บทที่ 3

ตัวจำลองระบบเครือข่าย NS เวอร์ชัน 2.0

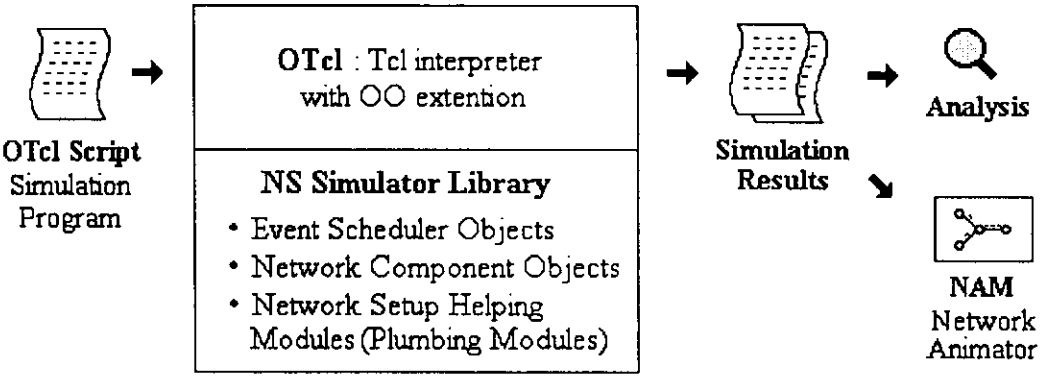
บทที่ผ่านมาได้กล่าวถึงตัวจำลองระบบเครือข่ายที่มีในปัจจุบัน หลังจากการศึกษาและทดลองใช้งานแล้ว ผู้ทำการวิจัยได้เลือกใช้ตัวจำลองระบบเครือข่าย NS เวอร์ชัน 2.0 สำหรับงานวิจัยชิ้นนี้ โดยในบทนี้จะกล่าวถึงเหตุผลในการตัดสินใจเลือกตัวจำลองระบบเครือข่ายดังกล่าว ส่วนประกอบทางซอฟต์แวร์ ส่วนติดต่อกับผู้ใช้ ส่วนแสดงผลลัพธ์ รวมถึงองค์ประกอบและโครงสร้างของโปรแกรมตัวจำลองระบบเครือข่าย NS

3.1 เหตุผลในการตัดสินใจเลือกตัวจำลองระบบเครือข่าย NS 2.0

จากโปรแกรมตัวจำลองระบบเครือข่ายทั้งหมดที่ได้กล่าวมาแล้วในบทที่ 2 เมื่อพิจารณาคุณสมบัติในด้านต่าง ๆ รวมถึงลักษณะการนำไปใช้งานของแต่ละโปรแกรม พบว่าโปรแกรม NS เวอร์ชัน 2.0 มีคุณสมบัติที่เหมาะสม และเป็นที่นิยมใช้กันอย่างแพร่หลาย และยังได้รับความไว้วางใจจากหลายสถาบัน ไม่ว่าจะเป็น Lawrence Berkeley National Laboratory (LBNL), Xerox Palo Alto Research Center (Xerox Parc), University of California, Berkeley (UCB), Information Sciences Institute of the University of Southern California (USC/ISI)

3.2 ส่วนประกอบทางซอฟต์แวร์ของ NS 2.0

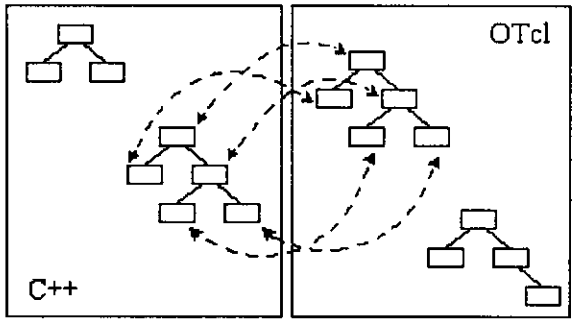
การทำงานของโปรแกรมตัวจำลองระบบเครือข่าย NS 2.0 ต้องทำการสร้างแบบจำลองโทโปโลยีของระบบเครือข่ายและทำการกำหนดค่าฟังก์ชันการทำงานต่าง ๆ ของเครือข่าย ด้วยการเรียกใช้อ้อปเจ็กต่าง ๆ จากไลบรารี โดยเขียนเป็น OTcl Script เพื่อป้อนให้กับตัวจำลองระบบเครือข่าย แสดงดังภาพประกอบ 3-1 รวมถึงต้องกำหนดเวลาในการเริ่มและหยุดส่งข้อมูลด้วย ส่วนผลลัพธ์ที่ได้หลังจากการทำงานของตัวจำลอง (Simulation Results) ประกอบด้วยรายละเอียดของข้อมูลที่ได้จากการจำลอง ซึ่งสามารถนำไปใช้ในการวิเคราะห์หาประสิทธิภาพของระบบเครือข่าย หรือนำไปแสดงผลแบบกราฟฟิกได้ โดยใช้ Network Animator (NAM) และจะได้กล่าวถึงต่อไปในหัวข้อ 3.3



ภาพประกอบ 3-1 ภาพโดยรวมในการใช้งานโปรแกรม NS 2.0

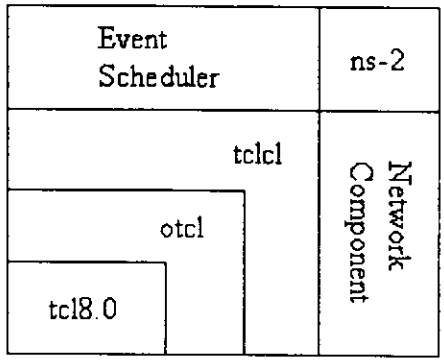
(ที่มา : http://nile.wpi.edu/NS/)

ตัวจำลองระบบเครือข่าย NS 2.0 เป็นโปรแกรมที่ประกอบด้วย 2 ภาษา ได้แก่ ภาษา C++ และภาษา OTcl ซึ่งก็คือ ภาษา Tcl แต่ได้รับการพัฒนาให้รองรับในด้านของ Object-oriented เนื่องมาจากเหตุผลในเรื่องของประสิทธิภาพการทำงาน NS 2.0 จึงได้แยกส่วนของข้อมูลออกจากส่วนที่ใช้ในการควบคุม เพื่อช่วยลดระยะเวลาที่ใช้ในการประมวลผลของแพ็กเกตและเหตุการณ์ต่าง ๆ เนื่องจากการจำลองโปรโตคอลในส่วนของรายละเอียดปลีกย่อย ต้องการภาษาที่มีความสามารถในการจัดการกับกลุ่มข้อมูล และสามารถใช้ในการสร้างอัลกอริทึมเพื่อจัดการกับกลุ่มข้อมูลชุดใหญ่ ๆ ได้อย่างมีประสิทธิภาพ ภาษาที่เหมาะสมกับงานดังกล่าวและมีความรวดเร็วในการทำงาน คือ ภาษา C++ และสาเหตุที่ใช้ภาษา OTcl เนื่องจากในการเปลี่ยนแปลงค่าพารามิเตอร์หรือค่า Configurartion ต่าง ๆ ของแบบจำลองเครือข่าย ต้องการความรวดเร็วในการทำงานด้วยเช่นกัน โดยความสัมพันธ์แสดงดังภาพประกอบ 3-2



ภาพประกอบ 3-2 ความสัมพันธ์ระหว่าง C++ และ OTcl

(ที่มา : http://nile.wpi.edu/NS/)

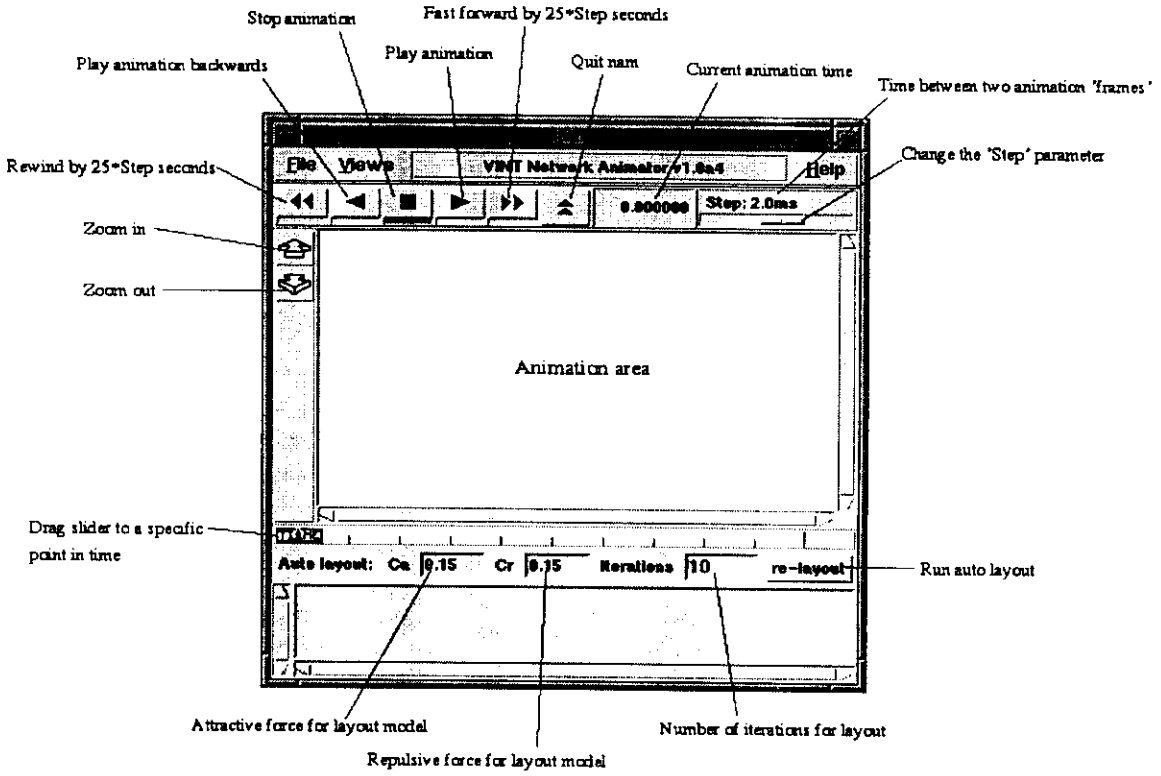


ภาพประกอบ 3-3 โครงสร้างในเชิงสถาปัตยกรรมของ NS 2.0
(ที่มา : http://nile.wpi.edu/NS/)

จากภาพประกอบ 3-3 ในการเขียน OTcl Script จะเขียนจากภาษา Tcl 8.0 และจะเรียกใช้อ้อปเจ็กต่าง ๆ ที่มีอยู่ในไลบรารีของ OTcl และในส่วนของ Event Scheduler และ Network Component ซึ่งเขียนด้วยภาษา C++ จะถูกเรียกใช้จากภาษา OTcl โดยผ่านทาง OTcl Linkage ซึ่งเขียนด้วย Tclcl (Tcl with classes) หรือเรียกอีกอย่างว่า OTcl/C++ Interface โดยทั้งหมดจะประกอบกันเป็นโปรแกรมตัวจำลองระบบเครือข่าย NS 2.0

3.3 ส่วนติดต่อกับผู้ใช้

NS 2.0 มีส่วนติดต่อกับผู้ใช้เป็นแบบกราฟฟิกชื่อ NAM เพื่อให้ง่ายต่อการใช้งาน แสดงดังภาพประกอบ 3-4



ภาพประกอบ 3-4 Network Animator (NAM)

(ที่มา : http://www.isi.edu/nsnam/ns/tutorial/)

◀◀ Rewind by 25*Step seconds

เมื่อคลิกปุ่มนี้ หรือกด 'r', 'R' Animation time จะมีค่าลดลงเท่ากับ 25 เท่าของ Step

◀ Play animation backwards

เมื่อคลิกปุ่มนี้ ภาพเคลื่อนไหวเคลื่อนไหวถอยหลัง

■ Stop animation

เมื่อคลิกปุ่มนี้ หรือกด 'p', 'P' จะหยุดภาพเคลื่อนไหว และกด 'c', 'C' เพื่อดำเนินต่อไป

▶ Play animation

เมื่อคลิกปุ่มนี้ ภาพเคลื่อนไหวเคลื่อนไหวไปข้างหน้า

▶▶ Fast forward by 25*Step seconds

เมื่อคลิกปุ่มนี้ หรือกด 'f', 'F' Animation time จะมีค่าเพิ่มขึ้นเท่ากับ 25 เท่าของ Step

Quit nam

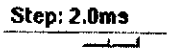
เมื่อคลิกปุ่มนี้ หรือกด 'q', 'Q', 'Ctrl-c' จะออกจาก NAM

Zoom in

เมื่อคลิกปุ่มนี้ หรือกด '>', '.' จะเพิ่มขนาดของภาพเคลื่อนไหวให้มีขนาดใหญ่ขึ้น

Zoom out

เมื่อคลิกปุ่มนี้ หรือกด '<', ',' จะลดขนาดของภาพเคลื่อนไหวให้มีขนาดเล็กลง



Time between two animation 'frames' (Step)

แสดงช่วงเวลาระหว่างภาพเคลื่อนไหวแต่ละเฟรม

Current animation time

แสดงเวลาปัจจุบันของภาพเคลื่อนไหว

3.4 ส่วนแสดงผลลัพธ์

รูปแบบของผลลัพธ์ที่ได้จากการทำงานของตัวจำลอง จะได้ออกมาเป็น Trace File (.tr) ซึ่งมีรูปแบบดังภาพประกอบ 3-5

event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

```
r : receive (at to_node)
+ : enqueue (at queue)                    src_addr : node.port (3.0)
- : dequeue (at queue)                    dst_addr : node.port (0.0)
d : drop    (at queue)
```

```
r 1.3556 3 2 ack 40 ------- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
```

ภาพประกอบ 3-5 รูปแบบของ Trace File

(ที่มา : http://www.isi.edu/nsnam/ns/tutorial/)

จากภาพประกอบ 3-5 แต่ละสดมภ์มีความหมายดังนี้

event หมายถึง เหตุการณ์ซึ่งมีด้วยกัน 4 เหตุการณ์ คือ r (Receive) แพ็กเกตที่ได้รับ + (Enqueue) แพ็กเกตที่รอเข้าคิว - (Dequeue) แพ็กเกตที่ออกจากคิว และ d (Drop) แพ็กเกตที่ถูกทิ้ง

time (Simulation Time) หมายถึง เวลาของการจำลอง มีหน่วยเป็นวินาที

from node หมายถึง ส่งมาจากโหนดไหน

to node หมายถึง ส่งไปยังโหนดไหน

pkt type (Packet Type) หมายถึง ประเภทของแพ็กเกต

pkt size (Packet Size) หมายถึง ขนาดของแพ็กเกต มีหน่วยเป็นไบต์ (Bytes)

flags ไม่ได้มีการกำหนดค่าใด ๆ

fid (Flow ID) ใช้สำหรับในการวิเคราะห์ข้อมูล และใช้ในการแสดงผลแบบกราฟฟิก (NAM)

src addr (Source Address) หมายถึง แอดเดรสต้นทาง อยู่ในรูปแบบของ node.port เพื่อบอกว่าข้อมูลถูกส่งมาจากโหนดต้นทางไหน

dst addr (Destination Address) หมายถึง แอดเดรสปลายทาง อยู่ในรูปแบบของ node.port เพื่อบอกว่าข้อมูลถูกส่งไปยังโหนดปลายทางไหน

seq num (Sequence Number) หมายถึง หมายเลขลำดับของแพ็กเกต

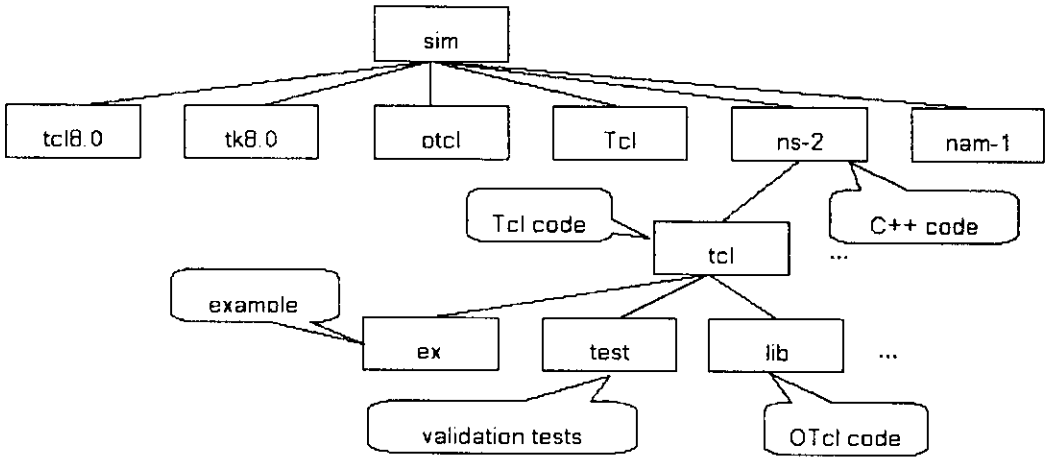
pkt id (Packet ID) หมายถึง หมายเลขประจำแพ็กเกต โดยแต่ละแพ็กเกตจะมีหมายเลขไม่เหมือนกัน (Unique)

ข้อมูลเหล่านี้สามารถนำมาใช้ในการวิเคราะห์ประสิทธิภาพของระบบเครือข่ายได้ โดยเลือกเฉพาะสดมภ์ที่สนใจจะใช้ในการวิเคราะห์ ซึ่งอาจใช้ภาษา awk เข้ามาช่วย และเมื่อได้ข้อมูลที่ต้องการแล้ว สามารถใช้ส่วนของ Xgraph ในการแสดงผลแบบกราฟได้

3.5 องค์ประกอบของโปรแกรม NS 2.0

องค์ประกอบของโปรแกรมตัวจำลองระบบเครือข่าย NS 2.0 ประกอบด้วยหลายส่วนด้วยกัน แสดงดังภาพประกอบ 3-6 ซึ่งแบ่งออกเป็นโฟลเดอร์ย่อย ๆ อาทิเช่น โฟลเดอร์ tcl 8.0 โฟลเดอร์ ns-2 และโฟลเดอร์ nam-1 เป็นต้น โดยในโฟลเดอร์ ns-2 จะเป็นส่วนที่เก็บโปรแกรมที่เขียนด้วยภาษา C++ และมีโฟลเดอร์ย่อย ซึ่งจะเป็นส่วนที่เก็บโปรแกรมที่เขียนด้วยภาษา Tcl คือ โฟลเดอร์ tcl ภายในยังประกอบด้วย ตัวอย่างโปรแกรมและการเรียกใช้ฟังก์ชันต่าง ๆ รวมถึง

โปรแกรมที่เขียนด้วยภาษา OTcl ซึ่งอยู่ในโฟลเดอร์ lib ส่วนโฟลเดอร์ nam-1 เป็นส่วนที่เก็บโปรแกรมที่ทำหน้าที่ติดต่อกับผู้ใช้แบบกราฟฟิก หรือที่เรียก NAM ดังที่ได้กล่าวไปแล้วข้างต้น

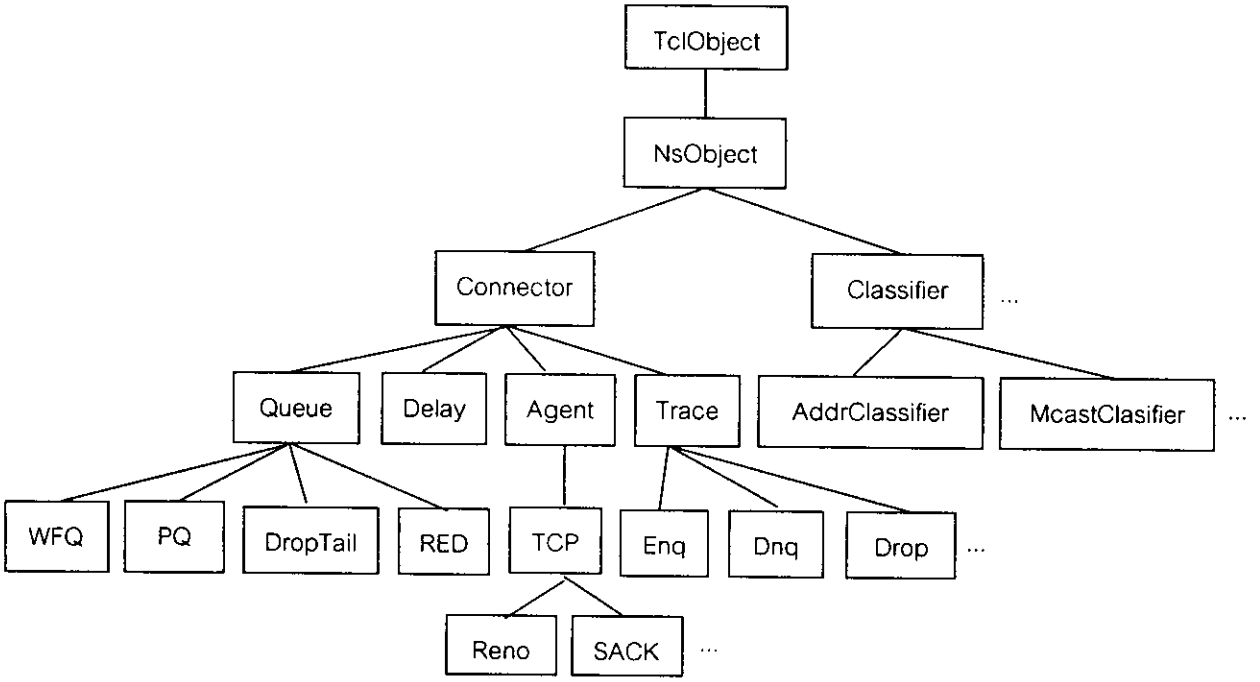


ภาพประกอบ 3-6 องค์ประกอบของโปรแกรม NS 2.0

(ที่มา : ftp://catarina.usc.edu/pub/huang/talk/ns-tutorial/intermediate-ns.ppt)

3.6 โครงสร้างของโปรแกรม NS 2.0

โครงสร้างของโปรแกรมตัวจำลองระบบเครือข่าย NS 2.0 จะประกอบด้วย ฟังก์ชันการทำงานหลายส่วน โดย TclObject ถือเป็นคลาสหลักของ NS 2.0 ส่วน Connector เป็นฟังก์ชันที่ทำหน้าที่รับแพ็กเกตเข้ามาแล้วจัดการอะไรบางอย่าง เช่น ทำการจัดคิวแพ็กเกตที่เข้ามา แล้วส่งแพ็กเกตออกไปหรืออาจทิ้งแพ็กเกตนั้น และ Classifier เป็นฟังก์ชันของโหนด โดยแต่ละโหนดอาจใช้ Classifier คนละประเภทกันขึ้นกับวัตถุประสงค์ของการใช้งาน ส่วนที่ทำการเพิ่มเติมขึ้นเป็นส่วนที่เกี่ยวข้องกับอัลกอริทึมที่ใช้ในการจัดคิว ได้แก่ การจัดคิวแบบ Weighted Fair Queuing (WFQ) และการจัดคิวแบบ Priority Queuing (PQ) (โปรแกรมแสดงอยู่ในภาคผนวก ข) แสดงดังภาพประกอบ 3-7 ส่วนหลักการทำงานจะได้กล่าวถึงในบทต่อไป



ภาพประกอบ 3-7 โครงสร้างของโปรแกรม NS 2.0

(ที่มา : ปรับปรุงจาก ftp://catarina.usc.edu/pub/huang/talk/ns-tutorial/intermediate-ns.ppt)

3.7 สรุป

โปรแกรมตัวจำลองระบบเครือข่าย NS เวอร์ชัน 2.0 สามารถใช้งานได้ทั้งในระบบปฏิบัติการวินโดวส์และระบบปฏิบัติการยูนิกซ์ แต่ถูกสร้างขึ้นเพื่อให้ใช้งานได้ดีกับระบบปฏิบัติการยูนิกซ์ และจากที่ได้กล่าวมาทั้งหมดเป็นภาพโดยรวมของตัวจำลองระบบเครือข่าย NS 2.0 ซึ่งในบทต่อไปจะกล่าวถึงหลักการทำงานของอัลกอริทึมที่ใช้ในการจัดคิวแบบต่าง ๆ