivity.put("datacategory", activity.getDataCategory());
            jactivity.put("dataId", activity.getDataId());
            String activityXml = XML.toString(jactivity);
            logger.info(activityXml);


        } catch (SecurityException e) { e.printStackTrace();}
          catch (IOException e) { e.printStackTrace();}
          catch (JSONException e) { e.printStackTrace();
        }
    }
    @Override
    public void showEvent(String accessType,String dataCategory,Date date) {
        // TODO Add your code to show a log of events.
        String fileName = "D:/log/"+accessType+"-"+dataCategory+"-"
                        +sdf.format(date)+".log";
        System.out.println(fileName);
        BufferedReader reader;
        try {
            reader = new BufferedReader(new FileReader(fileName));
            String line;
            while((line = reader.readLine()) != null){
                System.out.println(line);
            }
        } catch (FileNotFoundException e) { e.printStackTrace();}
          catch (IOException e) { e.printStackTrace();}
    }
}
```

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Sample Design and Code	When a user makes an inquiry, the class InquiryMessage that implements the interface ContactMessage is implemented to record the sender, receiver, and content of the message as shown in the sample code below. `interface ContactMessage {` `  public void recordSender(String sender);` `  public void recordReceiver(String receiver);` `  public void recordMessage(String message);` `}` `public class InquiryMessage implements ContactMessage{` `  private String sender;` `  private String receiver;` `  private String message;` `  public String getSender() {` `  return sender;` `  }` `  public String getReceiver() {` `  return receiver;` `  }` `  public String getMessage() {` `  return message;` `  }` `  @Override` `  public void recordSender(String sender) {` `  // TODO Add your code to record who send message.` `  this.sender = sender;` `  }` `  @Override` `  public void recordReceiver(String receiver) {` `  // TODO Add your code to record who receive message.` `  this.receiver = receiver;` `  }` `  @Override` `  public void recordMessage(String message) {` `  // TODO Add your code to record the message.` `  this.message = message;` `  }` `}` The class ContactByEmail that implements the interface ContactMethod is implemented to send a message from a sender to a receiver. `interface ContactMethod {` `  public void sendMessage(InquiryMessage msg);` `}`

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

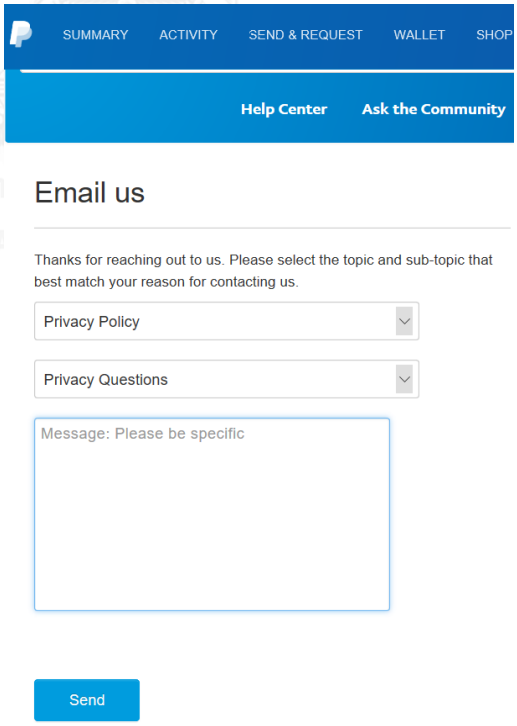
Sample Design and Code	

```
public class ContactByEmail implements ContactMethod{
    @Override
    public void sendMessage(InquiryMessage msg) {
        // TODO Add your code to send a message from a sender to a receiver.
        String to = msg.getReceiver();
        String from = msg.getSender();
        String host = "localhost";

        //Get the session object
        Properties properties = System.getProperties();
        properties.setProperty("mail.smtp.host", host);
        Session session = Session.getDefaultInstance(properties);

        //Compose the message
        try{
                MimeMessage message = new MimeMessage(session);
                message.setFrom(new InternetAddress(from));
                message.addRecipient
                    (Message.RecipientType.TO,new InternetAddress(to));
                message.setSubject("Ping");
                message.setText(msg.getMessage());

                //Send message
                Transport.send(message);
                System.out.println("message sent successfully....");
        }catch (MessagingException mex) {mex.printStackTrace();}
    }
}
```

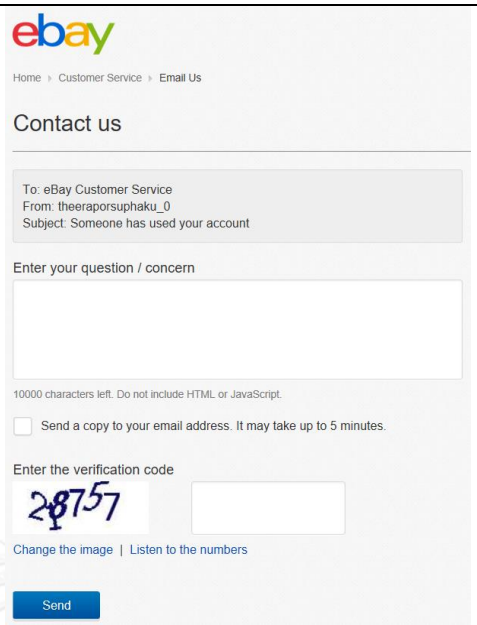
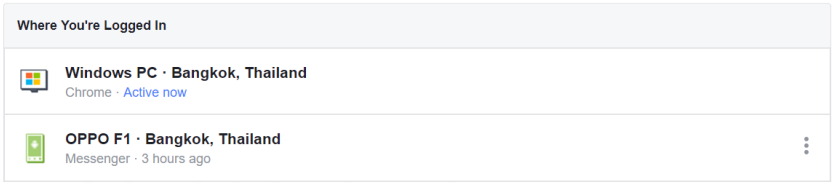
In the case that the system needs to check for suspicious data breach, the class RegistrationDataBreachMonitoring that implements the interface DataBreachMonitoring as shown in the sample code below would be called to check access behavior for data breach and notify suspicious access behavior (e.g. logging in from a different OS platform and location in this example). Furthermore, the system could handle data breach and generate a data breach report.

```
interface DataBreachMonitoring extends DataManager{
    public boolean checkDataBreach(UserOfSystem user,String date);
    public void notifyDataBreach(UserOfSystem user);
    public void HandleDataBreach();
    public void ReportDataBreach();
}
public class RegistrationDataBreachMonitoring
            extends RegistrationDataManager implements DataBreachMonitoring{

    @Override
    public boolean checkDataBreach(UserOfSystem user,String date) {
        boolean checker = false;
        /* TODO Add your code to check for suspicious access behavior.
         * if this access are suspicious the system should notify user
         */
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        Date dateCheck;
        try {
            dateCheck = sdf.parse(date);
            UserEventManager uEvent = new UserEventManager();
            uEvent.showEvent("Update", "Registration", dateCheck);
            //Add your code to check for suspicious access behavior

        } catch (ParseException e) {e.printStackTrace();}

        if(!checker)
            notifyDataBreach(user);
        return checker;
    }
```

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Sample Design and Code	<pre>@Override public void notifyDataBreach(UserOfSystem user) { /* TODO Add your code to notify data subject that access are * suspiciousi.e. notify email, send SMS and so on */ SimpleDateFormat ft = new SimpleDateFormat ("E yyyy.MM.dd 'at' hh:mm:ss a"); System.out.println("Send notice to email!!"); System.out.println("We noticed you logged into Account using"); System.out.println(user.getOs()); System.out.println("from "+user.getLocation()); System.out.println("date "+ft.format(user.getAccessDate())); } @Override public void HandleDataBreach() { // TODO Add your code to handle data breach } @Override public void ReportDataBreach() { // TODO Add your code to report data breach } }</pre>
Known use	Examples of existing systems that take the Accountability principle into account include: ● PayPal (Website): When a user wants to contact the data controller, an inquiry can be made via the system.  ● eBay (Website): When a user wants to contact the data controller, an inquiry can be made via the system.

ตารางที่ ง.8 แบบรูปการออกแบบสำหรับหลักความรับผิดชอบ (ต่อ)

Known use	 ● Facebook (Website): When a user logs in from an IP address, browser, OS, and device that differ from those in the previous logging-in, the web site tracks this unusual behavior and allows the user to block any access to the web site from such suspicious setting. 
Related Patterns	All privacy patterns The Accountability pattern will be effective when all other privacy patterns are applied to the system. The data controller can demonstrate its accountability to the data subject by having all privacy-enabling mechanisms, present in the other patterns, to protect the data subject's personal data.

ภาคผนวก จ.
ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบ

ในส่วนนี้เป็นการตรวจสอบความครบถ้วนของแบบรูปการออกแบบที่นำเสนอโดยใช้รายการตรวจสอบ (Checklist) เพื่อตรวจสอบว่าแบบรูปการออกแบบที่นำเสนอนั้นมีความครบถ้วนตามหลักการความเป็นส่วนตัวหรือไม่ โดยผลของการตรวจสอบมีดังนี้

1. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล แสดงดังตารางที่ จ.1
2. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักคุณภาพของข้อมูล แสดงดังตารางที่ จ.2
3. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ แสดงดังตารางที่ จ.3
4. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล แสดงดังตารางที่ จ.4
5. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล แสดงดังตารางที่ จ.5
6. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการเปิดเผย แสดงดังตารางที่ จ.6
7. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล แสดงดังตารางที่ จ.7
8. ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ แสดงดังตารางที่ จ.8

ตารางที่ จ.1 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล

รายละเอียดย่อย		คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
		ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Collection Limitation Principle						
1. There should be limits to the collection of personal data.						
1.1	The collected personal data should be of high quality so that they can be used to derive other sufficiently high quality information without having to collect such information directly.	Implementation	Implementation	Implement	Implementation	Implementation
1.2	The data collection should be restricted to the minimum necessary to fulfil the specified purpose.	Implementation, Behavior and Structure	Behavior and Structure	Behavior and Structure, Sample Design and Code	Implementation, Sample Design and Code	Behavior and Structure, Implementation
1.3	Personal data that are considered sensitive according to traditions and attitudes in each country should be of concern in the data collection.	Implementation	Implementation	Implementation	Implementation	Implementation
1.4	There should be limits to data collection activities of the data controller such that the data should not be collected by the means that is unknown or deceiving to the data subject.	Implementation, Behavior and Structure	Behavior and Structure	Implementation, Behavior and Structure, Sample Design and Code	Implementation, Sample Design and Code	Behavior and Structure

ตารางที่ จ.1 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการเก็บรวบรวมข้อมูล (ต่อ)

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Collection Limitation Principle					
1.5 Civil rights of the data subject should be of concern in the data collection.	Implementation	Implementation	Implementation	Implementation	Implementation
2. The data should be obtained by lawful and fair means and, where appropriate, with the knowledge of the data subject.	Intent, Behavior and Structure	Behavior and Structure	Behavior and Structure, Sample Design and Code	Intent, Sample Design and Code	Behavior and Structure
3. The data should be obtained by lawful and fair means and, where appropriate, with the consent of the data subject.	Intent, Behavior and Structure	Behavior and Structure	Behavior and Structure, Sample Design and Code	Intent, Sample Design and Code	Behavior and Structure
4. The Collection Limitation Principle does not exclude the possibility of a data subject being represented by another party, for instance in the case of minors, mentally disabled person, etc.	Implementation, Behavior and Structure	Behavior and Structure	Implementation, Behavior and Structure, Sample Design and Code	Implementation, Sample Design and Code	Behavior and Structure, Implementation

ตารางที่ จ.2 ผลการตรวจสอบความครบถ้วนของแบบรูปการณ์ออกแบบสำหรับหลักคุณภาพของข้อมูล

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Data Quality Principle					
1. Personal data should be relevant to the purposes for which they are to be used.	Motivation	Motivation	Implementation	Related Patterns	Related Patterns
2. Personal data, to the extent necessary for their purposes, should be accurate.	Motivation, Behavior and Structure	Motivation, Behavior and Structure, Collaboration	Behavior and Structure	Behavior and Structure	Behavior and Structure
3. Personal data, to the extent necessary for their purposes, should be complete.	Behavior and Structure	Collaboration	Behavior and Structure	Behavior and Structure	Behavior and Structure
4. Personal data, to the extent necessary for their purposes, should be kept up-to-date.	Behavior and Structure	Implementation, Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure

ตารางที่ จ.3 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Purpose Specification Principle					
1. The purposes for which personal data are collected should be specified not later than at the time of data collection.	Behavior and Structure, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Implementation	Intent, Motivation, Behavior and Structure
2. The subsequent use of personal data should be limited to the fulfilment of the purposes specified prior to or at the time of collection.	Implementation	Implementation	Implementation	Implementation	Motivation
3. The subsequent use of personal data may be for purposes other than the original purposes, but such purposes should be specified as change.	Implementation	Implementation	Implementation	Implementation	Implementation
4. Later changes of purposes should be specified.	Implementation	Implementation	Implementation	Implementation	Intent, Implementation
5. On each occasion of change of purpose, the new purposes should not be incompatible with the original purposes.	Implementation	Implementation	Implementation	Implementation	ไม่พบ
6. Specification of purpose should be made by public declarations or information to the data subject.	Implementation	Implementation	Implementation	Related Patterns	Implementation

ตารางที่ จ.3 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อกำหนดวัตถุประสงค์ (ต่อ)

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Purpose Specification Principle					
7. When data no longer serve a purpose, and if it is practicable, it may be necessary to have them destroyed (erased) or given an anonymous form.	Behavior and Structure, Implementation	Behavior and Structure, Collaboration, Implementation,	Behavior and Structure	Behavior and Structure	Behavior and Structure
8. The Purpose Specification Principle is closely associated with the Data Quality Principle and the Use Limitation Principle.	Related Patterns	Related Patterns	Implement	Related Patterns	Related Patterns

ตารางที่ จ.4 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักข้อจำกัดในการใช้ข้อมูล

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Use Limitation Principle					
1. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except with the consent of the data subject or his representative.	Behavior and Structure, Implementation, Collaboration	Behavior and Structure, Implementation	Implementation	Behavior and Structure, Implementation, Related Patterns	Behavior and Structure
2. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification Principle, except by the authority of law or licences granted by supervisory bodies.	Behavior and Structure	Behavior and Structure Implementation	Implementation	Behavior and Structure, Implementation, Related Patterns	Behavior and Structure

ตารางที่ จ.5 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Security Safeguards Principle					
1. Personal data should be protected by reasonable security safeguards against unauthorized disclosure of data, based on informational measures such as enciphering.	Behavior and Structure, Consequences	Behavior and Structure, Consequences, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
2. Personal data should be protected by reasonable security safeguards against unauthorized access, destruction, use, and modification of data, based on organizational measures such as authority levels with regard to access to data.	Behavior and Structure	Behavior and Structure, Consequences, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
3. Personal data should be protected by reasonable security safeguards against loss or destruction and modification of data, both accidentally and deliberately, where data integrity would be affected.	Collaboration	Behavior and Structure, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
4. Personal data should be protected by reasonable security safeguards against loss of access by which data availability would be affected.	Behavior and Structure,	Behavior and Structure, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
5. Personal data should be protected by reasonable security safeguards against threats of data breach, based on informational measures such as monitoring of unusual activities and responses to them.	Behavior and Structure,	Behavior and Structure, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration

ตารางที่ จ.5 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการรักษาความมั่นคงของข้อมูล (ต่อ)

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Security Safeguards Principle					
6. Personal data should be protected by reasonable security safeguards based on physical measures (such as locked doors and identification cards) and proper organizational procedure (such as employees' security training, up-to-date security safeguards).	Consequences	Behavior and Structure, Implementation	Implementation	ไม่พบ	ไม่พบ

ตารางที่ จ.6 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการเปิดเผย

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Openness Principle					
1. There should be a general policy of openness about developments, practices and policies with respect to personal data.	Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure
2. Means should be readily available of establishing the existence and nature of personal data.	Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure
3. Means should be readily available of establishing the main purposes of the use of personal data.	Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure	Behavior and Structure
4. Means should be readily available of establishing the identity and usual residence of the data controller.	Behavior and Structure	Behavior and Structure	class diagram	Behavior and Structure	Behavior and Structure
5. “Readily available” implies that data subjects should be able to obtain information without unreasonable effort as to time, advance knowledge, travelling, and so forth, and without unreasonable cost.	Intent, Behavior and Structure	Behavior and Structure	Intent, Behavior and Structure	Implementation	Behavior and Structure
6. Regular information from the data controller regarding the privacy policy should be published.	Behavior and Structure	Behavior and Structure	Behavior and Structure	Consequences	Behavior and Structure
7. The Openness Principle may be viewed as a prerequisite for the Individual Participation Principle.	Related patterns	Behavior and Structure	Behavior and Structure	Implementation	Implementation

ตารางที่ จ.7 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Individual Participation Principle					
1. A data subject can obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him/her (or his/her personal data).	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration	Behavior and Structure	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration
2. A data subject should have the right to access to his/her personal data within a reasonable time when a request to access is made.	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
3. A data subject should have the right to access to his/her personal data at a reasonable cost.	Behavior and Structure, Collaboration, Implementation, Related Patterns	Behavior and Structure, Collaboration, Implementation	Behavior and Structure	ไม่พบ	Behavior and Structure, Collaboration
4. A data subject should have the right to access to his/her personal data in a reasonable manner.	Behavior and Structure, Collaboration, Implementation, Related Patterns	Behavior and Structure, Collaboration, Implementation	Behavior and Structure	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration

ตารางที่ จ.7 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักการมีส่วนร่วมของบุคคล (ต่อ)

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Individual Participation Principle					
5. A data subject should have the right to access to his/her personal data in a form that is readily intelligible.	Behavior and Structure, Collaboration, Implementation, Related Patterns	Behavior and Structure, Collaboration, Implementation	Behavior and Structure	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration
6. A data subject should have the right to reasons for, and challenge, adverse decisions when requests for confirmation or access are denied.	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
7. A data subject should have the right to challenge his/her personal data.	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
8. A data controller should erase, rectify, complete, or amend the data subject's personal data if the challenge is successful.	Behavior and Structure, Collaboration	Behavior and Structure, Collaboration	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration

ตารางที่ จ.8 ผลการตรวจสอบความครบถ้วนของแบบรูปการออกแบบสำหรับหลักความรับผิดชอบ

รายละเอียดย่อย	คนที่ 1	คนที่ 2	คนที่ 3	คนที่ 4	คนที่ 5
	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ	ปรากฏในส่วนใดของแบบรูปการออกแบบ
Accountability Principle					
1. A data controller and a data processor who carries out the processing of personal data on behalf of the data controller (if any) are held accountable for complying with measures which give effect to all other principles.	Intent	Implementation	Implementation	Behavior and Structure	ไม่พบ
2. A data controller should have in place a privacy management program that is tailored to its operations and integrated into its governance structure.	Implementation	Implementation	Implementation	ไม่พบ	ไม่พบ
3. A data controller should provide ongoing monitoring, periodic assessment of data breach, and response to incidents.	Sample Design and Code	Behavior and Structure, Collaboration	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
4. There should be a method for the data subject to inquire about issues related to privacy of personal data.	Collaboration	Behavior and Structure	Collaboration	Behavior and Structure	Behavior and Structure, Collaboration
5. A data controller should provide evidence of its privacy management program to the privacy enforcement authorities.	Implementation	Behavior and Structure, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration
6. A data controller should provide notice to the privacy enforcement authorities and data subject in the event of significant data breach.	Implementation	Behavior and Structure, Implementation	Behavior and Structure	Behavior and Structure	Behavior and Structure, Collaboration

ประวัติผู้เขียนวิทยานิพนธ์

นางสาวธีราพร ศุภกุล เกิดวันที่ 19 ธันวาคม 2534 สำเร็จการศึกษาระดับปริญญาตรี หลักสูตรวิทยาศาสตรบัณฑิต (วท.บ.) สาขาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2556 ประสบการณ์ทำงานพนักงานองค์กรเอกชน บริษัท ทรู อินฟอเมชั่น เทคโนโลยี ตำแหน่งนักวิเคราะห์ระบบ เป็นเวลา 2 ปี และเข้าศึกษาต่อระดับปริญญามหาบัณฑิต ปีการศึกษา 2558 หลักสูตรวิทยาศาสตรมหาบัณฑิต (วท.ม.) สาขาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย