บทที่ 2

การพิจารณาองค์ประกอบพื้นฐาน

2.1 วัตถุประสงค์

บทนี้จะกล่าวถึงแนวทางในการออกแบบ File Editor โดยจะพิจารณาประเด็นที่สำคัญ คือ

ก. เครื่องมือในการพัฒนาโปรแกรม (development tool)

ข. โครงสร้างแฟ้มข้อมูล (file structure)

ค. การกำหนดชนิดข้อมูล (data type)

ง. การใช้รหัสอักขระ (character code)

จ. การติดต่อกับผู้ใช้ได้สะดวก (intelligent user interface)

การเลือกเครื่องมือในการพัฒนาโปรแกรม จะกล่าวถึงภาษาต่าง ๆ ที่ใช้ และข้อคำนึงถึงในการพิจารณาเลือกใช้ภาษานั้นเพื่อการพัฒนาโปรแกรม

การพิจารณาโครงสร้างแฟ้มข้อมูล จะกล่าวถึงโครงสร้างข้อมูลแบบต่างๆ อย่างย่อๆ เพื่อพิจารณาเปรียบเทียบการกำหนดโครงสร้างของข้อมูลในแฟ้มข้อมูลตามเป้าหมายที่วางไว้

การกำหนดชนิดข้อมูล จะกล่าวถึงการใช้ชนิดของข้อมูลต่าง ๆ และวิธีการที่ใช้ในการกำหนดระเบียนของแฟ้มข้อมูล

การใช้รหัสอักขระ จะกล่าวถึงรหัสอักขระที่มีใช้กันอยู่ รวมทั้งการกำหนดรหัสอักขระขึ้นเองเพื่อวัตถุประสงค์บางอย่าง และการพิจารณาความสามารถในการเปลี่ยนแปลงรหัส (portability consideration)

การติดต่อกับผู้ใช้ จะกล่าวถึงการติดต่อกับผู้ใช้โดยผ่านทางรูปแบบของฟอร์ม และลักษณะการใช้งานของฟอร์ม

2.2 เครื่องมือในการพัฒนาโปรแกรม

เครื่องมือในการพัฒนาโปรแกรมนี้ หมายถึง ภาษาที่ใช้ในการเขียนโปรแกรม (programming language) เพื่อให้ได้โปรแกรมตามที่ต้องการ

ในการเลือกใช้ภาษาเพื่อการพัฒนาโปรแกรมนั้นมีสิ่งที่ต้องคำนึงถึงคือ

ก. เครื่องคอมพิวเตอร์ที่ใช้ และภาษาที่มีใช้สำหรับเครื่องคอมพิวเตอร์นั้น

ข. ความสะดวกในการพัฒนาโปรแกรม

ค. ความเร็วในการทดสอบโปรแกรม

จากการที่เลือกทำงานบนระบบเครื่องไมโครคอมพิวเตอร์ตระกูล ไอบีเอ็ม พีซี (XT/AT or Compatible) ทำให้เลือกใช้ภาษาที่มีบนเครื่องนี้คือ BASIC, Pascal หรือ C (สำหรับ FORTRAN หรือ COBOL นั้นเป็นภาษารุ่นเก่าที่มีขนาดใหญ่เมื่อใช้สร้างโปรแกรมแล้วจะมีขนาดใหญ่ไม่เหมาะกับระบบไมโครคอมพิวเตอร์) สำหรับภาษาที่กล่าวถึงนี้ การเลือกใช้ก็ขึ้นอยู่กับสภาวะแวดล้อมการทำงาน และการใช้งานของภาษานั้น

ภาษา BASIC เป็นภาษาที่เรียนรู้ง่าย แต่โปรแกรมที่ได้ไม่มีโครงสร้าง (unstructured) การที่ไม่มีโครงสร้างนี้ทำให้การเขียนโปรแกรมแยกส่วนเป็นเครื่องมือทางซอฟต์แวร์ทำได้ไม่สะดวก และยากแก่การทดสอบโปรแกรมที่ได้

สำหรับภาษา C และ Pascal นั้น เป็นภาษาที่ดีทั้งคู่ มีโครงสร้างของภาษาคล้ายกัน เหมาะสำหรับผู้พัฒนาซอฟต์แวร์ เนื่องจากให้ประสิทธิภาพในการทำงานดีกล่าวคือ

- ทำให้โปรแกรมที่ได้ดูง่าย และสามารถแก้ไขได้ง่าย
- สามารถแบ่งโปรแกรมเป็นโมดูล
- มีความสามารถในการจัดการโครงสร้างข้อมูลที่ดี

ในขณะที่เริ่มพัฒนาโปรแกรมนั้น (พ.ศ. 2529) คอมไพเลอร์ (compiler) ที่มีสำหรับภาษาปาสคาลนั้นคือ เทอร์โบปาสคาล (Turbo Pascal Version 3.0) [17] ซึ่งเป็นคอมไพเลอร์ที่ให้ความสะดวก รวดเร็วในการใช้งานมากกว่าคอมไพเลอร์ภาษา C ที่มีในขณะนั้น และเทอร์โบปาสคาลนี้ ทำงานใต้เอ็มเอส-ดอส (MS-DOS, Microsoft Disk Operating System) บนเครื่องไมโครคอมพิวเตอร์ ไอบีเอ็ม-พีซี ได้ โดยมีความสามารถในการทำงานที่ต้องการได้อย่างเพียงพอ จึงทำให้เลือกใช้เทอร์โบปาสคาลในการพัฒนาโปรแกรมต่อไป

หัวข้อต่อไปจะกล่าวถึงโครงสร้างข้อมูล แต่การเลือกใช้จะคำนึงถึงโครงสร้างข้อมูลชนิดเดียวกับการจัดการแฟ้มข้อมูลโดยภาษาปาสคาล เนื่องจากจุดประสงค์ของการพัฒนาโปรแกรมนั้น เพื่อใช้เป็นเครื่องมือฐานข้อมูลอย่างหนึ่ง ที่มีการจัดการแฟ้มข้อมูลต่างๆ ตามแบบที่กำหนดให้ และการนำแฟ้มข้อมูลที่ได้ไปทำงานใด ๆ ต่อนั้น ก็จะต้องให้ง่าย หรือสะดวกในการใช้

นอกจากนี้ ในการจัดข้อมูลในระเบียนยังมีส่วนที่พิจารณาอีก คือ การเก็บข้อมูลบางอย่าง เช่น ในการเก็บค่าตัวเลข สามารถเก็บได้หลายวิธี คือ เก็บเป็นเลขฐานสอง เก็บเป็น BCD (Binary Coded Decimal) หรือ เก็บเป็นตัวอักขระ ซึ่งแต่ละวิธีก็มีข้อดีข้อเสียต่างกัน การเก็บเป็นเลขฐานสอง มีข้อเสียที่จำนวนตัวเลขนัยสำคัญมีตายตัว (เลขจำนวนเต็มใช้เนื้อที่ 2 ไบต์ เลขจำนวนจริงใช้เนื้อที่ 6 ไบต์) แต่มีข้อดีที่สามารถคำนวณทางคณิตศาสตร์ได้ง่ายและรวดเร็ว ส่วนอีก 2 วิธี สามารถเก็บตัวเลขมีจำนวนตัวเลขนัยสำคัญได้อิสระ แต่การคำนวณยุ่งยาก ดังนั้นเมื่อชั่งน้ำหนัก ข้อดีข้อเสียข้างต้นแล้ว ได้เลือกการเก็บตัวเลขเป็นเลขฐานสอง

ต่อไปจะกล่าวถึงรายละเอียดในการจัดการแฟ้มข้อมูล โดยพิจารณาที่ระดับการจัดข้อมูลภายใน โดยแยกกันพิจารณาตามที่ได้กล่าวมาแล้ว

2.3 โครงสร้างแฟ้มข้อมูล

ในที่นี้โครงสร้างแฟ้มข้อมูล หมายถึง ลักษณะของการเก็บระเบียนลงในจานแม่เหล็ก (disk) ถึงแม้ว่าในจานแม่เหล็กจะมีการแบ่งส่วนเนื้อที่หน่วยความจำออกเป็นส่วน ๆ ที่มีขนาดเท่ากัน (block หรือ sector) ซึ่งจัดการโดยระบบปฏิบัติการ (operating system) แต่ขนาดของระเบียน ก็สามารถเปลี่ยนแปลงได้ตามแต่จะจัดให้โดยโปรแกรมจัดการ (application program)

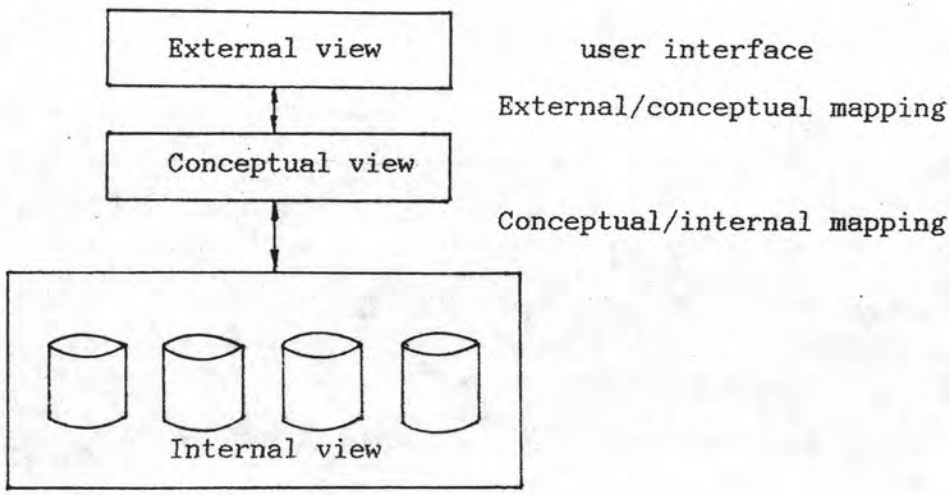
เราจะแยกพิจารณาโครงสร้างแฟ้มข้อมูล ดังนี้

ก. ขนาดระเบียนคงที่ (fixed-length records)

ข. ขนาดระเบียนไม่คงที่ (variable-length records)

ค. รวมระเบียนเป็นกลุ่มขนาดคงที่ (blocks)

การจัดโครงสร้างข้อมูลเหล่านี้ เป็นการจัดโครงสร้างที่เขียนลงยังจานแม่เหล็ก ที่เรียกว่าระดับการจัดการโครงสร้างภายใน (internal level) แต่ในการใช้งานทางผู้ใช้งานจะไม่เห็นการจัดการเหล่านี้ โดยผู้ใช้เพียงแต่กำหนดลักษณะของโครงสร้างข้อมูลเท่านั้น (conceptual level) ว่าประกอบด้วยข้อมูลย่อยอะไรบ้าง ดังรูปที่ 2.1 ส่วน external view เป็นส่วนที่ผู้ใช้มองเห็นข้อมูลของระเบียนอาจอยู่ในรูปของฟอร์ม โดยผ่านทาง conceptual view ที่กำหนดโดยลักษณะโครงสร้างข้อมูล เพื่อจัดการกับข้อมูลจริงที่เขียนในจานแม่เหล็ก (internal view)



รูปที่ 2.1 การจัดลำดับการมองแฟ้มข้อมูล

2.3.1 <u>ระเบียนขนาดคงที่</u>

ระเบียนขนาดคงที่ประกอบด้วยข้อมูลย่อยในแต่ละระเบียนเหมือนกัน ทำให้มีขนาดเท่ากันในทุกระเบียน เช่น ระเบียนของผลสอบของนักเรียน ซึ่งกำหนดไว้ดังนี้

```
type student   =    record
                        id      : integer;
                        name    : string[20];
                        score   : real;
                        grade   : char;
                    end
```

ในที่นี้ integer ใช้เนื้อที่ 2 ไบต์ string ใช้เนื้อที่ 21 ไบต์ (ไบต์แรกเก็บความยาวของข้อมูลจริง) real ใช้เนื้อที่ 6 ไบต์ และ char ใช้เนื้อที่ 1 ไบต์ ดังนั้นระเบียนนี้จะใช้เนื้อที่ 30 ไบต์ ในแฟ้มข้อมูลของผลสอบนักเรียนจะประกอบด้วย ระเบียนที่ 1 มีขนาด 30 ไบต์ ระเบียนที่ 2 มีขนาด 30 ไบต์ เป็นเช่นนี้ไปจนหมดแฟ้มข้อมูล

ระเบียนที่ 1	1	นายสมชาย	90.5	A
ระเบียนที่ 2	2	นายแดง	81.5	B
	⋮	⋮	⋮	⋮

รูปที่ 2.2 แสดงระเบียนขนาดคงที่

แฟ้มข้อมูลที่ใช้โครงสร้างข้อมูล ซึ่งมีขนาดระเบียนคงที่นี้ ในเทอร์โบปาสคาล เรียกว่า random access file ซึ่งหมายถึง แฟ้มข้อมูลที่สามารถอ่าน หรือเขียนระเบียน (หมายเลข) ที่เท่าไหร่ก็ได้

การทำงานกับแฟ้มข้อมูลที่มีระเบียนขนาดคงที่ อาจมีปัญหาที่ต้องพิจารณา คือ การลบระเบียน และการแทรกระเบียน (สำหรับการเพิ่มระเบียนต่อท้ายไม่เป็นปัญหา)

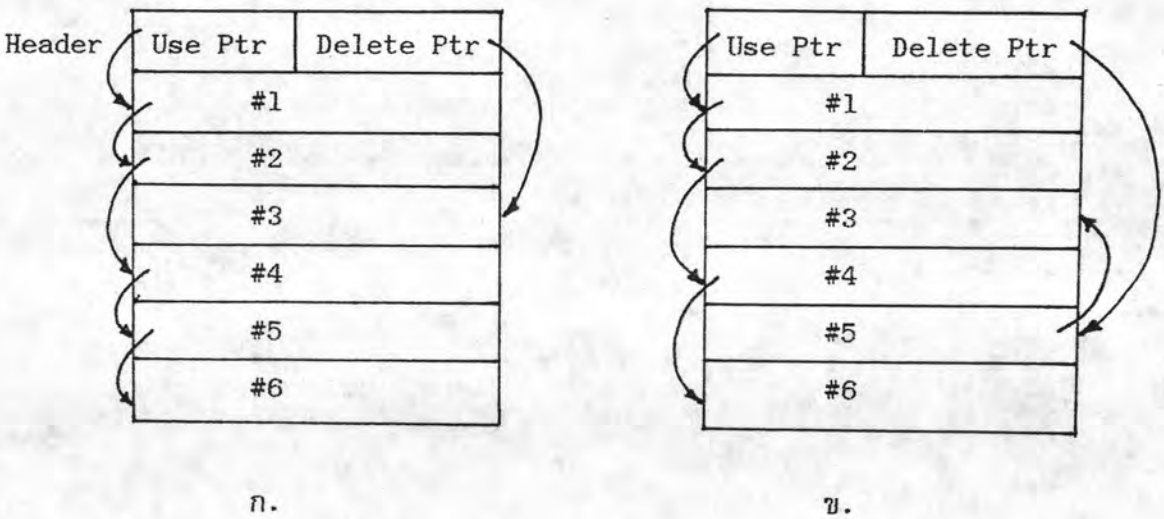
ปัญหาในการลบระเบียนออกจากแฟ้มข้อมูลนั้น ที่จะต้องพิจารณาเพิ่มเติมอีก คือ ลบออกไปอย่างถาวร ทิ้งเอาไว้เฉยๆ นำเอาไปใช้ใหม่ หรือตัดต่อเชื่อมโยง

การลบระเบียนออกไปอย่างถาวร โปรแกรมจัดการจะต้องอ่านระเบียนถัดจากระเบียนที่จะถูกลบ แล้วเขียนทับระเบียนที่ถูกลบ อ่านระเบียนต่อๆ ไป แล้วเขียนทับระเบียนก่อนหน้าจนหมดแฟ้มข้อมูล ซึ่งวิธีการเช่นนี้จะเสียเวลามาก

การลบระเบียนโดยทิ้งเอาไว้เฉยๆ โปรแกรมจัดการจะเขียนสถานะถูกลบทิ้งประจำระเบียนนั้นเอาไว้ (หรือใช้รหัสพิเศษเขียนไว้ที่ตำแหน่งแรกของระเบียน) เมื่อจะใช้งานแฟ้มข้อมูล ก็ตรวจดูสถานะถูกลบทิ้ง (หรือรหัสพิเศษ) ถ้ามีอยู่ ก็ข้ามระเบียนนั้นไป การทิ้งเอาไว้เฉยๆ มีข้อเสีย คือ เปลืองเนื้อที่

ในกรณีที่เขียนสถานะถูกลบทิ้งเอาไว้ สามารถแก้ไขสถานะนี้ หรือสามารถเขียนระเบียนใหม่ทับลงไป แต่การค้นหาระเบียนที่จะนำมาใช้ใหม่อาจเสียเวลามาก กล่าวคือ ต้องไปอ่านแฟ้มข้อมูลตั้งแต่ระเบียนแรก ตรวจสอบสถานะถูกลบ ถ้าไม่ใช่ อ่านระเบียนต่อไปจนกว่าจะพบ หรือจนกว่าจะหมดแฟ้ม เมื่ออ่านพบก็สามารถนำระเบียนมาใช้ใหม่ได้

การลบระเบียน อาจใช้วิธีตัดต่อเชื่อมโยงระเบียนใหม่ โดยใช้ตัวชี้ (หรือดัชนี) เป็นตัวเชื่อมโยง เช่น ใช้ระเบียนแรกเก็บตัวชี้ 2 ตัว ตัวหนึ่งชี้ไปยังระเบียนที่ใช้งานระเบียนแรก อีกตัวหนึ่งชี้ไปยังระเบียนแรกที่ถูกลบทิ้ง ระเบียนที่ใช้งานมีตัวชี้ชี้ไปยังระเบียนใช้งานถัดไป เมื่อมีการลบทิ้งระเบียนใช้งานใดๆ ก็ตัดต่อตัวชี้ของระเบียนใช้งานใหม่ แล้วเปลี่ยนตัวชี้ระเบียนที่ถูกลบทิ้งตัวแรกมายังตัวที่เพิ่งถูกลบ และตัวที่ถูกลบใหม่ชี้ไปยังตัวที่ถูกลบก่อนหน้า ตัวอย่างเช่น ลักษณะระเบียนใช้งาน และถูกลบเป็นดังรูป 2.3ก ลบระเบียน #5 จะได้ดังรูป 2.3ข



ก. ข.

รูปที่ 2.3 การลบระเบียนโดยวิธีตัดต่อเชื่อมโยง

แม้ว่าวิธีตัดต่อเชื่อมโยงระเบียนทำให้ได้ข้อมูลที่มีโครงสร้างดี การลบออกก็ไม่เสียเวลามาก การเรียกกลับมาใช้ใหม่ก็ง่าย แต่ข้อมูลมีลักษณะไม่เป็นอิสระ หากการเชื่อมโยงเกิดข้อผิดพลาดขึ้น อาจมีปัญหาในการต่อให้เหมือนเดิม หรือถ้าใช้โครงสร้างแบบนี้เอาไว้เวลาจะนำไปใช้โดยโปรแกรมอื่น ก็จะต้องรู้วิธีเรียกข้อมูลตามโครงสร้างนี้ ข้อเสียอีกประการหนึ่งก็คือ ถ้าระเบียนที่ถูกลบมีจำนวนมาก ก็จะเป็นการเปลืองเนื้อที่ การจะลบทิ้งอย่างถาวรก็ทำได้ยาก ดังนั้นการออกแบบ File Editor จึงไม่ใช้โครงสร้างข้อมูลแบบใช้ตัวเชื่อมโยง

เมื่อประมวลข้อดีข้อเสียข้างต้นแล้ว พบว่าวิธีการที่เหมาะสมเพื่อแก้ปัญหาการลบระเบียนคือ เขียนสถานะถูกลบ (สำหรับระเบียนที่ต้องการลบทิ้ง) เอาไว้ก่อน เมื่อมีระเบียนเช่นนี้มากพอ ก็สั่งลบอย่างถาวรเสียทีหนึ่ง

ปัญหาการแทรกระเบียน ที่จะต้องพิจารณาเพิ่มเติมอีก คือ แทรกโดยเลื่อนระเบียนที่ถูกแทรกออกไป จัดการเชื่อมโยงลำดับ หรือไม่มีการแทรก (ต่อท้าย)

การแทรกระเบียนนั้นมีปัญหาตรงกันข้ามกับลบแบบถาวร คือ ต้องเลื่อนระเบียนออกไปแทนที่จะเลื่อนขึ้นมา วิธีการที่ใช้ คือ อ่านระเบียนที่จะถูกแทรกขึ้นมา เขียนระเบียนที่ต้องการแทรกทับลงไป อ่านระเบียนถัดไปขึ้นมา เขียนระเบียนที่อ่านมาก่อนทับลงไป จนกระทั่งหมดแฟ้ม ซึ่งข้อเสียที่เห็นได้ชัดก็คือ เสียเวลามาก แต่ข้อดีก็คือ สามารถเพิ่มเติมข้อมูลที่ตกหล่นได้ และเก็บไว้ในลำดับที่ต้องการ

วิธีจัดการเชื่อมโยงลำดับเพื่อแทรกระเบียน ก็มีลักษณะตรงกันข้ามกับการตัดต่อเชื่อมโยงเพื่อลบระเบียน ดังนั้นจะไม่ขอกล่าวในที่นี้

ส่วนวิธีการเพิ่มข้อมูลต่อท้ายโดยไม่มีการแทรกนั้น มีข้อดีคือ ง่าย ส่วนข้อเสีย คือ ลำดับข้อมูลที่แท้จริงต้องอาศัยการประมวลผลจัดลำดับให้ (DBMS ที่ใช้บนเครื่องมินิคอมพิวเตอร์ ใช้วิธีการนี้ เพราะการนำเสนอข้อมูลมีการจัดลำดับข้อมูลที่มีประสิทธิภาพสูง) ซึ่งสำหรับไมโครคอมพิวเตอร์จะเสียเวลามาก แต่ในการใช้งานจริงพบว่า การแทรกข้อมูลเกิดขึ้นไม่บ่อยนัก ดังนั้นวิธีการแทรกระเบียนโดยเลื่อนระเบียนที่ถูกแทรกออกไปจึงเป็นวิธีที่เหมาะสมสำหรับไมโครคอมพิวเตอร์

2.3.2 ระเบียนขนาดไม่คงที่

ระเบียนขนาดไม่คงที่ในแฟ้มข้อมูลนั้น เกิดขึ้นได้หลายกรณี เช่น

- มีการเก็บระเบียนหลายแบบไว้ในแฟ้มข้อมูลเดียวกัน

- ชนิดของระเบียนนั้นมีข้อมูลย่อยบางตัวที่สามารถเปลี่ยนแปลงขนาดได้
- ระเบียนที่มีข้อมูลย่อยซ้ำ ๆ กันได้ แต่มีจำนวนที่ซ้ำไม่เท่ากัน

ระเบียนที่มีขนาดไม่เท่ากันแต่ละแบบข้างต้น สามารถเก็บไว้ในแฟ้มข้อมูลได้หลายวิธี เช่น

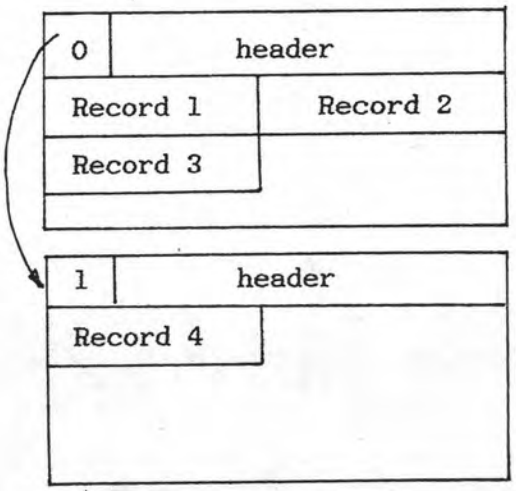
- เก็บเรียงต่อกันไปโดยมีตัวคั่นปิดท้ายระเบียน (sequencial record)
- เก็บโดยใช้ตัวชี้เชื่อมโยงระเบียน (linked record)

สำหรับวิธีเก็บระเบียนโดยใช้ตัวชี้เชื่อมโยงนั้นมีหลักการเชื่อมโยงทำนองเดียวกับที่กล่าวมาแล้วในหัวข้อย่อย 2.3.1 แต่ต่างกันตรงที่อ้างอิงหมายเลขระเบียนยากกว่า เพราะแฟ้มข้อมูลไม่ใช่ชนิด random access ดังนั้นการเก็บระเบียนขนาดไม่คงที่โดยวิธีใช้ตัวชี้เชื่อมโยงก็เป็นวิธีที่ไม่เหมาะสม (ถ้าทำงานในหน่วยความจำ การใช้ตัวชี้กับข้อมูลขนาดไม่คงที่เช่น ในโปรแกรมประมวลอักษร เป็นที่นิยมมาก แต่เวลาเก็บในแฟ้มข้อมูลจะเก็บแบบเรียงต่อกันไป)

วิธีเก็บระเบียนเรียงต่อกันไปโดยมีตัวคั่นปิดท้ายระเบียน วิธีการนี้เหมาะสำหรับเก็บข้อความ (text) ซึ่งระเบียนหนึ่งๆ ก็คือ ข้อความหนึ่งบรรทัด แต่ความยาวของข้อความแต่ละบรรทัดอาจไม่เท่ากัน ถ้าจะเก็บเป็นระเบียนขนาดคงที่ ต้องจองเนื้อที่ให้เพียงพอสำหรับเก็บบรรทัดที่มีข้อความที่ยาวที่สุดได้ ซึ่งจะทำให้เปลืองเนื้อที่โดยไม่จำเป็น ดังนั้นเพื่อเป็นการประหยัดเนื้อที่จึงเก็บข้อมูลเป็นระเบียนที่มีขนาดเท่ากับความยาวจริงของข้อความ ปิดท้ายด้วยรหัสจบบรรทัด (return code) วิธีเก็บระเบียนแบบนี้ใช้เก็บพารามิเตอร์สำหรับ File Editor (ดูหัวข้อย่อย 3.3.2) วิธีเก็บระเบียนเรียงต่อกันไปอีกแบบหนึ่งคือ เก็บตำแหน่ง (X,Y) บนจอภาพ และข้อความ ซึ่งได้แก่แฟ้มข้อมูลของฟอร์มที่สร้างโดย Screen Generator (ดูหัวข้อย่อย 3.3.5, *.SCN) ประเภทสุดท้ายที่เข้าข่ายเก็บระเบียนแบบเรียงต่อกัน แต่มีโครงสร้างระเบียนหลายแบบ คือ เก็บจำนวนข้อมูลแต่ละแบบตามด้วยเนื้อข้อมูลแบบนั้นๆ ซึ่งแฟ้มข้อมูลประเภทหลังสุดนี้ คือ แฟ้มข้อมูลของฟอร์มแบบลดขนาด (ดูหัวข้อย่อย 3.2.5, *.SEP)

2.3.3 รวมระเบียนเป็นกลุ่มขนาดคงที่

ในแฟ้มข้อมูลหนึ่งประกอบขึ้นด้วยระเบียน แต่ในการถ่ายข้อมูลระหว่างจานแม่เหล็กกับหน่วยความจำที่ใช้งานมีการถ่ายข้อมูลเป็นบล็อกหรือหน้า ดังนั้นถ้าสามารถจัดระเบียนที่เกี่ยวเนื่องกันรวมกันอยู่ในหน้าเดียวกันแล้วก็จะทำให้การอ่านข้อมูลของระเบียนต่างๆ จากจานแม่เหล็กมีประสิทธิภาพมากขึ้น เนื่องจากการอ่านข้อมูลจากจานแม่เหล็กนั้นเป็นจุดที่ช้าที่สุดในระบบ



รูปที่ 2.4 แสดงการรวมระเบียนเป็นกลุ่มขนาดคงที่

วิธีการนี้เป็นวิธีที่ดีอันหนึ่งในการทำให้สามารถอ่าน หรือเขียนข้อมูลกับจานแม่เหล็กได้อย่างมีประสิทธิภาพแต่ต้องมีการจัดการที่ซับซ้อนขึ้น เนื่องจากทางระบบปฏิบัติการ MS-DOS ไม่ได้รองรับการทำงานแบบนี้ จึงไม่ขอกล่าวรายละเอียดในที่นี้ ดังนั้นการออกแบบ File Editor จึงไม่ใช้โครงสร้างข้อมูลที่เก็บระเบียนเป็นกลุ่ม

2.4 ชนิดข้อมูลย่อยในระเบียน

ชนิดของข้อมูลย่อยในระเบียนนั้นเป็นไปตามภาษาปาสคาล ซึ่งมีรูปแบบข้อมูลอยู่ 5 ชนิด คือ ชนิดข้อความ ชนิดตัวอักษร ชนิดเลขจำนวนเต็ม ชนิดเลขจำนวนจริง และชนิดตรรกะ ในการเขียนโปรแกรมต้องกำหนดชนิดข้อมูลย่อยไว้ในโปรแกรมด้วย แต่ที่ต้องการให้สามารถ

กำหนดรูปแบบข้อมูลย่อยเหล่านี้จากภายนอกโปรแกรม จึงต้องกำหนดในรูปแบบของพารามิเตอร์ที่แทนรูปแบบของข้อมูลเหล่านั้น ด้วยวิธีนี้จะทำให้ผู้ใช้ไม่จำเป็นต้องทราบถึงข้อมูลที่เก็บลงในจานแม่เหล็กจริง (conceptual level)

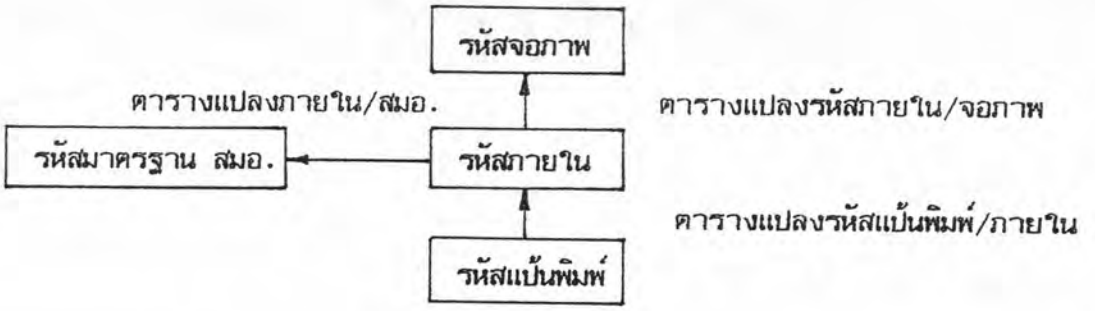
รหัสของรูปแบบเหล่านี้ คือ ชนิด S, C, I, R, B ซึ่งแทนข้อมูลของ ข้อความ ตัวอักษร เลขจำนวนเต็ม เลขจำนวนจริง และค่าตรรกะสำหรับข้อความนั้นต้องกำหนดตัวเลขตามหลังเพื่อบอกความยาวของข้อความนั้นตามหลัง เช่น ระเบียนผลสอบนักเรียน ในหัวข้อย่อย 2.3.1 มีรหัสแทนด้วย 'IS20RC'

นอกจากนั้น การที่จะให้รหัสนี้ใช้ในการควบคุมฟอร์มด้วย เนื่องจากฟอร์มจะต้องคู่กับแฟ้มข้อมูลนั้น จึงมีรหัสเพิ่มเติมคือ T, U, F สำหรับเปลี่ยนแป้นพิมพ์เป็นภาษาไทย เปลี่ยนเป็นข้อความอักษรตัวใหญ่ และกำหนดรูปแบบของวันที่ตามลำดับ รหัสที่เพิ่มมานั้นจะตรงกับชนิดข้อความของเดิม แต่ถ้าเป็น T การคำนวณความยาวของข้อมูลจะเป็นไปตามสูตร int (1.2 x length + 3) ดังนั้น T23 จะเท่ากับ S30 รหัส U ใช้เป็นการควบคุมฟอร์มอย่างเดียว จึงเทียบเท่ากับชนิด S สำหรับชนิด F นั้นใช้แทนวันที่ที่กำหนดความยาวไว้เท่ากับ 8 ตัวอักษรอยู่แล้วจึงไม่ต้องกำหนดความยาวของข้อความอีก ดังนั้น รหัส F เทียบเท่ากับ S8

สำหรับข้อมูลจริงที่เก็บลงในจานแม่เหล็กจะต่างออกไป ในขณะที่ผู้ใช้กำหนดเป็นรหัสแทนโครงสร้างแฟ้มข้อมูล ทางโปรแกรมภายในจะจัดการเพิ่มเติมสถานะการลบ ซึ่งเป็นชนิดตรรกะที่ตำแหน่งแรกของระเบียนด้วย การเขียนลงจานแม่เหล็กจึงเป็นระเบียนที่มีขนาดคงที่ตามชนิดข้อมูลย่อยที่กำหนดรวมทั้งสถานะการลบด้วย

2.5 รหัสอักขระ

รหัสอักขระที่ใช้จะแยกออกเป็น 3 ส่วนคือ รหัสอักขระที่ใช้ภายใน รหัสสำหรับออกทางจอภาพ และรหัสการรับจากทางแป้นพิมพ์ การกำหนดรหัสแยกเป็น 3 ส่วนนี้ ทำให้สามารถนำไปใช้กับจอภาพที่มีรหัสการออกจอภาพแตกต่างกันออกไป หรือใช้กับแป้นพิมพ์ที่มีการจัดเรียงตำแหน่งตัวอักษรต่างกัน การควบคุมความแตกต่างของรหัสทำได้โดยกำหนดตารางแปลงรหัสขึ้นเพื่อแปลงรหัสระหว่างรหัสภายใน รหัสออกจอภาพ และ รหัสจากแป้นพิมพ์ ดังรูปที่ 2.5



รูปที่ 2.5 แสดงการใช้ตารางแปลงรหัส

รหัสภายในที่ใช้นี้แตกต่างจากรหัสมาตรฐาน มอก.620-2529 เพื่อต้องการให้มีคุณสมบัติพิเศษอื่นๆ รหัสภายในส่วนภาษาไทย จะแบ่งเป็น 2 กลุ่ม คือ รหัสอักขระเดี่ยว และรหัสอักขระผสม

รหัสอักขระเดี่ยว ใช้เก็บเป็นข้อมูลในแฟ้มข้อมูล โดยมีการจัดเรียงลำดับรหัสอักขระเป็นแถวเดียวสำหรับอักขระที่แสดงอยู่ที่คอลัมน์เดียวกัน คือ อักขระกลาง อักขระบน/ล่าง และวรรณยุกต์ หรือไม้ทัณฑฆาต ตามลำดับ ส่วนรหัสผสม ใช้แสดงออกทางจอภาพ

การกำหนดรหัสอักขระเดี่ยวนี้ มีการจัดเรียงตำแหน่ง สระต่างๆ ในตารางรหัสให้เรียงกันเป็นกลุ่มต่อเนื่องกัน และเรียงตามลำดับ ตามการจัดลำดับสระในการเรียงคำตามพจนานุกรมไทย สำหรับรหัสผสม เป็นการผสมระหว่างสระบน และวรรณยุกต์ หรือไม้ทัณฑฆาต จัดวางเรียงเป็นกลุ่มเพื่อให้การผสมอักขระทำได้ง่าย ส่วนนี้กำหนดขึ้นเพื่อให้สามารถใช้กับการออกทางจอภาพที่ใช้อักขระผสม

การควบคุมความแตกต่างของรหัสโดยใช้ตารางแปลงรหัส ทำให้การกำหนดรหัสต่างๆ ทั้งทางการรับเข้า และแสดงออก ยืดหยุ่นมากขึ้น และถ้าต้องการถ่ายข้อมูลไปเป็นรหัสอื่น ก็ใช้ตารางการแปลงไปเป็นรหัสนั้น รายละเอียดตารางแปลงรหัส ดูที่ภาคผนวก ก.

นอกจากนี้ยังมีตารางการจำแนกอักขระออกเป็นกลุ่มเพื่อประโยชน์ในการตรวจสอบความถูกต้องการแสดงผล และการเปรียบเทียบด้วย การจัดรูปแบบคำไทยนั้นมีลักษณะที่แตกต่างจากภาษาอังกฤษ คือ สามารถแบ่งการแสดงผลออกได้เป็น 4 ระดับ คือ ส่วนอักษรกลาง ส่วนสระบน สระล่าง และวรรณยุกต์ ส่วนเหล่านี้จะมีการแสดงอยู่ประจำระดับ และในภาษาไทยมีกฎเกณฑ์ต่างๆ ในการเขียน เช่น สระบน ล่าง จะไม่อยู่พร้อมกันเป็นต้น จึงสามารถจำแนกกลุ่มอักขระ ออกเป็นกลุ่มดังนี้

DEC		0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0										◌		๐	ฯ	ฒ	ย	ะ
1	1										◌		๑	ก	ณ	ร	ั
2	2										◌		๒	ข	ด	ฤ	า
3	3										◌		๓	ค	ต	ล	ำ
4	4									◌	◌		๔	ฆ	ถ	ฦ	ิ
5	5									◌	◌		๕	ง	ท	ว	ี
6	6									◌	◌		๖	จ	ธ	ศ	ึ
7	7									◌	◌		๗	ฉ	น	ษ	ื
8	8										◌		๘	ช	บ	ส	ุ
9	9										◌		๙	ซ	ป	ห	ู
10	A										◌		ฺ	ฌ	ผ	ฬ	เ
11	B									◌	◌		฻	ญ	ฝ	อ	แ
12	C									◌	◌		฼	ฎ	พ	ฮ	โ
13	D									◌	◌		฽	ฏ	ฟ	ฯ	ใ
14	E									◌	◌		฾	ฐ	ภ	๎	ไ
15	F									◌	◌		฿	ฑ	ม	๏	

รูปที่ 2.6 ตารางรหัสภายใน

0 อักขระอังกฤษทั้งหมดตามตารางรหัส ASCII

1 ตัวเลขไทย

2 เครื่องหมาย ๆ ฯ

3 พยัญชนะ

4 ไม้หันอากาศ

5 ไม้ไต่คู้

6 สระหลัง/บน/ล่าง

7 สระหน้า

8 วรรณยุกต์

กลุ่มรหัสเหล่านี้แบ่งเป็นกลุ่มเพื่อประโยชน์ในการเปรียบเทียบ แต่ในการแสดงผลจะมีการแบ่งกลุ่มที่ละเอียดมากขึ้นกว่านี้ รายละเอียดกลุ่มรหัสดูที่ภาคผนวก ก.

2.6 การติดต่อกับผู้ใช้

ในการจัดการระบบฐานข้อมูลโดยทั่วไป ส่วนที่ทำหน้าที่ติดต่อกับผู้ใช้เป็นส่วนช่วยให้ผู้ใช้สามารถทำงานต่างๆ ในการจัดการฐานข้อมูลได้อย่างสะดวกง่ายดาย และรวดเร็ว ส่วนติดต่อกับผู้ใช้ เป็นส่วนที่อยู่บนสุดในการแบ่งระดับ การทำงาน (external level) ที่กล่าวถึงในหัวข้อ 2.3 โดยระดับนี้จะซ่อนการทำงานระดับล่างไว้จากผู้ใช้

การกำหนดโครงสร้างของแฟ้มข้อมูล และจัดการกับแฟ้มข้อมูล เป็นหน้าที่การทำงานของโปรแกรม File Editor โดยผู้ใช้กำหนดพารามิเตอร์ของโครงสร้างของแฟ้มข้อมูลให้แก่โปรแกรมที่ได้กล่าวในหัวข้อ 2.4 ซึ่งทำให้โปรแกรมสามารถกำหนดการทำงานกับจานแม่เหล็กได้ แต่ในการติดต่อกับผู้ใช้เพื่อให้ผู้ใช้สามารถเพิ่มเติม หรือแก้ไขข้อมูลภายในระเบียนได้โดยง่ายนั้น การใช้ฟอร์มในการติดต่อกับผู้ใช้เป็นวิธีที่ดีวิธีหนึ่งในการแสดงระเบียนนั้น เนื่องจากมีการใช้รูปแบบของฟอร์มอยู่แล้วในการเก็บข้อมูลแบบคนทำ

เครื่องมือทางฐานข้อมูลที่นิยมใช้บนไมโครคอมพิวเตอร์ เช่น dBASE และ PFS หรือ โปรแกรมอื่นๆ เช่น INGRES, FMS ที่สามารถจัดการกับแฟ้มข้อมูลได้ง่าย ก็ใช้ความสามารถในรูปแบบของฟอร์มในการจัดการกับระเบียนหนึ่ง ๆ แต่ยังขาดความสามารถทางด้านภาษาไทยอีกมาก

การจัดการทางด้านภาษาไทย ก็เป็นการแสดงคำไทยให้ถูกต้องตามลักษณะของภาษาไทย โดยจัดการแสดงสระบนล่างให้ถูกต้อง ตรวจสอบความเป็นไปได้ของคำไทยบางส่วนเพื่อป้องกันข้อผิดพลาดที่อาจเกิดขึ้นทางด้านข้อความภาษาไทย รวมทั้งการเปรียบเทียบในการค้นหา เช่น ในการหาข้อความที่มีพยัญชนะหน้าตามที่กำหนด ต่อไปจะกล่าวถึงการติดต่อกับผู้ใช้โดยใช้ฟอร์ม

ในการจัดเก็บรวบรวมข้อมูลเอกสารต่างๆ ทั้งทางธุรกิจ หรือเอกสารทางราชการนั้น การรวบรวมข้อมูลที่ง่ายที่สุดคือ การรวบรวมในรูปแบบของฟอร์ม เช่น ในการออกแบบสอบถาม การบันทึกการทำงาน เป็นต้น การเก็บข้อมูลในรูปแบบฟอร์ม หมายถึงว่า มีข้อความบางอย่างที่แสดงถึงรายละเอียดของข้อมูลต่าง ๆ โดยมีช่องว่างให้เติมข้อมูลที่ต้องการลงไป โดยข้อมูลที่สำคัญ คือข้อมูลส่วนที่เติมลงในช่องว่างนั้น จะเห็นได้ว่า ในทางปฏิบัติแล้วการรวบรวมข้อมูลก็ใช้ฟอร์มเหล่านี้เป็นจำนวนมาก ดังนั้นในการใช้คอมพิวเตอร์มาช่วยในการจัดการแฟ้มข้อมูลจึงใช้ในรูปแบบของฟอร์ม จะทำงานได้โดยง่าย

ฟอร์มที่กล่าวถึงนี้มิได้หมายถึงว่า จะใช้ได้เฉพาะกับการรับข้อมูลเท่านั้น ยังรวมถึงการแสดงข้อมูลเหล่านั้นออกมา ไม่ว่าจะอยู่ในรูปแบบของสื่อใด ๆ

ในเอกสารหนึ่ง ๆ ถ้ามีรูปแบบที่แน่นอนแล้ว และเราทราบโครงสร้างของฟอร์มนั้น เราก็สามารถดึงข้อมูลที่ต้องการออกจากเอกสารเหล่านั้นได้ โดยที่ไม่ต้องเก็บฟอร์มที่กรอกข้อมูลแล้วทั้งหมด ดังนั้นในฟอร์มหนึ่ง ๆ จะประกอบด้วย

ก. ส่วนที่ปรากฏคงที่ (text template)

ข. ช่องว่างให้เติมข้อมูลลงไป (slot)

ส่วนที่ปรากฏคงที่ ใช้อธิบายความเข้าใจในการเติมข้อมูลในช่องว่าง รวมทั้งใช้แสดงข้อความอื่น ๆ เพื่อประกอบกันเป็นเอกสารฉบับหนึ่ง

ส่วนช่องว่างให้เติมข้อมูล ข้อมูลที่เติมนั้นจะแตกต่างกันไปในแต่ละเอกสาร ส่วนนี้จึงเป็นส่วนที่เกี่ยวพันกับข้อมูลในฐานข้อมูล

เราสามารถกำหนดรูปแบบของฟอร์มโดยทั่วไปได้ดังนี้ [7]

ก. ฟอร์มที่ใช้ในการสื่อสารข้อมูล นอกจากที่ปรากฏในรูปข้อความแล้ว ยังปรากฏในรูปแบบอื่นๆ อีก ที่ต้องการแสดงข้อมูลบางอย่างในช่องว่าง ประกอบกับข้อมูลอื่นที่มีอยู่แล้วเพื่อสื่อให้ผู้ใช้เข้าใจ เช่น ฟอร์มในการพูด ซึ่งต้องมีข้อความที่พูดซ้ำ ๆ กัน โดยมีบางคำเปลี่ยนไป ตัวอย่างเช่น ในการทำ Tele-Banking ซึ่งอาจเป็นการถามยอดบัญชีเงินฝาก ใน

การตอบคำถามนั้น ต้องมีข้อความบางอย่างที่ส่งออกมาเพื่อความเข้าใจของผู้ใช้ ข้อความที่ต่างกันในแต่ละผู้ใช้ก็คือ ยอดเงินเท่านั้น โดยหลักการเดียวกันนี้ก็เป็นแบบเดียวกับการใช้ฟอร์มในการแสดงผลบนจอภาพ หรือฟอร์มของการแสดงผลทางเครื่องพิมพ์

ข. ฟอร์มควรมีรูปแบบที่เหมือนกัน ไม่ว่าจะเป็นสื่อแบบใด เมื่อมีข้อมูลชุดเดียวกัน สามารถที่จะใช้ template ได้หลายชุด เพื่อความสะดวกในการเปลี่ยนรูปแบบการมองข้อมูลชุดนั้น

ค. ความสัมพันธ์ระหว่างฟอร์มกับค่าที่อยู่ในฟอร์มต้องไม่ยึดติดกัน ซึ่งขึ้นกับการกำหนดว่า ส่วนใดของฟอร์มควรเป็นส่วนที่ปรากฏคงที่ หรือเป็นช่องว่างให้เติม หรือแสดงข้อมูลหรือให้ดึงค่าจากฐานข้อมูลมาแทนที่

ง. การทำงานต่าง ๆ ที่จะเกิดขึ้นในฟอร์มนั้น เช่น ในการใส่ข้อมูลในใบส่งของโดยฟอร์ม ผลรวมของข้อมูลที่ใส่อาจให้ผู้ใช้ใส่เอง หรือมีการคำนวณค่าโดยอัตโนมัติ การคำนวณอาจเกิดขึ้นในฟอร์มเอง ซึ่งอาจทำให้เกิดผลข้างเคียงอื่น ๆ ขึ้นได้ เช่นเมื่อมีการเปลี่ยนแปลงค่าใดค่าหนึ่งในใบส่งของผลที่ได้จากการคำนวณก็ต้องเปลี่ยนแปลงด้วย ซึ่งช่องผลรวมนี้อาจถูกจำกัดในการเปลี่ยนแปลงจากผู้ใช้

จ. มีการกำหนดชนิดของข้อมูลภายในฟอร์มนั้น ซึ่งนอกจากจะเกี่ยวพันกับการจัดรูปแบบในการแสดงข้อมูลแล้ว ยังสามารถใช้เป็นตัวตรวจสอบความถูกต้องของข้อมูล (integrity) นั้น ในระบบได้อีกด้วย เช่น ข้อมูลที่เป็นตัวเลข ถ้ามีตัวอักษรปนอาจทำให้การนำข้อมูลนั้นไปคำนวณต่อไป มีความผิดพลาดเกิดขึ้น

2.7 สรุป

ในบทนี้ได้กล่าวถึงการตัดสินใจเลือกภาษาที่ใช้ในการพัฒนาโปรแกรม โดยคำนึงถึงความสะดวกในการใช้เป็นหลัก ความง่ายในการพัฒนาโปรแกรม และสามารถใช้กับไมโครคอมพิวเตอร์ที่กำหนดได้ ซึ่งก็คือ คอมไพเลอร์ เทอร์โบ ปาสคาล และได้กล่าวถึงการจัดโครงสร้างแฟ้มข้อมูลแบบต่างๆ ที่เกิดจากการแบ่งระดับการมองจากผู้ใช้ จากโปรแกรม และจากระบบปฏิบัติการ

การกำหนดโครงสร้างแฟ้มข้อมูลได้กล่าวถึงข้อมูลภายในระเบียน ในการแทนค่าข้อมูลบางอย่างด้วยรหัส หรือใช้ตัวอักษรแทนชนิดข้อมูลนั้น การจัดระเบียนแบบต่าง ๆ ทั้งแบบ

ระเบียนขนาดคงที่ ระเบียนขนาดไม่คงที่ และระเบียนรวมกันเป็นหน้า การพิจารณาเลือกโครงสร้างแฟ้มข้อมูล และข้อมูลภายในระเบียน เลือกใช้แบบเดียวกับการจัดการแฟ้มข้อมูลของภาษาปาสคาลในแบบ random access เพื่อให้สามารถเขียนโปรแกรมอื่นๆ ด้วยภาษาปาสคาลมาจัดการกับแฟ้มข้อมูลที่สร้างขึ้นได้โดยง่าย

การแบ่งลักษณะการทำงานได้แบ่งออกเป็น 3 ระดับ คือ ส่วน internal level ที่เป็นส่วนจัดการกับระเบียนในการเขียน หรืออ่านจานแม่เหล็ก conceptual level เป็นการกำหนดโครงสร้างของระเบียนโดยผู้ใช้ โดยผู้ใช้ไม่ต้องรู้ถึงการเขียน หรืออ่านจานแม่เหล็ก ใช้การกำหนดรหัสเป็นพารามิเตอร์ของโปรแกรม และ external level เป็นส่วนติดต่อกับผู้ใช้โดยตรงซึ่งจะใช้ฟอร์มเป็นหลัก

นอกจากนี้ยังได้กล่าวถึงรหัสที่ใช้ภายในซึ่งแตกต่างจากรหัสมาตรฐาน สมอ. ที่ มอก. 620-2529 เนื่องจากกำหนดขึ้นเพื่อความเหมาะสมในการใช้งานภายในต่าง ๆ และการแบ่งกลุ่มรหัสอักขระเพื่อใช้ในการจัดวางและเปรียบเทียบ