ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΚΑΤΑΝΕΜΗΜΕΝΑ ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

Διδακτορική Διατριβή του

Παναγιώτη Θ. Κατσαρού
Πτ. Μαθηματικών Α.Π.Θ. – MSc in Software Engineering, Aston Un. UK

υπό την επίβλεψη του Καθηγητή κ. Κ. Λάζου

ΘΕΣΣΑΛΟΝΙΚΗ – ΙΑΝΟΥΑΡΙΟΣ 2002

Στη Δήμητρα και

στο μικρό Θεοφάνη

Κατανεμημένα Υπολογιστικά Συστήματα

Παναγιώτης Θ. Κατσαρός

Τμήμα Πληροφορικής
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
54006 - Θεσσαλονίκη
katsaros@csd.auth.gr

Περίληψη

Οι σύγχρονες ανάγκες επεξεργασίας δεδομένων οδηγούν στην ανάπτυξη κατανεμημένων διατάξεων αρχιτεκτονικής πελάτη - διακομιστή (client - server). Κάθε νέο υπολογιστικό σύστημα αποτελείται πλέον από ένα σύνολο τμημάτων υλικού και λογισμικού, πιθανώς διαφορετικών τεχνολογιών ή και διαφορετικών κατασκευαστών, η διασύνδεση των οποίων υποστηρίζεται από σταθμούς εργασίας - μέλη ενός δικτύου.
Μέσα σε αυτό το πλαίσιο, ο εντοπισμός των παραγόντων που επιδρούν καθοριστικά στην απόδοση και η εκτίμηση αυτής σε εναλλακτικές περιπτώσεις διαμόρφωσης του συστήματος προϋποθέτουν την υιοθέτηση μιας νέας προσέγγισης, πέρα από την κλασσική της χρήσης αναλυτικών ή προσομοιωτικών «επίπεδων» δικτύων ουρών.
Σκοπός της διατριβής αυτής ήταν η ανάπτυξη ή/και η βελτίωση των τεχνικών εκείνων, που είναι απαραίτητες στην ανάλυση της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής.

Ευχαριστίες

Πριν από όλους θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Κ. Λάζο για την εμπιστοσύνη, που έδειξε στο πρόσωπό μου, και με επέλεξε για να φέρω σε πέρας τη μελέτη αυτή. Η καθοδήγηση, η ενθάρρυνση, η υποστήριξη και η πολύτιμη κριτική του καθ΄ όλη τη διάρκεια της εξέλιξης του ερευνητικού μου έργου συνέβαλαν αποφασιστικά στην επιτυχή περαίωση αυτού.

Θα ήθελα ακόμη να αναφερθώ και στη συμβολή των κ. Γ. Μανωλόπουλου και Ε. Καρατζά, οι οποίοι ως τα δύο άλλα μέλη της τριμελούς επιτροπής παρακολούθησαν την εξέλιξη του ερευνητικού μου έργου, που είχε ως αποτέλεσμα τη διατριβή αυτή. Επίσης, θα ήταν παράλειψη να μη γίνει μνεία στους συναδέλφους μου υποψηφίους διδάκτορες και στο λοιπό προσωπικό και μέλη ΔΕΠ του Τμήματος Πληροφορικής του Α.Π.Θ., που όλοι μαζί αποτέλεσαν επί σειρά ετών ένα ευχάριστο περιβάλλον γόνιμης συνεργασίας.

Ένα μεγάλο ευχαριστώ ανήκει σίγουρα στους γονείς μου Θεοφάνη και Αθηνά για όσα κάνανε για τη μόρφωσή μου, καθώς αυτοί πρώτοι μου δείξανε το δρόμο της γνώσης. Με βοήθησαν επίσης και στα τελευταία στάδια της συγγραφής της διατριβής, τα οποία ήταν ιδιαίτερα δύσκολα με την παρουσία του μικρού Θεοφάνη, που διακαώς επέμενε να συμμετάσχει στην προσπάθειά μου.

Τέλος, θα ήθελα να αναφερθώ στη συμβολή της συζύγου μου Δήμητρας, η οποία με καρτερικότητα ανέχθηκε την αφοσίωσή μου στην προσπάθεια αυτή και τις ατελείωτες ώρες απουσίας μου στα εργαστήρια για την ανάπτυξη του λογισμικού προσομοίωσης, που χρησιμοποιήθηκε στη διεξαγωγή ενός σημαντικού αριθμού πειραμάτων.

Θεσσαλονίκη, Δεκέμβριος 2001

Παναγιώτης Θ. Κατσαρός

Περιεχόμενα

Κεφάλαιο 3 Μοντέλα Εκτίμησης Απόδοσης Συστημάτων

Κεφάλαιο 4 Στοχαστική Προσομοίωση Διακριτών Γεγονότων Δικτύων Ουρών

Κεφάλαιο 5 Μοντελοποίηση Απόδοσης Συστημάτων Κατανεμημένης Αρχιτεκτονικής

ΤΟ ΠΡΩΤΟ ΣΚΑΛΙ

Εις τον Θεόκριτο παραπονιούνταν
μια μέρα ο νέος ποιητής Ευμένης·
«Τώρα δυο χρόνια πέρασαν που γράφω
κι' ένα ειδύλλιο έκαμα μονάχα.[...]
Αλλοίμονου, είν' υψηλή το βλέπω,
πολύ υψηλή της Ποιήσεως η σκάλα· [...]».
Είπ' ο Θεόκριτος· «[...]
Κι αν είσαι στο σκαλί το πρώτο, πρέπει
νάσαι υπερήφανος κ' ευτυχισμένος.
Εδώ που έφτασες, λίγο δεν είναι·
Τόσο που έκαμες, μεγάλη δόξα.»

Κ. Π. Καβάφης

Πρόλογος

Η ανάλυση της απόδοσης (performance analysis) υπολογιστικών συστημάτων αποτέλεσε παλαιότερα ένα σημαντικό πεδίο έρευνας και εφαρμογών. Ο βασικός λόγος ήταν το γεγονός ότι το μεγαλύτερο μέρος του κόστους ενός υπολογιστικού συστήματος αντιστοιχούσε στην πρόσκτηση υλικού. Προηγούνταν η καταγραφή των προδιαγραφών εξυπηρέτησης, που έπρεπε να πληροί ένα σύστημα κάτω από έναν αντιπροσωπευτικό υπολογιστικό φόρτο. Ακολούθως έπρεπε να εφαρμοστούν οι κατάλληλες τεχνικές ανάλυσης της απόδοσης των προτεινόμενων συνθέσεων υλικού, έτσι ώστε να βρεθεί αυτή, που με το χαμηλότερο κόστος ανταποκρινόταν στις προδιαγραφές εξυπηρέτησης, που είχαν αρχικά τεθεί.

Σήμερα, οι σύγχρονες ανάγκες επεξεργασίας δεδομένων οδηγούν στην ανάπτυξη κατανεμημένων διατάξεων αρχιτεκτονικής πελάτη - διακομιστή (client - server). Το «σύστημα» πλέον είναι ένα σύνολο τμημάτων υλικού και λογισμικού πιθανώς διαφορετικών τεχνολογιών (ή/και κατασκευαστών), η διασύνδεση των οποίων υποστηρίζεται από σταθμούς εργασίας - μέλη ενός δικτύου. Πολλές φορές δε, η προτεινόμενη αρχιτεκτονική περιλαμβάνει διασύνδεση βασιζόμενη σε δίκτυο ευρείας ζώνης (Wide Area Network).

Σε ένα πλήθος λοιπόν διαφορετικών τύπων και τόσο μεγάλης πολυπλοκότητας αλληλεπιδράσεων, είναι πολύ δύσκολος ο εντοπισμός των *σημείων συμφόρησης* (bottlenecks) ή/και των παραγόντων εκείνων, που επιδρούν καθοριστικά στην απόδοση του συστήματος, με τις υπάρχουσες πρακτικές. Επιπλέον, κατά την ανάπτυξη ενός νέου συστήματος ή μιας νέας υπηρεσίας, συχνά ο πελάτης διαπραγματεύεται και συμφωνεί στην εγγυημένη παροχή υπηρεσιών σε συγκεκριμένα επίπεδα απόδοσης (*Quality of Service*). Έτσι, η ανάλυση της απόδοσης υπολογιστικών συστημάτων σήμερα εξελίσσεται σε ένα νευραλγικό πεδίο έρευνας με πλήθος πρακτικών εφαρμογών.

Σκοπός της μελέτης αυτής ήταν η ανάπτυξη ή/και βελτίωση των τεχνικών εκείνων, που είναι απαραίτητες στην ανάλυση της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής. Στα αποτελέσματα της ερευνητικής προσπάθειας περιλαμβάνονται:

- Η υιοθέτηση μιας ιεραρχικής μοντελικής προσέγγισης, για την ανάπτυξη ρεαλιστικών μοντέλων απόδοσης κατανεμημένου λογισμικού στο επιθυμητό επίπεδο αφαίρεσης.
- Η ανάπτυξη ενός πλαισίου διαδικασιών, που υποστηρίζει τη μοντελοποίηση σε περισσότερα του ενός επίπεδα αφαίρεσης και την εφαρμογή του κατάλληλου τύπου ανάλυσης στο επίπεδο εκείνο, που κρίνεται ως το πλέον πρόσφορο όσον αφορά τις απαιτήσεις κόστους - αξιοπιστίας της κάθε μελέτης απόδοσης.
- Η βελτίωση και η παραλληλοποίηση της προσομοιωτικής τεχνικής της αναγέννησης, που στη μορφή αυτή καθίσταται ελκυστική και για την αποτίμηση μεγάλου μεγέθους μοντέλων απόδοσης με ιδιαίτερα υψηλό υπολογιστικό κόστος. Η συγκεκριμένη μέθοδος είναι και η μοναδική, που προσφέρει την εντυπωσιακή δυνατότητα της *ανάλυσης ευαισθησίας* και *βελτιστοποίησης* με βάση τα αποτελέσματα ενός μόνο πειράματος προσομοίωσης. Η βιβλιογραφική έρευνα, που έγινε, έδειξε ότι η τεχνική της αναγέννησης διαθέτει την απαραίτητη θεωρητική θεμελίωση για χρήση της σε ιεραρχικά μοντέλα, όπως αυτά που

προκύπτουν κατά την ανάλυση της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής.

- Μία πρώτη προσέγγιση των δράσεων προσαρμογής *μεταμοντέλων* (πολυωνυμικών δηλαδή μοντέλων υψηλότερου αφαιρετικού επιπέδου) και ανάλυσης ευαισθησίας και των προβλημάτων, που πρέπει να αντιμετωπιστούν.
- Πλήθος θεμάτων, που αποτελούν το πεδίο εστίασης περαιτέρω ερευνητικών προσπαθειών.

Κεφάλαιο 1 *Εισαγωγή*

Στο κεφάλαιο αυτό γίνεται μία σύντομη εισαγωγή στο αντικείμενο της ανάλυσης απόδοσης συστημάτων κατανεμημένης αρχιτεκτονικής και στη δομή της διατριβής.

1.1 Κατανεμημένα συστήματα αρχιτεκτονικής πελάτη - διακομιστή (client - server)

Ο όρος «αρχιτεκτονική πελάτη - διακομιστή» ή «client - server» εμφανίζεται πλέον στις περισσότερες περιγραφές υπαρκτών ή σχεδιαζόμενων «σύγχρονων πληροφοριακών συστημάτων».

Από τεχνολογικής άποψης, ο όρος αναφέρεται σε μία μορφή διασύνδεσης λογισμικού. Η πλευρά του πελάτη αιτεί και η πλευρά του διακομιστή παρέχει την αιτούμενη υπηρεσία, χωρίς συνήθως να είναι γνωστό στον πελάτη το πως αυτή υλοποιείται.

Από την άποψη των παρεχόμενων υπηρεσιών, είναι συνήθως αδιάφορο σε ποια υπολογιστική μονάδα εκτελούνται οι διεργασίες του πελάτη και του διακομιστή και τι πόρους καταναλώνουν. Συχνά μία διεργασία διακομιστής γίνεται πελάτης μιας άλλης διεργασίας προκειμένου να παράσχει κάποια ζητηθείσα υπηρεσία. Έτσι δημιουργείται μία ευέλικτη κατανεμημένη αρχιτεκτονική καθώς σταδιακά εκλείπει η μονολιθική μορφή, που είχαν οι εφαρμογές.

Ο μετασχηματισμός αυτός του σύγχρονου λογισμικού έγινε εφικτός μέσα από δύο πολύ σημαντικές υπερβάσεις.

Πρώτο εμπόδιο υπήρξε ο τρόπος επικοινωνίας μεταξύ των διεργασιών πελάτη και των διεργασιών διακομιστή. Το σύνολο των κανόνων, που ρυθμίζουν αυτού του είδους την επικοινωνία ονομάστηκε πρωτόκολλο. Η αλληλεπίδραση πελάτη - διακομιστή υλοποιείται με τη χρήση ενός επιλεγμένου συνόλου πρωτοκόλλων, από τα οποία τα πλέον διαδεδομένα είναι αυτά, που συγκροτούν το γνωστό ως TCP/IP (Transmission Control Protocol/Internet Protocol). Έτσι, οποιαδήποτε διασύνδεση TCP/IP μεταξύ μιας διεργασίας πελάτη και κάποιας υπηρεσίας, που παρέχεται σε μία διαφορετική υπολογιστική μονάδα, απαιτεί τα εξής στοιχεία:

- μία IP διεύθυνση πελάτη της μορφής 1.2.3.4,

- μία IP διεύθυνση διακομιστή,
- τη *θύρα* (port) στην οποία «ακούει» η διεργασία - διακομιστής, η οποία μπορεί για παράδειγμα να είναι η θύρα 24,
- την *προσωρινή θύρα* (ephemeral port) της διεργασίας - πελάτη και τέλος,
- το *πρωτόκολλο διεργασιακής επικοινωνίας* (inter-process communication), που στην περίπτωση αυτή είναι το TCP.

Όλη αυτή η πληροφορία περικλείεται σε ένα ή περισσότερα *πακέτα IP*, τα οποία μεταβιβάζονται μέσα από το δίκτυο μαζί με το μήνυμα αίτησης ή τα επιστρεφόμενα δεδομένα. Αυτή, η σε φυσικό επίπεδο διαφάνεια, που προσέφερε η αρχιτεκτονική πελάτη - διακομιστή, αποτέλεσε και την πρώτη σημαντική υπέρβαση, που έδωσε ώθηση στην κατασκευή της πρώτης γενεάς τέτοιων συστημάτων. Μεγάλη άνθηση γνώρισαν τα συστήματα βάσεων δεδομένων με ένα μόνο διακομιστή και πλήθος εφαρμογών - πελάτη, που εξυπηρετούν ταυτόχρονα μεγάλο αριθμό χρηστών.

Παρόλα αυτά και το συγκεκριμένο μοντέλο ανάπτυξης επικρίθηκε για μικρά περιθώρια ευελιξίας, καθώς «συνέτεινε στη δημιουργία όχι ενός αλλά δύο μονολιθικών τμημάτων λογισμικού» ([OHE96]).

Η δεύτερη υπέρβαση έγινε με τη ραγδαία πρόοδο της αντικειμενοστρεφούς τεχνολογίας λογισμικού. Θεμελιώδεις ιδιότητες, όπως η *ενθυλάκωση* (encapsulation), η *κληρονομικότητα* (inheritance) και ο *πολυμορφισμός* (polymorphism), που συνθέτουν μία σε λογικό επίπεδο διαφάνεια στη διασύνδεση λογισμικού, έγινε επιθυμητό και τελικά εφικτό να γίνουν εκμεταλλεύσιμες σε δικτυακό περιβάλλον μέσα από το πρότυπο αλληλεπίδρασης πελάτη - διακομιστή. Στο γεγονός αυτό συνέτειναν επίσης εξελίξεις όπως: 1) η εκθετική αύξηση χαμηλού κόστους *εύρους ζώνης* (bandwidth) σε δίκτυα WAN, όπως το Διαδίκτυο, και 2) η ευρεία διάδοση λειτουργικών συστημάτων, που έκαναν την πολυεπεξεργασία διαθέσιμη ακόμη και σε επίπεδο σταθμών εργασίας χρήστη.

Η μορφή του σύγχρονου λογισμικού λοιπόν αλλάζει ραγδαία. Έχοντας επιτευχθεί υψηλά επίπεδα φυσικής και λογικής διαφάνειας αναπτύσσονται εφαρμογές, που κάνουν χρήση αλληλεπιδρώντων τμημάτων λογισμικού, προερχόμενων πιθανότατα από διαφορετικούς κατασκευαστές. Αυξάνεται έτσι η ευελιξία, η επεκτασιμότητα και η προσαρμοστικότητα των εφαρμογών. Μέσα σε αυτό το πολύπλοκο σκηνικό όμως, τι εγγυήσεις *αξιοπιστίας*, απόδοσης, *διαθεσιμότητας* και *δυνατοτήτων κλιμάκωσης* (scalability) δίνονται;

Στις περισσότερες των περιπτώσεων η απάντηση στο κρίσιμο αυτό ερώτημα είναι αρνητική. Συχνά τα νέα συστήματα έχουν χειρότερους χρόνους απόκρισης από ότι τα παλαιότερα κεντρογενή. Ο βασικότερος λόγος είναι το γεγονός ότι συνήθως δεν πραγματοποιείται από τον κατασκευαστή μία συγκροτημένη μελέτη ή πρόβλεψη της απόδοσης του προς παράδοση συστήματος. Ίσως αυτό θεωρείται ιδιαίτερα δαπανηρό, αμφιβόλου αποτελέσματος ή ακόμη και χρονοβόρο. Μήπως πράγματι οι τεχνικές ανάλυσης της απόδοσης συστημάτων δεν έχουν φθάσει ακόμη στον απαιτούμενο βαθμό ωριμότητας;

1.2 Ανάλυση απόδοσης (Performance Analysis)

Η ανάλυση της απόδοσης ενός συστήματος ασχολείται με τη λειτουργική συμπεριφορά αυτού όσον αφορά κάποιες μετρήσιμες ιδιότητες και τις απαιτήσεις χρήσης πόρων. Επικεντρώνεται βασικά, είτε στον εντοπισμό των σημείων εκείνων του συστήματος, που μπορεί να προκαλέσουν προβλήματα απόδοσης - *σημεία συμφόρησης* (bottlenecks) -, είτε

στην εκτίμηση της *δυναμικότητας* (capacity) εναλλακτικών σχεδιάσεων του συστήματος και τη μεταξύ τους σύγκριση.

Η πρώτη απόπειρα του ανθρώπου να επιλύσει τέτοιου είδους προβλήματα καταγράφεται το 1909 από τον A. K. Erlang, ο οποίος πειραματίστηκε πάνω σε καταστάσεις συνωστισμού σε τηλεφωνικές γραμμές. Η θεωρία, που ανέπτυξε, αποτέλεσε τη βάση της *θεωρίας των ουρών αναμονής*.

Στο χώρο των υπολογιστικών συστημάτων, οι πρώτες προσπάθειες ανάλυσης απόδοσης επικεντρώνονταν στην επιλογή εκείνης της σύνθεσης υλικού, που με το χαμηλότερο κόστος ανταποκρινόταν στις προδιαγραφές εξυπηρέτησης του συστήματος. Συχνά η προσέγγιση του προβλήματος γινόταν με βάση τη συσσωρευμένη εμπειρία και τη διαίσθηση. Παρόλα αυτά, τέτοιου είδους αποφάσεις δεν ήταν δυνατό να επαληθευθούν παρά μόνο από τα αποτελέσματα των επιλογών, όταν δηλαδή το σύστημα είχε ήδη στηθεί και άρα ήταν ήδη αργά για να ανατραπούν. Από την άλλη μεριά, κάθε απόπειρα εκτίμησης των εναλλακτικών λύσεων με τη διενέργεια συγκριτικών πειραμάτων, είχε στις περισσότερες περιπτώσεις απαγορευτικό κόστος. Ακόμη όμως και αν αυτό το πρόβλημα ήταν εφικτό να ξεπεραστεί, τα αποτελέσματα της πειραματικής εκτίμησης είχαν ισχύ μόνο για το συγκεκριμένο υπολογιστικό φόρτο, που χρησιμοποιούνταν, και ήταν πολύ δύσκολο να γενικευθούν και σε άλλες περιπτώσεις.

Σύντομα έγινε αντιληπτή η ανάγκη κατασκευής *μοντέλων* (modeling) και εξαγωγής αποτελεσμάτων μέσα από την μελέτη αυτών. Μία πρώτη βάση αποτέλεσαν οι θεωρίες των ουρών και των *δικτύων ουρών αναμονής* (Queuing Networks). Τα μοντέλα αυτά προσέφεραν έναν αποτελεσματικό τρόπο αναπαράστασης του *ανταγωνισμού* (contention) για τη διανομή των πόρων ενός συστήματος. Με την πάροδο των ετών προτάθηκαν νέα μοντέλα, όπως τα *δίκτυα Petri* (Petri nets) και τα βασιζόμενα σε αυτά, τα οποία είχαν τη δυνατότητα αναπαράστασης του συγχρονισμού παράλληλων επεξεργασιών. Μία επαρκής αναφορά στα πιο διαδεδομένα από τα μοντέλα απόδοσης, που χρησιμοποιούνται, γίνεται στο κεφάλαιο 3 αυτής της διατριβής. Παρουσιάζεται η αναλυτική και η προσεγγιστική επίλυση αυτών, καθώς και δύο περιπτώσεις μοντέλων λογισμικού κατανεμημένης αρχιτεκτονικής.

Η αποκλειστική χρήση αναλυτικών μεθόδων αποτίμησης προϋποθέτει την αποδοχή απλουστευτικών συμβιβασμών, έτσι ώστε να αποφεύγεται η δημιουργία μη επιλύσιμων μοντέλων. Με τη διαρκώς αυξανόμενη διαθεσιμότητα ολοένα και πιο φθηνής υπολογιστικής ισχύος έγινε εφικτή η επίλυση μοντέλων απόδοσης με τη χρήση *προσομοίωσης* (simulation). Παρόλα αυτά, τον πρώτο καιρό στα αποτελέσματα είναι αλήθεια ότι δεν εφαρμοζόταν η απαραίτητη στατιστική επεξεργασία. Στις περισσότερες των περιπτώσεων αγνοούνταν το γεγονός ότι μία προσομοίωση είναι βασικά ένα τυχαίο γεγονός και άρα το αποτέλεσμα μιας μόνο εκτέλεσης αυτής δεν μπορεί από μόνο του να είναι αρκετά αξιόπιστο. Στο κεφάλαιο 4 παρουσιάζονται κάποιες από τις πιο προωθημένες τεχνικές προσομοίωσης, οι οποίες μελετήθηκαν σε βάθος, τόσο όσον αφορά την αποτελεσματικότητα, όσο και τη δυνατότητα εφαρμογής αυτών σε πολύπλοκα μοντέλα δικτύων ουρών. Επίσης, παρουσιάζονται αξιόλογα ερευνητικά αποτελέσματα σχετικά με την αξιοποίηση των τεχνικών αυτών σε παράλληλες/κατανεμημένες διατάξεις προσομοίωσης για την επίτευξη υποπολλαπλάσιων χρόνων εκτέλεσης και ελεγχόμενης στατιστικής ακρίβειας.

Στο κεφάλαιο 5 περιγράφεται η μοντελική προσέγγιση, που επιλέχθηκε ως η πλέον κατάλληλη για την ανάπτυξη μοντέλων απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής. Παρουσιάζεται επίσης ένα πλαίσιο διαδικασιών, χαρακτηριστικό του οποίου είναι η δυνατότητα περιγραφής της απόδοσης του λογισμικού σε διαφορετικά επίπεδα αφαίρεσης. Τέλος, γίνεται χρήση της μοντελικής προσέγγισης, που επιλέχθηκε, για την περιγραφή περιπτώσεων σχεδίασης κατανεμημένου λογισμικού.

Στο κεφάλαιο 6 γίνεται σύντομη αναφορά σε μεγάλης πρακτικής σημασίας εργαλεία της στατιστικής για την προσαρμογή μεταμοντέλων (πολυωνυμικών δηλαδή μοντέλων υψηλότερου αφαιρετικού επιπέδου) και τη διεξαγωγή διαφόρων τύπων ανάλυσης ευαισθησίας. Επίσης, περιγράφεται μία ολοκληρωμένη εφαρμογή χρήσης αυτών, που μπορεί να αποτελέσει μία αρχική βάση για την ανάπτυξη ενός πλαισίου δράσεων για την ασφαλή και αποτελεσματική χρήση τους.

Η πρώτη αξιόλογη προσέγγιση του προβλήματος της απόδοσης λογισμικού έγινε από την Connie Smith [SMI90] το 1990. Στο έργο αυτό μάλιστα, κάνει λόγο για την ανάγκη μιας συστηματικής θεμελίωσης του αντικειμένου πάνω στη βάση μιας επιστήμης μηχανικού (performance engineering), με ότι αυτό συνεπάγεται:

- σαφείς περιγραφές της αλληλουχίας των διαδικασιών, οι οποίες συγκροτούν το έργο,
- ξεκάθαρο ορισμό των παραδοτέων των φάσεων εκτέλεσης αυτού και τέλος,
- κριτήρια ελέγχου και μέτρησης των προϊόντων και των διαδικασιών.

Πράγματι η Smith θίγει μία καίρια αδυναμία του χώρου, καθώς ακόμη και σήμερα είναι διάχυτη η εντύπωση ότι οι τεχνικές μελέτης της απόδοσης συστημάτων δεν έχουν ακόμη προσεγγίσει τον απαιτούμενο βαθμό ωριμότητας για την ενσωμάτωσή τους στον κύκλο ζωής του λογισμικού.

Παρόλα αυτά και παρά το γεγονός ότι το έργο της Smith κινείται προς την κατεύθυνση μιας συγκροτημένης προσέγγισης με μία σαφώς διατυπωμένη μεθοδολογία και σχετικά απλές στην εφαρμογή τους διαδικασίες, η προσπάθειά της δε βρήκε την απήχηση, που κανείς θα περίμενε. Στο κεφάλαιο 7 γίνεται μία σύντομη παρουσίαση της συγκεκριμένης προσέγγισης στη νεώτερη έκδοση, αυτή δηλαδή, που αφορά την ανάλυση της απόδοσης (κατανεμημένου) αντικειμενοστρεφούς λογισμικού και περιγράφεται στο [SW01].

Κεφάλαιο 2 *Λογισμικό Κατανεμημένης Αρχιτεκτονικής*

Στο κεφάλαιο αυτό γίνεται περιγραφή των διαφόρων τύπων λογισμικού κατανεμημένης αρχιτεκτονικής και των τεχνολογιών, που υποστηρίζουν την ανάπτυξή του.

2.1 Αρχιτεκτονική πληροφοριακού συστήματος

Η αρχιτεκτονική ενός πληροφοριακού συστήματος περιγράφει τις τεχνολογίες, που αυτό χρησιμοποιεί, και τον τρόπο με τον οποίο οι συγκεκριμένες τεχνολογίες συλλειτουργούν.

Η συνεχής εμφάνιση και επιδίωξη εφαρμογής νέων τεχνολογιών είναι το αποτέλεσμα της χρήσης νέων βελτιωμένων επιχειρηματικών πρακτικών για τη διατήρηση ανταγωνιστικού πλεονεκτήματος ή για περισσότερη αποτελεσματικότητα και μεγαλύτερη κερδοφορία. Πολλές τέτοιες πρωτοβουλίες περιλαμβάνουν αλλαγές στις υφιστάμενες δομές διαφόρων επιχειρηματικών δραστηριοτήτων. Έτσι παρατηρείται το φαινόμενο να δίνεται ολοένα και μεγαλύτερη βαρύτητα στους πελάτες και κατά συνέπεια στους πληροφοριακούς πόρους και τις υπηρεσίες, που απαιτούνται για την υποστήριξη των συναλλαγών με τους πελάτες. Ο όρος, που αναφέρεται στην προσπάθεια ολοκλήρωσης και ευθυγράμμισης των διαθέσιμων πληροφοριακών πόρων με τις υπαρκτές επιχειρηματικές ανάγκες, είναι γνωστός ως *αναδιάρθρωση επιχειρηματικών πρακτικών* (Business Process Reengineering). Τυπικό χαρακτηριστικό αυτής της μετάλλαξης αποτελεί το πρότυπο αλληλεπίδρασης πελάτη - διακομιστή και αυτό γιατί σε αντίθεση με τους περιορισμούς, που παρατηρούνται στα κεντρογενή συστήματα, το πρότυπο αυτό αντιπροσωπεύει την ευελιξία και την προσαρμοστικότητα στις μεταβαλλόμενες επιχειρηματικές συνθήκες.

Έτσι, η μετάβαση αυτή προς την κατεύθυνση του ολοένα και περισσότερο κατανεμημένου πληροφοριακού συστήματος υποκινείται από καθαρά επιχειρηματικούς λόγους και παρά τη γενικά αποδεκτή άποψη ότι τα συστήματα αυτά είναι πιο περίπλοκα στην κατασκευή και έχουν πιο ακριβή συντήρηση από ότι τα προηγούμενα κεντρογενή. Η σύγχρονη επιχείρηση είναι ήδη κατανεμημένη και γι΄ αυτό χρειάζεται ένα κατανεμημένο πληροφοριακό σύστημα για την υποστήριξη των δραστηριοτήτων της.

Σε ένα κατανεμημένο περιβάλλον είναι πολλοί οι παράγοντες από τους οποίους εξαρτάται η συνολική απόδοση μιας εφαρμογής. Ένα πληροφοριακό δίκτυο μπορεί να αποτελείται από πολλά υποδίκτυα με διαφορετικές ταχύτητες. Επιπλέον των κεντρικών μονάδων διακομιστών

το δίκτυο μπορεί να διασυνδέει πολλές κατανεμημένες μονάδες διακομιστών, κάθε μία με τις δικές της συσκευές δίσκων και πιθανότατα πολλούς διαφορετικούς επεξεργαστές και δίσκους με δυνατότητα επιλογής. Σε ένα τέτοιο περιβάλλον προκύπτουν σχεδιαστικά ζητήματα, που δεν υπάρχουν, όταν όλα τα δεδομένα αποθηκεύονται και επεξεργάζονται σε μία μόνο κεντρική υπολογιστική μονάδα. Πιο συγκεκριμένα για την κατανομή δεδομένων και διεργασιών πρέπει να αποφασίσουμε:

- Πως να διαχωρίσουμε τα δεδομένα και που να αποθηκεύσουμε τα τμήματα της βάσης. Η δραστηριότητα αυτή λέγεται κατανομή δεδομένων (data distribution, partitioning, fragmentation, declustering).
- Πως να διατηρήσουμε τη συνέπεια μεταξύ αλληλοσχετιζόμενων ή πολλαπλών δεδομένων, αποθηκευμένων σε βάσεις διαφορετικών υπολογιστικών μονάδων. Αυτή η δραστηριότητα έχει σχέση με την επεξεργασία *διεκπεραιώσεων* (transaction processing) και πιθανότατα κάνει χρήση τεχνολογιών κατανεμημένων βάσεων και επανάληψης δεδομένων (data replication).
- Πως να διαχωρίσουμε τη λογική της εφαρμογής και που να τοποθετήσουμε την επεξεργασία των διαφόρων τμημάτων αυτής. Η δραστηριότητα αυτή ονομάζεται *διαχωρισμός εφαρμογής* (application partitioning).
- Πως να φέρουμε σε επαφή κάποιο υποσύνολο δεδομένων και μία διεργασία, η οποία πρέπει να επιδράσει σε αυτά. Η συγκεκριμένη δραστηριότητα ονομάζεται *μεταφορά δεδομένων* (data shipping), όταν μετακινούμε δεδομένα, ή *μεταφορά λειτουργιών* (function shipping), όταν μετακινούμε τη διεργασία στα δεδομένα - στόχος.
- Πως διαφορετικές διεργασίες, που εκτελούνται σε διαφορετικές υπολογιστικές μονάδες, μπορούν να αλληλεπιδρούν. Ο γενικός όρος, με τον οποίο αναφερόμαστε στη δραστηριότητα αυτή, λέγεται *διεργασιακή επικοινωνία* (interprocess communication) και η τεχνολογική υποδομή, που την υποστηρίζει, παρέχεται από κάποια προϊόντα λογισμικού, που ανήκουν σε μία κατηγορία γνωστή με τον όρο *middleware*.

Από τα παραπάνω είναι σαφές ότι η ανάπτυξη ενός πληροφοριακού συστήματος κατανεμημένης επεξεργασίας είναι ένα ιδιαίτερα πολυσύνθετο έργο. Ο συνδυασμός δικτύων, διακομιστών διαφόρων μεγεθών, middleware, συστημάτων διαχείρισης βάσεων δεδομένων και λειτουργικών συστημάτων και το γεγονός ότι όλα αυτά δημιουργούν διαφορετικές απαιτήσεις σε πόρους υλικού και λογισμικού μπορεί να οδηγήσει σε ένα περίπλοκο σύμπλεγμα σημείων συμφόρησης. Στην πραγματικότητα, όσο περισσότερο κατανέμονται τα τμήματα των εφαρμογών μας, τόσο περισσότερες πιθανότητες δημιουργίας προβλημάτων απόδοσης υπάρχουν.

Η αρχιτεκτονική ενός συστήματος είναι μία υψηλού επιπέδου περιγραφή, η οποία επικεντρώνεται σε κάποια προεξάρχοντα χαρακτηριστικά αυτού, που η εμπειρία μας έχει δείξει ότι είναι περισσότερο σημαντικά και έχουν μακρόπνοη επίδραση στον κύκλο ζωής του συστήματος. Η βασική διαφορά του πελάτη από το διακομιστή, είναι το γεγονός ότι συνήθως ο πελάτης δρα για λογαριασμό κάποιου χρήστη. Τι γίνεται όμως στην περίπτωση, κατά την οποία οι υπολογιστικές μονάδες είναι προγραμματισμένες να αλληλεπιδρούν, έτσι ώστε ένα σύνολο τμημάτων λογισμικού να παράσχει κάποιες υπηρεσίες σε άλλα τμήματα λογισμικού; Στην περίπτωση αυτή ο πελάτης και ο διακομιστής καθίστανται ρόλοι και όχι συσκευές με φυσική υπόσταση. Γενικά η αρχιτεκτονική ενός πληροφοριακού συστήματος είναι ευκολότερο να περιγραφεί σε σχέση με τα λογικά τμήματα αυτού, παρά από τη φυσική του διαμόρφωση, η οποία μπορεί άλλωστε να διαφοροποιείται με την πάροδο του χρόνου.

Η ομαδοποίηση των λειτουργικών τμημάτων μιας εφαρμογής σε «στρώματα» είναι μία διαδεδομένη πρακτική, η οποία παρέχει μία βάση για τη γεωγραφική κατανομή αυτής. Σήμερα υπάρχουν κάποιες καλά θεμελιωμένες αρχές για τον καθορισμό του στρώματος, στο οποίο ανήκει η κάθε λειτουργία της εφαρμογής. Μία πρώτη προσέγγιση του προβλήματος είναι αυτή του [BER92], η οποία διαχωρίζει την εφαρμογή σε τρία λογικά στρώματα:

- τις λειτουργίες εμφάνισης (presentation functions),
- το λειτουργικό πυρήνα (business logic functions) και
- τις λειτουργίες διαχείρισης δεδομένων (data management functions).

Μία πιο αναλυτική προσέγγιση είναι αυτή του [WIN93], που συνοψίζεται στον πίνακα 2.1.

Πίνακας 2.1 Λειτουργικά στρώματα κατανεμημένων εφαρμογών

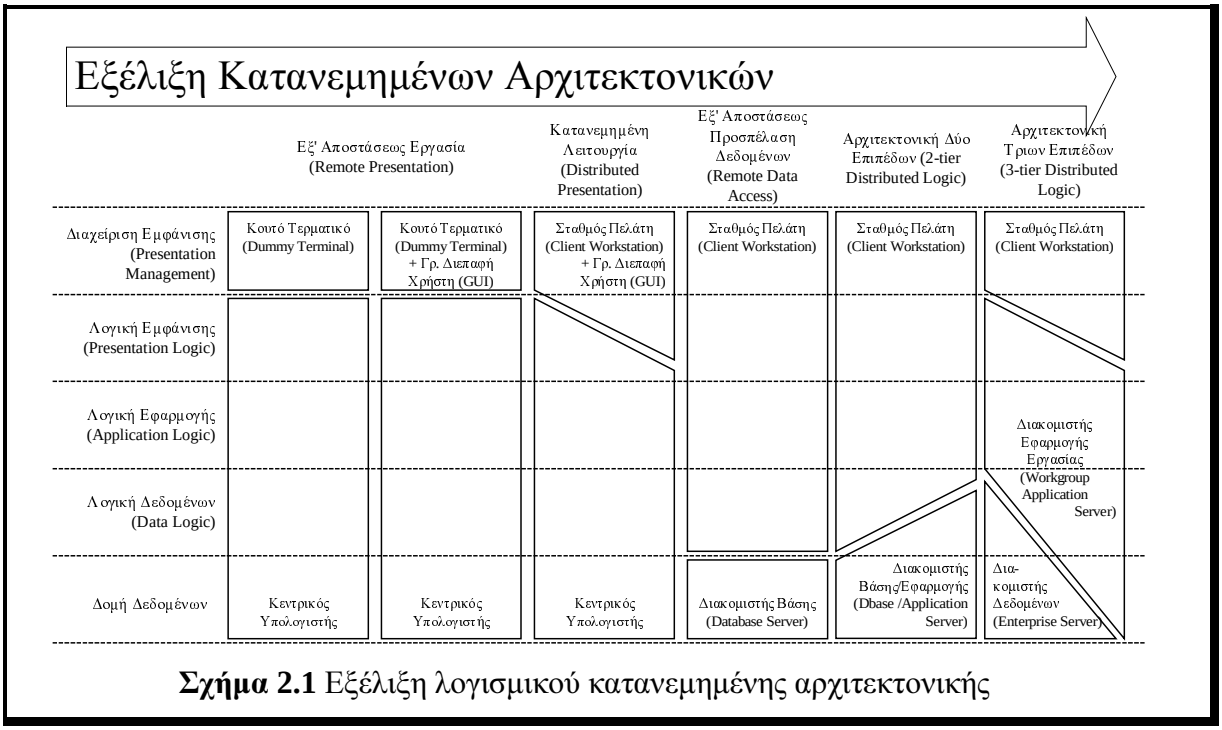
Διεπαφή χρήστη (user interface)	Διαχείριση εμφάνισης (presentation management)	Ελέγχει τη διάταξη των στοιχείων του γραφικού περιβάλλοντος χρήσης.
	Λογική εμφάνισης (presentation logic)	Όλη η λογική διαχείρισης της μορφής και του περιεχομένου των παραθύρων καθώς και της αλληλεπίδρασης με το χρήστη.
Λογική εφαρμογής	Λογική εφαρμογής	Οι λειτουργίες της εφαρμογής, που δεν ανήκουν στα άλλα στρώματα. Εδώ αναπτύσσεται ο κώδικας, που υλοποιεί τις υπηρεσίες, οι οποίες παρέχονται από το πληροφοριακό σύστημα.
Διαχείριση δεδομένων	Λογική δεδομένων	Όλη η λογική, που σχετίζεται με την αποθήκευση και την ανάσυρση των δεδομένων, και οι πολιτικές, που εφαρμόζονται για τη συνέπεια αυτών.
	Διαχείριση βάσεων δεδομένων	Αποθήκευση, ανάσυρση (retrieval), διαχείριση και ανάκτηση (recovery) παγίων (persistent) δεδομένων.

Με βάση το λογικό διαχωρισμό, που παρουσιάζεται στον πίνακα 2.1, η εξέλιξη της αρχιτεκτονικής των πληροφοριακών συστημάτων, όπως αυτή περιγράφεται στο [LD98] και φαίνεται στο σχήμα 2.1, έχει ως εξής:

- Αρχικά τα απλά συστήματα εξ΄ αποστάσεως εργασίας (remote presentation) βασίστηκαν σε ένα κεντρικό υπολογιστή και σε ένα σύνολο «κουτών» τερματικών (dummy terminals).
- Καθώς οι υπολογιστικές δυνατότητες των τερματικών αυξάνονταν, έκαναν την εμφάνισή τους σε αυτά λειτουργίες γραφικής διεπαφής χρήστη. Σε πολλές περιπτώσεις, ένα μέρος της λογικής εμφάνισης μεταφέρθηκε από τον κεντρικό υπολογιστή σε αυτό, που δεν ήταν πλέον ένα «κουτό» τερματικό αλλά ένας σταθμός εργασίας (workstation).
- Η εμφάνιση των X-τερματικών και του περιβάλλοντος Motif έδωσαν νέα ώθηση στο επίπεδο της λογικής της εμφάνισης, καθώς εκτός των άλλων παρουσιάστηκαν και νέα πρότυπα γραφικής διεπαφής.
- Οι πρώτες βάσεις δεδομένων τύπου πελάτη - διακομιστή, όπως η Sybase, η Oracle και η Informix, παρείχαν μόνο λειτουργίες διαχείρισης δεδομένων. Όλη η

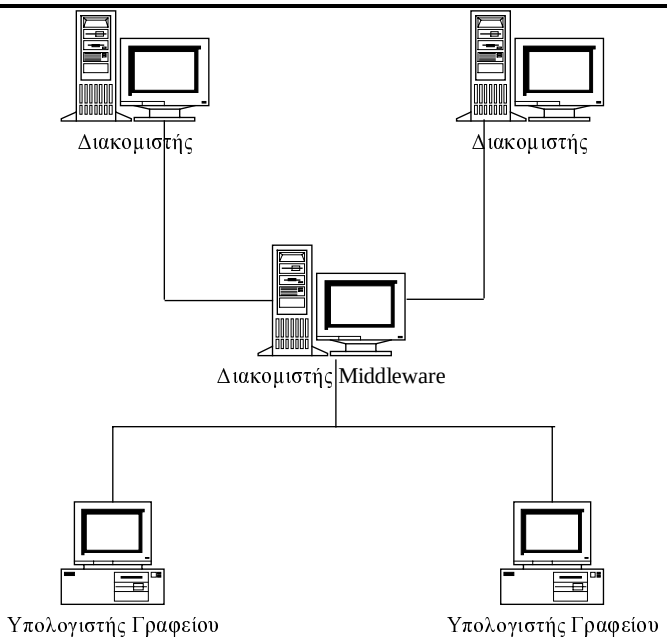
υπόλοιπη λειτουργικότητα της εφαρμογής βρισκόταν στο σταθμό εργασίας. Αυτού του τύπου η διάταξη χαρακτηρίσθηκε ως εξ΄ αποστάσεως προσπέλαση δεδομένων (remote data access).

- Βαθμιαία, τα προϊόντα βάσεων τύπου πελάτη - διακομιστή άρχισαν να ενσωματώνουν μέρος της λειτουργικότητας της εφαρμογής, που βρισκόταν στο σταθμό εργασίας, στο διακομιστή. Αυτό γινόταν μέσω της χρήσης triggers, stored procedures και άλλων ευκολιών. Το συγκεκριμένο πρότυπο επεξεργασίας έγινε γνωστό ως *αρχιτεκτονική δύο επιπέδων* (2-tier architecture).
- Τέλος, η αρχιτεκτονική δύο επιπέδων μετεξελίχθηκε σε *αρχιτεκτονική τριών επιπέδων* (3-tier architecture). Καταλύτης σε αυτή την εξελικτική διαδικασία ήταν η εμφάνιση μιας νέας γενιάς λογισμικού *middleware* για συστήματα πελάτη - διακομιστή. Με τον όρο middleware αναφερόμαστε σε όλα εκείνα τα προϊόντα λογισμικού, που επιτρέπουν τη διασύνδεση διαφορετικών στρωμάτων μιας εφαρμογής ή διαφορετικών τμημάτων αυτής σε ένα ενιαίο πληροφοριακό σύστημα κατανεμημένης αρχιτεκτονικής (σχήμα 2.2).



Σχήμα 2.1 Εξέλιξη λογισμικού κατανεμημένης αρχιτεκτονικής

Μία ειδικής μορφής αρχιτεκτονική πελάτη - διακομιστή είναι η βασιζόμενη σε *διακομιστή ιστοσελίδων*. Η αρχιτεκτονική αυτή αποτελείται από ένα διακομιστή ιστοσελίδων και από προγράμματα *φυλλομετρήματος ιστοσελίδων* (browsers), που λειτουργούν ως πελάτες. Ο ρόλος της είναι ιδιαίτερα σημαντικός για τα εσωτερικά δίκτυα (intranets) μεγάλων οργανισμών, όπου υποστηρίζει λειτουργίες διαχείρισης έργου, παρακολούθησης ροής εγγράφων, εκπαίδευσης κ.α. Η διάδοση όμως της συγκεκριμένης αρχιτεκτονικής οφείλεται κυρίως στον Παγκόσμιο Ιστό Πληροφοριών και στο γεγονός ότι η περιήγηση μέσα στις ιστοσελίδες δεν απαιτεί ιδιαίτερες γνώσεις ή μεγάλο βαθμό εξοικείωσης με τη χρήση των ηλεκτρονικών υπολογιστών. Το γεγονός αυτό ώθησε τους περισσότερους επιχειρηματικούς ομίλους στην ανάπτυξη εκτεταμένων *δικτυακών τόπων* με πληροφορίες προϊόντων, πωλήσεις λιανικής, υποστήριξη πελατών κ.α. Άνοιξε έτσι ο δρόμος για την εμφάνιση τεχνολογιών, που διευκολύνουν την ανάπτυξη υπηρεσιών σαν αυτές, στις οποίες αναφερόμαστε πλέον με όρους, όπως ηλεκτρονικό εμπόριο (e-commerce), ηλεκτρονική εκπαίδευση (e-learning) κ.λ.π.

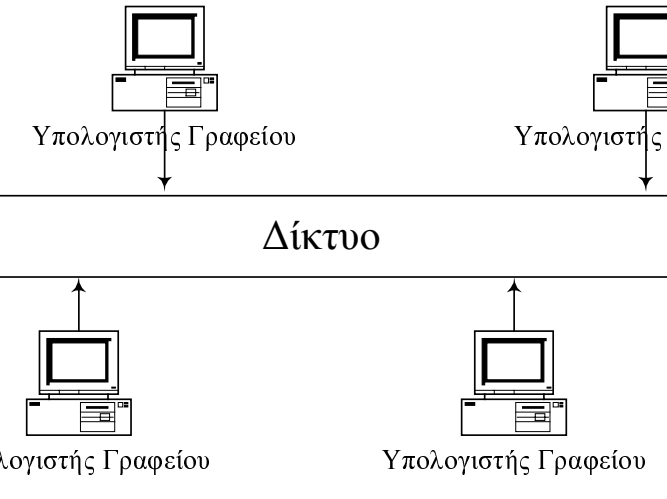


Σχήμα 2.2 Σύστημα αρχιτεκτονικής τριών επιπέδων

Συνοψίζοντας, πληροφοριακό σύστημα κατανεμημένης αρχιτεκτονικής είναι κάθε σύστημα, που περιλαμβάνει έναν αριθμό διακομιστών και κατά συνέπεια κάθε σύστημα της μορφής πελάτη - διακομιστή.

Έτσι, ένα σύστημα τριών επιπέδων είναι σύστημα κατανεμημένης αρχιτεκτονικής, αφού αυτό περιλαμβάνει έναν ή περισσότερους διακομιστές. Στην πραγματικότητα αυτός είναι και ο λόγος ύπαρξης του μεσαίου επιπέδου, η διευκόλυνση δηλαδή της πρόσβασης σε ένα ετερογενές σύνολο διακομιστών. Βέβαια στα συστήματα αυτού του τύπου οι πελάτες γενικά συνδέονται σε ένα μόνο διακομιστή, για μία δεδομένη χρονική στιγμή. Ένα αληθινά κατανεμημένο σύστημα περιλαμβάνει δυνατότητες σύγχρονης σύνδεσης σε περισσότερους του ενός διακομιστές και μία τέτοια αρχιτεκτονική είναι τα *ομότιμα δίκτυα* (peer-to-peer networks), όπου κάθε σταθμός είναι εν δυνάμει και πελάτης και διακομιστής (σχήμα 2.3).

Στη διατριβή αυτή, με τον όρο *κατανεμημένη αρχιτεκτονική* θα αναφερόμαστε σε όλες τις περιπτώσεις συστημάτων πελάτη - διακομιστή.



Σχήμα 2.3 Ομότιμο δίκτυο (peer-to-peer network)

2.2 Λογισμικό κατανεμημένων αντικειμένων (Distributed Object Systems)

Τα αντικείμενα είναι μονάδες λογισμικού, οι οποίες ενθυλακώνουν τα δεδομένα που αναφέρονται σε αυτές και τις λειτουργίες που επηρεάζουν τα συγκεκριμένα δεδομένα. Τα αντικείμενα, τα οποία βρίσκονται εκτός της υπολογιστικής μονάδας αναφοράς, ονομάζονται *απομακρυσμένα αντικείμενα* και το λογισμικό, που περιλαμβάνει τέτοια ονομάζεται *λογισμικό κατανεμημένων αντικειμένων*.

Η λέξη «κατανεμημένα» εκφράζει γεωγραφική κατανομή ή διασκορπισμό. Μία κατανεμημένη εφαρμογή λειτουργεί σε έναν αριθμό *κόμβων*, που εκτελούν υπολογιστικό έργο. Οι κόμβοι αυτοί μπορεί να είναι προσωπικοί ή μεγάλοι υπολογιστές ή άλλου είδους συσκευές. Όταν λοιπόν αναφερόμαστε στον κόμβο ενός συγκεκριμένου αντικειμένου, αυτός χαρακτηρίζεται ως *τοπικός κόμβος*, ενώ όλοι οι υπόλοιποι κόμβοι εκτέλεσης της εφαρμογής αποκαλούνται *απομακρυσμένοι κόμβοι*.

Η ποιοτική διαφοροποίηση, που φέρνει η αντικειμενοστρεφής τεχνολογία, μπορεί να συνοψιστεί σε μία και μοναδική λέξη: *διασύνδεση* (interface). Πρόκειται για ένα μέσο χειρισμού και αλληλεπίδρασης με το αντικείμενο. Έτσι, αν για παράδειγμα επιθυμούμε να χρησιμοποιήσουμε ένα αντικείμενο αλφαβητικής ταξινόμησης, τότε δε χρειάζεται να γνωρίζουμε το πως αυτό είναι κατασκευασμένο, ποιους αλγορίθμους και τι άλλα δεδομένα χρησιμοποιεί. Αρκεί μόνο να γνωρίζουμε τη διασύνδεση του συγκεκριμένου αντικειμένου και τίποτε περισσότερο. Εξάλλου αυτός είναι και ο τρόπος, με τον οποίο έχουμε μάθει να λειτουργούμε στην καθημερινή μας ζωή. Για να δούμε τηλεόραση δε χρειάζεται να γνωρίζουμε κάθε λεπτομέρεια για τα ηλεκτρονικά κυκλώματα, που βρίσκονται μέσα σε αυτή, αλλά αρκεί να γνωρίζουμε τη χρήση της διασύνδεσης, δηλαδή του τηλεχειριστηρίου. Μέχρι πριν από την έλευση των αντιεκιμενοστρεφών τεχνολογιών προγραμματισμού οι μηχανικοί λογισμικού ήταν υποχρεωμένοι να αντιμετωπίζουν καταστάσεις, ανάλογες αυτής, στην εκτέλεση του έργου τους.

Θεμελιώδεις ιδιότητες των αντικειμενοστρεφών τεχνολογιών, όπως ο *πολυμορφισμός*, η *κληρονομικότητα* και η *ενθυλάκωση*, αποβλέπουν τελικά στην παροχή πιο ευέλικτων και εύκολων στη χρήση διασυνδέσεων. Αυτές οι καλύτερες διασυνδέσεις λοιπόν, οδηγούν με τη σειρά τους σε δύο πολύ σημαντικές εξελίξεις: την *ανταλλαξιμότητα* (interchangeability) και την *αλληλοδραστικότητα* (interoperability).

Σήμερα γινόμαστε μάρτυρες της δημιουργίας νέων αγορών διάθεσης τμημάτων λογισμικού, όπως ορθογράφοι, τμήματα διασύνδεσης βάσεων δεδομένων, τμήματα διασύνδεσης με τον παγκόσμιο ιστό κ.α. Έτσι, είναι πλέον σύνηθες μέσα από ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment) να «ρίχνουμε» μέσα στην εφαρμογή, που κατασκευάζουμε, ένα έτοιμο τμήμα π.χ. ζωγραφικής, αντί να το κατασκευάζουμε από την αρχή. Όλες οι δυνατές ρυθμίσεις μπορούν με διάφανο τρόπο να προσαρμοσθούν στις απαιτήσεις της εφαρμογής, καθώς η τυποποιημένη διασύνδεση των τμημάτων, που είναι δυνατό να χρησιμοποιηθούν, τα καθιστά ανταλλάξιμα.

Οι τυποποιημένες διασυνδέσεις όμως, προάγουν και την αλληλοδραστικότητα, τη δυνατότητα δηλαδή των τμημάτων λογισμικού να αλληλεπιδρούν μεταξύ τους. Τμήματα λογισμικού από διαφορετικούς κατασκευαστές μπορούν πλέον να τοποθετηθούν σε ένα *δίαυλο αντικειμένων* (object bus), ο οποίος τους επιτρέπει να ανταλλάσσουν δεδομένα. Μπορούμε έτσι να κατασκευάσουμε ολόκληρα συστήματα από τμήματα λογισμικού, τα οποία ποτέ στο παρελθόν δεν είχαν συνυπάρξει.

Θεμελιώδεις έννοιες αντικειμενοστρεφών τεχνολογιών λογισμικού

Κλάση: *Μία συλλογή ομοίων αντικειμένων. Η κλάση λειτουργεί ως ένα πρότυπο για όλα αυτά τα αντικείμενα. Έτσι, κάθε αντικείμενο είναι μία περίπτωση (instance) της κλάσης, στην οποία ανήκει.*

Κληρονομικότητα: *Είναι η πρόσκτηση από τα αντικείμενα, ιδιοτήτων άλλων αντικειμένων. Τεχνικά, η κληρονομικότητα επιτρέπει τη δημιουργία υποκλάσεων. Οι υποκλάσεις κληρονομούν τις ιδιότητες των γονικών κλάσεων. Σε ορισμένες τεχνολογίες δίνεται η δυνατότητα κληροδότησης ιδιοτήτων, περισσοτέρων της μιας γονικών κλάσεων.*

Ενθυλάκωση: *Είναι ο περιορισμός της προσπέλασης στις πληροφορίες κατάστασης (state) του αντικειμένου μέσα από μία σαφώς καθορισμένη διασύνδεση. Έτσι λέμε ότι ένα αντικείμενο ενθυλακώνει τις πληροφορίες κατάστασης και τις λειτουργίες, οι οποίες παρέχουν πρόσβαση και είναι δυνατό να επηρεάσουν αυτές.*

Αφαίρεση: *Είναι η δυνατότητα ομαδοποίησης οντοτήτων μέσα από κάποιες κοινές ιδιότητες. Η κλάση είναι για παράδειγμα μία αφαίρεση των περιπτώσεων, που αυτή περιλαμβάνει.*

Πολυμορφισμός: *Είναι η δυνατότητα επικαλύψεων των ορισθέντων αφαιρέσεων. Ένα παράδειγμα πολυμορφισμού είναι όταν οι λειτουργίες κάποιας συγκεκριμένης κλάσης μπορούν επίσης να εφαρμοσθούν σε κάποια υποκλάση αυτής (για παράδειγμα η λειτουργία «έναρξη εκτυπωτή» μπορεί να εφαρμοστεί σε όλες τις υποκλάσεις εκτυπωτών). Η μορφή αυτή του πολυμορφισμού συχνά υλοποιείται μέσα από ένα μηχανισμό κληρονομικότητας.*

Έτσι, τα αντικείμενα εκ φύσεως προάγουν τις δυνατότητες: i) συνύπαρξης - αλληλεπίδρασης μέσα σε ένα πληροφοριακό δίκτυο, ii) εκτέλεσης σε ετερογενείς πλατφόρμες, iii) συνεργασίας με τις παλαιές μονολιθικές εφαρμογές με τη χρήση διασυνδέσεων - *περιτυλίξεων* (object wrappers) και επίσης μπορούν iv) να περιπλανώνται μέσα στο δίκτυο και να διαχειρίζονται τους εαυτούς τους και τους πόρους, που ελέγχουν και χρησιμοποιούν. Τα αντικείμενα δηλαδή καθίστανται εν δυνάμει αυτοδιαχειριζόμενες μονάδες λογισμικού.

2.3 Εισαγωγή στο Middleware

Middleware είναι το σύνολο εκείνο των υπηρεσιών, οι οποίες δε σχετίζονται με το σκοπό για τον οποίο κατασκευάζεται ένα σύστημα, αλλά δίνουν τη δυνατότητα στις εφαρμογές και στους τελικούς χρήστες να αλληλεπιδρούν μέσα από το δίκτυο. Στην ουσία δηλαδή, middleware είναι το λογισμικό, το οποίο βρίσκεται ένα επίπεδο πάνω από το δίκτυο και ένα επίπεδο κάτω από το λογισμικό εφαρμογών.

Οι υπηρεσίες, που παρέχει το middleware, είναι διαθέσιμες μέσω ενός συνόλου ρουτινών. Όπως ήδη τονίστηκε, δεν έχουν καμία σχέση με τη λογική των εφαρμογών, αλλά διατίθενται για χρήση από αυτές με τη μορφή *προγραμματιστικών διασυνδέσεων εφαρμογών* (Application Programming Interfaces). Οι τελικοί χρήστες έχουν πρόσβαση στις υπηρεσίες του

middleware, είτε μέσα από περιβάλλοντα ελέγχου με τη χρήση συγκεκριμένων εντολών, είτε μέσα από κατάλληλες γραφικές διεπαφές.

Το middleware επίσης είναι το λογισμικό, το οποίο δίνει τη δυνατότητα ανταλλαγής πληροφοριών μέσα από το δίκτυο. Στην ιδανική περίπτωση δε, καθιστά την ύπαρξη του δικτύου μη σημαντική για τις εφαρμογές και τους χρήστες, δηλαδή με άλλα λόγια αυτοί μπορούν να λειτουργήσουν μέσα στο δίκτυο, όπως ακριβώς λειτουργούν και στο κόμβο τους. Αυτό σημαίνει ότι το εν λόγω λογισμικό κρύβει τις λεπτομέρειες και τις διαφορές του χρησιμοποιούμενου υλικού, των λειτουργικών συστημάτων και άλλου λογισμικού όπως π.χ. των συστημάτων διαχείρισης βάσεων δεδομένων.

Επίσης, σύμφωνα με τον ορισμό του λογισμικού middleware, αυτό βρίσκεται ένα επίπεδο πάνω από το δίκτυο και ένα επίπεδο κάτω από το λογισμικό εφαρμογών, βρίσκεται δηλαδή και στους πελάτες, αλλά και στους διακομιστές, προκειμένου να παράσχει τις *διαφάνειες*, στις οποίες αναφερθήκαμε.

Αν επομένως κάποιο λογισμικό συμμορφώνεται στα παραπάνω, τότε μπορούμε να το χαρακτηρίσουμε ως middleware και μερικές περιπτώσεις τέτοιου λογισμικού είναι:

- Οι χαμηλού επιπέδου υπηρεσίες προγραμματισμού δικτύου, όπως τα *sockets* και το *NetBIOS*.
- Πρωτογενείς υπηρεσίες, όπως η *εξομοίωση τερματικού* (Terminal Emulation), η *μεταφορά αρχείων* (File transfer) και το *ηλεκτρονικό ταχυδρομείο* (e-mail).
- Βασικές υπηρεσίες αλληλεπίδρασης πελάτη - διακομιστή, όπως η *απομακρυσμένη κλήση διαδικασιών* (Remote Procedure Call), η *απομακρυσμένη προσπέλαση δεδομένων* (Remote Data Access), οι *υπηρεσίες ασφαλείας* (security services), *καταλόγου* (directory services), το *middleware ανταλλαγής μηνυμάτων* (Message Oriented Middleware), οι υπηρεσίες *διεκπεραιώσεων* (transaction services) κ.α.
- Middleware παγκόσμιου ιστού, όπως το πρωτόκολλο HTTP, η *κοινή διασύνδεση πύλης* (Common Gateway Interface) και διάφορα πρόσθετα διακομιστών παγκοσμίου ιστού, που επιτρέπουν το σε υψηλό επίπεδο προγραμματισμό για την υλοποίηση *δυναμικών ιστοσελίδων*, όπως τα Active Server Pages (ASP) και τα Java Server Pages (JSP).
- Υπηρεσίες κατανομής αντικειμένων, όπως το Common Object Request Broker Architecture (CORBA) και το Common Object Model (COM).
- Ειδικού τύπου middleware, όπως *συνεργασίας ομάδας* (groupware), *υπηρεσίες ροής εργασίας* (workflow services), πολυμέσων, ασύρματης εργασίας (wireless computing) κ.α.
- Οδηγοί βάσεων δεδομένων ανοικτών διασυνδέσεων πύλης, όπως το Open Data Base Connectivity (ODBC).

Κάθε λογισμικό middleware διαθέτει συνήθως το δικό του σύνολο προγραμματιστικών διασυνδέσεων εφαρμογών (APIs), οι οποίες μπορούν να χρησιμοποιηθούν κατά την ανάπτυξη εφαρμογών. Οι υπηρεσίες, που παρέχονται από το λογισμικό middleware, εξελίσσονται διαρκώς και προσφέρουν [UMA97]:

- Υποστήριξη νέου τύπου εφαρμογών, όπως για παράδειγμα οι εφαρμογές παγκόσμιου ιστού, οι εφαρμογές κινητής υπολογιστικής επεξεργασίας (mobile computing), οι κατανεμημένες εφαρμογές πολυμέσων και οι εφαρμογές συνεργασίας (Computer-Supported Collaborative Work).

- Υψηλά επίπεδα διαφάνειας για τους χρήστες (μη αντίληψης δηλαδή, που μέσα στο δίκτυο λαμβάνει χώρα η κάθε επεξεργασία), τους προγραμματιστές (απόκρυψη των χαμηλού επιπέδου λεπτομερειών λόγω της διαφορετικότητας των συστημάτων) και τους διαχειριστές (διαχείριση της κατανεμημένης εφαρμογής σα να πρόκειται για ένα ενιαίο σύστημα).
- Βελτιωμένη μεταφερσιμότητα και αλληλοδραστικότητα εφαρμογών μέσα στο σύνολο των χρησιμοποιούμενων υπολογιστικών συστημάτων.
- Συνέπεια και σταθερότητα στην παροχή υπηρεσιών.
- Βελτιωμένους χρόνους απόκρισης και βελτιωμένη διαθεσιμότητα (availability) υπηρεσιών.
- Βελτιωμένα εργαλεία και υποδομές διαχείρισης.

Η εξέλιξη αυτή πρέπει, όπως είναι φυσικό, να ελαχιστοποιεί τους κινδύνους και το κόστος ανάπτυξης νέων εφαρμογών. Έτσι, ήταν επιβεβλημένη η ανάπτυξη εκτεταμένων δραστηριοτήτων *τυποποίησης* (standardization), ώστε η κατασκευή νέων προϊόντων να βασίζεται σε κάποια *θεσπισμένα πρότυπα* (standards). Μία σύντομη παρουσίαση των σημαντικότερων τεχνολογιών κατανεμημένου λογισμικού γίνεται στην παράγραφο 2.5 και μία αναλυτικότερη περιγραφή κάποιων από αυτές γίνεται στις επόμενες παραγράφους του κεφαλαίου.

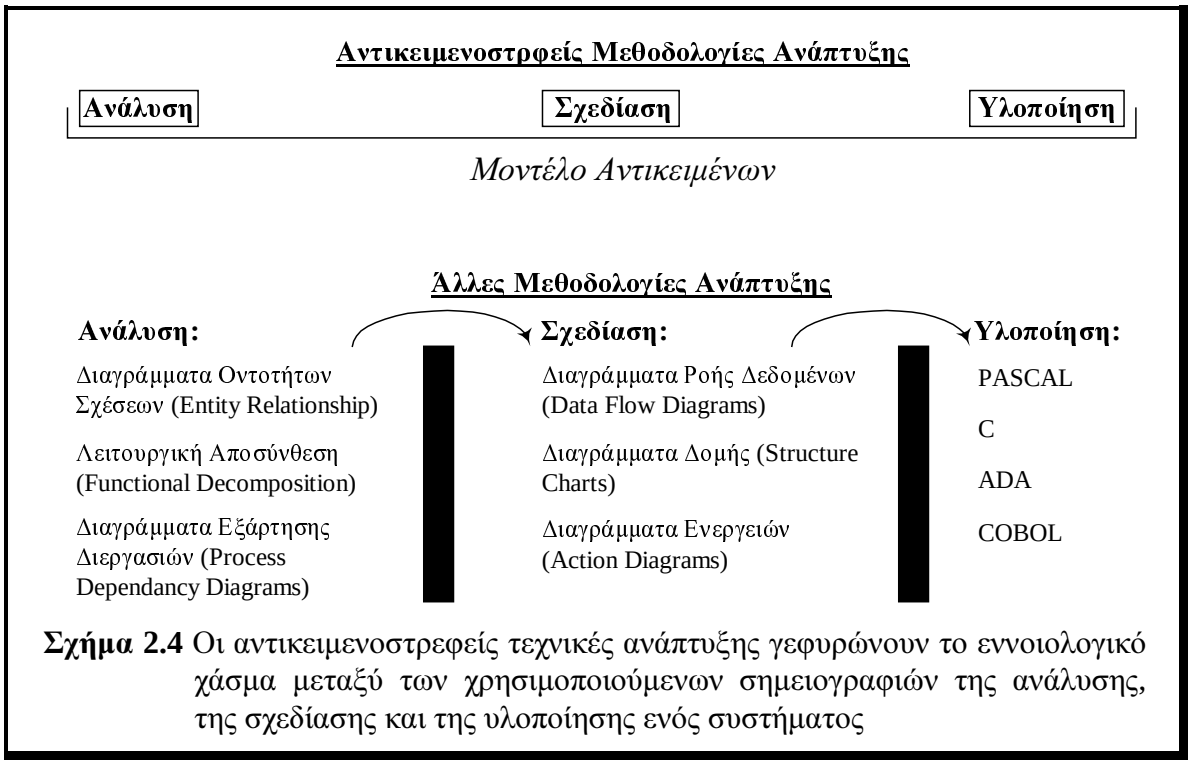
2.4 Ανάπτυξη αντικειμενοστρεφούς λογισμικού: Μεθοδολογίες και Υποδείγματα

Η αντικειμενοστρεφής προσέγγιση ανάπτυξης συστημάτων διαφέρει πολύ από την παραδοσιακή δομημένη προσέγγιση. Βασικό της χαρακτηριστικό είναι το γεγονός ότι επικεντρώνεται γύρω από την ίδια έννοια, σε όλη την πορεία της ανάπτυξης· την έννοια του αντικειμένου. Το γεγονός αυτό βοηθά στη γεφύρωση του εννοιολογικού χάσματος μεταξύ των φάσεων της ανάλυσης, της σχεδίασης και της υλοποίησης του συστήματος (σχήμα 2.4). Όταν χρησιμοποιούμε αντικειμενοστρεφείς τεχνικές, οι αναλυτές/σχεδιαστές και οι προγραμματιστές αλλά ακόμη και οι χρήστες, χρησιμοποιούνε την ίδιο σημειογραφικό μοντέλο αναφοράς - το *μοντέλο των αντικειμένων*. Έτσι, όλοι λειτουργούν στο ίδιο εννοιολογικό πλαίσιο, που προσδιορίζεται από το μοντέλο των αντικειμένων.

Οι αντικειμενοστρεφείς μεθοδολογίες ανάπτυξης, δίνουν μεγάλη σημασία στη φάση της ανάλυσης. Με τον τρόπο αυτό γίνεται ευκολότερος ο εντοπισμός λαθών περιγραφής του συστήματος, σχετικά νωρίς στη διαδικασία ανάπτυξης. Κατά την ανάλυση ενδιαφέρει βασικά το *τι προβλήματα θα επιλύει το σύστημα*, ενώ κατά τη σχεδίαση το *πως θα τα επιλύει*. Η αντικειμενοστρεφής ανάπτυξη καθιστά επίσης εφικτή την επαναλαμβανόμενη εναλλαγή μεταξύ των φάσεων της ανάλυσης, της σχεδίασης και της υλοποίησης, αφού και στις τρεις περιπτώσεις χρησιμοποιείται το ίδιο μοντέλο αντικειμένων. Διαμορφώνεται έτσι μία βήμα προς βήμα εξελικτική διαδικασία ανάπτυξης του συστήματος αποτέλεσμα, στο οποίο απέβλεπαν όλοι οι μέχρι σήμερα προτεινόμενοι κύκλοι ζωής λογισμικού.

Στο πλαίσιο αυτό, η αρχική εμφάνιση περισσοτέρων της μιας αντικειμενοστρεφούς σημειογραφίας - μεθοδολογίας ήρθε να προκαλέσει κάποια σύγχυση· σύγχυση όμως, η οποία διαλύθηκε με τα αποτελέσματα ενός έργου τυποποίησης της επονομαζόμενης *Unified Modeling Language* (UML), που εξελίσσεται στους κόλπους μιας μεγάλης σύμπραξης ομίλων, γνωστής ως *Object Management Group* (OMG). Η UML βρήκε μεγάλη απήχηση στους κύκλους των μηχανικών λογισμικού αφού ενσωματώνει με επιτυχία όλα τα ιδιαίτερα

προτερήματα των μέχρι πρότινος εμφανισθέντων μεθοδολογιών. Έτσι, στην παράγραφο αυτή, όπως επίσης και στην υπόλοιπη διατριβή, γίνεται χρήση της UML.



Σχήμα 2.4 Οι αντικειμενοστρεφείς τεχνικές ανάπτυξης γεφυρώνουν το εννοιολογικό χάσμα μεταξύ των χρησιμοποιούμενων σημειογραφιών της ανάλυσης, της σχεδίασης και της υλοποίησης ενός συστήματος

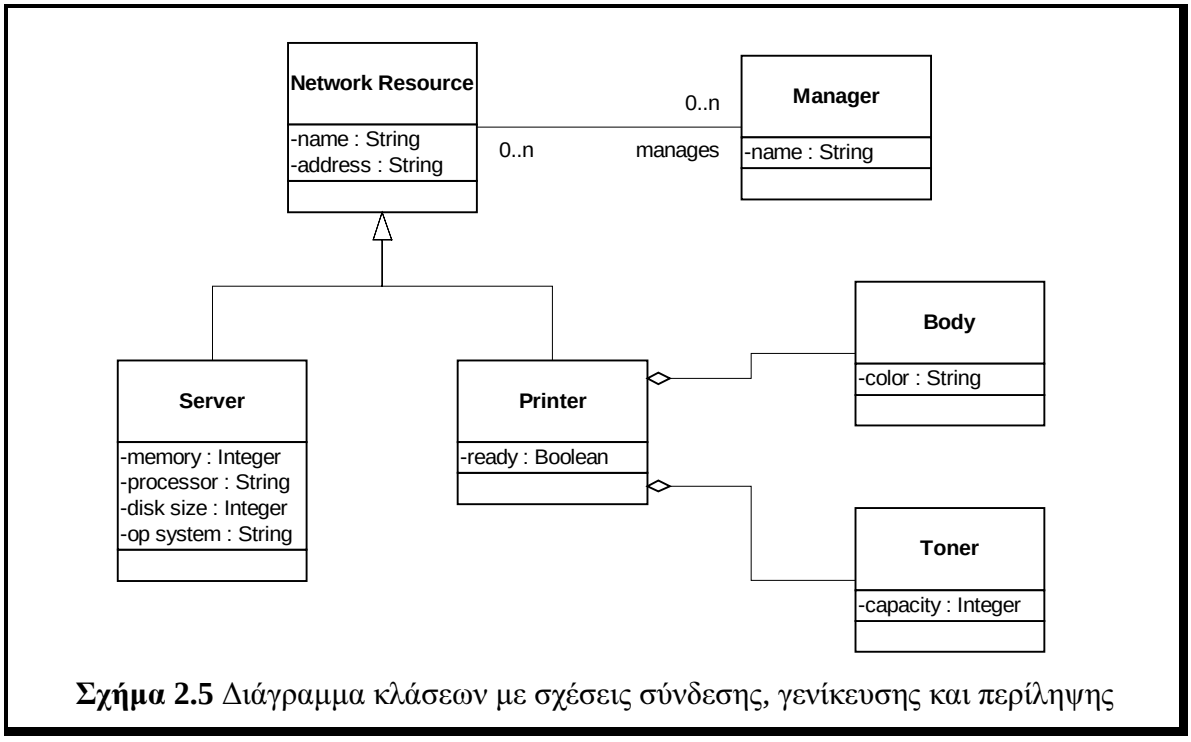
2.4.1 Αντικειμενοστρεφής Ανάλυση

Κατά τη φάση αυτή δημιουργούνται μοντέλα, που απεικονίζουν την πραγματικότητα του προς επίλυση προβλήματος. Ο σκοπός κάθε τέτοιου μοντέλου είναι βασικά η καταγραφή και η σημειογραφική αναπαράσταση (μέρους) των προδιαγραφών του συστήματος, με έναν αφαιρετικό τρόπο, που αποκρύπτει τις λεπτομέρειες σχεδίασης, που σε αυτή τη φάση δεν ενδιαφέρουν. Είναι λοιπόν βολικό να αντιλαμβανόμαστε το πρόβλημα με όρους αντικειμένων με ιδιότητες και συμπεριφορές. Έτσι, για παράδειγμα, ένα αυτοκίνητο έχει τις ιδιότητες χρώμα, αριθμός πλαισίου κ.α. και τη συμπεριφορά «μπορεί να κινείται», «μπορεί να σταματά», «μπορεί να σκάσει λάστιχο» κ.α. Έτσι, το μοντέλο αντικειμένων, που προκύπτει, περιγράφει τα αντικείμενα από τα οποία αποτελείται το σύστημα και εκφράζει ουσιαστικά τη *στατική δομή* αυτού. Κατά τη φάση της ανάλυσης όμως δημιουργείται και ένα *δυναμικό μοντέλο*, το οποίο περιγράφει τις ροές ελέγχου του συστήματος, τον τρόπο δηλαδή, με τον οποίο αλληλεπιδρούν τα αντικείμενα μεταξύ τους.

Ένα μοντέλο αντικειμένων στη UML μπορεί να έχει τη μορφή *διαγράμματος κλάσεων* ή *διαγράμματος αντικειμένων*. Τα διαγράμματα αυτά απεικονίζουν τις κλάσεις (αντικείμενα στη δεύτερη περίπτωση) και τις μεταξύ τους *σχέσεις*. Το διάγραμμα αντικειμένων είναι μία παραλλαγή του διαγράμματος κλάσεων, που χρησιμοποιεί σχεδόν ταυτόσημα σύμβολα, μόνο, που αντί των ιδίων των κλάσεων απεικονίζει *περιπτώσεις* των κλάσεων. Έτσι, ένα διάγραμμα αντικειμένων είναι ουσιαστικά ένα παράδειγμα ενός διαγράμματος κλάσεων, πιθανώς δηλαδή ένα στιγμιότυπο της εκτέλεσης του συστήματος. Στο σχήμα 2.5 έχουμε ένα παράδειγμα διαγράμματος κλάσεων, στη UML. Η διαφορά στα διαγράμματα αντικειμένων είναι στο ότι τα αντικείμενα γράφονται με τα ονόματά τους υπογραμμισμένα και στο ότι για κάθε σχέση απεικονίζονται όλες οι περιπτώσεις.

Η περιγραφή ενός συστήματος περιλαμβάνει συνήθως έναν αριθμό διαγραμμάτων κλάσης και κάθε κλάση μπορεί να απαντάται σε περισσότερα του ενός διαγράμματα. Οι τύποι των σχέσεων, που υποστηρίζει η UML, είναι:

- Η *σύνδεση*, η οποία είναι μία μορφή συσχέτισης μεταξύ κλάσεων (ή αντικειμένων). Οι μορφές των συνδέσεων, που αναγνωρίζονται είναι οι *κανονικές συνδέσεις*, οι *αναδρομικές* συνδέσεις, οι *συνδέσεις OR*, οι *πολυσυνδέσεις* και οι *περιλήψεις*. Μία περίληψη περιγράφει μία σχέση του τύπου «αποτελείται» ή αντίστροφα του τύπου «μέρος του».
- Η *γενίκευση*, η οποία είναι μια σχέση μεταξύ του πιο γενικού και του πιο ειδικού. Το πιο ειδικό μπορεί να περιλαμβάνει μόνο επιπλέον πληροφορίες σε σχέση με το πιο γενικό. Διακρίνονται σε *κανονικές* και σε *υπό συνθήκη γενικεύσεις*.
- Η *εξάρτηση*, που είναι μία σχέση μεταξύ ενός ανεξάρτητου και ενός εξαρτημένου στοιχείου. Οποιαδήποτε αλλαγή στο ανεξάρτητο στοιχείο επηρεάζει το εξαρτημένο.
- Η *εξειδίκευση*, η οποία είναι μία σχέση μεταξύ δύο περιγραφών της ίδιας οντότητας, σε διαφορετικά όμως επίπεδα αφαίρεσης.



Σχήμα 2.5 Διάγραμμα κλάσεων με σχέσεις σύνδεσης, γενίκευσης και περίληψης

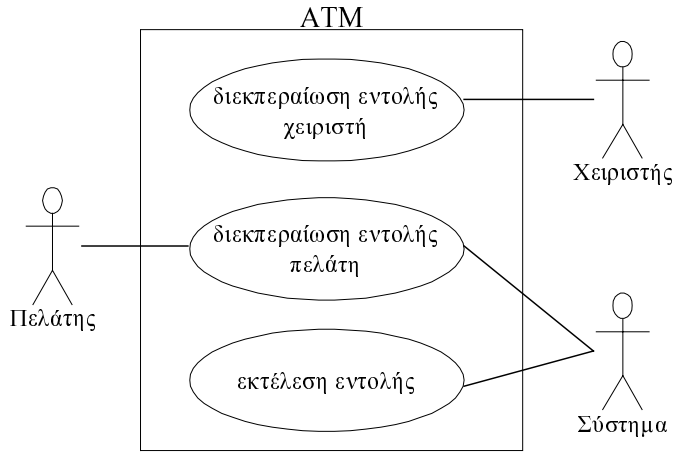
Κάθε σχέση έχει και τη δικιά της σημειογραφία, μερικές όμως από τις πιο συχνά χρησιμοποιούμενες, απεικονίζονται στο παράδειγμα του σχήματος 2.5. Επιπλέον, η UML δίνει τη δυνατότητα ορισμού νέων τύπων σχέσεων μέσα από ένα μηχανισμό *στερεοτύπων*.

Το *δυναμικό μοντέλο* ανάλυσης περιγράφει τα δυναμικά χαρακτηριστικά του συστήματος. Εδώ εντοπίζουμε, πως τα αντικείμενα αλληλεπιδρούν μεταξύ τους για την υλοποίηση της κάθε λειτουργίας του συστήματος, μη παραλείποντας να συμπεριλάβουμε τις αλληλεπιδράσεις αυτού με το χρήστη. Στη φάση αυτή, το δυναμικό μοντέλο καθορίζει τις προδιαγραφές των αντικειμένων, οι οποίες κατά τη φάση τις σχεδίασης θα οδηγήσουν στις *λειτουργίες* ή αλλιώς *μεθόδους* αυτών, που λείπουν από το στατικό μοντέλο του σχήματος 2.5.

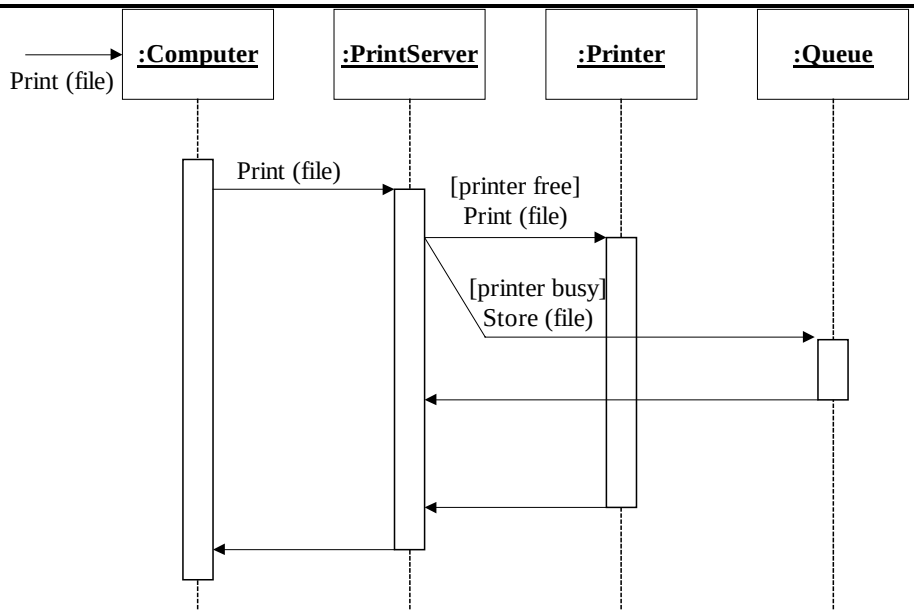
Οι τύποι των διαγραμμάτων, που υποστηρίζει η UML, για την κατασκευή ενός δυναμικού μοντέλου είναι:

- Οι *περιπτώσεις σεναρίων* (use-case diagrams), που περιγράφουν σενάρια χρήσης του συστήματος.
- Τα *διαγράμματα καταστάσεων* (state diagrams), που περιγράφουν τις δυνατές καταστάσεις του συστήματος και τις μεταβάσεις αυτού από τη μία στην άλλη.
- Τα *διαγράμματα αλληλουχιών μηνυμάτων* (message sequence diagrams), που περιγράφουν τις αλληλεπιδράσεις μεταξύ των αντικειμένων του συστήματος.
- Τα *διαγράμματα συνεργίας* (collaboration diagrams), όπου εκτός των αλληλεπιδράσεων απεικονίζονται και οι σχέσεις (δηλαδή το πλαίσιο συνύπαρξης) των αντικειμένων.
- Τα *διαγράμματα δράσεων* (activity diagrams), που χρησιμοποιούνται για την αναπαράσταση της χρονολογικής συσχέτισης και της ροής των δράσεων, οι οποίες συγκροτούν την εκτέλεση μιας λειτουργίας του συστήματος.

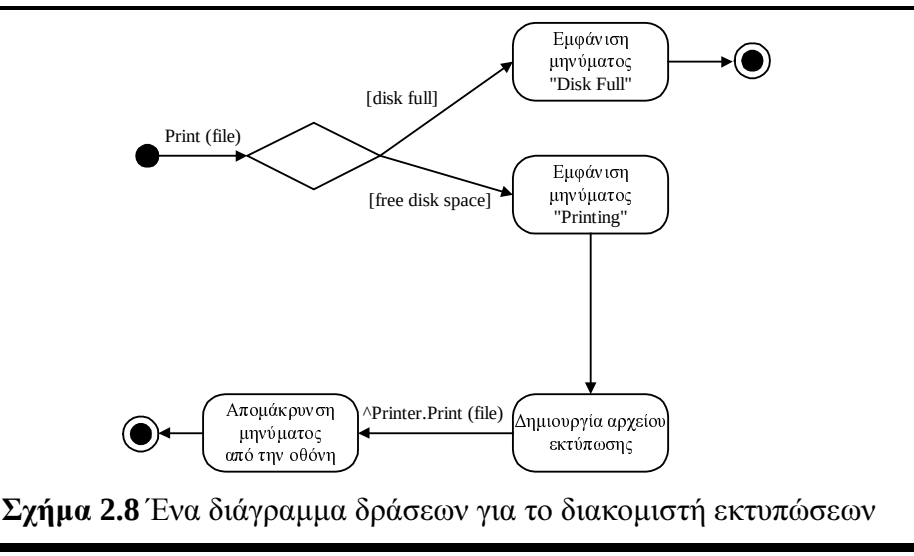
Στα σχήματα 2.6, 2.7 και 2.8 εικονίζονται αντίστοιχα από ένα παράδειγμα διαγράμματος περιπτώσεων σεναρίων, διαγράμματος αλληλουχίας μηνυμάτων και διαγράμματος δράσης.



Σχήμα 2.6 Ένα διάγραμμα περιπτώσεων σεναρίων μιας αυτόματης ταμειολογιστικής μηχανής



Σχήμα 2.7 Ένα διάγραμμα αλληλουχίας μηνυμάτων για ένα διακομιστή εκτυπώσεων



Σχήμα 2.8 Ένα διάγραμμα δράσεων για το διακομιστή εκτυπώσεων

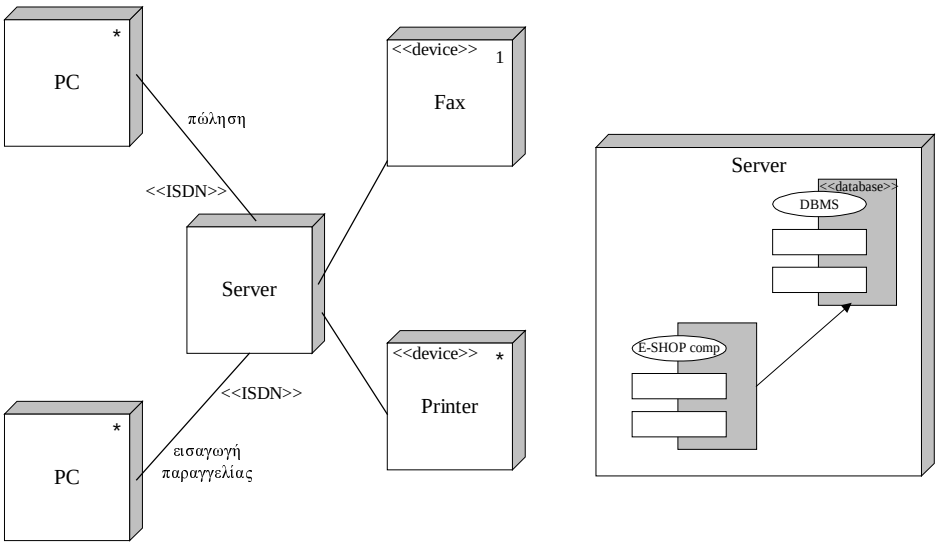
2.4.2 Αντικειμενοστρεφής Σχεδίαση

Στη σχεδίαση χρησιμοποιούνται τα προϊόντα της ανάλυσης για την παραγωγή αυτών, πάνω στα οποία θα βασισθεί η υλοποίηση του συστήματος. Η αντικειμενοστρεφής σχεδίαση διακρίνεται στη *σχεδίαση συστήματος* και στη *σχεδίαση αντικειμένων*.

Κατά την πρώτη, επιλέγονται η γλώσσα προγραμματισμού, οι βιβλιοθήκες κλάσεων, τα συστήματα βάσεων και όλα τα υπόλοιπα προϊόντα από τα οποία θα εξαρτάται η υλοποίηση του προς ανάπτυξη συστήματος. Λαμβάνονται επίσης σημαντικές αποφάσεις, που αφορούν το σύστημα ως ενιαία οντότητα, όπως:

- Τι μορφή θα έχει το υλικό, που θα χρησιμοποιηθεί;
- Τι πόροι μπορούν να χρησιμοποιηθούν από το σύστημα;
- Τι μορφή θα έχουν οι διασυνδέσεις μεταξύ αντικειμένων και άλλων πόρων, όπως για παράδειγμα των βάσεων δεδομένων, των χρηστών και άλλων διεργασιών;
- Πως μπορεί το σύστημα να χωρισθεί σε υποσυστήματα ή σε τμήματα;

Ειδικά όσον αφορά το τελευταίο, η UML διαθέτει τα *διαγράμματα τμημάτων* (component diagrams) και τα *διαγράμματα ανάπτυξης* (deployment diagrams).



Σχήμα 2.9 Ένα διάγραμμα ανάπτυξης ενός ηλεκτρονικού καταστήματος

Ένα διάγραμμα τμημάτων δείχνει τη φυσική δομή του κώδικα ως προς τα τμήματα αυτού. Ένα τμήμα μπορεί να είναι τμήμα πηγαίου κώδικα, δυαδικό ή εκτελέσιμο και περιέχει πληροφορίες για την κλάση ή τις κλάσεις, που αυτό υλοποιεί.

Σε ένα διάγραμμα ανάπτυξης τέλος, περιγράφεται η φυσική κατανομή του υλικού και του λογισμικού του συστήματος. Σε αυτό μπορούμε να εμφανίσουμε τους κόμβους, τις μεταξύ τους συνδέσεις και τους τύπους αυτών. Μέσα στους κόμβους τοποθετούνται εκτελέσιμα τμήματα και αντικείμενα και έτσι σχηματίζεται ένα μοντέλο αναπαράστασης της θέσης των μονάδων λογισμικού στους κόμβους του συστήματος.

Στο [BE96] ο G. Booch προσδιορίζει τι περιμαβάνει η σχεδίαση συστήματος ενός λογισμικού κατανεμημένης αρχιτεκτονικής:

- Την *κατανομή* αυτού, όπου περιγράφεται η φυσική τοποθέτηση των κλάσεων και των αντικειμένων στους κόμβους επεξεργασίας του δικτύου και για το σκοπό αυτό χρησιμοποιούνται διαγράμματα ανάπτυξης.
- Τη *φυσική ομαδοποίηση* κλάσεων και αντικειμένων για τα οποία πρέπει να εγγυάται η σχεδίαση την συνύπαρξή τους στον ίδιο κόμβο επεξεργασίας. Ο Booch προτείνει την εισαγωγή της έννοιας «μονάδα κατανομής» (Distribution Unit) και τη χρήση της σε συνδυασμό με τα διαγράμματα ανάπτυξης. Κάθε μονάδα κατανομής ορίζεται με τη χρήση του μηχανισμού ομαδοποίησης της UML, που είναι γνωστός ως *πακέτο*.
- Του *συγχρονισμού* της επικοινωνίας μεταξύ των κατανεμημένων αντικειμένων, που μπορεί να αναπαρασταθεί, είτε με διαγράμματα αλληλουχίας μηνυμάτων, είτε με διαγράμματα συνεργίας.
- Τις *διασυνδέσεις* και πιο συγκεκριμένα το φυσικό διαχωρισμό αυτών από τις υλοποίησεις των αντίστοιχων αντικειμένων. Ο διαχωρισμός εκφράζεται με κατάλληλο τρόπο μέσα από τα διαγράμματα κλάσεων.

- Της *αποδημίας* (migration) των κλάσεων και των αντικειμένων μέσα στο δίκτυο, που μπορεί να αναπαρασταθεί (αν υπάρχει) με τη χρήση διαγραμμάτων συνεργίας.

Στη σχεδίαση αντικειμένων καθορίζονται οι λεπτομέρειες του μοντέλου αντικειμένων, το οποίο προέκυψε κατά τη φάση της ανάλυσης. Αποφασίζονται οι μέθοδοι, που θα υλοποιεί η κάθε κλάση, και προστίθενται νέα εσωτερικά αντικείμενα, που η ύπαρξή τους κρίνεται απαραίτητη επειδή σχετίζονται, είτε με τα δεδομένα, είτε με τους αλγορίθμους, που επιλέγονται για την επεξεργασία αυτών.

2.4.3 Υποδείγματα

Τα υποδείγματα εκφράζουν λύσεις, που διατυπώθηκαν και εξελίχθηκαν μέσα στο χρόνο, για διάφορα, συχνά εμφανιζόμενα, προβλήματα σχεδίασης. Έτσι, σε καμία περίπτωση δε μπορούν να συγκριθούν με την ποιότητα των λύσεων, που αναπτύσσονται στα πλαίσια ενός έργου και μόνο για τη χρήση τους σε αυτό. Ο λόγος είναι ότι τα υποδείγματα ενσωματώνουν τα αποτελέσματα επανασχεδιάσεων και προσαρμογής αυτών, σε περισσότερα του ενός προβλήματα, που έχει ως αποτέλεσμα τη βελτίωση των περιθωρίων επεκτασιμότητας και επαναχρησιμοποίησης της λύσης που περιγράφουν. Η τεκμηρίωση των υποδειγμάτων γίνεται με τέτοιο τρόπο, ώστε να διευκολύνεται η εφαρμογή τους, χωρίς να απαιτούνται ιδιαίτερα χαρακτηριστικά συγκεκριμένων γλωσσών προγραμματισμού ή κάποια εξεζητημένα προγραμματιστικά τεχνάσματα.

Τα *αρχιτεκτονικά υποδείγματα* εκφράζουν, σύμφωνα με το [BMR96], θεμελιώδη πρότυπα δομικής οργάνωσης συστημάτων λογισμικού. Παρέχουν ένα σύνολο προκαθορισμένων υποσυστημάτων, περιγράφουν την αποστολή του καθενός από αυτά και περιλαμβάνουν κανόνες και οδηγίες για την οργάνωση των μεταξύ τους σχέσεων.

Από την άλλη, σύμφωνα με το [GHJ95], τα *σχεδιαστικά υποδείγματα* περιγράφουν συχνά εμφανιζόμενες δομές επικοινωνούντων τμημάτων λογισμικού, που επιλύουν ένα γενικό σχεδιαστικό πρόβλημα στο ειδικό πλαίσιο διαφόρων περιπτώσεων εμφάνισης αυτού.

Τέλος, τα *ιδιώματα* είναι χαμηλού επιπέδου υποδείγματα, προοριζόμενα για χρήση σε συγκεκριμένη ή συγκεκριμένες γλώσσες προγραμματισμού. Ένα ιδίωμα εκφράζει τον τρόπο υλοποίησης κάποιας άποψης των τμημάτων ή των μεταξύ τους σχέσεων, χρησιμοποιώντας τα χαρακτηριστικά μιας συγκεκριμένης γλώσσας προγραμματισμού.

Δυστυχώς, δε χρησιμοποιείται πάντα το ίδιο πρότυπο περιγραφής υποδειγμάτων. Γενικά όμως, θα μπορούσαμε να πούμε, ότι κάθε τέτοια περιγραφή περιλαμβάνει το πλαίσιο, το πρόβλημα και τη λύση αυτού. Η πρώτη σημαντική προσπάθεια τεκμηρίωσης ενός συνόλου υποδειγμάτων γενικής χρήσης ήταν αυτή του [GHJ95]. Ένα χρόνο αργότερα δημοσιεύθηκε το [BMR96], όπου περιλαμβάνεται και ένας αριθμός αρχιτεκτονικών υποδειγμάτων. Τέλος, ιδιαίτερο ενδιαφέρον παρουσιάζει και το [SSR00], όπου τεκμηριώνεται ένα σύνολο σχεδιαστικών και αρχιτεκτονικών υποδειγμάτων για λογισμικό κατανεμημένων αντικειμένων.

2.5 Τεχνολογίες κατανεμημένων αντικειμένων

Η κύρια υπηρεσία ανταλλαγής πληροφοριών σε ένα δίκτυο TCP/IP, είναι τα *sockets*. Οι βασικές λειτουργίες τους είναι: i) η διάκριση των διαφορετικών εφαρμογών, που εκτελούνται στην ίδια πλατφόρμα και ii) η παροχή ενδιάμεσων αποθηκών για την αποστολή και τη λήψη δεδομένων. Έτσι, τα sockets μπορούν να εκληφθούν ως λογικές διασυνδέσεις, τις οποίες οι εφαρμογές μπορούν να δημιουργούν, να συνδέονται και να τις χρησιμοποιούν για την αποστολή και λήψη των πακέτων. Τα IP sockets επιτρέπουν τη λήψη και αποστολή IP

πακέτων, ενώ τα UDP sockets επιτρέπουν τη λήψη και αποστολή UDP πακέτων. Στη δεύτερη περίπτωση, το TCP/IP δεν εγγυάται τη λήψη των πακέτων με τη σειρά με την οποία εστάλησαν, γι΄ αυτό το λόγο τα πακέτα UDP χρησιμοποιούνται πιο σπάνια και περισσότερο για μεταφορές μικρής ποσότητας δεδομένων με απλή δομή. Ένα παράδειγμα γλώσσας προγραμματισμού, που διαθέτει προγραμματιστική διασύνδεση εφαρμογής (API) για τη χρήση sockets, είναι η Java, μολονότι δεν είναι το μοναδικό. Τα sockets είναι για την ανάπτυξη εφαρμογών δικτύου, ότι είναι η assembly για την ανάπτυξη συμβατικών εφαρμογών: προγραμματισμός σε χαμηλό επίπεδο. Οι μεταφορές δεδομένων, αντιμετωπίζονται ως ροές χαρακτήρων. Αυτό έχει ως αποτέλεσμα να είναι εύκολες στη χρήση, όταν πρόκειται για μεταφορά δεδομένων απλής μορφής, αλλά και να έχουν στις περισσότερες περιπτώσεις καλύτερη απόδοση σε σχέση με άλλες υψηλότερου επιπέδου μορφές διεργασιακής επικοινωνίας. Παρόλα αυτά, όταν τα δεδομένα είναι πιο σύνθετα (π.χ. αντικείμενα), το μεγάλο κόστος ανάπτυξης και συντήρησης, που συνοδεύει τη χρήση sockets, καθιστά τη συγκεκριμένη επιλογή απαγορευτική. Στις περιπτώσεις αυτές, τη λύση δίνει κάποια από τις τεχνολογίες κατανεμημένων αντικειμένων σε συνδυασμό ίσως με τη πρόσκτηση ισχυρότερων διακομιστών ή/και ταχύτερου δικτύου για καλύτερη απόδοση.

Οι τεχνολογίες κατανεμημένων αντικειμένων έχουν ως στόχο την παροχή *διαφάνειας θέσης* έτσι ώστε, να διευκολύνεται η προσπέλαση και η χρήση ενός αντικειμένου κάποιου απομακρυσμένου κόμβου, ως αν αυτό να βρισκόταν στον ίδιο κόμβο. Η διαφάνεια θέσης εξασφαλίζεται συνήθως μέσα από τις ακόλουθες λειτουργίες:

- εντοπισμός και φόρτωμα απομακρυσμένων κλάσεων,
- εντοπισμός απομακρυσμένων αντικειμένων και παροχή αναφορών σε αυτά,
- δυνατότητα *απομακρυσμένης κλήσης μεθόδων* και περάσματος απομακρυσμένων αντικειμένων ως παραμέτρους κλήσης και επιστρεφόμενες τιμές και τέλος,
- ειδοποίηση των προγραμμάτων για σφάλματα δικτύου και άλλα προβλήματα.

Η πρώτη περίπτωση, βρίσκει εφαρμογή στις *μικροεφαρμογές* (applets) Java, οι οποίες είναι δυνατό να περιέχουν αναφορές σε κλάσεις, που ο φυλλομετρητής πρέπει να κατεβάσει από τον κόμβο, στον οποίο βρίσκεται η μικροεφαρμογή. Παρόλα αυτά, τα συστήματα κατανεμημένων αντικειμένων απαιτούν συνήθως μία περισσότερο ευέλικτη προσέγγιση, κατά την οποία θα δίνεται η δυνατότητα εύρεσης και μεταφοράς των κλάσεων όχι από έναν αλλά από περισσότερους κόμβους. Αυτή η δυνατότητα θα επέτρεπε στους μηχανικούς λογισμικού την αποθήκευση κλάσεων σε περισσότερους του ενός κόμβους και θα παρείχε έτσι καλύτερη απόδοση και διαθεσιμότητα.

Η δεύτερη λειτουργία απαιτεί την ύπαρξη κάποιας μορφής καταλόγου ή βάσης των διαθέσιμων αντικειμένων και ενός διακομιστή, ο οποίος θα παρέχει πρόσβαση στον κατάλογο αυτό. Όταν λοιπόν το πρόγραμμα αιτεί κάποια συγκεκριμένη υπηρεσία, ουσιαστικά ζητά από το διακομιστή καταλόγου την παροχή μιας αναφοράς στο κατάλληλο αντικείμενο. Οι αναφορές αυτές είναι συνήθως διευθύνσεις μνήμης ή απλά εγγραφές σε πίνακες αντικειμένων και φυσικά, όταν αποστέλλονται μέσω δικτύου πρέπει να εμπεριέχουν αναφορά και στον κόμβο προέλευσής τους. Επιπλέον, γλώσσες όπως η Java, που υποστηρίζουν τη *συλλογή άχρηστων αντικειμένων*, θα πρέπει να διαθέτουν μηχανισμούς αναγνώρισης, του αν υπάρχουν απομακρυσμένες αναφορές σε κάποιο αντικείμενο του κόμβου.

Στην τρίτη περίπτωση, όπου υποστηρίζεται η δυνατότητα απομακρυσμένης κλήσης λειτουργιών, απαιτείται η ύπαρξη μηχανισμών για την παροχή αναφορών σε υποψήφιες προς χρήση λειτουργίες, όπως επίσης και μηχανισμών για τη μεταφορά παραμέτρων και

επιστρεφόμενων τιμών μέσω δικτύου. Καθώς τα αντικείμενα μπορεί να εμπεριέχουν άλλα αντικείμενα, μία απλή κλήση μπορεί τελικά να απαιτήσει χρόνο μεγαλύτερο του αναμενόμενου.

Τέλος, η τέταρτη λειτουργία αποβλέπει ουσιαστικά στην ενημέρωση των προγραμμάτων όταν η διαφάνεια θέσης δε λειτουργεί. Από την άποψη αυτή, θα μπορούσαμε να πούμε ότι δεν είναι ποτέ επιθυμητή η παροχή πλήρους διαφάνειας θέσης. Αυτό συμβαίνει διότι αν σε ένα κατανεμημένο περιβάλλον αποτύχει μία προσπάθεια αναφοράς σε ένα αντικείμενο, αυτό δεν πρέπει να σημαίνει αυτόματα κατάρρευση της εφαρμογής. Στις περισσότερες τεχνολογίες κατανεμημένου λογισμικού ορίζονται *εξαιρέσεις*, οι οποίες ενεργοποιούνται όταν γίνεται μία αποτυχημένη προσπάθεια αναφοράς. Έτσι διευκολύνεται η συγγραφή κώδικα ανθεκτικού σε τέτοιου είδους σφάλματα.

2.6 Ανάπτυξη λογισμικού με τεχνολογίες Java

Οι τρεις κύριοι λόγοι, που κατέστησαν τη γλώσσα προγραμματισμού Java τόσο δημοφιλή, είναι:

- η δυνατότητα τα ίδια προγράμματα να εκτελούνται σε διαφορετικούς τύπους πλατφόρμας,
- το χαρακτηριστικό της αποδημίας του κώδικα των μικροεφαρμογών Java και
- το γεγονός ότι η γλώσσα προγραμματισμού Java είναι πλήρως αντικειμενοστρεφής.

Στην προ Java εποχή, η μεταφερσιμότητα είχε βαρύνουσα σημασία ειδικά στο χώρο των συστημάτων UNIX, όπου έβρισκε βασικά εφαρμογή στο επίπεδο της μετάφρασης κώδικα. Αυτό σημαίνει ότι για πολλά προγράμματα ήταν συνήθως δυνατή η μετάφραση του ιδίου κώδικα για εκτέλεση σε διαφορετικές πλατφόρμες UNIX.

Τα προγράμματα Java είναι μεταφέρσιμα με την έννοια ότι είναι δυνατό να εκτελεστούν σε οποιαδήποτε πλατφόρμα με υποστήριξη Java, χωρίς να χρειάζεται κάθε φορά να γίνεται μετάφρασή του κώδικα. Αυτό επιτυγχάνεται μέσα από τη χρήση *ψηφιοκώδικα* (bytecode).

Ο ψηφιοκώδικας είναι μία μορφή αντικειμενικού κώδικα (object code) ανεξάρτητου όμως από τον τύπο της υποκείμενης πλατφόρμας. Η εκτέλεσή του δε γίνεται απευθείας από το υλικό, αλλά από μία Υπερβατική Μηχανή Java (Java Virtual Machine), η οποία παίζει το ρόλο του διερμηνευτή. Αυτό σημαίνει ότι ένα πρόγραμμα Java με τη μορφή που έχει είναι εκτελέσιμο σε κάθε πλατφόρμα, η οποία διαθέτει JVM. Το γεγονός ότι αυτή η ειδικού τύπου διερμηνεία κώδικα δίνει ως αποτέλεσμα μία τάξη μεγέθους πιο αργό κώδικα μηχανής, οδήγησε στο να δοθεί ιδιαίτερη έμφαση στη δυνατότητα συγγραφής αποδοτικού κώδικα. Επιπλέον, οι τελευταίας γενιάς μεταφραστές και μηχανές Java διαθέτουν δυνατότητες, όπως η *άμεση μετάφραση* (just-in-time compilation) και η *βελτιστοποίηση*, που ενισχύουν την ταχύτητα εκτέλεσης του ψηφιοκώδικα. Έτσι, καλύπτεται σε σημαντικό βαθμό η διαφορά στην ταχύτητα εκτέλεσης σε σχέση με αυτή της εκτέλεσης προγραμμάτων της C ή άλλων μεταφράσιμων γλωσσών προγραμματισμού.

Η Java διαθέτει επίσης μία μεγάλη γκάμα βιβλιοθηκών κλάσεων, ελαχιστοποιώντας έτσι την ανάγκη συγγραφής νέου κώδικα για την εκτέλεση διαφόρων συχνά χρησιμοποιούμενων λειτουργιών.

Ένα από τα σημαντικά χαρακτηριστικά της Java είναι η δυνατότητα κατασκευής ενός νέου τύπου προγράμματος, το οποίο ονομάζεται *μικροεφαρμογή* (applet) για να διακρίνεται από τις κανονικές εφαρμογές Java.

Μία μικροεφαρμογή λοιπόν βρίσκεται αποθηκευμένη σε κάποιο διακομιστή ιστοσελίδων, που συνήθως περιλαμβάνει μία ή περισσότερες σελίδες με αναφορά σε αυτή. Όταν ο φυλλομετρητής με δυνατότητα εκτέλεσης Java καλεί την ιστοσελίδα, η οποία περιέχει την αναφορά, τότε φορτώνει επίσης και εκτελεί και τη συγκεκριμένη μικροεφαρμογή. Αυτή κατά την εκτέλεσή της μπορεί να αλληλεπιδρά με το χρήστη μέσα από μία γραφική διεπαφή, αλλά μπορεί ακόμη και να ανοίξει μία ή περισσότερες συνδέσεις για πρόσβαση σε βάσεις δεδομένων ή/και σε άλλους πόρους.

Η δυνατότητα χρήσης μικροεφαρμογών δίνει απάντηση στο πρόβλημα της συντήρησης και κατανομής του λογισμικού. Κάθε φορά, που ο χρήστης καλεί μία σελίδα, η οποία περιέχει αναφορά σε μικροεφαρμογή, ο φυλλομετρητής κατεβάζει ένα «φρέσκο» αντίγραφο αυτής. Έτσι, ακόμη και αν η μικροεφαρμογή έχει τροποποιηθεί πρόσφατα, ο χρήστης θα λάβει τη νέα έκδοση χωρίς να χρειάζεται από μέρους του κάποια επιπλέον ενέργεια.

Η Java δίνει επίσης τη δυνατότητα εύκολης δημιουργίας - με τη χρήση της κατάλληλης βιβλιοθήκης - μικροεφαρμογών διακομιστή, των επονομαζόμενων *servlets*. Κάθε servlet, όταν εκτελείται στο διακομιστή στον οποίο βρίσκεται, εξυπηρετεί ουσιαστικά κλήσεις GET και POST του πρωτοκόλλου HTTP.

Η Java υποστηρίζει ακόμη μία λειτουργία *σειριακής αναδιάταξης*, η οποία επιτρέπει την εύκολη δημιουργία *παγίων αντικειμένων*. Αντικειμένων δηλαδή, που πέφτουν σε λήθαργο, όταν παύει η επεξεργασία τους, αλλά μπορούν αργότερα να επανέρθουν στην ίδια κατάσταση, στη JVM της ίδιας ή κάποιας άλλης υπολογιστικής μονάδας. Με τη λειτουργία της σειριακής αναδιάταξης τα αντικείμενα μπορούν εύκολα να αποθηκεύουν την κατάστασή τους στο δίσκο και να «ταξιδεύουν» μέσα σε ένα πληροφοριακό δίκτυο.

Ένα κανονικό αντικείμενο συνοδεύεται πάντα από την κλάση του στη JVM, που βρίσκεται. Κάθε κλάση αποθηκεύεται με τη μορφή αρχείου .class σε κάποια δευτερεύουσα μονάδα αποθήκευσης και είναι έτσι δυνατή η αντιγραφή της από τη μία υπολογιστική μονάδα στην άλλη. Η λειτουργία σειριακής αναδιάταξης ουσιαστικά παρέχει μία παρόμοια δυνατότητα και για τα αντικείμενα. Μπορούμε, χρησιμοποιώντας την, να καταγράψουμε την κατάσταση ενός αντικειμένου σε κάποιο αρχείο και ακολούθως με ανάγνωση του αρχείου αυτού να επανασυστήσουμε το ίδιο αντικείμενο. Φυσικά, για την επανασύσταση του αντικειμένου χρειάζεται να είναι διαθέσιμο και το κατάλληλο αρχείο .class.

2.6.1 Νήματα ροής (threads) και σύγχρονη εκτέλεση στη Java

Η Java παρέχει επίσης δυνατότητες δημιουργίας προγραμμάτων με χαρακτηριστικά σύγχρονης εκτέλεσης, μέσα από τη χρήση των *νημάτων ροής*. Τα νήματα ροής δεν είναι παρά μέρη του ιδίου προγράμματος, που εκτελούνται όμως ταυτόχρονα. Ως μέρη του ιδίου προγράμματος μοιράζονται τα ίδια δεδομένα και τις ίδιες λειτουργίες σε μία από κοινού χρησιμοποιούμενη περιοχή μνήμης. Έτσι, για τη μεταξύ τους επικοινωνία δε χρειάζεται κάποιας μορφής διεργασιακή επικοινωνία και το κόστος, που αυτή συνεπάγεται. Επιπλέον, η δημιουργία και η κατάργηση νημάτων ροής γίνεται γρήγορα, χωρίς σημαντικό επιπλέον φόρτο για το λειτουργικό σύστημα.

Η χρήση νημάτων ροής δίνει τη δυνατότητα κατασκευής αποδοτικών διακομιστών, καθώς κάθε νέα αίτηση εξυπηρέτησης μπορεί να ικανοποιείται και από ένα διαφορετικό νήμα ροής, ανεξάρτητα από το αν ο διακομιστής είναι ήδη απασχολημένος. Έτσι επιτυγχάνεται η ταυτόχρονη εξυπηρέτηση περισσοτέρων του ενός πελατών.

Για το συγχρονισμό σε *κρίσιμα σημεία* του κώδικα, η Java παρέχει τις κατάλληλες λέξεις κλειδιά και μεθόδους.

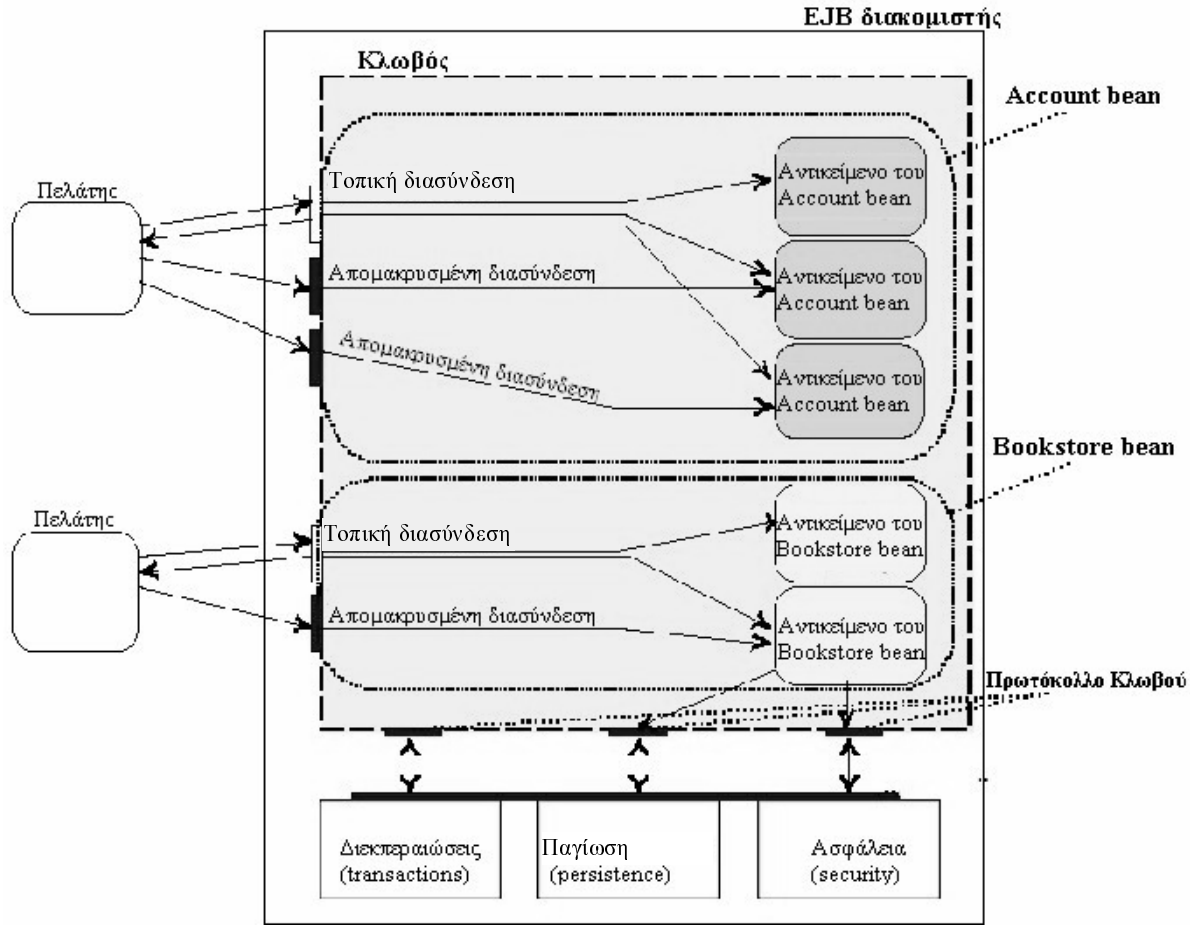
Σε κάθε περίπτωση η χρήση νημάτων ροής χρειάζεται ιδιαίτερη προσοχή, καθώς η εκτέλεσή τους επηρεάζεται από τον τρόπο με τον οποίο τα χειρίζεται ο χρονοδρομολογητής της πλατφόρμας εκτέλεσης. Κάποιες πλατφόρμες για παράδειγμα χρησιμοποιούν χρονοδρομολόγηση *κυκλικής εναλλαγής* για νήματα ροής της ιδίας προτεραιότητας, ενώ κάποιες άλλες χρησιμοποιούν χρονοδρομολόγηση *μη παραγκωνισμού* (non-preemptive) του νήματος, το οποίο εξυπηρετείται από τη CPU. Η λάθος χρήση των νημάτων ροής χωρίς γνώση των επιπτώσεων, που αυτή μπορεί να έχει, αποδίδει ως αποτέλεσμα ένα αργό και μη λειτουργικό λογισμικό.

2.6.2 Java Beans και Enterprise Java Beans (EJB)

Τα *Java Beans* είναι η λύση της Java στην ανάγκη υποστήριξης ενός μοντέλου ανάπτυξης και διάθεσης επαναχρησιμοποιήσιμων τμημάτων λογισμικού. Είναι δηλαδή μονάδες λογισμικού, οι οποίες παρέχουν ευκολίες ενσωμάτωσης και προσαρμογής τους σε ένα πρόγραμμα Java. Τα Java Beans μπορεί να παρέχουν λειτουργικότητα στοιχείων GUI, ή λειτουργικότητα λογιστικού φύλλου, ή οτιδήποτε άλλο επιθυμεί ο κατασκευαστής τους. Ο προγραμματιστής είναι εύκολο να τα προσαρμόσει στις ανάγκες του, μέσω της χρήσης ενός φύλλου ιδιοτήτων (property sheet), που γίνεται διαθέσιμο σε οποιοδήποτε ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού (IDE), που υποστηρίζει Java. Αυτό επιτυγχάνεται με τη παροχή μεταπληροφοριών (meta-information), οι οποίες γνωστοποιούν στο περιβάλλον ανάπτυξης τις ιδιότητες, τις μεθόδους και τους τύπους δεδομένων των ιδιοτήτων και των ορισμάτων των μεθόδων, που παρέχει το κάθε Java Bean. Η επικοινωνία μεταξύ των Java Beans και των αντικειμένων, που ανταλλάσσουν μηνύματα με αυτά, γίνεται με τη χρήση του σχεδιαστικού υποδείγματος δημοσιοποίηση/αποδοχή. Έτσι, όταν γίνεται αλλαγή κάποιας ιδιότητας ενός Java Bean, τότε αυτό πυροδοτεί ένα συμβάν. Ένα συμβάν όμως, το οποίο δε γίνεται αντιληπτό από όλα τα αντικείμενα, αλλά μόνο από εκείνα, που έχουν προηγουμένως αποδεχθεί την ενημέρωσή τους γι΄ αυτό.

Τα *Enterprise Java Beans* (EJBs) είναι μία αρχιτεκτονική λύση για την κατασκευή κατανεμημένων αντικειμενοστρεφών εφαρμογών από επαναχρησιμοποιήσιμα τμήματα λογισμικού. Στο [MH99] ορίζονται οι προδιαγραφές στις οποίες πρέπει να συμμορφώνονται οι επονομαζόμενοι *EJB διακομιστές*.

Ένα EJBean υλοποιεί λειτουργικότητα εφαρμογής και αποτελείται από αντικείμενα. Βρίσκεται πάντα μέσα σε ένα *κλωβό*. Ο κλωβός αυτός αποτελεί ουσιαστικά το περιβάλλον του EJBean, που του διαθέτει υπηρεσίες διεκπεραιώσεων, ασφάλειας και παγίωσης. Οι προδιαγραφές δεν περιγράφουν τον τρόπο με τον οποίο υλοποιούνται οι συγκεκριμένες υπηρεσίες (αυτό αφήνεται στη δημιουργική ευχέρεια του κατασκευαστή), αλλά μόνο τις διασυνδέσεις μέσω των οποίων αυτές γίνονται προσβάσιμες από τα αντικείμενα του EJBean.



Σχήμα 2.10 Ένας διακομιστής Enterprise Java Beans

Ο κώδικας της επικοινωνίας μεταξύ των αντικειμένων των EJBean και του κλωβού λέγεται *πρωτόκολλο κλωβού*. Επιπλέον, κάθε EJBean έχει μία *περιγραφή ανάπτυξης* (deployment descriptor), που καθορίζει το πως χρησιμοποιούνται οι βασικές και οι επιπρόσθετες υπηρεσίες του κλωβού. Ο κάθε κλωβός υποστηρίζει ένα σύνολο κανόνων επικοινωνίας με τους πελάτες, που καλείται *πρωτόκολλο πελάτη*. Το πρωτόκολλο πελάτη ορίζει δύο είδη διασυνδέσεων: την *τοπική* και την *απομακρυσμένη*. Η απομακρυσμένη διασύνδεση αναφέρεται στην καθαυτό λειτουργικότητα του κάθε EJBean. Η τοπική διασύνδεση υποστηρίζει μεθόδους για τη δημιουργία και διαγραφή αντικειμένων του bean, όπως επίσης και μεθόδους ερωτημάτων σχετικά με τον πληθυσμό των αντικειμένων του.

Για την κλήση των μεθόδων ενός EJBean, ο πελάτης λαμβάνει πρώτα μία αναφορά για την τοπική διασύνδεση αυτού, κάνοντας χρήση του Java Naming and Directory Interface (JNDI). Με τη συγκεκριμένη αναφορά ο πελάτης μπορεί να δημιουργήσει ή να αναζητήσει ένα αντικείμενο του EJBean και να λάβει μία αναφορά για την απομακρυσμένη του διασύνδεση. Μέσω της διασύνδεσης αυτής είναι στη συνέχεια δυνατή η μεταβίβαση κλήσεων μεθόδων του συγκεκριμένου αντικείμενου του EJBean.

Ένας EJB διακομιστής διαχειρίζεται τα EJBean, που βρίσκονται στην κεντρική του μνήμη, με διαφάνεια ως προς τους πελάτες. Όταν ο πληθυσμός των αντικειμένων των EJBean ενός κλωβού υπερβαίνει κάποιο επίπεδο, ο κλωβός αποθηκεύει κάποια από τα (όχι πρόσφατα χρησιμοποιημένα) αντικείμενα σε μία δευτερεύουσα μονάδα αποθήκευσης - λέμε τότε ότι αυτά περιέρχονται σε παθητική κατάσταση (passivated). Όταν λοιπόν μία μέθοδος έχει ως

στόχο ένα αντικείμενο, που βρίσκεται σε παθητική κατάσταση, τότε αυτό επαναφέρεται από τον κλωβό στη μνήμη (ενεργοποιείται).

Τα EJBeans διακρίνονται στα *beans συνόδου* (session beans) και στα *beans οντοτήτων* (entity beans). Τα beans συνόδου χρησιμοποιούνται για αντικείμενα μικρής διάρκειας ζωής. Τα beans οντοτήτων αντιπροσωπεύουν συνήθως κάποια δεδομένα βάσης και έχουν τη δυνατότητα μεσολάβησης για την εκτέλεση διεκπεραιώσεων και για την ταυτόχρονη προσπέλαση από περισσότερους του ενός πελάτες. Συνήθως η διάρκεια ζωής τους είναι μεγαλύτερη από τα αντικείμενα των EJBean συνόδου.

Τέλος, οι προδιαγραφές EJBean προσδιορίζουν επίσης και τις δυνατότητες αλληλοδραστικότητας με άλλες τεχνολογίες λογισμικού κατανεμημένων αντικειμένων. Έτσι, ένας CORBA πελάτης μπορεί μέσω της αντιστοίχησης CORBA, που πρέπει να διαθέτει κάθε EJB διακομιστής, να διοχετεύει κλήσεις εξυπηρέτησης σε ένα ή περισσότερα αντικείμενα αυτού.

Συμπερασματικά, η αρχιτεκτονική EJBeans ορίζει ένα μοντέλο για την ανάπτυξη επαναχρησιμοποιήσιμων τμημάτων Java για χρήση σε διακομιστές. Η υιοθέτηση της συγκεκριμένης τεχνολογίας επιτρέπει τον προγραμματισμό χωρίς να είναι απαραίτητη η κατανόηση χαμηλού επιπέδου λεπτομερειών επεξεργασίας διεκπεραιώσεων, πολυνημάτωσης, πολυπλεξίας συνδέσεων ή άλλων APIs. Έτσι, η αρχιτεκτονική EJBeans δεν είναι ανταγωνιστική αλλά συμπληρωματική άλλων τεχνολογιών λογισμικού κατανεμημένων αντικειμένων, όπως η τεχνολογία CORBA, που αποτελεί ουσιαστικά μία ολοκληρωμένη λύση αλληλεπίδρασης αντικειμένων.

2.6.3 Απομακρυσμένη Κλήση Μεθόδου (Remote Method Invocation)

Η *απομακρυσμένη κλήση μεθόδου* (RMI), επιτρέπει στα αντικείμενα διαφορετικών JVM να επικοινωνούν μεταξύ τους. Για το σκοπό αυτό γίνεται χρήση της λειτουργίας σειριακής αναδιάταξης της Java. Σε κάθε περίπτωση επικοινωνίας, προηγείται η σειριοποίηση των παραμέτρων ή των επιστρεφόμενων τιμών της κλήσης, ακολουθεί η μεταφορά αυτών μέσα από το δίκτυο και τέλος η αποσειριοποίησή τους.

Η αλληλεπίδραση αυτή υλοποιείται με τη χρήση τοπικών «αντικειμένων», που αντιπροσωπεύουν το έτερο αντικείμενο της κλήσης και είναι γνωστά ως *στέλεχος* (stub) και *σκελετός* (skeleton). Το στέλεχος βρίσκεται στη JVM του κόμβου του αντικειμένου - πελάτη, ενώ ο σκελετός βρίσκεται στον κόμβο του αντικειμένου, που καλείται. Το στέλεχος ενεργεί ουσιαστικά ως υποκατάστατο του απομακρυσμένου αντικειμένου, προάγοντας έτσι τη δυνατότητα κλήσης αυτού με διαφάνεια θέσης. Όταν αυτό λαμβάνει το μήνυμα από το αντικείμενο - πελάτη, το προωθεί μέσω του δικτύου στο απομακρυσμένο αντικείμενο, παραλαμβάνει την απόκρισή του και ακολούθως την επιστρέφει στον πελάτη. Ο σκελετός ενεργεί ανάλογα, για λογαριασμό όμως του αντικειμένου, που καλείται.

Όπως είναι αναμενόμενο, οι εφαρμογές, που χρησιμοποιούν RMI, είναι ευκολότερες στην ανάπτυξη και συντήρηση από ότι αυτές, που χρησιμοποιούν sockets. Επιπλέον, με το RMI είναι επίσης ευκολότερη και η υπέρβαση του λογισμικού προστασίας (firewall), που συχνά εμποδίζει στα προγράμματα την επικοινωνία με αντικείμενα εκτός του προστατευόμενου πεδίου. Αυτό γίνεται γιατί αν το RMI δεν επιτύχει άμεση επικοινωνία με το απομακρυσμένο αντικείμενο, τότε αυτομάτως προσπαθεί να κάνει χρήση του HTTP. Βέβαια το HTTP είναι μία τάξη μεγέθους πιο αργό από το κανονικό RMI, αλλά σίγουρα στην περίπτωση αυτή είναι προτιμότερο από τη μη επικοινωνία.

Σημαντικό επίσης είναι το γεγονός ότι υπάρχει η δυνατότητα χρήσης του RMI πάνω από το πρωτόκολλο Internet-ORB (IIOB), που χρησιμοποιείται κατά την αλληλεπίδραση

αντικειμένων CORBA. Αυτό δίνει τη δυνατότητα επικοινωνίας αντικειμένων, που κάνουν χρήση RMI, με άλλα, που κάνουν χρήση της τεχνολογίας CORBA.

Η κριτική που γίνεται στο RMI της Java, αφορά την εξάρτησή του από τη λειτουργία σειριακής αναδιάταξης, η οποία για μεγάλα αντικείμενα απαιτεί μεγάλα ποσά μνήμης και αριθμούς κύκλων CPU. Σε αυτές τις περιπτώσεις η αλόγιστη χρήση RMI μπορεί να δημιουργήσει προβλήματα απόδοσης.

2.7 Ανάπτυξη λογισμικού με την τεχνολογία CORBA (Common Object Request Broker Architecture)

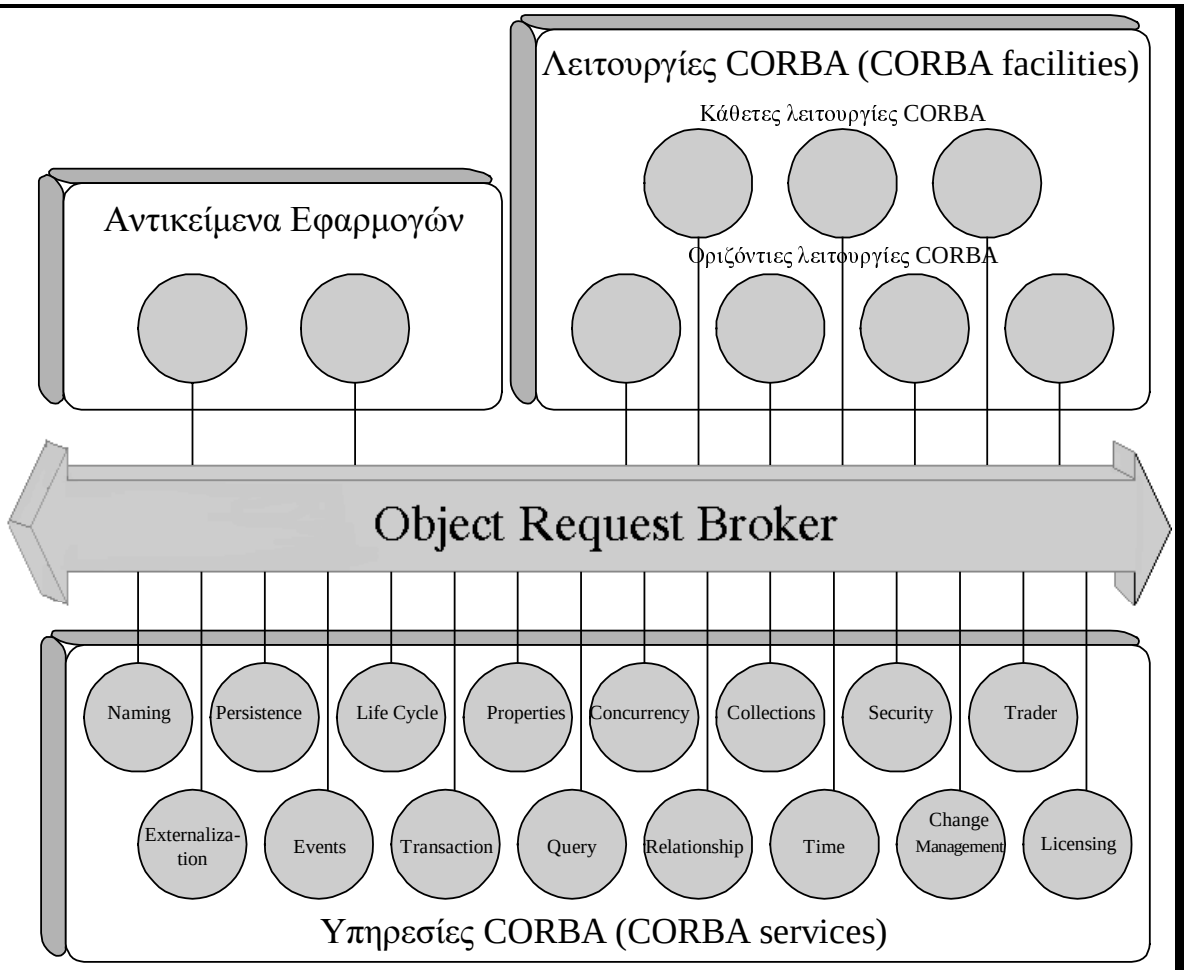
Το CORBA αποτελεί ακρώνυμο, που αναφέρεται στην τεχνολογία, που επιτρέπει στα αντικείμενα μιας υπολογιστικής μονάδας να επικοινωνούν με αντικείμενα μιας άλλης υπολογιστικής μονάδας. Τα αντικείμενα μπορεί να είναι γραμμένα σε οποιεσδήποτε γλώσσες προγραμματισμού από αυτές, που υποστηρίζονται από την υλοποίηση ORB (Object Request Broker), που χρησιμοποιείται, και μπορεί να εκτελούνται σε διαφορετικού τύπου πλατφόρμες.

Η τεχνολογία CORBA περιγράφεται από ένα σύνολο προδιαγραφών, που συνεχίζει να αναπτύσσεται από μία μεγάλη σύμπραξη ομίλων, υπό την επωνυμία OMG (Object Management Group). Είναι δε σημαντικό να τονιστεί ότι το OMG δεν παράγει ποτέ και σε καμία περίπτωση λογισμικό. Ο αποκλειστικός του σκοπός είναι η θέσπιση κοινά αποδεκτών προδιαγραφών. Όταν λοιπόν το OMG έχει πιστοποιήσει κάποιες προδιαγραφές, είναι μετά ευθύνη των διαφόρων κατασκευαστών να αναπτύξουν τα προϊόντα, που θα συμμορφώνονται σε αυτές. Η επιλογή χρήσης κάποιου από τα προϊόντα αυτά, είναι ο πιο ενδεδειγμένος τρόπος για την υλοποίηση μίας λύσης «ανοικτής υπολογιστικής αρχιτεκτονικής», της οποίας ο κύκλος ζωής δε θα είναι απαραίτητα αλληλένδετος με το λογισμικό ενός μόνο κατασκευαστή.

Βέβαια, οι προδιαγραφές CORBA περιγράφουν μόνο το πως τα αντικείμενα επικοινωνούν μεταξύ τους. Δεν υπάρχει δηλαδή ενσωματωμένη πληροφορία για άλλες υπηρεσίες, όπως για παράδειγμα η ασφάλεια, η παγίωση, η υποστήριξη διεκπεραιώσεων κ.λ.π. Αυτές περιγράφονται από άλλες προδιαγραφές, όπως οι *υπηρεσίες* και οι *λειτουργίες* CORBA, που μαζί με τα *αντικείμενα εφαρμογών*, συγκροτούν την *Αρχιτεκτονική Διαχείρισης Αντικειμένων* (Object Management Architecture).

2.7.1 Μοντέλο Αρχιτεκτονικής Διαχείρισης Αντικειμένων (OMA)

Οι *προδιαγραφές OMA* [OMG92] περιλαμβάνουν δύο σημαντικά τμήματα: το *μοντέλο αναφοράς* (reference model) και ένα υψηλού επιπέδου *μοντέλο αντικειμένων* (abstract object model), στο οποίο βασίζονται όλες οι υπόλοιπες προδιαγραφές του OMG.



Σχήμα 2.11 Το Μοντέλο της Αρχιτεκτονικής Διαχείρισης Αντικειμένων (Object Management Architecture)

Το μοντέλο αναφοράς OMA (σχήμα 2.11) διαχωρίζει τις τεχνολογίες ανάπτυξης λογισμικού κατανεμημένων αντικειμένων στις τεχνολογίες, οι οποίες προσφέρουν τη βασική υποδομή και αυτές, που έχουν να κάνουν με την ανάπτυξη εφαρμογών. Οι τεχνολογίες που προσφέρουν τη βασική υποδομή είναι:

- Ο *Αγωγός Κλήσεων Αντικειμένων* (ORB), που ορίζει το πως αντικείμενα γραμμένα σε διαφορετικές γλώσσες προγραμματισμού και προορισμένα να εκτελούνται σε διαφορετικές υπολογιστικές μονάδες, μπορούν να επικοινωνούν μεταξύ τους.
- Οι *υπηρεσίες CORBA*, που ορίζουν προδιαγραφές για επιπλέον λειτουργικότητα αντικειμένων, που είναι κοινή για όλες τις εφαρμογές. Κάποιες από τις υπηρεσίες για τις οποίες ήδη έχουν ολοκληρωθεί οι προδιαγραφές και έχουν αναπτυχθεί προϊόντα, που τις υποστηρίζουν, είναι:
  - Η υπηρεσία συλλογών, η οποία παρέχει πρόσβαση σε μία ποικιλία δομών δεδομένων.
  - Η υπηρεσία ελέγχου σύγχρονης εκτέλεσης, η οποία επιτρέπει σε έναν αριθμό πελατών να συντονίζουν την προσπέλαση σε κοινά διαμοιράσιμους πόρους.
  - Η υπηρεσία συμβάντων, που επιτρέπει τη διάδοση συμβάντων διαφορετικής προέλευσης σε έναν αριθμό αποδεκτών.

- Η υπηρεσία εξωτερίκευσης, που επιτρέπει σε ένα αντικείμενο ή αντικείμενα να αποθηκευθούν ως ροές χαρακτήρων.
- Η υπηρεσία αδειών, η οποία ασκεί έλεγχο, που άπτεται δικαιωμάτων πνευματικής ιδιοκτησίας.
- Η υπηρεσία κύκλου ζωής, που ορίζει συμβάσεις για τη δημιουργία, τη διαγραφή, την αντιγραφή και τη μετακίνηση αντικειμένων.
- Η υπηρεσία ονοματοδοσίας, η οποία αποδίδει ένα μοναδικό όνομα σε κάθε αντικείμενο.
- Η υπηρεσία παγίων αντικειμένων, που συμβάλλει στην αποθήκευσή τους σε κάποιο μέσο, το οποίο μπορεί να είναι μία σχεσιακή βάση ή μία βάση αντικειμένων ή κάτι άλλο.
- Η υπηρεσία ορισμού ιδιοτήτων, η οποία επιτρέπει τη συσχέτιση των αντικειμένων με ζεύγη του τύπου όνομα/τιμή.
- Η υπηρεσία ερωτημάτων, η οποία δίνει τη δυνατότητα εκτέλεσης ερωτημάτων, που αφορούν συλλογές αντικειμένων.
- Η υπηρεσία συσχετίσεων, η οποία προσφέρει ένα μέσο για τη λογική αναπαράσταση σχέσεων μεταξύ οντοτήτων.
- Η υπηρεσία ασφαλείας, η οποία διαχειρίζεται την προσπέλαση σε αντικείμενα, που διέπονται από περιορισμούς ανά χρήστη ή ανά ρόλο.
- Η υπηρεσία χρονικού προσδιορισμού, που χρησιμοποιείται για αναφορά στην τρέχουσα χρονική στιγμή.
- Η υπηρεσία προώθησης, η οποία βοηθάει στην ανεύρεση άλλων υπηρεσιών με βάση κάποια επιθυμητή λειτουργικότητα.
- Η υπηρεσία διεκπεραιώσεων, η οποία διαχειρίζεται ένα σύνολο ταυτόχρονων διεκπεραιώσεων, που λαμβάνουν χώρα σε ένα πιθανόν ετερογενές υπολογιστικό περιβάλλον.

Όσον αφορά τις τεχνολογίες CORBA, που διευκολύνουν την ανάπτυξη εφαρμογών, αυτές είναι:

- Οι *λειτουργίες CORBA*, που παρέχουν επιπλέον λειτουργικότητα σε ένα επίπεδο πιο κοντά προς το χρήστη. Αυτό γίνεται, είτε με τη μορφή κάθετων υπηρεσιών για την ανάπτυξη συγκεκριμένων εφαρμογών, είτε με τη μορφή οριζόντιων υπηρεσιών για την ανάπτυξη πάσης φύσης εφαρμογών. Οι πιο σημαντικές κάθετες λειτουργίες αφορούν τα πεδία:
  - λογιστικής,
  - ανάπτυξης εφαρμογών,
  - κατανεμημένης προσομοίωσης,
  - εικονογράφησης κ.α.

  Οι τέσσερις κατηγορίες στις οποίες κατατάσσονται οι οριζόντιες υπηρεσίες, που παρέχονται ως λειτουργίες CORBA είναι οι:
  - διεπαφής χρήστη,

- διαχείρισης πληροφοριών, που περιλαμβάνει λειτουργίες για τη μοντελοποίηση, τον ορισμό, την αποθήκευση και την ανταλλαγή πληροφοριών,
- διαχείρισης συστήματος, που περιλαμβάνει λειτουργίες για τη διαχείριση πληροφοριακών συστημάτων και
- διαχείρισης εργασιών, που περιλαμβάνει λειτουργίες για την αυτοματοποίηση διαφόρων εργασιών επιπέδου χρήστη ή συστήματος.

- Τα *αντικείμενα εφαρμογών*, που παρέχουν υπηρεσίες για την εφαρμογή ή τις εφαρμογές στα πλαίσια των οποίων αναπτύσσονται. Το μοντέλο αναφοράς ορίζει ότι οι εφαρμογές CORBA αποτελούνται από μεγάλο αριθμό αντικειμένων κάποια από τα οποία είναι αντικείμενα εφαρμογής και κάποια άλλα τα οποία ανήκουν στις λειτουργίες CORBA.

2.7.2 Το Μοντέλο Αντικειμένων του OMG

Το υψηλού επιπέδου μοντέλο αντικειμένων του OMG ορίζει ένα σημασιολογικό πλαίσιο για τις κατανεμημένες εφαρμογές, που συμμορφώνονται στα πρότυπα του OMG. Εισάγει έννοιες και κανόνες, που διασφαλίζουν τη μεταφερσιμότητα και την αλληλοδραστικότητα των τμημάτων λογισμικού μέσα σε ένα κατανεμημένο ετερογενές περιβάλλον.

Το μοντέλο αντικειμένων του OMG αποτελείται από δύο μέρη: το *βασικό πυρήνα* και ένα σύνολο κανόνων, που επιτρέπουν την επέκτασή του με διαφορετικούς τρόπους. Ο βασικός πυρήνας περιλαμβάνει τις εξής έννοιες:

- Το *αντικείμενο* είναι μία οντότητα με συγκεκριμένη *ταυτότητα, τρέχουσα κατάσταση* και παρατηρίσιμη *συμπεριφορά*. Η ταυτότητα του κάθε αντικειμένου παραμένει αμετάβλητη και ανεξάρτητη από την κατάσταση και τη συμπεριφορά του. Τα αντικείμενα μοντελοποιούν οντότητες κάθε είδους, όπως πρόσωπα, έγγραφα, διεργασίες συστήματος κ.α.

- Μία *μέθοδος* είναι μία κλήση επεξεργασίας. Κάθε μέθοδος έχει μία *δήλωση* (signature), που ορίζει το όνομά της, τις παραμέτρους και τα αποτελέσματα. Οι μέθοδοι ενός αντικειμένου είναι και ο μοναδικός τρόπος επίδρασης στην κατάσταση αυτού.

- Ένας *τύπος αντικειμένου* είναι μία δήλωση, που έχει ισχύ σε ένα σύνολο αντικειμένων. Εφόσον το μοντέλο αντικειμένων περιστρέφεται γύρω από εξωτερικά παρατηρήσιμες ιδιότητες, ο τύπος ενός αντικειμένου ορίζεται πλήρως από το σύνολο των μεθόδων, που αυτό υποστηρίζει. Οι μέθοδοι αυτές συλλογικά αποκαλούνται *διασύνδεση* του τύπου. Αν λοιπόν ένα αντικείμενο υποστηρίζει όλες τις μεθόδους της διασύνδεσης ενός τύπου, τότε λέμε ότι το αντικείμενο αυτό είναι μια *περίπτωση* του συγκεκριμένου τύπου.

- Ένας *υποτύπος* είναι μία εξειδίκευση ενός τύπου αντικειμένου, που περιλαμβάνει όλες τις μεθόδους της διασύνδεσής του, αλλά και κάποιες επιπλέον μεθόδους. Κάθε αντικείμενο λοιπόν, του οποίου ο τύπος είναι υποτύπος κάποιου τύπου αντικειμένου, μπορεί να χρησιμοποιηθεί και ανάλογα με τον τρόπο, που χρησιμοποιείται το αντικείμενο του αρχικού τύπου.

- *Κληρονομικότητα* είναι ο μηχανισμός που επιτρέπει σε ένα τύπο αντικειμένου να ορίζεται σε σχέση με κάποιον άλλο τύπο. Ο κληρονόμος τύπος διαθέτει όλες τις μεθόδους του γονεϊκού τύπου, όπως και κάποιες επιπλέον μεθόδους. Ως αποτέλεσμα, ο κληρονόμος τύπος είναι ένας υποτύπος του γονεϊκού τύπου. Ο

> βασικός πυρήνας του μοντέλου του OMG ασχολείται μόνο με την κληρονομικότητα διασυνδέσεων. Στο μέρος αυτό των προδιαγραφών δεν υπάρχει καμία πληροφορία για το τι συμβαίνει στην υλοποίηση των κληρονομημένων μεθόδων ή στα χαρακτηριστικά, που συγκροτούν την κατάσταση του αντικειμένου.

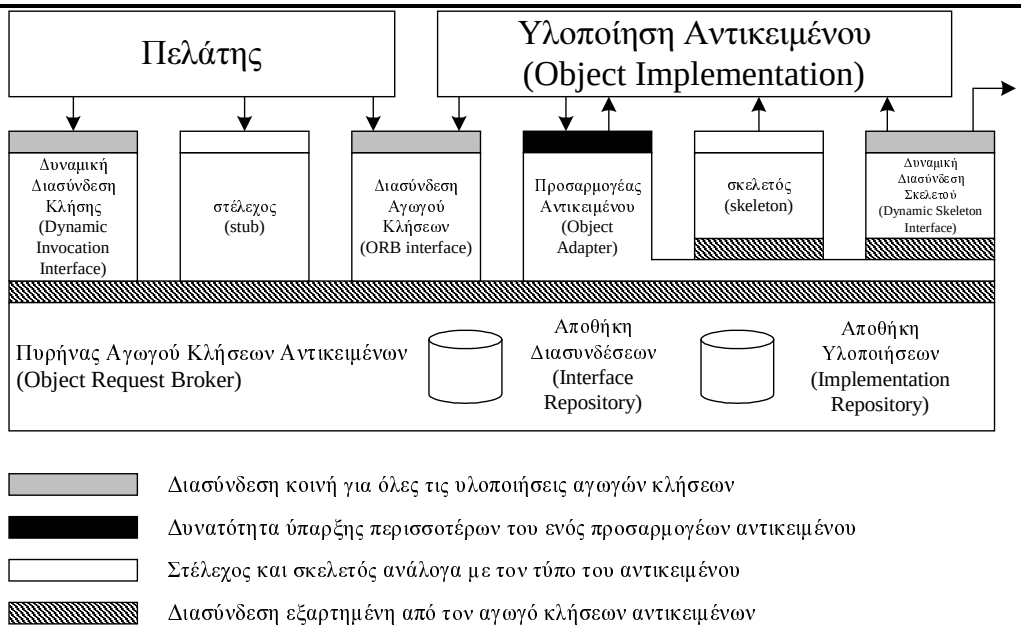
Παρόλο, που ο βασικός πυρήνας του μοντέλου αντικειμένων του OMG δεν περιλαμβάνει μεγάλο αριθμό εννοιών, είναι αρκετά σαφής σε ότι αφορά τον ορισμό διασυνδέσεων αντικειμένων και την υλοποίησή τους. Έτσι, επιτυγχάνονται οι δύο βασικοί στόχοι του μοντέλου, που είναι η μεταφερσιμότητα εφαρμογών και η αλληλοδραστικότητα συστημάτων στο επίπεδο μιας κοινής σημασιολογίας. Η μεταφερσιμότητα εφαρμογών επιτυγχάνεται μέσα από τη δυνατότητα που εξασφαλίζεται, για την περιγραφή ενός συστήματος με τη χρήση αντικειμένων, ανεξαρτήτως συγκεκριμένου λειτουργικού συστήματος ή γλώσσας υλοποίησης. Η αλληλοδραστικότητα επιτυγχάνεται εξαιτίας του γεγονότος ότι η σημασιολογία του βασικού πυρήνα του μοντέλου, ορίζεται με τρόπο ανεξάρτητο των σημασιολογικών μοντέλων των γλωσσών προγραμματισμού, που είναι δυνατό να χρησιμοποιηθούν. Κατά συνέπεια, στα αλληλεπιδρώντα συστήματα παρέχεται μία κοινή πλατφόρμα αλληλοδραστικότητας που περιλαμβάνει έννοιες, όπως οι τύποι αντικειμένων, η ταυτότητα αντικειμένων, η κλήση μεθόδου κ.α.

Είναι σαφές ότι ο βασικός πυρήνας του μοντέλου αντικειμένων του OMG επικεντρώνεται στον ορισμό ενός αφηρημένου σημασιολογικού πλαισίου και δεν παρέχει πλήρη υποστήριξη για την ανάπτυξη τεχνολογιών υλοποίησης. Έτσι για παράδειγμα, το μοντέλο δεν περιγράφει λεπτομέρειες για την κλήση μεθόδων σε επίπεδο κώδικα προγράμματος και κατά συνέπεια δεν μπορεί να χρησιμοποιηθεί για τη διασφάλιση της μεταφερσιμότητας πηγαίου κώδικα. Ομοίως, η απαίτηση ενός κοινού τρόπου αναπαραστάσης δεδομένων για την επίτευξη αλληλοδραστικότητας, δε μπορεί να επιτευχθεί χωρίς τον ορισμό μιας κοινής μορφής των απλών τύπων δεδομένων· δηλαδή των ακεραίων, των συμβολοσειρών κ.λ.π.

Για την αντιμετώπιση αυτών των πρακτικών ζητημάτων που ανακύπτουν στην υλοποίηση ενός αγωγού κλήσεων αντικειμένων, γίνεται χρήση μιας επέκτασης του βασικού πυρήνα του μοντέλου. Αυτή ασχολείται με πρακτικά θέματα υλοποίησης, όπως γλώσσες προγραμματισμού και πρωτόκολλα αλληλοδραστικότητας [OMG97]. Έτσι για παράδειγμα για την υποστήριξη του κλασικού προτύπου κλήσης μεθόδων στην κατανεμημένη του έκδοση, το μοντέλο CORBA αποκρυσταλλώνει την έννοια της δήλωσης μεθόδου, εισάγοντας τρία είδη παραμέτρων: τις παραμέτρους εισόδου, τις παραμέτρους εξόδου και τις παραμέτρους εισόδου/εξόδου. Επιπλέον, η κοινή αναπαράσταση δεδομένων επιτυγχάνεται με την εισαγωγή ενός συνόλου απλών τύπων δεδομένων και τη δυνατότητα ένωσης αυτών σε πίνακες, αλληλουχίες κ.α.

2.7.3 Η αρχιτεκτονική CORBA

Η *αρχιτεκτονική CORBA* [OMG97] είναι το κεντρικό τμήμα της Αρχιτεκτονικής Διαχείρισης Αντικειμένων (OMA). Ουσιαστικά περιγράφει των αγωγό κλήσεων αντικειμένων (σχήμα 2.12), που είναι υπεύθυνος για τη μεταφορά των κλήσεων των μεθόδων και των αποκρίσεων αυτών.



Σχήμα 2.12 Ο Αγωγός Κλήσεων Αντικειμένων (ORB)

Ένας *πελάτης CORBA* είναι οτιδήποτε έχει τη δυνατότητα κλήσεων μεθόδων αντικειμένων. Έτσι για παράδειγμα, το ρόλο ενός πελάτη CORBA μπορεί να παίξει κάποιο πρόγραμμα συστήματος, κάποια εφαρμογή χρήστη ή ένα σενάριο εκτέλεσης (script). Μία *υλοποίηση αντικειμένου* είναι ένα εκτελέσιμο τμήμα λογισμικού, που δημιουργεί ένα ή περισσότερα αντικείμενα CORBA και παρέχει την απαραίτητη υποστήριξη για τη κλήση μεθόδων των αντικειμένων σε χρόνο εκτέλεσης. Έτσι, μία υλοποίηση αντικειμένου περιέχει: (i) ορισμούς δεδομένων για τη δημιουργία αντικειμένων και (ii) τον κατάλληλο κώδικα προγράμματος για την εκτέλεση των μεθόδων των αντικειμένων. Οι υλοποιήσεις αντικειμένων είναι ουσιαστικά, τμήματα λογισμικού, υπεύθυνα για την παροχή των υπηρεσιών, οι οποίες διατίθενται στους πελάτες μέσω των διασυνδέσεων των αντικειμένων.

Οι προδιαγραφές CORBA υποστηρίζουν διαφορετικά επίπεδα μεγέθους (granularities) για τις υλοποιήσεις αντικειμένων. Έτσι, μία υλοποίηση μπορεί να δίνει υπόσταση σε ένα μόνο αντικείμενο ή μπορεί να υποστηρίζει ένα μεγάλο αριθμό αντικειμένων, όπως όταν υλοποιείται ένας διακομιστής βάσης δεδομένων. Επιπλέον, η υλοποίηση ενός αντικειμένου μπορεί να χωριστεί σε πολλαπλά προγράμματα ή διεργασίες του κόμβου του διακομιστή, κάθε μία από τις οποίες αντιστοιχεί σε διαφορετική μέθοδο.

Το *στέλεχος* επιτρέπει στον πελάτη την κλήση ενός αντικειμένου. Αρχικά, δέχεται την κλήση του πελάτη σα να ήταν μία τοπική κλήση. Ακολούθως, μετατρέπει την κλήση σε ένα μήνυμα δικτύου και την αποστέλλει στην κατάλληλη υλοποίηση αντικειμένου. Για πολλές γλώσσες προγραμματισμού δημιουργείται ξεχωριστό στέλεχος για κάθε μέθοδο, που περιλαμβάνεται στη διασύνδεση. Η μορφή του στελέχους, εξαρτάται γενικά από το προϊόν CORBA, που χρησιμοποιείται, και το λειτουργικό περιβάλλον του πελάτη. Τα στελέχη είναι υπεύθυνα για την *κωδικοποίηση* (marshaling) των παραμέτρων κλήσης, την προώθηση των κλήσεων στις κατάλληλες υλοποιήσεις αντικειμένων και την *αποκωδικοποίηση* (unmarshaling) των αποκρίσεων. Επιπλέον, τα στελέχη μπορούν να επιστρέφουν *εξαιρέσεις* στους πελάτες, έτσι ώστε αυτοί να μπορούν να διαχειρίζονται με κατάλληλο τρόπο τις όποιες μη αναμενόμενες καταστάσεις. Μία εξαίρεση μπορεί να οφείλεται στον αγωγό κλήσεων αντικειμένων (ORB) ή στην υλοποίηση του αντικειμένου.

Ο *προσαρμογέας αντικειμένου* είναι αυτός, που επιτρέπει στον αγωγό κλήσεων αντικειμένων (ORB) τη μεταφορά των κλήσεων μεθόδων στις κατάλληλες υλοποιήσεις αντικειμένων.

Επιπλέον, επιτρέπει στις υλοποιήσεις αντικειμένων τη χρήση των υπηρεσιών του αγωγού κλήσεων (ORB). Ο προσαρμογέας αντικειμένου υποστηρίζει λειτουργίες καταγραφής νέων αντικειμένων, ενεργοποίησης και απενεργοποίησης υπαρχόντων αντικειμένων και προσπέλασης πληροφοριών που συντηρούνται από τον αγωγό κλήσεων (ORB) και αφορούν αντικείμενα. Στη βασική τους έκδοση οι προδιαγραφές CORBA ορίζουν ένα μόνο τύπο προσαρμογέα, τον επονομαζόμενο *βασικό προσαρμογέα αντικειμένου* (Basic Object Adapter), για την υποστήριξη αντικειμένων, που υλοποιούνται από εκτελέσιμα προγράμματα και διεργασίες. Παρόλα αυτά, για άλλα είδη αντικειμένων χρησιμοποιούνται διαφορετικοί τύποι προσαρμογέων. Έτσι για παράδειγμα, στο [REV96] περιγράφεται ένας *προσαρμογέας βάσης αντικειμένων* για την ολοκληρωμένη λειτουργία του αγωγού κλήσεων (ORB) με βάσεις αντικειμένων. Επίσης, στο [OHE96] προτείνεται ένας *προσαρμογέας μηνυμάτων αντικειμένων* για υλοποιήσεις αντικειμένων, που κάνουν χρήση ασύγχρονης επικοινωνίας βασισμένης σε μηνύματα.

Ο *σκελετός* είναι το αντίστοιχο του στελέχους από τη μεριά του διακομιστή. Παραλαμβάνει κλήσεις μεθόδων από τον αγωγό κλήσεων (ORB) και τις μετατρέπει σε τοπικές κλήσεις με προορισμό την υλοποίηση του αντικειμένου, που αυτές απευθύνονται. Η δομή του σκελετού εξαρτάται από τον προσαρμογέα αντικειμένου και τη γλώσσα υλοποίησης του αντικειμένου. Έτσι, για αρκετές γλώσσες προγραμματισμού δημιουργείται ξεχωριστός σκελετός για κάθε μέθοδο, που περιλαμβάνεται σε μία διασύνδεση και οι σκελετοί αυτοί είναι μόνιμα συνδεδεμένοι με το εκτελέσιμο του αντικειμένου. Ο σκελετός είναι υπεύθυνος για την απόκρυψη των λεπτομερειών επικοινωνίας από τον προγραμματιστή που υλοποιεί το αντικείμενο. Οι λεπτομέρειες που αποκρύπτονται, είναι η αποκωδικοποίηση των παραμέτρων κλήσης, η κωδικοποίηση της απόκρισης και η επιστροφή της στον πελάτη και η διαχείριση μη αναμενόμενων καταστάσεων, οι οποίες θα μπορούσαν να οδηγήσουν σε αποτυχημένη επικοινωνία.

Η *δυναμική διασύνδεση κλήσης* (Dynamic Invocation Interface) επιτρέπει στους πελάτες την κλήση μεθόδων, οι οποίες δεν ήταν γνωστές κατά τη στιγμή της μετάφρασής των. Κατά τη χρήση της δυναμικής διασύνδεσης κλήσης ο πελάτης δε χρειάζεται στέλεχος. Αντί γι΄ αυτό απευθύνει μια σειρά ερωτήσεων στη δυναμική διασύνδεση, η οποία με τον τρόπο αυτό παρέχει μία λεπτομερή περιγραφή της ζητηθείσας μεθόδου. Ο πελάτης πρέπει να ταυτοποίησει το αντικείμενο - στόχο, τη μέθοδο που θα κληθεί, τις τιμές των παραμέτρων, τους τύπους των παραμέτρων, τον τύπο του αποτελέσματος κ.α. Όταν λοιπόν η κλήση έχει πλήρως καθορισθεί, τότε ο πελάτης την αποστέλλει στην κατάλληλη υλοποίηση αντικειμένου. Σε αντίθεση με την περίπτωση χρήσης στελέχους, η δυναμική διασύνδεση κλήσης επιτρέπει στον πελάτη τη συνέχιση της λειτουργίας του, παράλληλα με την εκτέλεση της κληθείσας μεθόδου από την υλοποίηση του αντικειμένου.

Η *δυναμική διασύνδεση σκελετού* (Dynamic Skeleton Interface) επιτρέπει στις υλοποιήσεις αντικειμένων να δέχονται κλήσεις, οι οποίες δεν ήταν γνωστές κατά τη στιγμή της μετάφρασης της εφαρμογής. Έτσι, αν μία κλήση μεθόδου ληφθεί μέσω της δυναμικής διασύνδεσης σκελετού και όχι μέσω ενός στατικού σκελετού, μετατρέπεται σε μία δομή δεδομένων, η οποία περιλαμβάνει όλες τις απαραίτητες πληροφορίες για την ταυτοποίηση της κλήσης. Όταν λοιπόν ολοκληρώνεται η εκτέλεση της κλήσης, τα αποτελέσματα επιστρέφονται στη δυναμική διασύνδεση σκελετού, η οποία τελικά τα μετατρέπει σε ένα μήνυμα, που στέλνεται μέσω του δικτύου στον πελάτη. Στον πελάτη δεν είναι ποτέ εμφανές το αν η κλήση έχει εκτελεστεί μέσω σκελετού ή μέσω της δυναμικής διασύνδεσης σκελετού.

Ο αγωγός κλήσεων αντικειμένων (ORB) είναι υπεύθυνος για τη μεταφορά κλήσεων και αποκρίσεων μεταξύ πελατών και υλοποιήσεων αντικειμένων. Επιπλέον, ο πυρήνας του αγωγού κλήσεων διασφαλίζει επίσης διαφάνεια θέσης, προσπέλασης και παγίωσης. Έτσι, τα

αντικείμενα CORBA δεν αναγνωρίζονται σε καμία περίπτωση από τη διεύθυνση δικτύου του κόμβου στον οποίο βρίσκονται. Αντί γι΄ αυτό, ο πυρήνας του αγωγού κλήσεων δημιουργεί και συντηρεί τις λεγόμενες *αναφορές αντικειμένων* (object references). Ακόμη, συντηρεί εσωτερικά μία υπηρεσία θέσης, η οποία αντιστοιχεί αναφορές αντικειμένων σε διευθύνσεις δικτύου. Έτσι, γίνεται απόκρυψη της λεπτομέρειας της θέσης των αντικειμένων από τις εφαρμογές, με κόστος βέβαια τον επιπλέον χρόνο για την ανεύρεση του κάθε αντικειμένου πριν από την προσπέλαση αυτού. Ακόμη, κατά τη μεταφορά κλήσεων και αποκρίσεων, ο αγωγός κλήσεων διαδραματίζει και ρόλους, όπως για παράδειγμα την επιβολή ασφαλούς επικοινωνίας και τη συντήρηση πληροφοριών κατάστασης κατά την επεξεργασία διεκπεραιώσεων.

Οι *αποθήκες υλοποιήσεων και διασυνδέσεων* (implementation & interface repositories) συντηρούν πληροφορίες για τις υλοποιήσεις αντικειμένων και τις διασυνδέσεις, που είναι διαθέσιμες στο σύστημα. Ο αγωγός κλήσεων αντικειμένων χρησιμοποιεί τις πληροφορίες αυτές κατά τη διάρκεια της λειτουργίας του, σε περιπτώσεις όπως για παράδειγμα, όταν κάνει ενεργοποίηση αντικειμένων σε χρόνο εκτέλεσης. Η ενεργοποίηση χρειάζεται να γίνει σε οποιοδήποτε αντικείμενο πρόκειται να λάβει μία κλήση, αλλά η υλοποίηση αυτού δε βρίσκεται σε κατάσταση εκτέλεσης. Τέλος, η διασύνδεση του αγωγού κλήσεων αντικειμένων υποστηρίζει λειτουργίες γενικής φύσεως για χρήση από τους πελάτες και τις υλοποιήσεις αντικειμένων. Λειτουργίες, όπως για παράδειγμα, την αρχικοποίηση του αγωγού κλήσεων ή την εύρεση ενός αρχικού συνόλου αναφορών απαραίτητου για την εκκίνηση λειτουργίας μιας εφαρμογής.

2.7.4 Η Γλώσσα Ορισμού Διασύνδεσης (Interface Definition Language)

Στις εφαρμογές CORBA, που συμμορφώνονται στα πρότυπα του OMG, οι διασυνδέσεις αντικειμένων περιγράφονται με μία *Γλώσσα Ορισμού Διασύνδεσης* (IDL), όπως αυτή που προδιαγράφεται στα πρότυπα του OMG. Έτσι, μία περιγραφή IDL ορίζει μία ή περισσότερες διασυνδέσεις, τις μεθόδους που περιλαμβάνονται σε αυτές και την ακριβή μορφή των μεθόδων. Ουσιαστικά δηλαδή, η IDL παρέχει στους μηχανικούς λογισμικού ένα τυποποιημένο τρόπο περιγραφής των εξωτερικών χαρακτηριστικών των αντικειμένων. Υποστηρίζει και στατικούς (ακέραιοι, συμβολοσειρές κ.α.) αλλά και δομημένους (δομές, ενώσεις κ.α.) τύπους δεδομένων. Η σύνταξη της IDL μοιάζει με αυτή της C++.

Επίσης, η IDL του OMG υποστηρίζει την επαναχρησιμοποίηση ήδη ορισθέντων διασυνδέσεων μέσα από ένα μηχανισμό κληρονομικότητας αυτών. Η κληρονομούσα διασύνδεση διαθέτει όλα τα στοιχεία της κληροδοτούσας, όπως μεθόδους και ορισμούς τύπων και εξαιρέσεων. Υποστηρίζεται επίσης και η πολλαπλή κληρονομικότητα, η οποία δίνει τη δυνατότητα στις διασυνδέσεις να κληρονομούν από περισσότερες της μιας ήδη ορισθείσες διασυνδέσεις. Παρόλα αυτά, είναι σημαντικό να τονιστεί ότι η κληρονομικότητα διασυνδέσεων υποστηρίζει την επαναχρησιμοποίηση μόνο των διασυνδέσεων και όχι των υλοποιήσεων. Η κληρονομικότητα υλοποιήσεων μπορεί να υποστηρίζεται στο πλαίσιο της γλώσσας προγραμματισμού, που χρησιμοποιείται για την υλοποίηση (π.χ. C++ ή Java).

2.7.5 Αντιστοιχήσεις γλωσσών

Μία αντιστοίχηση γλώσσας καθορίζει τον τρόπο μετάφρασης από την IDL σε μία γλώσσα προγραμματισμού. Πιο συγκεκριμένα, μία αντιστοίχηση γλώσσας περιγράφει τον τρόπο αναπαράστασης των τύπων δεδομένων της IDL, το πως καλούνται οι μέθοδοι της IDL και το πως υλοποιούνται τα αντικείμενα CORBA στην εν λόγω γλώσσα προγραμματισμού. Ο σκοπός είναι η γεφύρωση του σημασιολογικού κενού μεταξύ των περιγραφών IDL και του μεταφρασμένου κώδικα προγράμματος. Αυτό επιτρέπει στους προγραμματιστές να χρησιμοποιούν το περιβάλλον CORBA ως αν ήταν ένα μέρος του προγραμματιστικού

περιβάλλοντος, στο οποίο εργάζονται. Έτσι για παράδειγμα, στην αντιστοίχηση C++ τα αντικείμενα CORBA μεταφράζονται σε αντικείμενα της C++ και οι εξαιρέσεις της IDL μεταφράζονται σε εξαιρέσεις της C++.

Όταν αναπτύσσεται μία εφαρμογή CORBA, ένας *προ-μεταφραστής* μετατρέπει αυτόματα τις περιγραφές IDL στις κατάλληλες διατυπώσεις της επιλεγείσας γλώσσας υλοποίησης. Αυτό σημαίνει ότι οι προγραμματιστές δε χρειάζεται να γνωρίζουν τις λεπτομέρειες της αντιστοίχησης, που χρησιμοποιούν. Η ύπαρξη προκαθορισμένων αντιστοιχήσεων γλώσσας διασφαλίζει τη μεταφερσιμότητα στο επίπεδο του πηγαίου κώδικα. Επιπλέον, οι αντιστοιχήσεις γλωσσών είναι σχεδιασμένες με τέτοιο τρόπο, ώστε να υποστηρίζουν την αξιόπιστη μεταφορά κλήσεων μεθόδων, μέσα από διαφορετικές γλώσσες προγραμματισμού.

Ακόμη και στην ίδια εφαρμογή, είναι δυνατό να γίνεται χρήση περισσοτέρων της μιας γλώσσας υλοποίησης. Έτσι για παράδειγμα, ένας πελάτης σε C θα μπορούσε να καλέσει μεθόδους ενός αντικειμένου υλοποιημένου σε COBOL. Για τη διασφάλιση της συμβατότητας μεταξύ της υλοποίησης του πελάτη και του αντικειμένου που καλείται, πρώτα γράφεται η διασύνδεση IDL και ακολούθως μεταφράζεται από προ-μεταφραστή τόσο σε C όσο και σε COBOL. Οι αντιστοιχήσεις γλωσσών διασφαλίζουν το γεγονός ότι η κλήση που εκπορεύεται από τον πελάτη που είναι γραμμένος σε C, μεταφέρεται στην υλοποίηση αντικειμένου που είναι γραμμένη σε COBOL, χωρίς απώλεια πληροφοριών. Το OMG έχει ορίσει αντιστοιχήσεις γλωσσών για τις C, C++, Smalltalk, Ada, Cobol και Java, ενώ συνεχίζεται η ανάπτυξη επιπλέον αντιστοιχήσεων.

2.7.6 Αλληλοδραστικότητα (interoperability)

Η *αρχιτεκτονική αλληλοδραστικότητας CORBA* παρέχει ένα πλαίσιο και ένα σύνολο τεχνικών για την επίτευξη αλληλοδραστικότητας μεταξύ ORB προερχόμενων από διαφορετικούς κατασκευαστές και άλλων αντικειμενοστρεφών τεχνολογιών (EJBeans, MS-DCOM κ.λ.π.). Η αρχιτεκτονική αυτή υποδιαιρεί ένα λογισμικό κατανεμημένης αρχιτεκτονικής σε *πεδία* (domains), σε κάθε ένα από τα οποία, τα αντικείμενα διαθέτουν κάποιο κοινό χαρακτηριστικό ή συμμορφώνονται σε ένα σύνολο κοινών κανόνων. Τυπικές περιπτώσεις πεδίων είναι:

- το πεδίο αναφοράς (τα όρια ισχύος μιας αναφοράς αντικειμένου),
- το πεδίο απεικόνισης (τα όρια ισχύος ενός συγκεκριμένου πρωτοκόλλου),
- τα πεδίο ασφάλειας (τα όρια ισχύος μιας συγκεκριμένης πολιτικής ασφάλειας) και
- το πεδίο τύπου (τα όρια ισχύος ενός συγκεκριμένου τύπου)

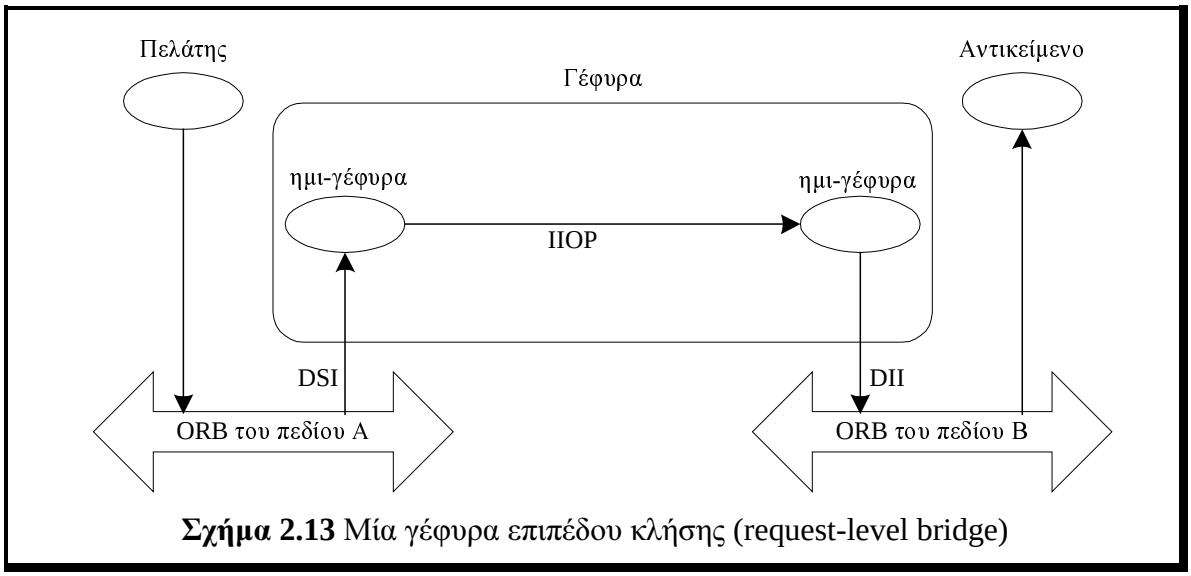
Ένα πεδίο είναι δυνατό να εκτείνεται σε περισσότερους του ενός ORB. Αν μία κλήση μεθόδου υπερβαίνει τα όρια ενός πεδίου, τότε πρέπει να περάσει από μία *γέφυρα*, που διασφαλίζει τη σωστή αντιστοίχιση των στοιχείων της κλήσης και της σημασιολογίας των δύο πεδίων. Υπάρχουν δύο δυνατές περιπτώσεις γεφύρωσης: η *άμεση γεφύρωση* και η *έμμεση γεφύρωση*. Η πρώτη προσέγγιση είναι η βέλτιστη και χρησιμοποιείται πιο συχνά, όταν ένας μόνο ORB υποδιαιρείται σε έναν αριθμό διαφορετικών πεδίων. Η δεύτερη προσέγγιση είναι πιο γενική και αναφέρεται στη μετατροπή των στοιχείων σε μία ενδιάμεση κοινά αποδεκτή μορφή μέσω της χρήσης *ημι-γέφυρας*.

Η αρχιτεκτονική αλληλοδραστικότητας υποστηρίζει δύο τεχνικές υλοποίησης της γεφύρωσης μεταξύ δύο ORB: την *in-line γεφύρωση* και την *γεφύρωση επιπέδου κλήσης* (request - level bridging). Η in-line γεφύρωση κάνει χρήση των εσωτερικών διασυνδέσεων προγραμματιμού εφαρμογών του κάθε ORB, έχει καλύτερη απόδοση, αλλά προϋποθέτει πρόσβαση στις λεπτομέρειες υλοποίησης και των δύο χρησιμοποιούμενων ORB. Από την άλλη, μία γέφυρα

επιπέδου κλήσης είναι δυνατό να υλοποιηθεί από ανεξάρτητο κατασκευαστή, δέχεται τυχαίες κλήσεις μέσω της δυναμικής διασύνδεσης σκελετού, κάνει τις απαραίτητες μετατροπές και τέλος διοχετεύει τη νέα μορφή της κλήσης στο άλλο πεδίο μέσω της δυναμικής διασύνδεσης κλήσης.

Για τη διασφάλιση της αλληλοδραστικότητας μεταξύ διαφορετικών τεχνολογικών πεδίων και μεταξύ διαφορετικών προϊόντων ORB το OMG έχει αναπτύξει τα εξής τρία πρωτόκολλα επικοινωνίας:

- το *γενικό inter-ORB πρωτόκολλο* (GIOP), που έχει σχεδιαστεί για την αποστολή κλήσεων μεθόδων και των αποκρίσεων αυτών μεταξύ δύο ORBs,
- το *Internet inter-ORB πρωτόκολλο* (IIOP), που περιγράφει πως το GIOP υλοποιείται πάνω από συνδέσεις TCP/IP και
- το *κοινό DCE inter-ORB πρωτόκολλο* (DCE-CIOP), που περιγράφει την επικοινωνία δύο ORBs πάνω από μία υποδομή DCE (Distributed Computing Environment).



Σχήμα 2.13 Μία γέφυρα επιπέδου κλήσης (request-level bridge)

Στο σχήμα 2.13 απεικονίζεται, πως μπορεί να συνδυασθεί μία γέφυρα επιπέδου κλήσης και το πρωτόκολλο IIOP για τη γεφύρωση δύο τυχαίων ORB. Η διάταξη αυτή είναι αρκετά γενική και μπορεί να υλοποιηθεί με τη χρήση μόνο των εξωτερικών διασυνδέσεων των ORB, που χρησιμοποιούνται· είναι όμως και αρκετά πολύπλοκη και μπορεί να προκαλέσει προβλήματα απόδοσης. Στην [ZUS96], οι συγγραφείς διαπιστώνουν ότι οι μεταπληροφορίες, των τύπων των δεδομένων της κλήσης που υφίσταται δύο διαδοχικές μετατροπές, μπορεί να έχουν σημαντική αρνητική επίδραση στην απόδοση της γέφυρας.

2.7.7 Επικοινωνία αντικειμένων CORBA

Τα κατανεμημένα αντικείμενα όταν επικοινωνούν μεταξύ τους μέσω μιας υποδομής αγωγού κλήσεων (ORB), κάνουν χρήση [VIN97] ενός από τους τρεις τύπους επικοινωνίας που υποστηρίζουν οι προδιαγραφές CORBA και είναι:

- Η *σύγχρονη κλήση*, κατά την οποία ο αποστολέας δρομολογεί την κλήση και σταματάει τη ροή επεξεργασίας αναμένοντας για τη λήψη της απόκρισης.
- Η *παρατεινόμενη σύγχρονη κλήση*, που διαφέρει από την απλή σύγχρονη κλήση στο γεγονός ότι ο αποστολέας συνεχίζει την επεξεργασία μέχρις ότου είναι έτοιμος για να λάβει την απόκριση, την οποία στο σημείο εκείνο περιμένει.

- Η *ασύγχρονη κλήση*, γνωστή επίσης και ως κλήση μιας κατεύθυνσης με χαρακτηριστικό το γεγονός ότι δε συνοδεύεται από απόκριση.

Οι σύγχρονες και ασύγχρονες κλήσεις μπορεί να είναι *στατικές* ή *δυναμικές*. Οι στατικές κλήσεις προωθούνται μέσω του στελέχους του αποστολέα στο σκελετό του παραλήπτη, ενώ οι δυναμικές κλήσεις προωθούνται μέσω της δυναμικής διασύνδεσης κλήσης του αποστολέα στην αντίστοιχη διασύνδεση του παραλήπτη. Οι παρατεινόμενες σύγχρονες κλήσεις είναι μόνο δυναμικές.

2.8 Άλλες τεχνολογίες middleware διεργασιακής επικοινωνίας

Στην παράγραφο 2.7 έγινε μία εκτεταμένη περιγραφή της τεχνολογίας CORBA, η οποία όμως δεν είναι και η μοναδική του είδους της. Κατά την εμφάνιση της τεχνολογίας CORBA βρισκόταν ήδη σε χρήση η επονομαζόμενη *DCE* (Distributed Computing Environment), η οποία έκανε χρήση της γνωστής τότε ως *Απομακρυσμένης Κλήσης Διαδικασιών* (Remote Procedure Call) για την επικοινωνία μεταξύ προγραμμάτων, που εκτελούνται σε διαφορετικές διεργασίες. Οι προδιαγραφές CORBA έχουν δανεισθεί πολλά από αυτές του DCE, αλλά υπερτερούν λόγω της αντικειμενοστρεφούς αρχιτεκτονικής, που εισάγουν.

Παρόλα αυτά, δεν είναι και η μόνη αντικειμενοστρεφής τεχνολογία διεργασιακής επικοινωνίας, καθώς υπάρχει απέναντί της το *Common Object Model* (COM) και το *DCOM* της Microsoft. Οι προδιαγραφές της Microsoft παρουσιάζουν αρκετές αναλογίες με αυτές της τεχνολογίας CORBA, αλλά και διαφορές, που όμως δεν είναι στους σκοπούς αυτής της διατριβής η διερεύνησή τους.

Τα DCE, CORBA και DCOM σχεδιάσθηκαν αρχικά για την εξυπηρέτηση σύγχρονων κλήσεων επικοινωνίας, χωρίς να δοθεί ιδιαίτερη έμφαση στην ανάγκη εξυπηρέτησης ασύγχρονων κλήσεων. Είναι χαρακτηριστικό ότι στις προδιαγραφές CORBA το κενό της ασύγχρονης επικοινωνίας το καλύπτουν οι λεγόμενες *κλήσεις μιας κατεύθυνσης*, οι κλήσεις δηλαδή, που δεν επιστρέφουν αποτέλεσμα.

Μία περισσότερο προωθημένη λύση ασύγχρονης επικοινωνίας προσφέρουν τα προϊόντα *middleware ανταλλαγής μηνυμάτων*. Όταν γίνεται χρήση ενός τέτοιου προϊόντος, ο πελάτης αποθηκεύει το μήνυμά του σε μία ουρά αναμονής και η επεξεργασία του από το διακομιστή γίνεται, όταν αυτός είναι διαθέσιμος. Αντίστοιχα, ο διακομιστής αποθηκεύει το αποτέλεσμα της επεξεργασίας σε μία άλλη ουρά αναμονής από την οποία παραλαμβάνεται από τον πελάτη, όταν επίσης αυτός γίνει διαθέσιμος. Η φιλοσοφία της λειτουργίας του middleware ανταλλαγής μηνυμάτων το καθιστά ιδανικό για εφαρμογές επεξεργασίας διεκπεραιώσεων, αφού η επίδραση της ουράς μηνυμάτων μπορεί εύκολα να αποκτήσει σημασία διεκπεραίωσης. Πρόσφατα επίσης, η τεχνολογία αυτή έχει προσελκύσει και το ενδιαφέρον της έρευνας και ανάπτυξης διατάξεων *νομαδικής επεξεργασίας* (nomadic computing). Στις διατάξεις αυτές η υλοποίηση δυνατοτήτων ασύρματης λειτουργίας προϋποθέτει τη χρήση ασύγχρονης επικοινωνίας, καθώς οι κινητοί υπολογιστές δεν έχουν πάντα τη δυνατότητα σύνδεσης, έτσι ώστε να γίνεται χρήση σύγχρονης επικοινωνίας.

Τέλος, θα ήταν παράλειψη να μη γίνει αναφορά στο middleware *αυτοδρώντων αντικειμένων* (agents) ή αλλιώς *πράκτορες*. Όταν λοιπόν πραγματοποιείται επικοινωνία πελάτη - διακομιστή σε ένα τέτοιο περιβάλλον, δημιουργείται πρώτα η περίπτωση (instantiation) ενός αυτοδρώντος αντικειμένου, ακολούθως αποστέλλεται αυτό στον κόμβο του διακομιστή, όπου και διεξάγει την επιθυμητή αλληλεπίδραση μετά από την οποία επιστρέφει στον πελάτη. Ένα τέτοιο middleware είναι το περιβάλλον Voyager της ObjectSpace, το οποίο είναι και το πιο ευρέως διαδεδομένο.

Η διαρκώς αυξανόμενη ποικιλομορφία του λογισμικού middleware εγείρει ολοένα και περισσότερα ερωτηματικά, όσον αφορά τα ποιοτικά χαρακτηριστικά των νέων εφαρμογών. Έτσι, η συστηματική μελέτη της απόδοσης αυτών καθίσταται πλέον ένα ερευνητικό πεδίο μείζονος σημασίας, καθώς το σύνολο των παραγόντων, που την επηρεάζουν, περιλαμβάνει πλέον:

- τη φυσική θέση των αντικειμένων, που απαρτίζουν την εφαρμογή,
- τη συχνότητα της μεταξύ τους επικοινωνίας,
- τον τύπο της χρησιμοποιούμενης επικοινωνίας (σύγχρονη ή ασύγχρονη) και το middleware μέσω του οποίου αυτή πραγματοποιείται,
- το μέγεθος των δεδομένων, που μεταβιβάζονται σε μία αλληλεπίδραση,
- το χρόνο επεξεργασίας από το απομακρυσμένο αντικείμενο και
- τη χρήση ή μη πολιτικών πολυνημάτωσης αντικειμένων.

2.9 Κατανεμημένες εφαρμογές παγκοσμίου ιστού

Οι εφαρμογές παγκοσμίου ιστού παρέχουν ένα μεγάλο μέρος της λειτουργικότητας των συμβατικών εφαρμογών, χρησιμοποιώντας όμως ως διεπαφή ένα κατάλληλα σχεδιασμένο δικτυακό τόπο. Οι ελκυστικά σχεδιασμένες σελίδες με κάρτες αγοράς ή ηλεκτρονικά προσβάσιμους καταλόγους παρέχουν ουσιαστικά ένα μέσο αποστολής αιτήσεων διεκπεραίωσης.

Η αρχιτεκτονική των εφαρμογών αυτών περιλαμβάνει συνήθως ένα στρώμα «πρόσοψης», που φιλοξενεί το φυλλομετρητή ιστοσελίδων και ένα στρώμα «βάθρο», που παρέχει τους πόρους δεδομένων. Περιλαμβάνει επίσης ένα ή περισσότερα ενδιάμεσα στρώματα, που παρέχουν λειτουργίες διακομιστή ιστοσελίδων και άλλων εφαρμογών.

Η λειτουργία των εφαρμογών παγκοσμίου ιστού είναι φαινομενικά απλή. Στην πιο στοιχειώδη μορφή, βασίζεται στη χρήση ενός φυλλομετρητή ιστοσελίδων στον κόμβο του χρήστη, ενός διακομιστή ιστοσελίδων σε κάποιο άλλο κόμβο και ενός συνόλου ιστοσελίδων γραμμένων σε HTML. Όταν ο χρήστης επιλέγει ένα σύνδεσμο της ιστοσελίδας, που εκείνη τη στιγμή προβάλλεται στο παράθυρο του φυλλομετρητή, ανοίγει μία *σύνδεση* με το διακομιστή και μέσω αυτής μεταβιβάζεται η κλήση μιας νέας σελίδας. Ο διακομιστής αναλύει την κλήση, ανασύρει την κατάλληλη σελίδα και τη μεταβιβάζει στο φυλλομετρητή. Όταν ολοκληρωθεί η εκτέλεση της κλήσης, παύει και η σύνδεση που χρησιμοποιήθηκε γι΄ αυτή. Με τον τρόπο αυτό, παρέχεται πρόσβαση σε όλες τις σελίδες, που περιλαμβάνουν στατικό περιεχόμενο.

Όταν όμως προκύπτει η ανάγκη χρήσης ενός επιπλέον στρώματος για την παροχή δυναμικού περιεχομένου, η κατάσταση περιπλέκεται. Οι δυναμικές αυτές σελίδες μπορεί να εμπεριέχουν τα αποτελέσματα ενός ερωτήματος ή μίας κλήσης ενημέρωσης κάποιας βάσης δεδομένων, τα αποτελέσματα της επεξεργασίας μιας online παραγγελίας ή οτιδήποτε άλλο καθιστά έτσι ένα δικτυακό τόπο δικτυακή εφαρμογή.

Για την παροχή λοιπόν δυναμικού περιεχομένου μπορεί να επιλεγεί κάποια από τις λύσεις, που ακολουθούν, ή ένας συνδυασμός αυτών:

- Χρήση ενός προ-επεξεργαστή/διερμηνευτή των σεναρίων (scripts), που μπορεί να περιέχονται μέσα στις ιστοσελίδες και έχουν τη δυνατότητα παραγωγής δυναμικού περιεχομένου, πιθανότατα μετά από προσπέλαση σε κάποια βάση

δεδομένων. Παραδείγματα τέτοιων τεχνολογιών είναι οι Active Server Pages (ASP) και οι Java Server Pages (JSP).

- Χρήση μιας κοινής διασύνδεσης πύλης (CGI) για την εκτέλεση εξωτερικών ως προς το διακομιστή διεργασιών. Διακρίνουμε δύο δυνατές περιπτώσεις: i) αυτή της δημιουργίας μιας νέας διεργασίας και ii) αυτή της χρήσης μιας ήδη εκτελούμενης διεργασίας παρασκηνίου στην οποία απλά προωθείται η κάθε νέα κλήση εξυπηρέτησης.
- Χρήση Java μικροεφαρμογών διακομιστή των επονομαζόμενων servlets.
- Χρήση πύλης διακομιστή (SSI), που δεν οδηγεί στη δημιουργία νέας διεργασίας όπως η πύλη CGI· ωστόσο, προϋποθέτει μία επιπλέον ανάλυση της HTML σελίδας από το διακομιστή πριν αυτός τη στείλει στο φυλλομετρητή. Αυτό προσθέτει επιπλέον κόστος επεξεργασίας και έχει ως αποτέλεσμα μία συγκριτικά λιγότερο αποδοτική λύση.
- Χρήση ειδικών πυλών, που υλοποιούνται συνήθως από κατασκευαστές συστημάτων βάσης δεδομένων με τη χρήση της προγραμματιστικής διασύνδεσης εφαρμογής του διακομιστή. Οι λύσεις αυτές χρησιμοποιούνται συνήθως ως πύλες διοχέτευσης SQL εντολών σε βάσεις και είναι ταχύτερες τόσο από τα CGIs, όσο και από τα SSIs.

Η εισαγωγή του επιπλέον στρώματος επεξεργασίας της εφαρμογής, για την παροχή δυναμικού περιεχομένου, δημιουργεί νέες απαιτήσεις στη μελέτη της απόδοσης αυτής, καθώς διευρύνεται το σύνολο των παραγόντων, που την επηρεάζουν. Ενδεικτικά αναφέρουμε:

- το φόρτο του διακομιστή ιστοσελίδων,
- το μηχανισμό εκτέλεσης, που επιλέγεται για την παραγωγή του δυναμικού περιεχομένου,
- τη θέση των εκτελέσιμων προγραμμάτων στους κόμβους επεξεργασίας,
- τους μηχανισμούς προσπέλασης των βάσεων,
- τους μηχανισμούς παροχής ασφάλειας δεδομένων,
- τη θέση της βάσης,
- τις πιθανές αλληλεπιδράσεις με άλλου τύπου middleware και
- τις πιθανές καθυστερήσεις για το «κατέβασμα» μικροεφαρμογών.

Κεφάλαιο 3 *Μοντέλα Εκτίμησης Απόδοσης Συστημάτων*

Στο κεφάλαιο αυτό γίνεται παρουσίαση ευρέως χρησιμοποιούμενων *τεχνικών μοντελοποίησης* απόδοσης συστημάτων, όπως η *στοιχειώδης λειτουργική ανάλυση* και τα *δίκτυα ουρών* με την αναλυτική ή προσεγγιστική τους επίλυση. Στη συνέχεια γίνεται αναφορά σε δύο μοντέλα ουρών για λογισμικό κατανεμημένης αρχιτεκτονικής. Τέλος, γίνεται σύντομη περιγραφή άλλων τεχνικών, όπως τα *δίκτυα Petri* (Petri nets), τα *στρωματοποιημένα δίκτυα ουρών* (Layered Queueing Networks) και οι *στοχαστικές διεργασιακές άλγεβρες*.

3.1 Δίκτυα Ουρών (Queueing Networks)

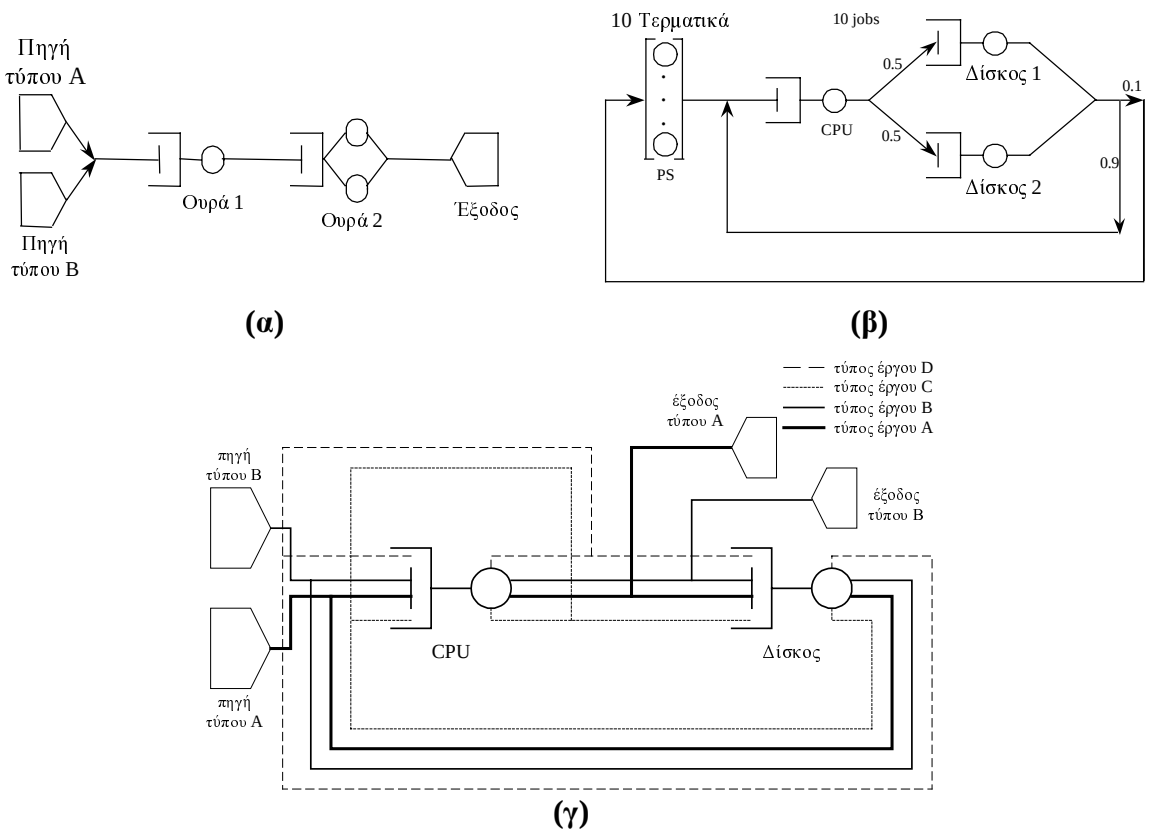
Σε κάθε υπολογιστικό σύστημα έχουμε ένα σύνολο *έργων* (jobs), που όλα διαμοιράζονται τους ίδιους πόρους όπως λ.χ. τη CPU, τους δίσκους ή άλλες μονάδες, το εύρος δικτύου κ.α. Καθώς οι πόροι χρησιμοποιούνται από περιορισμένο αριθμό έργων σε δεδομένη χρονική στιγμή, όλα τα άλλα έργα, που απαιτούν εξυπηρέτηση από κάποιο απασχολημένο πόρο, βρίσκονται σε *ουρά αναμονής*. Για το λόγο αυτό, η διάδοση της χρήσης μοντέλων ουρών για την ανάλυση απόδοσης υπολογιστικών συστημάτων φαντάζει ως μία φυσιολογική εξέλιξη. Η βιβλιογραφία του χώρου είναι πολύ μεγάλη, καθώς κάθε χρόνο δημοσιεύονται μερικές εκατοντάδες εργασίες. Ευτυχώς πάντως, το μεγαλύτερο ποσοστό προβλημάτων απόδοσης μπορεί να αντιμετωπιστεί με τη χρήση μόνο ενός μικρού αριθμού τεχνικών· μία σύντομη αναφορά γίνεται στην παράγραφο αυτή.

Κάθε υπολογιστικό σύστημα αναπαριστάται από ένα *δίκτυο ουρών*. Το δίκτυο ουρών είναι ένα σύνολο διασυνδεδεμένων *κέντρων εξυπηρέτησης*, τα οποία αναπαριστούν τους πόρους του συστήματος και τα έργα, που εξυπηρετούνται από αυτά, που συχνά στη βιβλιογραφία αναφέρονται και ως πελάτες. Ανάλογα με τον τύπο του *υπολογιστικού φόρτου*, τα έργα που κυκλοφορούν σε ένα δίκτυο ουρών έχουν διαφορετική έννοια. Πιο συγκεκριμένα:

- Κατά την *αλληλεπιδραστική επεξεργασία* (interactive), έχουμε έργα επεξεργασίας, που ενδεχομένως δημιουργούνται από έναν αριθμό προσωπικών υπολογιστών δικτύου ή σταθμών εργασίας, με δεδομένο *χρόνο προσμονής* (think time) σε αυτούς. Η αλληλεπίδραση είναι ένας συνδυασμός χρόνων αναμονής για χρήση πόρων, χρόνων επεξεργασίας και χρόνων προσμονής. Ο χρήστης εναλλάσσεται

ανάμεσα σε μία κατάσταση αναμονής της απόκρισης της αλληλεπίδρασης και σε μία κατάσταση «σκέψης» μέχρι την υποβολή της επόμενης εντολής ή την επόμενη χρήση του ποντικιού. Ένα παράδειγμα τέτοιας εφαρμογής είναι η περίπτωση μιας εκπαιδευτικής εφαρμογής εγκατεστημένης λ.χ. σε ένα εσωτερικό δίκτυο.

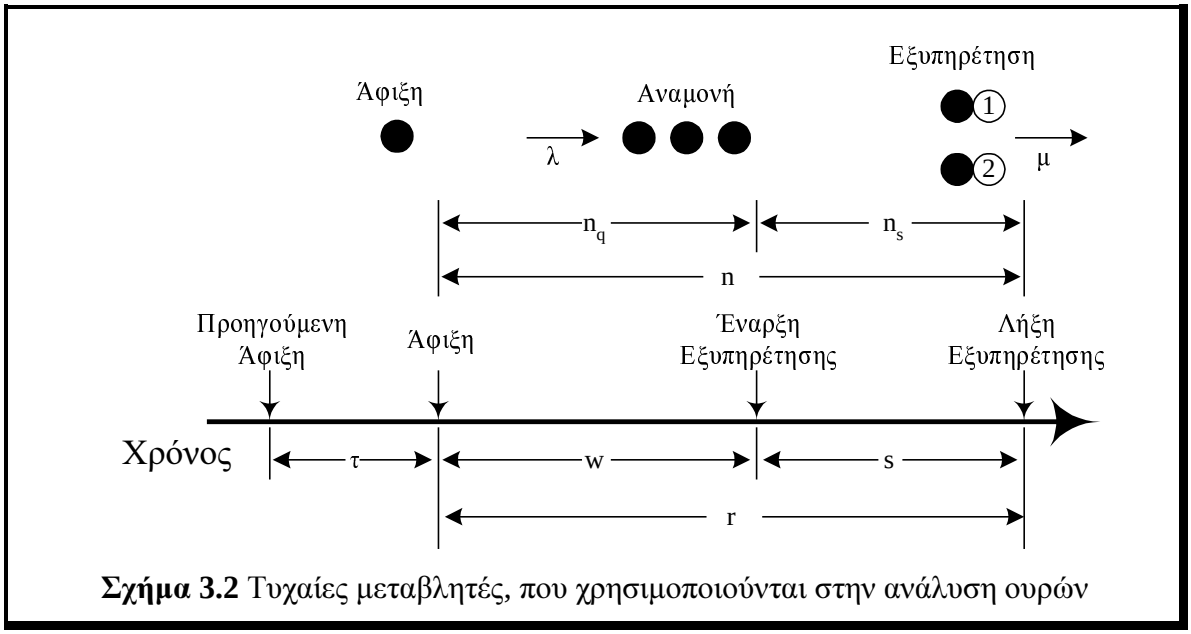
- Στην περίπτωση της *επεξεργασίας διεκπεραίωσης*, έχουμε έργα επεξεργασίας, τα οποία μπορούν για παράδειγμα να αναπαριστούν διεκπεραιώσεις ή κλήσεις εξυπηρέτησης και βασικό τους χαρακτηριστικό είναι το γεγονός ότι καταφθάνουν στο υπολογιστικό σύστημα με συγκεκριμένο *ρυθμό άφιξης*. Ένα τέτοιο παράδειγμα είναι η περίπτωση μιας εφαρμογής ηλεκτρονικού εμπορίου, μέσω της οποίας καταφθάνουν σε ένα διακομιστή ιστοσελίδων παραγγελίες προϊόντων.
- Τέλος, μία *επεξεργασία σε ομάδες έργων* (batch processing), περιγράφεται από τον αριθμό των ενεργών έργων επεξεργασίας μέσα στο σύστημα. Παράδειγμα επεξεργασίας σε ομάδες έργων είναι η λειτουργία κάποιας διεργασίας παρασκηνίου (daemon), που είτε βρίσκεται σε διαρκή λειτουργία περιμένοντας την εμφάνιση κάποιου συμβάντος, είτε εκτελεί κάποιες καθορισμένες εργασίες σε περιοδική βάση. Εδώ το χαρακτηριστικό είναι το ότι δεν υπάρχει αλληλεπίδραση με τερματικό ή σταθμό εργασίας.



Σχήμα 3.1 (α) Ένα ανοικτό δίκτυο ουρών με δύο διαφορετικούς τύπους έργων
(β) Μοντέλο κεντρικής επεξεργασίας (Brown et al) ως κλειστό δίκτυο ουρών
(γ) Μικτό δίκτυο ουρών με τέσσερις τύπους έργων

Τα δίκτυα ουρών διακρίνονται σε *ανοικτά* και *κλειστά*. Σε ένα ανοικτό δίκτυο ουρών έχουμε εξωτερικές αφίξεις και αναχωρήσεις και άρα ο αριθμός των έργων, που υπάρχουν στο σύστημα, μεταβάλλεται με το χρόνο. Έτσι, στην ανάλυση ενός ανοικτού συστήματος υποθέτουμε ότι η παραγωγή (throughput), ο αριθμός των έργων δηλαδή που εξυπηρετούνται στη μονάδα του χρόνου, είναι γνωστή - ίση με το ρυθμό άφιξης - και η μελέτη

επικεντρώνεται μεταξύ άλλων στην κατανομή του αριθμού των έργων στο σύστημα. Τα κλειστά δίκτυα ουρών δεν έχουν εξωτερικές αφίξεις ή αναχωρήσεις. Ο συνολικός αριθμός έργων στο σύστημα είναι σταθερός και η παραγωγή είναι ενδεχομένως ένα από τα μέτρα απόδοσης, που είναι επιθυμητό να μελετηθούν. Τέλος, υπάρχει και η περίπτωση των *μικτών δικτύων ουρών*, δικτύων δηλαδή που αποτελούνται από περισσότερους του ενός τύπους υπολογιστικών φόρτων, κάποιοι από τους οποίους αναπαριστώνται ως ανοικτές «αλυσίδες» και κάποιοι ως κλειστές και διαπλέκονται στο ίδιο δίκτυο ουρών. Κάθε «αλυσίδα» απεικονίζει ουσιαστικά την πορεία των έργων επεξεργασίας ενός συγκεκριμένου *τύπου* μέσα στο δίκτυο ουρών. Όλα τα έργα που ανήκουν στον ίδιο τύπο έχουν πάντα τις ίδιες απαιτήσεις χρόνου ανά κέντρο εξυπηρέτησης κατά την πορεία τους μέσα στο δίκτυο, η οποία επίσης ορίζεται με τις ίδιες πιθανότητες μετάβασης από κέντρο σε κέντρο.



Σχήμα 3.2 Τυχαίες μεταβλητές, που χρησιμοποιούνται στην ανάλυση ουρών

Ξεκινώντας από την πιο απλή περίπτωση δικτύου ουρών, του ανοικτού δηλαδή δικτύου, που αποτελείται από ένα μόνο κέντρο εξυπηρέτησης, τα χαρακτηριστικά, που περιγράφουν πλήρως ένα τέτοιο σύστημα (σχήμα 3.2), είναι:

- Η *διαδικασία άφιξης*: Αν υποθέσουμε ότι τα έργα επεξεργασίας καταφθάνουν τις χρονικές στιγμές t_1, t_2, . . ., t_j, τότε οι τυχαίες μεταβλητές $\tau_j = t_j - t_{j-1}$ ονομάζονται *ενδιάμεσοι χρόνοι αφίξεων* (interarrival times). Γενικά στις περισσότερες περιπτώσεις μελέτης τέτοιων συστημάτων, γίνεται δεκτή η υπόθεση ότι οι χρόνοι τ_j σχηματίζουν μία σειρά από ανεξάρτητες και όμοια κατανεμημένες τυχαίες μεταβλητές. Η πιο ευρέως χρησιμοποιούμενη διαδικασία άφιξης, είναι η επονομαζόμενη *άφιξη Poisson*, κατά την οποία οι ενδιάμεσοι χρόνοι αφίξεων είναι ανεξάρτητοι και όμοια κατανεμημένοι σύμφωνα με την εκθετική κατανομή. Άλλες συχνά χρησιμοποιούμενες κατανομές είναι η κατανομή Erlang και η υπερεκθετική κατανομή. Τέλος, όταν κάποια αποτελέσματα ισχύουν για όλες τις κατανομές ενδιάμεσων χρόνων άφιξης, τότε λέμε ότι ισχύουν για τη γενική κατανομή.
- Η *κατανομή χρόνων εξυπηρέτησης*: Εκφράζει το χρόνο, που κάποιο έργο απαιτεί στο σημείο εξυπηρέτησης, που βρίσκεται. Συνήθως οι χρόνοι αυτοί θεωρούνται τυχαίες μεταβλητές, ανεξάρτητες και όμοια κατανεμημένες. Η πιο συχνά χρησιμοποιούμενη κατανομή είναι και εδώ η εκθετική, παρόλο που έχουν

χρησιμοποιηθεί και κατανομές Erlang και υπερεκθετικές. Όταν γίνεται αναφορά σε γενική κατανομή, τότε τα αποτελέσματα ισχύουν για όλες τις κατανομές.

- Ο *αριθμός σημείων εξυπηρέτησης*: Εκφράζει τη δυνατότητα ταυτόχρονης εξυπηρέτησης από τον πόρο, ο οποίος αντιπροσωπεύεται από την ουρά αναμονής. Αν αυτή υπάρχει, τότε τα σημεία εξυπηρέτησης θεωρούνται ομοίων δυνατοτήτων. Αν παρόλα αυτά, τα σημεία εξυπηρέτησης δεν είναι όμοια, τότε ομαδοποιούνται σε σύνολα σημείων εξυπηρέτησης ομοίων δυνατοτήτων και η κάθε ομάδα διαθέτει και τη δικιά της ουρά αναμονής.

- Η *χωρητικότητα της ουράς*: Είναι ο μέγιστος αριθμός έργων επεξεργασίας, που μαζί με αυτά που εξυπηρετούνται μπορούν να περιληφθούν στην ουρά αναμονής. Πρακτικά, στις περισσότερες περιπτώσεις η χωρητικότητα είναι πεπερασμένη. Αυτό σημαίνει ότι όταν η ουρά φθάσει να περιέχει αριθμό έργων ίσο με τη χωρητικότητά της, τότε στην επόμενη άφιξη παρουσιάζεται το φαινόμενο του *αποκλεισμού*. Η τύχη, που θα έχει το επιπλέον έργο, εξαρτάται από το *μηχανισμό αποκλεισμού*, που έχει επιλεγεί. Βέβαια, όταν η χωρητικότητα είναι μεγάλη, είναι ευκολότερη η μελέτη κάτω από την υπόθεση της μη πεπερασμένης χωρητικότητας και η εν λόγω σύμβαση είναι αυτή, που συνήθως προτιμάται.

- Το *μέγεθος του πληθυσμού*: Είναι ο συνολικός αριθμός έργων, που εμφανίζονται. Και στην περίπτωση αυτή, ενώ πρακτικά το μέγεθος του πληθυσμού είναι πεπερασμένο, αν αυτό είναι μεγάλο, είναι ευκολότερη η ανάλυση κάτω από την υπόθεση του μη πεπερασμένου αριθμού έργων.

- Η *πειθαρχία εξυπηρέτησης*: Εκφράζει τη σειρά με την οποία εξυπηρετούνται τα έργα, που βρίσκονται συσσωρευμένα στην ουρά. Η πιο διαδεδομένη είναι η πειθαρχία First Come First Served (FCFS), όπου η εξυπηρέτηση γίνεται με τη σειρά άφιξης. Στην περίπτωση Last Come First Served (LCFS) η εξυπηρέτηση γίνεται αρχής γενομένης από το τελευταίο έργο, ενώ όταν έχουμε Last Come First Served Preemptive Resume (LCFSPR) γίνεται διακοπή του έργου, που ήδη βρίσκεται στο σημείο εξυπηρέτησης, προκειμένου να προωθηθεί για εξυπηρέτηση αυτό, που μόλις έχει καταφθάσει. Μία πειθαρχία, που χρησιμοποιείται ευρέως στη μοντελοποίηση των CPU, είναι η Round-Robin (RR), κατά την οποία γίνεται εκ περιτροπής εξυπηρέτηση των έργων της ουράς κάθε ένα από τα οποία λαμβάνει κάθε φορά καθορισμένη ποσότητα χρόνου εξυπηρέτησης. Όταν η ποσότητα αυτή είναι αρκετά μικρή σε σύγκριση με το μέσο χρόνο εξυπηρέτησης, τότε η πειθαρχία ονομάζεται Processor Sharing (PS) και αυτή ουσιαστικά οδηγεί σε ένα δίκαιο διαμοίρασμα του σημείου/ων εξυπηρέτησης σε όλα τα έργα της ουράς. Η πειθαρχία PS βρίσκει εφαρμογή και στην περίπτωση κέντρων με *άπειρα σημεία εξυπηρέτησης* (Infinite Server) ή αλλιώς *κέντρων καθυστέρησης* (delay centers), όπου πρακτικά δεν έχουμε καθόλου αναμονή σε ουρά. Τα κέντρα καθυστέρησης χρησιμοποιούνται συχνά για την αφαιρετική μοντελοποίηση μονάδων/συνδέσεων, όπως π.χ. τα τερματικά χρηστών σε έναν αλληλεπιδραστικό φόρτο με συγκεκριμένο μέσο χρόνο προσμονής, ή μία δορυφορική σύνδεση, όπου δεν υπάρχουν επιπλέον πληροφορίες για το φόρτο του πόρου, παρά μόνον ο μέσος χρόνος εξυπηρέτησης, που γίνεται αντιληπτός.

  Συχνά η πειθαρχία βασίζεται στους απαιτούμενους από τα έργα χρόνους εξυπηρέτησης. Παραδείγματα αποτελούν οι Shortest Processing Time First (SPTF) και Shortest Remaining Processing Time First (SRPTF). Τέλος, στην περίπτωση δικτύων ουρών με πολλαπλούς τύπους έργων είναι πολλές φορές επιθυμητή η εξασφάλιση καλύτερης ποιότητας υπηρεσίας για συγκεκριμένους

τύπους έργων. Στην περίπτωση αυτή είναι επιθυμητός ο ορισμός *πειθαρχίας προτεραιοτήτων* (priority scheduling) σε συνδυασμό ενδεχομένως με κάποια από τις πειθαρχίες, που προαναφέρθηκαν.

Για τον πλήρη ορισμό μιας ουράς αναμονής, ο συμβολισμός Kendall επιτρέπει τον καθορισμό όλων των παραπάνω παραμέτρων. Έτσι, μία ουρά M/M/3/23/136/FCFS χαρακτηρίζεται από αφίξεις Poisson, εκθετικά κατανεμημένους χρόνους στα 3 σημεία εξυπηρέτησης, που διαθέτει, η χωρητικότητά της όμως περιορίζεται στην ύπαρξη 23 το πολύ έργων μέσα στο σύστημα. Ο συνολικός αριθμός των έργων, που εξυπηρετούνται είναι 136 και η πειθαρχία εξυπηρέτησης είναι η FCFS. Ο συμβολισμός αυτός αφορά αφίξεις μεμονωμένων έργων επεξεργασίας. Για *μαζικές αφίξεις* Poisson θα χρησιμοποιούνταν ο συμβολισμός $M^{[x]}$, όπου το x αντιπροσωπεύει το μέγεθος της ομάδας, το οποίο συνήθως θεωρείται τυχαία μεταβλητή, που η κατανομή της ορίζεται χωριστά. Όταν οι τρεις τελευταίες παράμετροι δεν ορίζονται και δε γίνεται αναφορά σε αυτές, τότε μιλάμε για ουρά απεριόριστης χωρητικότητας, χωρίς περιορισμό στον αριθμό έργων, που εξυπηρετούνται και με πειθαρχία FCFS και σε αυτή την περίπτωση ο συμβολισμός G/G/1/∞/∞/FCFS γράφεται για συντομία με τη μορφή G/G/1.

Οι στοιχειώδεις μεταβλητές, που χρησιμοποιούνται στην ανάλυση ουρών φαίνονται στον πίνακα 3.1.

Πίνακας 3.1 Μεταβλητές που χρησιμοποιούνται στην ανάλυση ουρών

τ	ο ενδιάμεσος χρόνος μεταξύ δύο διαδοχικών αφίξεων	
λ	ο μέσος ρυθμός αφίξεων Σε κάποιες περιπτώσεις είναι συνάρτηση της κατάστασης του συστήματος. Έτσι, μπορεί π.χ. να εξαρτάται από τον αριθμό των έργων, που βρίσκονται στο σύστημα.	= 1/E[τ]
s	ο χρόνος εξυπηρέτησης έργου	
μ	ο μέσος ρυθμός εξυπηρέτησης ανά σημείο εξυπηρέτησης Όταν το σύστημα διαθέτει k σημεία εξυπηρέτησης ο συνολικός ρυθμός είναι $k \cdot \mu$.	= 1/E[s]
n	ο συνολικός αριθμός έργων στο κέντρο εξυπηρέτησης· μέγεθος γνωστό και ως *μήκος ουράς*	
n_q	ο αριθμός έργων, που περιμένουν στην ουρά για εξυπηρέτηση	
n_s	ο αριθμός έργων, που υφίστανται εξυπηρέτηση	
r	ο χρόνος απόκρισης Συμπεριλαμβάνει το χρόνο αναμονής στην ουρά και το χρόνο εξυπηρέτησης.	
w	ο χρόνος αναμονής στην ουρά	

Όλες οι μεταβλητές, που αναφέρονται στον πίνακα 3.1 είναι τυχαίες μεταβλητές, εκτός από το μέσο ρυθμό αφίξεων λ και το μέσο ρυθμό εξυπηρέτησης μ.

Για τη μελέτη ενός συστήματος ουράς, είναι απαραίτητο ο συνολικός αριθμός έργων n, μέσα στο σύστημα, να μην αυξάνεται διαρκώς χωρίς περιορισμό με την πάροδο του χρόνου. Αν συνέβαινε αυτό, τότε λέμε ότι το σύστημα είναι κορεσμένο και δε μπορεί να γίνει η ανάλυσή του σε κατάσταση *στατιστικής ισορροπίας* (steady-state ή equilibrium). Η συνθήκη για την ύπαρξη κατάστασης στατιστικής ισορροπίας είναι:

$$\lambda < k \cdot \mu \qquad (1)$$

όπου k είναι ο αριθμός των σημείων εξυπηρέτησης.

Γενικά, οι περισσότερες μελέτες απόδοσης συστημάτων βασίζονται στην υπόθεση της στατιστικής ισορροπίας, ενώ οι μελέτες, που αφορούν την αρχική *μεταβατική συμπεριφορά* (transient behavior) αυτών είναι ίσως μικρότερης πρακτικής σημασίας και σίγουρα μεγαλύτερης πολυπλοκότητας.

Μεταβλητές όπως ο συνολικός αριθμός έργων *n* ή ο χρόνος αναμονής στην ουρά *w* είναι συναρτήσεις, που εξαρτώνται από το χρόνο. Συναρτήσεις όπως αυτές ονομάζονται στοχαστικές διαδικασίες και στα δίκτυα ουρών είναι χρήσιμες για την αναπαράσταση της κατάστασης του συστήματος. Κάτω από ορισμένες προϋποθέσεις, οι στοχαστικές αυτές διαδικασίες διαθέτουν μία θεμελιώδη ιδιότητα γνωστή ως ιδιότητα Markov. Σε μία τέτοια περίπτωση, οι μελλοντικές καταστάσεις της διαδικασίας είναι ανεξάρτητες από το παρελθόν της και εξαρτώνται μόνο από την παρούσα κατάσταση. Η αναλυτική μελέτη αυτών των διαδικασιών μπορεί να βασιστεί στα αποτελέσματα που συνοψίζονται στο Παράρτημα Α της διατριβής, μόνο κάτω από την προϋπόθεση της στατιστικής ισορροπίας.

Πάντως σε κάθε περίπτωση, ο πιο θεμελιώδης νόμος, που διέπει τα συστήματα ουρών δεν έχει σε τίποτε να κάνει με τις θεωρίες Markov και είναι ο *νόμος του Little* [LIT61]. Ο νόμος του Little βρίσκει εφαρμογή σε όλα τα συστήματα ή τμήματα συστημάτων, στα οποία ο αριθμός των έργων, που εισέρχονται, είναι ίσος με τον αριθμό των έργων των οποίων ολοκληρώνεται η εξυπηρέτηση (job flow balance). Αυτό σημαίνει ότι προϋπόθεση εφαρμογής του νόμου του Little είναι η μη απώλεια έργων σε οποιοδήποτε μέρος του συστήματος λόγω π.χ. περιορισμένης χωρητικότητας. Η σχέση,

$$\overline{n} = \lambda \times \overline{r} \qquad (2)$$

που εκφράζει το νόμο του Little επιτρέπει τη συσχέτιση του μέσου αριθμού έργων στο σύστημα με το μέσο χρόνο παραμονής σε αυτό.

Τα σημαντικότερα από τα μέτρα απόδοσης, τα οποία ενδιαφέρουν σε μία μελέτη απόδοσης συστήματος φαίνονται στον πίνακα 3.2.

Πίνακας 3.2 Μέτρα απόδοσης

Μέτρα (Υπο) Συστήματος	R	μέσος χρόνος απόκρισης	$\overline{r}$
	X	παραγωγή συστήματος Ο μέσος αριθμός έργων, που διέρχονται στη μονάδα του χρόνου.	
	N	μέσος αριθμός έργων στο σύστημα	$\overline{n}$
Μέτρα Κέντρων Εξυπηρέτησης (Ουρά + Σημείο/α Εξυπηρέτησης)	U_i	αξιοποίηση (utilization) του κέντρου εξυπηρέτησης *i* Το κλάσμα του χρόνου κατά τον οποίο το κέντρο είναι απασχολημένο.	
	R_i	μέσος χρόνος απόκρισης του κέντρου εξυπηρέτησης *i* Ο χρόνος απόκρισης περιλαμβάνει τόσο το χρόνο αναμονής στην ουρά όσο και το χρόνο εξυπηρέτησης.	$\overline{r}_i = \overline{w}_i + \overline{s}_i$
	X_i	παραγωγή του κέντρου εξυπηρέτησης *i*	
	N_i	μέσος αριθμός έργων στο κέντρο εξυπηρέτησης *i*	$\overline{n}_i = \overline{n}_{i,q} + \overline{n}_{i,s}$
	W_i	μέσος χρόνος αναμονής στην ουρά *i*	$\overline{w}_i$

Με τους συμβολισμούς του πίνακα 3.2, η σχέση (2), που εκφράζει το νόμο του Little για σύστημα ή υποσύστημα ουρών, γράφεται και ως εξής,

$$N = X \times R \qquad (3)$$

ενώ για κάθε κέντρο εξυπηρέτησης χωριστά ισχύει η

$$N_i = X_i \times R_i \tag{4}$$

Βέβαια, μία ανάλογη σχέση ισχύει και για τα έργα, που βρίσκονται συσσωρευμένα στην ουρά αναμονής, αν εξαιρεθούν δηλαδή αυτά, που ήδη βρίσκονται σε σημείο εξυπηρέτησης. Έτσι έχουμε,

$$\overline{n}_{i,q} = X_i \times W_i \tag{5}$$

Στην περίπτωση δικτύων ουρών με πολλαπλούς τύπους έργων από 1 έως d, οι παράμετροι του πίνακα 3.1 πρέπει να οριστούν για κάθε τύπο έργου χωριστά. Τα μέτρα απόδοσης, που μπορεί να ενδιαφέρουν, συνοψίζονται στον πίνακα 3.3, που ακολουθεί:

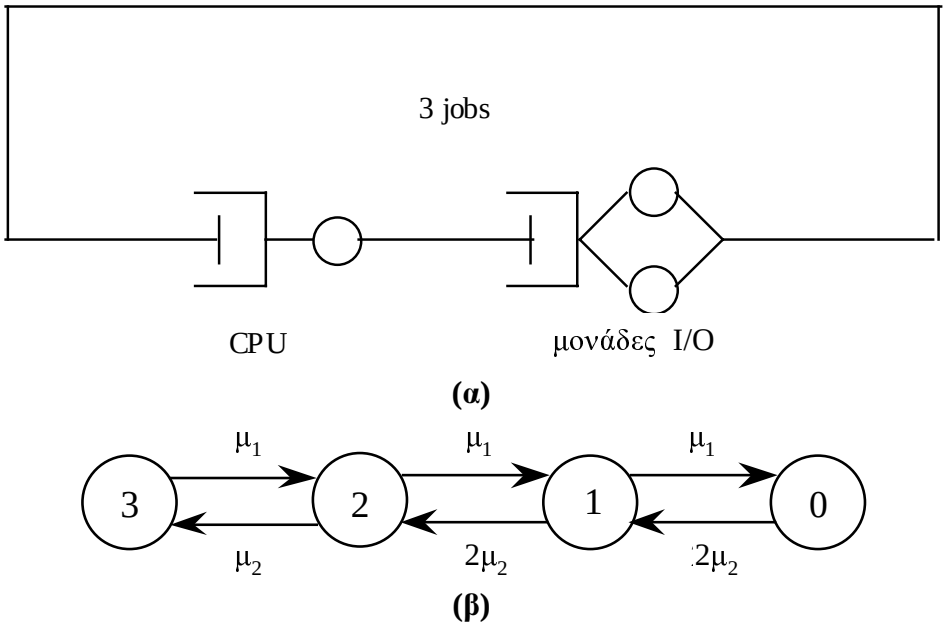
Πίνακας 3.3 Μέτρα απόδοσης για συστήματα ουρών με πολλαπλούς τύπους έργων

Μέτρα (Υπο) Συστήματος	Συνολικά	R	μέσος χρόνος απόκρισης
		X	παραγωγή συστήματος
		N	μέσος αριθμός έργων στο σύστημα
	Ανά τύπο	R_c	μέσος χρόνος απόκρισης έργων τύπου *c*
		X_c	παραγωγή συστήματος για έργα τύπου *c*
		N_c	μέσος αριθμός έργων τύπου *c* στο σύστημα
Μέτρα Κέντρων Εξυπηρέτησης (Ουρά + Σημείο/α Εξυπηρέτησης)	Συνολικά	U_i	αξιοποίηση του κέντρου εξυπηρέτησης *i*
		R_i	μέσος χρόνος απόκρισης του κέντρου εξυπηρέτησης *i*
		X_i	παραγωγή του κέντρου εξυπηρέτησης *i*
		N_i	μέσος αριθμός έργων στο κέντρο εξυπηρέτησης *i*
		W_i	μέσος χρόνος αναμονής στην ουρά *i*
	Ανά τύπο	$U_{c,i}$	αξιοποίηση του κέντρου *i* από έργα τύπου *c*
		$R_{c,i}$	μέσος χρόνος απόκρισης κέντρου *i* για έργα τύπου *c*
		$X_{c,i}$	παραγωγή κέντρου εξυπηρέτησης *i* για έργα τύπου *c*
		$N_{c,i}$	μέσος αριθμός έργων τύπου *c* στο κέντρο *i*
		$W_{c,i}$	μέσος χρόνος αναμονής έργων τύπου *c* στην ουρά *i*

3.1.1 Δίκτυα Ουρών Μορφής Γινομένου (Product Form Queueing Networks)

Στο σχήμα 3.3α εικονίζεται ένα απλό μοντέλο ουρών, το οποίο έχει χρησιμοποιηθεί για τη μελέτη της απόδοσης ενός συστήματος IBM 360/75 με λειτουργικό σύστημα OS/MVT στο Πανεπιστήμιο της California, στην Santa Barbara.

Υποθέτουμε ότι και οι δύο ουρές χαρακτηρίζονται από πειθαρχία FCFS και οι χρόνοι εξυπηρέτησης είναι ανεξάρτητοι και εκθετικά κατανεμημένοι με ρυθμούς μ_1 για τη CPU και μ_2 για τις μονάδες I/O. Η συνάρτηση $N_{CPU} = \{N(t); t \geq 0\}$ που εκφράζει τον αριθμό των έργων, που βρίσκονται στη CPU, είναι μία διαδικασία Markov με πιθανές καταστάσεις τις $\{3,2,1,0\}$. Όταν λοιπόν η συνάρτηση *N(t)* έχει τιμή 2, αυτό σημαίνει ότι υπάρχουν 2 έργα στη CPU ενώ το τρίτο έργο του συστήματος βρίσκεται στις μονάδες I/O.



Σχήμα 3.3 (α) Το δίκτυο ουρών του Chiu et al
(β) Το διάγραμμα καταστάσεων - μεταβάσεων της συνάρτησης *N(t)* του αριθμού έργων στη CPU

Το διάγραμμα κατάστασης - μετάβασης της συνάρτησης *N(t)* εικονίζεται στο σχήμα 3.3β. Σύμφωνα λοιπόν με το θεώρημα Α.3 του παραρτήματος Α, η κατανομή της πιθανότητας εύρεσης της CPU σε κάποια από τις δυνατές καταστάσεις δίνεται από την επίλυση του συστήματος εξισώσεων ισορροπίας,

$$
\begin{array}{lllllllll}
-\mu_1 P(3) & + & \mu_2 P(2) & & & & & & = 0 \\
\mu_1 P(3) & - & (\mu_1+\mu_2)P(2) & + & 2\,\mu_2 P(1) & & & & = 0 \\
 & & \mu_1 P(2) & - & (\mu_1+2\mu_2)P(1) & + & 2\mu_2 P(0) & & = 0 \\
 & & & & \mu_1 P(1) & - & 2\mu_2 P(0) & & = 0
\end{array}
$$

και της εξίσωσης κανονικοποίησης,

$$P(3) + P(2) + P(1) + P(0) = 1$$

όπου *P(i)*, *i* = 1, 2, 3 είναι η πιθανότητα να βρίσκονται *i* έργα στη CPU, όταν το σύστημα βρίσκεται σε κατάσταση στατιστικής ισορροπίας.
Η λύση του παραπάνω συστήματος εξισώσεων είναι,

$$P(3) = \mu_1^{-3}/G \qquad P(2) = \mu_1^{-2}\cdot\mu_2^{-1}/2G$$
$$P(1) = \mu_1^{-1}\cdot\mu_2^{-2}/2G \qquad P(0) = \mu_2^{-3}/4G$$

όπου:

$$G = \mu_1^{-3} + \frac{\mu_1^{-2}\cdot\mu_2^{-1}}{2} + \frac{\mu_1^{-1}\cdot\mu_2^{-2}}{2} + \frac{\mu_2^{-3}}{4}$$

Ακολούθως, ο υπολογισμός διαφόρων μέτρων απόδοσης γίνεται εύκολα, όπως για παράδειγμα στις περιπτώσεις, που ακολουθούν:

$$U_{CPU} = P(3) + P(2) + P(1)$$

$$X_{CPU} = \mu_1 \cdot (P(3) + P(2) + P(1))$$

$$N_{CPU} = 3 \cdot P(3) + 2 \cdot P(2) + 1 \cdot P(1)$$

Γενικά, κάθε δίκτυο M ουρών, του οποίου οι κατανομές πιθανοτήτων σε κατάσταση στατιστικής ισορροπίας έχουν τη μορφή,

$$P(n_1, n_2, ..., n_M) = \frac{1}{G} \cdot \prod_{i=1}^{M} f_i(n_i) \qquad (6)$$

όπου:

$$f_i(n_i) = \begin{cases} 1, & n_i = 0 \\ X_i^{n_i} \cdot S_i(n_i) \cdot S_i(n_i - 1) ... S_i(1), & n > 0 \end{cases} \quad \text{και}$$

$S_i(n)$ είναι συναρτήσεις εξυπηρέτησης, ενώ

G είναι μία σταθερά κανονικοποίησης,

λέμε ότι είναι ένα *δίκτυο μορφής γινομένου*.

Για ένα δοθέν δίκτυο ουρών, είναι σημαντικό να είναι γνωστό αν αυτό είναι μορφής γινομένου, μια και η ανάλυση των δικτύων της κατηγορίας αυτής είναι ευκολότερη από την ανάλυση των δικτύων, που δεν είναι μορφής γινομένου. Σε μία εργασία - σταθμό [BCM75] το 1975 οι Baskett, Chandy, Muntz και Palacios ανέπτυξαν ένα σύνολο κριτηρίων, η συμμόρφωση στα οποία οδηγεί στη μελέτη δικτύων μορφής γινομένου. Μία εξίσου σημαντική αλλά και διαφορετική θεμελίωση των δικτύων ουρών μορφής γινομένου έχει γίνει επίσης από τον Kelly ([KEL75], [KEL76] και [KEL79]). Πάντως, όταν ένα δίκτυο συμμορφώνεται στα κριτήρια της [BCM75], που παρατίθενται παρακάτω, τότε έχει επικρατήσει να γίνεται λόγος για ένα BCMP δίκτυο ουρών.

- *Πειθαρχίες εξυπηρέτησης*: Όλα τα κέντρα εξυπηρέτησης πρέπει είτε να είναι κέντρα καθυστέρησης, είτε η πειθαρχία τους να είναι μία εκ των FCFS, PS, LCFSPR.
- *Τύποι έργων*: Κάθε έργο είναι ενός μόνο τύπου ενώ αυτό βρίσκεται σε αναμονή σε ουρά ή υπό εξυπηρέτηση, μπορεί όμως να αλλάζει τύπο σύμφωνα με προκαθορισμένες πιθανότητες μετά από την ολοκλήρωση εξυπηρέτησης σε κάποιο κέντρο.
- *Κατανομές χρόνων εξυπηρέτησης*: Σε κέντρα FCFS οι κατανομές χρόνων εξυπηρέτησης πρέπει να είναι εκθετικές και ίδιες για όλους τους τύπους έργων. Σε άλλα κέντρα, όπου οι κατανομές των χρόνων εξυπηρέτησης έχουν μετασχηματισμό Laplace, διαφορετικοί τύποι έργων μπορούν να έχουν διαφορετικές κατανομές. Για την αναπαράσταση ενός τέτοιου δικτύου με τη μορφή γινομένου γίνεται προσέγγιση των μη εκθετικών κατανομών από αριθμό διαδοχικών σταδίων εκθετικής εξυπηρέτησης.
- *Εξυπηρέτηση βάσει κατάστασης*: Ο χρόνος εξυπηρέτησης σε ένα κέντρο FCFS μπορεί να εξαρτάται μόνο από το συνολικό αριθμό έργων σε αυτό. Σε κέντρα καθυστέρησης και κέντρα PS και LCFSPR ο χρόνος εξυπηρέτησης για κάποιο τύπο έργων μπορεί να εξαρτάται από τον αριθμό έργων του αυτού τύπου στο κέντρο, αλλά όχι από το συνολικό αριθμό έργων ανεξαρτήτως τύπου.
- *Διαδικασίες αφίξεων*: Σε ανοικτά δίκτυα, ο χρόνος μεταξύ διαδοχικών αφίξεων έργων του ιδίου τύπου πρέπει να κατανέμεται εκθετικά. Δεν επιτρέπονται μαζικές αφίξεις. Οι ρυθμοί αφίξεων μπορούν επίσης να εξαρτώνται από την κατάσταση

του συστήματος. Τέλος, ένα δίκτυο ουρών μπορεί να είναι ανοικτό ως προς κάποιους τύπους έργων και κλειστό ως προς κάποιους άλλους τύπους.

Η απαίτηση της ύπαρξης μετασχηματισμών Laplace για τις κατανομές των χρόνων εξυπηρέτησης προέκυψε από το γεγονός ότι είχε βρεθεί πως κάθε τέτοια συνάρτηση κατανομής μπορεί να προσεγγισθεί από ένα σύνολο διαδοχικών σταδίων εκθετικής εξυπηρέτησης. Εφαρμογή της μεθόδου αυτής, που είναι γνωστή και ως *μέθοδος εκθετικών σταδίων εξυπηρέτησης* γίνεται και στο [SC81]. Παρόλα αυτά, αργότερα ([BAR76], [CHT77], [COH79]) αποδείχθηκε ότι ο περιορισμός της ύπαρξης μετασχηματισμών Laplace για τις κατανομές των χρόνων εξυπηρέτησης δεν είναι απαραίτητος.

Μία προσπάθεια επέκτασης της κλάσης των δικτύων ουρών μορφής γινομένου σε δίκτυα, που δε χαρακτηρίζονται απαραίτητα από την ιδιότητα Markov έγινε από τους Denning και Buzen το 1978 [DB78]. Οι συνθήκες, που αυτοί εισάγουν είναι οι:

- *Ισορροπία ροής έργων*, όπου για κάθε τύπο ο αριθμός των αφίξεων σε ένα κέντρο εξυπηρέτησης ισούται με τον αριθμό των αναχωρήσεων από αυτό.
- *Βήμα προς βήμα κίνηση*, που υποδηλώνει τη μη ύπαρξη ταυτόχρονων κινήσεων έργων.
- *Ομογένεια συσκευών*, η οποία ορίζει ότι ο ρυθμός εξυπηρέτησης ενός συγκεκριμένου τύπου έργων δεν εξαρτάται από την κατάσταση του συστήματος με άλλο τρόπο εκτός από το συνολικό μήκος του κέντρου εξυπηρέτησης ή από τον αριθμό των έργων του συγκεκριμένου τύπου στο κέντρο. Αυτό έχει ως αποτέλεσμα:
  - Την απασχόληση ενός μόνου πόρου: Ένα έργο δεν μπορεί να έχει ταυτόχρονη παρουσία σε δύο ή περισσότερα κέντρα εξυπηρέτησης.
  - Το μη αποκλεισμό: Αν σε κάποιο κέντρο εξυπηρέτησης υπάρχει ένα τουλάχιστο έργο, τότε αυτό παρέχει εξυπηρέτηση. Η δυνατότητα παροχής εξυπηρέτησης δεν εξαρτάται από κανένα άλλο κέντρο εξυπηρέτησης.
  - Ανεξάρτητη συμπεριφορά έργων: Η αλληλεπίδραση των έργων περιορίζεται στην αναμονή τους σε ουρές. Έτσι, δεν μπορούν για παράδειγμα να υπάρξουν απαιτήσεις συγχρονισμού στη δρομολόγηση των έργων.
  - Τοπική πληροφορία: Ο ρυθμός εξυπηρέτησης ενός κέντρου εξαρτάται μόνο από την τοπική ουρά και όχι από την κατάσταση του υπόλοιπου συστήματος.
  - Δίκαιη εξυπηρέτηση: Αν οι ρυθμοί εξυπηρέτησης διαφέρουν ανάλογα με τον τύπο έργων, τότε ο ρυθμός εξυπηρέτησης ενός τύπου εξαρτάται μόνο από τον αριθμό έργων του τύπου στο κέντρο εξυπηρέτησης και όχι από τον αριθμό έργων άλλων τύπων.
  - Ομογένεια δρομολόγησης: Η δρομολόγηση έργων πρέπει να είναι ανεξάρτητη της κατάστασης του συστήματος.

Οι προσπάθειες επέκτασης της κλάσης των δικτύων ουρών μορφής γινομένου συνεχίζονται. Ενδιαφέρουσα προσέγγιση είναι αυτή του Towsley [TOW80], του οποίου η επέκταση επιτρέπει μία ειδική μορφή δρομολόγησης έργων βάσει της κατάστασης του συστήματος.

Στις [LAV89] και [LN91] παρουσιάζεται μία ιστορική επισκόπηση των δικτύων ουρών μορφής γινομένου και των αλγορίθμων επίλυσης αυτών.

3.1.2 Λειτουργική Ανάλυση (Operational Analysis)

Ένα μεγάλο μέρος των προβλημάτων ανάλυσης της απόδοσης υπολογιστικών συστημάτων μπορεί να λυθεί με την επίλυση απλών μαθηματικών σχέσεων, που η ισχύς τους δεν απαιτεί καμία συνθήκη για τις κατανομές των χρόνων εξυπηρέτησης και των ενδιάμεσων χρόνων αφίξεων. Οι μαθηματικές αυτές σχέσεις είναι γνωστές με τον όρο *λειτουργικοί νόμοι*. Για πρώτη φορά διατυπώθηκαν στη [DB78] και αποτελούν βεβαίως τη βάση για έναν απλοϊκό τρόπο μελέτης δικτύων ουρών γνωστό ως λειτουργική ανάλυση. Ο όρος «λειτουργική» παραπέμπει ευθέως στη δυνατότητα επαλήθευσης μέσω μετρήσεων.

Λειτουργικές ποσότητες είναι οι ποσότητες εκείνες, που είναι δυνατό να μετρηθούν κατά τη διάρκεια μιας συγκεκριμένης περιόδου παρατήρησης T του (υπο)συστήματος. Αυτές που κατά βάση χρησιμοποιούνται σε μία λειτουργική ανάλυση εικονίζονται στον πίνακα 3.4.

Πίνακας 3.4 Ποσότητες που χρησιμοποιούνται στη λειτουργική ανάλυση δικτύων ουρών	
A_i	ο αριθμός των αφίξεων στο κέντρο εξυπηρέτησης i
B_i	ο συνολικός χρόνος, που το κέντρο εξυπηρέτησης i είναι απασχολημένο (όταν δηλαδή $n_i > 0$)
C_{ij}	ο αριθμός των περιπτώσεων, που ένα έργο απαιτεί εξυπηρέτηση στο κέντρο j αμέσως μετά την ολοκλήρωση εξυπηρέτησής του στο κέντρο i
A_{0j}	ο αριθμός των έργων, που πρώτα αναζητούν εξυπηρέτηση στο κέντρο j
C_{i0}	ο αριθμός των έργων, που η τελευταία εξυπηρέτηση που λαμβάνουν είναι στο κέντρο i
T	περίοδος παρατήρησης

Από τους συμβολισμούς, που εισάγονται στον πίνακα 3.4, είναι προφανές ότι το κέντρο «0» εκφράζει ουσιαστικά τον «εξωτερικό κόσμο» του (τμήματος του) συστήματος, που μελετάται. Υποθέτουμε ότι $C_{00} = 0$, γιατί διαφορετικά θα υπήρχαν περιπτώσεις έργων, που αναχωρούν από το (υπο)σύστημα, χωρίς να έχουν κάνει προηγούμενα χρήση κάποιου κέντρου εξυπηρέτησης. Επίσης, δεν αποκλείεται να ισχύει $C_{ii} > 0$, καθώς μπορεί ένα έργο, που μόλις έχει εξυπηρετηθεί από κάποιο κέντρο εξυπηρέτησης να απαιτήσει πάλι εξυπηρέτηση από το ίδιο κέντρο. Σύμφωνα λοιπόν με τα παραπάνω ο αριθμός των έργων, που εξυπηρετήθηκαν στο κέντρο i είναι,

$$C_i = \sum_{j=1}^{K} C_{ij} \ , \qquad i = 1, 2, \ldots, K$$

και ο αριθμός των αφίξεων και των αναχωρήσεων από το (υπο)σύστημα είναι αντίστοιχα,

$$A_0 = \sum_{j=1}^{K} A_{0j} \quad \text{και} \quad C_0 = \sum_{i=1}^{K} C_{i0}$$

Είναι επίσης προφανές ότι σε ένα κλειστό σύστημα $A_0 = C_0$. Στον πίνακα 3.5 εκφράζονται κάποιες σημαντικές λειτουργικές ποσότητες σε συνάρτηση αυτών, που ήδη ορίσθηκαν.

Πίνακας 3.5 Σημαντικές λειτουργικές ποσότητες - αποτελέσματα		
U_i	αξιοποίηση του κέντρου εξυπηρέτησης i	$= B_i/T$
$\bar{s}_i$	ο μέσος χρόνος εξυπηρέτησης στο κέντρο i	$= B_i/C_i$
X_i	παραγωγή στο κέντρο i	$= C_i/T$
q_{ij}	η συχνότητα δρομολόγησης, ο λόγος δηλαδή των έργων, που κατευθύνονται στο κέντρο j μετά από το κέντρο i	$= \begin{cases} C_{ij}/C_i, & i = 1,\ldots,K \\ A_{0j}/A_0, & i = 0 \end{cases}$

Είναι σημαντικό να σημειωθεί ότι για κάθε *i*, $q_{i0}+q_{i1}+\ldots+q_{iK}=1$, όπου q_{i0} είναι η συχνότητα εκείνη των έργων, που εγκαταλείπουν το (υπο)σύστημα μετά από εξυπηρέτηση στο κέντρο *i* και το q_{0j} εκφράζει τη συχνότητα εκείνη των έργων, που καταφθάνουν στο (υπο)σύστημα και λαμβάνουν αρχικά εξυπηρέτηση στο κέντρο *j*. Αν επίσης λάβουμε υπόψη ότι ο λόγος των έργων, που λαμβάνουν εξυπηρέτηση και εγκαταλείπουν το (υπο)σύστημα δίνεται από τη σχέση,

$$X_0 = C_0 \big/ T$$

τότε είναι εύκολο να συνάγουμε τον ακόλουθο λειτουργικό νόμο:

$$X_0 = \sum_{i=1}^{K} X_i \cdot q_{i0} \qquad \textit{Νόμος της ροής εξόδου} \text{ (output flow law)}$$

Ακόμη, από τις σχέσεις που παρατίθενται στον πίνακα 3.5 εύκολα προκύπτει:

$$U_i = X_i \cdot \bar{s}_i \qquad \textit{Νόμος της αξιοποίησης}$$

που ισχύει για όλα τα κέντρα εξυπηρέτησης *i*.

Μία άλλη σχέση, που επίσης χρησιμοποιείται στη λειτουργική ανάλυση, είναι η σχέση (4), γνωστή ως

$$N_i = X_i \times R_i \qquad \textit{Νόμος του Little} \text{ (Little's law)}$$

Αν για μία περίοδο παρατήρησης *T* ισχύει η *ισορροπία ροής έργων*, τότε

$$A_i = C_i$$

καθώς ο αριθμός των αφίξεων σε κάθε κέντρο εξυπηρέτησης είναι ίδιος με τον αριθμό των αναχωρήσεων από αυτό. Γενικά, η μελέτη ενός συστήματος σε περιόδους παρατήρησης, που η αρχική και η τελική κατάσταση της κάθε ουράς είναι η ίδια, δεν είναι κάτι το ασυνήθιστο. Εξάλλου, η ιδέα αυτή αποτελεί και τη βάση της ιδιαίτερα ελκυστικής προσομοιωτικής *μεθόδου της ανανέωσης* (regenerative method), που περιγράφεται εκτενώς στο κεφάλαιο 4. Αν λοιπόν ισχύει η ισορροπία ροής έργων και αν θεωρήσουμε ότι κάθε έργο υποβάλλει V_i κλήσεις στο κέντρο εξυπηρέτησης *i*, τότε

$$V_i = \frac{C_i}{C_0} \qquad (7)$$

Από τον ορισμό όμως της παραγωγής στο κέντρο εξυπηρέτησης *i*, που δίνεται στον πίνακα 3.5 και από τη σχέση (7) έχουμε:

$$X_i = \frac{C_i}{T} = \frac{C_i}{C_0} \times \frac{C_0}{T} = V_i \cdot X_0 \qquad \textit{Νόμος της ασκούμενης ροής} \text{ (forced flow law)}$$

Όσον αφορά το συνολικό χρόνο εξυπηρέτησης ενός έργου στο (υπο)σύστημα, αυτός υπολογίζεται με την εφαρμογή του νόμου του Little σε επίπεδο (υπο)συστήματος, δηλαδή

$$N = X_0 \times R$$

Καθώς $N=N_1+\ldots+N_K$ και λαμβάνοντας υπόψη το νόμο του Little σε επίπεδο κέντρου εξυπηρέτησης και το νόμο της ασκούμενης ροής, προκύπτει

$$R = \sum_{i=1}^{K} V_i \cdot R_i \qquad \textit{Γενικός νόμος χρόνου απόκρισης}$$

Ο τελευταίος λειτουργικός νόμος βρίσκει εφαρμογή σε συστήματα αναπαράστασης αλληλεπιδραστικής επεξεργασίας, όπου ένας αριθμός χρηστών δημιουργεί, μέσω τερματικών, κλήσεις εξυπηρέτησης σε ένα κεντρικό σύστημα. Μετά από ένα χρόνο προσμονής *Z*, ο κάθε χρήστης μπορεί να υποβάλλει κάποια νέα κλήση εξυπηρέτησης. Αν ο χρόνος απόκρισης του συστήματος είναι *R*, τότε ο κύκλος των έργων επεξεργασίας στο σύστημα είναι *R*+*Z*. Έτσι, ο κάθε χρήστης δημιουργεί περίπου *T*/(*R*+*Z*) κλήσεις εξυπηρέτησης και αν υποθέσουμε ότι στο σύστημα εργάζονται *N* χρήστες, τότε

$$X_0 = \frac{N \cdot [T/(R+Z)]}{T} = \frac{N}{R+Z}$$

και τελικά,

$$R = (N / X_0) - Z$$ *Νόμος απόκρισης αλληλεπίδρασης* (interactive response time law)

Στον πίνακα 3.6, που ακολουθεί, συνοψίζεται το σύνολο των λειτουργικών νόμων που παρουσιάστηκαν.

Πίνακας 3.6 Βασικοί νόμοι λειτουργικής ανάλυσης

$U_i = X_i \cdot \bar{s}_i$	Νόμος της αξιοποίησης
$N_i = X_i \times R_i$	Νόμος του Little
$X_i = V_i \cdot X_0$	Νόμος της ασκούμενης ροής
$X_0 = \sum_{i=1}^{K} X_i \cdot q_{i0}$	Νόμος της ροής εξόδου
$R = \sum_{i=1}^{K} V_i \cdot R_i$	Γενικός νόμος χρόνου απόκρισης
$R = (N / X_0) - Z$	Νόμος της απόκρισης αλληλεπίδρασης

Αν κατά την περίοδο παρατήρησης ενός συστήματος μετρηθεί εξυπηρέτηση D_i στον πόρο *i*, τότε,

$$D_i = \bar{s}_i \times V_i \qquad (8)$$

όπου ο χρόνος $\bar{s}_i$ εκφράζει το μέσο χρόνο εξυπηρέτησης ανά επίσκεψη στο κέντρο *i*. Αν ένα δίκτυο ουρών είναι μορφής γινομένου, τότε αυτό είναι σημαντικό να το γνωρίζουμε, διότι στην περίπτωση αυτή ισχύει η ισορροπία ροής έργων. Έτσι, ουσιαστικά χρειάζεται μόνον η μέτρηση των ποσοτήτων D_i, καθώς οι άλλες ποσότητες μπορούν να προκύψουν μέσω της εφαρμογής λειτουργικών νόμων.

Γενικά, οι σχέσεις (7) και (8), καθώς και αυτές των πινάκων 3.5 και 3.6 συγκροτούν τη λειτουργική προσέγγιση στο πρόβλημα της εκτίμησης της απόδοσης συστημάτων. Συχνά η τεχνική αυτή είναι επαρκής για την συνολική επίλυση ενός προβλήματος. Στις περισσότερες όμως περιπτώσεις απλά αποτελεί ένα εργαλείο παραμετροποίησης του μοντέλου απόδοσης με βάση τις παρατηρούμενες μετρήσεις και γίνεται έτσι επιτακτική η ανάγκη χρήσης κάποιας περισσότερο προωθημένης τεχνικής αποτίμησης.

3.1.3 Αλγόριθμοι ακριβούς και προσεγγιστικής αποτίμησης δικτύων ουρών μορφής γινομένου

Ο *αλγόριθμος περιέλιξης* (convolution algorithm) ήταν η πρώτη προσπάθεια αλγοριθμικού υπολογισμού της σταθεράς κανονικοποίησης G δικτύων ουρών μορφής γινομένου. Εμφανίστηκε αρχικά στην [BUZ73] και επεκτάθηκε αργότερα στη [RK75] για αποτίμηση

δικτύων με περισσότερους τους ενός τύπους έργων (αλυσίδες). Είναι ένας αναδρομικός αλγόριθμος, που εκτός από τη σταθερά κανονικοποίησης επιτρέπει στη συνέχεια τον υπολογισμό όχι μόνο των μέσων τιμών, αλλά και των κατανομών και της διασποράς μέτρων απόδοσης όπως η παραγωγή, το μήκος ουρών και οι χρόνοι απόκρισης.

Στο πλαίσιο που ακολουθεί δίνεται η απλή μορφή (για ένα μόνο τύπο έργων) του αλγορίθμου για κλειστά δίκτυα ουρών, όπως αυτός παρουσιάζεται στο [JAI91].

Αλγόριθμος περιέλιξης (convolution algorithm)

Παράμετροι[1]:

α = παράγοντας κλίμακας
N = αριθμός χρηστών
Z = χρόνος προσμονής
K = αριθμός κέντρων εξυπηρέτησης (μη περιλαμβανομένων των τερματικών)
$\bar{s}_i$ = ο μέσος χρόνος εξυπηρέτησης ανά επίσκεψη στο κέντρο i
V_i = αριθμός επισκέψεων στο κέντρο εξυπηρέτησης i

Αποτελέσματα:

X = παραγωγή συστήματος
N_i = μέσος αριθμός έργων στο κέντρο εξυπηρέτησης i
R_i = μέσος χρόνος απόκρισης στο κέντρο εξυπηρέτησης i
R = μέσος χρόνος απόκρισης του συστήματος
U_i = αξιοποίηση του κέντρου εξυπηρέτησης i
$P(n)$ = πιθανότητα το διάνυσμα με τα μήκη ουρών να είναι n

Κλίμακα: $y_0=\alpha Z$
FOR i=1 to K DO { $y_i = a \cdot \bar{s}_i \cdot V_i$ }

Αρχικοποίηση:
G(0)=1
FOR n=1 TO N DO { $G(n) = y_0^n / n!$ }

Υπολογισμός του G(N):
FOR i=1 to K DO {
BEGIN
FOR n=1 TO N DO G(n)=G(n)+y_i G(n-1)
END}

Υπολογισμός μέτρων απόδοσης:

$$X = a \cdot \frac{G(N-1)}{G(N)} \qquad P(n_0, n_1, ..., n_K) = \frac{y_0^{n_0} \cdot y_1^{n_1} \cdot y_2^{n_2} .... y_K^{n_K}}{n_0! \cdot G(N)}$$

$$U_i = X \cdot \bar{s}_i \cdot V_i \qquad P(n_i \geq j) = y_i^j \cdot \frac{G(N-j)}{G(N)}, \quad i=1, 2, \ldots, K$$

$$N_i = \sum_{j=1}^{N} y_i^j \cdot \frac{G(N-j)}{G(N)} \qquad P(n_i = j) = \frac{y_i^j}{G(N)} \cdot [G(N-j) - y_i \cdot G(N-j-1)]$$

$$R_i = N_i / (X \cdot V_i)$$

$$R = \sum_{i=1}^{K} R_i \cdot V_i \qquad P(n_i \geq j, n_k \geq l) = y_i^j \cdot y_k^l \cdot \frac{G(N-j-l)}{G(N)}, \; i=1, 2, \ldots, K$$

[1] Στον αλγόριθμο αυτό με «0» συμβολίζονται τα τερματικά

Ο παράγοντας κλίμακας a δεν θα πρέπει να είναι ούτε πολύ μικρός αλλά ούτε πολύ μεγάλος σε σύγκριση με το 1. Μία καλή επιλογή θα μπορούσε να ήταν η τιμή,

$$a = \frac{1}{(1/K) \cdot \sum_{i=1}^{K} D_i}$$

Επίσης, σε περιπτώσεις, που οι τιμές V_i δεν είναι γνωστές, αυτές μπορούν να υπολογισθούν από την επίλυση του συστήματος των εξισώσεων,

$$V_j = \sum_{i=0}^{K} V_i \cdot q_{ij} , \qquad j = 0, 1, \ldots, K$$

που εκφράζει την ισορροπία ροής έργων.

Ο αλγόριθμος περιέλιξης είναι βέβαια η μοναδική αναλυτική τεχνική αποτίμησης, που μπορεί να δώσει απαντήσεις σε ερωτήματα του τύπου ποια είναι για παράδειγμα η πιθανότητα να είναι συγχρόνως απασχολημένοι σε ένα σύστημα και οι δύο του δίσκοι.

Όσον αφορά τώρα την ανάλυση ανοικτών δικτύων ουρών, αυτή δεν παρουσιάζει ιδιαίτερες δυσκολίες και συνοψίζεται στο πλαίσιο που ακολουθεί.

Αλγόριθμος αποτίμησης ανοικτών δικτύων ουρών

Παράμετροι:

X = ρυθμός εξωτερικών αφίξεων, παραγωγή συστήματος
K = αριθμός κέντρων εξυπηρέτησης
$\bar{s}_i$ = ο μέσος χρόνος εξυπηρέτησης ανά επίσκεψη στο κέντρο i
V_i = αριθμός επισκέψεων στο κέντρο εξυπηρέτησης i ($V_0 = 1$)

Αποτελέσματα:

N_i = μέσος αριθμός έργων στο κέντρο εξυπηρέτησης i
R_i = μέσος χρόνος απόκρισης στο κέντρο εξυπηρέτησης i
R = μέσος χρόνος απόκρισης του συστήματος
U_i = αξιοποίηση του κέντρου εξυπηρέτησης i
N = μέσος αριθμός έργων στο σύστημα

Συνολικές απαιτήσεις εξυπηρέτησης: $D_i = \bar{s}_i \cdot V_i$

Αξιοποίηση & παραγωγή κέντρων: $U_i = X \cdot D_i$ $\qquad X_i = X \cdot V_i$

Απόκριση κέντρων εξυπηρέτησης:
$$R_i = \begin{cases} \bar{s}_i/(1-U_i) & \text{κέντρα εξυπηρέτησης} \\ \bar{s}_i & \text{κέντρα καθυστέρησης} \end{cases}$$

Μέσος αριθμός έργων στα κέντρα:
$$N_i = \begin{cases} U_i/(1-U_i) & \text{κέντρα εξυπηρέτησης} \\ U_i & \text{κέντρα καθυστέρησης} \end{cases}$$

Χρόνος απόκρισης & αριθμός έργων: $R = \sum_{i=1}^{K} R_i \cdot V_i$ $\qquad N = \sum_{i=1}^{K} N_i$

Μία άλλη σημαντική εξέλιξη στην αλγοριθμική αναλυτική επίλυση δικτύων ουρών μορφής γινομένου ήταν η εμφάνιση του αλγορίθμου *ανάλυσης μέσων τιμών* (Mean Value Analysis) στην [RL80]. Ο συγκεκριμένος αλγόριθμος επιτρέπει τον υπολογισμό μέτρων απόδοσης παραγωγής, μέσου αριθμού έργων σε κέντρα εξυπηρέτησης και μέσων χρόνων απόκρισης, απευθείας, χωρίς δηλαδή τον προηγούμενο υπολογισμό της σταθεράς κανονικοποίησης. Στο πλαίσιο, που ακολουθεί, περιγράφεται μία παραλλαγή του αλγορίθμου για κλειστά δίκτυα με περισσότερους του ενός τύπους έργων, όπως αυτή δίνεται στο [LZGS84].

Αλγόριθμος ανάλυσης μέσων τιμών (MVA algorithm)

Παράμετροι:

$\vec{N}$ = αριθμός χρηστών, όπου $\vec{N} = (N_1, N_2, ..., N_d)$ οι αριθμοί έργων τύπου 1 έως d

$\vec{Z}$ = χρόνος προσμονής, όπου $\vec{Z} = (Z_1, Z_2, ..., Z_d)$ οι χρόνοι προσμονής του κάθε τύπου

K = αριθμός κέντρων εξυπηρέτησης (μη περιλαμβανομένων των τερματικών)

$D_{c,i}$ = οι συνολικοί χρόνοι εξυπηρέτησης έργων τύπου c στο κέντρο εξυπηρέτησης i

$V_{c,i}$ = αριθμός επισκέψεων έργων τύπου c στο κέντρο εξυπηρέτησης i

Αποτελέσματα:

X_c = παραγωγή έργων τύπου c

Q_i = $N_{1i}+N_{2i}+\ldots+N_{di}$ = μέσος αριθμός έργων όλων των τύπων στο κέντρο εξυπηρέτησης i

$R_{c,i}$ = μέσος χρόνος απόκρισης έργων τύπου c στο κέντρο εξυπηρέτησης i

R = μέσος χρόνος απόκρισης του συστήματος

$U_{c,i}$ = αξιοποίηση του κέντρου εξυπηρέτησης i από έργα τύπου c

FOR i=1 to K DO { $Q_i(\vec{0}) = 0$ }

FOR n=1 TO $\sum_{c=1}^{d} N_c$ DO {

BEGIN

για κάθε συνδυασμό $\vec{N} = (N_1, N_2, ..., N_d)$ με συνολικό αριθμό έργων n {

BEGIN

FOR c=1 TO d DO {

BEGIN

FOR i=1 TO K DO {

BEGIN

$$R_{c,i} = \begin{cases} D_{c,i} & \text{κέντρο καθυστέρησης} \\ D_{c,i} \cdot (1 + Q_i(\overrightarrow{N - c})) & \text{κέντρο εξυπηρέτησης} \end{cases}$$

END}

$$X_c = \frac{N_c}{Z_c + \sum_{i=1}^{K} R_{c,i}}$$

FOR i=1 TO K DO {

BEGIN

$$Q_i(\vec{N}) = \sum_{c=1}^{d} X_c \cdot R_{c,i}$$

END}

END}

END}

END}

Παραγωγή κέντρου i για έργα τύπου c: $X_{c,i}=X_c\, V_{c,i}$

Αξιοποίηση κέντρου i από έργα τύπου c: $U_{c,i}= X_c\, D_{c,i}$

Στο [LZGS84] δίνεται εξάλλου και μία άλλη παραλλαγή του αλγορίθμου, κατάλληλη για μικτά δίκτυα ουρών.

Παρατηρούμε ότι τόσο ο αλγόριθμος περιέλιξης, όσο και ο αλγόριθμος ανάλυσης μέσων τιμών, βασίζονται σε υπολογισμούς αναδρομικούς ως προς τον πληθυσμό των έργων του δικτύου. Αυτό έχει ως συνέπεια να δημιουργείται απαγορευτικό υπολογιστικό και αποθηκευτικό κόστος σε περιπτώσεις δικτύων ουρών με πολλές αλυσίδες δρομολόγησης ή σε περιπτώσεις με λίγες αλυσίδες αλλά μεγάλο αριθμό έργων ανά τύπο.

Οι εργασίες [SCH79b] και [BAR79] ήταν οι πρώτες, οι οποίες αντιμετώπισαν το συγκεκριμένο πρόβλημα προσπαθώντας να «σπάσουν» αυτή την αναδρομικότητα, οδηγώντας έτσι στη διατύπωση προσεγγιστικών *αλγορίθμων ανάλυσης μέσων τιμών* (approximate MVA), όπως αυτούς, που περιλαμβάνονται στο [LZGS84].

Η προσέγγιση αυτή είναι ισοδύναμη με την υπόθεση ότι για κάθε αλυσίδα η μέση τιμή του λόγου των έργων της σε κάθε ουρά δεν αλλάζει αν υπήρχε στο δίκτυο ένα έργο λιγότερο. Η πραγματικότητα όμως είναι ότι ο λόγος αυτός αλλάζει. Στην [CN82] διατυπώνεται μία τεχνική υπολογισμού της αλλαγής αυτής, η οποία οδηγεί τελικά σε έναν περισσότερο ακριβή προσεγγιστικό αλγόριθμο γνωστό ως *Linearizer*. Ο αλγόριθμος αυτός έχει δοκιμαστεί εμπειρικά σε μεγάλο αριθμό περιπτώσεων και έχει βρεθεί ότι δίνει αποτελέσματα, που στις πιο πολλές περιπτώσεις, δεν αποκλίνουν περισσότερο από 1% από την πραγματική τιμή.

Όταν, παρόλα αυτά, το αντικείμενο του προβλήματος είναι η μοντελοποίηση συστημάτων, όπου έργα που προέρχονται από κάποιο υπολογιστικό σύστημα μπορεί να κάνουν χρήση πόρων άλλων συστημάτων, τότε το κλειστό δίκτυο ουρών που προκύπτει, μπορεί να περιέχει ακόμη και εκατοντάδες διαφορετικών τύπων έργων. Τέτοιες περιπτώσεις εμφανίζονται στη μοντελοποίηση τοπικών δικτύων και κατανεμημένων συστημάτων και τότε ο αλγόριθμος Linearizer μπορεί να έχει ιδιαίτερα μεγάλο υπολογιστικό κόστος. Στη [SLM86] οι συγγραφείς προτείνουν μία διαφορετική προσέγγιση, η οποία αντιμετωπίζει αυτό ακριβώς το πρόβλημα. Με την τεχνική αυτή το δίκτυο ουρών χωρίζεται σε υποδίκτυα. Κάθε υποδίκτυο επιλύεται με την αντικατάσταση κάποιων αλυσίδων με κέντρα καθυστέρησης και κάποιων άλλων με αφίξεις Poisson. Οι μέσοι χρόνοι εξυπηρέτησης των κέντρων καθυστέρησης και οι ρυθμοί των αφίξεων Poisson εξαρτώνται από τα μέτρα απόδοσης του υπόλοιπου δικτύου. Το σύνολο των μη γραμμικών εξισώσεων, που προκύπτει για τα μέτρα απόδοσης του δικτύου, επιλύεται επαναληπτικά με τη χρήση διαδοχικών αντικαταστάσεων. Η τεχνική αυτή έχει βρεθεί εμπειρικά ότι είναι περισσότερο ακριβής από τους κλασικούς προσεγγιστικούς αλγορίθμους MVA και με μικρότερο υπολογιστικό κόστος όμως λιγότερο ακριβής από τον αλγόριθμο Linearizer.

Παρόλα αυτά, γενικά η μοντελοποίηση παράλληλων και κατανεμημένων συστημάτων με δίκτυα ουρών μορφής γινομένου απαιτεί στις περισσότερες περιπτώσεις την αποδοχή απλουστευτικών προσεγγίσεων, που όμως η σχέση τους με την πραγματικότητα πρέπει σε κάθε περίπτωση να είναι επαληθεύσιμη. Ενδιαφέρουσα είναι η εφαρμογή [HT82], που αφορά μία κλάση συστημάτων παράλληλης επεξεργασίας, όπου κάθε έργο διαχωρίζεται κατά το πρότυπο διάσπαση/ένωση (fission/fusion) σε έναν αριθμό ασύγχρονων εργασιών επεξεργασίας. Η λύση λοιπόν της περίπτωσης αυτής δικτύων ουρών, που δεν είναι μορφής γινομένου, προσεγγίζεται με την επαναληπτική επίλυση σειράς δικτύων ουρών μορφής γινομένου, που προκύπτουν από την *αποσύνθεση* του αρχικού δικτύου.

Τέλος, ενδιαφέρον παρουσιάζει και η ανάλυση δικτύων ουρών μορφής γινομένου με θετικά έργα και αρνητικά έργα ([GEL91], [GGS91] και [GEL94]), που ακυρώνουν την ύπαρξη των πρώτων. Μοντέλα αυτού του τύπου βρίσκουν εφαρμογή στη μελέτη εφαρμογών εξισορρόπησης φόρτου και κατανομής πόρων.

3.1.4 Δίκτυα ουρών τα οποία δεν είναι μορφής γινομένου

Παρά την ανάπτυξη προωθημένων αλγορίθμων αποτίμησης, η κλασσική θεωρία δικτύων ουρών δε μπορεί να δώσει λύσεις στη μοντελοποίηση μιας σειράς φαινομένων, που όμως είναι κοινός τόπος στην ανάλυση της απόδοσης υπολογιστικών συστημάτων. Τα πιο σημαντικά από αυτά είναι οι:

- *Μη εκθετικοί χρόνοι εξυπηρέτησης*: Τα περισσότερα από τα αποτελέσματα της κλασσικής θεωρίας δικτύων ουρών εκτός από αυτά της λειτουργικής ανάλυσης, προϋποθέτουν εκθετικούς χρόνους εξυπηρέτησης. Η εφαρμογή της τεχνικής των εκθετικών σταδίων εξυπηρέτησης είναι εφικτή, όπως ήδη αναφέρθηκε, μόνο για τις συναρτήσεις κατανομών που διαθέτουν μετασχηματισμό Laplace. Γενικά, από διάφορες μελέτες, όπως αυτή της [SUR83], συνάγεται ότι η αποδοχή της υπόθεσης της εκθετικής εξυπηρέτησης δεν έχει συνήθως σημαντικές επιπτώσεις στα αποτελέσματα. Αν όμως παρόλα αυτά η ακρίβεια στα αποτελέσματα είναι θέμα ζωτικής σημασίας, τότε στην περίπτωση αυτή, συνιστάται η χρήση κάποιας τεχνικής προσομοίωσης για μία πιο έγκυρη αναπαράσταση των εν λόγω κατανομών εξυπηρέτησης.

- *Μαζικές αφίξεις*: Μαζικές αφίξεις σε κάποιο κέντρο εξυπηρέτησης είναι δυνατό να μοντελοποιηθούν κάτω από κάποιες προϋποθέσεις, όπως αν για παράδειγμα οι μεταξύ των ομάδων χρόνοι αφίξεων κατανέμονται εκθετικά. Στην κίνηση των πακέτων δεδομένων στα δίκτυα έχει βρεθεί ότι υπάρχει σημαντική συσχέτιση μεταξύ των διαδοχικών ενδιάμεσων χρόνων αφίξεων. Αυτό βέβαια το φαινόμενο είναι διαφορετικό από τις μαζικές αφίξεις. Χαρακτηρίζεται συνήθως ως *πανομοιότυπη κίνηση* και για τη ρεαλιστική μοντελοποίησή του οι μελέτες επικεντρώνονται στις ιδιότητες μιας κλάσης υποεκθετικών κατανομών, γνωστών ως κατανομές Pareto ή κατανομές δυναμο-ουρών (power-tail distributions) [GJL99].

- *Διάσπαση/ένωση*: Διατάξεις διάσπασης/ένωσης χρησιμοποιούνται γενικά για τη μοντελοποίηση της δημιουργίας και του συγχρονισμού υποδιεργασιών. Έχουν ως συνέπεια αφενός μεν τη μεταβολή του συνολικού αριθμού έργων, και αφετέρου την καταστρατήγηση της απαίτησης για ανεξαρτησία στη συμπεριφορά τους (βλ. παράγραφο 3.1.1), που υπάρχει για δίκτυα ουρών μορφής γινομένου. Περιπτώσεις προσομοιωτικής ανάλυσης τέτοιων μοντέλων περιγράφονται στις [KAR98], [KAR00], [KAR01a], [KAR01b] και [KL00b].

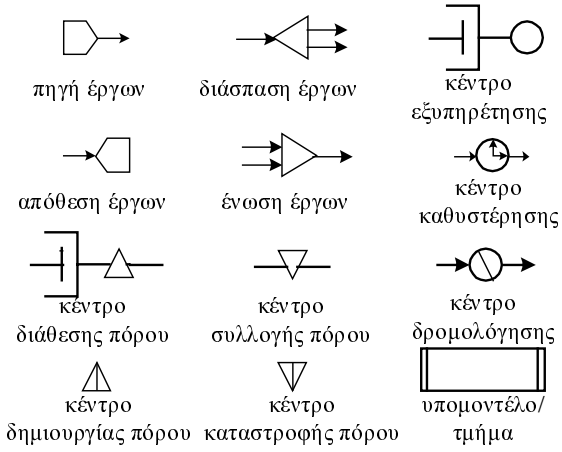
- *Αφίξεις εξαρτώμενες από το φόρτο*: Συχνά, τα δίκτυα υπολογιστών και τα κατανεμημένα συστήματα εφαρμόζουν προωθημένες πολιτικές εξισορρόπησης φόρτου, που κατευθύνουν τις αφίξεις έργων ή πακέτων στα κατάλληλα κέντρα εξυπηρέτησης. Τέτοιου είδους αφίξεις έργων είναι δύσκολο να μοντελοποιηθούν με τα κλασσικά δίκτυα ουρών μορφής γινομένου.

- *Αποκλεισμός*: Το φαινόμενο εμφανίζεται, όταν σε κάποιο κέντρο εξυπηρέτησης παρατηρείται η μέγιστη χωρητικότητα αυτού. Διακρίνουμε δύο τουλάχιστο περιπτώσεις αποκλεισμού: (i) τον *αποκλεισμό μεταφοράς*, όπου τα έργα ολοκληρώνουν την εξυπηρέτησή τους σε κάποιο κέντρο, αλλά δεν προωθούνται στο επόμενο, διότι εκεί παρατηρείται η μέγιστη χωρητικότητα αυτού και (ii) τον *αποκλεισμό εξυπηρέτησης*, όπου σταματάει η παροχή εξυπηρέτησης στα έργα του προηγούμενου κέντρου από αυτό, στο οποίο παρατηρείται η μέγιστη χωρητικότητα. Τέτοιου είδους μοντέλα δεν έχουν λύση μορφής γινομένου και η

ανάλυσή τους γίνεται, είτε προσεγγιστικά με τη χρήση αλγορίθμων αποσύνθεσης, όπως αυτών της [PER94], είτε προσομοιωτικά, όπως στην [KAR94].

- *Ανταγωνισμός*: Οι περισσότερες από τις πειθαρχίες εξυπηρέτησης, που χρησιμοποιούνται στα κλασσικά δίκτυα ουρών, είναι υπερβολικά απλοϊκές για τη μοντελοποίηση πραγματικών συστημάτων. Ακόμη και μία απλή περίπτωση δικτύου ουρών με πειθαρχίες προτεραιοτήτων δε μπορεί να αποτιμηθεί με κανέναν από τους αλγορίθμους για δίκτυα μορφής γινομένου. Ένα πιο σύνθετο παράδειγμα είναι η περίπτωση των τοπικών δικτύων Ethernet, όπου είναι πιθανό ένας αριθμός σταθμών να προσπαθούν να χρησιμοποιήσουν το μέσο μεταφοράς ταυτόχρονα, οπότε ο ανταγωνισμός διευθετείται στη βάση ενός συνόλου κανόνων. Τέτοιου είδους σύνθετες πειθαρχίες είναι γενικά δύσκολο να μοντελοποιηθούν με τη χρήση δικτύων ουρών.
- *Αμοιβαία εξαίρεση*: Συχνά παρατηρείται το φαινόμενο κάποια έργα επεξεργασίας να προσπαθούν να κάνουν ταυτόχρονη χρήση ενός πόρου, οπότε είναι απαραίτητο να συμμορφώνονται σε κάποιους κανόνες αμοιβαίας εξαίρεσης. Τέτοιοι κανόνες δεν αναπαριστώνται εύκολα από τα κλασσικά δίκτυα ουρών. Στην [THO98] αναπτύσσονται μοντέλα απόδοσης με φαινόμενα αμοιβαίας εξαίρεσης, που επίσης επιλύονται προσεγγιστικά μέσω αποσύνθεσης.
- *Μοντελοποίηση πόρων μνήμης*: Η μνήμη είναι ένας πόρος, που διαμοιράζεται σε έναν αριθμό έργων επεξεργασίας. Ο αριθμός αυτός των έργων περιορίζεται από τη διαθέσιμη ποσότητα μνήμης. Αυτού του είδους οι πόροι ονομάζονται *παθητικοί πόροι* και η μοντελοποίησή τους δεν επιτυγχάνεται με δίκτυα ουρών μορφής γινομένου, αλλά μόνο με προσεγγιστικά ή προσομοιωτικά μοντέλα.
- *Ταυτόχρονη κατοχή πόρων*: Όταν έχουμε ταυτόχρονη χρήση δύο πόρων, τότε ο ένας από αυτούς αναπαριστάται ως παθητικός από ένα κέντρο, που διαθέτει μερίδια από ένα σύνολο υπαρχόντων, και ένα άλλο βοηθητικό κέντρο, που απλά τα συλλέγει και τα καθιστά πάλι διαθέσιμα. Τα δίκτυα ουρών του τύπου αυτού επιλύονται μόνο προσεγγιστικά μέσα από αποσύνθεση.

Στο σχήμα 3.4 εικονίζονται τα σύμβολα, τα οποία έχουν επιλεγεί για τη γραφική αναπαράσταση κάποιων από τα χαρακτηριστικά, που αναφέρθηκαν παραπάνω. Τα σύμβολα αυτά δεν είναι άγνωστα στο χώρο, καθώς έχουν χρησιμοποιηθεί προηγούμενα και σε μεγάλης σημασίας βιβλιογραφικές συνεισφορές ([SMI90] και [SC81]).



Σχήμα 3.4 Συμβολισμός κέντρων εξυπηρέτησης για ανοικτά και κλειστά δίκτυα, που δεν είναι μορφής γινομένου (παθητικοί πόροι, διάσπαση/ένωση έργων)

Σύμφωνα με αυτά, που εκτέθηκαν, είναι σαφές ότι μία διαδεδομένη τεχνική προσεγγιστικής επίλυσης δικτύων ουρών, που δεν είναι μορφής γινομένου, είναι η τεχνική της αποσύνθεσης. Στην πιο συνηθισμένη μορφή της τεχνικής αυτής, το μοντέλο διασπάται σε μικρότερα υπομοντέλα. Αυτά ονομάζονται «μέρη» και απαρτίζονται από το χαρακτηριστικό εκείνο, που καταστρατηγεί τις συνθήκες ύπαρξης λύσης μορφής γινομένου, και το υποδίκτυο, που περικλείει (αυτό μπορεί να είναι μορφής γινομένου). Το υπόλοιπο δίκτυο ονομάζεται «συμπλήρωμα».

Πρώτα λοιπόν γίνεται η αποτίμηση της παραγωγής των «μερών», δημιουργώντας κλειστά δίκτυα, με παράκαμψη του «συμπληρώματος» και για διαφορετικούς αριθμούς έργων επεξεργασίας. Ακολούθως, τα «μέρη» αντικαθίστανται στο συνολικό μοντέλο από *κέντρα εξυπηρέτησης ισοδύναμης ροής* (Flow Equivalent Service Centers) και το δίκτυο, που προκύπτει, αποτιμάται, αν αυτό είναι εφικτό, με μία τροποποιημένη έκδοση του αλγορίθμου ανάλυσης μέσων τιμών (MVA), όπως αυτή, που παρατίθεται στο [JAI91].

Ως κέντρα εξυπηρέτησης ισοδύναμης ροής (FESC) χρησιμοποιούνται *κέντρα εξαρτώμενα από το φόρτο*. Οι ρυθμοί εξυπηρέτησης αυτών είναι συνάρτηση του αριθμού των έργων, που βρίσκονται στο κέντρο. Πιο συγκεκριμένα, είναι ίσοι με την παραγωγή του «μέρους», που το συγκεκριμένο κέντρο υποκαθιστά, για όλες τις πιθανές περιπτώσεις αριθμού έργων σε αυτό.

Είναι πάντως σημαντικό να τονιστεί ότι αν και η τεχνική φαίνεται να αποδίδει μία ισοδύναμη αναπαράσταση του ιδίου μοντέλου, στην πραγματικότητα είναι μόνο μία προσέγγιση. Αυτό προκύπτει από το γεγονός ότι ουσιαστικά ολόκληρα υποσυστήματα περιγράφονται από κέντρα εξυπηρέτησης, που εξαρτώνται από το φόρτο. Έτσι, πληροφορίες, που σχετίζονται με την ακριβή θέση των έργων επεξεργασίας στα κέντρα εξυπηρέτησης του υποσυστήματος, ουσιαστικά χάνονται. Η πραγματική κατανομή των μεταξύ των αναχωρήσεων από το εν λόγω «μέρος» χρόνων, υποκαθίσταται από μία επιλεγείσα κατανομή με μέση τιμή την παραγωγή του «μέρους» για τη συγκεκριμένη περίπτωση αριθμού έργων σε αυτό. Στην [CS78] περιγράφεται διεξοδικά η διαδικασία επιλογής της κατάλληλης κατανομής και της πειθαρχίας εξυπηρέτησης του κέντρου FESC.

Η τεχνική της αποσύνθεσης και της χρήσης κέντρων εξυπηρέτησης ισοδύναμης ροής βασίζεται σε σημαντικά θεωρητικά αποτελέσματα. Στην [CHW75] οι συγγραφείς απέδειξαν ότι η τεχνική αποδίδει ακριβή αποτελέσματα για δίκτυα ουρών μορφής γινομένου με ένα μόνο τύπο έργων. Το αποτέλεσμα αυτό ονομάστηκε θεώρημα Norton για δίκτυα μορφής γινομένου, εξαιτίας της αναλογίας του με το θεώρημα του Norton για τα ηλεκτρικά κυκλώματα.

Μία άλλη επίσης σημαντική θεωρητική θεμελίωση της τεχνικής αυτής δίνεται στη [COU75], όπου εφαρμόστηκαν αποτελέσματα της οικονομετρίας, για σχεδόν ολοκληρωτικά αποσυνθέσιμα συστήματα (nearly completely decomposable systems), στην προσεγγιστική ανάλυση της υποκείμενης διαδικασίας Markov κλειστών δικτύων ουρών. Η κατά αυτό τον τρόπο αποσύνθεση του πίνακα τάσεων (ορίζεται στην παράγραφο Α.4 του παραρτήματος) είναι πιο γενική, αν και σε μερικές περιπτώσεις ισοδύναμη με την αποσύνθεση με βάση τη χρήση FESC, όπως αυτή ορίζεται στην [CHW75]. Η προσέγγιση της σχεδόν ολοκληρωτικής αποσυνθεσιμότητας δίνει, σύμφωνα με την [COU75], ικανοποιητικά αποτελέσματα, όταν ο μέσος αριθμός κύκλων, που ένα έργο διαγράφει στο συγκεκριμένο «μέρος» του δικτύου, είναι σχετικά μεγάλος. Βέβαια, στην περίπτωση χρήσης FESC έχουμε αρκετά καλή ακρίβεια ακόμη και όταν ο μέσος αριθμός κύκλων στο υποσύστημα είναι μικρός.

Η αποσύνθεση με τη χρήση FESC βρίσκει εφαρμογή και σε δίκτυα με περισσότερους του ενός τύπους έργων. Στις περιπτώσεις αυτές οι ρυθμοί εξυπηρέτησης του κέντρου FESC είναι διαφορετικοί για κάθε τύπο έργου και ίσοι με την παραγωγή του υποσυστήματος σε έργα του

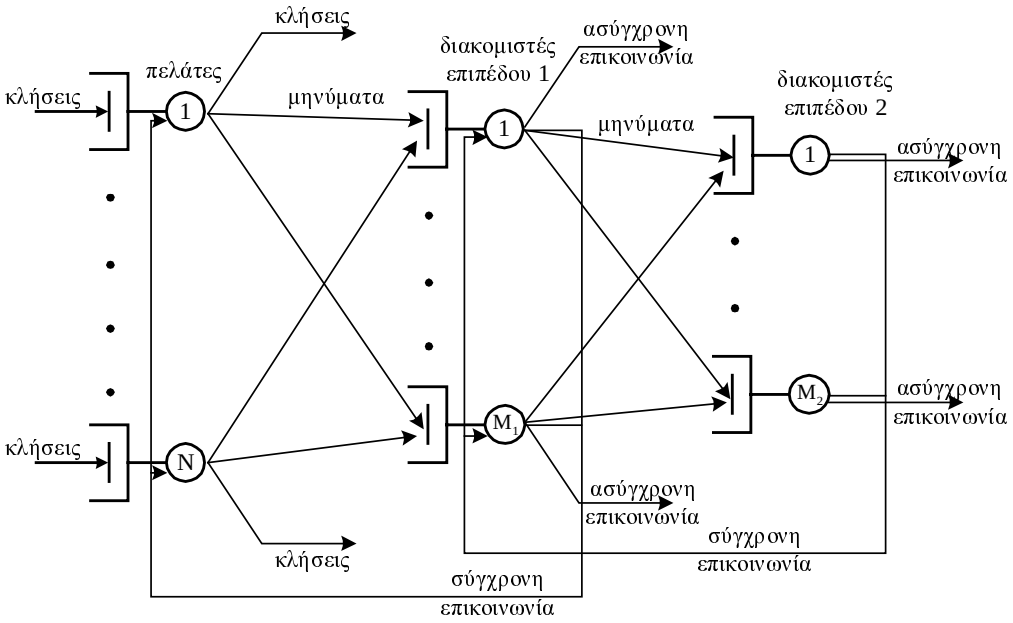
συγκεκριμένου τύπου. Ένα πρόβλημα, που έχει να κάνει με την ύπαρξη περισσότέρων του ενός τύπων έργων, είναι το γεγονός ότι ο αριθμός των μέτρων παραγωγής που πρέπει να υπολογισθούν, αυξάνεται ραγδαία με τον αριθμό των τύπων έργων του δικτύου.

Το πρόβλημα αυτό μπορεί κάποιες φορές να αντιμετωπιστεί με τη βήμα προς βήμα αποσύνθεση του μοντέλου σε πολλά επίπεδα και τη διαδοχική αποτίμηση - με ειδικά αναλυτικά μοντέλα, προσομοίωση, μέτρηση ή ακόμη και μεταμοντέλα, όπως αυτά του κεφαλαίου 6 - των μέτρων παραγωγής από το χαμηλότερο επίπεδο προς το υψηλότερο. Την τεχνική αυτή μπορεί να εκμεταλλευθεί και το εργαλείο HIT, που περιγράφεται στο κεφάλαιο 5 και το οποίο μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής.

3.2 Προσεγγιστικά μοντέλα αποτίμησης της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής

Στην παράγραφο αυτή παρουσιάζονται δύο προσεγγιστικά μοντέλα αποτίμησης της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής. Στην περίπτωση του μοντέλου της [RP00] γίνεται χρήση ενός πολυεπίπεδου δικτύου ουρών για την αναπαράσταση συστημάτων πελάτη - διακομιστή, όπου οι πελάτες και οι διακομιστές επικοινωνούν με σύγχρονα και ασύγχρονα μηνύματα. Στην περίπτωση της [LH00] αναλύεται η απόδοση του μηχανισμού εκτέλεσης συγχρονισμένων μεθόδων σε ένα διακομιστή Enterprise Java Beans, μέσω ενός δικτύου ουρών με ιδιαίτερα χαρακτηριστικά αποκλεισμού.

3.2.1 Ένα πολυεπίπεδο δίκτυο ουρών για συστήματα πελάτη - διακομιστή με σύγχρονα και ασύγχρονα μηνύματα



Σχήμα 3.5 Ένα σύστημα αρχιτεκτονικής πελάτη - διακομιστή με δύο επίπεδα διακομιστών

Κατά κανόνα, η επικοινωνία μεταξύ κατανεμημένων αντικειμένων, που κάνουν χρήση κάποιου ORB ή κάποιας άλλης μορφής διεργασιακής επικοινωνίας, λαμβάνει χώρα με την ανταλλαγή σύγχρονων και ασύγχρονων μηνυμάτων. Στην εργασία των Ramesh και Perros, που έγινε αναφορά προηγούμενα, έχουμε μία σημαντική συνεισφορά στην προσεγγιστική αποτίμηση συστημάτων, όπως αυτό του σχήματος 3.5. Συστήματα του τύπου αυτού

χαρακτηρίζονται από κλήσεις επικοινωνίας, που διαπλέκονται μέσα σε ένα σύνολο διακομιστών κάθε ένας από τους οποίους ανήκει σε κάποιο επίπεδο μιας ιεραρχικής διάταξης, όπως αυτής του σχήματος.

Κάθε ένας από τους πελάτες μπορεί να αποστέλλει σύγχρονα ή ασύγχρονα μηνύματα σε οποιουσδήποτε από τους M_1 διακομιστές του πρώτου επιπέδου. Με τη σειρά τους, κάθε ένας από τους διακομιστές αυτούς κατά τη διάρκεια της επεξεργασίας μιας κλήσης μπορεί να στείλει σύγχρονα ή ασύγχρονα μηνύματα σε οποιουσδήποτε από τους M_2 διακομιστές του δεύτερου επιπέδου. Έστω ότι συνολικά υπάρχουν R ιεραρχικά επίπεδα ανταλλαγής μηνυμάτων. Ένας διακομιστής του επιπέδου 1 μπορεί να λάβει ένα σύγχρονο ή ασύγχρονο μήνυμα μόνο από τους N πελάτες, ενώ ένας διακομιστής του επιπέδου i, $2 \le i \le R$ μπορεί να λάβει μηνύματα μόνο από τους διακομιστές του επιπέδου $i-1$.

Κάθε πελάτης ή διακομιστής αναπαριστάται από μία ουρά αναμονής. Έτσι, μοντελοποιείται ουσιαστικά η ουρά των κλήσεων σε κάθε πελάτη και η ουρά των μηνυμάτων σε κάθε διακομιστή. Το χαρακτηριστικό, που στο μοντέλο αυτό καταστρατηγεί τις συνθήκες ύπαρξης λύσης μορφής γινομένου, είναι ο αποκλεισμός εξυπηρέτησης που προκαλείται στους αποστολείς σύγχρονων μηνυμάτων κατά τη διάρκεια επεξεργασίας αυτών στους διακομιστές.

Σε ένα υποθετικό σενάριο αλληλεπίδρασης πελατών και διακομιστών διαφορετικών επιπέδων, ας υποθέσουμε ότι μετά από κάποια επεξεργασία, ένας πελάτης αποστέλλει το σύγχρονο μήνυμα SM_1 σε κάποιο διακομιστή επιπέδου 1. Ο πελάτης παρουσιάζει αποκλεισμό εξυπηρέτησης και το μήνυμα μπαίνει στην ουρά επεξεργασίας του διακομιστή επιπέδου 1. Όταν ο διακομιστής είναι διαθέσιμος τότε γίνεται η επεξεργασία του μηνύματος SM_1, που προκαλεί ενδεχομένως την αποστολή ενός άλλου σύγχρονου μηνύματος SM_2 σε διακομιστή του επιπέδου 2. Μετά την ολοκλήρωση της επεξεργασίας του SM_2 απελευθερώνεται ο διακομιστής επιπέδου 1, ο οποίος με την ολοκλήρωση της επεξεργασίας του SM_1 αποστέλλει ένα ασύγχρονο μήνυμα AM_2 σε διακομιστή επιπέδου 2. Μετά την επεξεργασία του το AM_2 εγκαταλείπει το σύστημα.

Το πολυεπίπεδο αυτό δίκτυο ουρών, που όπως ήδη τονίστηκε δεν έχει λύση μορφής γινομένου, αναλύεται προσεγγιστικά με τη χρήση ενός επαναληπτικού αλγορίθμου. Έστω λοιπόν ότι στις N ουρές πελατών, καταφθάνουν εξωτερικές κλήσεις εξυπηρέτησης. Ως αποτέλεσμα, ο αντίστοιχος πελάτης μπορεί να χρειαστεί την αποστολή συνολικά K_n μηνυμάτων στην ομάδα των M_1 διακομιστών επιπέδου 1. Αυτό προκαλεί την αποστολή από το διακομιστή m, J_m μηνυμάτων στην ομάδα των M_2 διακομιστών επιπέδου 2.

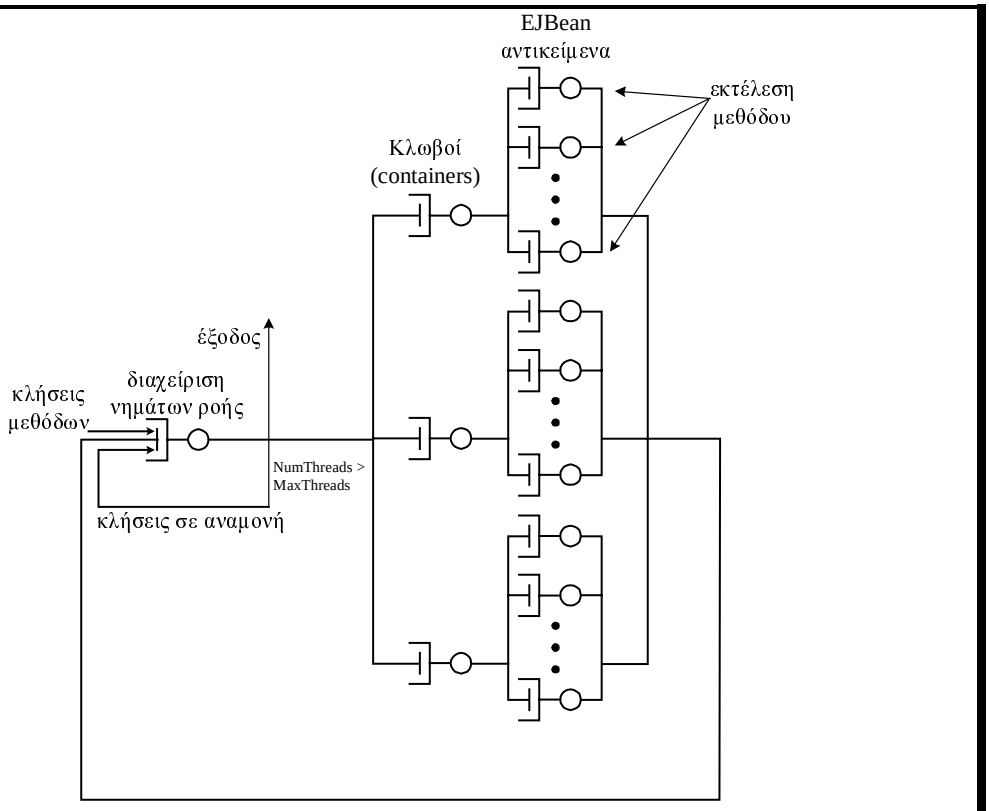
Το σύστημα *αποσυντίθεται* σε τρία υποσυστήματα: το υποσύστημα πελατών, το υποσύστημα διακομιστών επιπέδου 1 και το υποσύστημα διακομιστών επιπέδου 2. Το υποσύστημα πελατών αποτελείται από τις N ουρές πελατών, οι χρόνοι εξυπηρέτησης των οποίων πρέπει να είναι τέτοιοι ώστε να εκφράζουν και τις καθυστερήσεις αποκλεισμού εξυπηρέτησης εξαιτίας της αποστολής σύγχρονων μηνυμάτων. Το υποσύστημα διακομιστών επιπέδου 1 είναι ένα κλειστό δίκτυο ουρών αποτελούμενο από τους M_1 διακομιστές και τους N πελάτες ως κέντρα καθυστέρησης, χωρίς δηλαδή τις ουρές αναμονής που τους συνοδεύουν. Ο χρόνος εξυπηρέτησης κάθε πελάτη τροποποιείται, έτσι ώστε να λαμβάνεται υπόψη ο ανταγωνισμός στην ουρά που το συνοδεύει, ενώ οι χρόνοι εξυπηρέτησης στους M_1 διακομιστές επιπέδου 1 συμπεριλαμβάνουν και τις καθυστερήσεις αποκλεισμού εξυπηρέτησης εξαιτίας της αποστολής σύγχρονων μηνυμάτων σε διακομιστές επιπέδου 2. Το υποσύστημα διακομιστών επιπέδου 2 είναι αντίστοιχα ένα δίκτυο ουρών αποτελούμενο από τους M_2 διακομιστές επιπέδου 2 και τους M_1 διακομιστές επιπέδου 1, ως κέντρα καθυστέρησης.

Η ανάλυση του κάθε υποσυστήματος απαιτεί πληροφορίες, οι οποίες προκύπτουν από την ανάλυση των άλλων υποσυστημάτων. Η κατάσταση αυτή αντιμετωπίζεται με τη χρήση μιας

επαναληπτικής διαδικασίας επίλυσης. Κάθε επανάληψη αποτελείται από ένα προς τα πίσω και ένα προς τα εμπρός πέρασμα. Στην πρώτη περίπτωση αναλύεται πρώτα το υποσύστημα διακομιστών επιπέδου 2 και μετά το υποσύστημα διακομιστών επιπέδου 1. Στη δεύτερη περίπτωση αναλύεται πρώτα το υποσύστημα πελατών και ακολούθως τα υποσύστηματα διακομιστών επιπέδων 1 και 2. Ο αλγόριθμος ξεκινά με ένα προς τα πίσω πέρασμα ακολουθούμενο από ένα προς τα εμπρός πέρασμα. Η επαναληπτική αυτή διαδικασία συνεχίζει την εκτέλεσή της μέχρι την επίτευξη σύγκλισης των λύσεων.

Η διαδικασία άφιξης κλήσεων στους πελάτες συμμορφώνεται στην κατανομή Cox δύο φάσεων (C_2), ενώ στους πελάτες έχουμε εκθετική εξυπηρέτηση και στους διακομιστές μπορεί να χρησιμοποιηθεί είτε η εκθετική κατανομή, είτε η κατανομή Cox δύο φάσεων (C_2). Στην πρώτη περίπτωση, τα υποσυστήματα διακομιστών είναι κλειστά δίκτυα μορφής γινομένου με αλυσίδες τόσες, όσο ο αριθμός των κέντρων καθυστέρησης και εξυπηρέτηση σε αυτά εξαρτημένη από τον τύπο του έργου επεξεργασίας. Ένα τέτοιο δίκτυο μπορεί να επιλυθεί με τον αλγόριθμο ανάλυσης μέσων τιμών ή με τον αλγόριθμο περιέλιξης. Στην περίπτωση όμως εξυπηρέτησης στους διακομιστές με βάση την κατανομή Cox δύο φάσεων (C_2), τότε γίνεται χρήση του προσεγγιστικού αλγορίθμου των Baynat και Dallery [BD96].

3.2.2 Εκτέλεση συγχρονισμένων μεθόδων σε ένα διακομιστή Enterprise Java Beans



Σχήμα 3.6 Δίκτυο ουρών, που αναπαριστά την εκτέλεση συγχρονισμένων μεθόδων σε έναν EJB διακομιστή

Στο σχήμα 3.6 απεικονίζεται το μοντέλο με το οποίο αναπαριστάται η εκτέλεση συγχρονισμένων μεθόδων σε διακομιστές Enterprise Java Beans στην [LH00]. Κάθε κλωβός (βλ. ενότητα 2.6.2 για τη σχετική ορολογία) διαθέτει ένα συγκεκριμένο αριθμό αντικειμένων, που μπορούν να εκτελούνται ταυτόχρονα. Τα αντικείμενα του κάθε EJBean είναι ταυτόχρονα προσπελάσιμα από περισσότερους του ενός πελάτες. Ο συγχρονισμός γίνεται από τον αντίστοιχο κλωβό.

Πιο συγκεκριμένα, όταν δύο διαφορετικοί πελάτες καλούν κάποιο αντικείμενο, πριν από όλα ζητείται ένα κλείδωμα. Το κλείδωμα διατίθεται από τον αντίστοιχο κλωβό μόνο αν το αντικείμενο είναι ενεργό στη μνήμη και όχι κλειδωμένο. Σε διαφορετική περίπτωση η συγκεκριμένη κλήση μεθόδου αποκλείεται από την εξυπηρέτηση στο κέντρο του κλωβού, όπου βρίσκεται μέχρι να της διατεθεί το αναγκαίο για την εκτέλεσή της κλείδωμα. Επιπλέον, υπάρχει συγκεκριμένο όριο στον αριθμό των μεθόδων, που μπορούν να εκτελούνται την ίδια χρονική στιγμή. Όταν λοιπόν ο αριθμός αυτός γίνει ίσος με τον αριθμό των διαθέσιμων νημάτων ροής, τότε οι υπόλοιπες κλήσεις τίθενται σε αναμονή.

Είναι σαφές ότι το συγκεκριμένο δίκτυο ουρών διαθέτει χαρακτηριστικά αποκλεισμού εξυπηρέτησης εξαιτίας επεξεργασίας σε άλλο κέντρο, όπως ακριβώς και το μοντέλο της 3.2.1. Καθώς λοιπόν τα υποσυστήματα κλωβού - αντικειμένων καταστρατηγούν μία από τις συνθήκες ύπαρξης λύσης μορφής γινομένου, αυτά επιλύονται χωριστά και υποκαθίστανται στο συνολικό μοντέλο από αντίστοιχο αριθμό κέντρων εξυπηρέτησης ισοδύναμης ροής (FESCs). Απώτερος στόχος είναι ο υπολογισμός της παραγωγής του κάθε υποσυστήματος, που δίνεται από τη σχέση,

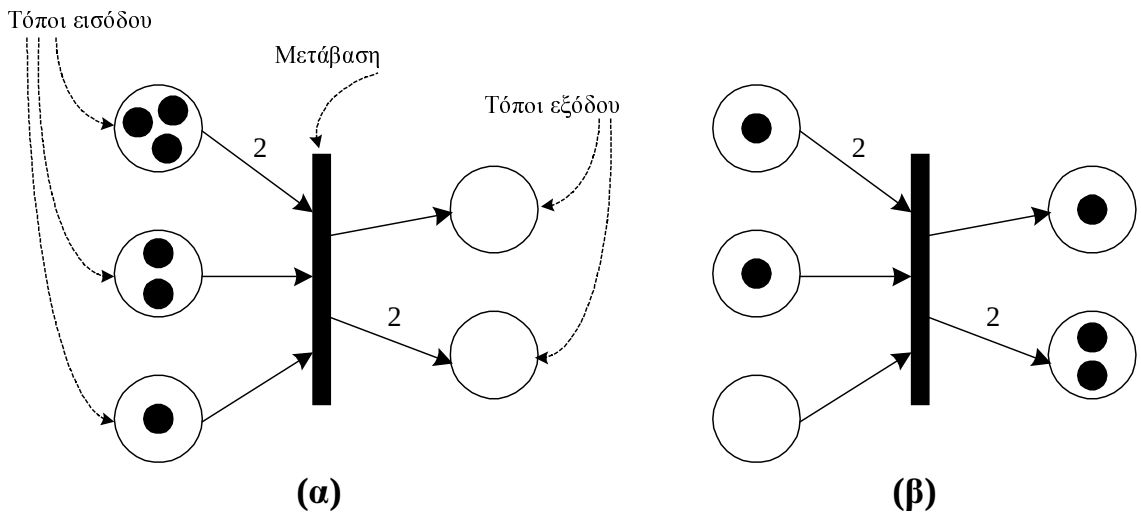
$$T(N) = \sum_{j=1}^{N} p(j) \cdot \mu_1(j)$$

όπου $p(j)$ είναι η πιθανότητα ύπαρξης j έργων στο κέντρο εξυπηρέτησης του κλωβού (άρα N-j έργων στα κέντρα των αντικειμένων) και $\mu_1(j)$ είναι ο ρυθμός εξυπηρέτησης στον κάθε κλωβό λαμβάνοντας υπόψη την πιθανότητα αποκλεισμού εξυπηρέτησης. Τέλος, τα μεγέθη $p(j)$ και $\mu_1(j)$ υπολογίζονται από τους συγγραφείς με κατάλληλες προσεγγιστικές αναπαραστάσεις από μοντέλα Markov και άμεση επίλυση αυτών.

3.3 Μοντέλα απόδοσης με τη χρήση δικτύων Petri

Τα *δίκτυα Petri* (Petri nets) είναι ένας φορμαλισμός κατάλληλος για τη μοντελοποίηση συστημάτων με χαρακτηριστικά σύγχρονης και ασύγχρονης εκτέλεσης, κατανομής ή/και παραλληλίας και μη ντετερμινιστικής/στοχαστικής συμπεριφοράς. Διαθέτουν έναν εύληπτο συμβολισμό και μια ολοκληρωμένη μαθηματική θεμελίωση, που επιτρέπει την τεκμηρίωση της ανάλυσης των ιδιοτήτων τους. Παρόλα αυτά, το σύνολο των εξισώσεων κατάστασης, που περιγράφουν ένα δίκτυο Petri, μεγαλώνει εκθετικά ως προς το μέγεθος αυτού, με αποτέλεσμα η αποτίμηση μεγάλων δικτύων Petri να είναι εφικτή μόνο μέσω προσομοίωσης. Η παρουσίαση των δικτύων Petri, που γίνεται στην παράγραφο αυτή, βασίζεται στις εργασίες [MUR89] και [MBD98], οι οποίες αποτελούν λεπτομερείς αναφορές των σημαντικότερων αποτελεσμάτων και της χρήσης τους στην ανάλυση της απόδοσης συστημάτων.

Ένα δίκτυο Petri είναι ένας κατευθυνόμενος, ζυγισμένος, διμερής γράφος, που αποτελείται από τα εξής είδη κόμβων: τους *τόπους* και τις *μεταβάσεις*. Αποκλείεται η ύπαρξη τόξων που να συνδέουν δύο τόπους ή δύο μεταβάσεις μεταξύ τους. Οι βασικές αρχές ενός δικτύου Petri φαίνονται στο σχήμα 3.7. Σύμφωνα με αυτό, οι τόποι αναπαριστώνται από κύκλους, ενώ οι μεταβάσεις από έντονες μαύρες γραμμές ή παραλληλόγραμμα. Δεν υπάρχει όριο στον αριθμό των εισερχόμενων ή των εξερχόμενων από έναν κόμβο τόξων. Κάθε τόξο συνοδεύεται από το *βάρος* του (θετικός ακέραιος). Ένα τόξο με βάρος k είναι ισοδύναμο με k παράλληλα τόξα. Μία *κατάσταση* ορίζεται με την απόδοση ενός μη αρνητικού ακεραίου σε κάθε τόπο του δικτύου Petri. Ένας τόπος λέμε ότι διαθέτει k *μερίδια* αν κατά τον ορισμό της κατάστασης έχει αποδοθεί ο αριθμός k στο συγκεκριμένο τόπο. Ένα δίκτυο Petri έχει μία *αρχική κατάσταση*, η οποία περιγράφει πως κατανέμονται αρχικά τα μερίδια στους τόπους.



Σχήμα 3.7 Δίκτυο Petri (α) πριν την εκτέλεση και (β) μετά την εκτέλεση

Μία πιθανή ερμηνεία των συστατικών στοιχείων των δικτύων Petri, όταν αυτά χρησιμοποιούνται στην ανάλυση της απόδοσης συστημάτων, είναι η εξής:

- Οι μεταβάσεις εκφράζουν έργα επεξεργασίας,
- οι τόποι εισόδου τους πόρους, που χρειάζονται για ένα έργο επεξεργασίας, και
- οι τόποι εξόδου τους πόρους, που απελευθερώνονται μετά από ένα έργο επεξεργασίας.

Μαθηματικά, ένα δίκτυο Petri ορίζεται ως μία διατεταγμένη πεντάδα $PN = (P, T, F, W, M_0)$, όπου:

$P = p_1, p_2, ..., p_m$ είναι ένα πεπερασμένο σύνολο τόπων,

$T = t_1, t_2, ..., t_n$ είναι ένα πεπερασμένο σύνολο μεταβάσεων,

$F \subseteq (P \times T) \cup (T \times P)$ είναι ένα σύνολο τόξων,

$W : F \rightarrow 1,2,3,.....$ είναι μία συνάρτηση βάρους,

$M_0 : P \rightarrow 0,1,2,3,.....$ είναι η αρχική κατάσταση και

$P \cap T = \emptyset$ και $P \cup T \neq \emptyset$

Ένα δίκτυο Petri $N = (P, T, F, W)$ χωρίς κάποια συγκεκριμένη αρχική κατάσταση αναπαριστάται μόνο από το συμβολικό του όνομα N, ενώ ένα δίκτυο Petri με δεδομένη αρχική κατάσταση συμβολίζεται με (N, M_0).

Η τρέχουσα κατάσταση ενός δικτύου Petri αλλάζει όταν μία ή περισσότερες μεταβάσεις του δικτύου είναι σε θέση να εκτελεστούν. Οι κανόνες μετάβασης είναι:

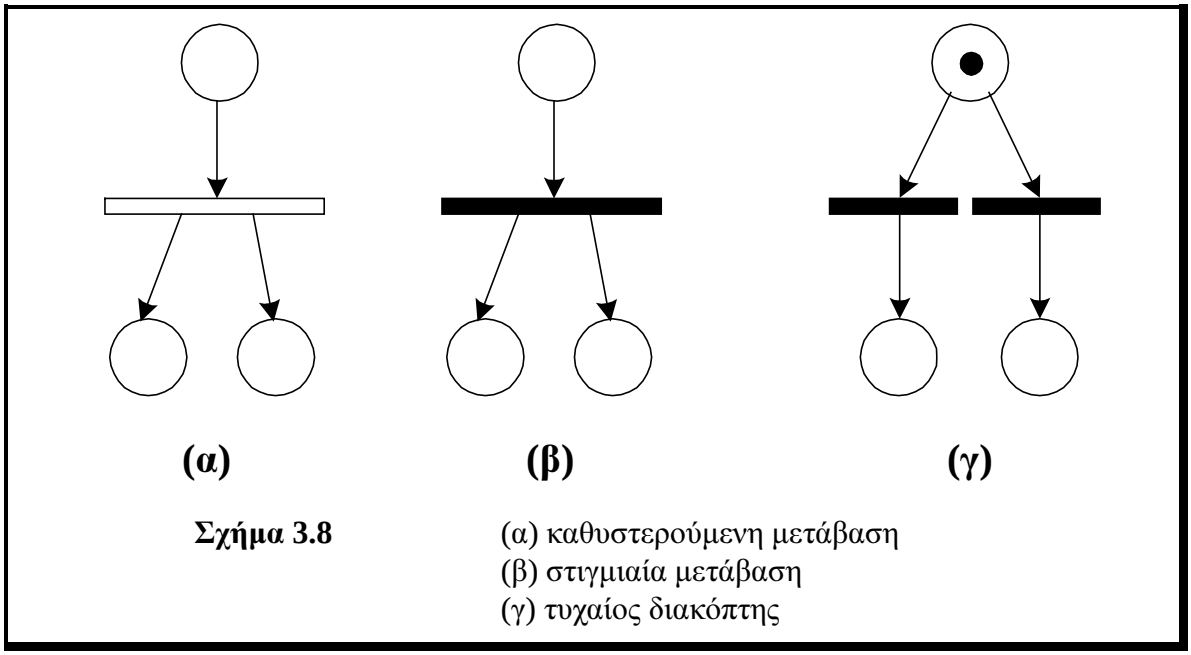
1. Μία μετάβαση t λέμε ότι *είναι σε θέση να εκτελεστεί* αν κάθε τόπος εισόδου p της t διαθέτει $w(p,t)$ μερίδια, όπου $w(p,t)$ είναι το βάρος του τόξου από τον p στην t.
2. Μία μετάβαση, που είναι σε θέση να εκτελεστεί μπορεί ή μπορεί να μην εκτελεστεί (ανάλογα με το αν θα συμβεί το συγκεκριμένο γεγονός ή όχι).
3. Η εκτέλεση μιας μετάβασης t αφαιρεί $w(p,t)$ μερίδια από κάθε τόπο εισόδου p του t και προσθέτει $w(t,p)$ μερίδια σε κάθε τόπο εξόδου p του t, όπου $w(t,p)$ είναι το βάρος του τόξου από την t στον p.

Μεταβάσεις χωρίς τόξα εισόδου ονομάζονται *μεταβάσεις πηγής*. Είναι πάντα σε θέση να εκτελεστούν και όταν εκτελούνται παράγουν μερίδια. Μεταβάσεις χωρίς τόξα εξόδου ονομάζονται *μεταβάσεις απόθεσης* και έχουν την ιδιότητα να «καταναλώνουν» μερίδια χωρίς να παράγουν νέα. Υπάρχει και η περίπτωση η χωρητικότητα ενός τόπου να είναι περιορισμένη, οπότε μιλάμε για *δίκτυα Petri πεπερασμένης χωρητικότητας*. Αν ένας τόπος διαθέτει τόσα μερίδια, που καθιστούν σε θέση εκτέλεσης δύο ή περισσότερες μεταβάσεις, τότε λέμε ότι το δίκτυο Petri είναι *μη ντετερμινιστικό*.

Για την ανάπτυξη μοντέλων απόδοσης, είναι απαραίτητη η εισαγωγή στα δίκτυα Petri της παραμέτρου του χρόνου. Στα κλασσικά δίκτυα Petri, όπως αυτά περιγράφηκαν προηγούμενα, η εκτέλεση της κάθε μετάβασης θεωρείται ότι είναι στιγμιαία.

Στα *στοχαστικά δίκτυα Petri* (Stochastic Petri Nets), σε κάθε μετάβαση αντιστοιχίζεται και μία εκθετικά κατανεμημένη καθυστέρηση. Όταν η μετάβαση έρθει σε θέση εκτέλεσης, τότε η εκτέλεσή της καθυστερεί για χρονικό διάστημα εκθετικά κατανεμημένο με ρυθμό λ. Οι ρυθμοί που αντιστοιχούν στις μεταβάσεις μπορεί να είναι διαφορετικοί. Αν κάποια δεδομένη χρονική στιγμή είναι σε θέση εκτέλεσης περισσότερες της μιας μεταβάσεις, τότε θα εκτελεστεί πρώτη αυτή με τη μικρότερη καθυστέρηση. Επίσης, σύμφωνα με την ιδιότητα απώλειας μνήμης (βλ. παράγραφο Α.2 του παραρτήματος) της εκθετικής κατανομής, η καθυστέρηση της επόμενης εκκρεμούς μετάβασης δεν επηρεάζεται.

Σε μία προσπάθεια περιορισμού του χώρου καταστάσεων των στοχαστικών δικτύων Petri, εισήχθησαν τα *γενικευμένα στοχαστικά δίκτυα Petri* (Generalised Stochastic Petri Nets). Στα GSPN είναι δυνατό να έχουμε και στιγμιαίες αλλά και καθυστερούμενες μεταβάσεις και η διάκρισή τους γίνεται όπως στα σχήματα 3.8α και 3.8β. Έτσι, η χρήση καθυστερούμενων μεταβάσεων μπορεί να περιορίζεται μόνο στις περιπτώσεις, που αυτή κρίνεται απαραίτητη. Οι στιγμιαίες μεταβάσεις, που είναι σε θέση εκτέλεσης, εκτελούνται πάντα πριν από τις καθυστερούμενες. Αν όμως υπάρχουν περισσότερες από μία στιγμιαίες μεταβάσεις σε θέση εκτέλεσης, τότε αυτή που θα εκτελεστεί επιλέγεται με βάση κάποια κατανομή πιθανοτήτων και η διάταξη, που αναπαριστά το φαινόμενο (σχήμα 3.8γ), ονομάζεται *τυχαίος διακόπτης*. Με αυτόν τον τρόπο εισάγονται σε ένα μοντέλο δικτύου Petri πιθανότητες δρομολόγησης.



Σχήμα 3.8
(α) καθυστερούμενη μετάβαση
(β) στιγμιαία μετάβαση
(γ) τυχαίος διακόπτης

Σε ένα *έγχρωμο δίκτυο Petri* (Coloured Petri Net) κάθε μερίδιο έχει και ένα χρώμα. Σε τέτοιου είδους δίκτυα η συμπεριφορά των μεριδίων μπορεί να διαμορφώνεται ανάλογα με το

χρώμα τους, όπως ακριβώς συμβαίνει στα δίκτυα ουρών με τα έργα επεξεργασίας διαφορετικών τύπων.

Αναλυτική περιγραφή διάφορων άλλων κλάσεων στοχαστικών δικτύων Petri, όπως τα

- *ημι-μαρκοβιανά στοχαστικά δίκτυα Petri* (Semi-Markov Stochastic Petri Nets), τα
- τα *στοχαστικά δίκτυα Petri τύπου φάσης* (PHase type Stochastic Petri Nets),
- τα *ντετερμινιστικά στοχαστικά δίκτυα Petri* (Deterministic Stochastic Petri Nets) και
- τα *στοχαστικά δίκτυα Petri αναγέννησης Markov* (Markov Regenerative Stochastic Petri Nets),

δίνεται στην [MBD98].

Τα αποτελέσματα της αποτίμησης δικτύων Petri είναι οι πιθανότητες κατάστασης. Από αυτές είναι δυνατό να υπολογιστούν όλα τα γνωστά μέτρα απόδοσης, όπως παραγωγή, αξιοποίηση και χρόνοι απόκρισης.

Τα δίκτυα Petri είναι κατάλληλα για τη μοντελοποίηση φαινομένων ταυτόχρονης κατοχής πόρων και συντονισμού. Για το λόγο αυτό μπορούν εύκολα να χρησιμοποιηθούν [DF98] για τη μοντελοποίηση κατανεμημένου λογισμικού, όπου έχουμε τη σύγχρονη εκτέλεση ενός αριθμού διεργασιών. Βέβαια και αυτή η τεχνική μοντελοποίησης δεν είναι χωρίς προβλήματα. Ένα από αυτά, στο οποίο ήδη έγινε αναφορά, είναι η ραγδαία αύξηση με το μέγεθος του δικτύου του χώρου καταστάσεων, που συχνά επιβάλλει την προσομοίωση ως τη μόνη λύση για την αποτίμηση τέτοιων μοντέλων. Ένα άλλο πρόβλημα όμως είναι και η δυσκολία στην αναπαράσταση του ανταγωνισμού για χρήση των διαθέσιμων πόρων μέσω π.χ. πειθαρχιών εξυπηρέτησης.

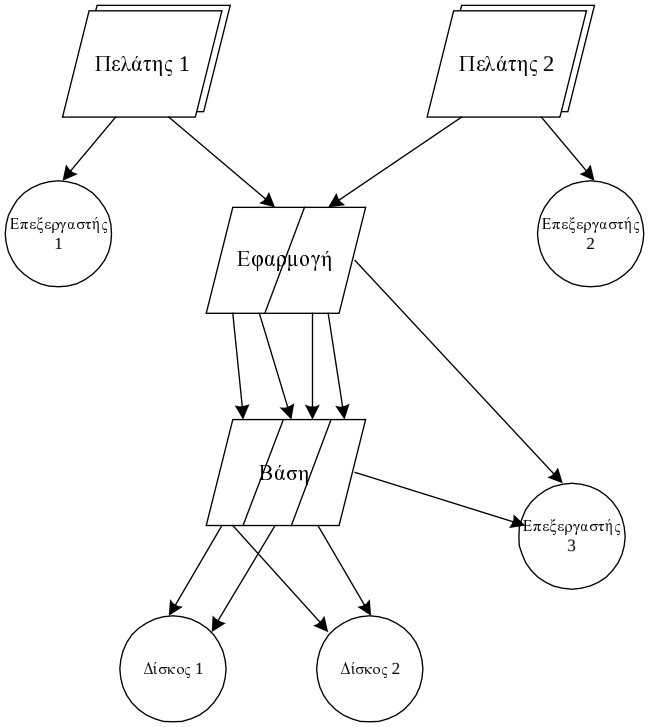
3.4 Στρωματοποιημένα Δίκτυα Ουρών (Layered Queueing Networks)

Τα στρωματοποιημένα δίκτυα ουρών (LQNs) αναπτύχθηκαν ως μία επέκταση των δικτύων ουρών, αρχικά ανεξάρτητα στις [RS95] και [WOO88] και ακολούθως από κοινού στην [WNP95]. Η βασική διαφορά ενός LQN σε σχέση με ένα απλό δίκτυο ουρών είναι το γεγονός ότι ένα κέντρο, στο οποίο καταφθάνουν κλήσεις εξυπηρέτησης και συνωστίζονται στην ουρά αυτού, μπορεί να λειτουργήσει ως πελάτης άλλων κέντρων, από τα οποία αιτεί εξυπηρέτηση, κατά τη διάρκεια της εξυπηρέτησης των δικών του κλήσεων.

Ένα LQN αναπαριστάται από ένα μη κυκλικό γράφο, του οποίου οι κόμβοι - ονομάζονται *εργασίες* - είναι μονάδες λογισμικού και υλικού και του οποίου τα τόξα συμβολίζουν κλήσεις εξυπηρέτησης (σχήμα 3.9). Οι μονάδες λογισμικού ζωγραφίζονται ως παραλληλόγραμμα και οι μονάδες υλικού ως κύκλοι. Οι κόμβοι με εξερχόμενα μόνο τόξα παίζουν ρόλο πελάτη. Οι ενδιάμεσοι κόμβοι με εισερχόμενα και εξερχόμενα τόξα παίζουν ρόλο πελάτη αλλά και διακομιστή και συνήθως αναπαριστούν μονάδες λογισμικού. Οι τελικοί κόμβοι παίζουν ρόλο μόνο κέντρου εξυπηρέτησης και συνήθως αναπαριστούν πόρους υλικού (όπως επεξεργαστές, μονάδες I/O, δίκτυο επικοινωνίας κ.α.). Ένα κέντρο εξυπηρέτησης μπορεί να έχει ένα ή περισσότερα όμοια σημεία εξυπηρέτησης. Μία εργασία σε ένα LQN μπορεί να διαθέτει περισσότερους του ενός τύπους υπηρεσιών, που χαρακτηρίζονται ως *είσοδοι* και ζωγραφίζονται ως τμήματα μέσα στο παραλληλόγραμμο της εργασίας. Κάθε είσοδος έχει το δικό της χρόνο εκτέλεσης και τις δικές της απαιτήσεις εξυπηρέτησης από άλλες υπηρεσίες. Οι εργασίες με περισσότερες της μιας εισόδους έχουν μία μοναδική ουρά αναμονής και οι κλήσεις για τις διαφορετικές εισόδους συνωστίζονται όλες σε αυτήν. Η προκαθορισμένη

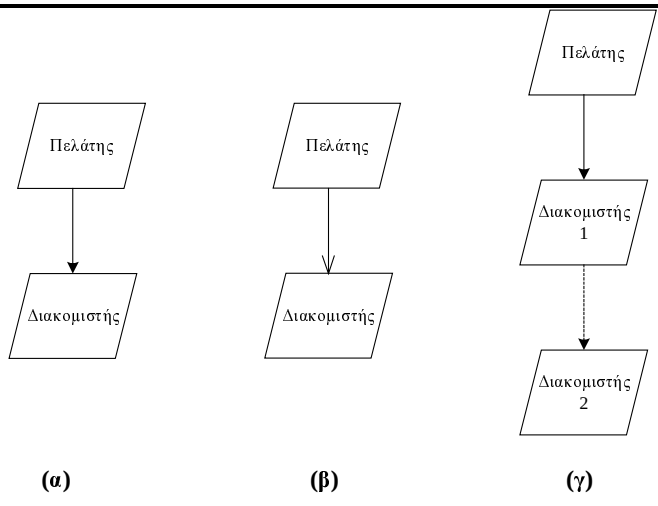
πειθαρχία εξυπηρέτησης είναι η FCFS, αλλά υποστηρίζονται επίσης και άλλες πειθαρχίες εξυπηρέτησης.

Στο σχήμα 3.9 παρουσιάζεται ως παράδειγμα ένα απλό LQN από την [PSJ00].



Σχήμα 3.9 Ένα απλό LQN

Στο μοντέλο του σχήματος εικονίζονται δύο διαφορετικοί τύποι πελατών κάθε ένας με γνωστό αριθμό στοχαστικά όμοιων περιπτώσεων. Οι πελάτες αποστέλλουν κλήσεις για συγκεκριμένη υπηρεσία της εργασίας «Εφαρμογή». Κάθε είσοδος της «Εφαρμογής» απαιτεί υπηρεσίες από δύο διαφορετικές εισόδους της εργασίας «Βάση», που συνολικά διαθέτει τρία είδη υπηρεσιών. Κάθε εργασία λογισμικού εκτελείται σε έναν κόμβο επεξεργαστή, που απεικονίζεται με κύκλο. Στο παράδειγμα, όλοι οι πελάτες του ιδίου τύπου διαμοιράζονται τον ίδιο επεξεργαστή, ενώ η «Εφαρμογή» και η «Βάση» διαμοιράζονται έναν άλλο επεξεργαστή.



Σχήμα 3.10 Μηνύματα στα LQNs: (α) σύγχρονα, (β) ασύγχρονα και (γ) προωθούμενα

Είναι σημαντικό να τονιστεί ότι μία εργασία μπορεί να καλεί άλλες εργασίες στο ίδιο ή σε άλλα στρώματα. Στα LQNs χρησιμοποιούνται τρία είδη μηνυμάτων (σχήμα 3.10): τα *σύγχρονα*, τα *ασύγχρονα* και τα *προωθούμενα* (forwarding).

Ένα σύγχρονο μήνυμα αναπαριστά μία κλήση εξυπηρέτησης από πελάτη σε διακομιστή, όπου ο πελάτης υπόκειται σε αποκλεισμό εξυπηρέτησης μέχρι τη λήψη κάποιας απόκρισης από τον παροχέα της ζητηθείσας υπηρεσίας. Αν ο διακομιστής, που παρέχει τη συγκεκριμένη υπηρεσία, είναι απασχολημένος, τότε η κλήση εξυπηρέτησης μπαίνει στην αντίστοιχη ουρά αναμονής. Όταν έρθει η σειρά της, τότε η κλήση εξυπηρετείται με την εκτέλεση μιας αλληλουχίας *φάσεων*. Στο τέλος της πρώτης φάσης ο διακομιστής αποκρίνεται στον πελάτη, ο οποίος απελευθερώνεται και συνεχίζει την επεξεργασία του. Ο διακομιστής συνεχίζει με την εκτέλεση των επόμενων φάσεων, αν υπάρχουν, ή της επόμενης κλήσης εξυπηρέτησης στην ουρά αναμονής, αν δεν υπάρχουν άλλες φάσεις εξυπηρέτησης. Κατά τη διάρκεια οποιασδήποτε φάσης εξυπηρέτησης ο διακομιστής μπορεί να λειτουργεί ως πελάτης άλλων διακομιστών αιτούμενος την παροχή «συμπεριλήψιμων» υπηρεσιών.

Στην περίπτωση ενός ασύγχρονου μηνύματος ο πελάτης δεν υπόκειται σε αποκλεισμό εξυπηρέτησης μετά την αποστολή του μηνύματος και ο διακομιστής δεν αποκρίνεται.

Τέλος, τα προωθούμενα μηνύματα αντιστοιχούν σε σύγχρονες κλήσεις, που εξυπηρετούνται διαδοχικά από περισσότερους του ενός διακομιστές. Ο πελάτης αποστέλλει μία σύγχρονη κλήση στον πρώτο διακομιστή, που ξεκινάει την επεξεργασία της κλήσης και στο τέλος της πρώτης φάσης την προωθεί σε ένα δεύτερο διακομιστή. Ο πρώτος διακομιστής ενδεχομένως συνεχίζει την επεξεργασία με τις εναπομείνασες φάσεις παράλληλα με το δεύτερο διακομιστή, ο οποίος μετά από μία πρώτη φάση επεξεργασίας αποκρίνεται στον πελάτη και τον απελευθερώνει.

Οι παράμετροι ενός LQN μοντέλου είναι:

- οι τύποι των πελατών και οι αντίστοιχοι πληθυσμοί τους ή οι ρυθμοί αφίξεών τους,
- για κάθε είσοδο εργασίας μονάδας λογισμικού, ο μέσος χρόνος εκτέλεσης ανά φάση,
- για κάθε είσοδο εργασίας μονάδας λογισμικού, που λειτουργεί ως πελάτης κάποιας μονάδας υλικού, ο μέσος χρόνος εξυπηρέτησης από αυτή και ο μέσος αριθμός επισκέψεων ανά φάση της κάθε εισόδου,
- για κάθε είσοδο εργασίας μονάδας λογισμικού, που λειτουργεί ως πελάτης μιας άλλης εισόδου εργασίας, ο μέσος αριθμός επισκέψεων ανά φάση της κάθε αιτούμενης εισόδου,
- για κάθε τόξο κλήσης, η μέση καθυστέρηση μηνύματος και
- για κάθε κόμβο λογισμικού και υλικού, η πειθαρχία εξυπηρέτησης.

Τα αποτελέσματα ενός LQN μοντέλου συνήθως περιλαμβάνουν χρόνους απόκρισης, παραγωγές και αξιοποίηση διακομιστών για διαφορετικούς τύπους κλήσεων. Οι τεχνικές αποτίμησης, που έχουν προταθεί, είναι η *μέθοδος των στρωμάτων* (Method of Layers) στην [RS95] και τα *στοχαστικά δίκτυα συναντήσεων* (Stochastic Rendezvous Networks) στην [WOO88] και αμφότερες βασίζονται σε προσεγγιστικούς αλγορίθμους ανάλυσης μέσων τιμών (MVA). Η μέθοδος των στρωμάτων πιο συγκεκριμένα, *αποσυνθέτει* ένα LQN σε μία σειρά απλών δικτύων ουρών. Για κάθε ένα από αυτά υπολογίζονται κάποια μέτρα απόδοσης, τα οποία χρησιμοποιούνται ως παράμετροι εισόδου στα υπόλοιπα. Έτσι, καταλήγουμε σε μία επαναληπτική τεχνική υπολογισμού ανάλογη εκείνης, που περιγράφηκε στην ενότητα 3.2.1.

Στόχος είναι η εύρεση μιας λύσης, έτσι ώστε οι υπολογιζόμενες τιμές χρόνων απόκρισης και αξιοποίησης να συμπίπτουν σε όλα τα απλά μοντέλα δικτύων ουρών, που προκύπτουν.

3.5 Στοχαστικές διεργασιακές άλγεβρες (stochastic process algebra)

Από την αρχή της εμφάνισής τους, οι διεργασιακές άλγεβρες επικεντρώθηκαν στη συστηματική ανάπτυξη πολύπλοκων κατανεμημένων συστημάτων από μικρότερα δομικά τμήματα. Τα πιο στοιχειώδη τμήματα είναι οι *ενέργειες* και οι *διεργασίες*, που εκτελούν ενέργειες. Μέσω της χρήσης κατάλληλων τελεστών είναι δυνατή μία ιεραρχική, τμηματική περιγραφή του προς μελέτη συστήματος, ενώ με την αξιοποίηση της κατάλληλης αλγεβρικής επεξεργασίας είναι εφικτή η σύγκριση περιγραφών και η επαλήθευση και δομημένη ανάλυση των διεργασιών.

Οι διεργασιακές άλγεβρες που πρωτοεμφανίστηκαν, δηλαδή η LOTOS [BB89], η Communicating Sequential Processes [HOA85] και η CCS [MIL89] επικεντρώθηκαν στη λειτουργική περιγραφή και ανάλυση συστημάτων, όχι όμως και στη συμπεριφορά αυτών σε σχέση με την παράμετρο του χρόνου. Οι στοχαστικές διεργασιακές άλγεβρες ενσωματώνουν συγκεκριμένα στοχαστικά χαρακτηριστικά στις κλασσικές διεργασιακές άλγεβρες, επιτρέποντας έτσι την περιγραφή και την ανάλυση ποσοτικών ιδιοτήτων των συστημάτων που μελετώνται.

Στη γλώσσα μιας στοχαστικής διεργασιακής άλγεβρας οι ενέργειες μπορεί είτε να είναι στιγμιαίες, είτε καθυστερούμενες, οπότε η καθυστέρηση μπορεί να είναι εκθετική ή όχι. Μία περιγραφή συστήματος μπορεί με τη χρήση συγκεκριμένων δομικών λειτουργικών κανόνων να μετασχηματιστεί σε ένα διάγραμμα καταστάσεων - μεταβάσεων σημασμένο από τις ενέργειες των διεργασιών. Αν οι καθυστερήσεις των ενεργειών είναι εκθετικά κατανεμημένες, τότε είναι εύκολο να αποδειχθεί ότι το διάγραμμα καταστάσεων - μεταβάσεων αντιπροσωπεύει μία στοχαστική διαδικασία Markov σε συνεχή χρόνο. Επειδή συνήθως ο χώρος καταστάσεων της συγκεκριμένης διαδικασίας δεν είναι αρκετά μικρός, ώστε αυτή να είναι αριθμητικά επιλύσιμη, κάθε στοχαστική διεργασιακή άλγεβρα διαθέτει ένα σύνολο σχέσεων ισοδυναμίας (bisimulation equivalence), που διευκολύνουν την αποσύνθεση μιας διαδικασίας Markov σε μικρότερα επιλύσιμα τμήματα. Η τεχνική είναι ανάλογη αυτής, που περιγράφεται στην ενότητα 3.1.4 για την τμηματική αποτίμηση «μερών» ενός δικτύου ουρών και την υποκατάστασή τους στο συνολικό μοντέλο από κέντρα εξυπηρέτησης ισοδύναμης ροής.

Ωστόσο, το σημαντικό πλεονέκτημα, που συνοδεύει την περιγραφή ενός συστήματος με τη χρήση κάποιας στοχαστικής διεργασιακής άλγεβρας, είναι η δυνατότητα ανάλυσης μέσα από το ίδιο μοντέλο όχι μόνο ποσοτικών, αλλά και λειτουργικών ιδιοτήτων του συστήματος, όπως η πιθανότητα μπλοκαρίσματος επεξεργασίας (deadlock), η εκτίμηση της δυνατότητας προσέγγισης κάποιας συγκεκριμένης κατάστασης, η αξιοπιστία του συστήματος κ.α.

Οι στοχαστικές διεργασιακές άλγεβρες αποτελούν σήμερα ένα ραγδαία αναπτυσσόμενο ερευνητικό πεδίο, στο οποίο έχουν ήδη κάνει την εμφάνιση τους τα πρώτα εργαλεία περιγραφής και αποτίμησης μοντέλων. Τα σημαντικότερα από αυτά είναι το TIPPtool ([HHK00]), το PEPA Workbench ([HIL94], [GH94]) και το Two Towers ([BCS98]). Αξίζει τέλος να γίνει αναφορά στις προσπάθειες που γίνονται ([CLA00], [BBG98]), για την άρση του περιορισμού των εκθετικών χρόνων καθυστέρησης των ενεργειών, οι οποίες οδηγούν αναπόφευκτα στην αποτίμηση πιο σύνθετων στοχαστικών διαδικασιών.

Κεφάλαιο 4 *Στοχαστική Προσομοίωση Διακριτών Γεγονότων Δικτύων Ουρών*

Η χρήση αναλυτικών μεθόδων για την επίλυση μοντέλων δικτύων ουρών στις πιο πολλές περιπτώσεις προϋποθέτει, όπως φάνηκε από το κεφάλαιο 3, την αποδοχή μιας σειράς απλουστευτικών υποθέσεων, που ως στόχο έχουν είτε τον περιορισμό της πολυπλοκότητας του μοντέλου, είτε μία λύση μορφής γινομένου. Έτσι, παρά το συγκριτικά υψηλότερο υπολογιστικό κόστος, συχνά προτιμάται η ανάπτυξη προσομοιωτικών μοντέλων για την προσεγγιστική αποτίμηση της απόδοσης τμημάτων ή ακόμη και του συνόλου μιας διάταξης κατανεμημένης αρχιτεκτονικής. Στο κεφάλαιο αυτό παρουσιάζονται κάποιες από τις πιο προωθημένες τεχνικές προσομοίωσης, οι οποίες μελετήθηκαν σε βάθος, τόσο όσον αφορά την αποτελεσματικότητα, όσο και τη δυνατότητα εφαρμογής αυτών σε πολύπλοκα μοντέλα δικτύων ουρών. Επίσης, παρουσιάζονται αξιόλογα ερευνητικά αποτελέσματα σχετικά με την αξιοποίηση των τεχνικών αυτών σε παράλληλες/κατανεμημένες διατάξεις προσομοίωσης για την επίτευξη υποπολλαπλάσιων χρόνων εκτέλεσης και ικανοποιητικής στατιστικής ακρίβειας.

4.1 Αρχές στοχαστικής προσομοίωσης διακριτών γεγονότων

Η προσομοίωση ορίζεται στο [PRI84] ως η αναπαράσταση της δυναμικής συμπεριφοράς ενός *συστήματος* με τη μετακίνησή του από κατάσταση σε κατάσταση σύμφωνα με ένα καλά ορισμένο πλαίσιο λειτουργικών κανόνων. Σύστημα είναι ένα σύνολο από αλληλοσχετιζόμενες *οντότητες* κάθε μία από τις οποίες έχει τις δικές της *ιδιότητες*. Το σύνολο των τιμών των ιδιοτήτων όλων των οντοτήτων ορίζει μία *κατάσταση συστήματος*. Καθώς λοιπόν οι δραστηριότητες των οντοτήτων αλληλεπιδρούν μεταξύ τους, προκαλούν αλλαγές στην κατάσταση του συστήματος. Αν ορίσουμε την κατάσταση του συστήματος ως ένα διάνυσμα *S(t)* τυχαίων μεταβλητών, που μεταβάλλονται με το χρόνο, τότε η μελέτη του *S(t)* για μία χρονική περίοδο *T* λέμε ότι αποδίδει την *λειτουργική πορεία* του συστήματος στην *περίοδο παρατήρησης T*.

Τα συστήματα, τα οποία μελετώνται στη διατριβή αυτή, εμπίπτουν στην κατηγορία των *συστημάτων διακριτών γεγονότων* (discrete - event systems), όρος που χρησιμοποιείται για να αποδώσει το γεγονός ότι η κατάσταση των συστημάτων αυτών αλλάζει ακαριαία σε συγκεκριμένες χρονικές στιγμές. Προσομοιώσεις διακριτών γεγονότων είναι τόσο οι

προσομοιώσεις δικτύων ουρών, όσο και οι προσομοιώσεις δικτύων Petri. Αντίθετα, στα *συνεχή συστήματα* η κατάστασή τους μεταβάλλεται *συνεχώς* με την πάροδο του χρόνου.

Αν σε ένα πείραμα προσομοίωσης το σύστημα περιέχει μόνο μη τυχαίες οντότητες, τότε μιλάμε για *ντετερμινιστική προσομοίωση*. Σε ένα ντετερμινιστικό μοντέλο όλες οι μαθηματικές και λογικές σχέσεις μεταξύ των μεταβλητών του συστήματος είναι εκ των προτέρων γνωστές και δεν υπόκεινται σε τυχαίους παράγοντες. Ένα τυπικό παράδειγμα τέτοιας προσομοίωσης είναι ένα πολύπλοκο και αναλυτικά μη επιλύσιμο σύστημα διαφορικών εξισώσεων, το οποίο μπορεί για παράδειγμα να περιγράφει μία χημική αντίδραση. Από την άλλη μεριά ένα μοντέλο, το οποίο περιέχει μία τουλάχιστο τυχαία μεταβλητή εισόδου, ονομάζεται *στοχαστικό*.

Όταν η μελέτη του συστήματος γίνεται σε *διακριτές χρονικές στιγμές*, τότε η λειτουργική πορεία ενός τέτοιου πειράματος προσομοίωσης εκφράζεται ως μία σειρά καταστάσεων συστήματος, $\{S(t_1), S(t_2), ...\}$. Όταν όμως η παράμετρος του χρόνου θεωρείται ότι είναι *συνεχής*, τότε η λειτουργική πορεία του συστήματος εκφράζεται ως συνάρτηση $\{S(t), t \in (0,T)\}$, η οποία όμως αντανακλά τις ακαριαίες αλλαγές της κατάστασης του συστήματος κατά τις χρονικές στιγμές $t_1 < t_2 < .....$ εμφάνισης *γεγονότων*. Είναι σαφές ότι η προσομοιωτική μελέτη δικτύων ουρών έχει πρακτικό ενδιαφέρον, όταν διεξάγεται ως πείραμα στοχαστικής προσομοίωσης διακριτών γεγονότων συνεχούς χρόνου.

Στο [SHE93] η προσομοίωση διακριτών γεγονότων, σε έναν πεπερασμένο ή άπειρο αλλά αριθμήσιμο χώρο καταστάσεων, ορίζεται ως μία *Γενικευμένη Ημι-Μαρκοβιανή Διαδικασία* (Generalized Semi-Markov Process) και τίθεται έτσι ένα μαθηματικό υπόβαθρο για τη μελέτη αυτής.

Άτυπα, μια GSMP είναι μια στοχαστική διαδικασία, η οποία κάθε φορά που προκαλείται κάποιο γεγονός, που θα μπορούσε να συμβεί από την τρέχουσα κατάσταση αυτής, εκτελεί μία μετάβαση σε μία νέα κατάσταση. Κάθε ένα από τα γεγονότα, που είναι πιθανό να συμβούν από την τρέχουσα κατάσταση, ανταγωνίζεται για την πρόκληση της επόμενης μετάβασης και κάθε ένα από αυτά διαθέτει τη δικιά του κατανομή για τον καθορισμό της νέας κατάστασης. Σε κάθε μετάβαση της GSMP «προγραμματίζονται» νέα γεγονότα. Σε κάθε ένα από αυτά αντιστοιχεί ένα «ρολόι», που καθορίζει τη χρονική στιγμή εμφάνισης του γεγονότος με έναν ανεξάρτητο (στοχαστικό) τρόπο. Αν ένα «προγραμματισμένο» γεγονός δεν προκαλέσει τελικά την επόμενη μετάβαση, αλλά η ύπαρξή του δεν ακυρώνεται από τη νέα κατάσταση, τότε το «ρολόι» του συνεχίζει να τρέχει. Σε διαφορετική περίπτωση το γεγονός ακυρώνεται και το «ρολόι» του παύει να λειτουργεί.

Έχει αποδειχθεί ([SHE93]) ότι αν η $\{X(t); t \geq 0\}$ είναι μία αδιαχώριστη (βλ. παράγραφο Α.3 του παραρτήματος) GSMP με πεπερασμένο χώρο καταστάσεων, τότε η *οριακή κατανομή* ή *κατανομή σε κατάσταση στατιστικής ισορροπίας* της X υπάρχει, δηλαδή το δεξί μέρος της

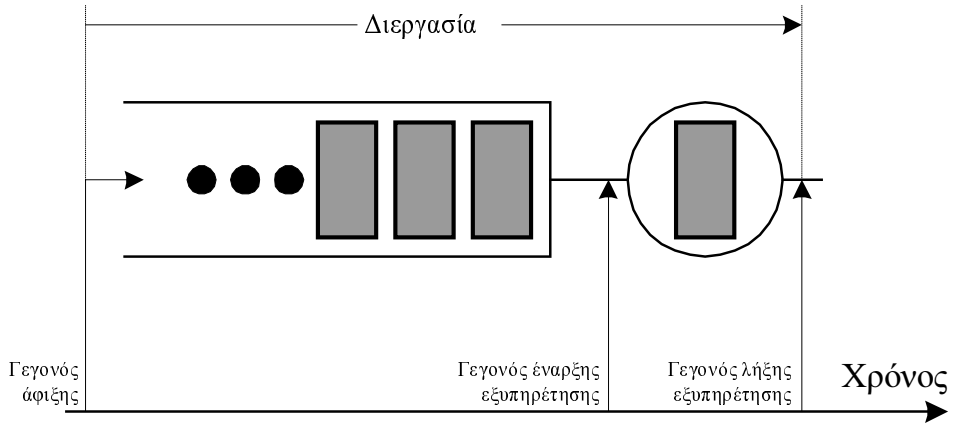
$$\lim_{t \to \infty} P\{X(t) \leq x\} = P\{X \leq x\}$$

είναι συνεχές για όλα τα x.

Πρακτικά, ένα μοντέλο προσομοίωσης διακριτών γεγονότων μπορεί να σχηματιστεί:

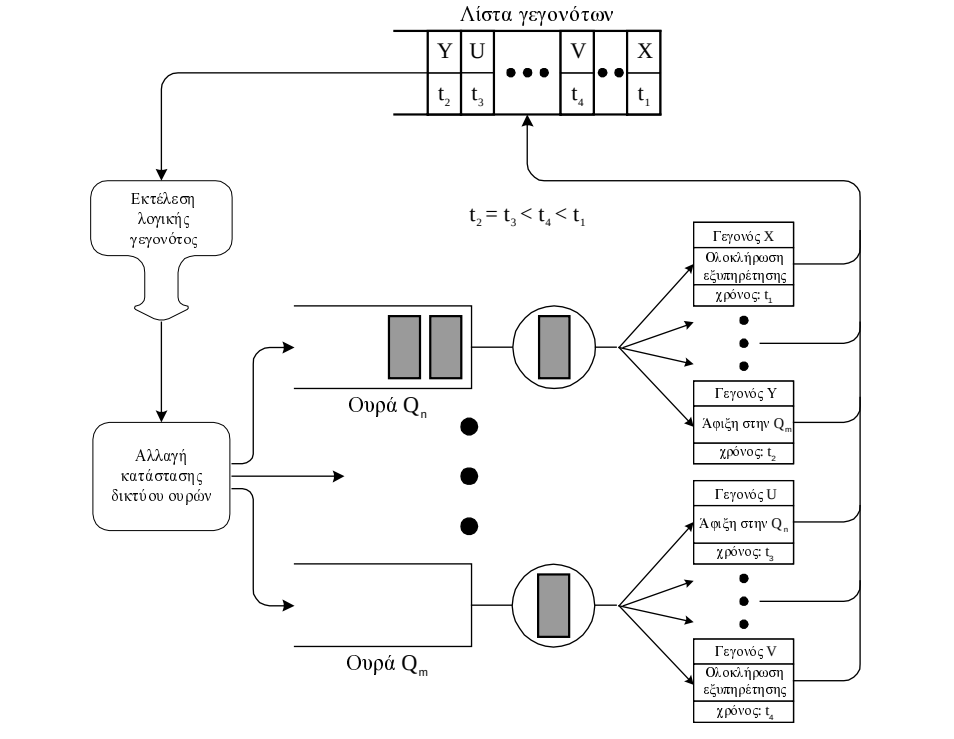
- με τον ορισμό των αλλαγών της κατάστασης του συστήματος, που συμβαίνουν για κάθε *γεγονός*, ή
- με την περιγραφή των *διεργασιών* μέσα από τις οποίες περνάνε οι οντότητες του συστήματος.

Στο σχήμα 4.1 εικονίζονται τα γεγονότα, που χαρακτηρίζουν τη συμπεριφορά μιας διεργασίας, όπως είναι αυτή της ουράς αναμονής και εξυπηρέτησης έργων επεξεργασίας.



Σχήμα 4.1 Προσομοίωση διακριτών γεγονότων μιας ουράς αναμονής και εξυπηρέτησης

Σε μία προσομοίωση δικτύου ουρών, πρώτα γίνεται αναγνώριση των γεγονότων που είναι δυνατό να αλλάξουν την κατάσταση του συστήματος (π.χ. ολοκλήρωση εξυπηρέτησης, άφιξη έργου επεξεργασίας κ.α.) και ακολούθως περιγράφεται η λογική που αφορά κάθε ένα από αυτά. Το πείραμα της προσομοίωσης (σχήμα 4.2) λαμβάνει χώρα με την εκτέλεση της λογικής των εκκρεμών γεγονότων με χρονολογική σειρά. Για το σκοπό αυτό γίνεται χρήση μιας δομής δεδομένων, γνωστής ως *λίστα γεγονότων*. Η λίστα γεγονότων είναι ταξινομημένη με χρονολογική σειρά και κατά τη διάρκεια του πειράματος προσομοίωσης αφαιρούνται γεγονότα από αυτή και προωθείται η εκτέλεσή τους. Όταν η νέα κατάσταση του συστήματος προκαλεί τον «προγραμματισμό» νέων γεγονότων, τότε αυτά εισέρχονται στην κατάλληλη θέση της λίστας.



Σχήμα 4.2 Προσομοίωση διακριτών γεγονότων δικτύου ουρών

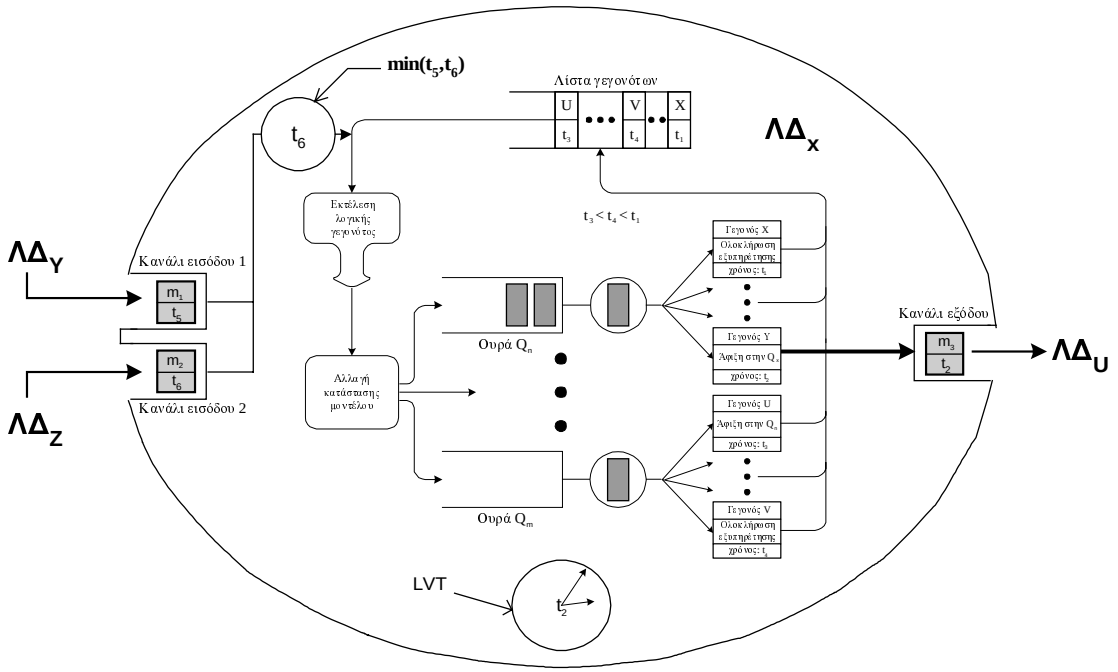
Ο προγραμματισμός εκτέλεσης γεγονότων σε συγκεκριμένες χρονικές στιγμές γίνεται σύμφωνα με κατανομές, οι οποίες σε κάθε περίπτωση ορίζονται από τις παραμέτρους εισόδου του δικτύου ουρών.

Στην [ΚΑΤ93] περιγράφεται ένα λογισμικό προσομοίωσης δικτύων ουρών με μη αναλυτικά επιλύσιμα χαρακτηριστικά, όπως αυτά, που εικονίζονται στο σχήμα 3.4. Το λογισμικό αναπτύχθηκε στη γλώσσα SIMSCRIPT II.5 και επιτρέπει την προσομοίωση ανοικτών ή κλειστών δικτύων ουρών με περισσότερους του ενός τύπους έργων, στοχαστική δρομολόγηση έργων επεξεργασίας και πειθαρχίες προτεραιοτήτων.

4.2 Παράλληλη/Κατανεμημένη προσομοίωση

Η παράλληλη/κατανεμημένη προσομοίωση έχει ως στόχο τον κατακερματισμό της εκτέλεσης μεγάλων και πολύπλοκων μοντέλων, όπως αυτά που προκύπτουν από την ανάλυση λογισμικού κατανεμημένης αρχιτεκτονικής, σε περισσότερους του ενός επεξεργαστές. Στόχος είναι η εκμετάλλευση της υπολογιστικής ισχύος των σύγχρονων συστημάτων πολυεπεξεργασίας για τη επίτευξη ταχύτερων χρόνων εκτέλεσης. Η τεχνική της έχει μελετηθεί σε διαφόρους τύπους προσομοίωσης διακριτών γεγονότων· στη μεγάλη πλειοψηφία των περιπτώσεων σε δίκτυα ουρών και δίκτυα Petri, αλλά και σε κυψελωτά αυτόματα (cellular automata) [ΤΜ87]. Για τους σκοπούς της μελέτης αυτής όμως, η χρήση της επικεντρώνεται μόνο στην περίπτωση της προσομοίωσης δικτύων ουρών.

Σε μία παράλληλη/κατανεμημένη προσομοίωση, έχουμε κατακερματισμό των υπολογιστικών επεξεργασιών σε έναν αριθμό *Λογικών Διεργασιών* (ΛΔ). Κάθε ΛΔ διαθέτει τη δικιά της λίστα γεγονότων και έναν αριθμό ουρών του δικτύου, ανάλογα με τον τρόπο με τον οποίο κατακερματίζεται το μοντέλο. Διακρίνουμε βασικά τρεις μηχανισμούς παράλληλης/κατανεμημένης προσομοίωσης.



Σχήμα 4.3 Προσομοίωση Chandy – Misra: Η $ΛΔ_x$ στέλνει ένα γεγονός άφιξης με χρόνο t_2 στην $ΛΔ_U$ και ακολούθως εκτελεί το γεγονός m_2 μόνο αν $t_6<t_3$

Στην περίπτωση ([CM79]) του *μηχανισμού συντηρητικής εκτέλεσης*, που είναι επίσης γνωστός ως *προσομοίωση Chandy - Misra*, κάθε ΛΔ περιλαμβάνει ένα υπομοντέλο του συνολικού

δικτύου ουρών και η εκτέλεσή της αντιστοιχίζεται σε ένα συγκεκριμένο επεξεργαστή του χρησιμοποιούμενου συστήματος πολυεπεξεργασίας (σχήμα 4.3). Η λίστα γεγονότων κάθε ΛΔ χρησιμοποιείται μόνο για τα επονομαζόμενα εσωτερικά γεγονότα, τα γεγονότα δηλαδή που προγραμματίστηκαν μέσα στην ίδια ΛΔ. Η *τοπική ώρα* (Local Virtual Time) αντιπροσωπεύει τη χρονική στιγμή, που συνέβη το τελευταίο γεγονός στη ΛΔ. Η εκτέλεση εξωτερικών γεγονότων δεν πρέπει να παραβιάζει τον *τοπικό περιορισμό αιτιότητας* (local causality constraint), που επιβάλλει την επεξεργασία των γεγονότων σε μη φθίνουσα σειρά χρονικών στιγμών. Κάτω λοιπόν από την προϋπόθεση ότι το υποκείμενο σύστημα επικοινωνίας διασφαλίζει την παραλαβή των μηνυμάτων με τη σειρά αποστολής τους, ο τοπικός περιορισμός αιτιότητας επιτυγχάνεται με την εφαρμογή δύο κανόνων:

- Του *κανόνα αναμονής εισόδου*, ο οποίος ικανοποιείται όταν μία ΛΔ δεν επεξεργάζεται εξωτερικά γεγονότα μέχρι τη στιγμή, που θα διαθέτει ένα τουλάχιστο σε κάθε ένα από τα κανάλια εισόδου που διαθέτει. Τότε επιλέγεται για επεξεργασία το εξωτερικό γεγονός, που είναι προγραμματισμένο να συμβεί πιο σύντομα.
- Του *κανόνα αναμονής εξόδου*, ο οποίος ισχύει όταν μία ΛΔ δεν αποστέλλει εξωτερικά γεγονότα σε άλλες ΛΔ παρά μόνο αν διασφαλίσει το μη προγραμματισμό στο μέλλον εξωτερικών γεγονότων σε συντομότερο χρόνο εκτέλεσης. Με την υπόθεση ισχύος του κανόνα αναμονής εισόδου, αυτό επιτυγχάνεται όταν μία ΛΔ δε στέλνει γεγονότα προγραμματισμένα σε χρονικές στιγμές μεταγενέστερες της τοπικής ώρας.

Έτσι, στην προσομοίωση Chandy - Misra κάθε ΛΔ εκτελεί επεξεργασία γεγονότων βασισμένη στα γεγονότα της λίστας που διαθέτει, αλλά επεξεργάζεται και εξωτερικά γεγονότα από τα αντίστοιχα κανάλια εισόδου αν ο ελάχιστος από τους χρόνους εκτέλεσης αυτών είναι μικρότερος από το χρόνο εκτέλεσης του επόμενου εσωτερικού γεγονότος (σχήμα 4.3).

Παρόλα αυτά, ο μηχανισμός εκτέλεσης Chandy - Misra, εξαιτίας της εφαρμογής του κανόνα αναμονής εισόδου, είναι αλληλένδετος με το ενδεχόμενο μπλοκαρίσματος επεξεργασίας. Αυτό συμβαίνει γιατί, όταν σε μία ΛΔ δεν υπάρχουν εξωτερικά γεγονότα σε όλα τα κανάλια εισόδου, αυτή παύει την επεξεργασία γεγονότων. Αυτό έχει ως αποτέλεσμα την μη παραγωγή εξωτερικών γεγονότων, που συνεπάγεται το διαδοχικό μπλοκάρισμα και των υπολοίπων ΛΔ.

Δύο μηχανισμοί αντιμετώπισης του προβλήματος έχουν προταθεί: η αποφυγή μπλοκαρίσματος [CM79] και η ανίχνευση μπλοκαρίσματος και επαναφορά ([CM81], [MIS86]).

Στην πρώτη περίπτωση, σε κάθε κανάλι εξόδου αποστέλλονται «κενά» γεγονότα (null messages) ως εγγύηση της μη αποστολής στο μέλλον άλλων γεγονότων με χρονική στιγμή εκτέλεσης μικρότερη αυτών. Αυτό συμβαίνει κάθε φορά, που μία ΛΔ επεξεργάζεται ένα γεγονός, το οποίο δε δημιουργεί εξωτερικά γεγονότα. Έτσι, δίνεται η δυνατότητα στην παραλήπτρια ΛΔ να διασφαλίσει το γεγονός ότι έχει λάβει στο συγκεκριμένο κανάλι εισόδου όλα τα γεγονότα που έχουν προγραμματιστεί να συμβούν μέχρι και τη συγκεκριμένη χρονική στιγμή, οπότε αυτή μπορεί να συνεχίσει την επεξεργασία της.

Στη δεύτερη περίπτωση η προσομοίωση εκτελείται μέχρι τη στιγμή που θα περιπέσει σε κατάσταση μπλοκαρίσματος. Στη συνέχεια, επιλέγεται μία ΛΔ για την οποία είναι γνωστό ότι δεν πρόκειται να υπάρξουν γεγονότα στα κανάλια εισόδου που διαθέτει και η τοπική της ώρα προωθείται στη χρονική στιγμή του επομένου εσωτερικού γεγονότος, επιτρέποντας έτσι την εκτέλεση αυτού. Βέβαια, στην περίπτωση αυτή, είναι απαραίτητη η ύπαρξη ενός κεντρικού

σημείου ελέγχου ή εναλλακτικά η χρήση κάποιου κατανεμημένου αλγορίθμου διαχείρισης μπλοκαρίσματος, όπως αυτού της [CMH83].

Ένας άλλος μηχανισμός εκτέλεσης είναι η επονομαζόμενη *βέλτιστη προσέγγιση*, επίσης γνωστή και ως *μηχανισμός Time Warp* ([JEF85], [JS85]). Ο μηχανισμός βέλτιστης εκτέλεσης σε αντίθεση με αυτόν της συντηρητικής εκτέλεσης δεν επιβάλλει τη συμμόρφωση στον τοπικό περιορισμό αιτιότητας. Αντί για αυτό, οι ΛΔ επιτρέπουν την εμφάνιση λαθών αιτιότητας και παρέχουν ένα μηχανισμό επαναφοράς αυτών στην πριν την εμφάνιση του λάθους κατάσταση. Πιο συγκεκριμένα, αν σε μία ΛΔ καταφθάσει εξωτερικό γεγονός του οποίου ο χρόνος εκτέλεσης είναι μικρότερος από το χρόνο γεγονότων, που ήδη εκτελέστηκαν, τότε η ΛΔ επιστρέφει στην πιο πρόσφατη αποθηκευμένη κατάσταση της ιστορίας του πειράματος προσομοίωσης, που είναι συνεπής με την άφιξη του εν λόγω εξωτερικού γεγονότος. Αν στο εντωμεταξύ η ΛΔ έχει ήδη αποστείλει ένα ή περισσότερα εξωτερικά γεγονότα από τη χρονική στιγμή επιστροφής, τότε για κάθε ένα από αυτά στέλνονται *αντι-γεγονότα* (antimessages). Με την παραλαβή ενός αντι-γεγονότος από μία ΛΔ υπάρχουν δύο πιθανές περιπτώσεις:

- Το γεγονός στο οποίο αναφέρεται έχει ήδη παραληφθεί, οπότε αν μεν δεν έχει εκτελεσθεί απλά ακυρώνεται, ενώ αν έχει εκτελεσθεί, τότε ολόκληρη η ΛΔ επιστρέφει στην πριν την εκτέλεση αυτού κατάσταση.
- Το γεγονός στο οποίο αναφέρεται δεν έχει παραληφθεί (συμβαίνει αν το υποκείμενο σύστημα επικοινωνίας δε διασφαλίζει μεταφορά FCFS), οπότε το αντι-γεγονός αποθηκεύεται σε ένα από τα κανάλια εισόδου της ΛΔ και περιμένει την άφιξή του.

Κατά την εφαρμογή της βέλτιστης προσέγγισης, κάθε ΛΔ πρέπει να διατηρεί επαρκείς πληροφορίες, όπως αποθήκες παρελθόντων καταστάσεων (past state buffers) και εξωτερικών γεγονότων αλλά και αντι-γεγονότων. Το γεγονός αυτό όμως καθιστά πιθανή την εμφάνιση καταστάσεων έλλειψης μνήμης, που μπορούν να οδηγήσουν σε απρόσμενα αργή εκτέλεση της προσομοίωσης. Έτσι, η εφαρμογή της συγκεκριμένης προσέγγισης συνοδεύεται συνήθως από τη χρήση κάποιου αλγορίθμου διαχείρισης διαθέσιμης μνήμης, που εγγυάται την επάρκεια αυτής, καθ΄ όλη τη διάρκεια του πειράματος προσομοίωσης.

Είναι φανερό ότι η επιτάχυνση, που μπορεί να επιτευχθεί μέσω μιας διάταξης παράλληλης/κατανεμημένης προσομοίωσης, εξαρτάται από το βαθμό παραλληλίας του προς ανάλυση μοντέλου. Στη [NH96] οι συγγραφείς αναφέρουν ότι για την επίτευξη σημαντικής επιτάχυνσης είναι απαραίτητο το μοντέλο να:

- είναι αρκετά μεγάλο,
- έχει υψηλό λόγο υπολογισμών/επικοινωνίας (μερικές δεκάδες γεγονότων ανά πακέτο),
- μη δημιουργεί υπερβολικά μεγάλο αριθμό γεγονότων συγχρονισμού και
- να έχει καλά ισορροπημένο φόρτο επεξεργασίας.

Ένας άλλος μηχανισμός εκτέλεσης, που σαφώς δεν υπόκειται στους παραπάνω περιορισμούς είναι αυτός της εκτέλεσης σε *παράλληλο χρόνο* ή σε *παράλληλα ρεύματα χρόνου*. Στην πρώτη περίπτωση η εκτέλεση διασπάται σε ΛΔ, κάθε μία από τις οποίες αντιστοιχεί σε διαδοχικές και μη επικαλυπτόμενες χρονικές περιόδους. Για την υλοποίηση της τεχνικής αυτής, απαραίτητη προϋπόθεση είναι η ταύτιση της κατάστασης του μοντέλου στα όρια αυτών των χρονικών περιόδων, κάτι που, όπως θα δειχθεί στην παράγραφο 4.7, δεν είναι καθόλου δύσκολο να επιτευχθεί. Στη δεύτερη περίπτωση έχουμε πολλαπλές επαναλήψεις του ιδίου

πειράματος σε παράλληλες ροές χρόνου από τη χρονική στιγμή 0. Και οι δύο τεχνικές διαθέτουν ιδιαίτερα χαμηλό (έως και μηδενικό) κόστος συγχρονισμού και επικοινωνίας μεταξύ των ΛΔ και είναι ιδιαίτερα αποδοτικές στη στατιστική επεξεργασία των αποτελεσμάτων πειραμάτων προσομοίωσης.

Η ανάπτυξη διατάξεων παράλληλης/κατανεμημένης προσομοίωσης προϋποθέτει μία συστηματική ενασχόληση με θέματα συγχρονισμού και διαχείρισης της επικοινωνίας μεταξύ των ΛΔ, που δεν έχει σε τίποτε να κάνει με τα βασικά προβλήματα της αποτίμησης της απόδοσης ενός συστήματος. Η χρήση εξειδικευμένων γλωσσών ή βιβλιοθηκών συμβάλλει στον περιορισμό του κόστους ανάπτυξης τέτοιων διατάξεων, διαθέτοντας στον αναλυτή προκατασκευασμένους πυρήνες παράλληλης/κατανεμημένης προσομοίωσης και ευκολίες ανάπτυξης εφαρμογών. Γνωστές γλώσσες είναι η Parsec [BMT98] και η YADDES [PRE89], ενώ η πιο γνωστή ίσως από τις βιβλιοθήκες, που χρησιμοποιούνται για το σκοπό αυτό, είναι η Parasol.

Τέλος, συχνά είναι επιθυμητή η εύκολη και γρήγορη παραλληλοποίηση λογισμικού προσομοίωσης σειριακής εκτέλεσης. Στο επίπεδο αυτό υπάρχουν σημαντικές δυνατότητες μόνον όσον αφορά τις προσομοιώσεις παράλληλου χρόνου και αυτό εξαιτίας της εμφάνισης εύχρηστων και πρακτικών βιβλιοθηκών προγραμματισμού παράλληλων εφαρμογών (APIs). Τέτοιες είναι όλες εκείνες, που συμμορφώνονται στις προδιαγραφές OpenMP [OPE97] για προγραμματισμό σε περιβάλλον διαμοιράσιμης μνήμης, καθώς και η επονομαζόμενη BSP, που μπορεί να χρησιμοποιηθεί σε σύνολο σταθμών εργασίας διασυνδεδεμένων με χαμηλού επιπέδου δυνατότητες MPI (Message Passing Interface) επικοινωνίας. Ένας OpenMP μεταφραστής χρησιμοποιήθηκε και στην [KL01a], της οποίας τα συμπεράσματα παρουσιάζονται στην παράγραφο 4.8.

4.3 Γεννήτριες τυχαίων αριθμών

Ένα από τα θεμελιώδη προβλήματα, που πρέπει να αντιμετωπιστούν κατά την ανάπτυξη και εκτέλεση πειραμάτων προσομοίωσης διακριτών γεγονότων, είναι η επιλογή και η αποδοτική υλοποίηση του κατάλληλου αλγορίθμου παραγωγής τυχαίων αριθμών. Βέβαια, καθώς μέσω ενός αλγορίθμου είναι εφικτή η πρόγνωση της αλληλουχίας των «τυχαίων» αριθμών με πιθανότητα 1, είναι σαφές ότι στην πραγματικότητα αναφερόμαστε σε αλγορίθμους παραγωγής ψευδο-τυχαίων αριθμών. Αριθμών δηλαδή που είναι τυχαίοι υπό την έννοια ότι θα μπορούσαν να περάσουν τα σχετικά στατιστικά τεστ τυχαιότητας.

Στις περισσότερες περιπτώσεις, η παραγωγή τέτοιων αριθμών επιτυγχάνεται μέσω μιας αναδρομικής σχέσης που καθορίζει τον επόμενο τυχαίο αριθμό, ως συνάρτηση του προηγούμενου ή των προηγούμενων αριθμών. Η τιμή που χρησιμοποιείται για την αρχικοποίηση της αλληλουχίας αυτής των αριθμών ονομάζεται *σπόρος* (τιμή εκκίνησης).

Ένας πρώτος ικανοποιητικός αλγόριθμος για την παραγωγή ψευδο-τυχαίων αριθμών προτάθηκε από τον D. H. Lehmer το 1951 και βασίζεται στην αναδρομική σχέση:

$$x_n = a \cdot x_{n-1} \cdot \mathrm{mod}\, m \qquad (1)$$

Οι παράμετροι *a* και *m* καλούνται αντίστοιχα *πολλαπλασιαστής* και *υπόλοιπο*. Πολλές από τις χρησιμοποιούμενες σήμερα γεννήτριες τυχαίων αριθμών βασίζονται σε μια γενίκευση της (1), της μορφής:

$$x_n = (a \cdot x_{n-1} + b) \cdot \mathrm{mod}\, m \qquad (2)$$

και αποτελούν την κατηγορία εκείνη των αλγορίθμων, που είναι γνωστή ως *linear-congruential γεννήτριες* τυχαίων αριθμών. Μια χρήσιμη ανασκόπηση των σημαντικότερων εξελίξεων της ομάδας αυτής των αλγορίθμων μπορεί κανείς να βρει στη [PM98].

Αν στην (2) θεωρήσουμε a=5, b=1, m=16 και σπόρο 5, τότε η αλληλουχία αριθμών, που προκύπτει, είναι η:

.....10, 3, 0, 1, 6, 15, 12, 13, 2, 11, 8, 9, 14, 7, 4, 5, 10, 3, 0, ……

Είναι σαφές ότι μόνο οι 16 πρώτοι αριθμοί είναι μοναδικοί, ενώ από τον 17ο και πέρα επαναλαμβάνονται κυκλικά. Στην περίπτωση αυτή, λέμε ότι η συγκεκριμένη γεννήτρια τυχαίων αριθμών έχει *μήκος κύκλου* 16. Μερικές γεννήτριες δεν επαναλαμβάνουν ένα αρχικό μέρος της αλληλουχίας των αριθμών και το μέρος αυτό ονομάζεται *ουρά*. Ως *περίοδο* μιας γεννήτριας τυχαίων αριθμών χαρακτηρίζουμε το άθροισμα του αριθμού των στοιχείων της ουράς και αυτού των στοιχείων του κύκλου.

Γενικά, η περίοδος καθορίζεται από την επιλογή των a, b και m. Στη [SCH79a] θεμελιώθηκε ότι για $m=2^{31}-1$ έχουμε μεταφέρσιμη υλοποίηση του αλγορίθμου για όλες τις μηχανές, που μπορούν να αναπαραστήσουν ακεραίους στο διάστημα από -2^{31} έως $2^{31}-1$. Επιπλέον, στην [LGM69] είχε προταθεί ο συντελεστής a=16807, γιατί έτσι επιτυγχάνεται η μέγιστη περίοδος της γεννήτριας. Οι παράμετροι αυτοί δοκιμάστηκαν με επιτυχία σε πολλά στατιστικά τεστ τυχαιότητας και ο συγκεκριμένος αλγόριθμος παραμένει σε χρήση μέχρι και σήμερα.

Παρόλα αυτά, η υλοποίηση διατάξεων παράλληλης/κατανεμημένης προσομοίωσης προβάλλει νέες απαιτήσεις [KL01b] για την επιλογή της κατάλληλης γεννήτριας τυχαίων αριθμών. Οι τεχνικές που χρησιμοποιούνται αποβλέπουν στην παραγωγή μη συσχετισμένων τυχαίων αριθμών στις παράλληλες διεργασίες μιας τέτοιας διάταξης. Αυτό πρακτικά σημαίνει ότι έχοντας γνωστό ένα αρχικό τμήμα της αλληλουχίας των αριθμών σε μία διεργασία, αλλά και ολόκληρες τις αλληλουχίες των αριθμών, που παράγονται στις άλλες διεργασίες, δεν πρέπει να είναι εφικτή η πρόγνωση των επόμενων αριθμών της πρώτης διεργασίας. Γενικά, είναι αλήθεια ότι οι δια-διεργασιακές συσχετίσεις είναι λιγότερο σημαντικές από τις ενδο-διεργασιακές συσχετίσεις, αλλά σίγουρα χρειάζεται η ανάπτυξη νέων στατιστικών τεστ τυχαιότητας, που θα λαμβάνουν υπόψη αυτού του είδους τα προβλήματα.

Οι πιο διαδεδομένες τεχνικές παραγωγής τυχαίων αριθμών για παράλληλη/κατανεμημένη προσομοίωση είναι:

- Η κεντρική παραγωγή, όπου κάποια συγκεκριμένη διεργασία παίζει το ρόλο μιας κεντρικής γεννήτριας τυχαίων αριθμών. Η διάθεση των τυχαίων αριθμών γίνεται με ανταλλαγή μηνυμάτων. Αυτό έχει ως συνέπεια αυξημένο κόστος επικοινωνίας και περιορισμό των περιθωρίων επίτευξης σημαντικής επιτάχυνσης, λόγω της ανάγκης για αποκλειστική πρόσβαση της κάθε διεργασίας σε αυτήν, που διαθέτει τη γεννήτρια τυχαίων αριθμών.
- Η διαίρεση του κύκλου της γεννήτριας, που έχει ως κύριο χαρακτηριστικό τη χρήση της ίδιας επαναληπτικής διαδικασίας σε κάθε μία από τις διεργασίες χωριστά. Εμφανίζεται σε τρεις διαφορετικές παραλλαγές:
  - Αυτή κατά την οποία η διαίρεση του κύκλου επιτυγχάνεται με τη χρήση κατάλληλα επιλεγμένων διαφορετικών τιμών σπόρου σε κάθε διεργασία.
  - Τη γνωστή ως *άλμα του βατράχου*, κατά την οποία η διεργασία i επιλέγει τους αριθμούς x_i, x_{i+M}, ……, όπου M ο αριθμός των διεργασιών της προσομοίωσης.

- Τη διάσπαση του κύκλου της γεννήτριας, κατά την οποία η i+1 διεργασία επιλέγει τους αριθμούς $x_{iL/M}, x_{(iL/M)+1}, \ldots\ldots$, όπου L είναι το μήκος του κύκλου της γεννήτριας.

Είναι σαφές ότι τόσο η δεύτερη όσο και η τρίτη παραλλαγή της τεχνικής της διαίρεσης του κύκλου της γεννήτριας απαιτούν τη στιγμιαία προώθηση του αλγορίθμου κατά M και κατά L/M βήματα αντίστοιχα, σε κάθε επανάληψη αυτού. Επιπλέον, τα στατιστικά τεστ τυχαιότητας, στα οποία υπόκειται η αρχική αλληλουχία αριθμών, δεν είναι απαραίτητα επαρκή και για αυτές που προκύπτουν από τη διαίρεση του κύκλου αυτής.

Συνοπτικά, τα χαρακτηριστικά, που μία γεννήτρια τυχαίων αριθμών είναι απαραίτητο να διαθέτει για χρήση σε διατάξεις παράλληλης/κατανεμημένης προσομοίωσης, είναι:

- *Τυχαιότητα*: Αν η αλληλουχία τυχαίων αριθμών αποτελείται από ακεραίους στο διάστημα [1, d], τότε η πιθανότητα λήψης οποιουδήποτε από αυτούς θα πρέπει να είναι 1/d. Παρόλα αυτά είναι επιθυμητή η διασφάλιση της ομοιομορφίας σε μεγαλύτερες διαστάσεις και αυτό ορίζεται ως εξής:
  Ας θεωρήσουμε τις διατεταγμένες n-άδες διαδοχικών τυχαίων αριθμών,

  $$X_i^n = (x_{i+1}, \ldots\ldots, x_{i+n})$$

  οι οποίες υποδιαιρούν το μοναδιαίο n-διάστατο υπερκύβο σε πολλά ίσα τμήματα. Μία αλληλουχία τυχαίων αριθμών είναι ομοιόμορφη, αν, όταν ο αριθμός αυτών τείνει στο άπειρο, τότε τα τμήματα, που ορίζονται από τις διαφορετικές n-άδες τυχαίων αριθμών, έχουν τις ίδιες συχνότητες εμφάνισης. Καθώς λοιπόν η όποια συσχέτιση μεταξύ διαδοχικών n-άδων μπορεί να οδηγήσει σε λανθασμένα αποτελέσματα, είναι συνήθως επιθυμητή η ύπαρξη ομοιομορφίας σε διαστάσεις μεγαλύτερες του 4.
- *Ταχύτητα*: Μία γεννήτρια τυχαίων αριθμών είναι πάντα επιθυμητό να παράγει αριθμούς με μεγάλη ταχύτητα ακόμη και όταν η παραγωγή αυτών αποτελεί ένα μικρό μόνο μέρος του χρόνου επεξεργασίας του μοντέλου. Αυτό πρακτικά σημαίνει ότι κατά την υλοποίηση του αλγορίθμου της γεννήτριας θα πρέπει να αποφεύγεται η άμεση χρήση υπολογιστικά ακριβών πράξεων, όπως για παράδειγμα αυτή του υπολοίπου, που χρησιμοποιείται στις linear-congruential γεννήτριες.
- *Μεγάλος κύκλος*: Από αυτά που ήδη εκτέθηκαν, είναι σαφές ότι μία γεννήτρια τυχαίων αριθμών με όχι επαρκώς μεγάλο κύκλο είναι επιρρεπής στη δημιουργία τόσο δια-διεργασιακών όσο και ενδο-διεργασιακών συχετίσεων στα ρεύματα τυχαίων αριθμών παράλληλων και κατανεμημένων διατάξεων προσομοίωσης.

Οι τελευταίες εξελίξεις ([MK92] και [MK94]) στις γεννήτριες του τύπου *Twisted Generalized Feedback Shift Register* (TGFSR) τις καθιστούν μία ελκυστική επιλογή για χρήση σε μηχανισμούς εκτέλεσης παράλληλων/κατανεμημένων προσομοιώσεων. Οι λόγοι είναι: α) το γεγονός ότι η παραγωγή αριθμών είναι ταχύτατη, αφού για αυτή χρειάζονται μόνο ένας μικρός αριθμός αναφορών μνήμης και μόνο μία χρήση αποκλειστικού OR και β) το γεγονός ότι η αλληλουχία των αριθμών, που παράγουν, έχει μεγάλο μέγεθος κύκλου, που σε καμία περίπτωση δεν περιορίζεται από τις προδιαγραφές κατασκευής της μηχανής εκτέλεσης (32-bit ή 64-bit μέγεθος λέξης).

Η πιο χαρακτηριστική ίσως περίπτωση είναι αυτή του επονομαζόμενου *αλγορίθμου Mersenne Twister* ([MN98]), ο οποίος παράγει τυχαίους αριθμούς, ομοιόμορφα κατανεμημένους σε διάσταση 623 και περίπου στην ίδια ταχύτητα με αυτήν της κλασσικής γεννήτριας τυχαίων

αριθμών rand της C, με τον απίστευτο όμως κύκλο 2^{19937}-1. Ο αλγόριθμος αυτός χρησιμοποιήθηκε με πολύ μεγάλη επιτυχία στην [KL01a], τα συμπεράσματα της οποίας παρουσιάζονται στην παράγραφο 4.8.

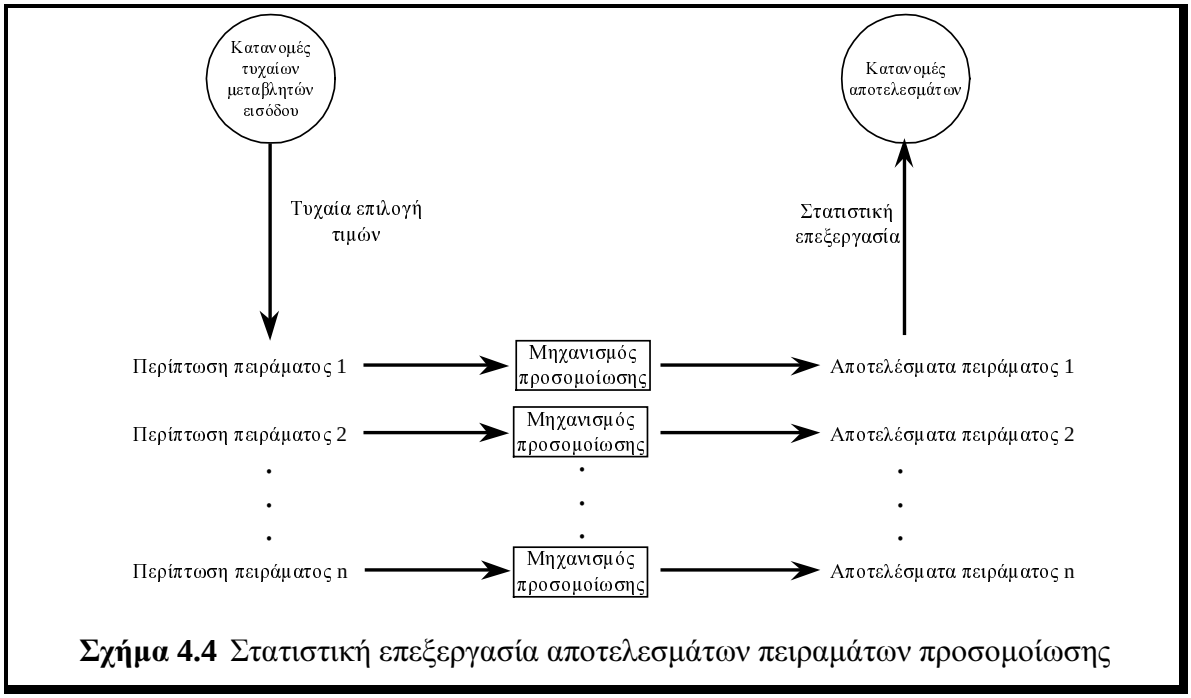
4.4 Στατιστική επεξεργασία αποτελεσμάτων προσομοίωσης

Όπως και στην παράγραφο 3.1, έτσι και στο κεφάλαιο αυτό, το ενδιαφέρον εστιάζεται στην ανάλυση προσομοιωτικών μοντέλων δικτύων ουρών όταν αυτά βρίσκονται σε κατάσταση στατιστικής ισορροπίας. Αυτό δηλαδή που ενδιαφέρει είναι η συμπεριφορά του συστήματος μετά την πάροδο χρονικού διαστήματος αρκετά μεγάλου, ώστε η επιλογή αρχικοποίησης αυτού να μην παίζει σημαντικό ρόλο στη διαμόρφωση των μέτρων απόδοσης που μελετώνται.

Αυτού του τύπου τα πειράματα προσομοίωσης είναι και τα πιο συχνά χρησιμοποιούμενα. Παρόλα αυτά, είναι σαφές ότι στις περισσότερες περιπτώσεις αγνοείται το γεγονός ότι ένα πείραμα προσομοίωσης είναι βασικά ένα τυχαίο πείραμα, τα αποτελέσματα του οποίου σε καμία περίπτωση δεν είναι αξιόπιστα, αν πρώτα δεν υποστούν την κατάλληλη στατιστική επεξεργασία.

> *If the random nature of the simulation results is ignored, instead of an expensive simulation model, a toss of the coin had better be used.*
>
> **-Jack P. C. Kleijnen, "The role of statistical methodology in simulation", In Methodology in Systems Modelling and Simulation, North-Holland, Amsterdam, 1979**

Στο σχήμα 4.4 απεικονίζεται μία γραφική αναπαράσταση συλλογής και στατιστικής επεξεργασίας *n* παρατηρήσεων πειραμάτων προσομοίωσης. Η επεξεργασία αυτή είναι γνωστή ως *τεχνική των ανεξάρτητων επαναλήψεων* και είναι ίσως η πιο συχνά χρησιμοποιούμενη αν και όχι η πιο αποδοτική μέθοδος επεξεργασίας των αποτελεσμάτων πειραμάτων προσομοίωσης.



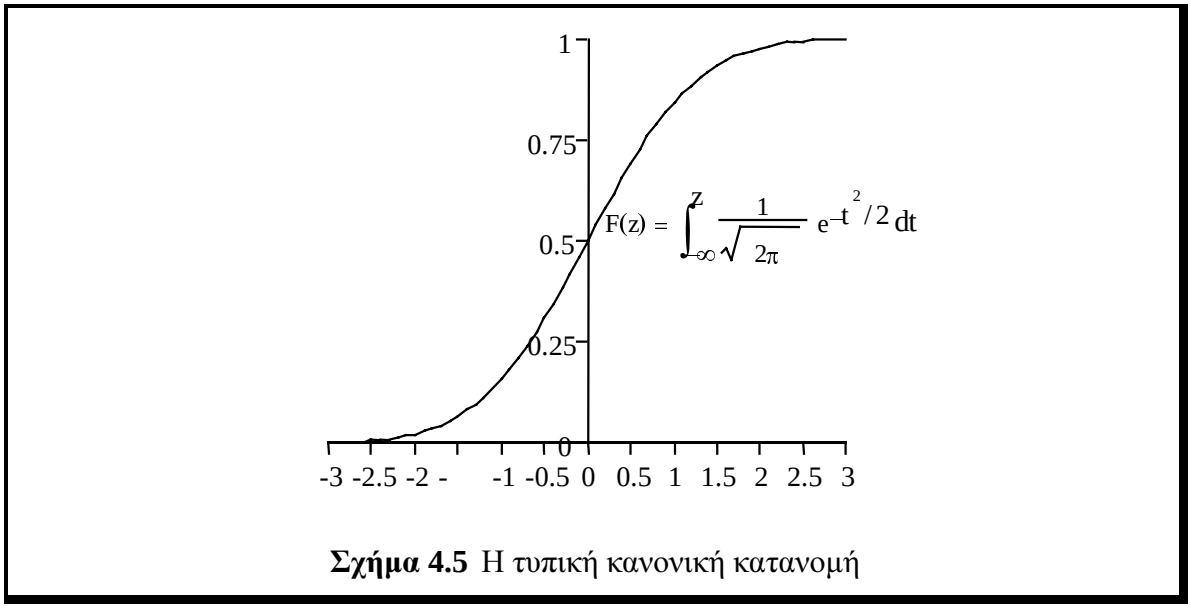
Σχήμα 4.4 Στατιστική επεξεργασία αποτελεσμάτων πειραμάτων προσομοίωσης

Αν λοιπόν θεωρήσουμε ότι ο σκοπός μιας προσομοιωτικής ανάλυσης είναι ο υπολογισμός της μέσης τιμής ενός συγκεκριμένου μέτρου απόδοσης από μια σειρά παρατηρήσεων $x_1, x_2, ..., x_n$, τότε η

$$\overline{X}(n) = \sum_{i=1}^{n} \frac{x_i}{n} \tag{3}$$

είναι απλά ένας εκτιμητής του ζητούμενου μεγέθους με βάση το δείγμα των *n* παρατηρήσεων που χρησιμοποιήθηκαν. Βέβαια, σύμφωνα με τους στατιστικούς νόμους των μεγάλων αριθμών, όσο μεγαλύτερο είναι το μέγεθος του δείγματος, τόσο πιο ακριβής είναι η εκτίμηση. Πώς όμως μπορούμε να αποφασίσουμε για τον αριθμό των παρατηρήσεων που χρειάζονται για τον υπολογισμό του ζητούμενου μεγέθους με κάποια δεδομένη ακρίβεια, και πώς περιγράφουμε την απαιτούμενη ακρίβεια; Έχει επικρατήσει η ακρίβεια των υπολογισμών μιας προσομοιωτικής ανάλυσης να προδιαγράφεται με τη χρήση *διαστημάτων εμπιστοσύνης*.

Αν λοιπόν μία τυχαία μεταβλητή *X* είναι κανονικά κατανεμημένη με μέση τιμή μ και τυπική απόκλιση σ, τότε είναι δυνατό να αποδειχθεί ότι η μεταβλητή *Z* με τιμές $Z = (X - \mu)/\sigma$ είναι επίσης κανονικά κατανεμημένη με μέση τιμή 0 και τυπική απόκλιση 1. Η συγκεκριμένη κατανομή εικονίζεται στο σχήμα 4.5 και είναι η *τυπική κανονική κατανομή*.



Σχήμα 4.5 Η τυπική κανονική κατανομή

Είναι λοιπόν σαφές ότι,

$$P\{Z \le F^{-1}(a)\} = a, \text{ για } 0 \le a \le 1 \tag{4}$$

όπου F^{-1} είναι η αντίστροφη της κατανομής F. Αν λάβουμε υπόψη τη συμμετρία της τυπικής κανονικής κατανομής όπως φαίνεται και στο σχήμα 4.5, τότε η (4) δίνει την:

$$P\{0 \le Z \le F^{-1}(a)\} = a - 0.5, \text{ για } 0.5 \le a \le 1$$

και συνακόλουθα την:

$$P\{-F^{-1}(a) \le Z \le F^{-1}(a)\} = 2 \cdot a - 1, \text{ για } 0 \le a \le 1 \text{ ή αλλιώς,}$$

$$P\left\{-F^{-1}\left(\frac{1+a}{2}\right) \le Z \le F^{-1}\left(\frac{1+a}{2}\right)\right\} = a, \text{ για } 0 \le a \le 1 \tag{5}$$

Οι τιμές της αντίστροφης τυπικής κανονικής κατανομής λαμβάνονται συνήθως από σχετικούς πίνακες. Έτσι, αν α=0.9, τότε $F^{-1}(0.95)$=1.645 και λέμε ότι το 90% των τιμών της Z περιέχονται στο διάστημα [-1.645, 1.645].

Σύμφωνα με ένα από τα κεντρικά οριακά θεωρήματα, αν οι $x_1, x_2, ..., x_n$ είναι ανεξάρτητες μεταξύ τους και έχουν την ίδια κατανομή με πεπερασμένη μέση τιμή και διασπορά, τότε η κατανομή του αθροίσματός τους τείνει στην κανονική, για αρκούντως μεγάλα n. Καθώς λοιπόν οι $x_1, x_2, ..., x_n$ προέρχονται από ανεξάρτητες επαναλήψεις, είναι ανεξάρτητες μεταξύ τους με μέση τιμή μ - το ζητούμενο μέγεθος - και διασπορά σ^2· οι μεταβλητές $x_1/n, x_2/n, ..., x_n/n$ είναι και αυτές ανεξάρτητες μεταξύ τους και προέρχονται από την ίδια κατανομή με μέση τιμή μ/n και διασπορά $(\sigma/n)^2$. Ο δειγματικός μέσος $\overline{X}(n)$ της (3) μπορεί να θεωρηθεί ως το άθροισμα των $x_1/n, x_2/n, ..., x_n/n$ με μέση τιμή $n(\mu/n)=\mu$ και διασπορά $n(\sigma/n)^2=\sigma^2/n$.

Σύμφωνα λοιπόν με το κεντρικό οριακό θεώρημα, αν το n είναι αρκετά μεγάλο, τότε η $\overline{X}(n)$ ακολουθεί την κανονική κατανομή και κατά συνέπεια η $(\overline{X}(n)-\mu)\cdot\sqrt{n}/\sigma$ ακολουθεί την τυπική κανονική κατανομή, οπότε ισχύει για αυτήν η (5)· δηλαδή,

$$P\left\{-F^{-1}\left(\frac{1+a}{2}\right) \le \frac{(\overline{X}(n)-\mu)\cdot\sqrt{n}}{\sigma} \le F^{-1}\left(\frac{1+a}{2}\right)\right\} = a \text{, για } 0 \le a \le 1$$

και τελικά,

$$P\left\{\overline{X}(n) - \frac{F^{-1}\left(\frac{1+a}{2}\right)\cdot\sigma}{\sqrt{n}} \le \mu \le \overline{X}(n) + \frac{F^{-1}\left(\frac{1+a}{2}\right)\cdot\sigma}{\sqrt{n}}\right\} = a \text{, για } 0 \le a \le 1 \qquad (6)$$

Η (6) πρακτικά σημαίνει ότι αν εκτελέσουμε μεγάλο αριθμό πειραμάτων προσομοίωσης, τότε αναμένουμε ένα $(100\cdot a)\%$ των διαστημάτων που ορίζει η (6), να περιέχουν την πραγματική μέση τιμή μ. Βέβαια, καθώς η σ^2 δεν είναι γνωστή, μπορούμε για αρκετά μεγάλα n να την προσεγγίσουμε με τη δειγματική της τιμή,

$$s^2(n) = \frac{1}{n-1}\cdot\sum_{i=1}^{n}(x_i - \overline{X}(n))^2 = \frac{1}{n-1}\cdot\left(\sum_{i=1}^{n}x_i^2 - n\cdot\overline{X}^2(n)\right) \qquad (7)$$

Αν ο σκοπός μας είναι η παραγωγή ενός 100α% διαστήματος εμπιστοσύνης για το μ με μήκος όχι μεγαλύτερο από το $(100\cdot 2\delta)\%$ του μ, τότε ο αριθμός των παρατηρήσεων που θα χρειαστούν, δίνεται από τη σχέση,

$$n \ge \left(\frac{F^{-1}\left(\frac{1+a}{2}\right)}{\delta}\right)^2 \cdot \left(\frac{s(n)}{\overline{X}(n)}\right)^2 \qquad (8)$$

Πάντως ο εκτιμητής της (3) είναι σαφώς επιβαρημένος με την επίδραση της αυθαίρετης κατά ανάγκη αρχικοποίησης του συστήματος στη συμπεριφορά αυτού. Επίδραση, η οποία μπορεί βέβαια να είναι μικρή όταν τα $x_1, x_2, ..., x_n$ είναι οι μέσες τιμές ενός αρκετά μεγάλου αριθμού παρατηρήσεων ανά περίπτωση, δεν παύει όμως να είναι υπαρκτή και να επηρεάζει την ποιότητα του εκτιμητή της (3). Γενικά, η υλοποίηση μεθόδου για τον εντοπισμό και την

απομάκρυνση των παρατηρήσεων, που βρίσκονται εντός των ορίων της αρχικής μεταβατικής περιόδου, είναι αρκετά περίπλοκη. Έτσι, προτιμάται συνήθως η εκτέλεση μεγάλου μήκους πειραμάτων προσομοίωσης, ώστε να περιορίζεται δραστικά η επίδραση της αρχικής μεταβατικής περιόδου στην ποιότητα των εκτιμήσεων.

Η τεχνική όμως, των ανεξάρτητων επαναλήψεων δεν είναι η μόνη που εγείρει θέματα ποιότητας των εκτιμήσεων. Μία πλειάδα άλλων τεχνικών που θα παρουσιασθούν στην παράγραφο 4.6, βασίζεται στην εκτίμηση ενός συνόλου παρατηρήσεων προερχόμενων από ένα μόνο πείραμα προσομοίωσης. Παρατηρήσεων όμως, που εκτός από την επίδραση της αρχικής μεταβατικής περιόδου, στις περισσότερες περιπτώσεις έχουν και το πρόβλημα της μη ανεξαρτησίας.

Τα μεγέθη που συνήθως χρησιμοποιούνται για τη μέτρηση της ποιότητας κάποιου εκτιμητή, είναι τρία:

- Η *μεροληψία*, η οποία μετρά τη συστηματική απόκλιση ενός εκτιμητή από την πραγματική τιμή και δίνεται από τη σχέση,

$$\text{Bias}[\overline{X}(n)] = E[\overline{X}(n) - \mu] \tag{9}$$

- Η *διασπορά* του εκτιμητή, που μετρά τη μέση τετραγωνική απόκλιση του εκτιμητή από τη μέση του τιμή και ορίζεται ως,

$$\sigma^2[\overline{X}(n)] = E[\{\overline{X}(n) - E[\overline{X}(n)]\}^2] \tag{10}$$

- Το *μέσο τετραγωνικό σφάλμα* του εκτιμητή, που ορίζεται ως

$$MSE[\overline{X}(n)] = E\{[\overline{X}(n) - \mu]^2\} \tag{11}$$

Από τις (9) και (10) εύκολα συνάγεται η,

$$MSE[\overline{X}(n)] = \{Bias[\overline{X}(n)]\}^2 + \sigma^2[\overline{X}(n)] \tag{12}$$

Επίσης, είναι προφανές ότι η μεροληψία και το μέσο τετραγωνικό σφάλμα μπορούν να μελετηθούν μόνο, όταν η μέση τιμή μ είναι εκ των προτέρων γνωστή από την αναλυτική επίλυση του μοντέλου (αν αυτή είναι εφικτή).

Όσον αφορά τώρα την ποιότητα της τεχνικής των ανεξάρτητων επαναλήψεων, αυτή έχει μελετηθεί κυρίως, σε σχέση με τη μετρήσιμη μεροληψία και το μέσο τετραγωνικό σφάλμα εξαιτίας της παρουσίας στο δείγμα των παρατηρήσεων της αρχικής μεταβατικής περιόδου. Στο [FIS78a] προτείνεται η χρήση τουλάχιστο 100 παρατηρήσεων σε κάθε επανάληψη, έτσι ώστε να διασφαλίζεται η κανονικότητα των υπολογιζόμενων μέσων τιμών ανά επανάληψη. Επιπλέον, αποτελέσματα που δημοσιεύονται στην [KL84] δείχνουν ότι είναι προτιμητέα η χρήση επαναλήψεων μεγαλύτερης διάρκειας, παρά η χρήση μεγάλου αριθμού αυτών.

4.5 Τεχνικές περιορισμού διασποράς

Στην παράγραφο αυτή επικεντρωνόμαστε σε μία συνοπτική παρουσίαση κάποιων από τις τεχνικές προσομοιωτικής ανάλυσης υψηλής ακρίβειας και μειωμένου υπολογιστικού κόστους. Οι τεχνικές αυτές είναι γνωστές ως *τεχνικές περιορισμού διασποράς* (Variance Reduction Techniques) και η δυνατότητα εφαρμογής της κάθε μιας από αυτές εξαρτάται, τόσο από τη δομή του μοντέλου προσομοίωσης, όσο και από την τεχνική στατιστικής επεξεργασίας των αποτελεσμάτων που χρησιμοποιείται.

Από τη σχέση (8), είναι ήδη σαφές ότι ο αριθμός των παρατηρήσεων που χρειάζονται για την εκτίμηση ενός μέτρου απόδοσης του μοντέλου προσομοίωσης, είναι ευθέως ανάλογος της

διασποράς των παρατηρήσεων. Οι τεχνικές περιορισμού της διασποράς λοιπόν, αποβλέπουν στην ταχύτερη μείωση αυτής, έτσι ώστε να περιορίζεται δραστικά το υπολογιστικό κόστος μιας προσομοιωτικής ανάλυσης, διατηρώντας παράλληλα ένα επιθυμητό επίπεδο ακρίβειας των αποτελεσμάτων.

Μειώσεις της διασποράς γενικά μπορούν να επιτευχθούν με:

- την παραγωγή παρατηρήσεων, οι οποίες δεν είναι ανεξάρτητες, μέσω της χρήσης κοινών τυχαίων αριθμών ή μέσω μετασχηματισμών κάποιων μεταβλητών εισόδου,
- τη χρήση πληροφορίας για συναφείς μεταβλητές εξόδου και εισόδου (όπως για παράδειγμα αποκρίσεις παρομοίων συστημάτων) με γνωστές στοχαστικές ιδιότητες,
- τη χρήση πληροφορίας γνωστής από τη θεωρία (όπως π.χ. το νόμο του Little) για τυχαίες μεταβλητές εισόδου και εξόδου και
- τη χρήση προωθημένων τεχνικών δειγματοληψίας.

Στη συνέχεια, θα γίνει μία συνοπτική παρουσίαση των πιο γνωστών από τις τεχνικές περιορισμού διασποράς, που έχουν κατά καιρούς προταθεί. Μία πιο πλήρη και περισσότερο κατατοπιστική περιγραφή αυτών μπορεί κανείς να βρει στο [LO89].

4.5.1 Αντιθετική δειγματοληψία (Antithetic sampling)

Η αντιθετική δειγματοληψία είναι μία τεχνική περιορισμού διασποράς, που βασίζεται στην παράλληλη εκτέλεση δύο πειραμάτων προσομοίωσης του ίδιου μοντέλου με συμπληρωματικές αλληλουχίες τυχαίων αριθμών. Οι δύο αυτές αλληλουχίες τυχαίων αριθμών έχουν ακριβώς την αντίθετη επίδραση στο μέτρο απόδοσης που ενδιαφέρει. Αν τα αποτελέσματα των δύο προσομοιώσεων είναι αρνητικά συσχετισμένα μεταξύ τους, τότε η μέση τιμή αυτών είναι ένας εκτιμητής με μικρότερη διασπορά από ότι ο κάθε ένας από αυτούς χωριστά.

Σύμφωνα με το [LK82], είναι σημαντικό οι δύο προσομοιώσεις να παραμένουν συγχρονισμένες καθ΄ όλη τη διάρκεια της εκτέλεσής τους. Αυτό σημαίνει ότι αν για παράδειγμα ένας συγκεκριμένος τυχαίος αριθμός, χρησιμοποιείται για τον υπολογισμό κάποιου χρόνου εξυπηρέτησης σε ένα δίκτυο ουρών, τότε ο συμπληρωματικός του πρέπει να χρησιμοποιηθεί για τον υπολογισμό του αντίστοιχου χρόνου εξυπηρέτησης στο δεύτερο μοντέλο προσομοίωσης. Η έλλειψη συγχρονισμού μπορεί να οδηγήσει όχι μόνο στο μη περιορισμό της διασποράς του εκτιμητή, αλλά αντίθετα στην αύξηση αυτής.

Επιπλέον, σύμφωνα με την [FIS83] σε μεγάλα δίκτυα ουρών, η πολυπλοκότητα των αλληλεπιδράσεων που παρατηρείται μεταξύ των ουρών συχνά καθιστά αδύνατη την επίτευξη σημαντικής συσχέτισης μεταξύ των μέτρων απόδοσης των δύο συμπληρωματικών προσομοιώσεων. Στις περιπτώσεις αυτές, η αποτελεσματικότητα της τεχνικής είναι αμφισβητήσιμη.

4.5.2 Κοινοί τυχαίοι αριθμοί

Η τεχνική των κοινών τυχαίων αριθμών είναι μία τεχνική περιορισμού διασποράς, με πολλά κοινά σημεία με αυτήν της αντιθετικής δειγματοληψίας. Εδώ, η τυχαία μεταβλητότητα της διαφοράς στην απόδοση δύο μοντέλων, που αντιπροσωπεύουν εναλλακτικές σχεδιάσεις του αυτού συστήματος, ελαχιστοποιείται μέσω της χρήσης της ίδιας αλληλουχίας τυχαίων αριθμών. Έτσι επιτυγχάνεται η σύγκριση συστημάτων κάτω από τις ίδιες συνθήκες εκτέλεσης πειράματος.

Στα προτερήματα της τεχνικής αυτής, συγκαταλέγονται η σχετική ευκολία εφαρμογής της και ο πιθανός περιορισμός του υπολογιστικού κόστους μιας τέτοιας ανάλυσης. Ως μειονεκτήματα, θα μπορούσαμε να αναφέρουμε την ανάγκη συγχρονισμού στις τυχαίες μεταβλητές εισόδου των μοντέλων και την πιθανότητα μη αποτελεσματικότητας της τεχνικής, όταν αυτή εφαρμόζεται σε πολύπλοκα συστήματα με μη σημαντική συσχέτιση εισόδου - εξόδου.

4.5.3 Μεταβλητές ελέγχου (Control variates)

Μία μεταβλητή ελέγχου είναι μία τυχαία μεταβλητή συσχετισμένη με το μέτρο απόδοσης, που ενδιαφέρει, με γνωστή μέση τιμή. Ας υποθέσουμε για παράδειγμα ότι η τυχαία μεταβλητή X εκφράζει το μέτρο απόδοσης που θέλουμε να υπολογίσουμε και η Y είναι μία άλλη τυχαία μεταβλητή συσχετισμένη με τη X και με γνωστή μέση τιμή, έστω $E[Y]$. Τότε, ένας εναλλακτικός εκτιμητής είναι ο

$$X_C = X - \alpha \cdot (\mathrm{Y} - \mathrm{E}[\mathrm{Y}])$$

με διασπορά,

$$Var[X_C] = Var[X] + a^2 \cdot Var[Y] - 2 \cdot a \cdot \mathrm{cov}[X, Y]$$

Η διασπορά της X_C θα είναι μικρότερη από αυτήν της X, αν υπάρχει αρκετά μεγάλη συσχέτιση μεταξύ των X και Y και η διασπορά της Y είναι αρκετά μικρή. Η βέλτιστη σταθερά a επιλέγεται μέσω γραμμικής παλινδρόμησης των δεδομένων της προσομοίωσης για την ελαχιστοποίηση της διασποράς του εκτιμητή X_C.

Αν η τυχαία μεταβλητή ελέγχου είναι μέρος του προσομοιωτικού μοντέλου, τότε ονομάζεται *εσωτερική*, ενώ αν αυτή δημιουργείται μέσω μιας βοηθητικής προσομοίωσης, τότε λέμε ότι χρησιμοποιούμε μία *εξωτερική* μεταβλητή ελέγχου.

Στην [LMW82] παρουσιάζεται μια ενδιαφέρουσα γενίκευση της τεχνικής με χρήση περισσοτέρων της μιας μεταβλητών ελέγχου σε προσομοιώσεις δικτύων ουρών. Τα αποτελέσματα που αναφέρονται, καθώς και αυτά της [IL79] περιγράφουν μείωση διασποράς μέχρι και σε ποσοστό 50%.

4.5.4 Δειγματοληψία σημαντικότητας (Importance sampling)

Η δειγματοληψία σημαντικότητας αυξάνει την αποτελεσματικότητα μιας προσομοίωσης επικεντρώνοντας τους υπολογισμούς στις τιμές εκείνες των τυχαίων μεταβλητών εισόδου, που έχουν τη μεγαλύτερη επίδραση στα μέτρα απόδοσης που μελετώνται. Είναι ουσιαστικά μία προσπάθεια δειγματοληψίας από τις τυχαίες μεταβλητές εισόδου όχι σύμφωνα με την κατανομή αυτών, αλλά σύμφωνα με το μέγεθος της επίδρασής τους στα μέτρα απόδοσης που εκτιμώνται.

Η υλοποίηση της τεχνικής γίνεται με την προσομοίωση ενός μοντέλου με κάποια μεροληπτική κατανομή εισόδου που παράγει όμως το ίδιο αποτέλεσμα για το μέτρο απόδοσης που ενδιαφέρει. Το αποτέλεσμα της μεροληπτικής αυτής προσομοίωσης διακρίνεται από μικρότερη διασπορά, καθώς ως τιμές των τυχαίων μεταβλητών εισόδου έχουν χρησιμοποιηθεί εκείνες, στις οποίες παρατηρούνται οι μεγαλύτερες μεταβολές αυτού.

Μία εκτενέστερη παρουσίαση εφαρμογών της τεχνικής αυτής στην προσομοίωση δικτύων ουρών υπάρχει στις [WIL84], [WP84a] και [WP84b].

4.6 Προωθημένες τεχνικές επεξεργασίας αποτελεσμάτων

Η τεχνική των ανεξάρτητων επαναλήψεων, που περιγράφηκε στην παράγραφο 4.4, προϋποθέτει τη συλλογή αποτελεσμάτων από αρκετά μεγάλο αριθμό πειραμάτων προσομοίωσης. Στην παράγραφο αυτή, δίνεται έμφαση στις τεχνικές στατιστικής επεξεργασίας παρατηρήσεων ενός μόνο, αλλά αρκετά μεγάλου πειράματος προσομοίωσης. Το βασικό πρόβλημα, που ανακύπτει στην περίπτωση αυτή, είναι η μη ανεξαρτησία ενός τέτοιου συνόλου παρατηρήσεων, που βασικά καταστρατηγεί τη συνθήκη ισχύος του κεντρικού οριακού θεωρήματος, και έτσι καθιστά αδύνατη τη χρήση της υπολογιστικής διαδικασίας, που περιγράφηκε στην παράγραφο 4.4.

Οι τεχνικές, που έχουν προταθεί για την αντιμετώπιση του συγκεκριμένου προβλήματος, έχουν σκοπό είτε την απομάκρυνση των όποιων συσχετίσεων μεταξύ των παρατηρήσεων, είτε τον περιορισμό αυτών, είτε ακόμη τη συνεκτίμησή τους στην επεξεργασία του αποτελέσματος. Οι τεχνικές αυτές είναι:

- Η *τεχνική των μέσων ομάδων*
- Η *τεχνική των μέσων επικαλυπτόμενων ομάδων*
- Η *τεχνική της αναγέννησης*
- Η *τεχνική της φασματικής ανάλυσης*
- Η *τεχνική της αυτοπαλινδρούμενης αναπαράστασης*
- Η *τεχνική των τυποποιημένων χρονοσειρών*

Άλλα προβλήματα, που αντιμετωπίζει η εκτίμηση με βάση τις παρατηρήσεις ενός μόνο πειράματος προσομοίωσης, είναι αυτό της αρχικής μεταβατικής περιόδου, αλλά και το πρόβλημα του καθορισμού της διάρκειας εκτέλεσης του πειράματος. Οι *διαδικασίες συνεχούς ελέγχου* είναι κατάλληλα διαμορφωμένοι αλγόριθμοι, που ασκούν δυναμικό έλεγχο στην επεξεργασία των παρατηρήσεων ενός πειράματος προσομοίωσης, κατά τη διάρκεια της εκτέλεσης αυτού. Η χρήση ή η μη χρήση της κατάλληλης διαδικασίας συνεχούς ελέγχου φαίνεται ότι παίζει καθοριστικό ρόλο στην ποιότητα των αποτελεσμάτων μιας τέτοιας προσομοιωτικής ανάλυσης.

Για την εκτίμηση της ποιότητας των παραγόμενων αποτελεσμάτων, μελετάται συνήθως κάποιο ή κάποια από τα μεγέθη των σχέσεων (9), (10) και (11) της παραγράφου 4.4, όταν αυτό είναι εφικτό (ύπαρξη αναλυτικής λύσης του μοντέλου). Στη μελέτη όμως της αποτελεσματικότητας των τεχνικών επεξεργασίας παρατηρήσεων, τα σημαντικότερα αποτελέσματα προέρχονται από την εφαρμογή μιας τεχνικής, γνωστής με την επωνυμία *ανάλυση κάλυψης* (coverage analysis).

Ως *κάλυψη* ορίζεται η σχετική συχνότητα των περιπτώσεων που η πραγματική τιμή του υπολογιζόμενου μέτρου απόδοσης περιέχεται στα διαστήματα εμπιστοσύνης που παράγονται από μία συγκεκριμένη τεχνική επεξεργασίας παρατηρήσεων. Θεωρητική μελέτη του λάθους κάλυψης και των βασικών αιτίων αυτού γίνεται στις [GLY82a], [KG85] και [SCH80].

Στην [PAW90] μπορεί κάποιος να ανατρέξει για μία εκτεταμένη ανασκόπηση και αναλυτική περιγραφή του συνόλου των τεχνικών επεξεργασίας παρατηρήσεων που αναφέρθηκαν. Στην παράγραφο αυτή γίνεται μόνο μία συνοπτική περιγραφή των πιο διαδεδομένων, αρχής γενομένης από την τεχνική των μέσων ομάδων.

Κατά την τεχνική αυτή, το σύνολο των παρατηρήσεων $x_1, x_2, ..., x_n$, που συλλέγονται από την εκτέλεση ενός πειράματος προσομοίωσης, χωρίζονται σε μία σειρά μη επικαλυπτόμενων

ομάδων μεγέθους *m*. Οι μέσες τιμές αυτών των ομάδων $\overline{X}_1(m), \overline{X}_2(m),..., \overline{X}_k(m)$, όπου *k* είναι ο αριθμός αυτών, χρησιμοποιούνται σε ένα δεύτερο στάδιο στατιστικής ανάλυσης όμοιας με αυτήν της παραγράφου 4.4. Η ζητούμενη μέση τιμή δίνεται τελικά από την

$$\overline{X}(n) = \overline{\overline{X}}(k,m)$$

ενώ ένα κατάλληλο διάστημα εμπιστοσύνης δίνεται από τη σχέση (6) της παραγράφου 4.4. Στην περίπτωση αυτή, *n* είναι ο αριθμός *k* των ομάδων και x_i είναι οι μέσες τιμές $\overline{X}_i$ αυτών. Η προσέγγιση αυτή βασίζεται στην υπόθεση ότι οι πιο απομακρυσμένες χρονικά παρατηρήσεις παρουσιάζουν μικρότερη συσχέτιση. Έτσι, για αρκετά μεγάλες ομάδες παρατηρήσεων, οι μέσες τιμές αυτών είναι σχεδόν ανεξάρτητες μεταξύ τους.

Ένα πρόβλημα βέβαια, που πρέπει να αντιμετωπιστεί κατά την εφαρμογή της συγκεκριμένης τεχνικής, είναι η επιλογή του κατάλληλου μεγέθους ομάδων, έτσι ώστε, να επιτυγχάνεται μία όσο το δυνατό μικρότερη συσχέτιση μεταξύ των μέσων τιμών αυτών. Στην [SCH82] προτείνεται γενικά ο αριθμός των ομάδων να μην είναι μικρότερος του 10 και μεγαλύτερος του 30, με την προϋπόθεση βέβαια ότι η διάρκεια του πειράματος προσομοίωσης είναι αρκετά μεγάλη, για να διασφαλίζεται η κανονικότητα και η κατά το δυνατόν ανεξαρτησία των μέσων τιμών των ομάδων.

Η μέθοδος των μέσων επικαλυπτόμενων ομάδων είναι ουσιαστικά μία παραλλαγή της απλής μεθόδου μέσων ομάδων, με σκοπό τη μείωση της τιμής του εκτιμητή της διασποράς $\sigma^2[\overline{X}(n)]$. Έτσι, σύμφωνα με την [MS84] η διασπορά του $\overline{X}(n)$ εκτιμάται ως,

$$\hat{\sigma}^2[\overline{X}(n)] = (\frac{n}{m} - 1)^{-1} \cdot \sum_{j=1}^{n-m+1} \frac{\{\overline{X}_j(m) - \overline{X}(n)\}^2}{n-m+1}, \text{ όπου } \overline{X}_j(m) = \frac{1}{m}\sum_{i=0}^{m-1} x_{j+i}$$

είναι η μέση τιμή ομάδας μετά από τη λήψη της παρατήρησης x_j και $\overline{X}(n)$ είναι η συνολική μέση τιμή, που αναφέρεται στο σύνολο των παρατηρήσεων.

Έχει αποδειχθεί [WEL87] ότι επικαλυπτόμενες ομάδες, ακόμη και μόνο στο μισό του μεγέθους τους, δίνουν διασπορά $\sigma^2[\overline{X}(n)]$ στο 75% της διασποράς του αντίστοιχου εκτιμητή της περίπτωσης των μη επικαλυπτόμενων ομάδων και βέβαια για το ίδιο μέγεθος αυτών. Επιπλέον, η μορφή της υπολογιστικής διαδικασίας που χρησιμοποιείται είναι τέτοια, ώστε να είναι πιθανή μέσα στην ίδια διάρκεια πειράματος η αύξηση του μεγέθους των ομάδων χωρίς τη μείωση του αριθμού τους. Έτσι, η συγκεκριμένη τεχνική αποτελεί μία ελκυστική εναλλακτική λύση, παρά τη μεγαλύτερη υπολογιστική της πολυπλοκότητα σε σχέση με την απλή τεχνική των μέσων ομάδων.

Η τεχνική της φασματικής ανάλυσης, που για πρώτη φορά προτάθηκε στην [FK67], διατηρεί τη δομή της τεχνικής των επικαλυπτόμενων ομάδων, με μια σημαντική όμως διαφοροποίηση στον τρόπο υπολογισμού της διασποράς $\sigma^2[\overline{X}(n)]$. Η πιο διαδεδομένη ίσως σήμερα έκδοση της τεχνικής είναι αυτή, που παρουσιάζεται στην [HW81].

Μία κατάλληλα διαμορφωμένη διαδικασία συνεχούς ελέγχου για χρήση της σε προσομοιώσεις με ανάλυση των παρατηρήσεων ανά ομάδες, είναι αυτή που προτείνεται στην [LC79]. Η συγκεκριμένη διαδικασία καθορίζει δυναμικά τόσο τον αριθμό των παρατηρήσεων που χρειάζονται για την επίτευξη διαστημάτων εμπιστοσύνης κάποιας επιθυμητής ακρίβειας, όσο και το πλέον κατάλληλο μέγεθος ομάδας. Όσον αφορά το δεύτερο, στην [FIS78b] ο συγγραφέας προτείνει το δυναμικό έλεγχο της συσχέτισης των μέσων τιμών των ομάδων με τη χρήση ενός κατάλληλου στατιστικού τεστ. Πάντως σε κάθε περίπτωση, δεν πρέπει ποτέ να αγνοείται το γεγονός ότι καμία από τις τεχνικές, που

περιγράφηκαν στην παράγραφο αυτή, δεν είναι απαλλαγμένη από το πρόβλημα της αρχικής μεταβατικής περιόδου. Πρόβλημα, που θα μπορούσε ίσως να αντιμετωπιστεί με την ενσωμάτωση στη διαδικασία συνεχούς ελέγχου κάποιας τεχνικής ανίχνευσης της αρχικής μεταβατικής περιόδου, όπως αυτή που προτείνεται στην [PAW90].

4.7 Η τεχνική της αναγέννησης

Η τεχνική της αναγέννησης επιλέχθηκε ως η πλέον ελκυστική προσέγγιση στατιστικής επεξεργασίας αποτελεσμάτων· μελετήθηκε σε βάθος ([KL00a]) και υλοποίηθηκε με τη μορφή εκτέλεσης σε παράλληλο χρόνο. Διερευνήθηκαν επίσης οι δυνατότητες εφαρμογής της για μοντέλα αυξημένης πολυπλοκότητας, σε θεωρητικό αλλά και σε πρακτικό επίπεδο ([KL00b]). Η αποτελεσματικότητά της εκτιμήθηκε ([KL01a]), τόσο ως προς τις επιτευχθείσες επιταχύνσεις, όσο και ως προς την ποιότητα των αποτελεσμάτων με ανάλυση κάλυψης.

Οι λόγοι για τους οποίους η τεχνική της αναγέννησης επιλέχθηκε ως το πλέον αποτελεσματικό εργαλείο για την προσομοίωση δικτύων ουρών που προκύπτουν κατά την ανάλυση απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής, είναι οι εξής:

- Είναι η μόνη, που αποδίδει και βασίζει την υπολογιστική της διαδικασία σε ένα σύνολο ανεξάρτητων μεταξύ τους παρατηρήσεων.
- Είναι η μόνη, που είναι απαλλαγμένη από το πρόβλημα της αρχικής μεταβατικής περιόδου του συστήματος.
- Είναι από τις λίγες, που υποστηρίζεται από ένα τόσο προωθημένο θεωρητικό υπόβαθρο.
- Ο εκτιμητής της διασποράς σ^2 συγκλίνει στην πραγματική τιμή ταχύτερα ([HG99]) από ότι στις περιπτώσεις των τεχνικών της φασματικής ανάλυσης και των μέσων τιμών μη επικαλυπτόμενων και επικαλυπτόμενων ομάδων.
- Είναι η μοναδική τεχνική, που επιτρέπει την ενσωμάτωση αλγορίθμων εκτίμησης των παραγώγων των μέτρων απόδοσης και άρα τη διενέργεια αναλύσεων ευαισθησίας και βελτιστοποίησης από τα αποτελέσματα ενός μόνο πειράματος προσομοίωσης.

Η διαδικασία αναγέννησης $\{X(t): t \geq 0\}$ με χώρο καταστάσεων R^k είναι ([IGL78]), σύμφωνα με έναν από τους εμπνευστές της τεχνικής, μία στοχαστική διαδικασία, που ξεκινώντας από την αρχική της κατάσταση, διέρχεται μέσα από μία αλληλουχία χρονικών στιγμών αναγέννησης $\{\beta_i: i \geq 1\}$. Αυτό σημαίνει ότι το τμήμα της στοχαστικής διαδικασίας $\{X(t): \beta_i \leq t < \beta_{i+1}\}$ μεταξύ δύο οποιωνδήποτε διαδοχικών χρονικών στιγμών αναγέννησης β_i και β_{i+1}, έχει όμοια αλλά ανεξάρτητη κατανομή σε σχέση με οποιοδήποτε άλλο τμήμα της διαδικασίας, που περιέχεται μεταξύ δύο άλλων διαδοχικών χρονικών στιγμών αναγέννησης. Παρόλα αυτά, το τμήμα της διαδικασίας μεταξύ των χρονικών στιγμών 0 και β_1, μπορεί αν και ανεξάρτητο από τα άλλα τμήματα να έχει μία διαφορετική κατανομή. Η τυπική περίπτωση κατά την οποία ικανοποιείται η υπόθεση της αναγέννησης είναι, όταν κάθε β_i αναπαριστά την i είσοδο του συστήματος σε κάποια συγκεκριμένη κατάσταση s. Μετά το πέρασμα σε μία τέτοια κατάσταση, η εξέλιξη της διαδικασίας συνεχίζει χωρίς καμιά πληροφορία για την ιστορία αυτής. Η συγκεκριμένη κατάσταση ονομάζεται *κατάσταση αναγέννησης* και το τμήμα της διαδικασίας μεταξύ δύο διαδοχικών εισόδων στην κατάσταση αυτή καλείται *κύκλος αναγέννησης*.

Έτσι, το πρόβλημα εφαρμογής της τεχνικής της αναγέννησης παίρνει τη μορφή εντοπισμού μιας τέτοιας επαναληπτικής κατάστασης που μπορεί να θεωρηθεί ότι αντιπροσωπεύει το μοντέλο προσομοίωσης σε κάποια χρονική στιγμή, όταν αυτό βρίσκεται σε κατάσταση στατιστικής ισορροπίας. Η συγκεκριμένη δυσκολία είναι το αντικείμενο της κριτικής που έχει δεχθεί η τεχνική αυτή, μιας κριτικής, που αμφισβητεί της δυνατότητα εφαρμογής της σε πολύπλοκα προσομοιωτικά μοντέλα.

Παρόλα αυτά, μία σε βάθος μελέτη των σχετικών εργασιών, έδειξε ότι η κριτική αυτή είναι σε μεγάλο βαθμό ατεκμηρίωτη.

Ξεκινώντας από την περίπτωση των κλειστών δικτύων ουρών με στοχαστική δρομολόγηση έργων επεξεργασίας και εκθετικούς χρόνους εξυπηρέτησης, είναι σαφές ότι η υποκείμενη στοχαστική διαδικασία ενός τέτοιου μοντέλου είναι μία διαδικασία Markov με πεπερασμένο χώρο καταστάσεων, αλλά όχι απαραίτητα αδιαχώριστη. Σε ένα τέτοιο μοντέλο απόδοσης, είναι δυνατό να υπάρχουν μία ή περισσότερες παροδικές καταστάσεις και περισσότερα από ένα κλειστά αδιαχώριστα σύνολα επαναληπτικών καταστάσεων (βλ. παράγραφο Α.3 του παραρτήματος για τους ορισμούς των σχετικών εννοιών). Παρόλα αυτά, στο [SHE93] αποδεικνύεται ότι:

> *Θεώρημα 1*: Αν υπάρχει τουλάχιστο ένα κέντρο εξυπηρέτησης, το οποίο δέχεται έργα επεξεργασίας μιας μόνο κλάσης, ή τα έργα μικρότερης προτεραιότητας, που δέχεται, υπόκεινται σε διακοπή εξυπηρέτησης (pree-mption), τότε η διαδικασία Markov του συγκεκριμένου συστήματος αποτελείται από ένα και μοναδικό κλειστό και αδιαχώριστο σύνολο επαναληπτικών καταστάσεων.

Εδώ είναι σημαντικό να αποσαφηνισθεί η έννοια της κλάσης έργων, όπως ακριβώς την εννοεί ο συγγραφέας. Ενώ λοιπόν κάθε έργο επεξεργασίας είναι ενός συγκεκριμένου τύπου, που σε καμία περίπτωση δεν αλλάζει κατά την πορεία του μέσα στο δίκτυο, μπορεί κατά την άφιξη αυτού στο κέντρο εξυπηρέτησης *m* να γίνεται δεκτό ως έργο κλάσης *a* και κατά την άφιξή του στο κέντρο εξυπηρέτησης *n* να γίνεται δεκτό ως έργο κλάσης *b*. Οι κλάσεις έργων δηλαδή ορίζονται τοπικά ως προς κάθε κέντρο εξυπηρέτησης και διευκολύνουν ουσιαστικά τη μοντελοποίηση των διαφορετικών απαιτήσεων των έργων στα σημεία εξυπηρέτησης που αυτό διαθέτει. Έτσι, οι τύποι των έργων επεξεργασίας είναι πληροφορία που επηρεάζει τη δρομολόγηση και τις απαιτήσεις εξυπηρέτησης σε κάθε κέντρο και οι κλάσεις έχουν μόνο τοπική ισχύ στο κάθε κέντρο και αποτελούν ουσιαστικά ένα μέσο διαφοροποίησης των απαιτήσεων εξυπηρέτησης σε αυτό.

Αυτό σημαίνει ότι, αν εξαιρέσει κανείς την υπόθεση εκθετικής εξυπηρέτησης, τότε το θεώρημα 1 είναι αρκετά γενικό, καθώς βρίσκει εφαρμογή σε ένα σημαντικό αριθμό περιπτώσεων. Στις περιπτώσεις αυτές λοιπόν η διαδικασία Markov, που προκύπτει από τον περιορισμό της αρχικής διαδικασίας στο σύνολο των επαναληπτικών της καταστάσεων, είναι μία αδιαχώριστη και θετικά επαναληπτική διαδικασία Markov και σύμφωνα με σχετικό θεώρημα χαρακτηρίζεται από την ιδιότητα της αναγέννησης. Τότε, κάθε κατάσταση *s* του συστήματος με όλα τα έργα τοποθετημένα σε κέντρο εξυπηρέτησης, που είτε δέχεται το σύνολο αυτών ως έργα της ίδιας κλάσης είτε τα έργα μικρότερης προτεραιότητας υπόκεινται σε διακοπή εξυπηρέτησης, είναι μία κατάσταση αναγέννησης. Έτσι, αυτή μπορεί να αποτελέσει την αρχική κατάσταση μιας προσομοίωσης αναγέννησης.

Τι γίνεται όμως στις περιπτώσεις δικτύων ουρών με μη εκθετικούς χρόνους εξυπηρέτησης; Τότε η υποκείμενη στοχαστική διαδικασία, που μελετάται, είναι η γενικευμένη ημι-μαρκοβιανή διαδικασία (GSMP), που εκφράζει την εκτέλεση του πειράματος προσομοίωσης (βλ. παράγραφο 4.1). Ας υποθέσουμε την ύπαρξη μιας κατάστασης *s*, στην οποία όλα τα έργα επεξεργασίας βρίσκονται συσσωρευμένα σε κέντρο εξυπηρέτησης, που είτε δέχεται κάθε

έργο επεξεργασίας στην ίδια κλάση, είτε τα έργα της μικρότερης προτεραιότητας υπόκεινται σε διακοπή εξυπηρέτησης. Έστω επίσης το σύνολο *D* όλων των καταστάσεων που είναι προσβάσιμες από την *s*. Τότε, όπως αποδεικνύεται στο [SHE93] η GSMP που προκύπτει από τον περιορισμό της GSMP της προσομοίωσης στο σύνολο *D*, είναι μία διαδικασία αναγέννησης.

Αυτού του είδους η θεωρητική ανάλυση όμως, βασίζεται στην υπόθεση ύπαρξης της συγκεκριμένης κατάστασης *s* του συστήματος, γνωστής και ως *μοναδικής κατάστασης*. Στην [GLY82b], ο συγγραφέας επεκτείνει τον ορισμό χρονικών στιγμών αναγέννησης σε περιπτώσεις προσομοίωσης διακριτών γεγονότων χωρίς τον προαναφερθέν περιορισμό, ενώ στην [GLY89] ο ίδιος συγγραφέας διατυπώνει εύκολα επαληθεύσιμες συνθήκες για την ύπαρξη ή μη της ιδιότητας της αναγέννησης σε μία GSMP.

Σε περιπτώσεις ανοικτών δικτύων ουρών είναι απαραίτητη η χρήση ενός ανώτατου ορίου στο μέγεθος του κύκλου αναγέννησης. Με τον τρόπο αυτό ελέγχεται το ενδεχόμενο μη ύπαρξης κατάστασης στατιστικής ισορροπίας του μοντέλου (βλ. σχέση (1) της παραγράφου 3.1). Όσον αφορά την επιλογή κατάστασης αναγέννησης, αυτή που συνήθως προτείνεται είναι το κενό δίκτυο ουρών (χωρίς δηλαδή έργα επεξεργασίας), καθώς φαίνεται [KL00a] ότι στις περισσότερες περιπτώσεις αποτελεί μία από τις πιο συχνά εμφανιζόμενες επαναληπτικές καταστάσεις.

Ας υποθέσουμε ότι ο σκοπός ενός πειράματος προσομοίωσης είναι ο υπολογισμός της μέσης τιμής ενός χαρακτηριστικού (π.χ. ενός μεγέθους παραγωγής) κάποιου δικτύου ουρών, που δίνεται ως μία συνάρτηση *f* κάποιας διαδικασίας αναγέννησης $X = \{X(t); t \geq 0\}$,

$$k(f) = E[f(X)] \tag{13}$$

Ας θεωρήσουμε επίσης την,

$$Z_k(f) = \int_{T_{k-1}}^{T_k} f(X(u)) \cdot du$$

που αντιπροσωπεύει την παρατήρηση του *k* κύκλου αναγέννησης. Τότε ένα $(100 \cdot a)\%$ διάστημα εμπιστοσύνης για την *k*(*f*) μετά από τη συμπλήρωση *N* κύκλων αναγέννησης δίνεται ([IGL78]) από τη σχέση,

$$\left[\hat{k}(N) - \frac{s(N) \cdot F^{-1}\left(\frac{1+a}{2}\right)}{\sqrt{N} \cdot \bar{\tau}(N)}, \hat{k}(N) + \frac{s(N) \cdot F^{-1}\left(\frac{1+a}{2}\right)}{\sqrt{N} \cdot \bar{\tau}(N)} \right] \tag{14}$$

όπου

$\bar{\tau}(N)$ είναι το μέσο μήκος των κύκλων αναγέννησης

και

$$\hat{k}(N) = \frac{\bar{Z}(N)}{\bar{\tau}(N)} \tag{15}$$

$$s^2(N) = s_{11}^2(N) - 2\hat{k}(N)s_{12}^2(N) + (\hat{k}(N))^2 s_{22}^2(N) \tag{16}$$

με

$$s_{11}^2(N) = \frac{1}{N-1}\sum_{k=1}^{N}(Z_k(f) - \overline{Z}(N))^2$$

$$s_{22}^2(N) = \frac{1}{N-1}\sum_{k=1}^{N}(\tau_k - \overline{\tau}(N))^2$$

$$s_{12}^2(N) = \frac{1}{N-1}\sum_{k=1}^{N}(Z_k(f) - \overline{Z}(N))(\tau_k - \overline{\tau}(N))$$

Εύκολα αποδεικνύεται ότι ο εκτιμητής της (15) είναι *συνεπής*, που σημαίνει ότι τείνει στην πραγματική μέση τιμή με πιθανότητα 1, όταν $N \to \infty$. Επιπλέον, η εκτίμηση, με βάση τη συγκεκριμένη υπολογιστική διαδικασία, είναι απαλλαγμένη από τη μεροληψία εξαιτίας της αρχικής μεταβατικής περιόδου, καθώς το σύστημα αρχικοποιείται σε κατάσταση στατιστικής ισορροπίας. Παρόλα αυτά, το γεγονός ότι ο εκτιμητής της (15) εκφράζεται ως λόγος εκτιμήσεων, εισάγει μία νέα πηγή μεροληψίας.

Οι εκτιμητές, που σύμφωνα με την [IGL78] είναι δυνατό να χρησιμοποιηθούν, με απώτερο στόχο τη μείωση της μεροληψίας, που εισάγει η κλασσική διαδικασία υπολογισμού με βάση την τεχνική της αναγέννησης, είναι:

- Ο *εκτιμητής Fieller*,

  $\hat{k}(N) = \dfrac{\overline{Z}(N)\cdot\overline{\tau}(N) - q\cdot s_{12}}{[\overline{\tau}(N)]^2 - q\cdot s_{22}}$, όπου $q = (F^{-1}(\dfrac{1+a}{2}))^2 \Big/ N$

- Ο *εκτιμητής Beale*,

  $\hat{k}(N) = \dfrac{\overline{Z}(N)}{\overline{\tau}(N)} \cdot \dfrac{1 + s_{12}/N\cdot\overline{Z}(N)\cdot\overline{\tau}(N)}{1 + s_{22}/N\cdot(\overline{\tau}(N))^2}$

- Ο *εκτιμητής jackknife*,

  $\hat{k}(N) = \dfrac{1}{N}\cdot\sum_{i=1}^{N}\theta_i$, όπου

  $\theta_i = N\cdot(\overline{Z}(N)/\overline{\tau}(N)) - (N-1)(\sum_{j\neq i} Z_j \Big/ \sum_{j\neq i}\tau_j)$

- Ο *εκτιμητής Tin*,

  $\hat{k}(N) = \dfrac{\overline{Z}(N)}{\overline{\tau}(N)}\cdot\left[1 + \left(\dfrac{s_{12}}{\overline{Z}(N)\cdot\overline{\tau}(N)} - \dfrac{s_{22}}{(\overline{\tau}(N))^2}\right)\cdot N^{-1}\right]$

Η διαδικασία εκτίμησης της διασποράς σ^2 παραμένει η ίδια με την κλασσική σε όλες τις περιπτώσεις, εκτός από αυτές των εκτιμητών Fieller και jackknife.

Επιπλέον, στην [CLL99] προτείνεται ένας αλγόριθμος για την παραγωγή διαστημάτων εμπιστοσύνης με την προσέγγιση *bootstrap*. Με τον τρόπο αυτό, γίνεται προσπάθεια αντιμετώπισης του ενδεχομένου μη επαρκούς κανονικότητας των παρατηρήσεων. Παράλληλα, οι παραγόμενες εκτιμήσεις δε διακρίνονται από ασύμμετρη κατανομή, γεγονός που αποτελεί τη βασική αιτία εμφάνισης λάθους κάλυψης. Στα μείον της προσέγγισης bootstrap, καταγράφεται το σαφώς πιο υψηλό υπολογιστικό κόστος σε σχέση με τις προαναφερθείσες.

Οι δοκιμές, που έγιναν με τους εκτιμητές που αναφέρονται στην [IGL78], έδειξαν ότι ασυμπτωτικά για μεγάλης διάρκειας πειράματα προσομοίωσης οι εκτιμητές Fieller/ Fieller,

jackknife/jackknife, Tin/κλασσικός και κλασσικός/κλασσικός έδωσαν ακριβή αποτελέσματα για όλα τα μοντέλα, που δοκιμάσθηκαν. Αντίθετα, για μικρής διάρκειας πειράματα, παρόλο που κανένας εκτιμητής δεν έδωσε ικανοποιητικά αποτελέσματα στην ανάλυση κάλυψης, που έγινε, υπερείχαν οι jackknife/jackknife, Tin/κλασσικός και κλασσικός/κλασσικός. Όσον αφορά τώρα τη χρήση της προσέγγισης bootstrap, οι πρώτες πειραματικές συγκρίσεις σε σχέση με τους προαναφερθέντες εκτιμητές είναι ιδιαίτερα ενθαρρυντικές, παρόλο που, όπως τονίζεται στη σχετική εργασία ([CLL99]), ούτε αυτή η υπολογιστική διαδικασία μπορεί να εγγυηθεί την απουσία λάθους κάλυψης.

Η τελευταία αυτή διαπίστωση αναδεικνύει τη σημασία της εφαρμογής διαδικασίας συνεχούς ελέγχου και στην περίπτωση της τεχνικής της αναγέννησης. Η πρώτη από τις δύο πιο γνωστές, που έχουν προταθεί, είναι αυτή, που για πρώτη φορά εφαρμόσθηκε στην [LS77] και προκύπτει με παρόμοιο τρόπο, όπως και η σχέση (8) της παραγράφου 4.4. Πιο συγκεκριμένα, στην περίπτωση αυτή, ο αριθμός των κύκλων αναγέννησης που χρειάζονται, για την παραγωγή ενός 100α% διαστήματος εμπιστοσύνης με μήκος όχι μεγαλύτερο από το $(100 \cdot 2\delta)\%$ της εκτιμηθείσας τιμής καθορίζεται δυναμικά από τη σχέση,

$$N \geq \left(\frac{F^{-1}\left(\frac{1+a}{2}\right)}{\delta} \right)^2 \cdot \left(\frac{s(l)}{\hat{k}(l) \cdot \bar{\tau}(l)} \right)^2 \qquad (17)$$

όπου $s(l), \hat{k}(l), \bar{\tau}(l)$ είναι οι εκτιμήσεις μετά το τέλος του l κύκλου αναγέννησης. Έτσι, ο αριθμός των κύκλων που απαιτούνται για την επίτευξη της επιθυμητής ακρίβειας, επανεκτιμάται στο τέλος κάθε κύκλου αναγέννησης. Όταν βρεθεί ότι ήδη ικανοποιείται η σχέση (17), τότε ολοκληρώνεται η εκτέλεση.

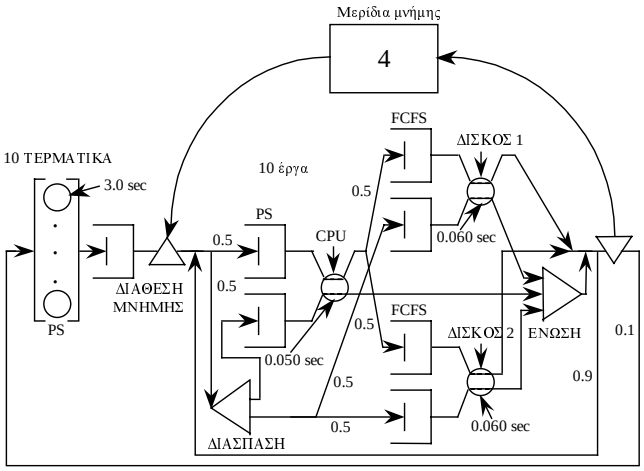
Μία άλλη διαδικασία συνεχούς ελέγχου είναι αυτή που διατυπώνεται στην [FIS77]. Σε πρώτη φάση χρησιμοποιείται ο εκτιμητής Tin για την παραγωγή μιας σειράς ανεξάρτητων και κανονικά κατανεμημένων παρατηρήσεων. Η κανονικότητα των παρατηρήσεων ελέγχεται με τη χρήση του τεστ Shapiro - Wilk και από το σύνολο αυτών, παράγεται ένα 100α% διάστημα εμπιστοσύνης. Η διαδικασία ολοκληρώνει την εκτέλεση, όταν το μήκος του παραγόμενου διαστήματος δεν είναι μεγαλύτερο από το $(100 \cdot 2\delta)\%$ της εκτιμηθείσας τιμής, όπου δ ορίζει την επιθυμητή ακρίβεια.

Μία ενδιαφέρουσα συγκριτική ανάλυση κάλυψης των δύο διαδικασιών συνεχούς ελέγχου για την τεχνική της αναγέννησης και αυτής της [LC79], που είναι κατάλληλη για ανάλυση παρατηρήσεων προσομοίωσης ανά ομάδες, μπορεί κανείς να βρει στην [LK82b]. Πάντως το μειονέκτημα της συγκεκριμένης μελέτης έγκειται στο γεγονός ότι τα εμπειρικά αποτελέσματα που παραθέτει, βασίζονται σε δείγματα μόνο 100 πειραμάτων προσομοίωσης, δείγματα που μεταγενέστερες μελέτες, όπως η [PME98] και η [KL01a], έδειξαν ότι δεν μπορούν να θεωρηθούν στατιστικά σημαντικά και άρα τα αποτελέσματα δεν μπορούν να θεωρηθούν αξιόπιστα.

Μία πειραματική εκτίμηση της λειτουργίας ενός αλγορίθμου αναγέννησης, με χρήση της διαδικασίας συνεχούς ελέγχου της [LS77], έγινε στις [KL00a] και [KL00b]. Στην πρώτη εργασία έγινε εφαρμογή της μεθόδου σε ανοικτά και κλειστά δίκτυα ουρών, απλής μορφής, με έναν ή περισσότερους τύπους έργων και πειθαρχίες προτεραιοτήτων. Μία σημαντική παρατήρηση, στην εφαρμογή της συγκεκριμένης διαδικασίας, είναι το γεγονός ότι συχνά η σχέση (17) ικανοποιούνταν προσωρινά μετά από ένα πολύ μικρό αριθμό κύκλων, γεγονός που οδηγούσε στη μη αναμενόμενη παύση των υπολογισμών, και στην παραγωγή αποτελεσμάτων αμφίβολης ακρίβειας. Η διαπίστωση αυτή οδήγησε στο συμπέρασμα ότι

πρέπει πάντα να χρησιμοποιείται ένας ελάχιστος αριθμός κύκλων αναγέννησης και μόνο μετά την πάροδο αυτών να τίθεται σε ισχύ η συνθήκη της σχέσης (17). Συμπέρασμα, που εξάλλου επιβεβαιώνεται και στην [LMP99], όπου μετά από μεγάλο αριθμό πειραμάτων σε σύστημα M/M/1, προτείνεται ως ένας τέτοιος ελάχιστος αριθμός κύκλων το 30.

Επιπλέον, στην [KL00b] παρουσιάσθηκαν τα αποτελέσματα της προσομοίωσης αναγέννησης του μοντέλου του σχήματος 4.6, που αναπαριστά ένα αλληλεπιδραστικό υπολογιστικό σύστημα με ανταγωνισμό για χρήση πόρων μνήμης και επικάλυψη διεργασιών. Διακρίνουμε τη χρήση παθητικού πόρου, αλλά και κέντρων διάσπασης και ένωσης έργων. Η κατάσταση αναγέννησης, που επιλέχθηκε είναι αυτή, με όλα τα έργα επεξεργασίας, τοποθετημένα στο κέντρο καθυστέρησης των τερματικών. Τα 90% διαστήματα εμπιστοσύνης των μέτρων απόδοσης, που υπολογίζονται, απαιτείται να μην είναι μεγαλύτερα από το 8% της εκτιμηθείσας τιμής.



Σχήμα 4.6 Ένα κλειστό δίκτυο ουρών με παθητικό πόρο και κέντρα διάσπασης /ένωσης έργων

```
ΧΡΟΝΟΣ ΠΡΟΣΟΜΟΙΩΣΗΣ:                  2938 sec
ΑΡΙΘΜΟΣ ΚΥΚΛΩΝ:                       132
ΜΕΣΟ ΜΗΚΟΣ ΚΥΚΛΟΥ:                    22.087 sec
90% ΔΙΑΣΤΗΜΑ ΕΜΠΙΣΤΟΣΥΝΗΣ:            (17.311, 26.862)
ΣΥΝΟΛΙΚΟΣ ΑΡΙΘΜΟΣ ΓΕΓΟΝΟΤΩΝ:          106853
ΜΕΣΟΣ ΑΡΙΘΜΟΣ ΓΕΓΟΝΟΤΩΝ ΑΝΑ ΚΥΚΛΟ:    809.492

                          ΠΑΡΑΓΩΓΗ έργα/sec              ΑΞΙΟΠΟΙΗΣΗ
                        LB     MEAN     UB          LB     MEAN     UB
ΤΕΡΜΑΤΙΚΑ             1.688   1.715   1.743      5.174   5.214   5.255
ΚΕΝΤΡΟ ΔΙΑΘΕΣΗΣ ΜΝΗΜΗΣ 1.688   1.715   1.743      3.468   3.534   3.601
CPU                  16.955  17.330  17.705       .848    .867    .886
ΔΙΣΚΟΣ1               8.563   8.741   8.920       .509    .518    .527
ΔΙΣΚΟΣ2               8.409   8.589   8.768       .503    .513    .523

                       ΣΥΝΟΛΙΚΟ ΜΗΚΟΣ έργα      ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ sec
                        LB     MEAN     UB          LB     MEAN     UB
ΤΕΡΜΑΤΙΚΑ             5.174   5.214   5.255      2.977   3.040   3.102
ΚΕΝΤΡΟ ΔΙΑΘΕΣΗΣ ΜΝΗΜΗΣ 4.722   4.786   4.849      2.681   2.790   2.899
CPU                   2.241   2.275   2.309       .129    .131    .133
ΔΙΣΚΟΣ1                .866    .876    .887       .098    .100    .103
ΔΙΣΚΟΣ2                .840    .852    .864       .098    .099    .101
```

Σχήμα 4.7 Αποτελέσματα προσομοίωσης του μοντέλου του σχήματος 4.6

Τα αποτελέσματα του πειράματος προσομοίωσης εικονίζονται στο σχήμα 4.7. Το συγκεκριμένο μοντέλο απόδοσης αποτελεί και μία επίδειξη της αποτελεσματικότητας της

τεχνικής της αναγέννησης σε δίκτυα ουρών που δεν είναι μορφής γινομένου, με ρεαλιστικά χαρακτηριστικά και σημαντική πολυπλοκότητα.

4.8 Προσομοίωση αναγέννησης σε παράλληλο χρόνο

Από τις σχέσεις (14), (15) και (16) της παραγράφου 4.7, που περιγράφουν την κλασσική διαδικασία εκτίμησης της τεχνικής της αναγέννησης, είναι σαφές ότι οι εκτιμήσεις εξαρτώνται μόνο από τις διαφορές των χρονικών στιγμών έναρξης από τις χρονικές στιγμές λήξης των κύκλων αναγέννησης. Οι διαφορές αυτές όμως, συνιστούν τυχαία διαδικασία, η οποία επανεκκινείται στοχαστικά στο τέλος κάθε κύκλου αναγέννησης. Αυτό πρακτικά σημαίνει ότι η παράμετρος του χρόνου είναι σημαντική, μόνον όσον αφορά τη συμπλήρωση του απαιτούμενου αριθμού κύκλων αναγέννησης.

Σε μία διάταξη παράλληλου χρόνου ([KL00b]), επιχειρείται η επίτευξη επαρκή αριθμού κύκλων σε χρόνο υποπολλαπλάσιο της απλής σειριακής προσομοίωσης. Κάθε ΛΔ περιέχει το σύνολο του μοντέλου προσομοίωσης. Ένας αριθμός ΛΔ - ίσος με τον αριθμό των διαθέσιμων επεξεργαστών - ξεκινά ταυτόχρονα την εκτέλεση του πειράματος, την οποία ολοκληρώνει, όταν ικανοποιηθεί η συνθήκη εξόδου της διαδικασίας συνεχούς ελέγχου, που εφαρμόζεται.

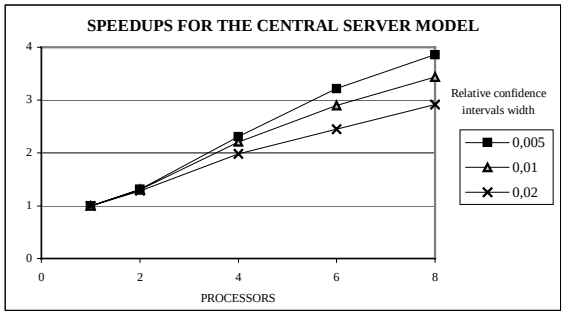
Εφόσον οι υπολογισμοί στατιστικών στο τέλος κάθε κύκλου αναγέννησης και ο έλεγχος της συνθήκης εξόδου της διαδικασίας συνεχούς ελέγχου λαμβάνουν χώρα κάτω από συνθήκες προστασίας σύγχρονης προσπέλασης, η επιτάχυνση, που είναι δυνατό να επιτευχθεί, είναι σαφές ότι εξαρτάται από το υπολογιστικό κόστος των κύκλων αναγέννησης του μοντέλου. Εναλλακτικά, είναι εφικτή η χρήση συγκεκριμένων σημείων ελέγχου της συνθήκης εξόδου, κατά τη ροή της εκτέλεσης του πειράματος. Έτσι, μειώνεται η επίδραση του κρίσιμου τμήματος του λογισμικού προσομοίωσης, που στην προηγούμενη περίπτωση εκτελείται στο τέλος κάθε κύκλου αναγέννησης κάτω από προστασία σύγχρονης προσπέλασης, περιορίζοντας ουσιαστικά τα περιθώρια επίτευξης πιο μεγάλων επιταχύνσεων.

Η [KL01a] είναι η πρώτη εργασία, που παρουσιάζει πειραματικά δεδομένα, τα οποία αφορούν την ακρίβεια των αποτελεσμάτων μιας τέτοιας διάταξης προσομοίωσης. Η διαδικασία συνεχούς ελέγχου, που εφαρμόστηκε, είναι αυτή της [LS77] τροποποιημένη, έτσι ώστε να υποστηρίζει τον ταυτόχρονο υπολογισμό περισσοτέρων του ενός μέτρων απόδοσης. Έλεγχος διεξάγεται στο τέλος κάθε κύκλου αναγέννησης, με σκοπό τη διασφάλιση της μη εκτέλεσης επιπλέον κύκλων μετά την επίτευξη της επιθυμητής ακρίβειας στα διαστήματα εμπιστοσύνης.

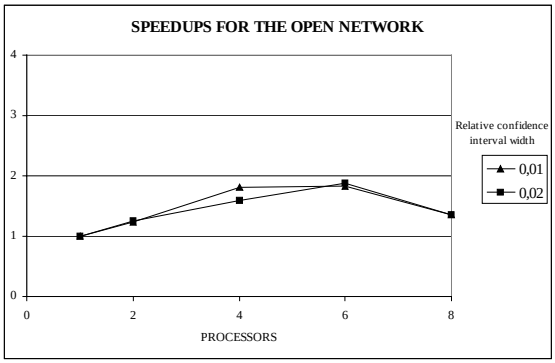
Το λογισμικό προσομοίωσης υλοποιήθηκε σε C++ και ακολούθως παραλληλοποιήθηκε με τη χρήση μεταφραστή OpenMP ([OPE97]) σε σύστημα διαμοιραζόμενης μνήμης SUN E3500 του Edinburgh Parallel Computing Centre (EPCC). Η εκτίμηση της ακρίβειας των αποτελεσμάτων έγινε με ανάλυση κάλυψης στα αποτελέσματα προσομοίωσης ενός κλειστού και ενός ανοικτού δικτύου ουρών με 1, 2, 4, 6, και 8 επεξεργαστές. Σε κάθε περίπτωση εκτελέστηκαν 200 επαναλήψεις του ιδίου πειράματος με ομοιόμορφα μεταβαλλόμενο σπόρο και για μεγέθη δ των μισών 90% διαστημάτων εμπιστοσύνης όχι μεγαλύτερα από 2%, 1% και 0.5% για το κλειστό δίκτυο και 2% και 1% για το ανοικτό δίκτυο ουρών. Τα μέτρα απόδοσης που μελετήθηκαν είναι οι μέσες παραγωγές, η αξιοποίηση, οι χρόνοι απόκρισης και οι αριθμοί έργων στα κέντρα εξυπηρέτησης των δικτύων ουρών.

Στα σχήματα 4.8 και 4.9 εμφανίζονται οι επιτευχθείσες επιταχύνσεις. Οι αριθμοί αντιπροσωπεύουν τους μέσους όρους 200 επαναλήψεων για κάθε αριθμό επεξεργαστών και επίπεδο επιθυμητής ακρίβειας. Είναι σαφές ότι ενώ για το κλειστό δίκτυο ουρών είχαμε μέχρι και 4 φορές πιο γρήγορη εκτέλεση, στην περίπτωση του ανοικτού δικτύου ουρών τα κέρδη ήταν περιορισμένα εξαιτίας του μικρού κύκλου αναγέννησης (μόνο 77 γεγονότα κατά μέσο

όρο) του μοντέλου και των μεγάλων απαιτήσεων συγχρονισμού, που για το λόγο αυτό δημιουργήθηκαν. Η παρατήρηση αυτή όμως δεν έχει καμία πρακτική σημασία, καθώς σε ρεαλιστικές μελέτες δεν εμφανίζονται ποτέ τόσο απλά μοντέλα με τόσο μικρό κύκλο αναγέννησης.



Σχήμα 4.8 Επιταχύνσεις προσομοίωσης αναγέννησης για ένα κλειστό δίκτυο ουρών (200 επαναλήψεις/σημείο)



Σχήμα 4.9 Επιταχύνσεις προσομοίωσης αναγέννησης για ένα απλό ανοικτό δίκτυο ουρών (200 επαναλήψεις/σημείο)

Μελετήθηκε επίσης, όπως ήδη τονίστηκε, η ακρίβεια των αποτελεσμάτων, που αποδίδει μία τέτοια διάταξη προσομοίωσης, με ανάλυση κάλυψης των αναλυτικών λύσεων από τα 90% διαστήματα εμπιστοσύνης των πειραμάτων που εκτελέστηκαν. Σύμφωνα με την [LS77], μία εκτιμητική διαδικασία μπορεί να θεωρηθεί αξιόπιστη, όταν το άνω όριο του διαστήματος εμπιστοσύνης της κάλυψης υπερκαλύπτει την ονομαστική ακρίβεια του πειράματος, που στην περίπτωση της συγκεκριμένης μελέτης ήταν 0.9.

Η ανάλυση κάλυψης έδειξε ότι η ενσωμάτωση του νόμου του Little στους υπολογισμούς για την εκτίμηση των χρόνων απόκρισης στα κέντρα εξυπηρέτησης, απέδωσε αποτελέσματα μη αποδεκτής ποιότητας. Σίγουρα το φαινόμενο απαιτεί μία σε βάθος θεωρητική μελέτη, καθώς τα αποτελέσματα εμφάνισαν απαράδεκτη κάλυψη ακόμη και στην περίπτωση της σειριακής εκτέλεσης, παρόλο που η τεχνική αυτή περιορισμού της διασποράς προτείνεται από σημαντική μερίδα της βιβλιογραφίας. Πάντως, ως εναλλακτική λύση μπορεί να χρησιμοποιηθεί η τεχνική της σήμανσης των έργων επεξεργασίας, που προτείνεται και θεμελιώνεται θεωρητικά στο [SHE93].

Όσον αφορά τα άλλα μέτρα απόδοσης, στις περιπτώσεις χρήσης περισσοτέρων του ενός επεξεργαστές υπάρχει μία πιο εντυπωσιακή βελτίωση της κάλυψης, καθώς αυξάνονται οι απαιτήσεις ακρίβειας (ως ποσοστό δ της εκτιμηθείσας τιμής) του πειράματος. Παρατηρείται ακόμη μία τάση μείωσης της ακρίβειας των αποτελεσμάτων με τη χρήση μεγαλύτερου αριθμού επεξεργαστών. Παρόλα αυτά, ακόμη και στην περίπτωση χρήσης 8 επεξεργαστών η

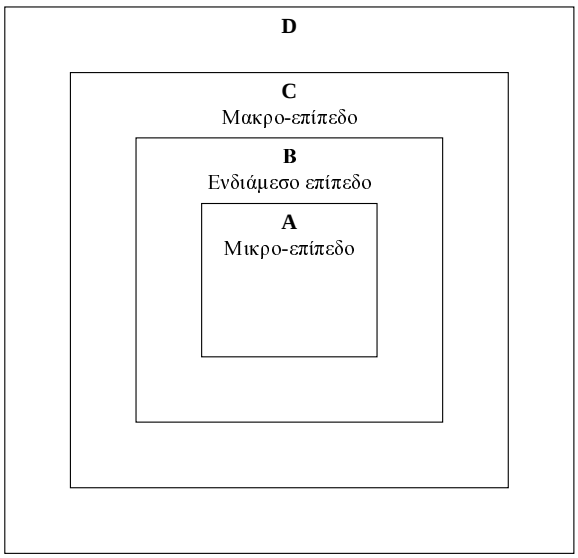
απαίτηση υψηλής ακρίβειας (δ=0.005) απέδωσε αποτελέσματα με κάλυψη, της οποίας το 90% διάστημα εμπιστοσύνης, περιλαμβάνει την ονομαστική ακρίβεια 0.9.

Από τα παραπάνω, προκύπτει ότι μία κατάλληλη διαδικασία συνεχούς ελέγχου δεν είναι απλά ένα εργαλείο ελέγχου της διάρκειας του πειράματος, αλλά και ένα εργαλείο διασφάλισης της ποιότητας των αποτελεσμάτων, με ακόμη πιο σημαντικό ρόλο στις περιπτώσεις χρήσης μεγάλου αριθμού επεξεργαστών. Όσο πιο πολλοί επεξεργαστές χρησιμοποιούνται, τόσο πιο αυστηρές απαιτήσεις ακρίβειας πρέπει να καθοδηγούν την εκτέλεση του πειράματος.

Πειραματικές εκτιμήσεις διατάξεων προσομοίωσης παράλληλου χρόνου έχουν επίσης εμφανισθεί για τις τεχνικές των μέσων τιμών ομάδων και των μέσων τιμών επικαλυπτόμενων ομάδων ([PME98], [PME94]) και για την τεχνική της φασματικής ανάλυσης ([PME98] , [PME94] και [RAA92]). Επίσης, στην [YAU99] παρουσιάζεται ένα λογισμικό αυτοματοποιημένης παραλληλοποίησης απλών σειριακών μοντέλων και ελέγχου της ακρίβειας των εκτιμήσεων των παράλληλων προσομοιώσεων. Τέλος, στην [PME98] προτείνεται η χρήση διαδικασίας συνεχούς ελέγχου και για την ανάλυση κάλυψης των αποτελεσμάτων ενός πειράματος προσομοίωσης.

4.9 Ιεραρχική μοντελοποίηση και προσομοίωση αναγέννησης

Το πρόβλημα που συνήθως εμφανίζεται, κατά την ανάπτυξη του μοντέλου απόδοσης ενός πολύπλοκου συστήματος, είναι το γεγονός ότι ένας πόρος μπορεί να λειτουργεί ως ένα κέντρο εξυπηρέτησης, που και το ίδιο αιτεί και λαμβάνει εξυπηρέτηση από ένα σύνολο πόρων χαμηλότερου επιπέδου. Μία πρακτική προσέγγιση είναι η υιοθέτηση μιας ιεραρχικής δομής, όπως αυτής που απεικονίζεται με αφαιρετικό τρόπο στο σχήμα 4.10.



Σχήμα 4.10 Μία ιεραρχική δομή μοντέλου συστήματος

Στην αναπαράσταση του σχήματος, το μοντέλο Α, που βρίσκεται στο χαμηλότερο επίπεδο, επιλύεται και ενσωματώνεται με μία κατάλληλη μορφή (π.χ. ένα κέντρο FESC) στο μοντέλο Β του αμέσως υψηλότερου ιεραρχικά επιπέδου. Τα αποτελέσματα της αποτίμησης του μοντέλου Β ενσωματώνονται με ανάλογο τρόπο στο μοντέλο C κ.ο.κ.

Φυσικά, η ολοκληρωτική αποσύνθεση του συνόλου του μοντέλου σε ανεξάρτητα μεταξύ τους υποσυστήματα είναι συνήθως ανέφικτη. Στις περισσότερες περίπτωσεις όμως, βρίσκει

εφαρμογή η έννοια του σχεδόν ολοκληρωτικά αποσυνθέσιμου συστήματος ([COU75]), που αναφέρθηκε και στην ενότητα 3.1.4. Σύμφωνα με αυτήν, η ανάλυση του μοντέλου Α μπορεί να γίνει σχεδόν ανεξάρτητα από την ανάλυση του συμπληρωματικού του μέρους στο μοντέλο Β και ομοίως για τα υπόλοιπα μοντέλα της ιεραρχίας.

Η βασική απαίτηση της σχεδόν ολοκληρωτικής αποσυνθεσιμότητας είναι, κατά την [COU75], το υποσύστημα Α να παρουσιάζει μία μεταβατική περίοδο πολύ μικρής διάρκειας σε σχέση με το μέσο χρόνο μεταξύ των αλληλεπιδράσεων με το υποσύστημα του μοντέλου Β. Πρακτικά, αυτό ισοδυναμεί με την εμφάνιση μεγάλου αριθμού αλλαγών κατάστασης (ή γεγονότων) του Α, μεταξύ δύο διαδοχικών αλλαγών κατάστασης του Β. Με τον τρόπο αυτό γίνεται μία αντιστοίχιση της διαίρεσης του χώρου καταστάσεων του συστήματος σε μία διάκριση της συμπεριφοράς του μέσα στο χρόνο. Διαχωρίζεται δηλαδή η *βραχυπρόθεσμη δυναμική* των υποσυστημάτων, από την *μακροπρόθεσμη δυναμική* των αλληλεπιδράσεών τους. Είναι σαφές ότι η συγκεκριμένη υπόθεση βρίσκει εφαρμογή στη μεγάλη πλειονότητα των μοντέλων απόδοσης των υπολογιστικών συστημάτων, καθώς κατά κανόνα, η χρήση των χαμηλού επιπέδου πόρων αυτών λαμβάνει χώρα σε ρυθμούς πολύ μεγαλύτερους από τη χρήση των πόρων υψηλότερου επιπέδου, που προσφέρουν για παράδειγμα υπηρεσίες σε επίπεδο λογισμικού.

Η διαπίστωση ότι αυτού του είδους η προσέγγιση αποδίδει σχετικά ακριβή αποτελέσματα, για μοντέλα υπολογιστικών συστημάτων, οδήγησε στην κατασκευή εργαλείων, που διευκολύνουν την ανάπτυξη ιεραρχικών μοντέλων απόδοσης. Ένα τέτοιο εργαλείο είναι και το HIT, που παρουσιάζεται πιο αναλυτικά στο κεφάλαιο 5, καθώς η μοντελική προσέγγιση που υποστηρίζει κρίθηκε ότι διευκολύνει την ανάπτυξη μοντέλων απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής.

Επιπλέον, η προσέγγιση με βάση τη σχεδόν ολοκληρωτική αποσυνθεσιμότητα είναι τόσο γενική, που καθιστά εφικτή και πολλές φορές συμφέρουσα μία *υβριδική αποτίμηση* των μοντέλων απόδοσης· μία αποτίμηση δηλαδή, που συνδυάζει αναλυτική, προσομοιωτική ή ακόμη και αριθμητική επίλυση των υποσυστημάτων του μοντέλου. Στην [SCH78] ο συγγραφέας πειραματίστηκε με ένα υβριδικό προσομοιωτικό/αναλυτικό μοντέλο απόδοσης δύο επιπέδων. Ο χρόνος CPU, που χρησιμοποιήθηκε για την αποτίμηση του μοντέλου ήταν από 18 έως και 200 φορές πιο μικρός σε σχέση με την αποκλειστικά προσομοιωτική έκδοση αυτού. Η προσέγγιση του μέσου χρόνου παραμονής των έργων, στο υποσύστημα του χαμηλότερου επιπέδου, δεν έδωσε σφάλμα μεγαλύτερο του 3%, αν και αυτό ήταν σαφές ότι παρουσίασε αύξηση, όταν οι αλληλεπιδράσεις μεταξύ των δύο επιπέδων ήταν πιο συχνές και άρα το σύστημα διέθετε σχετικά μειωμένη αποσυνθεσιμότητα.

Είναι όμως εφικτή η χρήση της προσομοίωσης αναγέννησης σε ένα ιεραρχικό μοντέλο απόδοσης; Η απάντηση στο κρίσιμο αυτό ερώτημα αναφέρεται σε δύο διαφορετικές περιπτώσεις χρήσης της τεχνικής:

- Η πρώτη περίπτωση αφορά τη συνολική προσομοιωτική αποτίμηση ενός ιεραρχικού μοντέλου απόδοσης αποτελούμενου από υποσυστήματα, τα οποία ανεξαρτήτως της τεχνικής επίλυσης αυτών ενσωματώνονται στο σύστημα με αντικατάσταση από κέντρο εξυπηρέτησης ισοδύναμης ροής (FESC). Η εξυπηρέτηση σε ένα τέτοιο κέντρο παρέχεται, όπως ήδη τονίστηκε στην ενότητα 3.1.4, σύμφωνα με την εκθετική ή άλλη κατανομή με παραμέτρους, που εξαρτώνται από τον αριθμό των έργων επεξεργασίας, που βρίσκονται στο υποσύστημα. Η ύπαρξη όμως κέντρου ισοδύναμης ροής θεωρητικά δεν αποκλείει τη χρήση της τεχνικής της αναγέννησης. Το γεγονός αυτό επιβεβαιώνεται ακριβώς επειδή ο ορισμός της υποκείμενης στοχαστικής διαδικασίας ([SHE93]), που διέπεται από την ιδιότητα της αναγέννησης, συμπεριλαμβάνει τη δυνατότητα

ύπαρξης κέντρων με εξυπηρέτηση εξαρτώμενη από την κατάσταση του συστήματος.

- Η δεύτερη περίπτωση αφορά την προσομοιωτική αποτίμηση υποσυστήματος χαμηλότερου επιπέδου με τη χρήση της τεχνικής της αναγέννησης και την ενσωμάτωση αυτού στο συνολικό μοντέλο, με αντικατάσταση από κέντρο ισοδύναμης ροής. Αυτό προϋποθέτει τη δυνατότητα εκτίμησης του μέσου χρόνου παραμονής των έργων στο υποσύστημα για την παροχή της αιτούμενης εξυπηρέτησης. Στο [SHE93] ορίζονται οι υποκείμενες στοχαστικές διαδικασίες, για τον υπολογισμό των μέσων χρόνων παραμονής των έργων σε κάποιο τμήμα ενός συστήματος και αποδεικνύεται η ισχύς της ιδιότητας της αναγέννησης και για αυτές. Επίσης, για την υλοποίηση της συγκεκριμένης εκτιμητικής διαδικασίας προτείνεται η τεχνική της σήμανσης των έργων επεξεργασίας.

Από τα παραπάνω, είναι σαφές ότι η τεχνική της αναγέννησης διαθέτει τη θεωρητική θεμελίωση για την ενσωμάτωσή της σε ένα περιβάλλον ανάπτυξης ιεραρχικών μοντέλων απόδοσης, όπως αυτό του HIT, που προς το παρόν διαθέτει μόνο την τεχνική της αυτοπαλινδρούμενης αναπαράστασης.

4.10 Ερευνητικές κατευθύνσεις

Η σε βάθος μελέτη της τεχνικής της αναγέννησης και η παραλληλοποίηση αυτής έφερε στο προσκήνιο πλήθος προβλημάτων αλλά και προοπτικών, που σίγουρα απαιτούν διεξοδικότερη διερεύνηση. Μερικά από αυτά είναι:

- Η συστηματική συγκριτική μελέτη των παραλλαγών της τεχνικής της αναγέννησης, που κατονομάζονται στην παράγραφο 4.7, σε συνδυασμό με διάφορες διαδικασίες συνεχούς ελέγχου σε διάταξη παράλληλου χρόνου. Μελέτη, η οποία θα αφορά όχι μόνο τα κέρδη σε επιτάχυνση των πειραμάτων προσομοίωσης, αλλά και την ποιότητα των αποτελεσμάτων αυτών. Πέρα από την πειραματική ανάλυση κάλυψης, θα μπορούσε να διατυπωθεί κάποιο κριτήριο στατιστικής αποδοτικότητας, όπως ο λόγος:

$$r = \frac{MSE[\overline{X}_{method-2}(t)]}{MSE[\overline{X}_{method-1}(t)]}$$

  και στη συνέχεια να γίνει συσχέτισή του με την επιτάχυνση της παράλληλης διάταξης. Αυτού του είδους η θεωρητική ανάλυση προτείνεται στην [HEI86] για τη συγκριτική μελέτη της τεχνικής των παράλληλων ανεξάρτητων επαναλήψεων.

- Η ανάπτυξη κατάλληλης διαδικασίας συνεχούς ελέγχου, που να λαμβάνει υπόψη το πρόβλημα της ταυτόχρονης πολλαπλής εκτίμησης. Το τελευταίο προκύπτει από την ανάγκη ταυτόχρονης ανάλυσης περισσοτέρων του ενός μέτρων απόδοσης, που βασικά παρουσιάζεται στη μεγάλη πλειοψηφία των περιπτώσεων πρακτικής εφαρμογής των τεχνικών προσομοίωσης. Στο λογισμικό, που περιγράφηκε στην παράγραφο 4.8, υλοποιήθηκε η συνήθης προσέγγιση του ανεξάρτητου υπολογισμού διαστημάτων εμπιστοσύνης του ιδίου ονομαστικού επιπέδου σημαντικότητας για κάθε μέτρο απόδοσης. Παρόλα αυτά, η συγκεκριμένη προσέγγιση δεν ανταποκρίνεται στο ότι το συνολικό επίπεδο σημαντικότητας, όλες οι μέσες τιμές να περιέχονται στα αντίστοιχα διαστήματα εμπιστοσύνης, είναι μικρότερο από το ονομαστικό επίπεδο σημαντικότητας του κάθε ενός από αυτά ξεχωριστά. Παρά το γεγονός ότι έχουν προταθεί διαδικασίες συνεχούς ελέγχου (όπως για παράδειγμα αυτές των [CK88] και [RAA93]) κατάλληλες για

επεξεργασία *πολυμεταβλητών αποτελεσμάτων προσομοίωσης*, καμία από αυτές δεν είναι σήμερα σε ευρεία χρήση. Η εφαρμογή παρόμοιας διαδικασίας και στην περίπτωση της τεχνικής της αναγέννησης σε παράλληλο χρόνο σίγουρα παρουσιάζει μεγάλο ενδιαφέρον.

- Η μελέτη της δυνατότητας εφαρμογής και των πιθανών οφελών από τη χρήση μεταβλητών ελέγχου για τον ταχύτερο περιορισμό της διασποράς σε προσομοιώσεις αναγέννησης παράλληλου χρόνου. Σχετικές εργασίες, που αφορούν την περίπτωση σειριακής προσομοίωσης είναι οι [LMW82], [IL79] και [LR91].
- Η διατύπωση πρακτικά χρήσιμων συνθηκών ύπαρξης της ιδιότητας της αναγέννησης και περισσότερο γενικών από αυτές, που περιγράφηκαν στην παράγραφο 4.7. Το πρόβλημα έχει αναθερμάνει το ερευνητικό ενδιαφέρον για την τεχνική της αναγέννησης, καθώς σχετίζεται και με τη διατύπωση μιας διαδικασίας αναγνώρισης όχι μίας τυχαίας, αλλά της περισσότερο πιθανής κατάστασης αναγέννησης.
- Η διερεύνηση των προοπτικών εφαρμογής της τεχνικής και στην προσομοίωση γενικευμένων στοχαστικών δικτύων Petri (GSPNs). Μία σημαντική ερευνητική αφετηρία για το σκοπό αυτό αποτελεί η [HS86].

Γενικά, η οποιαδήποτε βελτίωση της τεχνικής της αναγέννησης έχει μεγάλη πρακτική σημασία και για έναν ακόμη ιδιαίτερα σημαντικό λόγο: Η τεχνική αυτή είναι η μοναδική - λόγω της ανεξαρτησίας των παρατηρήσεων - μέσω της οποίας μπορούν να ληφθούν εκτιμήσεις για τις τιμές των παραγώγων των μέτρων απόδοσης, τις επονομαζόμενες *ευαισθησίες* (sensitivities). Με τη χρήση των εκτιμήσεων, που προκύπτουν ουσιαστικά από τα αποτελέσματα ενός μόνο πειράματος προσομοίωσης, είναι δυνατή η διενέργεια αναλύσεων ευαισθησίας και βελτιστοποιήσης. Καθώς βέβαια τα αποτελέσματα τέτοιων προσομοιώσεων δε δίνουν τις πραγματικές τιμές των παραγώγων αλλά μόνο εκτιμήσεις αυτών, στην περίπτωση της βελτιστοποίησης είναι επιβεβλημένη και η χρήση κάποιου *αλγορίθμου στοχαστικής προσέγγισης*, όπως για παράδειγμα του γνωστού αλγορίθμου Robbins - Monroe [RM51].

Για μία πιο αναλυτική περιγραφή των τεχνικών αυτών μπορεί κανείς να αναφερθεί στο [RM98] και στις [GLY87], [GLY90], [GLA91], [GG92] και [RW86].

Κεφάλαιο 5 *Μοντελοποίηση Απόδοσης Συστημάτων Κατανεμημένης Αρχιτεκτονικής*

Στο κεφάλαιο αυτό εισάγεται η προσέγγιση, που επιλέχθηκε ως η πλέον κατάλληλη για την ανάπτυξη μοντέλων απόδοσης συστημάτων κατανεμημένης αρχιτεκτονικής. Η συγκεκριμένη προσέγγιση προσδιορίζει με σαφή τρόπο την τεχνική ορισμού ενός ιεραρχικού μοντέλου απόδοσης αποτελούμενου από τμήματα και μπορεί να υποστηριχθεί από το εργαλείο HIT. Η περιγραφή των διαδικασιών της συγκρότησης, του κύκλου ζωής και των διαφορετικών τύπων ανάλυσης ενός μοντέλου συνιστά μία *μέθοδο μοντελοποίησης*. Οι διαδικασίες, που προτείνονται στο κεφάλαιο αυτό μπορούν να αποτελέσουν τη βάση για τη θεμελίωση μιας μεθόδου, η οποία θα χαρακτηρίζεται από τη δυνατότητα περιγραφής της απόδοσης του λογισμικού σε περισσότερα του ενός επίπεδα αφαίρεσης. Τέλος, γίνεται χρήση της μοντελικής προσέγγισης, που επιλέχθηκε για την περιγραφή μοντέλων απόδοσης περιπτώσεων λογισμικού κατανεμημένης αρχιτεκτονικής.

5.1 Ιεραρχική μοντελοποίηση - αποτίμηση της απόδοσης λογισμικού στο HIT

Παραδοσιακά, όταν η μελέτη της απόδοσης υπολογιστικών συστημάτων γινόταν με τη χρήση δικτύων ουρών, τα κέντρα εξυπηρέτησης, που συγκροτούσαν το δίκτυο αναπαριστούσαν τους πόρους υλικού του συστήματος. Η επίδραση του λογισμικού ενσωματωνόταν με την προσαρμογή των ρυθμών εξυπηρέτησης των κέντρων του δικτύου, έτσι ώστε αυτοί να αντανακλούν τον υπολογιστικό φόρτο, που συνεπάγεται η επεξεργασία του προς μελέτη λογισμικού, όταν αυτό λειτουργεί σε μία συγκεκριμένη διαμόρφωση (λογισμικό συστήματος). Με την αύξηση της πολυπλοκότητας του λογισμικού και τη συχνή αλληλεπίδρασή του με άλλου είδους λογισμικό, η κατασκευή και η παραμετροποίηση μοντέλων ουρών ήταν μία εξαιρετικά περίπλοκη διαδικασία. Η ποιότητα των αποτελεσμάτων της διαδικασίας αυτής εξαρτιόταν από τις ικανότητες και την εμπειρία του αναλυτή. Επίσης, η προσαρμογή και επαναχρησιμοποίηση των αποτελεσμάτων μιας μελέτης σε περιπτώσεις διαφορετικής αρχιτεκτονικής διαμόρφωσης του λογισμικού ή διαφορετικού φόρτου ήταν αδύνατη.

Η πρώτη συστηματική προσέγγιση ήταν αυτή, που πρωτοπαρουσιάστηκε από την Connie Smith στο [SMI90]. Στη συγκεκριμένη προσέγγιση η ροή της εκτέλεσης του λογισμικού,

μαζί με τις απαιτήσεις εξυπηρέτησης στους πόρους υλικού του συστήματος, μοντελοποιούνται από τα *διαγράμματα εκτέλεσης λογισμικού*. Ακολούθως, οι πληροφορίες, που απεικονίζονται στα διαγράμματα αυτά, χρησιμοποιούνται για την κατασκευή και την παραμετροποίηση των *μοντέλων εκτέλεσης συστήματος*. Εκεί το σύστημα περιγράφεται ως ένα δίκτυο από ουρές με διαφόρους τύπους κέντρων εξυπηρέτησης για αυξημένη ευελιξία μοντελοποίησης, ακόμη και αν αυτή, συνήθως, δεν οδηγεί σε μοντέλα μορφής γινομένου.

Η πρώτη προσπάθεια άμεσης ενσωμάτωσης μονάδων λογισμικού και απαιτήσεων εξυπηρέτησης στο ίδιο δίκτυο ουρών καταγράφεται το 1982 στην [AT82]. Στη συγκεκριμένη προσέγγιση, τα μοντέλα βασίζονται σε μία περιγραφή δύο επιπέδων, το επίπεδο λογισμικού και το επίπεδο υλικού. Τα κέντρα εξυπηρέτησης του δικτύου ουρών αναπαριστούν μονάδες λογισμικού, οι οποίες είτε μπορεί να έχουν τη δυνατότητα ταυτόχρονης εξυπηρέτησης περισσοτέρων του ενός χρηστών, είτε όχι. Ενώ στην πρώτη περίπτωση υιοθετείται πειθαρχία PS στο κέντρο εξυπηρέτησης της εν λόγω μονάδας λογισμικού, στη δεύτερη περίπτωση η πειθαρχία, που χρησιμοποιείται, είναι FCFS. Στο άλλο επίπεδο, αυτό δηλαδή του υλικού, κάθε μονάδα λογισμικού δημιουργεί απαιτήσεις εξυπηρέτησης σε κάποιο πόρο από αυτούς, που είναι διαθέσιμοι στο σύστημα. Το μοντέλο, που τελικά προκύπτει με την εφαρμογή της προτεινόμενης τεχνικής παραμετροποίησης, είναι ένα δίκτυο με έργα περισσοτέρων του ενός τύπων και με ρυθμούς εξυπηρέτησης ανάλογα με την κατάσταση του συστήματος. Δίκτυο δηλαδή, που γενικά δεν είναι μορφής γινομένου και η επίλυσή του μπορεί να γίνει μόνο με κάποια προσεγγιστική τεχνική προσομοίωσης ή κάποια αναλυτική τεχνική, ανάλογη αυτής, που εφαρμόζεται στην αναφερόμενη εργασία. Πάντως, ένας σημαντικός περιορισμός της προτεινόμενης τεχνικής είναι το γεγονός ότι κάθε μονάδα λογισμικού μπορεί να απευθύνει απαιτήσεις εξυπηρέτησης σε όχι περισσότερους από έναν πόρο υλικού. Επίσης, δε φαίνεται να έγινε καμία μεταγενέστερη προσπάθεια βελτίωσής της, καθώς από την αρχή δόθηκε έμφαση στη συνεισφορά της στην αποτίμηση της διαφοράς στην απόδοση λογισμικού ταυτόχρονης εξυπηρέτησης (reentrant), σε σχέση με αντίστοιχο λογισμικό σειριακής εξυπηρέτησης (nonreentrant).

Μία διαφορετική προσέγγιση στην αναπαράσταση της αλληλεπίδρασης μεταξύ μονάδων λογισμικού και του ανταγωνισμού για τους διαθέσιμους πόρους υλικού, που αυτή συνεπάγεται, είναι τα *στρωματοποιημένα δίκτυα ουρών* (Layered Queueing Network), που περιγράφονται εκτενώς στην παράγραφο 3.4. Χαρακτηριστικό της συγκεκριμένης αναπαράστασης είναι το γεγονός ότι μία μονάδα λογισμικού μπορεί να λειτουργεί ως πελάτης κάποιων μονάδων λογισμικού ή συσκευών υλικού και ταυτόχρονα να παρέχει εξυπηρέτηση σε άλλες μονάδες λογισμικού.

Η προσέγγιση, που προκρίνεται από τη μελέτη αυτή ως η πλέον κατάλληλη για τη μοντελοποίηση - αποτίμηση της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής έχει αρκετές ομοιότητες με τα στρωματοποιημένα δίκτυα ουρών. Ειδικώτερα, διευκολύνει την ιεραρχική από πάνω προς τα κάτω (top-down) ή από κάτω προς τα πάνω (bottom-up) ανάπτυξη μοντέλων απόδοσης, με δυνατότητες οριζόντιας και κάθετης αποκρυστάλλωσης των λεπτομερειών σχεδίασης του συστήματος. Με τον τρόπο αυτό υποστηρίζει ουσιαστικά μία σταδιακή διαδικασία ανάπτυξης μοντέλων απόδοσης· μία διαδικασία δηλαδή, ανάλογη αυτών των περισσότερων σύγχρονων μεθόδων ανάπτυξης λογισμικού.

Η συγκεκριμένη προσέγγιση έχει αποτελέσει τη φιλοσοφία σχεδίασης του εργαλείου HIT ([BS87], [BMW88], [BEI89] και [WWM00]). Η ανάπτυξη του HIT αρχικά αποσκοπούσε στην αποτίμηση της απόδοσης υπολογιστικών συστημάτων. Παρόλα αυτά και εξαιτίας της ευελιξίας της μοντελικής προσέγγισης, που υποστηρίζει το HIT έχει χρησιμοποιηθεί και στην ανάλυση συστημάτων γραφείου, συστημάτων επικοινωνιών, μεταφορών και δικτύων λογιστικής υποστήριξης (logistics).

Οι *υπηρεσίες*, που παρέχονται από τα *τμήματα* από τα οποία αποτελείται ένα μοντέλο απόδοσης στο HIT, περιγράφονται με ένα διαδικασιακό - αλγοριθμικό τρόπο. Έτσι, συνδυάζεται η εκφραστική ευελιξία της περιγραφής μιας ροής εκτέλεσης (όπως στα διαγράμματα εκτέλεσης λογισμικού της προσέγγισης του [SMI90]) και η ευκολία πρόσληψης και κομψότητα στην αναπαράσταση του ανταγωνισμού για χρήση των πόρων (υλικού και λογισμικού) ενός δικτύου ουρών. Όλα αυτά υποστηρίζονται από τη χρήση μιας υψηλού επιπέδου περιγραφικής γλώσσας γνωστής ως HI-SLANG. Το HIT βέβαια διαθέτει και ένα περιβάλλον γραφικής περιγραφής - ανάπτυξης μοντέλων, το επονομαζόμενο HITGRAPHIC.

Εδώ, τα ενδεικτικά μοντέλα περιπτώσεων λογισμικού κατανεμημένης αρχιτεκτονικής, που παρουσιάζονται στην παράγραφο 5.3, περιγράφονται γραφικά με τη χρήση δικτύων ουρών με κέντρα εξυπηρέτησης, που εκφράζουν λειτουργικότητα λογισμικού (όπως στο [AT82]).

Στο σημείο αυτό είναι σημαντικό να γίνει σύντομη παράθεση των τεχνικών αποτίμησης των μοντέλων απόδοσης, που υποστηρίζει το HIT. Αυτές είναι:

- η στοχαστική προσομοίωση διακριτών γεγονότων με στατιστική επεξεργασία αποτελεσμάτων,
- η ακριβής αποτίμηση για μοντέλα μορφής γινομένου και διάφορες προσεγγιστικές τεχνικές για «μεγάλα» μοντέλα ή μοντέλα, που δεν είναι μορφής γινομένου,
- η αριθμητική επίλυση αλυσίδας Markov για μοντέλα γενικής μορφής και
- η ανάλυση υπομοντέλων με στόχο τη δημιουργία υψηλότερου επιπέδου «ισοδύναμης» αναπαράστασης για χρήση σε συνολική ετερογενή (π.χ. αναλυτική και προσομοιωτική) αποτίμηση του μοντέλου.

Το HIT υποστηρίζει αποτίμηση μέτρων απόδοσης ανά τμήμα σε οποιοδήποτε επίπεδο της ιεραρχίας του μοντέλου, γεγονός που συμβάλλει λόγου χάρη στον εντοπισμό σημείων συμφόρησης λογισμικού ή υλικού. Ακόμη, το HIT υλοποιεί το πολύ χρήσιμο και αξιόλογο χαρακτηριστικό της *ιεραρχίας φιλτραρίσματος φόρτου* (load filtering hierarchy). Έτσι, οι κλήσεις εξυπηρέτησης, που προέρχονται από τα ανώτερα στρώματα και προκαλούν φόρτο επεξεργασίας σε τμήματα των κατωτέρων στρωμάτων, μπορούν να αποτιμηθούν και ξεχωριστά.

5.1.1 Μηχανές, μοντέλα επεξεργασίας, στρώματα, μοντέλα, τμήματα και υπηρεσίες

Στο HIT, σε κάθε στρώμα της ιεραρχίας ενός μοντέλου απόδοσης, συνυπάρχουν μία *μηχανή* και ένα *μοντέλο επεξεργασίας*, που αντιπροσωπεύει τις απαιτήσεις, που απευθύνονται στη μηχανή. Η μηχανή συγκροτείται από ένα σύνολο *τμημάτων* κάθε ένα από τα οποία παρέχει συγκεκριμένες *υπηρεσίες*. Το σύνολο των υπηρεσιών όλων των τμημάτων ενός στρώματος αποτελούν τη βάση για την ανάπτυξη ενός νέου στρώματος του μοντέλου (bottom-up model development). Ο φόρτος του μοντέλου απόδοσης αποτελείται από ένα σύνολο *διεργασιών*. Κάθε διεργασία δεν είναι τίποτε άλλο από την περιγραφή της δυναμικής χρήσης κάποιας ή κάποιων παρεχομένων υπηρεσιών. Η δημιουργία των διεργασιών του φόρτου μπορεί να γίνει είτε με βάση το χρόνο προσομοίωσης είτε με βάση την εμφάνιση συγκεκριμένων γεγονότων.

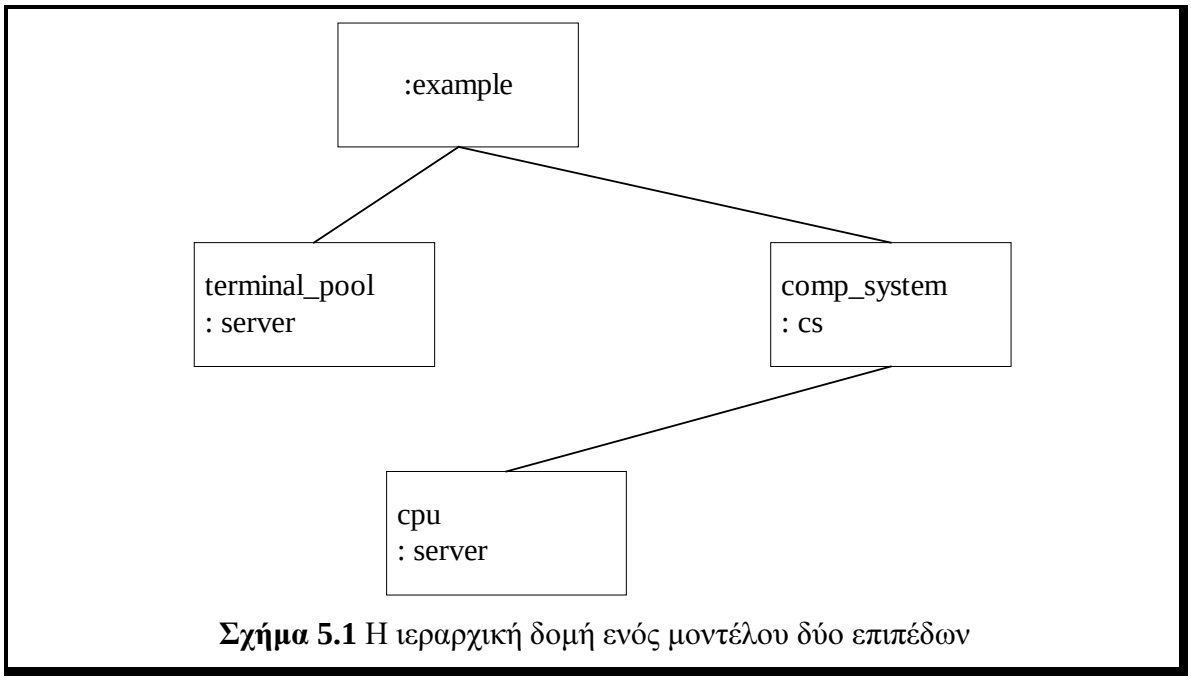
Τα τμήματα, που χρησιμοποιούνται, δηλώνονται ως στοιχεία συγκεκριμένων *τύπων τμημάτων* (component types). Το HIT διαθέτει κάποιους προκαθορισμένους τύπους τμημάτων, σημαντικότερος των οποίων είναι ο τύπος `server`, που παρέχει τη στοιχειώδη υπηρεσία `request (amount: real)`. Η παράμετρος `amount` βεβαίως συμβολίζει τη χρονική διάρκεια παροχής της υπηρεσίας `request`.

Αν μία μηχανή αποτελείται αποκλειστικά από αντικείμενα του τύπου `server`, τότε έχουμε ως αποτέλεσμα ένα κλασσικό «επίπεδο» μοντέλο απόδοσης. Κάθε μοντέλο μπορεί να

μετατραπεί σε τμήμα, αν κάποιες από τις υπηρεσίες, που ορίζονται εσωτερικά, δηλωθούν ως εξωτερικά προσπελάσιμες. Έτσι, δημιουργείται η βάση για την ανάπτυξη ενός νέου στρώματος και κατά συνέπεια ενός ιεραρχικού μοντέλου απόδοσης.

5.1.2 Παράδειγμα ιεραρχικού μοντέλου απόδοσης

Στην ενότητα αυτή χρησιμοποιείται ένα από τα παραδείγματα του [WWM00], αυτό δηλαδή που επιλέχθηκε ως το πλέον κατάλληλο για μία αρχική εξοικείωση με τη σε περισσότερα του ενός στρώματα ανάπτυξη ενός μοντέλου απόδοσης. Το μοντέλο του παραδείγματος αποτελείται από ένα τμήμα `terminal_pool`, που εκφράζει τα τερματικά ενός συστήματος, το οποίο αναπαριστάται από το τμήμα `comp_system` (σχήμα 5.1).



Σχήμα 5.1 Η ιεραρχική δομή ενός μοντέλου δύο επιπέδων

Το μοντέλο επεξεργασίας περιγράφεται από δύο διαφορετικούς τύπους υπηρεσιών· τις `cmd1` και `cmd2`, τις οποίες οι χρήστες ενεργοποιούν μετά από κάποιο χρόνο προσμονής (think time). Το τμήμα `terminal_pool` αναπαριστάται από τον προκαθορισμένο τύπο τμημάτων `server`, ενώ το τμήμα `comp_system` αναπαριστάται από τμήμα του οριζόμενου από τον αναλυτή τύπου `cs`, που διαθέτει τις υπηρεσίες `cmd1_processing` και `cmd2_processing` ως εξωτερικά προσβάσιμες.

Ο HI-SLANG κώδικας του τύπου τμημάτων `cs` με τη μορφή που αυτός εικονίζεται στην ιεραρχία του σχήματος 5.1 έχει ως εξής:

```
TYPE cs COMPONENT;
      PROVIDE
            SERVICE
             cmd1_processing;
             cmd2_processing;
      END PROVIDE;

      TYPE cmd1_processing SERVICE;
            USE SERVICE
             compute (m: REAL);
            END USE;
      BEGIN
            AVERAGE 10 TIMES
            LOOP
             compute (negexp(1/0.045));
```

```
            END LOOP;
      END TYPE cmd1_processing;

      TYPE cmd2_processing SERVICE;
            USE SERVICE
             compute (m: REAL);
            END USE;
      BEGIN
            AVERAGE 20 TIMES
            LOOP
             compute (negexp(1/0.135));
            END LOOP;
      END TYPE cmd2_processing;

      COMPONENT cpu: server  (LET schedule:= immediate,
                              LET dispatch:= shared);

      REFER cmd1_processing, cmd2_processing TO cpu
      EQUATING
       cmd1_processing.compute WITH cpu.request;
       cmd2_processing.compute WITH cpu.request;
      END REFER;

      END TYPE cs;
```

Ο τύπος μοντέλου `example` με τις δύο υπηρεσίες `cmd1` και `cmd2`, που περιγράφουν το μοντέλο επεξεργασίας του συστήματος, φαίνεται στο πλαίσιο που ακολουθεί. Οι παράμετροι `n1` και `n2` αντιπροσωπεύουν τους αριθμούς των υπηρεσιών, που συνθέτουν το φόρτο του συστήματος.

```
TYPE example MODEL (n1, n2:INTEGER);
      TYPE cmd1 SERVICE;
            USE SERVICE
             think (thinktime : REAL);
             run;
            END USE;
            BEGIN
            LOOP
             think (negexp(1/5));
             run;
            END LOOP;
      END TYPE cmd1;

      TYPE cmd2 SERVICE;
            USE SERVICE
             think (thinktime : REAL);
             run;
            END USE;
            BEGIN
            LOOP
             think (negexp(1/10));
             run;
            END LOOP;
      END TYPE cmd2;

      COMPONENT
            terminal_pool: server
                       ( LET accept := always,
                         LET schedule := immediate,
                         LET dispatch := equal,
                         LET offer := all);

      COMPONENT
            comp_system: cs
```

```
                    ( LET accept := always,
                      LET offer := all);

      REFER cmd1, cmd2 TO terminal_pool, comp_system
      EQUATING
       cmd1.think WITH terminal_pool.request;
       cmd1.run WITH comp_system.cmd1_processing;
       cmd2.think WITH terminal_pool.request;
       cmd2.run WITH comp_system.cmd2_processing;
      END REFER;

      BEGIN
       CREATE n1 PROCESS cmd1;
       CREATE n2 PROCESS cmd2;
      END TYPE example;
```

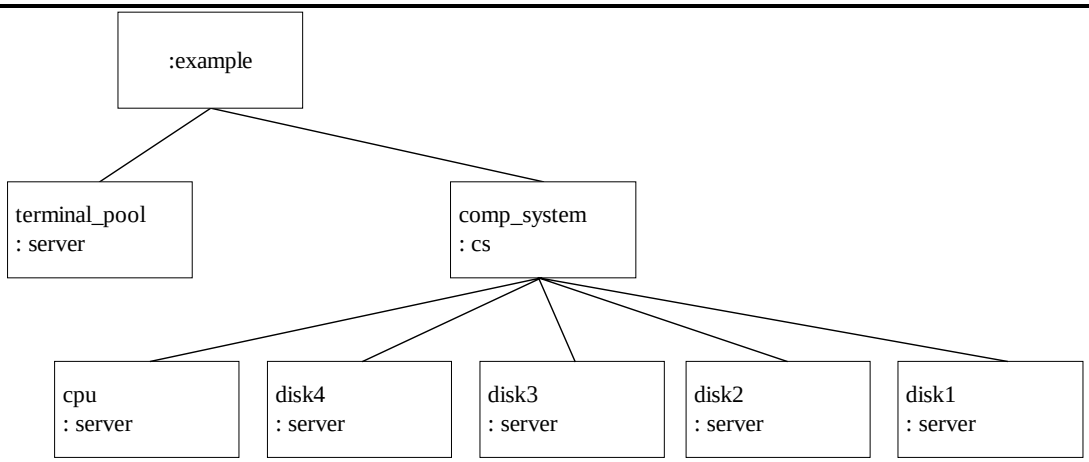
Στο πλαίσιο που ακολουθεί, φαίνονται οι εντολές περιγραφής μιας αναλυτικής αποτίμησης του μέσου αριθμού διεργασιών (processes) και του χρόνου απόκρισης του τμήματος `computer:comp_system`.

```
EXPERIMENT experiment1 METHOD ANALITICAL"DOQ4";
BEGIN
  EVALUATE MODEL model1:example(20,2);

    EVALUATIONOBJECT
      computer VIA model1.comp_system;

    BEGIN
      MEASURE POPULATION, TURNAROUNDTIME
            AT computer;
    END EVALUATE;

END EXPERIMENT experiment1;
```

Όπως φαίνεται, στη HI-SLANG ο αναλυτής περιγράφει το προς εκτέλεση πείραμα, ανεξάρτητα από το όλο μοντέλο απόδοσης. Το τμήμα λοιπόν αυτό των εντολών περιλαμβάνει:

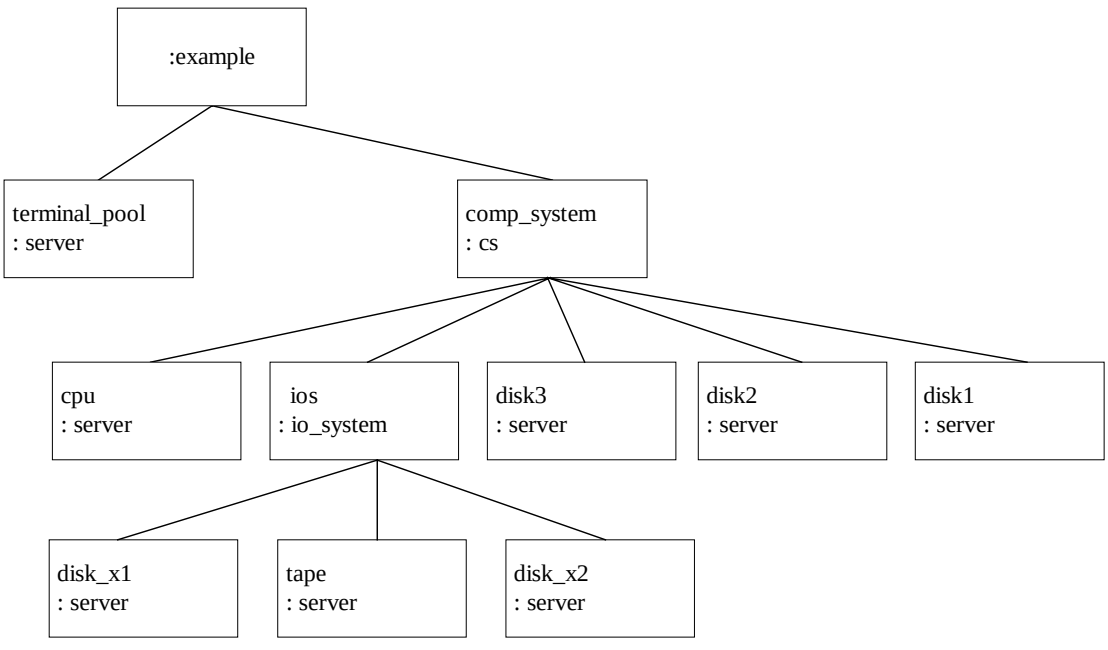
- Δημιουργία μιας περίπτωσης μοντέλου από έναν τύπο μοντέλου, που έχει περιγραφεί νωρίτερα, συμπεριλαμβανομένης και της παραμετροποίησης αυτού.
- Προσδιορισμός της τεχνικής ανάλυσης του μοντέλου.
- Προσδιορισμός των προς αποτίμηση αντικειμένων, των τμημάτων δηλαδή του μοντέλου, όπου είναι επιθυμητό να γίνουν μετρήσεις.
- Προσδιορισμός των *ροών μέτρησης*· των μέτρων απόδοσης δηλαδή για τα οποία είναι επιθυμητό να γίνουν μετρήσεις στα αντικείμενα αποτίμησης, που ορίσθηκαν.
- Λεπτομέρειες της αποτίμησης, αν χρειάζονται.
- Προσδιορισμός κανόνων συλλογής δεδομένων (έναρξη και λήξη)· στην περίπτωση που επιλέγεται η προσομοίωση ως η τεχνική αποτίμησης του μοντέλου.

Στο σχήμα 5.2 απεικονίζεται η νέα ιεραρχία τμημάτων, που προκύπτει από μια οριζόντια αποκρυστάλλωση λεπτομερειών του μοντέλου του σχήματος 5.1. Όπως φαίνεται, μόνο ο ορισμός του τύπου τμήματος `CS` αλλάζει, ενώ όλοι οι άλλοι ορισμοί παραμένουν ίδιοι.



Σχήμα 5.2 Οριζόντια αποκρυστάλλωση λεπτομερειών του μοντέλου του σχήματος 5.1

Τέλος, στο σχήμα 5.3 απεικονίζεται το αποτέλεσμα μιας κάθετης αποκρυστάλλωσης λεπτομερειών του τμήματος `cs`. Είναι σημαντικό να τονιστεί ότι σε καμία περίπτωση δεν επηρεάζονται οι ορισμοί των τύπων τμημάτων ή του τύπου του μοντέλου, που ιεραρχικά βρίσκονται σε υψηλότερα στρώματα. Επίσης, δεν επηρεάζεται το τμήμα εντολών, που περιγράφει το πείραμα, αν φυσικά τα αντικείμενα αποτίμησης παραμένουν σε υψηλότερα στρώματα από αυτά στα οποία λαμβάνει χώρα η αποκρυστάλλωση λεπτομερειών.



Σχήμα 5.3 Κάθετη αποκρυστάλλωση λεπτομερειών του μοντέλου του σχήματος 5.2

5.2 Διαδικασίες μοντελοποίησης της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής

Στην προηγούμενη παράγραφο, έγινε περιγραφή μιας ιεραρχικής μοντελικής προσέγγισης, με χαρακτηριστικά σταδιακής ανάπτυξης μοντέλων με τμηματική αποκρυστάλλωση των επί μέρους λεπτομερειών σχεδίασης. Στην παράγραφο αυτή περιγράφονται διαδικασίες, οι οποίες εκμεταλλεύονται τη συγκεκριμένη μοντελική προσέγγιση για την ανάπτυξη αξιόπιστων μοντέλων απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής. Η συστηματοποίηση των διαδικασιών αυτών μπορεί να αποτελέσει τη βάση για τη θεμελίωση μιας *μεθόδου ανάπτυξης μοντέλων* ή πιο σύντομα μιας *μεθόδου μοντελοποίησης*. Η συγκεκριμένη μέθοδος θα υποστηρίζει τη χρήση περισσοτέρων του ενός μοντέλων του ιδίου συστήματος, σε διαφορετικά όμως επίπεδα αφαίρεσης. Ο τρόπος εκμετάλλευσης των μοντέλων αυτών μπορεί πάντα να ισορροπεί τις αντικρουόμενες ανάγκες για υψηλή αξιοπιστία και χαμηλό κόστος στην αποτίμηση της απόδοσης εναλλακτικών περιπτώσεων σχεδίασης.

Σημαντικό ρόλο στη διαμόρφωση των συγκεκριμένων διαδικασιών έπαιξαν οι ιδέες, που αναπτύσσονται στις [KD80] και [SAD00]. Στην [KD80] οι συγγραφείς εισάγουν την ιδέα της ανάπτυξης πολλαπλών μοντέλων σε διαφορετικά επίπεδα αφαίρεσης και με διαφορετικές τεχνικές, για τη σχεδίαση ενός υπολογιστικού συστήματος. Στα υψηλά επίπεδα αφαίρεσης μπορούν για παράδειγμα να χρησιμοποιηθούν μοντέλα χαμηλού κόστους αποτίμησης, είτε αυτά είναι (συνήθως αναλυτικά επιλύσιμα) δίκτυα ουρών, είτε μεταμοντέλα άλλων μοντέλων. Στα χαμηλά επίπεδα αφαίρεσης, αναπτύσσονται περισσότερο λεπτομερή αλλά μεγαλύτερου κόστους προσομοιωτικά μοντέλα απόδοσης. Η μελέτη αυτών αποδίδει αποτελέσματα, που χρησιμοποιούνται για την προσαρμογή - παραμετροποίηση (calibration) των μοντέλων των υψηλότερων επιπέδων.

Στη [SAD00] οι συγγραφείς αναφέρονται στα αποτελέσματα του ευρωπαϊκού προγράμματος ESPRIT υπό την επωνυμία HELIOS. Το HELIOS είχε ως σκοπό την προσαρμογή του εργαλείου SMART, το οποίο προϋπήρχε, σε προβλήματα ανάλυσης της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής βασισμένης σε τεχνολογίες CORBA, DCOM και OLE-DB ή PL/SQL ως λύση πρόσβασης σε δεδομένα. Είναι χαρακτηριστικό το γεγονός ότι για την περιγραφή μοντέλων απόδοσης εισάγεται ([SP98], [SAP98]) μία νέα προσέγγιση με την επωνυμία HOOMA (Hierarchical Object-Oriented Modeling Approach), η οποία παρουσιάζει εκπληκτικές αναλογίες με την ιεραρχική μοντελοποίηση, που περιγράφεται στην παράγραφο 5.1.

5.2.1 Συγκρότηση μοντέλων απόδοσης

Η ανάπτυξη ενός μοντέλου απόδοσης για λογισμικό κατανεμημένης αρχιτεκτονικής αποτελείται από τα στάδια καθορισμού (πίνακας 5.1): i) του στόχου - πλαίσιο της απόδοσης, ii) της αρχιτεκτονικής, iii) της εφαρμογής, iv) της διαμόρφωσης και v) του φόρτου του συστήματος. Κάθε μοντέλο απόδοσης, είτε αυτό περιγράφει μόνο την αρχιτεκτονική, είτε περιγράφει και την εφαρμογή αλλά και τη διαμόρφωση του συστήματος, συγκροτείται από ένα *μοντέλο φόρτου*, ένα *μοντέλο επεξεργασίας* και ένα *μοντέλο ανταγωνισμού*.

Μοντέλα ανταγωνισμού για χρήση των διαθέσιμων πόρων είναι τα δίκτυα ουρών. Όσον αφορά το φόρτο, σχετική επεξήγηση γίνεται στο αντίστοιχο τμήμα του πίνακα 5.1. Τέλος, σε ένα μοντέλο επεξεργασίας, περιγράφονται οι απαιτήσεις εξυπηρέτησης σε υπολογιστικούς πόρους ανά έργο ή κλήση εξυπηρέτησης ή διεκπεραίωση.

Μοντέλο Απόδοσης	=	***Μοντέλο Φόρτου***
	+	***Μοντέλο Επεξεργασίας***
	+	***Μοντέλο Ανταγωνισμού***

Πίνακας 5.1 Στάδια ανάπτυξης μοντέλου απόδοσης κατανεμημένης αρχιτεκτονικής

Στόχος - πλαίσιο απόδοσης συστήματος	Είναι τα κριτήρια αποτίμησης της απόδοσης του συστήματος. Η επιλογή ανάμεσα στο χρόνο απόκρισης ή την παραγωγή εξαρτάται από το αν αυτό που ενδιαφέρει περισσότερο είναι η ταχύτητα εξυπηρέτησης ή η αποδοτικότητα στην παροχή της συγκεκριμένης υπηρεσίας. Συχνά επιλέγεται μία εξισορρόπηση μεταξύ των δύο μέτρων απόδοσης, όπως π.χ. η παροχή 55 εξυπηρετήσεων το δευτερόλεπτο με χρόνο απόκρισης λιγότερο του ενός δευτερολέπτου. Οι στόχοι χρόνων απόκρισης μπορεί να αφορούν συγκεκριμένες επεξεργασίες ή συνολικούς (end-to-end) χρόνους, όταν πρόκειται για εργασίες που περιλαμβάνουν περισσότερες της μιας επεξεργασίας. Μπορεί επίσης να τίθενται στόχοι απόδοσης βάσει υποθέσεων, που αφορούν λ.χ. τον αριθμό των χρηστών ή τη σύνθεση του φόρτου ενός συχνά εμφανιζόμενου σεναρίου λειτουργίας.
Αρχιτεκτονική	Ο αναλυτής ορίζει τη διάταξη και τον τρόπο με τον οποίο αλληλεπιδρούν τα τμήματα υλικού και λογισμικού, που συνθέτουν το περιβάλλον, στο οποίο θα εκτελείται η εφαρμογή.
Εφαρμογή	Ο αναλυτής προχωρά στην περιγραφή λειτουργιών της εφαρμογής με τη μορφή επεξεργασίας δεδομένων, που προκαλεί χρήση και παροχή υπηρεσιών εξυπηρέτησης. Είναι σημαντικό να τονιστεί ότι η μοντελοποίηση της εφαρμογής δεν αλληλεπιδρά με τη μοντελοποίηση της αρχιτεκτονικής πέρα από τη χρήση των υπηρεσιών, που παρέχονται από τα τμήματα αυτής, τα οποία άλλωστε μπορούν να προϋπάρχουν ή να αναπτύσσονται παράλληλα.
Διαμόρφωση	Η αντιστοίχηση των τμημάτων της εφαρμογής σε αυτά του περιβάλλοντος, στο οποίο εκτελείται ονομάζεται διαμόρφωση.
Φόρτος	Ο αναλυτής ορίζει σενάρια λειτουργίας που τον ενδιαφέρουν. Κάθε σενάριο λειτουργίας εισάγει κάποιο φόρτο επεξεργασίας στην εφαρμογή, η οποία μελετάται κάτω από τη συγκεκριμένη διαμόρφωση. Το ζητούμενο είναι η απόδοση τιμών σε μεταβλητές εισόδου του μοντέλου, όπως ο χρόνος προσμονής, ο αριθμός χρηστών - πελατών ή ο ρυθμός άφιξης, ανάλογα με τον τύπο του υπολογιστικού φόρτου, όπως αυτοί ορίζονται στην παράγραφο 3.1.

Τα βήματα ανάπτυξης ενός μοντέλου επεξεργασίας, όπως αυτά καθορίζονται στη [HL84], είναι:

- Επιλογή των αντικειμένων επεξεργασίας, των οποίων είναι επιθυμητό να γίνει χαρακτηρισμός απαιτήσεων, π.χ. διεκπεραιώσεων, λειτουργιών ή έργων.
- Επιλογή των παραμέτρων, που χαρακτηρίζουν κάθε αντικείμενο επεξεργασίας. Οι παράμετροι αυτοί μπορεί να είναι απαιτήσεις πόρων υλικού, όπως π.χ. αριθμός μικροεντολών CPU ή χρόνου CPU, απαιτήσεις χώρου μνήμης, προσπελάσεων δίσκου κ.α. Μπορεί όμως να είναι και απαιτήσεις πόρων λογισμικού, όπως για παράδειγμα ο αριθμός μηνυμάτων δικτύου ή κλήσεων σε κάποιο διακομιστή βάσεων δεδομένων.
- Μετρήσεις επεξεργασίας, που γίνονται για τη λήψη τιμών παραμέτρων για κάθε αντικείμενο επεξεργασίας.
- Διερευνητική ανάλυση δεδομένων, όπου εξετάζονται προσεκτικά δείγματα τιμών παραμέτρων, κατανομές αυτών και απομονωμένες τιμές (outliers).

- Ελαχιστοποίηση του μεγέθους του μοντέλου επεξεργασίας (cluster analysis), όπου ο σκοπός είναι ο περιορισμός των αντικειμένων επεξεργασίας με παρόμοιες τιμές παραμέτρων.

Ο βαθμός λεπτομέρειας, στην επιλογή των αντικειμένων επεξεργασίας, καθορίζει το βαθμό της λεπτομέρειας των μέτρων απόδοσης, που προκύπτουν από την αποτίμηση του μοντέλου ανταγωνισμού. Έτσι, αν για παράδειγμα τα αντικείμενα επεξεργασίας επιλεγούν να είναι οι διεκπεραιώσεις, τότε τα αποτελέσματα του μοντέλου ανταγωνισμού θα εκφράζονται σε σχέση με αυτές.

Στο HIT, τα τμήματα συγκροτούν το μοντέλο ανταγωνισμού και οι υπηρεσίες, που αυτά παρέχουν και χρησιμοποιούν, εκφράζουν το μοντέλο επεξεργασίας. Η δυνατότητα οριζόντιας και κάθετης αποκρυστάλλωσης των λεπτομερειών σχεδίασης των τμημάτων οδηγεί όχι μόνο σε μία ιεραρχία τμημάτων, αλλά και σε μία ιεραρχία αντικειμένων επεξεργασίας. Η σύνθετη αυτή μορφή του μοντέλου επεξεργασίας επιτρέπει τη χρήση του με μοντέλα ανταγωνισμού διαφορετικών βαθμών λεπτομέρειας.

Επιπλέον, μία μοντελική προσέγγιση, όπως αυτή του HIT, όπου το «δέσιμο» των τμημάτων ενός μοντέλου δεν είναι άλλο από τις υπηρεσίες, που παρέχονται και χρησιμοποιούνται, διευκολύνει την αντιστοίχηση των τμημάτων της εφαρμογής σε ένα διαφορετικό περιβάλλον (αρχιτεκτονική) δίχως να χρειάζεται αυτά να αλλάξουν. Τότε λέμε ότι το σύστημα μελετάται σε διαφορετικές περιπτώσεις διαμόρφωσης. Η ανάπτυξη ενός μοντέλου απόδοσης γίνεται σύνθεση από προϋπάρχοντα τμήματα, τα οποία επαναχρησιμοποιούνται. Έτσι αναδεικνύεται η σημαντικότητα της ύπαρξης και συντήρησης μιας αξιόλογης *μοντελικής βάσης*, η οποία συμβάλλει με τη χρήση των τμημάτων, που αυτή περιλαμβάνει, στη μείωση του κόστους της μελέτης της απόδοσης ενός συστήματος.

5.2.2 Κύκλος ζωής μοντέλων

Ο κύκλος ζωής κάθε μοντέλου περνάει επαναληπτικά από τις ακόλουθες φάσεις:

- *Επιλογή περιοχής πειραματικής μελέτης*: Χαρακτηρίζει την περιοχή, στην οποία περιορίζονται οι τιμές των μεταβλητών εισόδου, που προέρχονται από το μοντέλο φόρτου. Ένα μοντέλο μπορεί να μην είναι έγκυρο για τιμές μεταβλητών έξω από την περιοχή πειραματικής μελέτης, που επιλέχθηκε, αλλά εκείνο που ενδιαφέρει είναι η εγκυρότητά του μέσα σε αυτήν.

- *Προσαρμογή μοντέλου - παραμετροποίηση*: Είναι η διαδικασία υπολογισμού των παραμέτρων, που περιγράφουν με τον καλύτερο τρόπο ένα μοντέλο στην πειραματική περιοχή, που επιλέχθηκε. Αν π.χ. το σύστημα είναι επιθυμητό να περιγραφεί από κάποιο πολυωνυμικό μεταμοντέλο, τότε η παραμετροποίηση αυτού θα περιελάμβανε την προσαρμογή κάποιας εξίσωσης παλινδρόμησης για τον υπολογισμό των συντελεστών του πολυωνύμου. Αν όμως πρόκειται για πρωτογενή παραμετροποίηση δικτύου ουρών, τότε γίνεται χρήση του μοντέλου επεξεργασίας.

- *Πρόγνωση με τη χρήση του μοντέλου*: Αν οι τιμές των μεταβλητών εισόδου, για τις οποίες υπάρχει ενδιαφέρον να γίνει πρόγνωση, ανήκουν στην πειραματική περιοχή της μελέτης, τότε αυτό μπορεί πράγματι να χρησιμοποιηθεί για πρόγνωση. Αν όμως η πειραματική περιοχή αλλάξει, τότε το μοντέλο πρέπει πάλι να περάσει από τη φάση προσαρμογής - παραμετροποίησης αυτού.

- *Εκτίμηση εγκυρότητας*: Είναι η διαδικασία επαλήθευσης της εγκυρότητας του μοντέλου, μέσω της σύγκρισης τιμών πρόγνωσης της απόδοσης, που προέρχονται

από αυτό, με τιμές παρατήρησης του συστήματος. Αν ο έλεγχος εγκυρότητας επιτύχει, τότε οι τιμές πρόγνωσης μέσα στην επιλεγείσα πειραματική περιοχή γίνονται δεκτές. Αν ο έλεγχος εγκυρότητας αποτύχει, τότε το μοντέλο πρέπει να υποστεί ξανά τις απαραίτητες διαδικασίες προσαρμογής με χρήση νέων τιμών παρατήρησης.

Όταν η μελέτη ενός (υπο)συστήματος γίνεται με τη χρήση περισσοτέρων του ενός μοντέλων κάθε ένα από το οποία περιγράφει το σύστημα σε ένα διαφορετικό επίπεδο αφαίρεσης, τότε ξεκινώντας από τα χαμηλότερα επίπεδα και προχωρώντας στα υψηλότερα, τα μοντέλα παρουσιάζουν:

- μικρότερη εγκυρότητα,
- λιγότερο λεπτομερείς πληροφορίες για την απόδοση του συστήματος,
- λιγότερο ακριβείς πληροφορίες και
- μικρότερες υπολογιστικές απαιτήσεις και κατά συνέπεια μικρότερο κόστος αποτίμησης.

Πάντως, η περιγραφή ενός (υπο)συστήματος με περισσότερο και λιγότερο λεπτομερή μοντέλα δίνει επίσης τη δυνατότητα (ακριβέστερης) προσαρμογής - παραμετροποίησης των δεύτερων με βάση τα αποτελέσματα της μελέτης των πρώτων. Η επιλογή της τεχνικής, που θα χρησιμοποιηθεί για τη διαδικασία αυτή, γίνεται ανάλογα με τον τύπο του μοντέλου. Έτσι, όσον αφορά την προσαρμογή πολυωνυμικών μεταμοντέλων, παρουσίαση των κατάλληλων τεχνικών γίνεται στο κεφάλαιο 6. Σε ότι αφορά την προσαρμογή - παραμετροποίηση δικτύων ουρών αυτή πρέπει πάντα να γίνεται, λαμβάνοντας υπόψη τις διαφορές (μεταξύ των δύο μοντέλων) στο κριτήριο απόδοσης που ενδιαφέρει περισσότερο. Κάποιες από τις ευρέως διαδεδομένες τεχνικές είναι [FD89] οι:

- Μεταβολή των αποτελεσμάτων του μοντέλου υψηλότερου επιπέδου αφαίρεσης κατά ποσοστό ή απόλυτη τιμή, που αντιπροσωπεύει το μέγεθος της διαφοράς του κριτηρίου απόδοσης αναφοράς, από αυτό ενός αντίστοιχου μοντέλου χαμηλότερου επιπέδου.
- Προσαρμογή του αριθμού έργων, όταν πρόκειται για κλειστό δίκτυο ουρών, έτσι ώστε οι τιμές του κριτηρίου απόδοσης αναφοράς στα δύο μοντέλα να ταυτίζονται.
- Μεταβολή των απαιτήσεων σε όλα τα κέντρα εξυπηρέτησης μέχρι την επίτευξη σύμπτωσης στις τιμές του κριτηρίου απόδοσης αναφοράς.
- Μεταβολή των απαιτήσεων εξυπηρέτησης στο σημείο συμφόρησης (δηλαδή το κέντρο εξυπηρέτησης με την υψηλότερη αξιοποίηση) μέχρι την επίτευξη σύμπτωσης στις τιμές του κριτηρίου απόδοσης αναφοράς.
- Ενσωμάτωση ενός πλασματικού κέντρου καθυστέρησης και χαρακτηρισμός αυτού με απαιτήσεις εξυπηρέτησης τέτοιες, ώστε οι τιμές του κριτηρίου απόδοσης αναφοράς στα δύο μοντέλα να ταυτίζονται.

Στην αναπαράσταση με πολλαπλά μοντέλα, όταν αυτό που ενδιαφέρει είναι η βελτιστοποίηση ενός ήδη υπαρκτού συστήματος, τότε η μελέτη απόδοσης μπορεί να ξεκινήσει από μοντέλα χαμηλότερου επιπέδου, καθώς η πληροφορία για τη δομή του συστήματος υπάρχει. Όταν όμως ο σκοπός είναι η σχεδίαση ενός συστήματος από την αρχή, τότε στις περισσότερες περιπτώσεις, η μελέτη ξεκινά από μοντέλα υψηλότερου επιπέδου και καθώς η πληροφορία για τη δομή του συστήματος γίνεται περισσότερο συγκεκριμένη, προχωρά στην ανάπτυξη μοντέλων χαμηλότερου επιπέδου. Βέβαια με την εξέλιξη της

τεχνολογίας του αντικειμενοστρεφούς λογισμικού, τα σχεδιαστικά υποδείγματα, που έχουν καταγραφεί και τις τεχνολογίες κατανομής αντικειμένων που έχουν αναπτυχθεί, συχνά η πληροφορία για τη δομή του συστήματος όσον αφορά την αρχιτεκτονική αυτού είναι εκ των προτέρων λίγο ή πολύ γνωστή. Αυτό καθιστά εφικτή την εξ αρχής μελέτη της επίδρασης της αρχιτεκτονικής του συστήματος στην απόδοσή του, σε ένα αρκετά χαμηλό επίπεδο λεπτομέρειας.

Στις επόμενες παραγράφους του κεφαλαίου γίνεται παρουσίαση χαμηλού επιπέδου μοντέλων - τμημάτων κατάλληλων για χρήση στην αρχιτεκτονική περιγραφή μιας κατανεμημένης εφαρμογής. Επίσης, στο κεφάλαιο 3 έχουν ήδη αναφερθεί δύο υψηλού επιπέδου μοντέλα ουρών κατάλληλα για εφαρμογές CORBA/DCOM το πρώτο και εφαρμογές Enterprise JavaBeans το δεύτερο. Τέλος, στο κεφάλαιο 6 θεμελιώνονται διαδικασίες εξαγωγής υψηλού επιπέδου αξιόπιστων στατιστικών μεταμοντέλων από χαμηλότερου επιπέδου προσομοιωτικά μοντέλα· μεταμοντέλων, που ουσιαστικά αποτελούν αφαίρεση του μοντέλου απόδοσης, στο οποίο βασίζονται.

5.2.3 Τύποι ανάλυσης

Η ύπαρξη περισσοτέρων του ενός μοντέλων, τα οποία περιγράφουν το ίδιο (υπο)σύστημα σε διαφορετικά επίπεδα αφαίρεσης, εξυπηρετεί με τη μέγιστη δυνατή ευελιξία την ανάγκη εφαρμογής διαφορετικών τύπων ανάλυσης. Κάθε ένας από αυτούς εξετάζει την απόδοση του (υπο)συστήματος από διαφορετικές απόψεις, έτσι ώστε να διασφαλίζεται η επίτευξη του στόχου - πλαίσιο, ή η βελτιστοποίηση των ζητούμενων κριτηρίων απόδοσης. Μία ανάλυση περιγράφεται από μία ή περισσότερες εναλλακτικές διαδικασίες και λαμβάνει χώρα στο επίπεδο αφαίρεσης, το οποίο κρίνεται ως το πλέον πρόσφορο με βάση κριτήρια κόστους και αξιοπιστίας. Περιπτώσεις τέτοιων αναλύσεων είναι οι:

- *Ανάλυση - εντοπισμός σημείων συμφόρησης*: Βασίζεται στον εντοπισμό του σημείου/ων με τα υψηλότερα ποσοστά αξιοποίησης. Τα σημεία αυτά σε περιπτώσεις υψηλού φόρτου προκαλούν εκφυλισμό της απόδοσης του συστήματος και μπορεί να αντιπροσωπεύουν είτε υποσυστήματα λογισμικού, είτε πόρους υλικού. Όταν εντοπίζεται ένα σημείο συμφόρησης λογισμικού, είναι πιθανό η ανάλυσή του σε ένα χαμηλότερο επίπεδο αφαίρεσης να προκρίνει ως πιο πρόσφορη μία διαφορετική σχεδίαση αυτού. Αν γίνεται χρήση μιας ιεραρχικής μοντελικής προσέγγισης, ανάλογη αυτής, που περιγράφεται στην παράγραφο 5.1, η διαδικασία ανάλυσης - εντοπισμού σημείων συμφόρησης διευκολύνεται από τη δομή των μοντέλων απόδοσης, που χρησιμοποιούνται. Μια ανάλογη διαδικασία, που βασίζεται στη χρήση στρωματοποιημένων δικτύων ουρών (LQNs), αναπτύσσεται στη [NWP95].
- *Ανάλυση ορίων*: Σκοπό έχει την εύκολη και γρήγορη παροχή ενδείξεων για τον εντοπισμό των βασικών παραγόντων επηρεασμού της απόδοσης του (υπο)συστήματος. Σε υψηλού επιπέδου αφαίρεσης μοντέλα τα όρια της απόδοσης είναι συχνά εφικτό να υπολογίζονται πολύ γρήγορα και εύκολα οδηγώντας έτσι στο γρήγορο αποκλεισμό των ανεπαρκών λύσεων σχεδίασης. Σε περισσότερο λεπτομερή προσομοιωτικά μοντέλα μία ανάλυση ορίων σχετίζεται με τις διαδικασίες προσαρμογής μεταμοντέλων και ανάλυσης ευαισθησίας του κεφαλαίου 6. Σε πολλές περιπτώσεις η ανάλυση ορίων δίνει τη δυνατότητα ταυτόχρονης μελέτης ενός συνόλου και όχι απλά μίας εναλλακτικής λύσης. Έτσι, οι τεχνικές αυτές είναι ιδιαίτερα χρήσιμες σε μελέτες προσδιορισμού μεγέθους (sizing). Μία εισαγωγή στις τεχνικές ανάλυσης ορίων μπορεί κανείς να βρει στο

[LZGS84], ενώ σημαντικά ερευνητικά αποτελέσματα παρουσιάζονται στη [BS96].

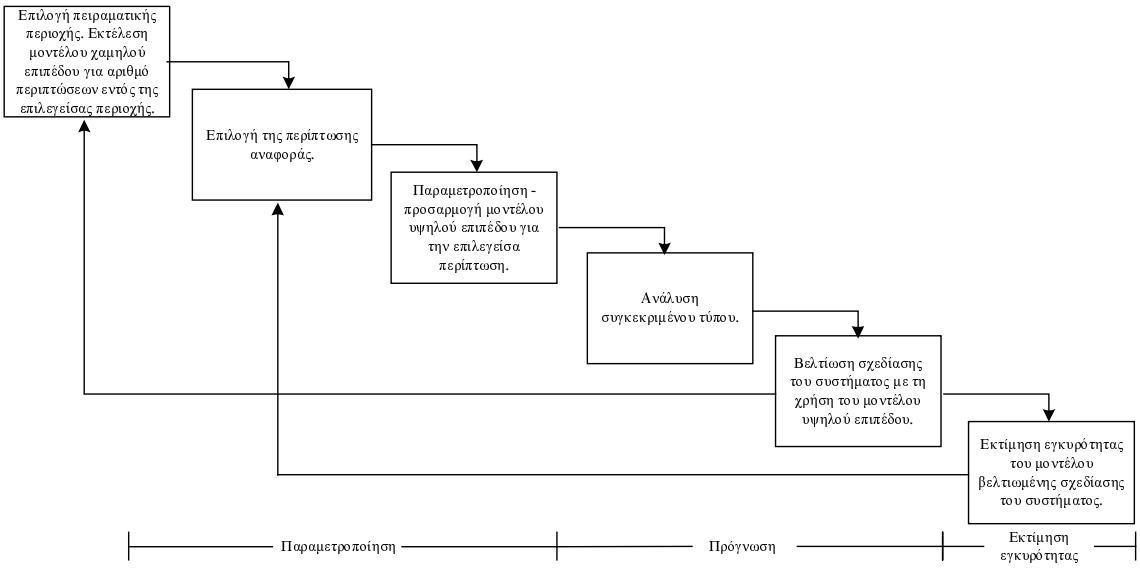
- *Παραμετρική ανάλυση*: Εφαρμόζεται όταν κάποιες από τις παραμέτρους του μοντέλου απόδοσης (όπως για παράδειγμα κάποιες απαιτήσεις εξυπηρέτησης) είναι δύσκολο να ορισθούν με ακρίβεια ή υπάρχουν πληροφορίες μόνο για τα πλαίσια διακύμανσης αυτών. Σε μία τέτοια περίπτωση, η κατάλληλη τεχνική παραμετρικής ανάλυσης αποδίδει ως αποτέλεσμα τη δυνατότητα γρήγορου υπολογισμού των επιθυμητών κριτηρίων απόδοσης για διαφορετικές τιμές των παραμέτρων, που χαρακτηρίζονται από περιθώρια αβεβαιότητας. Τέτοια τεχνική είναι αυτή που περιγράφεται στη [MLH01], ενώ και οι διαδικασίες προσαρμογής μεταμοντέλων του κεφαλαίου 6 μπορούν επίσης να χρησιμοποιηθούν για το σκοπό μιας ανάλυσης αυτού του τύπου. Τέλος, συχνά γίνονται προσπάθειες ανάπτυξης αναλυτικών αφαιρέσεων, προσπάθειες όμως που συνοδεύονται από υψηλό κόστος.

- *Ανάλυση ευαισθησίας*: Στόχο έχει το γρήγορο προσδιορισμό της απόδοσης του (υπο)συστήματος σε ενδεχόμενη αλλαγή παραμέτρου/ων, χωρίς να χρειάζεται επιπλέον ανάλυση. Στη [LN91] εισάγονται αναλυτικά αποτελέσματα ευαισθησίας για δίκτυα ουρών μορφής γινομένου. Από την άλλη μεριά, στο κεφάλαιο 6 της διατριβής περιγράφονται διαδικασίες ανάλυσης ευαισθησίας σε προσομοιωτικά μοντέλα. Ενώ η πρώτη τεχνική υπερτερεί της δεύτερης σε επίπεδο υπολογιστικού κόστους, η δεύτερη είναι σαφώς πιο γενική και έχει τη δυνατότητα να δίνει απαντήσεις σε περιπτώσεις αλλαγών ποιοτικών χαρακτηριστικών του (υπο)συστήματος.

- *Ανάλυση κλιμάκωσης*: Πρόβλημα φύσει πολυδιάστατο μια και στα συστήματα κατανεμημένης αρχιτεκτονικής ενδιαφέρει συνήθως η δυνατότητα κλιμάκωσης όσον αφορά όχι μόνον ένα, αλλά περισσότερους παράγοντες. Έτσι, μία ανάλυση αυτού του τύπου, αν και μπορεί να βασιστεί στην ανάλυση ευαισθησίας, απαιτεί υψηλό βαθμό συστηματοποίησης εξαιτίας της μεγάλης πολυπλοκότητάς της. Στη [FG98] οι συγγραφείς εισάγουν ένα σύνολο εννοιών, που μπορεί να αποτελέσει τη βάση για την ανάπτυξη μιας συστηματοποιημένης διαδικασίας ανάλυσης κλιμάκωσης. Σύμφωνα λοιπόν με την προσέγγιση αυτή, μία σχεδίαση λογισμικού κατανεμημένης αρχιτεκτονικής λέμε ότι διαθέτει δυνατότητα κλιμάκωσης ως προς κάποιο *παράγοντα*, αν το μοντέλο προγιγνώσκει την εκπλήρωση της απόδοσης στόχου - πλαίσιο, που αναμένει ο χρήστης, μέσα στα *πλαίσια ανοχής* και *ορίων κλιμάκωσης*. Έτσι για παράδειγμα, μία κατανεμημένη εφαρμογή διαχείρισης δικτύου μπορούμε να πούμε ότι διαθέτει την επιθυμητή δυνατότητα κλιμάκωσης, αν το μοντέλο απόδοσης δείχνει ότι οι μέσοι χρόνοι απόκρισης δε χειροτερεύουν περισσότερο από 50% για τη διαχείριση μέχρι και ενός εκατομμυρίου συσκευών. Στην περίπτωση αυτή, το κριτήριο απόδοσης στόχου - πλαίσιο είναι ο συνολικός χρόνος απόκρισης, ο παράγοντας κλιμάκωσης είναι ο αριθμός των συσκευών, που διαχειρίζεται το σύστημα, και η ανοχή κλιμάκωσης είναι η μεταβολή μέχρι και 50%. Όριο κλιμάκωσης για τη συγκεκριμένη περίπτωση είναι ο αριθμός του ενός εκατομμυρίου διαχειριζόμενων συσκευών. Επιπλέον, η έννοια των ορίων κλιμάκωσης μπορεί ακόμη να χρησιμοποιηθεί για το συσχετισμό επιτρεπόμενων επιπέδων κόστους με το μέγεθος των μεταβολών του παράγοντα κλιμάκωσης.

Στο σχήμα 5.4 απεικονίζεται μία γενική διαδικασία βελτίωσης της σχεδίασης ενός συστήματος, όταν αυτό περιγράφεται από μοντέλα διαφορετικών επιπέδων αφαίρεσης. Ο

τύπος της ανάλυσης, που σε κάθε περίπτωση εφαρμόζεται, εξαρτάται από το στόχο - πλαίσιο της απόδοσης του συστήματος ή/και την κατά το δυνατό αποφυγή της πιθανότητας κορεσμού αυτού (*μελέτη προσδιορισμού μεγέθους*).



Σχήμα 5.4 Διαδικασία βελτίωσης της σχεδίασης ενός συστήματος, που περιγράφεται από μοντέλα απόδοσης διαφορετικών επιπέδων αφαίρεσης

5.2.4 Τυποποίηση

Η ιεραρχική μοντελική προσέγγιση, που προτείνεται, συμβαίνει επίσης να είναι πλήρως συμβατή με τη στρωματοποιημένη ερμηνεία του «Γενικού Μοντέλου Πόρων», που αποτελεί τον πυρήνα της πρότασης [AIR01] του OMG για τη μοντελοποίηση και ανάλυση προβλεψιμότητας, απόδοσης και χρόνου (Response to the OMG RFP for Schedulability, Performance and Time). Επιπλέον, το «Γενικό Μοντέλο Πόρων» είναι δομημένο με τέτοιο τρόπο, ώστε να επιτρέπει την περιγραφή ενός συστήματος με μοντέλα διαφορετικών επιπέδων αφαίρεσης.

Τα στοιχεία αυτά συγκροτούν βέβαια ένα στέρεο θεωρητικό υπόβαθρο της μοντελικής προσέγγισης και των διαδικασιών, που την αξιοποιούν. Βεβαίως, η φυσιολογική εξέλιξη της προσπάθειας αυτής θα είναι η προσαρμογή των εννοιών, οι οποίες χρησιμοποιήθηκαν στις προηγούμενες ενότητες, έτσι ώστε αυτές να είναι πλήρως συμβατές με τις εν λόγω προδιαγραφές. Έτσι, η μοντελοποίηση της απόδοσης συστημάτων κατανεμημένης αρχιτεκτονικής θα γίνεται αρχικά με τη χρήση των επεκτάσεων της UML, που προτείνονται στην [AIR01]. Τέλος, η περαιτέρω αποκρυστάλλωση των διαδικασιών, που ήδη σκιαγραφήθηκαν στις προηγούμενες ενότητες, θα οδηγήσει στη συστηματική θεμελίωση μιας μεθόδου ανάλυσης της απόδοσης, δυνατότητα που εξάλλου προβλέπεται από την [AIR01].

5.3 Μοντέλα απόδοσης λογισμικού αρχιτεκτονικής CORBA

Η παράγραφος αυτή επικεντρώνεται στη δυνατότητα πρώιμης ανάλυσης, μέσω μοντέλων απόδοσης, των εναλλακτικών λύσεων αρχιτεκτονικής σχεδίασης εφαρμογών τεχνολογίας CORBA. Μία τέτοιου είδους ανάλυση της αρχιτεκτονικής του προς ανάπτυξη συστήματος μπορεί:

- να εντοπίσει και να διορθώσει σημεία συμφόρησης λόγω κόστους επικοινωνίας μεταξύ των μονάδων λογισμικού,
- να εντοπίσει κρίσιμα στοιχεία της αρχιτεκτονικής, τα οποία χρειάζονται λεπτομερέστερη μοντελοποίηση, έτσι ώστε οι περαιτέρω προσπάθειες σχεδίασης να επικεντρωθούν σε αυτά, ούτως ώστε
- να επιτευχθούν επίπεδα δυνατοτήτων κλιμάκωσης, που να ανταποκρίνονται στις απαιτήσεις της εφαρμογής.

Στο πλαίσιο αυτό δεν υπάρχει απαίτηση μεγάλης ακρίβειας στην παραμετροποίηση των χρησιμοποιούμενων μοντέλων. Καθώς δε, συχνά οι πληροφορίες για το περιβάλλον ανάπτυξης της εφαρμογής είναι ελλιπείς, μπορούν για απαιτήσεις εξυπηρέτησης, να χρησιμοποιηθούν αριθμοί, που είτε προέρχονται από μετρήσεις benchmarks, είτε από μετρήσεις σε κάποιο πρωτότυπο, είτε από κάποια βάση μετρήσεων. Βέβαια το πιθανότερο είναι οι αριθμοί αυτοί να έχουν ληφθεί σε διαφορετικό περιβάλλον (υλικό, υποδομή δικτύου, λειτουργικό σύστημα) από αυτό, που πρόκειται να χρησιμοποιηθεί, γεγονός όμως όχι σημαντικό για την εξυπηρέτηση των τριών στόχων, που αναφέρθηκαν παραπάνω. Αν πάντως οι απαιτήσεις εξυπηρέτησης έχουν μετρηθεί σε μονάδες ανεξάρτητες του περιβάλλοντος εκτέλεσης, όπως π.χ. σε αριθμούς μικροεντολών μηχανής, τότε αυτές εύκολα μετατρέπονται σε χρόνους εξυπηρέτησης βάσει της ταχύτητας του ρολογιού της προς χρήση μηχανής.

Συχνά οι απαιτήσεις εξυπηρέτησης για την εκτέλεση κάποιας λειτουργίας εξαρτώνται από μεταβλητές εισόδου του μοντέλου, όπως π.χ. το μέγεθος του προς αποστολή μηνύματος ή ο αριθμός των αντικειμένων, που θα παρέχουν εξυπηρέτηση στον κόμβο κάποιου διακομιστή. Στις περιπτώσεις αυτές είναι επιθυμητό τα αποτελέσματα των μετρήσεων να έχουν υποστεί την κατάλληλη στατιστική επεξεργασία για την προσαρμογή πολυωνυμικών μεταμοντέλων γνωστών και ως *συναρτήσεις πόρων*. Περιγραφή αυτού του είδους της στατιστικής επεξεργασίας γίνεται στην παράγραφο 6.2.

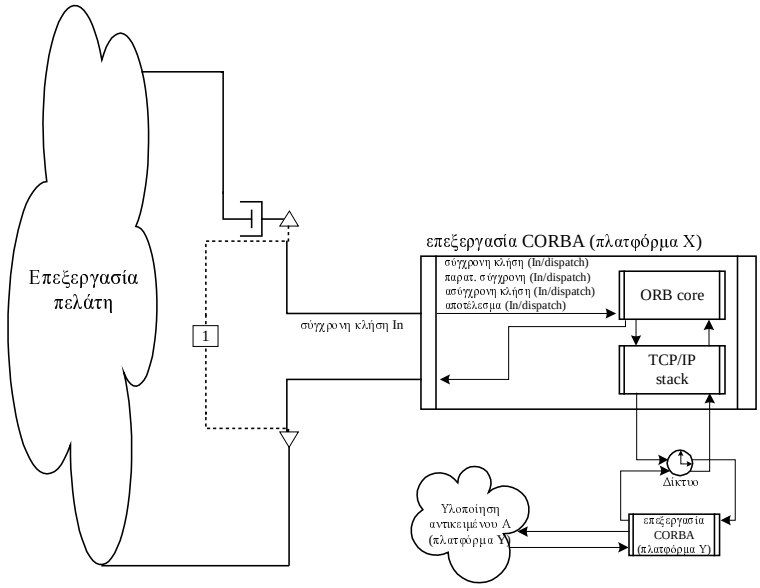
5.3.1 Μοντελοποίηση διαφορετικών τύπων επικοινωνίας μέσω ORB

Όπως αναφέρθηκε στην ενότητα 2.7.7, όταν τα κατανεμημένα αντικείμενα αλληλεπιδρούν μεταξύ τους μέσω μιας υποδομής αγωγού κλήσεων (ORB) κάνουν χρήση είτε σύγχρονης, είτε παρατεινόμενης σύγχρονης, είτε ασύγχρονης επικοινωνίας.

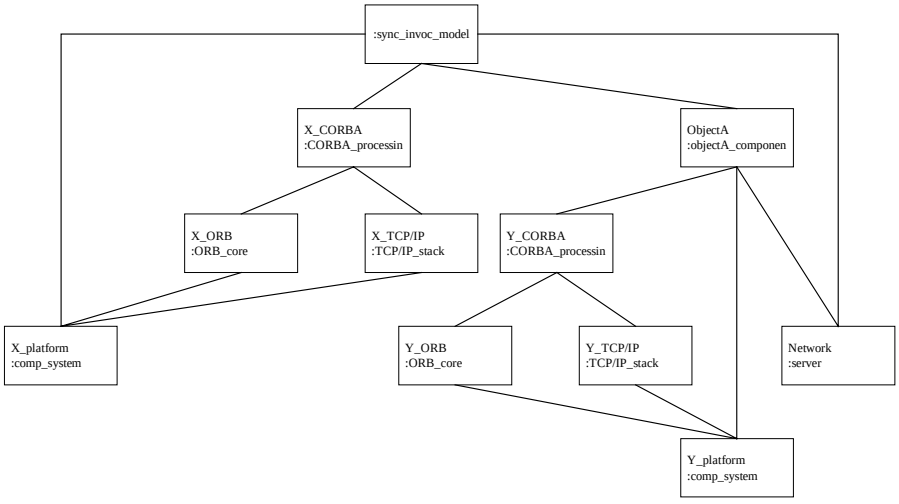
Ανάλογα με τον τύπο που επιλέγεται, κάθε επικοινωνία συνοδεύεται από κάποιο κόστος. Πέρα όμως από το φόρτο επεξεργασίας, που διοχετεύεται σε απομακρυσμένους κόμβους υπάρχει και αυτός, που εξαιτίας της επικοινωνίας δημιουργείται στο σταθμό αποστολής. Επιπλέον, είναι σαφές ότι οι δύο πρώτοι τύποι επικοινωνίας επιβάλλουν την ύπαρξη κάποιας μορφής συγχρονισμού στον πελάτη, που αναπόφευκτα επηρεάζει το χρόνο απόκρισης αυτού (σχήμα 5.5).

Έτσι, συχνά είναι προτιμητέα η ανάπτυξη πελατών πολυνημάτωσης (multithreading), που αν και πιο περίπλοκη, αποκλείει τις περιπτώσεις μπλοκαρίσματος επεξεργασίας λόγω αναμονής ολοκλήρωσης κάποιας σύγχρονης κλήσης. Έχουν προταθεί διάφορες επιλογές πολυνημάτωσης πελάτη. Σε μία από αυτές [HSV96], όταν ο πελάτης αποστέλλει κάποια κλήση μεθόδου, αυτόματα δημιουργείται ένα νέο νήμα ροής για την αναμονή της απόκρισης.

Το κύριο νήμα ροής συνεχίζει την εκτέλεση του κώδικα της εφαρμογής μέχρι τη στιγμή, που θα προσπαθήσει να χρησιμοποιήσει τα αποτελέσματα της κλήσης. Τότε, αν ο διακομιστής έχει ήδη επιστρέψει τα αποτελέσματα, ο πελάτης μπορεί να έχει πρόσβαση σε αυτά. Σε διαφορετική περίπτωση η εκτέλεση του κώδικα του πελάτη μπλοκάρει μέχρι να γίνουν διαθέσιμα τα αποτελέσματα.

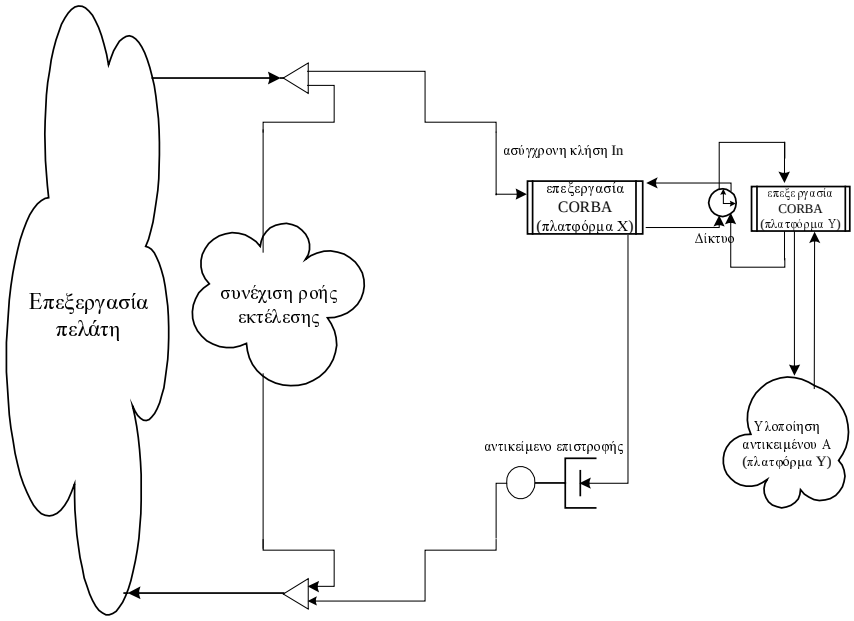


Σχήμα 5.5 Σύγχρονη κλήση χωρίς πολυνημάτωση στον πελάτη



Σχήμα 5.6 Μία πιθανή ιεραρχία υλοποίησης του μοντέλου του σχήματος 5.5

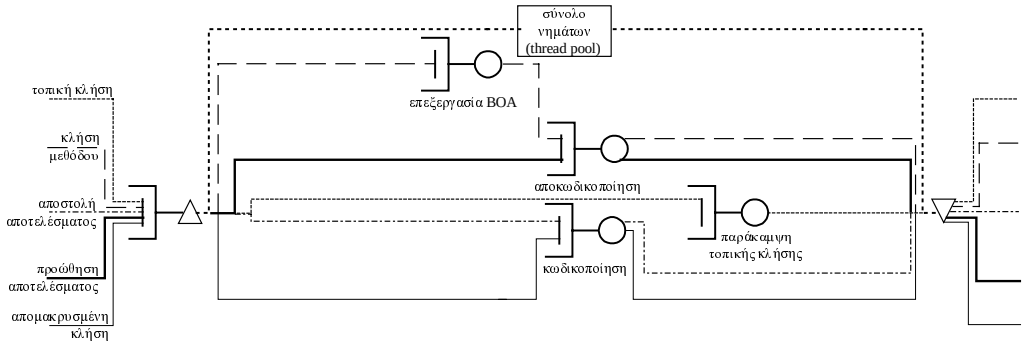
Εκείνη όμως η τεχνική, που χρησιμοποιείται πολύ συχνά [OMG98], είναι η τεχνική της *κλήσης επιστροφής* (callback). Κατά την επικοινωνία με κλήση επιστροφής, όλες οι κλήσεις είναι ασύγχρονες (σχήμα 5.7). Αρχικά λοιπόν ο πελάτης περνάει ως μέρος της κλήσης την αναφορά ενός *αντικειμένου επιστροφής* (callback object) και συνεχίζει τη ροή εκτέλεσής του χωρίς να μπλοκάρει. Όταν ολοκληρωθεί η επεξεργασία της κλήσης από το καλούμενο αντικείμενο, τότε καλείται το αντικείμενο επιστροφής, στο οποίο μεταβιβάζονται τα αποτελέσματα της αρχικής κλήσης. Η τεχνική αυτή έχει το πλεονέκτημα της μη χρήσης νημάτων ροής επιπέδου εφαρμογής. Έτσι, είναι πιο εύκολα υλοποιήσιμη και βέβαια απαλλαγμένη από το όποιο κόστος συνεπάγεται η δημιουργία ενός νέου νήματος ροής.



Σχήμα 5.7 Η τεχνική της ασύγχρονης κλήσης επιστροφής (callback)

Στο σχήμα 5.5 απεικονίζεται ένα μοντέλο απόδοσης πελάτη στην περίπτωση μιας σύγχρονης κλήσης και στο σχήμα 5.6 μία πιθανή ιεραρχία υλοποίησης στο HIT. Το τμήμα «επεξεργασία CORBA» παρέχει οκτώ διαφορετικές υπηρεσίες, που αντιστοιχούν στη δρομολόγηση (In) ή στην προώθηση (dispatch) των τριών τύπων κλήσεων επικοινωνίας και του αποτελέσματος. Οι περιγραφές των παρεχόμενων υπηρεσιών κάνουν χρήση υπηρεσιών, που παρέχονται από τα τμήματα «ORB Core» και «TCP/IP stack», που επίσης απεικονίζονται στο σχήμα. Το μεν πρώτο αφορά το φόρτο, που προκαλεί η επεξεργασία του πυρήνα του αγωγού κλήσεων, ενώ το δεύτερο αφορά το φόρτο, που προκαλεί η επεξεργασία του πρωτοκόλλου TCP/IP του λειτουργικού συστήματος.

5.3.2 Μοντελοποίηση επεξεργασίας ORB



Σχήμα 5.8 Τμήμα «ORB core» (αρχιτεκτονική πολυνημάτωσης με πολιτική συνόλου νημάτων ροής οδηγού - ακολούθων [SCH98] και βελτιστοποίηση τοπικών κλήσεων). Απεικονίζεται μόνο η επεξεργασία στατικών κλήσεων.

Στην περίπτωση που υπάρχουν αναλυτικά στοιχεία μετρήσεων για τη λειτουργία του αγωγού κλήσεων, το σχήμα 5.8 απεικονίζει (μόνο για τις στατικές κλήσεις) μία από τις πιθανές αρχιτεκτονικές αυτού. Το τμήμα αυτό του μοντέλου παρέχει τις υπηρεσίες:

- *Κλήση μεθόδου*: Υπηρεσία που χρησιμοποιείται για την προώθηση μιας εισερχόμενης κλήσης στην υλοποίηση αντικειμένου, που απευθύνεται.
- *Απομακρυσμένη κλήση*: Υπηρεσία που χρησιμοποιείται στη δρομολόγηση κλήσης, που απευθύνεται σε απομακρυσμένη υλοποίηση αντικειμένου.
- *Τοπική κλήση*: Υπηρεσία που επιχειρεί να εκφράσει τη βελτιστοποιημένη επεξεργασία τοπικών κλήσεων, ένα χαρακτηριστικό, που διαθέτουν πολλά προϊόντα, αλλά δεν περιγράφεται στις προδιαγραφές CORBA (για το λόγο αυτό και αναπαριστάται από ξεχωριστή ουρά αναμονής).
- *Αποστολή αποτελέσματος*: Υπηρεσία που εκφράζει την επεξεργασία επιστροφής αποτελέσματος.
- *Προώθηση αποτελέσματος*: Υπηρεσία που περιγράφει την επεξεργασία του επιστρεφόμενου αποτελέσματος, όταν αυτό προωθείται πίσω στο αντικείμενο, που δρομολόγησε την αρχική κλήση.

Οι επεξεργασίες BOA, κωδικοποίησης και αποκωδικοποίησης αντιπροσωπεύουν εκτέλεση λογισμικού (πειθαρχία εξυπηρέτησης PS), που δημιουργεί υπολογιστικό φόρτο στην υποκείμενη πλατφόρμα υλικού. Αποτελούν δε τις πιο σημαντικές πηγές καθυστέρησης στην επικοινωνία δύο αντικειμένων.

- *Κωδικοποίηση* (marshaling): Η κωδικοποίηση των παραμέτρων μιας κλήσης για την αποστολή τους μέσα από μια υποδομή CORBA έχει βρεθεί [GS97a] ότι συχνά κυμαίνεται μεταξύ 25% και 42% του συνολικού χρόνου επεξεργασίας στον πυρήνα του αγωγού κλήσεων από τη μεριά του πελάτη. Η ταχύτητα της κωδικοποίησης εξαρτάται βέβαια από το προϊόν CORBA, που χρησιμοποιείται, λόγω διαφορών στη διαχείριση των προσωρινών αποθηκών μηνυμάτων δικτύου και την αποφυγή ή όχι (ανάλογα με τον αλγόριθμο) μεγάλου αριθμού αντιγραφών δεδομένων. Παρόλα αυτά, βασικός παράγοντας καθορισμού της ταχύτητας κωδικοποίησης φαίνεται ότι είναι το μέγεθος της προς αποστολή κλήσης και οι τύποι δεδομένων των παραμέτρων, που την απαρτίζουν.
- *Επεξεργασία βασικού προσαρμογέα αντικειμένου* (BOA): Η λειτουργικότητα του BOA, που ενδιαφέρει βέβαια μία ανάλυση απόδοσης σε κατάσταση στατιστικής ισορροπίας δεν είναι η δημιουργία, η ενεργοποίηση και η απενεργοποίηση αντικειμένων, αλλά αυτή της αποσυμπλοκής (demultiplexing) και προώθησης των εισερχόμενων κλήσεων μεθόδων στους κατάλληλους σκελετούς. Οι επεξεργασίες αυτές μετρήθηκαν ([GS96] και [GS97a]) ότι μπορεί να καταλαμβάνουν περίπου το 17% του συνολικού χρόνου επεξεργασίας στον πυρήνα του αγωγού κλήσεων από τη μεριά του διακομιστή. Είναι επίσης σαφές [GS97b] ότι ο φόρτος, που προκαλούν, εξαρτάται από τον αριθμό των αντικειμένων, που βρίσκονται στο διακομιστή, αλλά και τον αριθμό των μεθόδων, που έχουν δηλωθεί στη διασύνδεση IDL. Στη [GS97b] γίνεται σύγκριση τριών διαφορετικών περιπτώσεων αποσυμπλοκής, που υλοποιούνται σε προϊόντα CORBA της αγοράς. Η σειριακή αναζήτηση είναι αυτή, που βρέθηκε ότι αποτελεί σημαντικό ανασταλτικό παράγοντα όσον αφορά τις δυνατότητες κλιμάκωσης ως προς τον αριθμό αντικειμένων, που βρίσκονται στο διακομιστή.
- *Αποκωδικοποίηση* (unmarshaling): Η αποκωδικοποίηση είναι μία άλλη χρονοβόρα δραστηριότητα. Στην περίπτωση των παραμέτρων μιας κλήσης, μαζί με την επεξεργασία BOA, έχει βρεθεί [GS97a] ότι μπορεί να καταλαμβάνει το 72% του συνολικού χρόνου επεξεργασίας στον πυρήνα του αγωγού κλήσεων από

τη μεριά του διακομιστή. Όταν βέβαια πρόκειται για την προώθηση του αποτελέσματος μιας κλήσης ο χρόνος αυτός είναι ίσως μικρότερος, αλλά επίσης σημαντικός, ώστε να δικαιολογεί την αναπαράσταση στο μοντέλο του φόρτου επεξεργασίας που προκαλεί. Η επεξεργασία αποκωδικοποίησης λοιπόν είναι παρόμοια με αυτή της κωδικοποίησης με σημαντικότερες διαφορές ίσως:

- την κατανομή μνήμης, καθώς ο πυρήνας αγωγού κλήσεων διαθέτει μνήμη για τα αποκωδικοποιημένα δεδομένα και
- την αποκωδικοποίηση αναφορών αντικειμένων, καθώς κατά τη διαδικασία αυτή γίνεται αναζήτηση μέσα σε έναν πίνακα αντικειμένων γεγονός, που επίσης εγείρει θέμα δυνατοτήτων κλιμάκωσης.

Σημαντικό ρόλο επίσης στην επικοινωνία μεταξύ των αντικειμένων παίζει η αρχιτεκτονική του πυρήνα αγωγού κλήσεων ως προς το αν αυτός έχει δομή πολυνημάτωσης ή όχι. Οι επιπτώσεις της πολυνημάτωσης είναι:

- Η απλοποίηση της σχεδίασης της εφαρμογής, καθώς αυτή επιτρέπει την ανεξάρτητη εκτέλεση πολλαπλών στρωμάτων διακομιστών, που μπορούν να επικοινωνούν μεταξύ τους με τη χρήση συμβατικών σύγχρονων κλήσεων.
- Η βελτίωση της συνολικής παραγωγής και των χρόνων καθυστέρησης λόγω επικοινωνίας με την εκμετάλλευση υλικού παράλληλης επεξεργασίας και, όπου είναι δυνατό, την επικάλυψη μεταξύ λειτουργιών υπολογισμού και επικοινωνίας.
- Τη βελτίωση του συνολικού χρόνου απόκρισης με την αντιστοίχηση διαφορετικών νημάτων ροής σε διαφορετικές κλήσεις, έτσι ώστε να αποφεύγεται το μπλοκάρισμα της λειτουργίας ενός πελάτη εξαιτίας κάποιας άλλης κλήσης.

Ο προγραμματισμός με προϊόντα CORBA, που δε διαθέτουν πολυνημάτωση είναι αρκετά δύσκολος ειδικά όσον αφορά την ανάπτυξη διακομιστών εξυπηρέτησης. Στην [SCH98] γίνεται μία περιγραφή των αρχιτεκτονικών πολυνημάτωσης, που χρησιμοποιούνται από τα πιο γνωστά προϊόντα CORBA και των επιπτώσεων, που αυτές έχουν, στην απόδοση της επικοινωνίας μεταξύ των αντικειμένων. Στο σχήμα 5.8 φαίνεται η δομή ενός πυρήνα αγωγού κλήσεων με αρχιτεκτονική πολυνημάτωσης και πολιτική συνόλου νημάτων ροής οδηγού-ακολούθων (leader/follower thread pool), όπως αυτή που υποστηρίζει το προϊόν miniCOOL της Sun.

Οι σημαντικότερες αρχιτεκτονικές πολυνημάτωσης, που χρησιμοποιούνται από τα προϊόντα CORBA είναι:

- η αρχιτεκτονική νήματος ροής ανά κλήση,
- η αρχιτεκτονική νήματος ροής ανά σύνδεση,
- η αρχιτεκτονική νήματος ροής ανά αντικείμενο,
- η αρχιτεκτονική συνόλου νημάτων ροής εργασίας,
- η αρχιτεκτονική συνόλου νημάτων ροής οδηγού - ακολούθων

και διάφορες υβριδικές αρχιτεκτονικές συνδυασμών των παραπάνω.

Τα κέντρα εξυπηρέτησης, που περιλαμβάνονται στο δίκτυο ουρών του σχήματος 5.8, όπως και αυτά που απεικονίζονται ή δεν απεικονίζονται αναλυτικά στα σχήματα 5.5 και 5.7, εκφράζονται, σε μία ιεραρχική μοντελική προσέγγιση, από τη χρήση υπηρεσιών επεξεργασίας, που παρέχονται από ένα άλλο τμήμα, το οποίο αντιπροσωπεύει την υποκείμενη πλατφόρμα υλικού. Οι λεπτομέρειες του τμήματος αυτού (που θα μπορούσε για

παράδειγμα να ονομάζεται CompSys) δεν είναι σημαντικές για τους σκοπούς της παραγράφου. Αυτό θα μπορούσε να περιλαμβάνει μία ή περισσότερες CPU και ενδεχομένως μηχανισμό καταμερισμού μνήμης. Θα μπορούσε επίσης εύκολα να αντικατασταθεί από κάποιο άλλο τμήμα, που εκφράζει τα χαρακτηριστικά μιας νέας πλατφόρμας σε ενδεχόμενη αλλαγή της διαμόρφωσης του συστήματος.

Οι απαιτήσεις εξυπηρέτησης, που δημιουργούνται στην υποκείμενη πλατφόρμα υλικού μπορούν, όπως αναφέρθηκε, να υπολογιστούν από μετρήσεις benchmarks ή από μετρήσεις σε κάποιο πρωτότυπο ή από κάποια βάση μετρήσεων. Πέρα από τα άρθρα, τα οποία ήδη αναφέρθηκαν κατά την ανάλυση των σημαντικότερων πηγών καθυστέρησης στην επικοινωνία αντικειμένων, υπάρχει και μία λεπτομερής αναφορά [MLC98] ενός έργου σύγκρισης των πιο διαδεδομένων υλοποιήσεων CORBA, που έλαβε χώρα στο Charles University της Τσεχίας. Η σύγκριση διενεργήθηκε με τη χρήση μεγάλου αριθμού μετρήσεων benchmarks και έδειξε σημαντικές διαφορές στην απόδοση μεταξύ των προϊόντων που εξετάσθηκαν. Πρόσφατα επίσης το OMG διερεύνησε [OMG99] τις προοπτικές καθορισμού οδηγιών για τη διενέργεια μετρήσεων benchmarks σε προϊόντα CORBA.

Παρόλα αυτά, είναι σαφές ότι δεν μπορούμε να μιλήσουμε για επάρκεια δεδομένων. Επίσης, αυτά τα οποία υπάρχουν, δεν κάνουν χρήση στατιστικών μεθόδων για την προσαρμογή συναρτήσεων πόρων και έτσι η επαναχρησιμοποίηση τους καθίσταται δύσκολη.

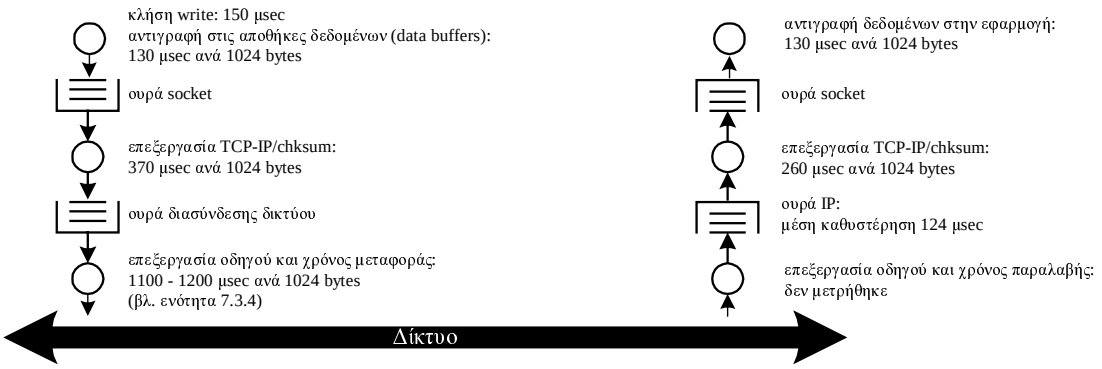
Από την άλλη, είναι γεγονός ότι οι περισσότεροι κατασκευαστές προϊόντων CORBA γενικά δεν αποκαλύπτουν τις λεπτομέρειες σχεδίασης των προϊόντων τους. Έτσι για παράδειγμα δεν είναι πάντα γνωστοί π.χ. οι μηχανισμοί, που χρησιμοποιούνται από ένα προϊόν CORBA για τον εντοπισμό των αντικειμένων (lookup messages) ή/και οι μηχανισμοί ενεργοποίησης και απενεργοποίησης του βασικού προσαρμογέα αντικειμένου (BOA). Για το λόγο αυτό οι όποιες καθυστερήσεις προκαλούνται από τις συγκεκριμένες λειτουργίες, είτε δεχόμαστε ότι ενσωματώνονται στα υπόλοιπα κέντρα εξυπηρέτησης του μοντέλου κατά την παραμετροποίηση αυτού, είτε γίνεται χρήση επιπλέον κέντρων καθυστέρησης, όταν είναι επιθυμητό να εκφρασθούν στο μοντέλο άμεσα. Οι χρόνοι καθυστέρησης των κέντρων αυτών θα αντιπροσωπεύουν όλες τις καθυστερήσεις, που δεν εκφράζονται από τα άλλα κέντρα εξυπηρέτησης και που στην περίπτωση δυναμικής δημιουργίας νημάτων ροής (π.χ. νήμα ροής ανά κλήση και νήμα ροής ανά σύνδεση) είναι επιθυμητό να περιλαμβάνουν και τις καθυστερήσεις, που αυτή συνεπάγεται.

5.3.3 Μοντελοποίηση επεξεργασίας πρωτοκόλλου μεταφοράς

Ένα άλλο τμήμα των μοντέλων, που παρουσιάσθηκαν, είναι το τμήμα «TCP/IP stack», που εμφανίζεται στο σχήμα 5.5. Το τμήμα αυτό εκφράζει, όπως προαναφέρθηκε, το φόρτο και τις καθυστερήσεις, που προκαλεί η επεξεργασία του πρωτοκόλλου TCP/IP του λειτουργικού συστήματος. Οι καθυστερήσεις αυτές αποτελούν ένα σημαντικό μέρος της συνολικής, που παρατηρείται στην επικοινωνία των αντικειμένων, και όσον αφορά τη διάρκειά τους, καθοριστικό ρόλο φαίνεται ότι παίζει το μέγεθος της ουράς ανά socket (socket queue size), που χρησιμοποιείται από το λειτουργικό σύστημα και το μέγεθος των αποθηκών δεδομένων, που εξαρτάται από τον τύπο και τον αριθμό των παραμέτρων της κλήσης.

Από άρθρα, τα οποία έχουν δημοσιευθεί και ασχολούνται με το συγκεκριμένο πρόβλημα, φαίνεται [DMO94] ότι οι καθυστερήσεις της επεξεργασίας TCP είναι προβλέψιμες και μεταβάλλονται με τα επίπεδα παραγωγής. Επίσης, στις [MKK94], [KP93] και [PP93] παρουσιάζονται λεπτομερείς μετρήσεις (όπως π.χ. στο σχήμα 5.9), ενδεικτικά χρήσιμες για την παραμετροποίηση μοντέλων, όπου στις περισσότερες περιπτώσεις το κόστος της επεξεργασίας αυτής εκφράζεται με ένα μόνο κέντρο εξυπηρέτησης. Βέβαια σε μία ιεραρχική

μοντελική προσέγγιση, ένα τέτοιο κέντρο καθυστέρησης κάνει χρήση υπηρεσίας, που παρέχεται από το τμήμα της υποκείμενης πλατφόρμας υλικού.



Σχήμα 5.9 Καθυστερήσεις λόγω επεξεργασίας πρωτοκόλλου στην επικοινωνία δύο σταθμών Sun Sparcstation 1 με λειτουργικό σύστημα SunOS 4.0.3, BSD stream sockets και δίκτυο Ethernet με οδηγό AMD Am7990 LANCE [PP93]. Οι μετρήσεις έγιναν με χρήση του προκαθορισμένου μεγέθους ουράς ανά socket.

5.3.4 Μοντελοποίηση καθυστέρησης δικτύου μεταφοράς δεδομένων

Μία πρακτική προσέγγιση του χρόνου του κέντρου καθυστέρησης, που εκφράζει το κόστος μεταφοράς δεδομένων μέσω ενός ή περισσοτέρων δικτύων, είναι αυτή που αναπτύσσεται στο [MA98]. Κατά την προσέγγιση αυτή διακρίνουμε δύο περιπτώσεις: i) αυτήν κατά την οποία οι δρομολογητές τεμαχίζουν τα δεδομένα, αν το δίκτυο στο οποίο αυτά εισέρχονται δε δέχεται την ίδια *μέγιστη ποσότητα ανά πακέτο* (Maximum Transmission Unit) και ii) την περίπτωση κατά την οποία ο αποστολέας διαπιστώνει πρώτα πια είναι η ελάχιστη MTU των δικτύων της διαδρομής ως τον παραλήπτη και αυτή χρησιμοποιεί ως μέγεθος πακέτων (datagrams).

Η δεύτερη περίπτωση είναι αυτή, που προτείνεται από τις προδιαγραφές του πρωτοκόλλου IP μια και αποφεύγει τις καθυστερήσεις τεμαχισμού και συναρμολόγησης των πακέτων από ενδιάμεσους δρομολογητές. Ας υποθέσουμε λοιπόν το συμβολισμό ενός συνόλου μεταβλητών, όπως αυτό που περιγράφεται στον πίνακα 5.2.

Πίνακας 5.2 Μεταβλητές μεταφοράς δεδομένων μέσω ενός ή περισσοτέρων δικτύων

MessageSize	το μέγεθος της ποσότητας δεδομένων, που μεταφέρεται από κάποιο κόμβο σε κάποιο άλλο
MTU_n	η μέγιστη ποσότητα ανά πακέτο, του δικτύου n
TCPOvhd	επιπρόσθετα δεδομένα (header) του πρωτοκόλλου TCP σε bytes
IPOvhd	επιπρόσθετα δεδομένα (header) του πρωτοκόλλου IP σε bytes
$FrameOvhd_n$	επιπρόσθετα δεδομένα των πακέτων στο δίκτυο n, σε bytes
$Overhead_n$	συνολικά επιπρόσθετα δεδομένα (όλων των δικτύων) των πακέτων στο δίκτυο n, σε bytes
$Bandwidth_n$	εύρος ζώνης του δικτύου n
NDatagrams	αριθμός IP πακέτων για τη μεταφορά μιας ποσότητας δεδομένων
N	αριθμός δικτύων, που μεσολαβούν μεταξύ του πελάτη και του διακομιστή

Αν ο αποστολέας δημιουργεί πακέτα, των οποίων το μέγεθος δεν ξεπερνά την ελάχιστη MTU όλων των *N* δικτύων της διαδρομής, τότε ο συνολικός αριθμός των πακέτων, που δημιουργούνται για την αποστολή μιας ποσότητας δεδομένων δίνεται από τη σχέση,

$$NDatagrams = \left\lceil \frac{MessageSize + TCPOvhd}{\min_{n=1}^{N} MTU_n - IPOvhd} \right\rceil$$

Έτσι, τα συνολικά επιπρόσθετα δεδομένα, που διαθέτει η προς αποστολή ποσότητα πληροφοριών, όταν αυτή ταξιδεύει μέσα στο δίκτυο n είναι

$$Overhead_n = TCPOvhd + NDatagrams \times (IPOvhd + FrameOvhd_n)$$

Τέλος, ο χρόνος εξυπηρέτησης για την προώθηση της ποσότητας δεδομένων μέσα από το δίκτυο n, είναι

$$ServiceTime_n = \frac{8 \times (MessageSize + Overhead_n)}{10^6 \times Bandwidth_n}$$

Πίνακας 5.3 Χαρακτηριστικά διαφόρων πρωτοκόλλων δικτύου

Πρωτόκολλο	Μέγιστο Μέγεθος Πακέτου (bytes)	Επιπρόσθετα δεδομένα (bytes)	Καθαρή χωρητικότητα δεδομένων ανά πακέτο (bytes)
TCP	65535	20	65515
UDP	*	8	*
IPv4	65535	20	65515
IPv6	65535	40	65495
ATM	53	5	48
Ethernet	1518	18	1500
IEEE 802.3	1518	21	1497
IEEE 802.5 Token Ring	4472	28	4444
FDDI (RFC 1390)	4500	28	4472

* περιορίζεται από το μέγεθος πακέτου του χρησιμοποιούμενου πρωτοκόλλου IP

Όσον αφορά την περίπτωση κατά την οποία οι δρομολογητές τεμαχίζουν τα δεδομένα, όταν χρειάζεται για το πέρασμα από δίκτυο σε δίκτυο, ο αναγνώστης μπορεί να αναφερθεί στο [MA98] για περισσότερες λεπτομέρειες. Στον πίνακα 5.3 δίνονται στοιχεία για τα μεγέθη των πακέτων και τα επιπρόσθετα δεδομένα, που χρησιμοποιούν μερικά από τα πιο διαδεδομένα πρωτόκολλα.

Μία άλλη ενδιαφέρουσα και ίσως περισσότερο ακριβής προσέγγιση [AL79], για τη μοντελοποίηση ενός τοπικού δικτύου Ethernet, επιχειρεί να εκφράσει την καθυστέρηση στη μεταφορά δεδομένων με κέντρο, που εξαρτά τις απαιτήσεις εξυπηρέτησης από το εύρος και την αποδοτικότητα του δικτύου. Η αποδοτικότητα με τη σειρά της εξαρτάται από τον αριθμό των σταθμών του δικτύου και το χρόνο *ταξιδιού - κύκλου* (Round - Trip Time), που είναι μέγεθος μετρήσιμο.

5.3.5 Σχεδιαστικά υποδείγματα και απόδοση λογισμικού αρχιτεκτονικής CORBA

Στις προηγούμενες ενότητες αναλύθηκαν τα φαινόμενα, που είναι σημαντικά και επιβάλλεται η αναπαράστασή τους σε ένα μοντέλο απόδοσης λογισμικού τεχνολογίας CORBA. Από αυτά, που εκτέθηκαν, είναι σαφές ότι η ανάλυση επικεντρώθηκε στην αναπαράσταση:

- της απόδοσης της αλληλεπίδρασης τύπου πελάτη - υλοποίηση αντικειμένου σε περιβάλλον CORBA και

- των εναλλακτικών επιλογών σχεδίασης, που από ότι φαίνεται είναι κοινό σημείο αναφοράς των μηχανικών λογισμικού, μια και αυτές κωδικοποιούνται και δημοσιεύονται με τη μορφή σχεδιαστικών υποδειγμάτων.

Έτσι για παράδειγμα οι περιπτώσεις πολυνημάτωσης, που αναφέρθηκαν, κωδικοποιήθηκαν και δημοσιεύθηκαν στην [SP97], ενώ η περίπτωση της επιστροφής κλήσης δημοσιεύθηκε πολλές φορές και στο τέλος συμπεριλήφθηκε και στις προδιαγραφές [OMG98] .

Άλλα σχεδιαστικά υποδείγματα επικοινωνίας αντικειμένων, που δεν εξετάσθηκαν, είναι αυτά της μορφής πελάτη - διαμεσολαβητή - υλοποίηση αντικειμένου. Οι αρχιτεκτονικές αυτές χρησιμοποιούνται, όταν είναι επιθυμητός ο διαχωρισμός των λειτουργιών ολοκλήρωσης και κατανεμημένου ελέγχου από αυτές, που αφορούν τη βασική λειτουργικότητα της εφαρμογής. Τέτοια σχεδιαστικά υποδείγματα περιγράφονται στην [ADL95] και η απόδοσή τους μελετάται με μετρήσεις στην [AM98] και στρωματοποιημένα δίκτυα ουρών (LQNs) στην [PAM00].

Η τάση αφαίρεσης όμως περιπτώσεων επικοινωνίας αντικειμένων και κωδικοποίησής των με τη μορφή σχεδιαστικών υποδειγμάτων θέτει τις βάσεις για έναν ανάλογο προβληματισμό, όσον αφορά την απόδοση αυτών. Πιο συγκεκριμένα, τα ερωτήματα, που προκύπτουν και χρήζουν επισταμένης μελέτης, έχουν ως εξής:

- Είναι εφικτή η αφαίρεση των χαρακτηριστικών απόδοσης των περιπτώσεων σχεδίασης, που κωδικοποιούνται με τη μορφή σχεδιαστικών υποδειγμάτων, και αν ναι, με ποιο τρόπο πρέπει αυτά να περιγραφούν;
- Είναι εφικτή η αφαίρεση αποτιμήσεων της απόδοσης σχεδιαστικών υποδειγμάτων, βάσει διαδικασιών ανάπτυξης μεταμοντέλων, όπως αυτές του κεφαλαίου 6 και αν ναι, ποια η χρησιμότητά της;

Μία συστηματική προσέγγιση προς την κατεύθυνση, που οριοθετούν τα δύο παραπάνω ερωτήματα θα μείωνε δραματικά το κόστος της μελέτης απόδοσης. Τα νέας μορφής σχεδιαστικά υποδείγματα μαζί με τα μεταμοντέλα, που θα τα συνοδεύουν, θα προσφέρουν ουσιαστικά αποτιμήσεις λύσεων σχεδίασης, ανεξάρτητες της υποκείμενης πλατφόρμας εκτέλεσης. Αποτιμήσεις, που είτε θα μπορούν να προσαρμόζονται και να αποτελούν τη βάση για τη διεξαγωγή μιας ή περισσοτέρων αναλύσεων, όπως αυτές που αναφέρονται στην ενότητα 5.2.3, είτε θα μπορούν να επαναχρησιμοποιούνται για τη δόμηση πιο σύνθετων μοντέλων. Η απόσβεση βέβαια του αρχικά μεγάλου κόστους αφαίρεσης και κωδικοποίησης χαρακτηριστικών απόδοσης θα γίνεται με την σε ευρεία κλίμακα επαναχρησιμοποίηση των σχεδιαστικών υποδειγμάτων, στα οποία αναφέρονται.

5.3.6 Άλλες προσεγγίσεις – Σχετικές εργασίες

Ένα σημαντικό πρόβλημα στην ανάπτυξη μοντέλων για την απόδοση κατανεμημένων συστημάτων τεχνολογίας CORBA είναι η πολυπλοκότητα, που αναπόφευκτα αυτά έχουν εξαιτίας της δομής των συστημάτων. Όπως είδαμε στις προηγούμενες ενότητες, ακόμη και μία μόνο κλήση μεθόδου, στο επίπεδο της εφαρμογής, έχει ως αποτέλεσμα την ενεργοποίηση σημαντικού αριθμού επεξεργασιών λογισμικού και την προσπέλαση μεγάλου αριθμού πόρων. Έτσι, η περιγραφή ενός δικτύου ουρών, που με ικανοποιητικό τρόπο εκφράζει την απόδοση του συστήματος, απαιτεί την ολοκλήρωση μιας εκτεταμένης ανάλυσης της ροής εκτέλεσης των έργων στην πορεία τους για λήψη των απαιτήσεων εξυπηρέτησης από τους πόρους του συστήματος.

Στην παράγραφο αυτή φάνηκε ξεκάθαρα η αναγκαιότητα χρήσης μιας ιεραρχικής (στρωματοποιημένης κατά άλλους) μοντελικής προσέγγισης, υποστηριζόμενης από διαδικασίες σταδιακής περιγραφής της αρχιτεκτονικής, της επεξεργασίας της εφαρμογής, της

διαμόρφωσης και τέλος του φόρτου. Η προτεινόμενη αντιμετώπιση του προβλήματος παρουσιάζει εμφανείς αναλογίες με άλλες, όπως π.χ. αυτή που περιγράφεται στη [SAD00] και στην οποία έχει ήδη γίνει αναφορά και αυτή που αναπτύσσεται στη [KAH00] και συνοψίζεται στη [KAH01].

Στη δεύτερη περίπτωση, για την περιγραφή των μοντέλων απόδοσης, γίνεται χρήση της UML. Οι διαδικασίες, που προτείνονται, αποσκοπούν στη δόμηση των μοντέλων, έτσι ώστε σταδιακά αυτά να αναπαριστούν το σύστημα στα επίπεδα: (i) της εφαρμογής, (ii) της διασύνδεσης, (iii) της συμπεριφοράς, (iv) της υποδομής, (v) του δικτύου και (vi) της διαμόρφωσης. Το μοντέλο απόδοσης, που προκύπτει, μετατρέπεται αυτόματα σε «εμπλουτισμένο δίκτυο ουρών» (Augmented Queuing Network), όπου κάθε περίπτωση πόρου του συστήματος αναπαριστάται άμεσα. Στη συνέχεια, με τη χρήση ενός αλγορίθμου, το «εμπλουτισμένο δίκτυο ουρών» αποσυντίθεται σε πολλαπλά δίκτυα μορφής γινομένου με περισσότερους του ενός τύπους έργων, τα οποία επιλύονται ξεχωριστά και παράλληλα. Τέλος, η συνολική αποτίμηση γίνεται με τη χρήση ενός αλγορίθμου για μικτά δίκτυα μορφής γινομένου, που προτείνεται στο [MAD94].

Τα «εμπλουτισμένα δίκτυα ουρών» είναι ουσιαστικά μοντέλα στρωματοποιημένης αναπαράστασης. Η χρήση τους υπαγορεύεται από την ανάγκη μοντελοποίησης χαρακτηριστικών, που δε συμμορφώνονται με τις απαιτήσεις για δίκτυα μορφής γινομένου. Τέτοιο είναι, για παράδειγμα, η περίπτωση της ταυτόχρονης κατοχής πόρων, που χρησιμοποιείται στην αναπαράσταση σύγχρονων κλήσεων (βλ. σχήμα 5.5) και στην αναπαράσταση αρχιτεκτονικών πολυνημάτωσης (βλ. σχήμα 5.8). Παρόλα αυτά, επειδή ακριβώς η συγκεκριμένη τεχνική είναι προσκολλημένη στην προσεγγιστική επίλυση του μοντέλου, δεν επιτρέπει τη χρήση επιπλέον χαρακτηριστικών, όπως για παράδειγμα αυτό της διάσπασης/ένωσης (βλ. σχήμα 5.7), που επίσης πιστεύουμε ότι είναι απαραίτητο σε περιπτώσεις, όπως αυτή της κλήσης επιστροφής. Βέβαια, όταν χρησιμοποιούνται τέτοια χαρακτηριστικά, αυτό γίνεται σε βάρος του χρόνου αποτίμησης του μοντέλου, καθώς τότε αυτή μπορεί να γίνει είτε εξολοκλήρου προσομοιωτικά, είτε μέσω αποσύνθεσης με αναλυτική επίλυση υπομοντέλων, όπου αυτό είναι εφικτό.

Μία άλλη διαφορά στις δύο προσεγγίσεις είναι το γεγονός ότι, ενώ αυτή της [KAH00] εκμεταλλεύεται τη περιγραφική ευελιξία της UML για μία αναπαράσταση πιο κοντά στη λειτουργική περιγραφή του συστήματος, η προτεινόμενη προσέγγιση δε διαθέτει ουσιαστικά κάποια συγκεκριμένη γραφική σημειογραφία. Η επιλογή αυτή έγινε προκειμένου να αποφευχθεί η χρήση αυθαίρετων επεκτάσεων της UML, λίγο πριν από την ολοκλήρωση της τυποποίησής της, για τη μοντελοποίηση χαρακτηριστικών απόδοσης [AIR01].

Πάντως, η σημαντική καινοτομία της προσέγγισης, που προτείνεται στη διατριβή αυτή, είναι η ενσωμάτωση διαδικασιών ανάλυσης της απόδοσης με μοντέλα, που περιγράφουν το όλο σύστημα ή υποσυστήματα αυτού σε διαφορετικά επίπεδα αφαίρεσης. Τέτοια χαμηλού επιπέδου μοντέλα μπορούν για παράδειγμα να αναπτύσσονται σε ένα περιβάλλον, όπως αυτό του εργαλείου HIT, ενώ ως υψηλού επιπέδου μοντέλα μπορούν να θεωρηθούν οι οποιασδήποτε μορφής λειτουργικές προσεγγίσεις, αναλυτικά μοντέλα ή και μεταμοντέλα, που δημιουργούνται σύμφωνα με τις διαδικασίες, που καθορίζονται στο κεφάλαιο 6 της διατριβής. Με τη χρήση αυτών των υψηλού επιπέδου μοντέλων διευκολύνονται οι διαδικασίες βελτίωσης της σχεδίασης και ανάλυσης κλιμάκωσης του συστήματος.

Παρόλα αυτά, η προτεινόμενη προσέγγιση απέχει πολύ από τα επίπεδα συστηματοποίησης και αυτοματισμού της μεθόδου που αναπτύσσεται στη [KAH00]. Στον πίνακα 5.4 παρατίθεται μία συγκριτική σύνοψη με τα χαρακτηριστικά των μοντελικών προσεγγίσεων ανάλυσης συστημάτων κατανεμημένης αρχιτεκτονικής, στις οποίες έγινε αναφορά.

Πίνακας 5.4 Συγκριτική σύνοψη μοντελικών προσεγγίσεων απόδοσης συστημάτων κατανεμημένης αρχιτεκτονικής

	Layered Queueing Networks ([RS95], [WOO88] & [WNP95])	HELIOS ([SAD00])	[KAH00]	C. Smith et al [SW98a] & [SW98b]	Προτεινόμενη προσέγγιση & διαδικασίες ανάπτυξης μοντέλων
Συνάφεια με τη μοντελική προσέγγιση λειτουργικής περιγραφής (π.χ. UML)	ΟΧΙ	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΟΧΙ
Συστηματοποίηση και ολοκληρωμένη μεθοδολογική συγκρότηση	ΟΧΙ	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΟΧΙ
Λογισμικό υποστήριξης (εργαλείο)	ΝΑΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ
Μοντελική ευελιξία	ΜΙΚΡΗ	ΜΕΓΑΛΗ	ΜΙΚΡΗ	ΜΕΓΑΛΗ	ΜΕΓΑΛΗ
Αναλυτική αποτίμηση	ΝΑΙ	ΝΑΙ	ΝΑΙ	?	ΝΑΙ
Προσομοιωτική αποτίμηση	ΝΑΙ	ΝΑΙ	ΟΧΙ	ΝΑΙ	ΝΑΙ
Υβριδική & τμηματική αποτίμηση	?	?	ΟΧΙ	?	ΝΑΙ
Επιλεκτική αποτίμηση (φιλτράρισμα φόρτου)	?	?	ΟΧΙ	?	ΝΑΙ

Στον πίνακα αυτό γίνεται επίσης αναφορά και στη μοντελική προσέγγιση C. Smith et al, που αποτελεί ουσιαστικά μία προσπάθεια προσαρμογής της συστηματοποιημένης μεθόδου, που αναπτύσσεται στο [SMI90], στα προβλήματα της ανάλυσης συστημάτων κατανεμημένης αρχιτεκτονικής. Η συγκεκριμένη προσέγγιση περιγράφεται στις [SW98a], [SW98b] και [SW01] και τα βασικά χαρακτηριστικά της είναι:

- Περιγραφή μιας περίπτωσης σεναρίου χρήσης μέσω ενός διαγράμματος αλληλουχίας μηνυμάτων (Message Sequence Chart) της UML.
- Μετάφραση του MSc σε διάγραμμα εκτέλεσης λογισμικού (software execution graph) με άμεση αναπαράσταση των περιπτώσεων συγχρονισμού λόγω κλήσεων αντικειμένων. Από ότι φαίνεται δεν υπάρχει αυτοματοποιημένη υποστήριξη για τη διαδικασία αυτή.
- Δήλωση απαιτήσεων πόρων και καθυστερήσεων επεξεργασίας σε κάθε βήμα της ροής του διαγράμματος εκτέλεσης.
- Επίλυση του διαγράμματος εκτέλεσης λογισμικού για τη λήψη αρχικών εκτιμήσεων καλύτερης - χειρότερης περίπτωσης, χωρίς να λαμβάνεται υπόψη η επίδραση του ανταγωνισμού για χρήση των πόρων.
- Κατασκευή και επίλυση του μοντέλου εκτέλεσης συστήματος, που δίνει ακριβέστερη αποτίμηση των κριτηρίων απόδοσης που ενδιαφέρουν.

Όλα τα βήματα, εκτός από την αναπαράσταση του σεναρίου σε MSc και τη μετάφρασή του σε διάγραμμα εκτέλεσης λογισμικού, υποστηρίζονται από το εργαλείο SPE•ED, που αυτοματοποιεί τις διαδικασίες της συγκεκριμένης μοντελικής προσέγγισης.

Τέλος, θα ήταν παράλειψη να μη γίνει αναφορά στη συνεισφορά της [LRS98]. Στην εργασία αυτή οι συγγραφείς συνδυάζουν με πρωτότυπο τρόπο τεχνικές μη γραμμικού προγραμματισμού και αναλυτικής αποτίμησης της απόδοσης, για τον καθορισμό πολιτικών

πολυνημάτωσης, επανάληψης διεργασίων και ενεργοποίησης βάσει δεδομένων ορίων αξιοποίησης μονάδων λογισμικού και συσκευών. Τα όρια αυτά επιβάλλονται από περιορισμούς, που τίθενται λόγου χάρη από το λειτουργικό σύστημα. Έτσι, επίπεδα επανάληψης διεργασιών και πολυνημάτωσης, πέρα από αυτά που υπολογίζονται, όχι μόνο δε βελτιώνουν το χρόνο απόκρισης, αλλά μπορεί να έχουν και αρνητική επίπτωση στην απόδοση του συστήματος.

5.4 Μοντέλα απόδοσης εφαρμογών παγκοσμίου ιστού

Ο παγκόσμιος ιστός αποτελεί ένα πεδίο ραγδαίας ανάπτυξης οικονομικών δραστηριοτήτων. Χιλιάδες επιχειρήσεις πλέον, σε όλο τον κόσμο, επιδιώκουν την εκμετάλλευση της υποδομής αυτής, για τη δημιουργία μιας πιο άμεσης σχέσης με τον καταναλωτή των υπηρεσιών/ προϊόντων τους. Έτσι, μία εφαρμογή παγκοσμίου ιστού με όχι ικανοποιητικούς χρόνους απόκρισης μπορεί να προκαλεί δυσαρέσκεια, απώλεια εσόδων και καταναλωτών, μειωμένη παραγωγικότητα και πλήθος άλλων ανεπιθύμητων συνεπειών.

Μέσα από την παράγραφο αυτή γίνεται προσπάθεια να αναδειχθεί η πρακτική σημασία της ιεραρχικής μοντελικής προσέγγισης και του πλαισίου διαδικασιών, που παρουσιάστηκαν στις παραγράφους 5.1 και 5.2, για την ανάλυση της απόδοσης εφαρμογών παγκοσμίου ιστού ή/και τον προσδιορισμό μεγέθους των διακομιστών τους.

Μία εφαρμογή παγκοσμίου ιστού, σήμερα, μπορεί να βασισθεί σε ένα πλήθος επιλογών υλοποίησης, που κάθε μία από αυτές επηρεάζει με διαφορετικό τρόπο την αρχιτεκτονική σχεδίαση αυτής. Τέτοιες είναι:

- Το αν η εφαρμογή θα εκτελείται στο διακομιστή και ποια τεχνολογία είναι η πλέον κατάλληλη (οι διεργασίες CGI, οι σελίδες ASP, οι Java μικροεφαρμογές servlets ή η χρήση της προγραμματιστικής διασύνδεσης του διακομιστή υλοποίησης);
- Αν μέρος της εφαρμογής θα εκτελείται στον πελάτη και ποια τεχνολογία θα χρησιμοποιηθεί για το σκοπό αυτό (κάποια γλώσσα σεναρίων, μικροεφαρμογές applets ή κάτι άλλο);
- Αν θα γίνει χρήση κάποιας τεχνολογίας επαναχρησιμοποιήσιμων τμημάτων λογισμικού (των Enterprise Java Beans της Sun ή ActiveX controls της Microsoft);
- Ο τρόπος προσπέλασης σε σχεσιακές βάσεις δεδομένων (εξ αποστάσεως εκτέλεση SQL εντολών, αποθηκευμένες διαδικασίες στο σύστημα διαχείρισης βάσης ή κάποιο προϊόν middleware);

Οι διακομιστές ιστοσελίδων σήμερα είναι απαραίτητο να διαθέτουν δυνατότητα ταυτόχρονης επεξεργασίας κλήσεων εξυπηρέτησης και αποδοτική παρακράτηση (caching) αρχείων.

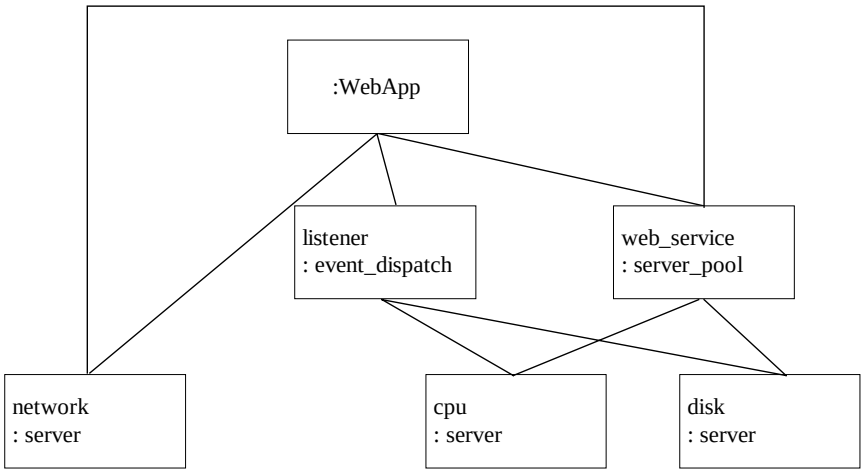
Οι κλήσεις, που φθάνουν σε ένα διακομιστή ιστοσελίδων, παραλαμβάνονται από έναν αποστολέα συμβάντων, που δεν είναι τίποτε άλλο από μία μηχανή αποσυμπλοκής συμβάντων, η οποία δέχεται αιτήσεις TCP συνδέσεων και συντονίζει τα χειριστήρια των sockets και τα νήματα ροής, που χρησιμοποιούνται για την παραλαβή και επεξεργασία των κλήσεων GET του πρωτοκόλλου HTTP. Κάθε τέτοια κλήση επεξεργάζεται από ένα χειριστή πρωτοκόλλου, ο οποίος αναλύει και καταγράφει την κλήση, εξάγει πληροφορίες κατάστασης αρχείου, ενημερώνει την *περιοχή παρακράτησης*, μεταφέρει το αρχείο στον πελάτη, ο οποίος το ζήτησε, και απελευθερώνει τους πόρους, που χρησιμοποίησε.

Κάποια από τα χαρακτηριστικά σχεδίασης ενός διακομιστή, που σαφώς επηρεάζουν την απόδοση αυτού, είναι:

- Η επιλογή του μοντέλου συγχρονισμού, το οποίο μπορεί για παράδειγμα να έχει τη μορφή νήματος ροής ανά κλήση ή μία παραλλαγή συνόλου νημάτων ροής.
- Η επιλογή του μοντέλου αποσυμπλοκής συμβάντων, δηλαδή σύγχρονη ή ασύγχρονη εκτέλεση.
- Η επιλογή του μοντέλου παρακράτησης αρχείων, που μπορεί για παράδειγμα να βασίζεται στην πολιτική του πιο πρόσφατα χρησιμοποιημένου ή του πιο συχνά χρησιμοποιημένου αρχείου.
- Η επιλογή του πρωτοκόλλου μεταφοράς περιεχομένου, το οποίο μπορεί για παράδειγμα να είναι το HTTP/1.0 ή το HTTP/1.1.

Τα χαρακτηριστικά σχεδίασης ενός διακομιστή ιστοσελίδων θα πρέπει να είναι προσαρμοσμένα στις ανάγκες των τελικών χρηστών και στον προβλεπόμενο υπολογιστικό φόρτο. Δεν υπάρχει συγκεκριμένη διαμόρφωση, η οποία να είναι η βέλτιστη για όλες τις πλατφόρμες και για όλα τα πιθανά επίπεδα υπολογιστικού φόρτου. Ο στόχος, όσον αφορά την απόδοση ενός διακομιστή ιστοσελίδων, είναι η επίτευξη υψηλών επιπέδων παραγωγής και δυνατοτήτων κλιμάκωσης και χαμηλών επιπέδων καθυστέρησης της απόκρισης.

Παρόλα αυτά, συχνά, ακόμη και ένας καλά προσαρμοσμένος διακομιστής ιστοσελίδων είναι αδύνατο να ανταποκριθεί σε ιδιαίτερα υψηλά επίπεδα φόρτου, τα οποία όμως μπορεί να αποτελούν μία πραγματικότητα, που κανείς δε μπορεί να αγνοήσει. Στις περιπτώσεις αυτές επιλέγεται η λύση της ανάπτυξης *συστάδας διακομιστών* (Web server clustering) με πανομοιότυπο περιεχόμενο. Κατά τη διαμόρφωση αυτή, όταν φθάνει κάποια κλήση εξυπηρέτησης, η *υπηρεσία ονομάτων πεδίου* επιλέγει κάποια από τις διευθύνσεις IP της συστάδας των διακομιστών. Αν η πολιτική που εφαρμόζεται είναι η «round-robin», τότε γίνεται προσπάθεια καταμερισμού φόρτου επιλέγοντας κάθε φορά την επόμενη IP διεύθυνση στη λίστα των διακομιστών της συστάδας.



Σχήμα 5.10 Μία πιθανή ιεραρχία υλοποίησης HIT ενός απλού μοντέλου απόδοσης εφαρμογής παγκοσμίου ιστού

Στο σχήμα 5.10 απεικονίζεται μία ιεραρχία υλοποίησης, στο HIT, ενός μοντέλου απόδοσης εφαρμογής παγκοσμίου ιστού. Το τμήμα `listener` του τύπου `event_dispatcher` παρέχει αποκλειστικά την υπηρεσία `accept`, η οποία αναπαριστά τη διαχείριση των TCP συνδέσεων στο διακομιστή ιστοσελίδων. Το τμήμα `web_service` του τύπου

`service_pool` αναλαμβάνει την εξυπηρέτηση της HTTP κλήσης ανάλογα με τις δυνατότητες ταυτόχρονης εξυπηρέτησης κλήσεων. Στην απλούστερη περίπτωση, παρέχει υπηρεσίες μεταφοράς αρχείων εικόνας, αρχείων HTML ή δημιουργίας και επεξεργασίας διεργασιών CGI, που βεβαίως χρησιμοποιούν τους πόρους υλικού του συστήματος με διαφορετικό τρόπο η κάθε μία.

Η παράθεση επιπλέον λεπτομερειών ενός μοντέλου εφαρμογής παγκοσμίου ιστού δεν είναι σημαντική για την αποσαφήνιση της μοντελικής προσέγγισης και του πλαισίου διαδικασιών, που προτείνονται, καθώς αυτές περιγράφηκαν εκτενώς στις παραγράφους 5.1 και 5.2. Αντίθετα, είναι ενδιαφέρον να γίνει αναφορά στα σημαντικότερα προβλήματα, που χρειάζεται να αντιμετωπίσει η ανάπτυξη των μοντέλων επεξεργασίας και φόρτου για την επιτυχή παραμετροποίηση ενός μοντέλου απόδοσης.

Ο προσδιορισμός της κίνησης, που αντιμετωπίζει ένας διακομιστής ιστοσελίδων, γίνεται μέσα από μοντέλα μελέτης της συμπεριφοράς των χρηστών, που καταδεικνύουν τις προτιμήσεις τους, όσον αφορά τον αριθμό επισκέψεων και το μέγεθος των καλούμενων ιστοσελίδων. Ένα μέσο ομαδοποίησης των συμπεριφορών αυτών είναι τα *προφίλ χρήσης*, αναλογικοί συνδυασμοί των οποίων συγκροτούν τελικά αντιπροσωπευτικούς φόρτους για το μοντέλο απόδοσης, που μελετάται. Ο κάθε φόρτος εκφράζεται ως γνωστό, στο HIT, με την κλήση υπηρεσιών, που παρέχονται στο υψηλότερο στρώμα του μοντέλου, σε αριθμούς και διαστήματα, που τον εκφράζουν. Μία περίπτωση εμπεριστατωμένης περιγραφής προφίλ χρήσης μιας τυπικής εφαρμογής ηλεκτρονικού εμπορίου αποτελεί το τυποποιημένο benchmark TPC W [TPC00] της σύμπραξης Transaction Processing Performance Council.

Παρόλα αυτά, είναι σημαντικό να τονιστεί το γεγονός ότι ο χρόνος απόκρισης, που τελικά αποτιμάται από ένα μοντέλο απόδοσης, όπως αυτό του σχήματος 5.10, αντιστοιχεί στην εξυπηρέτηση μίας μόνο HTTP κλήσης και όχι στη λήψη μιας ολόκληρης ιστοσελίδας. Είναι εξάλλου γνωστό ότι, αν μία HTML σελίδα περιέχει έστω και μία εικόνα, τότε προκαλεί την αποστολή επιπλέον HTTP κλήσεων. Το γεγονός αυτό είτε πρέπει να ληφθεί υπόψη κατά την εισαγωγή του φόρτου στο μοντέλο απόδοσης, είτε θα πρέπει να γίνει ομαδοποίηση των HTTP κλήσεων ανά χρήστη προέλευσης κατά τις μετρήσεις υπολογισμού των απαιτήσεων εξυπηρέτησης. Το δεύτερο βέβαια δεν είναι πάντα εφικτό, ειδικά στην περίπτωση κατά την οποία μέρος των HTTP κλήσεων περνάει μέσα από λογισμικό προστασίας πρόσβασης (firewall), οπότε αυτό έχει ως συνέπεια την εμφάνιση προέλευσής τους από τη ίδια IP διεύθυνση, πληροφορία όμως που δεν είναι αληθής. Μία ιδιαίτερα σημαντική ερευνητική συνεισφορά στη μέτρηση και την επεξεργασία των μετρήσεων για την παραμετροποίηση ενός μοντέλου LQN είναι αυτή, που αναπτύσσεται από τους συγγραφείς της [DFJ98].

Κεφάλαιο 6 *Προσαρμογή Μεταμοντέλων και Ανάλυση Ευαισθησίας*

Ο όρος *μεταμοντέλο* έχει χρησιμοποιηθεί στη σχετική με τη μελέτη της απόδοσης συστημάτων βιβλιογραφία με περισσότερες της μιας έννοιες. Στη διατριβή αυτή, ο συγκεκριμένος όρος αναφέρεται σε όλα εκείνα τα εργαλεία της στατιστικής, που εκφράζουν με ευσύνοπτο τρόπο την επίδραση ενός συνόλου μεταβλητών (*παράγοντες*) στις τιμές ενός συνόλου δεδομένων. Σε μία προσομοιωτική μελέτη, η αποτελεσματική εφαρμογή διαδικασιών προσαρμογής μεταμοντέλων είναι προαπαιτούμενο, τόσο για την πρόγνωση των απαιτήσεων εξυπηρέτησης από τους πόρους ενός μοντέλου απόδοσης (*συναρτήσεις πόρων*), όσο και για τη διεξαγωγή μιας σειράς αναλύσεων (ανάλυση ορίων, παραμετρική ανάλυση και ανάλυση κλιμάκωσης), που περιγράφονται συνοπτικά στην ενότητα 5.2.3. Στο παρόν κεφάλαιο, θα αναφερόμαστε σε όλες τις αναλύσεις αυτού του είδους με το γενικό όρο *ανάλυση ευαισθησίας*. Γίνεται σύντομη αναφορά σε μεγάλης πρακτικής σημασίας εργαλεία της στατιστικής και στη χρήση τους στη μελέτη της απόδοσης συστημάτων. Επίσης, περιγράφεται μία ολοκληρωμένη εφαρμογή χρήσης αυτών, που μπορεί να αποτελέσει μία αρχική βάση για την ανάπτυξη ενός συστηματοποιημένου πλαισίου διαδικασιών για την ασφαλή και αποτελεσματική χρήση τους.

6.1 Ανάλυση παλινδρόμησης και πειραματική σχεδίαση

Στην παράγραφο αυτή γίνεται μία συνοπτική περιγραφή τεχνικών στατιστικής επεξεργασίας, που είναι δυνατό να χρησιμοποιηθούν:

- Στην πρόγνωση των απαιτήσεων εξυπηρέτησης από τους πόρους ενός μοντέλου απόδοσης (*συναρτήσεις πόρων*).
- Στην ερμηνεία των επιδράσεων των μεταβλητών εισόδου ενός μοντέλου (*ανεξάρτητες μεταβλητές* ή *παράγοντες*) στα μέτρα απόδοσης, που ενδιαφέρουν.
- Στην πρόγνωση της απόδοσης ενός μοντέλου σε υποθετικές καταστάσεις και στην ανάλυση σεναρίων σχετικά με τη σχεδίαση του συστήματος, που αυτό αντιπροσωπεύει.
- Στον προσδιορισμό της βέλτιστης δομής του συστήματος λαμβάνοντας υπόψη μία σειρά περιορισμών στη διάθεση των απαραίτητων πόρων.

Οι μεταβλητές διακρίνονται σε *ποιοτικές* και *ποσοτικές*. Ποιοτικές είναι οι μεταβλητές, που τους αποδίδονται ονομαστικές μόνο τιμές, όπως για παράδειγμα οι πειθαρχίες εξυπηρέτησης. Όλες οι υπόλοιπες μεταβλητές χαρακτηρίζονται ως ποσοτικές.

Μία διαδεδομένη τεχνική για τη διερεύνηση της *σημαντικότητας* του κάθε παράγοντα είναι η προσαρμογή ενός γραμμικού μοντέλου πρόβλεψης με τη μέθοδο των ελαχίστων τετραγώνων και η σύγκριση της συνεισφοράς του κάθε παράγοντα, στη διασπορά της εξαρτημένης μεταβλητής, σε σχέση με τη συνεισφορά του στα παρατηρούμενα *σφάλματα*. Αν η διασπορά, που εξαιτίας των σφαλμάτων δε μπορεί να ερμηνευθεί, κυμαίνεται σε υψηλά επίπεδα, ακόμη και ένας παράγοντας, με μεγάλη συνεισφορά στη διασπορά της εξαρτημένης μεταβλητής, στατιστικώς θεωρείται ως μη σημαντικός. Η τεχνική αυτή είναι γνωστή ως *ανάλυση διασποράς* (ANalysis Of VAriance) και είναι δυνατό να χρησιμοποιηθεί στον εντοπισμό των παραγόντων, που μπορούν να θεωρηθούν σημαντικοί για μία εξαρτημένη μεταβλητή, κάτω από τις ακόλουθες προϋποθέσεις:

- Οι παράγοντες χρησιμοποιούνται ως ποιοτικές μεταβλητές.
- Θεωρούμε ότι η εξάρτηση της μεταβλητής εξόδου είναι γραμμική ως προς τους παράγοντες.
- Τα παρατηρούμενα σφάλματα είναι αθροιστικά, ανεξάρτητα από τις τιμές των παραγόντων (*στάθμες*) και κατανεμημένα κανονικά και με την ίδια διασπορά για όλες τις τιμές αυτών.

Οι προϋποθέσεις αυτές μπορούν να ελεγχθούν, είτε μέσω των κατάλληλων γραφημάτων ([JAI91]), είτε με τη χρήση των κατάλληλων στατιστικών τεστ.

Η ANOVA αποτελεί ένα σημαντικό στατιστικό εργαλείο για τον εντοπισμό όχι μόνο των παραγόντων, αλλά και των μεταξύ τους *αλληλεπιδράσεων*, που είναι δυνατό να θεωρηθούν ως στατιστικώς σημαντικές. Τέτοιες αλληλεπιδράσεις παρατηρούνται, όταν η επίδραση ενός παράγοντα στην εξαρτημένη μεταβλητή εξαρτάται και από τις τιμές άλλων παραγόντων.

Αν όμως η ANOVA έχει ως στόχο απλά τον εντοπισμό των παραγόντων, με τη στατιστικώς σημαντική επίδραση στις τιμές μιας εξαρτημένης μεταβλητής, η *ανάλυση παλινδρόμησης* (regression analysis) αποφέρει και την εκτίμηση του μεγέθους της επίδρασης αυτής. Η τεχνική αυτή βασίζεται στην προσαρμογή ενός κατάλληλης μορφής *μεταμοντέλου πρόβλεψης*. Ας θεωρήσουμε αρχικά την απλή περίπτωση του γραμμικού μοντέλου:

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + ... + \beta_k \cdot X_k + \varepsilon \qquad (1)$$

όπου: Y είναι μία μεταβλητή εξόδου του μοντέλου απόδοσης, που θα χαρακτηρίζεται ως *εξαρτημένη μεταβλητή*

$X_1, X_2, ..., X_k$ οι k *παράγοντες* ή *ανεξάρτητες* ή αλλιώς *προβλέπουσες* μεταβλητές

$\beta_0, \beta_1, ..., \beta_k$ είναι (k+1) άγνωστες *παράμετροι*, που ζητείται να εκτιμηθούν

ε είναι το σφάλμα του μεταμοντέλου πρόβλεψης, που οφείλεται είτε σε παράλειψη μεταβλητών, είτε σε χρήση μεταβλητών, που δε σχετίζονται με την Y

Οι μεταβλητές $X_1, X_2, ..., X_k$ βεβαίως σε καμία περίπτωση δε μπορούν να θεωρηθούν ως στατιστικώς ανεξάρτητες. Μπορεί για παράδειγμα να ισχύει $X_2 = X_1^2$ ή $X_3 = X_1 + X_2$, αλλά οι μεταβλητές αυτές χαρακτηρίζονται ως ανεξάρτητες, επειδή ακριβώς η τιμή τους καθορίζεται από τον αναλυτή και για το λόγο αυτό λέγονται και προβλέπουσες. Η δυνατότητα αυτή των προβλεπουσών μεταβλητών, να συσχετίζονται μεταξύ τους, διευρύνει

τις περιπτώσεις εφαρμογής του μοντέλου, καθώς με κατάλληλους μετασχηματισμούς «μη γραμμικά» ή και «εκθετικά» μοντέλα ανάγονται στο γενικό γραμμικό μοντέλο (1).

Για τον υπολογισμό των παραμέτρων β_i, i=0, 1, …, k υποθέτουμε ότι τα σφάλματα ε_i είναι τυχαίες μεταβλητές, ασυσχέτιστες και με κοινή διασπορά σ^2. Στην περίπτωση λοιπόν n παρατηρήσεων θα ικανοποιείται το σύστημα:

$$\begin{array}{lllllllllllll} y_1 & = & \beta_0 & + & \beta_1 \cdot x_{11} & + & \beta_2 \cdot x_{21} & + & \ldots & + & \beta_k \cdot x_{k1} & + & \varepsilon_1 \\ y_2 & = & \beta_0 & + & \beta_1 \cdot x_{12} & + & \beta_2 \cdot x_{22} & + & \ldots & + & \beta_k \cdot x_{k2} & + & \varepsilon_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ y_n & = & \beta_0 & + & \beta_1 \cdot x_{1n} & + & \beta_2 \cdot x_{2n} & + & \ldots & + & \beta_k \cdot x_{kn} & + & \varepsilon_n \end{array}.$$

Για τον υπολογισμό των β_i, έτσι ώστε να ελαχιστοποιείται το άθροισμα των τετραγώνων των σφαλμάτων $\sum_{i=1}^{n} \varepsilon_i^2$, η μέθοδος των ελαχίστων τετραγώνων δίνει,

$$\underset{\sim}{\hat{\beta}} = (X^{'} X)^{-1} X^{'} \underset{\sim}{Y} \qquad (2)$$

όπου:

$$\underset{\sim}{Y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_n \end{bmatrix}, \qquad X = \begin{bmatrix} 1 & x_{11} & x_{21} & . & x_{k1} \\ 1 & x_{12} & x_{22} & . & x_{k2} \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & x_{1n} & x_{2n} & . & x_{kn} \end{bmatrix}, \qquad \underset{\sim}{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ \beta_k \end{bmatrix} \text{ και } \qquad \underset{\sim}{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ . \\ . \\ \varepsilon_n \end{bmatrix}$$

Μία από τις προϋποθέσεις για την ανάλυση παλινδρόμησης είναι η σταθερότητα της διασποράς των σφαλμάτων. Όταν η προϋπόθεση αυτή δεν ισχύει, τότε λέμε ότι τα δεδομένα εμφανίζουν *ετεροσκεδαστικότητα* και στις περιπτώσεις αυτές πρέπει να δοκιμάζονται και εναλλακτικές μέθοδοι υπολογισμού των β_i, όπως ([KLE87]):

- η μέθοδος των *σταθμισμένων ελαχίστων τετραγώνων*,
- η μέθοδος των *διορθωμένων ελαχίστων τετραγώνων*,
- η μέθοδος των *κανονικών ελαχίστων τετραγώνων*,

με σκοπό πάντα την εύρεση του καλύτερου γραμμικού αμερόληπτου εκτιμητή $\underset{\sim}{\hat{\beta}}$.

Αν θεωρήσουμε,

$$R^2 = \frac{\sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \qquad (3)$$

τότε το R^2 ονομάζεται *συντελεστής προσδιορισμού*, παίρνει τιμές μεταξύ 0 και 1 και μάλιστα όσο πλησιάζει το 1, τόσο η προσαρμογή του μεταμοντέλου στα δεδομένα είναι καλύτερη. Παρόλο, που το R^2 χρησιμοποιείται συνήθως ως κριτήριο ελέγχου εγκυρότητας του μεταμοντέλου, στην [KS00] τονίζεται ότι δεν πρέπει να εκλαμβάνεται ως τέτοιο και προτείνονται αξιόπιστες τεχνικές, που δίνουν απάντηση στο πρόβλημα.

Αν θεωρήσουμε το μεταμοντέλο,

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + ... + \beta_{i-1} \cdot X_{i-1} + \beta_{i+1} \cdot X_{i+1} + ... + \beta_k \cdot X_k + \varepsilon$$

που προκύπτει για $\beta_i = 0$ και $R^2_{(i)}$ ο συντελεστής προσδιορισμού για το αντίστοιχο μεταμοντέλο, που προσαρμόζεται στο περιορισμένο αυτό μοντέλο, τότε [MM90] το

$$\bar{r}^2_{yi;k} = R^2 - R^2_{(i)} \tag{4}$$

ονομάζεται *επιμέρους συντελεστής προσδιορισμού* του παράγοντα *i*. Ο λόγος *F* που ορίζεται ως

$$F = \frac{(n-k-1) \cdot \bar{r}^2_{yi;k}}{1-\bar{r}^2_{yi;k}} \tag{5}$$

χρησιμοποιείται ως κριτήριο στις πιο γνωστές διαδικασίες σταδιακής επιλογής παραγόντων για την τελική προσαρμογή μεταμοντέλου, που να εξηγεί με ικανοποιητικό τρόπο τη μεταβλητότητα των δεδομένων.

Στην «*προς τα εμπρός επιλογή*», η διαδικασία ξεκινάει από το μεταμοντέλο θέσης $Y = \beta_0 + \varepsilon$ και προχωράει προσθέτοντας σταδιακά μία - μία μεταβλητή, κάθε φορά, εκείνη, που δίνει το μεγαλύτερο μερικό συντελεστή συσχέτισης και με την προϋπόθεση ότι ο λόγος F της μεταβλητής, που πρόκειται να μπει στο μεταμοντέλο, είναι μεγαλύτερος από κάποια κρίσιμη τιμή F_{IN}. Στην «*προς τα πίσω απαλοιφή*» η διαδικασία ακολουθεί την ακριβώς αντίστροφη πορεία, ξεκινώντας από το πλήρες μεταμοντέλο, αυτό δηλαδή με όλους τους παράγοντες και τις αλληλεπιδράσεις και τις δυνάμεις αυτών, αν φυσικά έχει ζητηθεί κάτι τέτοιο. Η διαδικασία της «*βήμα προς βήμα προς τα εμπρός επιλογής*» είναι μια βελτιωμένη παραλλαγή της «*προς τα εμπρός επιλογής*» ξεκινώντας και αυτή από το απλό μεταμοντέλο θέσης και προσθέτοντας σε κάθε βήμα έναν παράγοντα. Η διαφορά είναι ότι με τη μετάβαση στο νέο μεταμοντέλο ελέγχονται κάθε φορά οι ήδη υπάρχουσες μεταβλητές με τη διαδικασία της «*προς τα πίσω απαλοιφής*».

Αν οι προαναφερθείσες διαδικασίες σταδιακής επιλογής παραγόντων δε δώσουν αρκούντως ικανοποιητικά μεταμοντέλα, τότε προχωράμε είτε

- στο μετασχηματισμό μιας ή περισσοτέρων μεταβλητών, είτε
- στην προσαρμογή μεταμοντέλου υψηλότερης τάξης.

Εφόσον σε καμία από τις δύο περιπτώσεις δεν επιτευχθεί κάποια ικανοποιητική προσέγγιση, τότε πρέπει, ή να περιοριστεί η αρχικά επιλεγείσα περιοχή πειραματικής μελέτης (βλ. ενότητα 5.2.1), ή να αναζητηθεί κάποια περισσότερο κατάλληλη τεχνική προσαρμογής μεταμοντέλου.

Η ανάλυση παλινδρόμησης επιτρέπει και τη χρήση ποιοτικών προβλεπουσών μεταβλητών. Αυτές αναπαριστώνται από συνδυασμούς *βουβών μεταβλητών* (dummy variables), που στον αριθμό είναι κατά μία λιγότερες από τις ονομαστικές τιμές της ποιοτικής μεταβλητής.

Οι ερευνητικές προσπάθειες προς την κατεύθυνση της προσαρμογής αξιόπιστων μεταμοντέλων έχουν αποφέρει αποτελέσματα, όπως:

- τα *μεταμοντέλα μορφής spline*, συναρτήσεων δηλαδή, που ορίζονται ως τμηματικά πολυώνυμα με την απαίτηση συνέχειας των παραγώγων τους στα σημεία των ορίων αυτών,
- τα *μεταμοντέλα με χρήση αξονικών συναρτήσεων βάσης* (radial basis functions),

- τα *μεταμοντέλα εξομάλυνσης κεντρικού σημείου* (kernel smoothing metamodels),
- τα *μεταμοντέλα συσχέτισης στο χώρο* (spatial correlation metamodels) και
- οι *συναρτήσεις βάσεις πεδίου συχνοτήτων* (frequency domain basis functions).

Μία ενδιαφέρουσα συγκριτική ανασκόπηση αυτών, όσον αφορά τη χρήση τους για την προσέγγιση αποτελεσμάτων στοχαστικής προσομοίωσης διακριτών γεγονότων, μπορεί κανείς να βρει στη [BAR94].

Το αποτέλεσμα μιας διαδικασίας προσαρμογής μεταμοντέλου είναι ένα μοντέλο υψηλότερου αφαιρετικού επιπέδου (βλ. ενότητα 5.2.3), το οποίο μπορεί να χρησιμοποιείται και να επαναχρησιμοποιείται για τη διεξαγωγή οποιασδήποτε μορφής ανάλυσης ευαισθησίας (ανάλυση ορίων, παραμετρική ανάλυση ή ανάλυση κλιμάκωσης), όπως στο σχήμα 5.4. Απώτερος στόχος είναι πάντα η γρήγορη πρόγνωση της συμπεριφοράς του συστήματος σε ενδεχόμενη αλλαγή παραμέτρου/ων χωρίς να χρειάζεται επιπλέον ανάλυση.

Όσον αφορά τη βελτιστοποίηση του συστήματος ως προς κάποια σαφώς καθορισμένη *συνάρτηση κόστους* και αυτή επιτυγχάνεται μέσω της προσέγγισης της συμπεριφοράς της συνάρτησης κόστους με κάποιο μεταμοντέλο. Ακολούθως εφαρμόζεται, είτε κάποιος ευρεστικός αλγόριθμος, όπως αυτοί, που περιγράφονται στη [BJS86] ή οι επονομαζόμενοι γενετικοί αλγόριθμοι, είτε η γνωστή ως *μεθοδολογία επιφάνειας απόκρισης* (Response Surface Methodology).

Η μεθοδολογία επιφάνειας απόκρισης ([MON91]) είναι μία επαναληπτική διαδικασία, που βασίζεται αρχικά στη χρήση ενός πολυωνυμικού μεταμοντέλου πρώτης τάξης. Οι επιφάνειες, που προκύπτουν από τη μεταβολή των τιμών δύο προβλεπουσών μεταβλητών κρατώντας σταθερές τις υπόλοιπες, ονομάζονται *επιφάνειες απόκρισης*. Αυτές απεικονίζονται γραφικά στο χώρο ή με διάγραμμα ισοϋψών στο επίπεδο. Ακολούθως εφαρμόζεται η κατάλληλη *μέθοδος μεγαλύτερης αλλαγής*, που είναι ο απλούστερος τρόπος για να οδηγηθούμε στην περιοχή του ολικού (για τη συνάρτηση κόστους) βέλτιστου, από την περίπτωση αναφοράς (βλ. σχήμα 5.4), που επιλέχθηκε. Μία ένδειξη προσέγγισης της περιοχής αυτής είναι συνήθως η μη προσαρμογή του μεταμοντέλου πρώτης τάξης, που χρησιμοποιήθηκε. Τότε ακολουθεί η προσαρμογή ενός μεταμοντέλου δεύτερης τάξης - περιορίζοντας την πειραματική περιοχή σε μία συγκριτικά μικρή απόσταση γύρω από το συγκεκριμένο σημείο - και η βελτιστοποίηση ολοκληρώνεται με την ανάλυση αυτού.

Για μεγαλύτερη αποτελεσματικότητα, είναι σημαντικό το σύνολο των περιπτώσεων, που θα επιλεγούν μέσα στην πειραματική περιοχή, να συμμορφώνεται σε κάποια κατάλληλη *πειραματική σχεδίαση* (design of experiments). Ο σκοπός κάθε τέτοιας σχεδίασης είναι η λήψη της μέγιστης δυνατής πληροφορίας από έναν ελάχιστο αριθμό πειραμάτων. Επίσης, συμβάλλει στον εντοπισμό των παραγόντων με τη σημαντικότερη επίδραση στις μεταβολές της εξαρτημένης μεταβλητής, απομονώνοντας τις όποιες τυχαίες μεταβολές λόγω αποκλίσεων στην εκτίμηση (όταν πρόκειται για μετρήσεις για την προσαρμογή συναρτήσεων πόρων) ή λόγω άλλων ανεξέλεγκτων παραγόντων (που δεν έχουν ληφθεί υπόψη στη μελέτη).

Μία *πλήρης παραγοντική σχεδίαση* (full factorial design), κατά την οποία κάθε ένας από τους *k* παράγοντες παίρνει δύο διαφορετικές τιμές (*επίπεδα*), περιλαμβάνει 2^k πειραματικές μονάδες. Παρόλα αυτά, συνήθως χρησιμοποιούνται διάφοροι περιορισμοί της πλήρους παραγοντικής σχεδίασης της μορφής 2^{k-p}, οι οποίοι περιγράφονται διεξοδικά στο [JAI91].

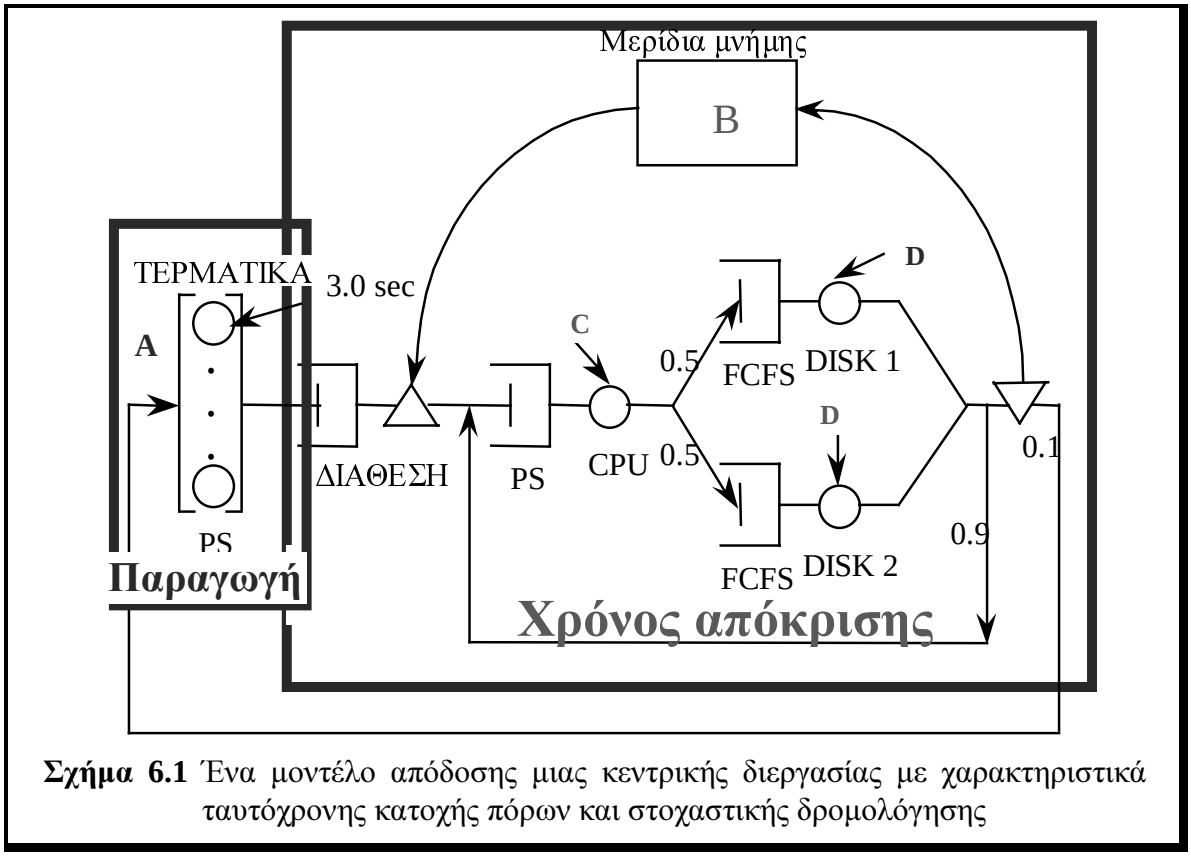
Ανάλογα με τον τύπο πειραματικής σχεδίασης, που επιλέγεται, προεξοφλείται ουσιαστικά η δυνατότητα/μη δυνατότητα μελέτης των αλληλεπιδράσεων δύο ή ακόμη και περισσότερων παραγόντων. Κάποιες περισσότερο προηγμένες σχεδιάσεις περιγράφονται στο [KLE87], ενώ ένα πλήρες εγχειρίδιο είναι το [CD96]. Εξειδικευμένες σχεδιάσεις έχουν επίσης αναπτυχθεί

([BAR94]) για την προσαρμογή ειδικού τύπου μεταμοντέλων, όπως αυτά, που βασίζονται σε splines, αξονικές συναρτήσεις βάσης κ.α. Ιδιαίτερο ενδιαφέρον όμως παρουσιάζει και η βελτιωμένη πειραματική σχεδίαση ([CK99]), που έχει αναπτυχθεί για πειράματα προσομοίωσης δικτύων ουρών με αποτελέσματα, που εμφανίζουν υψηλή ετεροσκεδαστικότητα.

Τέλος, είναι σημαντικό να τονισθεί ότι η πρώτη προσπάθεια διατύπωσης μιας συστηματικής διαδικασίας προσαρμογής μεταμοντέλων, που εμφανίζεται στη βιβλιογραφία, είναι αυτή στην [KS00].

6.2 Χρήση μεταμοντέλων στην εκτίμηση της απόδοσης συστημάτων

Στην [KAL01] περιγράφεται μία μελέτη προσαρμογής μεταμοντέλων και ανάλυσης ευαισθησίας για το μοντέλο του σχήματος 6.1.



Σχήμα 6.1 Ένα μοντέλο απόδοσης μιας κεντρικής διεργασίας με χαρακτηριστικά ταυτόχρονης κατοχής πόρων και στοχαστικής δρομολόγησης

Τα μέτρα απόδοσης, που μελετήθηκαν, ήταν ο χρόνος απόκρισης και η παραγωγή, όπως αυτά εμφανίζονται στο σχήμα. Οι παράγοντες, που θεωρήθηκε ότι επηρεάζουν τις μεταβολές των μέτρων απόδοσης, που ενδιαφέρουν, είναι:

A: ο αριθμός των τερματικών, που εξυπηρετούνται από το σύστημα

B: ο αριθμός των μεριδίων μνήμης, που διαθέτει το σύστημα

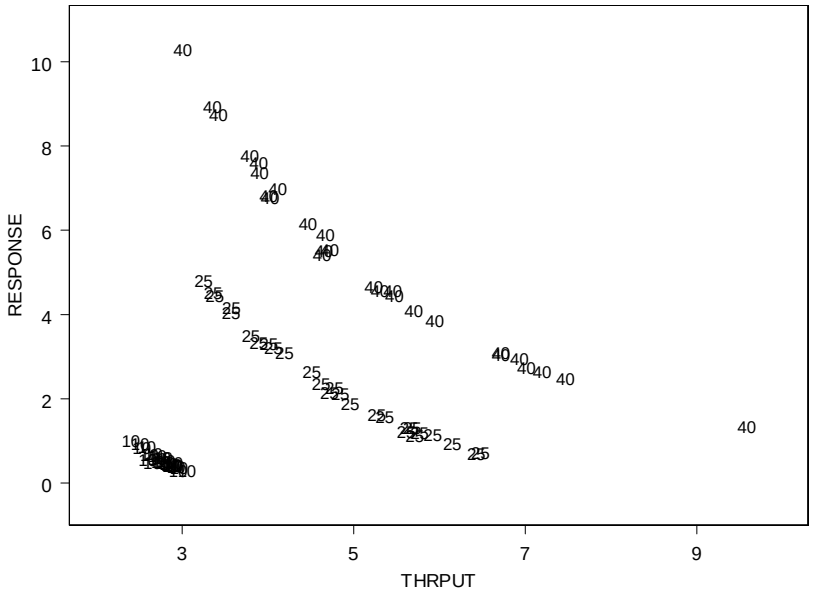
C: ο ρυθμός εξυπηρέτησης στη CPU

D: η ταχύτητα του κάθε ένα από τους δίσκους, που χρησιμοποιούνται

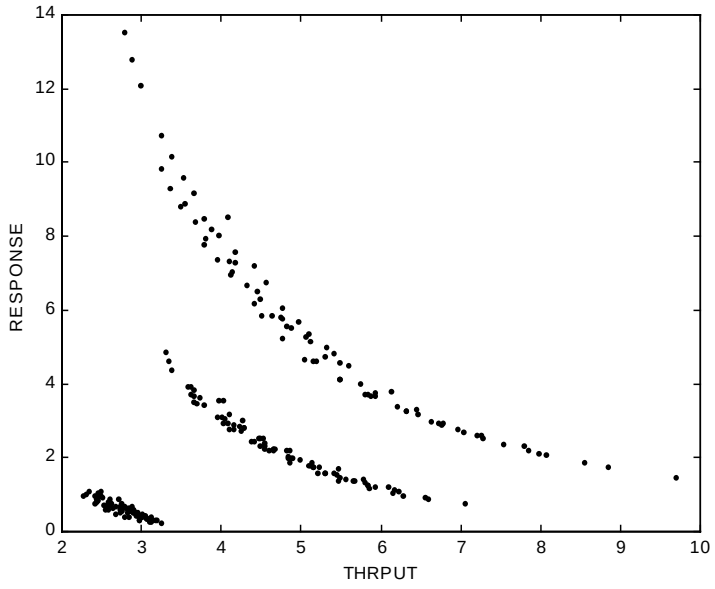
E: ο αριθμός των (ομοιόμορφα εκμεταλλευόμενων) δίσκων

Κάθε παράγοντας μεταβλήθηκε σε τρία διαφορετικά επίπεδα, σύμφωνα με μία 3^{5-1} πειραματική σχεδίαση από το [CD96] (81 περιπτώσεις από το σύνολο των 253 της πλήρους

παραγοντικής σχεδίασης). Τα πειράματα προσομοίωσης εκτελέστηκαν με ένα ιεραρχικό μοντέλο προσομοίωσης του HIT, όπως αυτά, που περιγράφηκαν στην κεφάλαιο 5. Μεταξύ των άλλων, έγινε προσαρμογή μεταμοντέλων και για τα δύο μέτρα απόδοσης. Στο σχήμα 6.2 εικονίζονται τα αποτελέσματα της εκτέλεσης των πειραματικών μονάδων και στο σχήμα 6.3 οι τιμές πρόγνωσης με τη χρήση των μεταμοντέλων, που προσαρμόσθηκαν, για όλους τους δυνατούς συνδυασμούς των επιπέδων των παραγόντων.

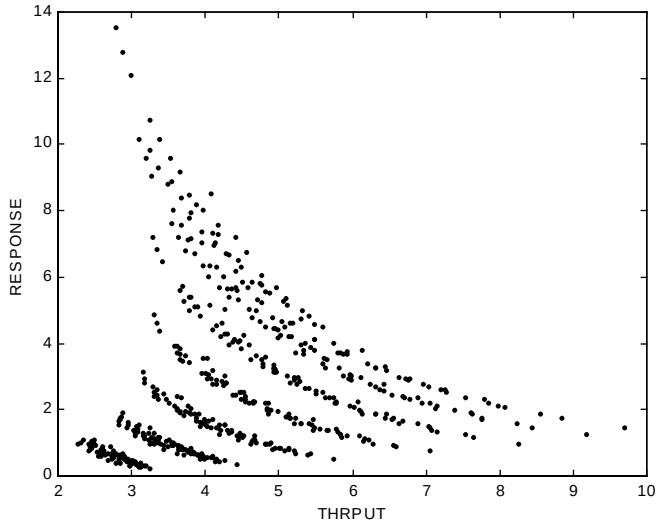


Σχήμα 6.2 Τα 81 αποτελέσματα της εκτέλεσης των πειραματικών μονάδων του πειράματος του σχήματος 6.1

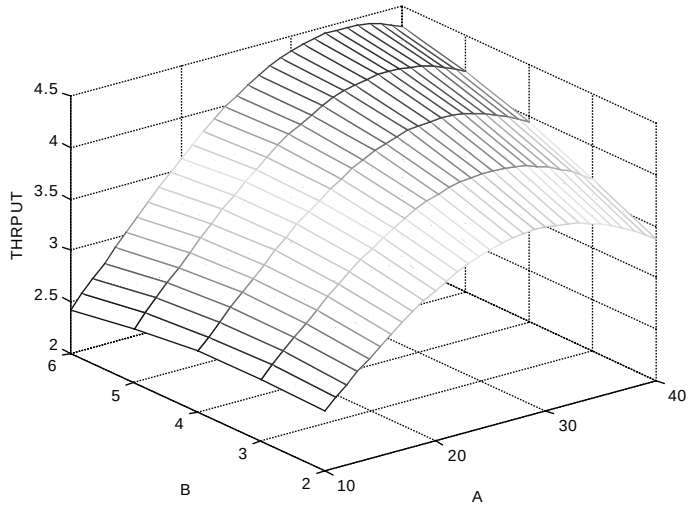


Σχήμα 6.3 Πρόγνωση για τις 253 περιπτώσεις της πλήρους παραγοντικής σχεδίασης του πειράματος του σχήματος 6.1

Στο σχήμα 6.4 εμφανίζονται σημεία πρόγνωσης με παρεμβολή για περιπτώσεις 10, 15, 20, 25, 30, 35 και 40 τερματικών, ενώ στο 6.5 εικονίζεται μία επιφάνεια απόκρισης για το μέτρο της παραγωγής με σταθερές τιμές στους παράγοντες C, D και E.



Σχήμα 6.4 Πρόγνωση με παρεμβολή για περιπτώσεις 10, 15, 20, 25, 30, 35 και 40 τερματικών για το μοντέλο του σχήματος 6.1



Σχήμα 6.5 Επιφάνεια απόκρισης για την παραγωγή με σταθερές τιμές στους παράγοντες C, D και E του μοντέλου του σχήματος 6.1

Η βελτιστοποίηση του λόγου RESPONSE/THRPUT μετά την προσαρμογή του κατάλληλου μεταμοντέλου και τη χρήση ενός αλγορίθμου simulated annealing από την [BJS86], έδωσε τα αναμενόμενα αποτελέσματα, γεγονός, που αποτελεί μία ένδειξη εγκυρότητας της προσέγγισης, που ακολουθήθηκε.

Η υποδειγματική αυτή ανάλυση οδήγησε στην ανάδειξη συγκεκριμένων προβλημάτων, τα οποία εμφανίζονται σε αυτού του είδους τις μελέτες μοντέλων δικτύων ουρών και αποτελούν ασφαλώς πεδία μεγάλου ερευνητικού ενδιαφέροντος. Τα προβλήματα αυτά αναφέρονται συνοπτικά στην παράγραφο 6.3.

Μία άλλη χρήση ([WVC01]) των μεταμοντέλων στην εκτίμηση της απόδοσης συστημάτων είναι η πρόγνωση των απαιτήσεων εξυπηρέτησης από τους πόρους ενός μοντέλου απόδοσης (*συναρτήσεις πόρων*).

Έτσι, αν επικεντρωθούμε στην ιεραρχική μοντελική προσέγγιση, που περιγράφηκε στο κεφάλαιο 5, τότε μία υπηρεσία Α, η οποία παρέχεται στα πλαίσια ενός τμήματος της ιεραρχίας του μοντέλου ανταγωνισμού, υποβάλλει μία σειρά απαιτήσεων εξυπηρέτησης (service demands), που μπορεί να περιλαμβάνουν:

- το κόστος χρόνου CPU για την εκτέλεση της Α στα όρια του τμήματος, από το οποίο αυτή παρέχεται,
- τη χρήση πόρων μνήμης για την εκτέλεση της Α,
- τον αριθμό λειτουργιών I/O και
- τον αριθμό κλήσης υπηρεσιών, που παρέχονται από άλλα τμήματα (λογισμικού ή υλικού) του μοντέλου.

Οι ανεξάρτητες μεταβλητές μιας συνάρτησης πόρου είναι αυτές, που καθορίζουν τελικά τις απαιτήσεις εξυπηρέτησης, και αυτές μπορεί για παράδειγμα να είναι:

- το μέγεθος του μηνύματος για περιπτώσεις κόστους επικοινωνίας ή άλλα μεγέθη δεδομένων, όπως αυτά των πινάκων, που ταξινομούνται, ή γίνεται αναζήτηση σε αυτούς,
- ο τύπος της υποκείμενης πλατφόρμας, που περιλαμβάνει υλικό, έκδοση λειτουργικού συστήματος, μεταφραστή κ.α.

Έτσι, στην ενότητα 5.3.2 είδαμε ότι το κόστος χρόνου CPU για την αποσυμπλοκή και προώθηση των κλήσεων μεθόδων στις αντίστοιχες υλοποιήσεις αντικειμένων εξαρτάται από τον αριθμό των υλοποιήσεων αντικειμένων, που εξυπηρετεί ο κόμβος, και τον αριθμό των μεθόδων αλλά και τη σειρά, στην οποία έχει δηλωθεί η συγκεκριμένη μέθοδος μέσα στη διασύνδεση IDL, που χρησιμοποιείται.

Γενικά ο εντοπισμός των παραγόντων, από τους οποίους εξαρτάται μία περιγραφή απαιτήσεων εξυπηρέτησης, είναι το κεντρικό πρόβλημα στην προσαρμογή της κατάλληλης συνάρτησης πόρου. Ένας παράγοντας, που δε λαμβάνεται υπόψη κατά την προσαρμογή μιας συνάρτησης πόρου, είναι πιθανό να αποτελεί μία πηγή λάθους, ειδικά αν στην τελική ανάπτυξη του προϊόντος χρησιμοποιηθεί με κάποια διαφορετική τιμή. Στην πράξη πάντως, καμία ανάλυση προσαρμογής συνάρτησης πόρου δεν μπορεί να είναι πλήρης.

Ένα παράδειγμα συνάρτησης πόρου είναι αυτό, που περιγράφεται στην [PVR95]. Στη συγκεκριμένη εργασία, οι συγγραφείς ανέπτυξαν έναν κώδικα κατάλληλα προσαρμοσμένο για τη διεξαγωγή μετρήσεων του χρόνου CPU, που απαιτείται για την αποστολή μηνυμάτων πάνω από κάποια συγκεκριμένη έκδοση του πρωτοκόλλου TCP/IP. Μετά από σημαντικό αριθμό πειραμάτων - σε απομονωμένο σταθμό εργασίας χωρίς επιπλέον φόρτο - και την απαραίτητη ανάλυση δεδομένων, κατέληξαν στην αναπαράσταση του κόστους σε χρόνο CPU μέσω ενός Multivariate Adaptive Regression Spline (MARS) [FRI91].

Τέλος στην [XH96], οι συγγραφείς μελέτησαν το χρόνο CPU και την καθυστέρηση ενδοδιασύνδεσης (interconnection network) για τρεις διαφορετικές λειτουργίες επικοινωνίας και χρήση δύο εναλλακτικών βιβλιοθηκών επικοινωνίας με ανταλλαγή μηνυμάτων (message passing libraries) σε ένα IBM SP2. Τα αποτελέσματα έδειξαν ότι εξαρτώνται από το μέγεθος των μηνυμάτων και των αριθμό των διασυνδεδεμένων κόμβων στο υπολογιστικό σύστημα. Οι συναρτήσεις πόρων, που προσαρμόστηκαν, πέρα από τις προαναφερθείσες ποσοτικές

μεταβλητές εξαρτώνται και από τον τύπο της βιβλιοθήκης επικοινωνίας, που χρησιμοποιείται ως ποιοτική προβλέπουσα μεταβλητή.

6.3 Ερευνητικές κατευθύνσεις

Η μελέτη προσαρμογής μεταμοντέλων και ανάλυσης ευαισθησίας, που περιγράφεται στην παράγραφο 6.2, έφερε στο προσκήνιο σειρά προβλημάτων, τα οποία αποτελούν πεδίο εστίασης επιπλέον ερευνητικών προσπαθειών. Τα σημαντικότερα από αυτά είναι:

- Η συνήθως ταυτόχρονη μελέτη περισσοτέρων του ενός μέτρων απόδοσης. Στις περισσότερες περιπτώσεις, όπως και σε αυτή της [KAL01], επιλέγεται η προσέγγιση της παράλληλης ανάλυσης του κάθε μέτρου απόδοσης ανεξάρτητα από τα υπόλοιπα. Καθώς όμως σε κάθε μοντέλο απόδοσης τα μέτρα που ενδιαφέρουν, παρουσιάζουν συσχέτιση, θα ήταν ορθότερη η εφαρμογή κάποιας διαδικασίας *ανάλυσης πολυμεταβλητών αποτελεσμάτων* (multivariate output analysis). Ο βασικός λόγος, που αυτό δε συμβαίνει, είναι το γεγονός ότι η ανάλυση πολυμεταβλητών δεδομένων, σύμφωνα με την [KS00], δεν έχει φθάσει ακόμη στα επίπεδα ωριμότητας της κλασσικής ανάλυσης δεδομένων. Πάντως, μία ενδελεχής αναφορά όλων των τελευταίων εξελίξεων στο συγκεκριμένο πεδίο μπορεί κανείς να βρει στην [KHU96].

- Η πληθώρα των ποιοτικών μεταβλητών, που συχνά επηρεάζουν τα μέτρα απόδοσης του μοντέλου, όπως για παράδειγμα οι πειθαρχίες εξυπηρέτησης, οι χρησιμοποιούμενες πολιτικές αποσυμπλοκής μηνυμάτων ή άλλοι αλγόριθμοι, τύποι βιβλιοθηκών, μεταφραστών κ.α. Ένας μεγάλος αριθμός τέτοιων μεταβλητών εγείρει ερωτηματικά για τις δυνατότητες επαρκούς προσαρμογής ενός κατάλληλου μεταμοντέλου και αναπόφευκτα οδηγεί στην αναζήτηση κατάλληλων εξειδικευμένων τεχνικών, που αντιμετωπίζουν το πρόβλημα.

- Η υψηλή ετεροσκεδαστικότητα, που τα περισσότερα μοντέλα απόδοσης εμφανίζουν, εξαιτίας της ραγδαίας αύξησης των μέσων τιμών και της διασποράς των μέτρων απόδοσης σε συνθήκες υψηλού φόρτου. Το φαινόμενο αυτό περιπλέκει τις διαδικασίες αναζήτησης ενός κατάλληλου μεταμοντέλου. Μία ενδιαφέρουσα προσέγγιση, είναι αυτή της [CK99], όπου γίνεται χρήση αναλυτικών αποτελεσμάτων για συνθήκες υψηλού φόρτου, για την υποβοήθηση της κατασκευής βέλτιστων πειραματικών σχεδιάσεων. Το μεταμοντέλο, που προσαρμόζεται με τη διαδικασία αυτή, αποτελείται από δύο μέρη: ένα πολυωνυμικό μοντέλο χαμηλής τάξης και μία συνάρτηση, που προοδευτικά αναπαριστά τη συμπεριφορά του μέτρου απόδοσης, που μελετάται, καθώς ο φόρτος τείνει να δημιουργήσει συνθήκες κορεσμού. Πάντως γενικά, η χρήση αναλυτικών αποτελεσμάτων και τεχνικών περιορισμού διασποράς (βλ. παράγραφο 4.5) παρουσιάζει ιδιαίτερο ενδιαφέρον όσον αφορά την κατασκευή βελτιωμένων πειραματικών σχεδιάσεων.

- Το γεγονός ότι ο μεγάλος αριθμός μεταβλητών εισόδου, που αναπόφευκτα εμφανίζονται σε λεπτομερή μοντέλα απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής, συχνά καθιστά μία μελέτη προσαρμογής μεταμοντέλων υπολογιστικά αδύνατη. Η απάντηση στο πρόβλημα κρύβεται σε προωθημένες τεχνικές «κοσκινίσματος» των σημαντικών παραγόντων, όπως αυτή της [BK96].

Κεφάλαιο 7 *Θέματα Σχεδίασης της Απόδοσης Λογισμικού (Software Performance Engineering)*

Η Σχεδίαση της Απόδοσης Λογισμικού (ΣΑΛ) είναι ο όρος, που εισήχθη στο [SMI90] για την περιγραφή μιας συστηματικής, ποσοτικής προσέγγισης για την ανάπτυξη λογισμικού, έτσι ώστε αυτό να ανταποκρίνεται σε συγκεκριμένους στόχους απόδοσης. Ο σκοπός του κεφαλαίου αυτού είναι να δώσει μία συνολική άποψη των τεχνικών και της διαδικασίας σχεδίασης της απόδοσης λογισμικού, καθώς επίσης και του τρόπου, με τον οποίο οι διαδικασίες μοντελοποίησης της παραγράφου 5.2 εντάσσονται στο συγκεκριμένο πλαίσιο.

7.1 Σχεδίαση απόδοσης αντικειμενοστρεφών συστημάτων

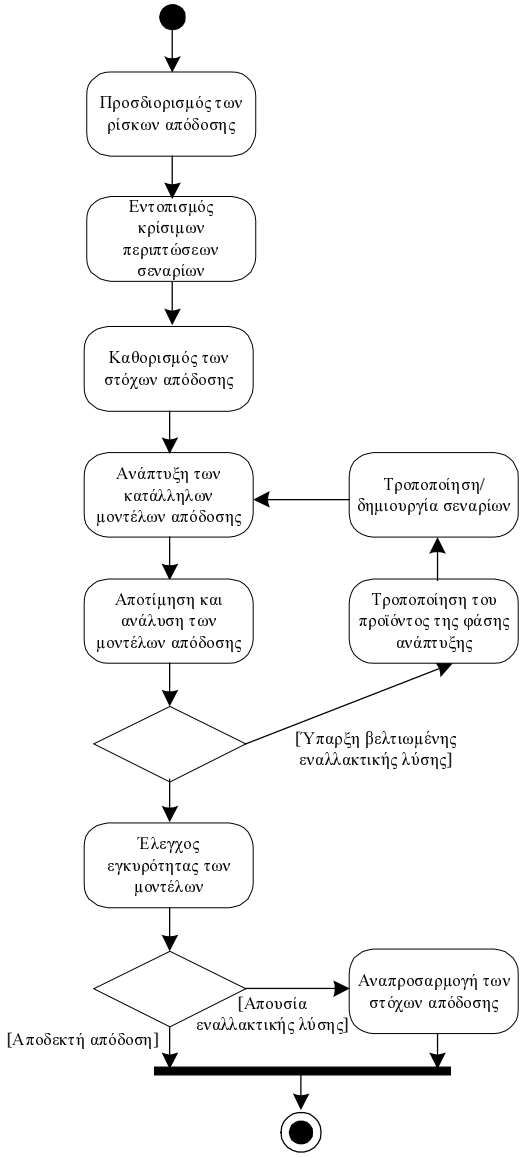
Η Σχεδίαση της Απόδοσης Λογισμικού είναι μία συστηματική προσέγγιση στην ανάπτυξη λογισμικού, που αποφεύγει τις ακραίες επιλογές της σχεδίασης με αποκλειστικό κριτήριο την αποδοτικότητα ή της προοπτικής της μετά την ανάπτυξη ρύθμισης της απόδοσης αυτού.

Η Σχεδίαση της Απόδοσης Λογισμικού κάνει χρήση της πρόγνωσης, βάσει μοντέλων, για την εξισορρόπηση των απαιτήσεων λειτουργικότητας, του μεγέθους του υλικού, της ποιότητας των παραγόμενων αποτελεσμάτων και των απαιτήσεων σε πόρους. Τα μοντέλα συμβάλλουν στον έλεγχο των παραπάνω, για την επιλογή της κατάλληλης αρχιτεκτονικής και σχεδιαστικής λύσης με τα επιθυμητά χαρακτηριστικά απόδοσης. Με την αντιμετώπιση των προβλημάτων απόδοσης, σε όλες τις φάσεις της διαδικασίας ανάπτυξης λογισμικού, περιορίζεται η πιθανότητα εμφάνισής τους στα τελευταία στάδια του κύκλου ζωής του. Διατυπώνονται επίσης αρχές και υψηλού επιπέδου υποδείγματα για τη δημιουργία αποδοτικού λογισμικού, διαδικασίες καθορισμού προδιαγραφών απόδοσης και κατευθυντήριες γραμμές για τον τύπο της ανάλυσης (βλ. ενότητα 5.2.3), που είναι ο πλέον κατάλληλος σε κάθε στάδιο της ανάπτυξης.

Οι τεχνικές Σχεδίασης της Απόδοσης Λογισμικού (ΣΑΛ) παρέχουν, σύμφωνα με την [SMI01], τις εξής πληροφορίες για το προς ανάπτυξη σύστημα:

- αποσαφήνιση των απαιτήσεων απόδοσης,
- πρόγνωση της απόδοσης με ακρίβεια, ανάλογη των λεπτομερειών σχεδίασης και της ποιότητας των εκτιμήσεων για τις απαιτήσεις πόρων, που είναι διαθέσιμες, σε κάθε στάδιο του κύκλου ζωής του λογισμικού,

- εκτιμήσεις της ευαισθησίας των προγνώσεων σε σχέση με την ακρίβεια των πληροφοριών χρήσης πόρων, αλλά και με τις μεταβολές των επιπέδων φόρτου του συστήματος,
- αποσαφήνιση της επίδρασης των εναλλακτικών σχεδιάσεων στην απόδοση του συστήματος,
- εκτίμηση των δυνατοτήτων κλιμάκωσης,
- εντοπισμός των κρίσιμων σημείων του συστήματος,
- εντοπισμός των υποθέσεων, που η μη ισχύς τους οδηγεί σε διαφοροποίηση της μελέτης της απόδοσης του συστήματος,
- ενδείξεις για τον προϋπολογισμό των πόρων, που θα κάνουν χρήση τα τμήματα του συστήματος, και τέλος
- ενδείξεις χρήσιμες στη σχεδίαση δοκιμών απόδοσης.



Σχήμα 7.1 Διάγραμμα δράσεων της διαδικασίας Σχεδίασης της Απόδοσης Λογισμικού

Το διάγραμμα δράσεων του σχήματος 7.1 απεικονίζει τη ροή εργασίας της διαδικασίας Σχεδίασης της Απόδοσης Λογισμικού (ΣΑΛ). Η συγκεκριμένη ροή εργασίας βασίζεται σε αυτή, που προτείνεται στο [SW01] και είναι κατάλληλα τροποποιημένη, έτσι ώστε να ενσωματώνει τις διαδικασίες μοντελοποίησης και ανάλυσης της παραγράφου 5.2. Τα βήματα της εικονιζόμενης διαδικασίας επαναλαμβάνονται σε κάθε φάση της ανάπτυξης του λογισμικού.

Στις αρχικές φάσεις, το αντικείμενο της διαδικασίας είναι η αρχιτεκτονική, οι προδιαγραφές του λογισμικού και κάποια υψηλού επιπέδου σχέδια για την ικανοποίηση των προδιαγραφών. Καθώς η ακρίβεια των προγνώσεων εξαρτάται από την ακρίβεια των εκτιμήσεων για τις απαιτήσεις σε πόρους και η λήψη αυτών είναι συνήθως δύσκολη στις πρώτες φάσεις ανάπτυξης, στις περισσότερες περιπτώσεις προτιμάται η εφαρμογή ανάλυσης ορίων (βλ. ενότητα 5.2.3). Η χρήση μιας διαδικασίας ανάλυσης του συγκεκριμένου τύπου αποδίδει εκτιμήσεις για την καλύτερη και χειρότερη περίπτωση απόδοσης. Αν η καλύτερη περίπτωση απόδοσης δεν είναι ικανοποιητική, τότε αναζητείται κάποια εναλλακτική περίπτωση σχεδίασης. Αν η χειρότερη περίπτωση εκφράζεται από ικανοποιητικά αποτελέσματα, τότε η διαδικασία ανάπτυξης προχωρά στην επόμενη φάση. Αν τίποτε από τα δύο δε συμβαίνει, τότε χρησιμοποιείται κάποια διαδικασία ανάλυσης - εντοπισμού σημείων συμφόρησης για τον προσδιορισμό των κρίσιμων τμημάτων του λογισμικού, των οποίων οι εκτιμήσεις σε απαιτήσεις πόρων έχουν και τη μεγαλύτερη επίδραση στα αποτελέσματα. Η εφαρμογή των διαδικασιών μοντελοποίησης σε περισσότερα του ενός επίπεδα αφαίρεσης, που περιγράφονται στην παράγραφο 5.2, συμβάλλει στην περαιτέρω εξειδίκευση των κρίσιμων τμημάτων του λογισμικού και στην ακριβέστερη παραμετροποίηση των υψηλού επιπέδου μοντέλων απόδοσης αυτού. Σε μεταγενέστερες φάσεις, το αντικείμενο της διαδικασίας ΣΑΛ περιλαμβάνει αναλυτικότερες επιλογές σχεδίασης, αλγορίθμους, εναλλακτικές περιπτώσεις υλοποίησης κ.λ.π. Σε όλες τις φάσεις ανάπτυξης οι μηχανικοί εφαρμόζουν κάποιες γενικές αρχές και κάποια υψηλού αφαιρετικού επιπέδου υποδείγματα για τη δημιουργία αποδοτικών αρχιτεκτονικών και σχεδιάσεων.

7.1.1 Προσδιορισμός των κινδύνων απόδοσης

Με τον όρο κίνδυνοι απόδοσης αναφερόμαστε σε οτιδήποτε δημιουργεί αμφιβολίες για την επιτυχία ενός έργου, όσον αφορά την απόδοση του προς ανάπτυξη προϊόντος. Αυτό μπορεί να είναι η υιοθέτηση μιας νέας τεχνολογίας, η δυνατοτήτα της επιλεγείσας αρχιτεκτονικής να ενσωματώνει αλλαγές, καθώς αυτή εξελίσσεται κ.α. Η διαδικασία Σχεδίασης της Απόδοσης Λογισμικού προϋποθέτει τον προσδιορισμό του κινδύνου της απόδοσης. Αν ο κίνδυνος αυτός είναι μικρός, τότε το σύνολο των πόρων του έργου, που επενδύονται στη διαδικασία ΣΑΛ, πρέπει να είναι μικρό. Σε κάθε περίπτωση το σύνολο των πόρων του έργου, που διατίθενται στη συγκεκριμένη διαδικασία, είναι ανάλογο της σημαντικότητας του κινδύνου της απόδοσης.

Όπως κάθε κίνδυνος, έτσι και αυτός της απόδοσης συνίσταται στην πιθανότητα εκδήλωσής του και στη σοβαρότητα της ζημιάς, που θα προκαλούσε. Στην περίπτωση της ύπαρξης περισσοτέρων του ενός κινδύνων απόδοσης, η κατάταξη αυτών σε σειρά σημαντικότητας βοηθά στη συστηματική αντιμετώπισή τους.

7.1.2 Εντοπισμός κρίσιμων περιπτώσεων σεναρίων

Οι περιπτώσεις σεναρίων εκφράζουν την από την άποψη του χρήστη λειτουργικότητα του προς ανάπτυξη συστήματος. Αποτελούν μέρος των προδιαγραφών αυτού και στη UML περιγράφονται από τα διαγράμματα περιπτώσεων σεναρίων (βλ. ενότητα 2.4.1). Οι κρίσιμες περιπτώσεις σεναρίων είναι εκείνες, που είναι σημαντικές, είτε για την αποδοτικότητα του συστήματος, όπως αυτή γίνεται αντιληπτή από τους χρήστες του, είτε για τους

διαπιστωμένους κινδύνους απόδοσης. Συνήθως γίνεται εφαρμογή του εμπειρικού κανόνα 80 - 20, που στηρίζεται στην αρχή ότι ένα μικρό υποσύνολο περιπτώσεων σεναρίων ($\leq$20%) εκφράζει το μεγαλύτερο μέρος ($\geq$80%) της χρήσης του συστήματος. Καθώς λοιπόν η απόδοση του συστήματος καθορίζεται πρωτίστως από το σύνολο αυτό των συχνά χρησιμοποιούμενων λειτουργιών, αυτές είναι και εκείνες στις οποίες πρέπει βασικά να εστιασθεί η διαδικασία ΣΑΛ. Για την αναπαράσταση της κάθε περίπτωσης σεναρίου απόδοσης, προτείνεται στο [SW01] η χρήση των διαγραμμάτων αλληλουχιών μηνυμάτων της UML, εμπλουτισμένων με χαρακτηριστικά από τις προδιαγραφές Message Sequence Charts του [ITU96].

7.1.3 Καθορισμός των στόχων απόδοσης

Σε κάθε περίπτωση σεναρίου αντιστοιχεί ένας τουλάχιστο στόχος απόδοσης, ο οποίος καθορίζει τα ποσοτικά κριτήρια αποτίμησης των χαρακτηριστικών απόδοσης αυτής. Οι στόχοι απόδοσης τίθενται από τα πρώτα στάδια της ανάπτυξης του συστήματος. Μία αναλυτικότερη περιγραφή της διαδικασίας δίνεται στον πίνακα 5.1.

7.1.4 Ανάπτυξη, αποτίμηση και ανάλυση μοντέλων απόδοσης

Η δράση αυτή περιγράφεται από τις διαδικασίες της παραγράφου 5.2 για τη μοντελοποίηση σε περισσότερα του ενός επίπεδα αφαίρεσης. Ανάλογα με τη φάση ανάπτυξης, κατά την οποία διεξάγεται η συγκεκριμένη επανάληψη της διαδικασίας ΣΑΛ, εφαρμόζεται ο κατάλληλος τύπος ανάλυσης (βλ. ενότητα 5.2.3) με βάση το μοντέλο του επιπέδου, του οποίου το κόστος αποτίμησης είναι ανάλογο της σημαντικότητας του κινδύνου απόδοσης.

Αν η αποτίμηση και η ανάλυση του κατάλληλου μοντέλου απόδοσης δείξει την ύπαρξη προβλημάτων σε σχέση με τους στόχους απόδοσης, που έχουν τεθεί, τότε ακολουθούν:

- Η διαδικασία βελτίωσης του προϊόντος της φάσης ανάπτυξης: Αυτό μπορεί να περιλαμβάνει αλλαγές στην αρχιτεκτονική, στη σχεδίαση της επεξεργασίας των δεδομένων ή στη διαμόρφωση του συστήματος, ανάλογα με το τι έχει καθορισθεί ως προϊόν της φάσης ανάπτυξης, που διανύεται. Όπως ήδη έχει αναφερθεί, στη διαδικασία αυτή χρησιμοποιούνται κάποιες γενικές αρχές απόδοσης και κάποια υψηλού επιπέδου υποδείγματα, τα σημαντικότερα εκ των οποίων αναφέρονται στην παράγραφο 7.2. Κάθε φορά, που προκύπτει μία νέα ελκυστικότερη λύση, γίνεται τροποποίηση του αντίστοιχου σεναρίου ή δημιουργούνται νέα, που αντιπροσωπεύουν τα αναθεωρημένα σχέδια του λογισμικού. Η διαδικασία αυτή επαναλαμβάνεται μέχρις ότου καταλήξουμε σε απόδοση συμβατή με τους στόχους, που έχουν τεθεί, ή μέχρις ότου διαπιστωθεί ότι δεν υπάρχει κάποια πιο βελτιωμένη εναλλακτική λύση.
- Η διαδικασία εκτίμησης της εγκυρότητας του ή των μοντέλων απόδοσης, που χρησιμοποιήθηκαν, όπως ακριβώς περιγράφεται στην ενότητα 5.2.2.
- Εφόσον δεν έχουν ικανοποιηθεί οι στόχοι απόδοσης, που τέθηκαν, ακολουθεί η διαδικασία αναπροσαρμογής αυτών. Αν και ασφαλώς ένα τέτοιο αποτέλεσμα δεν είναι το ό,τι καλύτερο, είναι προτιμότερο να έχουμε έγκαιρη γνώση και να συνεχίζουμε την ανάπτυξη με αυτό ως δεδομένο. Σε κάποιες περιπτώσεις είναι πιθανή ακόμη και η επιλογή της ακύρωσης του έργου, κάτι, που σίγουρα θα είχε μεγαλύτερο κόστος αν αποφασιζόταν μετά την ολοκλήρωση του κύκλου ανάπτυξης και τη διαπίστωση ότι το σύστημα, που αναπτύχθηκε, δεν είναι λειτουργικό λόγω προβληματικής απόδοσης.

7.1.5 Παραδοτέα της διαδικασίας ΣΑΛ

Τα σημαντικότερα από τα παραδοτέα της διαδικασίας ΣΑΛ, που περιγράφονται αναλυτικά στο [SW01], είναι:

- Τα *πλάνα διαχείρισης της απόδοσης*: Αναφέρονται στις εργασίες διαχείρισης της διαδικασίας ΣΑΛ, δηλαδή τις αρμοδιότητες, τα χρονοδιαγράμματα, τον προϋπολογισμό και την κατανομή προσωπικού.
- Το *σχέδιο επαλήθευσης και εκτίμησης εγκυρότητας* (performance V&V plan): Περιγράφει τις μετρήσεις, που θα διεξαχθούν για τον έλεγχο της συμμόρφωσης του συστήματος στις προδιαγραφές απόδοσης, που έχουν τεθεί.
- Το *σχέδιο διαχείρισης της διαμόρφωσης της διαδικασίας ΣΑΛ*: Περιγράφει τη διαχείριση των αλλαγών στα υπόλοιπα παραδοτέα της διαδικασίας.
- Οι *οδηγοί απόδοσης*: Σε αυτούς περιγράφονται τα στοιχεία - κλειδιά, που καθορίζουν την απόδοση του συστήματος. Τέτοια στοιχεία μπορεί να είναι για παράδειγμα ο αριθμός των πελατών μιας τράπεζας, ο αριθμός των λογαριασμών ανά πελάτη κ.λ.π.
- Τα *σενάρια απόδοσης*: Περιγράφουν τις κρίσιμες περιπτώσεις σεναρίων λειτουργίας.
- Οι *στόχοι απόδοσης*: Απαριθμούνται μετρήσιμα κριτήρια απόδοσης στα οποία πρέπει να συμμορφώνεται το σύστημα.
- Η *περιγραφή του περιβάλλοντος εκτέλεσης*: Καθορίζεται το υλικό, η διαμόρφωση του δικτύου πάνω στο οποίο θα εκτελείται το λογισμικό, καθώς και άλλα προϊόντα λογισμικού από τα οποία θα εξαρτάται, όπως λειτουργικά συστήματα, βάσεις δεοδμένων κ.λ.π.
- Τα *μοντέλα απόδοσης* και τα αποτελέσματα της αποτίμησης και των διαφορετικών τύπων ανάλυσης αυτών.
- Οι *αναφορές επαλήθευσης και εκτίμησης εγκυρότητας*: Καταγράφουν τα λάθη, που εμφανίζονται στα άλλα παραδοτέα της διαδικασίας ΣΑΛ.
- Τα *πλάνα δοκιμών της απόδοσης* και τα αποτελέσματα αυτών.

7.1.6 Ενσωμάτωση της διαδικασίας ΣΑΛ στην ανάπτυξη λογισμικού

Η ενσωμάτωση της διαδικασίας ΣΑΛ στη διαδικασία ανάπτυξης λογισμικού, που χρησιμοποιείται, δεν είναι συνήθως κάτι δύσκολο. Περιλαμβάνει βασικά τον καθορισμό των ορόσημων και των παραδοτέων, που σχετίζονται με τη σχεδίαση της απόδοσης του λογισμικού. Σχετικά παραδείγματα, που αναφέρονται στις περιπτώσεις του μοντέλου ανάπτυξης του καταρράκτη, του μοντέλου της σπείρας και της ενιαίας διαδικασίας αντικειμενοστρεφούς ανάπτυξης δίνονται τόσο στο [LD98], όσο και στο [SW01].

Η ενιαία διαδικασία ανάπτυξης [JBR99] ολοκληρώνει σε μία προσέγγιση τις μέχρι τότε ανεξάρτητες μεθόδους των Jacobson, Booch και Rumbaugh, καθώς επίσης και την εμπειρία μεγάλου αριθμού κατασκευαστών του χώρου της πληροφορικής και των τηλεπικοινωνιών. Επειδή ακριβώς η συγκεκριμένη διαδικασία υποστηρίζει τη χρήση αντικειμενοστρεφών τεχνικών ανάπτυξης, καθώς και τη χρήση της UML, παρέχει ένα ιδανικό πλαίσιο ενσωμάτωσης της διαδικασίας ΣΑΛ, η οποία περιγράφηκε συνοπτικά στις προηγούμενες ενότητες.

Η ενιαία διαδικασία ανάπτυξης είναι μία επαναληπτική διαδικασία σταδιακής ανάπτυξης· κάθε επανάληψη αυτής είναι ένα μικρό έργο, που περιλαμβάνει συγκεκριμένες δραστηριότητες για τη σταδιακή ενσωμάτωση νέων δυνατοτήτων στο προϊόν. Απαρτίζεται από τέσσερις φάσεις, κάθε μία από τις οποίες εστιάζεται και σε μία διαφορετική άποψη της διαδικασίας ανάπτυξης. Οι φάσεις αυτές είναι:

- Η *φάση της έναρξης*: Επικεντρώνεται στον ορισμό του τελικού προϊόντος και του πεδίου του έργου.
- Η *φάση της λεπτομερούς περιγραφής*: Επικεντρώνεται στη διαχείριση του έργου και στην περιγραφή των χαρακτηριστικών και της αρχιτεκτονικής του προϊόντος.
- Η *φάση της κατασκευής*.
- Η *φάση της μετάβασης*: Αφορά τη μεταφορά του λογισμικού, των κειμένων υποστήριξης και των κατάλληλων υπηρεσιών στους χρήστες.

Το τέλος της κάθε φάσης αποτελεί ένα ορόσημο, δηλαδή, ένα σημείο λήψης παραδοτέων της διαδικασίας ΣΑΛ μαζί με τα υπόλοιπα παραδοτέα της διαδικασίας ανάπτυξης. Επιπλέον, κάθε φάση συγκροτείται από μία σειρά επαναλήψεων πέντε βασικών ροών εργασίας: i) *της διατύπωσης των απαιτήσεων του συστήματος με τη χρήση περιπτώσεων σεναρίων*, ii) *της ανάλυσης*, iii) *της σχεδίασης*, iv) *της υλοποίησης* και v) *της δοκιμής*. Το τέλος κάθε τέτοιας επανάληψης αποτελεί ένα εσωτερικό ορόσημο της κάθε φάσης και θα μπορούσε επίσης να αποτελεί και ένα σημείο λήψης παραδοτέων της διαδικασίας ΣΑΛ.

Η ενιαία διαδικασία ανάπτυξης διαθέτει τρία χαρακτηριστικά, που καθιστούν εύκολη την ενσωμάτωση της διαδικασίας ΣΑΛ:

- Βασίζεται στον πρώιμο καθορισμό της αρχιτεκτονικής του λογισμικού. Έτσι, δίνεται η δυνατότητα διασφάλισης, μέσω της διαδικασίας ΣΑΛ, του ότι η επιλεγείσα αρχιτεκτονική ανταποκρίνεται στους στόχους απόδοσης, που έχουν τεθεί.
- Βασίζεται στην περιγραφή των απαιτήσεων του συστήματος με τη χρήση περιπτώσεων σεναρίων, κάποια από τα οποία χαρακτηρίζονται ως κρίσιμα για τη μελέτη τους με την εφαρμογή της διαδικασίας ΣΑΛ.
- Η εξέλιξή της καθοδηγείται από τον προσδιορισμό των κινδύνων, όπως ακριβώς και η διαδικασία ΣΑΛ, που όμως αφορά μόνο τους κινδύνους απόδοσης.

7.2 Αρχές και υποδείγματα απόδοσης

Στην παράγραφο αυτή γίνεται μία σύντομη περιγραφή των αρχών και των υποδειγμάτων απόδοσης, που προτείνονται στο [LD98] και στο [SW01] και καθοδηγούν, όπως ήδη τονίστηκε, τη δράση τροποποίησης του προϊόντος της φάσης της διαδικασίας SPE.

7.2.1 Αρχή της ελαχιστοποίησης του υπολογιστικού φόρτου

Για την ελαχιστοποίηση του συνολικού υπολογιστικού φόρτου, που προκαλείται από την επεξεργασία ενός αριθμού μονάδων λογισμικού, πρέπει να ελαχιστοποιηθεί το άθροισμα των γινομένων της συχνότητας εκτέλεσης επί του κόστους ανά εκτέλεση για όλες τις μονάδες λογισμικού. Η αρχή αυτή συμπληρώνεται από την αρχή της επικέντρωσης, που προτείνεται στο [SW01] και υπαγορεύει την ελαχιστοποίηση αρχικά του κόστους του 20% των λειτουργιών, που αντιπροσωπεύουν το 80% της χρήσης του συστήματος.

Για το σκοπό αυτό επιδιώκουμε τη βελτιστοποίηση της απόδοσης των μονάδων λογισμικού μέσω:

- της απαλοιφής του περιττού κώδικα,
- της χρήσης τεχνικών βελτιστοποίησης του κώδικα των μονάδων λογισμικού,
- της συστηματικής απομάκρυνσης των περιττών χαρακτηριστικών της εφαρμογής,
- της βελτίωσης του τρόπου με τον οποίο η εφαρμογή χρησιμοποιεί τις κλήσεις συστήματος, που μπορεί να αποδώσει σημαντικά ωφέλη,
- της χρήσης λογισμικού άλλων κατασκευαστών μόνο όταν η τεκμηρίωση αυτού αναφέρεται σε συγκεκριμένα στοιχεία απόδοσης,
- της επικέντρωσης στις στοιχειώδεις λειτουργίες (διεργασιακής επικοινωνίας, δημιουργίας διεργασιών κ.α.) υψηλού κόστους,
- του περιορισμού του αριθμού των λειτουργιών I/O ενδεχομένως με τη χρήση τεχνικών συμπίεσης δεδομένων,
- του περιορισμού της επικοινωνίας με απομακρυσμένους πόρους λογισμικού στις απολύτως απαραίτητες περιπτώσεις,
- της εστίασης στη βελτίωση της σχεδίασης των μονάδων διακομιστών της εφαρμογής,
- της μη χρήσης ιεραρχιών κληρονομικότητας πολλών επιπέδων, καθώς η πρακτική αυτή προσθέτει επιπλέον κόστος επεξεργασίας,
- του περιορισμού του αριθμού των μετατροπών τύπων δεδομένων,
- της ελαχιστοποίησης των δεδομένων, που επιστρέφονται στην εφαρμογή, μέσω της όσο το δυνατό μεγαλύτερης χρήσης των ενσωματωμένων ευκολιών των συστημάτων διαχείρισης δεδομένων,
- της βελτίωσης της τοπικότητας των αναφορών στη μνήμη και στην περιοχή παρακράτησης (cache),
- της εστίασης σε μία ορθή ομαδοποίηση των μονάδων λογισμικού, καθώς αυτή επηρεάζει και τη φυσική ομαδοποίηση αυτών σε συγκεκριμένους κόμβους επεξεργασίας,
- της ελαχιστοποίησης της κίνησης στο δίκτυο με την εφαρμογή κάποιας από τις αρχιτεκτονικές, που περιγράφονται στην παράγραφο 2.1.

7.2.2 Αρχή της βελτίωσης της αποδοτικότητας

Αποδοτική σχεδίαση είναι αυτή, που μεγιστοποιεί το λόγο της χρήσιμης επεξεργασίας ως προς τις διάφορες καθυστερήσεις, που προκαλούνται κατά τη διάρκεια αυτής. Στο [SW01] η αρχή αυτή αναφέρεται ως αρχή της έγκαιρης αποκατάστασης της σχέσης μεταξύ δύο ή περισσότερων στοιχείων επεξεργασίας (fixing - point principle). Παραδείγματα έγκαιρης αποκατάστασης της σχέσης μεταξύ των στοιχείων επεξεργασίας είναι η χρήση ευρετηρίων, τεχνικών Hash και δεικτών, αντί της χρήσης αλγορίθμων αναζήτησης, η χρήση στατικών διασυνδέσεων αντί αυτής δυναμικών διασυνδέσεων διεργασιακής επικοινωνίας, η χρήση στατικής (μεταφρασμένης) κληρονομικότητας αντί αυτής της δυναμικής αναζήτησης της υπερκλάσης και των ιδιοτήτων της, η χρήση μεταφράσιμης γλώσσας προγραμματισμού αντί μιας διερμηνεύσιμης κ.α.

7.2.3 Αρχή της εγγύτητας

Η αρχή της εγγύτητας αναφέρεται στην ομαδοποίηση των μονάδων λογισμικού με βάση τη χρήση αυτών. Διακρίνεται:

- Στην *τοπική εγγύτητα*: Αναφέρεται στην χωρική τους απόσταση.
- Στη *χρονική εγγύτητα*: Αναφέρεται στη χρονική απόσταση.
- Στην *τελεσφόρο εγγύτητα*: Αναφέρεται στη συνάφεια σκοπού.
- Στην *εγγύτητα βαθμού*: Έχει να κάνει με τη δυναμικότητα και τη χωρητικότητα μιας σχεδίασης.

Η τοπική εγγύτητα υπαγορεύει, οι μονάδες λογισμικού, που παρουσιάζουν μεγάλη συχνότητα αλληλεπίδρασης, να μην είναι φυσικώς απομακρυσμένες. Γενικά, η ομαδοποίηση των μονάδων λογισμικού είναι μία απόφαση - κλειδί, που καθορίζει τη σωστή ισορροπία και το βαθμό της μεταξύ τους συνεκτικότητας και ανεξαρτησίας. Σε ένα μεγάλο κατανεμημένο σύστημα δεν είναι συνήθως εύκολη η ανά πάσα στιγμή πρόσβαση στην καθολική κατάσταση αυτού. Επίσης, είναι δύσκολη και η ενημέρωση κατανεμημένων ή επαναλαμβανόμενων δομών δεδομένων. Οι σχεδιάσεις, που σπάνια απαιτούν πρόσβαση σε πληροφορία για την καθολική κατάσταση του συστήματος, έχουν πάντα μεγαλύτερα περιθώρια δυνατοτήτων κλιμάκωσης.

Κάποιες γενικές κατευθυντήριες γραμμές, που οδηγούν σε μία σχεδίαση με ικανοποιητικό βαθμό τοπικής εγγύτητας, είναι:

- η ελαχιστοποίηση των μεταφορών δεδομένων μεταξύ των πελατών και των διακομιστών,
- η εκτέλεση όλης της λογικής της διεπαφής χρήστη στο σταθμό - πελάτη,
- η συγκέντρωση όλων των σχετιζόμενων δεδομένων σε μία μόνο θέση,
- η επιλεκτική επανάληψη των ευρέως χρησιμοποιούμενων δεδομένων σε περισσότερους του ενός κόμβους, κ.α.

Εκτός από την ομαδοποίηση μονάδων λογισμικού με ικανοποιητικό βαθμό τοπικής εγγύτητας, είναι, σε κάποιες περιπτώσεις, επιθυμητή και η αποδοτική αλληλεπίδραση ενός διαφορετικού συνόλου μονάδων λογισμικού. Στην περίπτωση αυτή εφαρμόζονται συνήθως τεχνικές, που διασφαλίζουν τη χρονική αλληλουχία στην εκτέλεση των λειτουργιών, οι οποίες απαιτούν αποδοτική αλληλεπίδραση μεταξύ των μονάδων του συγκεκριμένου συνόλου. Τότε μιλάμε για χρονική εγγύτητα. Παράδειγμα τέτοιας τεχνικής αποτελεί η προσωρινή συσσώρευση σε τοπικό πίνακα ή αρχείο μεγάλου αριθμού διεκπεραιώσεων και η κατά ομάδες επεξεργασία αυτών σε καθορισμένα χρονικά διαστήματα.

Η τελεσφόρος εγγύτητα αποβλέπει στη σχεδίαση των λειτουργιών του λογισμικού και την κατανομή του υπολογιστικού φόρτου στους διαθέσιμους πόρους επεξεργασίας, έτσι ώστε να ταιριάζουν:

- οι τρόποι συμπεριφοράς και οι ανάγκες επεξεργασίας πληροφοριών των χρηστών,
- οι λειτουργίες και οι απαιτήσεις επεξεργασίας του λογισμικού και
- η διαθεσιμότητα, η αποκρισιμότητα και οι δυνατότητες των πόρων υλικού.

Παράδειγμα περίπτωσης τελεσφόρου εγγύτητας είναι η εκμετάλλευση κατάλληλου τύπου υλικού (π.χ. παράλληλης επεξεργασίας) για την εκτέλεση ειδικών λειτουργιών. Συχνά επίσης, όταν για μία εφαρμογή τίθενται κρίσιμοι στόχοι απόδοσης σε συνθήκες υψηλού φόρτου,

επιλέγεται η χρήση ενός κατάλληλα διαμορφωμένου διακομιστή αποκλειστικής εξυπηρέτησης της εφαρμογής.

Τέλος, η εγγύτητα βαθμού αναφέρεται στο ταίριασμα των απαιτήσεων αποθήκευσης, μεταφοράς και επεξεργασίας δεδομένων, που προκύπτουν από τον υπαρκτό φόρτο επιχειρηματικών λειτουργιών, με τη χωρητικότητα και τις δυνατότητες των πόρων υλικού και λογισμικού, που συγκροτούν την εφαρμογή.

7.2.4 Αρχή της διανομής πόρων

Η αρχή της διανομής πόρων υπαγορεύει το διαμοίρασμα πόρων λογισμικού και υλικού με αποδοτικό τρόπο. Αυτό απλά σημαίνει ότι, όταν είναι απαραίτητη η αποκλειστική χρήση κάποιου πόρου, είναι σημαντικό να υπάρχει μέριμνα για την ελαχιστοποίηση του χρόνου αναμονής για χρήση και του χρόνου κατοχής του πόρου.

Κάποιες απλές κατευθυντήριες γραμμές για την ελαχιστοποίηση του χρόνου κατοχής ενός πόρου είναι:

- ο περιορισμός του αριθμού και του μεγέθους των κλήσεων εξυπηρέτησης,
- η επιτάχυνση της επεξεργασίας των κλήσεων εξυπηρέτησης και
- η όσο είναι δυνατό κατεύθυνση κάποιων από αυτές σε άλλους πόρους.

7.2.5 Αρχή της παράλληλης επεξεργασίας

Όταν χρησιμοποιείται παράλληλη επεξεργασία, αυτή μπορεί να είναι:

- πραγματικά παράλληλη, όπου οι διεργασίες ή τα νήματα ροής εκτελούνται ταυτόχρονα σε ένα σύνολο επεξεργαστών, ή
- φαινομενικά παράλληλη, όπου η εκτέλεση ενός αριθμού διεργασιών ή νημάτων ροής πολυπλέκεται στον ίδιο επεξεργαστή.

Και οι δύο περιπτώσεις παράλληλης επεξεργασίας συνοδεύονται από ένα κόστος επικοινωνίας και συντονισμού των διεργασιών ή των νημάτων ροής, που εκτελούνται. Επιπλέον, αν και στη δεύτερη περίπτωση είναι πιθανή η επικάλυψη της επεξεργασίας κάποιας διεργασίας ή νήματος ροής με παράλληλη διεξαγωγή λειτουργιών I/O από κάποια άλλη διεργασία ή νήμα ροής, συχνά προκαλείται ανταγωνισμός για το διαμοίρασμα του ιδίου πόρου.

Η αρχή της παράλληλης επεξεργασίας υπαγορεύει τη χρήση αυτής μόνο όταν η επιτάχυνση της επεξεργασίας, που έχει ως αποτέλεσμα, υπερτερεί του κόστους επικοινωνίας και ανταγωνισμού για χρήση των ιδίων πόρων, που προκαλείται.

7.2.6 Αρχή του συμβιβασμού των στόχων

Η ικανοποιητική απόδοση είναι μόνο μία περίπτωση ενός συνόλου αντικρουόμενων - στις περισσότερες περιπτώσεις - στόχων, στους οποίους επικεντρώνεται κάθε διαδικασία ανάπτυξης λογισμικού. Ενδιαφέρουν ακόμη, η σωστή λειτουργικότητα, η υψηλή αξιοπιστία, η ακεραιότητα δεδομένων, η μεταφερσιμότητα, η συντηρισιμότητα και επεκτασιμότητα κ.α.

Ακόμη και αυτός καθαυτός ο στόχος επίτευξης ικανοποιητικής απόδοσης είναι σαφές ότι έχει διαφορετική έννοια για διαφορετικούς ανθρώπους, ανάλογα με το ρόλο του καθενός ως προς τη χρήση του προς ανάπτυξη λογισμικού. Από την άποψη των χρηστών λοιπόν, η απόδοση εκφράζεται βασικά από τους χρόνους απόκρισης της εφαρμογής, ενώ όσον αφορά το διαχειριστή, αυτός ενδιαφέρεται κυρίως για τη συνολική παραγωγή και τη δυναμικότητά της ως προς την υποστήριξη του εκάστωτε συνόλου των χρηστών. Επειδή όμως οι διαθέσιμοι

πόροι υλικού και ο προϋπολογισμός χρήσης αυτών θέτουν συνήθως όρια στη δυναμικότητα του συστήματος, συχνά χρειάζεται μία υπεύθυνη διαχείριση της ισορροπίας των αναγκών για συγκεκριμένα πλαίσια απόδοσης της εφαρμογής με τις ανάγκες των χρηστών των υπολοίπων εφαρμογών του συστήματος.

Η αρχή του συμβιβασμού των στόχων, λοιπόν, υπαγορεύει την τήρηση των κατάλληλων ισορροπιών μεταξύ των αντικρουόμενων στόχων, που εμφανίζονται κατά την ανάπτυξη του λογισμικού. Σε ένα υψηλό επίπεδο καθορισμού των στόχων, σημαντικό ρόλο θα μπορούσε να παίξει κάποια πολυκριτηριακή μεθοδολογία λήψης αποφάσεων, όπως αυτή, που χρησιμοποιείται στις [SRK99] και [SRK00]. Σε ένα άλλο επίπεδο όμως, συχνά τα προβλήματα, που προκύπτουν κατά την ανάπτυξη λογισμικού, αντιμετωπίζονται με τη χρήση σχεδιαστικών ή αρχιτεκτονικών υποδειγμάτων, που αποτελούν ουσιαστικά δοκιμασμένες λύσεις ισορροπημένης σχεδίασης.

7.2.7 Υποδείγματα απόδοσης

Τα υποδείγματα απόδοσης, που εισάγονται στο [SW01], περιγράφουν λύσεις σε ένα πιο αφηρημένο επίπεδο από αυτό των σχεδιαστικών υποδειγμάτων. Κάθε υπόδειγμα απόδοσης ενσωματώνει μία ή περισσότερες αρχές απόδοσης. Ένα σχεδιαστικό υπόδειγμα μπορεί να συνιστά μία υλοποίηση ενός υποδείγματος απόδοσης. Έτσι για παράδειγμα το υπόδειγμα απόδοσης «Γρήγορος Δρόμος» βασίζεται στην αρχή της επικέντρωσης ή αλλιώς της ελαχιστοποίησης του υπολογιστικού φόρτου και για την υλοποίησή του μπορεί να χρησιμοποιηθεί το σχεδιαστικό υπόδειγμα της «αντιπροσώπευσης» (Proxy pattern), όπως αυτό ορίζεται στο [GHJ95].

Καθώς τα υποδείγματα απόδοσης είναι ένα αφαιρετικό επίπεδο πιο ψηλά από τα σχεδιαστικά υποδείγματα, δεν είναι δυνατή η περιγραφή τους με διαγράμματα κλάσεων. Αντίθετα, οι σχετικές κλάσεις και οι μεταξύ τους αλληλεπιδράσεις μπορούν να καθορισθούν μόνο μετά από την επιλογή του τρόπου υλοποίησης ενός υποδείγματος απόδοσης.

Τα επτά υποδείγματα απόδοσης, που εισάγονται στο [SW01], περιγράφονται με τυποποιημένο τρόπο από το όνομα, το πρόβλημα, που αντιμετωπίζουν, τη λύση αυτού, τα οφέλη και τις συνέπειες από την εφαρμογή τους. Αυτά είναι:

- Ο *«Γρήγορος Δρόμος»*: Εντοπισμός των λειτουργιών υψηλού υπολογιστικού φόρτου και αποδοτική οργάνωση αυτών με έμφαση στην εκτέλεση μόνο των απολύτως απαραίτητων.
- Το *«Πρώτα τα Σημαντικά»*: Αν δεν υπάρχει βεβαιότητα για την ολοκλήρωση της επεξεργασίας όλων των προς εκτέλεση εργασιών μέσα στο διαθέσιμο χρόνο, πρέπει πρώτα να δίνεται έμφαση στην ολοκλήρωση των πιο σημαντικών από αυτές.
- Το *«Ταίριασμα»*: Διαμόρφωση της διασύνδεσης των αντικειμένων σύμφωνα με τις πιο συχνές χρήσεις αυτών. Με τον τρόπο αυτό ελαχιστοποιούνται τα κόστη επικοινωνίας.
- Η *«Ομαδοποίηση»*: Ομαδοποίηση και εκτέλεση των κλήσεων εξυπηρέτησης, έτσι ώστε οι όποιες καθυστερήσεις παρατηρούνται κατά την επεξεργασία αυτών να εμφανίζονται μόνο μία φορά.
- Οι *«Εναλλακτικοί Δρόμοι»*: Χωρικό διαμοίρασμα της ζήτησης των συχνά χρησιμοποιούμενων αντικειμένων, δηλαδή κατεύθυνση μέρους αυτής σε άλλα αντικείμενα ή σε άλλες θέσεις.

- Η *«Ελαστικότητα του Χρόνου»*: Χρονικό διαμοίρασμα της ζήτησης των συχνά χρησιμοποιούμενων αντικειμένων, όπου αυτό είναι εφικτό.
- Οι *«Λιγοστές Περιοδικές Λειτουργίες»*: Ελαχιστοποίηση του όγκου των εργασιών, που πρέπει να εκτελούνται σε περιοδικά χρονικά διαστήματα.

Στο [SW01] εισάγονται επίσης και πέντε αντι-υποδείγματα, των οποίων η εμφάνιση σε μία σχεδίαση έχει αρνητικές επιπτώσεις στην απόδοση αυτής. Αυτά είναι:

- Η *«Κλάση Θεός»* ("god" Class): Απαντάται στις περιπτώσεις, που: i) αυτή περιλαμβάνει όλη την επεξεργασία μιας εφαρμογής ή, όταν ii) αυτή κατέχει όλα τα δεδομένα της εφαρμογής. Και στις δύο περιπτώσεις το αποτέλεσμα είναι η εμφάνιση υψηλών επιπέδων κυκλοφορίας μηνυμάτων και ο εκφυλισμός της απόδοσης της εφαρμογής.
- Ο *«Εκτεταμένος Δυναμικός Επιμερισμός»*: Κάνει την εμφάνισή του, όταν μία εφαρμογή - χωρίς να είναι απαραίτητο - δημιουργεί και καταστρέφει μεγάλο αριθμό αντικειμένων. Οι καθυστερήσεις, που προκαλούνται κατά τη δημιουργία και καταστροφή των αντικειμένων αυτών, έχουν αρνητική επίπτωση στην απόδοση της εφαρμογής.
- Το *«Διαρκές Κυνήγι του Θησαυρού»*: Έτσι χαρακτηρίζεται μία σχεδίαση, όπου ένα αντικείμενο κάνει αναζήτηση πληροφοριών σε διαφορετικές θέσεις. Αν κάθε αναζήτηση απαιτεί σημαντικό χρόνο επεξεργασίας, τότε υπάρχει σημαντική αρνητική επίπτωση στην απόδοση της εφαρμογής.
- Η *«Γέφυρα Μιας Διόδου»*: Απαντάται στα σημεία εκείνα της επεξεργασίας της εφαρμογής (π.χ. κατά την προσπέλαση μιας βάσης), όπου υπάρχει η δυνατότητα ταυτόχρονης εκτέλεσης μόνο μιας ή έστω ενός μικρού αριθμού διεργασιών. Στις περιπτώσεις αυτές οι άλλες διεργασίες καθυστερούν περιμένοντας τη σειρά εκτέλεσής τους.
- Η *«Κυκλοφοριακή Συμφόρηση»*: Κάνει την εμφάνισή της, όταν κάποιο πρόβλημα δημιουργεί σωρό καθυστερημένων έργων επεξεργασίας, που έχει ως αποτέλεσμα μία μεγάλη μεταβλητότητα στους χρόνους απόκρισης για σημαντικό χρονικό διάστημα μετά την εξαφάνιση του προβλήματος.

Κεφάλαιο 8 *Σύνοψη*

Συνοψίζοντας, στη διατριβή αυτή, αναδείχθηκε η σημασία της ανάλυσης της απόδοσης στην ανάπτυξη λογισμικού κατανεμημένης αρχιτεκτονικής. Διατυπώθηκε ένα πλαίσιο συνδυασμένης χρήσης των τεχνικών, που εξυπηρετούν το συγκεκριμένο στόχο και προέκυψε ένα σύνολο αποτελεσμάτων, που όλα μαζί θέτουν τις βάσεις, για μία περαιτέρω ερευνητική προσπάθεια. Θεμελιώθηκε η αναγκαιότητα χρήσης επιλύσιμων και μη αναλυτικά επιλύσιμων μοντέλων και μεταμοντέλων, συχνά ακόμη και μέσα στο πλαίσιο της ίδιας ανάλυσης.

Στο κεφάλαιο 2, έγινε μία ανασκόπηση των διατάξεων λογισμικού κατανεμημένης αρχιτεκτονικής και των πιο σημαντικών από τις σύγχρονες τεχνολογίες middleware, που υποστηρίζουν την ανάπτυξη ενός τέτοιου συστήματος.

Στο κεφάλαιο 3, παρουσιάστηκε μία αναδρομή στις πιο σημαντικές εξελίξεις, που αφορούν την επίλυση δικτύων ουρών. Έγινε επίσης περιγραφή δύο χαρακτηριστικών μοντέλων απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής. Τέλος, έγινε μία σύντομη εισαγωγή στη χρήση μοντέλων απόδοσης διαφορετικού τύπου, όπως οι διάφορες κλάσεις μοντέλων δικτύων Petri, αλλά και οι πολλά υποσχόμενες στοχαστικές διεργασιακές άλγεβρες.

Στο κεφάλαιο 4, παρουσιάστηκαν τα σημαντικότερα προβλήματα στην ανάπτυξη μοντέλων παράλληλης/κατανεμημένης προσομοίωσης διακριτών γεγονότων. Ακόμη, έγινε μία σύντομη αναδρομή στις πιο προωθημένες στατιστικές τεχνικές επεξεργασίας των αποτελεσμάτων των πειραμάτων προσομοίωσης. Από αυτές επιλέχθηκε η τεχνική της αναγέννησης, εξαιτίας του ανεπτυγμένου θεωρητικού υπόβαθρου, που διαθέτει, της δυνατότητας παραγωγής ανεξάρτητων παρατηρήσεων, αλλά και της δυνατότητας διενέργειας ανάλυσης ευαισθησίας και βελτιστοποίησης από τα αποτελέσματα ενός μόνο πειράματος προσομοίωσης. Μέσα από μία συστηματική βιβλιογραφική έρευνα, κατέστη δυνατή η επίλυση του προβλήματος αρχικοποίησης των μοντέλων, σε κατάσταση αναγέννησης και ο αποτελεσματικός έλεγχος της διάρκειας, αλλά και της ακρίβειας των αποτελεσμάτων των πειραμάτων προσομοίωσης.

Αναπτύχθηκε λογισμικό προσομοίωσης αναγέννησης δικτύων ουρών με πολλαπλούς τύπους έργων, στοχαστική δρομολόγηση, διαφορετικές πειθαρχίες εξυπηρέτησης, παθητικούς πόρους και κέντρων διάσπασης και ένωσης έργων επεξεργασίας. Η τεχνική της αναγέννησης δοκιμάστηκε πειραματικά σε πολύπλοκα μοντέλα και η παράλληλη έκδοση του λογισμικού

προσομοίωσης βοήθησε στη διατύπωση ενός πλαισίου χρήσης αυτής, που εγγυάται την ακρίβεια των αποτελεσμάτων σε διάταξη αποτελούμενη από πολλούς επεξεργαστές.

Μέσα από βιβλιογραφική έρευνα, έγινε επίσης θεωρητική θεμελίωση της δυνατότητας χρήσης της τεχνικής σε ιεραρχικά (υβριδικά) μοντέλα, όπως αυτά που προκύπουν κατά την ανάλυση της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής. Απαραίτητη προϋπόθεση, για τη λήψη αποτελεσμάτων ικανοποιητικής ακρίβειας, είναι η όσο το δυνατόν σχετικά μικρή αλληλεπίδραση των υποσυστημάτων, που βρίσκονται σε διαφορετικά επίπεδα της ιεραρχίας του μοντέλου. Τέλος, στην τελευταία παράγραφο του κεφαλαίου καταγράφεται πλήθος προβλημάτων, που χρήζουν διεξοδικότερης θεωρητικής και πειραματικής διερεύνησης.

Στο κεφάλαιο 5, έγινε συνοπτική παρουσίαση μιας ιεραρχικής μοντελικής προσέγγισης, που επιλέχθηκε ως κατάλληλη, για την ανάπτυξη μοντέλων απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής, μέσα από τη χρήση ενός περιβάλλοντος μοντελοποίησης, όπως αυτό του HIT. Η συγκεκριμένη προσέγγιση πλαισιώθηκε από ένα σύνολο διαδικασιών, που υποστηρίζει τη μοντελοποίηση σε περισσότερα του ενός επίπεδα αφαίρεσης και την εφαρμογή της κατάλληλης τεχνικής ανάλυσης στο επίπεδο εκείνο, που κρίνεται ως το πλέον πρόσφορο, όσον αφορά τις απαιτήσεις κόστους - αξιοπιστίας της κάθε μελέτης απόδοσης. Μέσα από την ανασκόπηση της λειτουργίας ενός σύγχρονου middleware διεργασιακής επικοινωνίας τύπου CORBA, παρουσιάστηκαν μερικά από τα πιο συχνά εμφανιζόμενα προβλήματα στην ανάπτυξη μοντέλων απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής και δείχθηκε πως αυτά είναι δυνατό να ξεπερασθούν με τη χρήση συγκεκριμένων μοντελικών επιλογών. Τέλος, στην ενότητα 5.2.4 επισημαίνεται η αναγκαιότητα εξέλιξης του προτεινόμενου πλαισίου διαδικασιών ανάπτυξης μοντέλων, έτσι ώστε:

- Να είναι δυνατή η περιγραφή μοντέλων απόδοσης με τη χρήση των επεκτάσεων της UML, που προτείνονται στην [AIR01].
- Να επιτευχθεί ένα υψηλότερο επίπεδο συστηματοποίησης, με σαφή καθορισμό των δράσεων, των προϊόντων, που χρησιμοποιούνται και παράγονται, και των κριτηρίων μέτρησης των δράσεων και των προϊόντων των διαδικασιών.

Στο κεφάλαιο 6, έγινε μία πρώτη προσέγγιση των δράσεων προσαρμογής μεταμοντέλων και ανάλυσης ευαισθησίας και των προβλημάτων, που αυτές αντιμετωπίζουν. Επίσης, παρουσιάστηκε μία απλή εφαρμογή χρήσης της προτεινόμενης ιεραρχικής μοντελικής προσέγγισης και των κατάλληλων δράσεων για την ανάλυση ευαισθησίας και τη βελτιστοποίηση του συστήματος. Ακόμη, στην παράγραφο 6.3 καταγράφονται θέματα μεγάλου ερευνητικού ενδιαφέροντος.

Τέλος, στο κεφάλαιο 7, περιγράφηκε η διαδικασία Σχεδίασης της Απόδοσης Λογισμικού, όπως αυτή προτείνεται από την Connie Smith στα [SMI90] και [SMI01] και του τρόπου με τον οποίο το πλαίσιο διαδικασιών ανάπτυξης μοντέλων, που προτείνεται στο κεφάλαιο 5, εντάσσεται σε αυτήν.

Στη διατριβή αυτή δε θίχθηκαν τα εξίσου σημαντικά θέματα μοντελοποίησης της αξιοπιστίας, της διαθεσιμότητας, αλλά και της απόδοσης λογισμικού κατανεμημένης αρχιτεκτονικής σε καταστάσεις παροχής μειωμένων δυνατοτήτων υπολογιστικής επεξεργασίας. Ωστόσο, η ραγδαία ανάπτυξη των στοχαστικών διεργασιακών αλγεβρών (βλ. παράγραφο 3.5), κατά τα τελευταία χρόνια, είναι σαφές ότι προσφέρει μία προοπτική ενιαίας αντιμετώπισης των συγκεκριμένων προβλημάτων, μαζί με αυτό της απόδοσης, μέσω της χρήσης του ιδίου μοντέλου στοχαστικής διαδικασίας και της με διαφορετικούς τρόπους ανάλυσης αυτού.

Κλείνοντας, είναι σημαντικό να τονιστεί ότι η μεγάλη πρόκληση του χώρου παραμένει η δυνατότητα ανάπτυξης μοντέλων, που πέρα από τη δυνατότητα ανάλυσης ποσοτικών χαρακτηριστικών, όπως αυτά της απόδοσης του συστήματος, θα δίνουν και τη δυνατότητα ανάλυσης λειτουργικών ιδιοτήτων αυτού, όπως για παράδειγμα την πιθανότητα μπλοκαρίσματος επεξεργασίας ή την πιθανότητα προσέγγισης κάποιας συγκεκριμένης κατάστασης.

Αναφορές

[AM98] Abdul-Fatah, I. and Majumdar S. *Performance Comparison of Architectures for Client-Server Interactions in CORBA*. In Proceedings of the IEEE International Conference on Distributed Computing Systems, 2-11, 1998

[ADL95] Adler, R. M. *Distributed Coordination Models for Client/Server Computing*. IEEE Computer, 14-22, April 1995

[AT82] Agre, J. R. and Tripathi, S. K. *Modeling Reentrant and Nonreentrant Software*. Proceedings of the ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, 163-178, 1982

[AL79] Almes, G. T. and Lazowska, E. D. *The behaviour of Ethernet-like computer communications networks*. In Proceedings of the 7th Symposium on Operating Systems Principles, 1979

[AIR01] ARTiSAN Software Tools, Inc., I-Logix Inc., Rational Software Corp., Telelogic AB, TimeSys Corporation, Tri-Pacific Software Inc. *Response to the OMG RFP for Schedulability, Performance, and Time*. Selic, B. and Moore, A. (ed.), Revised Submission, OMG document number: ad/2001-06-14, 2001

[BMT98] Bagrodia, R., Meyer, R., Takai, M., Chen, Y., Zeng, X., Martin, J., Park, B., Song, H. *Parsec: A Parallel Simulation Environment for Complex Systems*, IEEE Computer, 31 (10), 77-85, 1998

[BS96] Balbo, G. and Serazzi, G. *Asymptotic analysis of multiclass closed queueing networks: Common bottleneck*. Performance Evaluation, 26, 51-72, 1996

[BAR76] Barbour, A. D. *Networks of queues and the method of stages*. Advances in Applied Probability, 8, 584-591. 1976

[BAR79] Bard, Y. *Some Extensions to Multiclass Queueing Network Analysis*. In M. Arato, A. Butrimenko and E. Gelenbe (ed.): Performance of Computer Systems, North-Holland, 1979

[BAR94] Barton, R. R. *Metamodeling: A state of the art review*, In Proceedings of the Winter Simulation Conference, 237-244, 1994

[BCM75] Baskett, F., Chandy, K. M., Muntz, R. R. and Palacios, F. G. *Open, Closed, and Mixed Networks of Queues with Different Classes of Customers*. Journal of the ACM, 22 (2), 248-260, 1975

[BD96] Baynat, B. and Dallery, Y. *A Product-Form Approximation Method for General Closed Queueing Networks with Several Classes of Customers*, Performance Evaluation, 24, 165-188, 1996

[BS87] Beilner, H. and Stewing, F. J. *Concepts and techniques of the performance modeling tool HIT*, Proc. of the 1987 European Simulation Multiconference, Vienna, Austria, 1987

[BEI89] Beilner, H. *Structured Modeling – Heterogeneous Modeling*. Proc. of the 1989 European Simulation Multiconference, Rome, 1989

[BMW88] Beilner, H., Mater, J. and Weiβenberg, N. *Towards a performance modeling environment: News on HIT*, Proc. of the 4th International Conference on Modeling Techniques and Tools for Computer Performance Evaluation, Palma de Mallorca, Plenum Publishing Corporation, 1988

[BCS98] Bernardo, M., Cleaveland, W. R., Sims, S. T. and Stewart, W. J. *Two Towers: a tool integrating functional and performance analysis of concurrent systems*, In FORTE/PSTV, 1998

[BER92] Berson, A. *Client/Server Architecture*. McGraw-Hill, New York, 1992

[BK96] Bettonvil, B. and Kleijnen, J. P. C. *Searching for important factors in simulation models with many factors: Sequential bifurcation*, European Journal of Operational Research, 96, 180-194, 1996

[BJS86] Bohachevsky, I. O., Johnson, M. E. and Stein, M. L. *Generalized simulated annealing for function optimization*, Technometrics, 28 (3), 209-217, 1986

[BB89] Bolognesi, T., and Brinksma, E., *Intorduction to the ISO specification language LOTOS*, In: P. H. J. van Eijk, C. A. Vissers, M. Diaz (Ed.): The Formal Description Technique LOTOS, North-Holland, Amsterdam, 23-73, 1989

[BE96] Booch, G. and Eykholt, E. (ed.) *The best of Booch*, SIGS Books, New York, 1996

[BBG98] Bravetti, M., Bernardo, M. and Gorrieri, R. *Towards Performance Evaluation with General Distributions in Process Algebras*, In Proceedings of the 9th International Conference on Concurrency Theory (CONCUR'98), Nice, France, LNCS 1466, 405-422, 1998

[BMR96] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P. and Stal, M. *Pattern – Oriented Software Architecture: A System of Patterns*, Wiley, 1996

[BUZ73] Buzen, J. P. *Computational algorithms for closed queueing networks with exponential servers*. Communications of ACM, 16, 527-531, 1973

[CM81] Chandy, K. M. and Misra, J. *Asynchronous Distributed Simulation via a Sequence of Parallel Computations*, Communications of the ACM, 24 (11), 198-206, 1981

[CM79] Chandy, K. M. and Misra, J. *Distributed Simulation: A Case Study in Design and Verification of Distributed Programs*, IEEE Transactions on Software Engineering, 5 (5), 440-452, 1979

[CN82] Chandy, K. M. and Neuse, D. *Linearizer: A heuristic algorithm for queueing network models of computer systems*. Communications of the ACM, 25, 126-134, 1982

[CHW75] Chandy, K. M., Herzog, U. and Woo, L. *Parametric analysis of queueing networks*, IBM Journal of Research and Development, 19, 36-42, 1975

[CHT77] Chandy, K. M., Howard, J. H. and Towsley, D. F. *Product form and local balance in queueing networks*. Journal of the ACM, 24, 250-263. 1977

[CMH83] Chandy, K. M., Misra, J. and Haas, L. M. *Distributed Deadlock Detection*, ACM Transactions on Computer Systems, 1 (2), 144-156, 1983

[CS78] Chandy, K. M. and Sauer, C. H. *Approximate Methods for Analyzing Queueing Network Models of Computing Systems*, ACM Computing Surveys, 10 (3), 281-317, 1978

[CK88] Charnes, J. M. and Kelton, W. D. *A comparison of confidence region estimators for multivariate simulation output*, In Proceedings of the Winter Simulation Conference, 458-465, 1988

[CK99] Cheng, R. C. H. and Kleijnen, J. P. C. *Improved Design of Queueing Simulation Experiments with Highly Heterosxedastic Responses*, Operations Research, 47 (5), 762-777, 1999

[CLL99] Choquet, D., L' Ecuyer, P. and Leger, C. *Bootstrap Confidence Intervals for Ratios of Expectations*, ACM Transactions on Modeling and Computer Simulation, 9 (4), 326-348, 1999

[CLA00] Clark, G. *Techniques for the Construction and Analysis of Algebraic Performance Models*, Ph.D. Thesis, University of Edinburgh, Edinburgh, 2000

[COH79] Cohen, J. W. *The multiple phase service network with generalized processor sharing*. Acta Informatica, 12, 245-284. 1979

[CD96] Colbourn, J. C. and Dinitz, J. H. *The CRC handbook of combinatorial designs*, CRC Press Inc., 1996

[COU75] Courtois, P. J. *Decomposability, instabilities and saturation in multiprogramming systems*, Communications of the ACM, 18, 371-377, 1975

[DB78] Denning, P. J. and Buzen, J. P. *The Operational Analysis of Queueing Network Models*. Computing Surveys, 10 (3), 225-261. 1978

[DMO94] Dharnikota, S., Maly, K. and Overstreet, C. M. *Performance Evaluation of TCP(UDP)/IP over ATM networks*. Department of Computer Science, Technical Report CSTR_94_23, Old Dominion University, 1994

[DFJ98] Dilley, J., Friedrich R., Jin, T. and Rolia J. *Web server performance measurement and modeling techniques*. Performance Evaluation, 33, 5-26, 1998

[DF98] Donatelli, S. and Franceschinis, G. *Modelling and Analysis of Distributed Software using GSPNs*, In W. Reisig, G. Rozenberg (ed.): Lectures on Petri Nets II - Applications, LNCS 1492, Springer - Verlag, 1998

[FK67] Fishman, G. S. and Kiviat, P.J. *The analysis of simulation-generated time series*, Management Science, 13, 525-557, 1967

[FIS78b] Fishman, G. S. *Grouping observations in digital simulation*, Management Science, 24, 510-521, 1978

[FIS78a] Fishman, G. S. *Principles of Discrete Event Simulation*, John Wiley, New York, 1978

[FIS83] Fishman, G. S., *Accelerated accuracy in the simulation of Markov chains*, Operations Research, 31, 466-486, 1983

[FIS77] Fishman, G. S., *Achieving specific accuracy in simulation output analysis*, Communications of the ACM, 20, 310-315, 1977

[FD89] Flowers, F. J. and Dowdy, L. W. *A Comparison of Calibration Techniques for Queueing Networks Models*. In Proceedings of the 15th International Conference of the Computer Measurement Group, 1989

[FRI91] Friedman, J. H. *Multivariate Adaptive Regression Splines*, Annals of Statistics, 19 (1), 1-141, 1991

[FG98] Frolund, S. and Garg, P. *Design-Time Simulation of a Large-Scale, Distributed Object System*. ACM Transactions on Modeling and Computer Simulation, 8 (4), 1998

[GHJ95] Gamma, E., Helm, R., Johnson, R. and Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995

[GEL94] Gelenbe, E. *G-networks: A unifying model for neural and queueing networks*. Annals of Operations Research, 48, 433-461, 1994

[GEL91] Gelenbe, E. *Product-form queueing networks with negative and positive customers*. Journal of Applied Probability, 28, 656-663, 1991

[GGS91] Gelenbe, E., Glynn, P. and Sigman, K. *Queues with negative arrivals*. Journal of Applied Probability, 28, 245-250, 1991

[GH94] Gilmore, S. and Hillston, J. *The PEPA workbench: a tool to support a process algebra-based approach to performance modeling*, In Proceedings of the 7th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation, Wien, 353-368, 1994

[GG92] Glasserman, P. and Glynn, P. W. *Gradient Estimation for Regenerative Processes*, In Proceedings of the Winter Simulation Conference, 280-288, 1992

[GLY82a] Glynn, P. W. *Coverage error for confidence intervals arising in simulation output analysis*, In Proceedings of the ACM/IEEE Winter Simulation Conference, San Diego, California, 369-375, 1982

[GLY82b] Glynn, P. W. *Simulation Output Analysis for General State Space Markov Chains*, Ph.D. thesis, Department of Operations Research, Stanford University, Stanford, California, 1982

[GLY87] Glynn, P. W. *Likelihood Ratio Gradient Estimation: An Overview*, In Proceedings of the Winter Simulation Conference, 366-375, 1987

[GLY89] Glynn, P. W. A *GSMP formalism for discrete event systems*, Proceedings of the IEEE, 77, 14-23, 1989

[GLY90] Glynn, P. W. *Likelihood Ratio Gradient Estimation for Stochastic Systems*, Communications of the ACM, 33 (10), 75-84, 1990

[GLA91] Glynn, P. W., L' Ecuyer, P. and Ades, M. *Gradient Estimation for Ratios*, In Proceedings of the Winter Simulation Conference, 986-993, 1991

[GS97a] Gokhale, A. S. and Schmidt, D. C. *Evaluating CORBA Latency and Scalability Over High-Speed ATM Networks*. Proceedings of International Conference of Distributed Computing Systems, Baltimore, Maryland, 1997

[GS97b] Gokhale, A. S. and Schmidt, D. C. *Evaluating the Performance of Demultiplexing Strategies for Real-time CORBA*. Proceedings of the GLOBECOM'97, Phoenix, AZ, 1997

[GS96] Gokhale, A. S. and Schmidt, D. C. *Measuring the Performance of Communication Middleware on High-Speed Networks*. Proceedings of the ACM SIGCOMM'96, Stanford, CA, 1996

[GJL99] Greiner, M., Jobmann, M. and Lipsky, L. *The importance of power-tail distributions for modeling queueing systems*. Operations Research, 47 (2), 313-326, 1999

[HS86] Haas, P. J. and Shedler, G. S. *Regenerative Stochastic Petri Nets*, Performance Evaluation, 6, 189-204, 1986

[HL84] Heidelberger, P. and Lavenberg, S. S. *Computer Performance Evaluation Methodology*. IEEE Transactions on Computers, 33 (12), 1195-1220, 1984

[HT82] Heidelberger, P. and Trivedi, K. S. *Queueing network models for parallel processing with asynchronous tasks*. IEEE Transactions on Computers, 31, 1099-1109, 1982

[HW81] Heidelberger, P. and Welch, P. D. *A spectral method for confidence interval generation and run length control in simulations*, Communications of the ACM, 24 (4), 233-245, 1981

[HEI86] Heidelberger, P. *Statistical analysis of parallel simulations*, In Proceedings of the Winter Simulation Conference, 1986

[HSV96] Hellemans, P., Steegmans, F., Vanderstraeten, H. and Zuidweg, H. *Implementation of Hidden Concurrency in CORBA Clients*. In: Spaniol, O., Linnhoff-Popien, C., Meyer, B. (ed.), Trends in Distributed Systems, CORBA and Beyond, LNCS 1161, Springer-Verlag, Aachen, Germany, 1996

[HG99] Henderson, S. G. and Glynn, P. W. *Can the regenerative method be applied to discrete-event simulation?*, In Proceedings of the Winter Simulation Conference, 367-373, 1999

[HHK00] Hermanns, H., Herzog, U., Klehmet, U., Mertsiotakis, V. and Siegle, M. *Compositional performance modeling with the TIPPtool*, Performance Evaluation, 39, 5-35, 2000

[HIL94] Hillston, J. *A compositional approach to performance modeling*, Ph.D. Thesis, University of Edinburgh, Edinburgh, 1994

[HOA85] Hoare, C. A. R. *Communicating Sequential Processes*, Prentice-Hall, Englewood Cliffs, NJ, 1985

[IL79] Iglehart, D. L. and Lewis, P. A. W. *Regenerative simulation with internal controls*, Journal of the ACM, 26, 271-282, 1979

[IGL78] Iglehart, D. L. *The regenerative method for simulation analysis*, In K. M. Chandy and P. T. Yeh (ed.): Current Trends in Programming Methodology, Vol. III, Software Modeling, Prentice Hall, Englewood Cliffs, NJ, 52-71, 1978

[ITU96] International Telecommunication Union, *Criteria for the Use and Applicability of Formal Description Techniques, Message Sequence Chart (MSC)*, 1996

[JBR99] Jacobson, I., Booch, G. and Rumbaugh, J. *The Unified Software Development Process*, Reading, Addison-Wesley, 1999

[JAI91] Jain, R. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, New York, 1991

[JEF85] Jefferson, D. A. *Virtual Time*. ACM Transactions on Programming Languages and Systems, 7 (3), 404-425, 1985

[JS85] Jefferson, D. and Sowizral, H. *Fast Concurrent Simulation Using the Time Warp Mechanism*. In P. Reynolds (ed.): Distributed Simulation, SCS – The Society for Computer Simulation, California, 63-69, 1985

[KAH00] Kahkipuro, P. *Performance Modeling Framework for CORBA Based Distributed Systems*. Phd Thesis, Report A-2000-3. Department of Computer Science, University of Helsinki, 2000

[KAH01] Kahkipuro, P. *UML-Based Performance Modeling Framework for Component-Based Distributed Systems*. In R. Dumke et al. (ed.): Performance Engineering. LNCS 2047, Springer-Verlag, 167-184, 2001

[KG85] Kang, K. and Goldsman, D. *The correlation between mean and variance estimators*. In Proceedings of the ACM/IEEE Winter Simulation Conference, San Francisco, California, 211-216, 1985

[KAR94] Karatza, H. D. *Simulation Study of a System with Two Processors Linked in Tandem*. Journal of Systems and Software, 26, 285-292, 1994

[KAR98] Karatza, H. D. *Task routing and resequencing in a multiprocessor system*. Journal of Systems and Software, 41, 189-197, 1998

[KAR00] Karatza, H. D. *A comparative analysis of scheduling policies in a distributed system using simulation*. International Journal of Simulation, 1, 12-20, 2000

[KAR01a] Karatza, H. D. *Task cluster scheduling and I/O scheduling in a workstation cluster*. In Proceedings of the 15th European Simulation Multiconference, Society for Computer Simulation, Prague, The Czech Rep., 2001

[KAR01b] Karatza, H. D. *Epoch scheduling in a distributed system*. In Proceedings of the 4th International EUROSIM Congress, Delft, The Netherlands, 2001

[KL00a] Katsaros, P. and Lazos, C. *A technique for determining queuing network simulation length based on desired accuracy*. International Journal of Computer Systems Science & Engineering, 15 (6), 399-404, 2000

[KL00b] Katsaros, P. and Lazos, C. *Regenerative Queuing Network Distributed Simulation*. In Proceedings of the 14th European Simulation Multiconference, Ghent, Belgium, 109-113, 2000

[KL01a] Katsaros, P. and Lazos, C. *Shared memory parallel regenerative queueing network simulation*. In Proceedings of the 15th European Simulation Multiconference, Society for Computer Simulation, Prague, The Czech Rep., 736-740, 2001

[KL01b] Katsaros, P. and Lazos, C. *Steady-state Simulation of Queuing Processes in Parallel Time Streams: Problems and Potentialities*, In Proceedings of the 5th Hellenic European Research Conference on Computer Mathematics and its Applications, Athens, Greece, 2001

[KAT93] Katsaros, P. *Computer Systems Performance Modelling*, MSc Thesis, Aston University in Birmingham, 1993

[KAL01] Katsaros, P., Angelis, E. and Lazos, C. *Applied Multiresponse Metamodeling for Queueing Network Simulation Experiments: Problems and Perspectives*. In Proceedings of the 4th International EUROSIM Congress, Delft, The Netherlands, 2001

[KP93] Kay, J. and Pasquale, J. *The Importance of Non-Data Touching Processing Overheads in TCP/IP*. In Proceedings of SIGCOMM'93, ACM, San Francisco, CA, 259-269, 1993

[KEL75] Kelly, F. P. *Networks of queues with customers of different types*. Journal of Applied Probability, 542-554. 1975

[KEL76] Kelly, F. P. *Networks of queues*. Advances in Applied Probability, 416-432. 1976

[KEL79] Kelly, F. P. *Reversibility and Stochastic Networks*. Wiley, New York, 1979

[KL84] Kelton, W. D. and Law, A. M. *An analytical evaluation of alternative strategies in steady-state simulation*, Operations Research, 32, 169-184, 1984

[KHU96] Khuri, A. I. *Analysis of multiresponse experiments: A review*, In S. Ghosh (ed.): Statistical Design and Analysis of Industrial Experiments, Marcel Dekker, New York, 231-246, 1996

[KS00] Kleijnen, J. P. C. and Sargent, R. G. *A methodology for fitting and validating metamodels in simulation*, European Journal of Operational Research, 120, 14-29, 2000

[KLE87] Kleijnen, J. P. C. *Statistical Tools for Simulation Practitioners*, Marcel Dekker, New York, 1987

[KD80] Kumar, B. and Davidson, E. S. *Computer System Design Using a Hierarchical Approach to Performance Evaluation*. Communications of the ACM, 23 (9), 511-521, 1980

[LAV89] Lavenberg, S. S. *A Perspective on Queueing Models of Computer Performance*. Performance Evaluation, 10, 53-76, 1989

[LS77] Lavenberg, S. S. and Sauer, C. H. *Sequential stopping rules for the regenerative method of simulation*, IBM Journal of Research and Development, 21, 545-558, 1977

[LMW82] Lavenberg, S. S., Moeller, T. L. and Welch, P. D. *Statistical results on control variables with application to queueing network simulation*, Operations Research, 30, 182-202, 1982

[LC79] Law, A. M. and Carson, J. C. *A sequential procedure for determining the length of a steady state simulation*, Operations Research, 27, 1011-1025, 1979

[LK82] Law, A. M. and Kelton, W. D. *Simulation Modeling and Analysis*, McGraw-Hill, New York, 1982

[LK82b] Law, A. M. and Kelton, W. D. *Confidence intervals for steady-state simulations II: A survey of sequential procedures*, Management Science, 28 (5), 550-562, 1982

[LZGS84] Lazowska, E. D., Zahorjan, J., Graham, G. S. and Sevcik, K. C. *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*. Prentice-Hall, Englewood Cliffs, NJ, 1984

[LMP99] Lee, J. R., McNickle, D. and Pawlikowski, K. *Quality of sequential regenerative simulation*, In Proceedings of the 13th European Simulation Multiconference, Warsow, Poland, 161-167, 1999

[LO89] Lewis, P. A. W. and Orav, E. J. *Simulation Methodology for Statisticians, Operations Analysts*, and Engineers: Volume 1, Wadsworth & Brooks/Cole, California, 1989

[LR91] Lewis, P. A. W. and Ressier, R. L. *Average Regression - Adjusted Controlled Regenerative Estimates*, In Proceedings of the Winter Simulation Conference, 921-926, 1991

[LGM69] Lewis, P. A., Goodman, A. S. and Miller, J. M. *A pseudo-random number generator for the System/360*, IBM Systems Journal, 8 (2), 136-146, 1969

[LRS98] Litoiu, M., Rolia, J. and Serazzi, G. *Designing Process Replication and Threading Policies: a Quantitative Approach*. In Tools'98, LNCS 1469, Springer-Verlag, 15-26, 1998

[LIT61] Little, J. D. C. *A Proof for the Queueing Formula: $L=\lambda W$*. Operations Research, 9 (3), 383-387. 1961

[LN91] Liu, Z. and Nain, P. *Sensitivity results in open, closed, and mixed product-form queueing networks*. Performance Evaluation, 13, 237-251, 1991

[LH00] Llado, C. M. and Harrison, P. G. *Performance Evaluation of an Enterprise JavaBean Server Implementation*, In Proceedings of the ACM Workshop on Software and Performance, 2000

[LD98] Loosley, C. and Douglas, F. *High-Performance Client/Server*. John Wiley & Sons, New York, 1998

[MLH01] Majumdar, S., Luthi, J., Haring, G. and Ramadoss, R. *Characterization and Analysis of Software and Computer Systems with Uncertainties and Variabilities*. In R. Dumke et al. (Eds.): Performance Engineering, Lecture Notes in Computer Science 2047, Springer-Verlag, 202-221, 2001

[MBD98] Marsan, M. A., Bobbio, A. and Donatelli, S. *Petri Nets in Performance Analysis: An Introduction*, In W. Reisig, G. Rozenberg (ed.): Lectures on Petri Nets I - Basic Models, LNCS 1491, Springer - Verlag, 1998

[MH99] Matena, V. and Hapner, M. *Enterprise JavaBeans 1.1 Specification*. Public Release, Sun Microsystems Inc., August 10, 1999

[MK94] Matsumoto, M. and Kurita, Y. *Twisted GFSR Generators II*, ACM Transactions on Modeling and Computer Simulation, 4 (3), 254-266, 1994

[MK92] Matsumoto, M. and Kurita, Y. *Twisted GFSR Generators*, ACM Transactions on Modeling and Computer Simulation, 2 (3), 179-194, 1992

[MN98] Matsumoto, M. and Nishimura, T. *Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator*, ACM Transactions on Modeling and Computer Simulation, 8, 3-30, 1998

[MS84] Meketon, M. S. and Schmeiser, B. *Overlapping batch means: Something for nothing?*, In Proceedings of the ACM/IEEE Winter Simulation Conference, Dallas, Texas, 227-230, 1984

[MA98] Menasce, D. A. and Almeida, V. A. F. *Capacity Planning for Web Performance: Metrics, Models & Methods*. Prentice Hall, New Jersey, 1998

[MAD94] Menasce, D. A., Almeida, V. A. F. and Dowdy, L. W. *Capacity Planning and Performance Modeling*. Prentice Hall, Englewood Cliffs, NJ, USA, 1994

[MIL89] Milner, R. *Communication and Concurrency*, Prentice-Hall, London, 1989

[MIS86] Misra, J. *Distributed Discrete – Event Simulation*, ACM Computing Surveys, 18 (1), 39-65, 1986

[MLC98] MLC Systeme, Charles University. *CORBA Comparison Project – Final Project Report*. MLC Systeme GmbH, Rattingen, Germany, Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic, 1998

[MKK94] Modeklev, K., Klovning, E. and Kure, O. *TCP/IP Behavior in a High-Speed Local ATM Network Environment*. In Proceedings of the 19th Conference on Local Computer Networks, IEEE, Minneapolis, MN, 176-185, 1994

[MON91] Montgomery, D. C. *Design and Analysis of Experiments*, 3rd ed., John Wiley & Sons, 1991

[MUR89] Murata, T. *Petri Nets, Properties, Analysis and Applications*, Proceedings of the IEEE, 77 (4), 541-580, 1989

[NWP95] Neilson, J. E., Woodside, C. M., Petriu, D. C. and Majumdar, S., *Software Bottlenecking in Client-Server Systems and Rendez-vous Networks*. IEEE Trans. on Software Engineering, 21 (9), 776-782, 1995

[NH96] Nicol, D. and Heidelberger, P. *Parallel Execution for Serial Simulators*, ACM Transactions on Modeling and Computer Simulation, 6 (3), 210-242, 1996

[OMG92] Object Management Group, Richard Mark Soley (ed.), *Object Management Architecture Guide*, OMG TC Document 92.11.1, Revision 2.0, John Wiley & Sons, New York, USA, 1992

[OMG97] Object Management Group, *The Common Object Request Broker: Architecture and Specification*, Revision 2.1, Framingham, Massachusetts, USA, 1997

[OMG98] Object Management Group, *CORBA Messaging*, OMG TC Document orbos/98-05-05, Framingham, MA, USA, 1998

[OMG99] Object Management Group, *White Paper on Benchmarking*, Version 1.0, OMG TC Document bench/99-12-01, Framingham, MA, USA, 1999

[OPE97] OpenMP Architecture Review Board. *OpenMP: A Proposed Industry Standard API for Shared Memory Programming*, White Paper, 1997

[OHE96] Orfali, R., Harkey, D., and Edwards, J. *The Essential Distributed Objects Survival Guide*. John Wiley & Sons, New York, 1996

[PP93] Papadopoulos, C. and Parulkar, G. *Experimental Evaluation of SUNOS IPC and TCP/IP Protocol Implementation*. IEEE/ACM Transactions on Networking, 1 (2), 199-216, 1993

[PM98] Park, S. K. and Miller, K. W. *Random number generators: Good ones are hard to find*, Communications of the ACM, 31 (10), 1192-1201, 1998

[PAW90] Pawlikowski, K. *Steady-State Simulation of Queueing Processes: A Survey of Problems and Solutions*, ACM Computing Surveys, 22 (2), 123-170, 1990

[PME98] Pawlikowski, K., McNickle, D. C. and Ewing, G. *Coverage of confidence intervals in sequential steady-state simulation*, Simulation Practice and Theory, 6, 255-267, 1998

[PME94] Pawlikowski, K., McNickle, D. C. and Ewing, G. *Distributed stochastic discrete event simulation in parallel time streams*, In Proceedings of the Winter Simulation Conference, 723-730, 1994

[PER94] Perros, H. G. *Queueing Networks with Systems with Multi-Class Requests*. ACM Sigmetrics, 77-86, 1994

[PAM00] Petriu, D., Amer, H., Majumdar, S. and Abdull-Fatah, I. *Using Analytic Models for Predicting Middleware Performance*. In Proceedings of the 2nd ACM International Workshop on Software and Performance (WOSP'2000), Ottawa, Canada, 189-194, 2000

[PSJ00] Petriu, D., Shousha, C. and Jalnapurkar, A. *Architecture-Based Performance Analysis Applied to a Telecommunication System*, IEEE Transactions on Software Engineering, 26 (11), 1049-1065, 2000

[PVR95] Pozzetti, E., Vetland, V., Rolia, J. and Serazzi, G. *Characterizing the Resource Demands of TCP/IP*, Proceedings of the 1995 Conference on High Performance Computing and Networking (HPCN95), 1995

[PRE89] Preiss, B. R. *The Yaddes distributed discrete event simulation specification language and execution environments*, In Distributed Simulation, Society for Computer Simulation, California, 21, 139-144, 1989

[PRI84] Pritsker, A. A. B. *Introduction to simulation and SLAM II*, Systems Publishing Corporation, West Lafayette – Indiana, 1984

[RAA93] Raatikainen, K. E. *A sequential procedure for simultaneous estimation of several means*, ACM Transactions on Modeling and Computer Simulation, 3 (2), 108-133, 1993

[RAA92] Raatikainen, K. *Run Length Control using Parallel Spectral Methods*, In Proceedings of the Winter Simulation Conference, 1992

[RP00] Ramesh, S. and Perros, H. G. *A Multilayer Client - Server Queueing Network Model with Synchronous and Asynchronous Messages*, IEEE Transactions on Software Engineering, 26 (11), 1086-1100, 2000

[RW86] Reiman, M. I. And Weiss, A. *Sensitivity Analysis Via Likelihood Ratios*, In Proceedings of the Winter Simulation Conference, 285-289, 1986

[RK75] Reiser, M. and Kobayashi, H. *Queueing networks with multiple closed chains: Theory and computational algorithms*. IBM Journal of Research and Development, 19, 283-294, 1975

[RL80] Reiser, M. and Lavenberg, S. S. *Mean-value analysis of closed multichain queueing networks.* Journal of the ACM, 27, 313-322. 1980

[REV96] Reverbel, F. *Persistence in Distributed Object Systems: ORB/ODBMS Integration*. Ph. D. Dissertation, University of New Mexico, USA, 1996

[RM51] Robbins, H. and Monroe, S. *On a Stochastic Approximation Technique*, Annals of Mathematical Statistics, 22, 400-407, 1951

[RS95] Rolia, J. A. and Sevcik K. C. *The Method of Layers*. IEEE Transactions on Software Engineering, 21 (8), 689-700, 1995

[RM98] Rubinstein, R. Y. and Melamed, B. *Modern Simulation and Modeling*, Wiley, New York, 1998

[SC81] Sauer, C. H. and Chandy, K. M. *Computer Systems Performance Modeling*, New Jersey, Prentice-Hall, 1981

[SP98] Savino, N. N. and Puigjaner, R. *An object-oriented approach for the design of hierarchical queueing network simulation models with QNAP2*. Proceedings of the Object Oriented Simulation Conference (OOS'98), San Diego, CA, 101-106, 1998

[SAP98] Savino, N. N., Anciano, J. L. and Puigjaner, R. *Including methods and inheritance of hierarchical submodels in HOOMA*. Proceedings of the 30th Summer Computer Simulation Conference (SCSC'98), Reno, Nevada, 293-298, 1998

[SAD00] Savino-Vazquez, N. N., Anciano-Martin, J. L., Dumas, S., Corbacho, J. A., Puigjaner, R., Boudigue, D. and Gardarin, G. *Predicting the behaviour of three-tiered applications: dealing with distributed-object technology and databases*. Performance Evaluation, 39, 207-233, 2000

[SCH82] Schmeiser, B. *Batch size effects in the analysis of simulation output*, Operations Research, 30, 556-568 1982

[SCH98] Schmidt, D. C. *Evaluating Architectures for Multi-threaded CORBA Object Request Brokers*. Communications of the ACM, 41 (10), 54-60, 1998

[SSR00] Schmidt, D., Stal, M., Rohnert, H. and Buschmann, F. *Pattern - Oriented Software Architecture: Patterns for Concurrent and Networked Objects*, Wiley, 2000

[SCH79a] Schrage, L. *A more portable FORTRAN random number generator*, ACM Transactions on Mathematical Software, 5 (2), 132-138, 1979

[SCH80] Schruben, L. W. *A coverage function for interval estimators of simulation response*, Management Science, 26, 18-27, 1980

[SCH79b] Schweitzer, P. *Approximate analysis of multiclass closed networks of queues*. In Proceedings of the International Conference on Stochastic Control and Optimization, Amsterdam, The Netherlands, 1979

[SCH78] Schwetman, H. D. *Hybrid Simulation Models of Computer Systems*, Communications of the ACM, 21 (9), 718-723, 1978

[SHE93] Shedler, G. *Regenerative stochastic simulation*, Academic Press, California, 1993

[SMI90] Smith, C. U. *Performance Engineering of Software Systems*. Assison Wesley, Reading, MA, 1990

[SMI01] Smith, C. U. *Origins of Software Performance Engineering: Highlights and Outstanding Problems*. In R. Dumke et al. (ed.): Performance Engineering. LNCS 2047, Springer-Verlag, 96-118, 2001

[SW98a] Smith, C. U. and Williams, L. G. *Performance Evaluation of a Distributed Software Architecture*. In Proceedings of the Computer Measurement Group, Anaheim, 1998

[SW98b] Smith, C. U. and Williams, L. G. *Performance Engineering Models of CORBA-based Distributed Object Systems*. In Proceedings of the Computer Measurement Group, Anaheim, 1998

[SW01] Smith, C. U. and Williams, L. G. *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*, Addison - Wesley, 2001

[SP97] Somadder, G. and Petriu, D. *A Pattern Language for Improving the Capacity of Layered Client/Server Systems with Multi-Threaded Servers*. In Proceedings of the European Conference on Pattern Languages of Programming (EuroPLOP'97), Irsee, Germany, 179-189, 1997

[SLM86] de Souza e Silva, E., Lavenberg, S. S. and Muntz, R. R. *A clustering approximation technique for queueing network models with a large number of chains*. IEEE Transactions on Computers, 35, 419-430, 1986

[SRK99] Stamelos, J., Refanidis, J., Katsaros, P., Vlachavas, J., Tsoukias, A. and Pombortsis, J., *An Expert System for Educational Software Evaluation*, In Proceedings of the 5th International Conference of the Decision Science Institute, Athens, 1999

[SRK00] Stamelos, J., Refanidis, J., Katsaros, P., Tsoukias, A., Vlachavas, J. and Pombortsis, A. *An Adaptable Framework for Educational Software Evaluation*, In Zanakis & Doukidis (ed.): Decision Making: Recent Developments and Worldwide Applications, Applied Optimization Vol. 45, Kluwer Academics, 2000

[SUR83] Suri, R. *Robustness of Queueing Network Formulas*. Journal of the ACM, 30 (3), 564-594, 1983

[THO98] Thomasian, A. *Concurrency Control: Methods, Performance, and Analysis*. ACM Computing Surveys, 30 (1), 70-119, 1998

[TM87] Toffoli, T. and Margolus, N. *Cellular Automata Machines: a New Environment for Modeling*, MIT Press, 2nd edition, 1987

[TOW80] Towsley, D. *Queueing network models with state-dependent routing*. Journal of the ACM, 27, 323-337. 1980

[TPC00] Transaction Processing Performance Council. *TPC Benchmark W (Web Commerce)*. Version 1.0, 2000

[UMA97] Umar, A. *Object-Oriented Client/Server Internet Environments*. Prentice Hall, New Jersey, 1997

[VIN97] Vinoski, S. *CORBA: Integrating Diverse Applications within Distributed Hterogeneous Environments*. IEEE Communications Magazine, 14 (2), 46-55, 1997

[WWM00]Weiβenberg, N., Wilde, A., Muller-Clostermann B., Shaban, S. and Dittrich, W. *HIT and HI-SLANG: An Introduction*. Universitat Dortmund, Informatik IV, Prof. Dr.-Ing. H. Beilner, Document Version 1.2.00, 2000

[WEL87] Welch, P. D. *On the relationship between batch means, overlapping batch means and spectral estimation*, In Proceedings of the ACM/IEEE Winter Simulation Conference, Atlanta, 320-323, 1987

[WP84b] Wilson, J. R. and Pritsker, A. A. B. *Experimental Evaluation of Variance Reduction Techniques for Queueing Simulation Using Generalized Concomitant Variables*, Management Science, 30 (12), 1459-1472, 1984

[WIL84] Wilson, J. R. *Variance Reduction Techniques for Digital Simulation*, American Journal of Mathematical and Management Sciences, 4 (3 and 4), 277-312, 1984

[WP84a] Wilson, J. R. and Pritsker, A. A. B. *Variance Reduction in Queueing Simulation Using Generalized Concomitant Variables*, Journal of Statistical Computation and Simulation, 19, 129-153, 1984

[WIN93] Winsberg, P. *Database Programming & Design*. Vol. 6 (7), 1993

[WOO88] Woodside, C. M. *Throughput Calculation for Basic Stochastic Rendezvous Networks*. Performance Evaluation, 9 (2), 143-160, 1988

[WNP95] Woodside, C. M., Neilson, J. E., Petriu D. C. and Majumdar, S. *The Stochastic Rendezvous Network Model for Performance of Synchronous Client-Server-like Distributed Software*. IEEE Transactions on Computers, 44 (1), 20-34, 1995

[WVC01] Woodside, M., Vetland, V., Courtois, M. and Bayarov, S. *Resource Function Capture for Performance Aspects of Software Components and Sub-systems*, In R. Dumke et al. (ed.): Performance Engineering. LNCS 2047, Springer-Verlag, 239-256, 2001

[XH96] Xu, Z. and Hwang, K. *Modeling Communication Overhead: MPI and MPL Performance on the IBM SP2*, IEEE Parallel and Distributed Technology, 9-23, Spring 1996

[YAU99] Yau, V. *Automating Parallel Simulation Using Parallel Time Streams*, ACM Transactions on Modeling and Computer Simulation, 9 (2), 171-201, 1999

[ZUS96] Zielinski, K., Uszok, A. and Steinder, M. *Crossing Technological Domains Using the Inter-ORB Request Level Bridge - Preliminary Performance Study*, In Trends in Distributed Systems: CORBA and Beyond, Proceedings of the International Workshop TreDS'96, Springer-Verlag, Aachen, 1996

[VAS89] Βασιλείου, Π. –Χ. Γ., *Στοχαστικές μέθοδοι στις επιχειρησιακές έρευνες*, Εκδόσεις Ζήτη, Θεσσαλονίκη, 1989

[MM90] Μπόρα – Σέντα, Ε. και Μωυσιάδης, Χ. *Εφαρμοσμένη Στατιστική*, Εκδόσεις Ζήτη, Θεσσαλονίκη, 1990

Παράρτημα Α *Στοιχεία Θεωρίας Πιθανοτήτων και Στοχαστικών Διαδικασιών*

Στο παράρτημα αυτό παρουσιάζονται στοιχειώδεις έννοιες και αποτελέσματα της θεωρίας των στοχαστικών διαδικασιών, καθώς αυτά αποτελούν το θεωρητικό υπόβαθρο για πολλά από τα αποτελέσματα, που παρατίθενται στα κεφάλαια 3 και 4 της διατριβής.

Α.1 Τυχαίες μεταβλητές, πιθανότητες και κατανομές

Τα μαθηματικά μοντέλα είναι αφαιρετικές αναπαραστάσεις πραγματικών φαινομένων. Μπορεί να είναι *ντετερμινιστικά* ή *στοχαστικά*. Αν οι συνέπειες μιας αλλαγής στο σύστημα, που αναπαριστά ένα μοντέλο, μπορούν να προβλεφθούν με βεβαιότητα, τότε μιλάμε για ένα ντετερμινιστικό μοντέλο. Αν από την άλλη μεριά οι αλλαγές αυτές μπορούν να εκφρασθούν μαθηματικά μόνο από τυχαίες μεταβλητές, τότε μιλάμε για ένα στοχαστικό μοντέλο.

Τυχαία μεταβλητή είναι οποιαδήποτε συνάρτηση από το σύνολο όλων των πιθανών αποτελεσμάτων ενός τυχαίου συμβάντος, που συνήθως καλείται *δειγματικός χώρος* ή *δειγματοχώρος Ω*, στο σύνολο των πραγματικών αριθμών.

Στην ανάλυση απόδοσης ενδιαφερόμαστε συνήθως για χρόνους επεξεργασίας λογισμικού, που κάνει χρήση των πόρων ενός συστήματος. Όταν παρατηρούμε τους χρόνους αυτούς απευθείας, χωρίς να μπορούμε να προσδιορίσουμε επακριβώς τους παράγοντες που τους καθορίζουν, είναι βολικό να τους θεωρούμε ως τυχαία συμβάντα. Έτσι, μέσα από τη χρήση τυχαίων μεταβλητών μπορούμε τελικά να εξάγουμε χρήσιμα συμπεράσματα.

Ως *συνάρτηση κατανομής* μιας τυχαίας μεταβλητής *X* ορίζεται η ποσότητα,

$$F(x) = P(X \le x), \qquad -\infty < x < \infty$$

η *πιθανότητα* δηλαδή η τυχαία μεταβλητή *X*, να μην είναι μεγαλύτερη από *x*.

Ο Kolmogorov όρισε την πιθανότητα ενός συμβάντος *A*, ως τη συνάρτηση *P*, που ικανοποιεί τα εξής τρία αξιώματα:

1. $0 \le P(A) \le 1, \quad \forall A \in \Omega$
2. $P(\Omega) = 1$

3. Αν $A_i, i=1,...,k$ και $A_i \cap A_j = \varnothing$ για όλα τα i, j τότε,

$$P(A_1 \cup A_2 \cup ... \cup A_k) = P(A_1) + P(A_2) + ... + P(A_k)$$

Παρόλα αυτά, άτυπα μπορούμε να θεωρήσουμε ότι η πιθανότητα ορίζεται από τη *σχετική συχνότητα* ενός συμβάντος.

Αν θεωρήσουμε μια *διακριτή τυχαία μεταβλητή X*, μια μεταβλητή δηλαδή που παίρνει τιμές $x_i, i=1,2,...$ με θετικές πιθανότητες $p_i, i=1,2,...$, τότε ορίζουμε ως *μέση τιμή* αυτής, την

$$E(X) = \sum_i x_i P(X = x_i) = \sum_i x_i p_i$$

Ωστόσο, συχνά η πιθανότητα μία τυχαία μεταβλητή να λάβει κάποια συγκεκριμένη τιμή, δεν μπορεί να οριστεί. Τέτοιες τυχαίες μεταβλητές είναι οι *συνεχείς* μεταβλητές.

Η συνάρτηση κατανομής $F(x)$, λέμε ότι είναι συνεχής, αν υπάρχει συνάρτηση $f(x)$ τέτοια ώστε

$$F(x) = \int_{-\infty}^{x} f(t) \cdot dt \qquad -\infty < x < \infty$$

Όσο για τη συνάρτηση $f(x)$, αυτή ονομάζεται *συνάρτηση πυκνότητας πιθανότητας* ή απλά, *συνάρτηση πυκνότητας*.

Σύμφωνα λοιπόν με τα παραπάνω, ισχύει

$$f(x) = \frac{dF(x)}{dx}$$

Όταν η X είναι συνεχής μεταβλητή, τότε η μέση τιμή αυτής ορίζεται ως,

$$E(X) = \int_{-\infty}^{+\infty} x \cdot f(x) \cdot dx$$

Άλλα χαρακτηριστικά των τυχαίων μεταβλητών είναι οι *ν-στές ροπές*,

$$E(X^n) = \sum_i x_i^n \cdot P(X = x_i) = \sum_i x_i^n \cdot p_i$$

που στην περίπτωση συνεχούς μεταβλητής δίνονται από τη σχέση,

$$E(X^n) = \int_{-\infty}^{+\infty} x^n \cdot f(x) \cdot dx$$

Ιδιαίτερα σημαντικές πάντως είναι και οι επονομαζόμενες *κεντρικές ροπές*:

$$E[(X - E(X))^n] = \sum_i (x_i - E(X))^n \cdot p_i$$

και

$$E[(X - E(X))^n] = \int_{-\infty}^{+\infty} (x - E(X))^n \cdot f(x) \cdot dx$$

Η κεντρική ροπή δεύτερης τάξης ονομάζεται *διασπορά* και η μελέτη της παρουσιάζει ιδιαίτερο ενδιαφέρον για κάθε τυχαία μεταβλητή. Γενικά, είναι εύκολο να αποδειχθεί ότι

$$\sigma^2 = Var(X) = E(X^2) - [E(X)]^2$$

Η τετραγωνική ρίζα της διασποράς ονομάζεται *τυπική απόκλιση*. Συχνά όμως, ως μία ένδειξη της παρατηρούμενης διασποράς, λαμβάνεται υπόψη ο *συντελεστής διασποράς*, που δίνεται από τη σχέση

$$C = \frac{\sigma}{E(X)}$$

Α.2 Η εκθετική κατανομή, η κατανομή Poisson και οι διαδικασίες Markov

Μία από τις σημαντικότερες κατανομές είναι η επονομαζόμενη *εκθετική κατανομή*,

$$F(x) = 1 - e^{-\lambda x} \qquad x \geq 0$$

με παράμετρο (ρυθμό) λ και συνάρτηση πυκνότητας,

$$f(x) = \lambda \cdot e^{-\lambda x} \qquad x \geq 0$$

και τα ακόλουθα χαρακτηριστικά:

$$E(X) = \frac{1}{\lambda}$$

$$\sigma^2 = \frac{1}{\lambda^2}$$

Ένα άλλο χαρακτηριστικό είναι αυτό της *απώλειας μνήμης*, με την έννοια ότι μία εκθετικά κατανεμημένη μεταβλητή στη χρονική στιγμή t δεν επηρεάζει την κατανομή της στο υπόλοιπο της διάρκειας μελέτης της. Το χαρακτηριστικό αυτό όμως είναι συμβατό με τη *ιδιότητα Markov*, η οποία θέλει, αν η κατάσταση μιας στοχαστικής διαδικασίας είναι γνωστή σε μία δεδομένη χρονική στιγμή, τότε η από κει και πέρα συμπεριφορά της, να είναι ανεξάρτητη της προϊστορίας της. Αυτό πρακτικά σημαίνει ότι ένα σύστημα, στο οποίο τα συμβάντα, που αλλάζουν την κατάστασή του, παρουσιάζονται σε χρονικά διαστήματα ανεξάρτητα μεταξύ τους και εκθετικά κατανεμημένα μπορεί να μοντελοποιηθεί ως *διαδικασία Markov*. Καθώς όμως οι διαδικασίες Markov είναι ίσως ένα από τα πιο σημαντικά εργαλεία μελέτης μαθηματικών μοντέλων, εύκολα συνάγεται ο λόγος για τον οποίο είναι επιθυμητή η χρήση της εκθετικής κατανομής στα μοντέλα που εξετάζουμε. Δεν είναι τυχαίο εξάλλου ότι πολλές φορές γίνεται χρήση μιας τεχνικής αναπαράστασης της κατανομής μιας μεταβλητής από διαδοχικά στάδια εκθετικής κατανομής (*μέθοδος εκθετικών σταδίων εξυπηρέτησης*).

Μία άλλη σημαντική κατηγορία διαδικασιών είναι οι διαδικασίες Poisson. Ως *διαδικασία Poisson* ορίζεται αυτή κατά την οποία:

- οι εμφανίσεις συμβάντων σε μη επικαλυπτόμενα χρονικά διαστήματα είναι ανεξάρτητες μεταξύ τους και
- για κάθε στοιχειώδες χρονικό διάστημα $(t, t+\Delta t)$ η πιθανότητα εμφάνισης ενός μόνο συμβάντος κατά τη διάρκεια του διαστήματος είναι $\lambda \cdot \Delta t + o(\Delta t)$ και η πιθανότητα εμφάνισης περισσοτέρων του ενός συμβάντων είναι $o(\Delta t)$, όπου $o(\Delta t)$ είναι αμελητέα ποσότητα.

Είναι εύκολο να αποδειχθεί ότι σε μία τέτοια στοχαστική διαδικασία η πιθανότητα εμφάνισης k συμβάντων στο χρονικό διάστημα $(0, t)$ δίνεται από τη σχέση:

$$p_k = \frac{(\lambda \cdot t)^k}{k!} \cdot e^{-\lambda \cdot t} \qquad k = 0, 1, \ldots .$$

Η κατανομή αυτή είναι γνωστή ως *κατανομή Poisson* με παράμετρο λt. Ο μέσος αριθμός συμβάντων σε ένα χρονικό διάστημα διάρκειας t είναι ίσος με λt.

Αν λάβουμε υπόψη τον ορισμό της *διαδικασίας Poisson*, είναι εύκολο να αποδειχθεί η ισοδυναμία της με διαδικασία κατά την οποία οι μεταξύ συμβάντων χρόνοι της είναι ανεξάρτητοι μεταξύ τους και εκθετικά κατανεμημένοι. Κατά συνέπεια μπορεί να αναπαρασταθεί από μία διαδικασία Markov.

Α.3 Βασικά αποτελέσματα για τις αλυσίδες Markov

Ας θεωρήσουμε την περίπτωση *διαδικασιών Markov διακριτών καταστάσεων*. Στις διαδικασίες αυτές, οι καταστάσεις τους μπορούν να αριθμηθούν από το σύνολο ή ένα υποσύνολο των φυσικών αριθμών. Αν επιπλέον και η παράμετρος του χρόνου είναι και αυτή διακριτή, τότε η διαδικασία αυτή ονομάζεται *αλυσίδα Markov*. Μία άμεση συνέπεια της ιδιότητας Markov είναι το γεγονός ότι μία αλυσίδα Markov μπορεί να περιγραφεί από τις πιθανότητες $p_{ij}(t)$ μετάβασης της αλυσίδας στην κατάσταση j κατά τη χρονική στιγμή t+1, δοθέντος ότι κατά τη χρονική στιγμή t βρίσκεται στην κατάσταση i. Αν επιπλέον οι πιθανότητες αυτές δεν εξαρτώνται από την παράμετρο του χρόνου, τότε οι διαδικασίες αυτές ονομάζονται *ομογενείς διαδικασίες*.

Κατά συνέπεια, μία ομογενής αλυσίδα Markov περιγράφεται πλήρως από τον *πίνακα πιθανοτήτων μετάβασης*,

$$\begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0n} \\ p_{10} & p_{11} & \cdots & p_{1n} \\ \cdots & \cdots & \cdots & \cdots \\ p_{n0} & p_{n1} & \cdots & p_{nn} \end{bmatrix}$$

του οποίου το άθροισμα των στοιχείων της κάθε γραμμής είναι 1.

Ένας άλλος χρήσιμος τρόπος αναπαράστασης αλυσίδων Markov είναι τα *διαγράμματα κατάστασης μετάβασης*. Έτσι για παράδειγμα η ακόλουθη αλυσίδα Markov,

$$P = \begin{bmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \end{bmatrix}$$

μπορεί επίσης να περιγραφεί από το διάγραμμα,

Συχνά όμως, είναι επιθυμητός ο υπολογισμός της, *μετά από n βήματα πιθανότητας μετάβασης* μιας αλυσίδας Markov στην κατάσταση j, δοθέντος το γεγονότος ότι βρίσκεται στην κατάσταση i,

$$p_{ij}^{(n)} = P(X_n = j / X_0 = i)$$

Ορίζουμε ως,

$$p_j^{(n)} = P(X_n = j) \qquad n= 0, 1, 2, \ldots \; j= 0, 1, \ldots$$

και το διάνυσμα,

$$\underset{\sim}{p}^{(n)} = \left[p_0^{(n)}, p_1^{(n)}, ..., p_k^{(n)}\right]$$

αν υποθέσουμε k διαφορετικές καταστάσεις στο σύστημα.

Τότε, το διάνυσμα

$$\underset{\sim}{p}^{(0)} = \left[p_0^{(0)}, p_1^{(0)}, ..., p_k^{(0)}\right]$$

δίνει τις αρχικές πιθανότητες κατάστασης του συστήματος. Είναι εύκολο να αποδειχθεί ([VAS89]) ότι,

$$\underset{\sim}{p}^{(n)} = \underset{\sim}{p}^{(0)} \cdot P^n$$

δηλαδή, οι μετά από *n* βήματα πιθανότητες μετάβασης μιας αλυσίδας Markov είναι άμεσα εξαρτημένες από την αρχική κατάσταση του συστήματος. Για το λόγο αυτό και επειδή γενικά ο υπολογισμός δυνάμεων του πίνακα μετάβασης καταστάσεων έχει υψηλό υπολογιστικό κόστος, τα προβλήματα εκτίμησης της απόδοσης ενός συστήματος κατά την *αρχική μεταβατική συμπεριφορά* αυτού είναι συνήθως αρκετά πολύπλοκα.

Αντίθετα, από την άλλη μεριά, η ανάλυση της μακροπρόθεσμης συμπεριφοράς μιας αλυσίδας Markov είναι σαφώς απλούστερη. Ο λόγος είναι ότι όσο περισσότερες μεταβάσεις της αλυσίδας σε νέες καταστάσεις έχουν πραγματοποιηθεί, τόσο μικρότερη σημασία έχει η αρχική κατάσταση του συστήματος. Επιπλέον, αν οι οριακές πιθανότητες p_j, που δίνονται από τη σχέση

$$p_j = \lim_{n\to\infty} P(X_n = j / X_0 = i) = \lim_{n\to\infty} p_{ij}^{(n)} \qquad \text{j = 0, 1, . . .}$$

υπάρχουν, τότε αναφερόμαστε σε αυτές, ως την *κατανομή στατιστικής ισορροπίας* (steady-state distribution) μιας αλυσίδας Markov.

Για τη διατύπωση της συνθήκης ύπαρξης κατανομής στατιστικής ισορροπίας είναι απαραίτητοι οι ακόλουθοι ορισμοί:

Αν υπάρχει τουλάχιστον ένα *k* τέτοιο ώστε, η πιθανότητα

$$p_{ij}^{(k)} = P(X_k = j / X_0 = i) \qquad \text{i, j = 0, 1, . . .}$$

είναι μη μηδενική, τότε λέμε ότι η κατάσταση j είναι *προσιτή από την κατάσταση i*. Αυτό βέβαια δε σημαίνει απαραίτητα ότι και η κατάσταση *i* είναι προσιτή από την κατάσταση *j*. Αν μία κατάσταση είναι προσιτή μόνον από τον εαυτό της, τότε αυτή ονομάζεται *κατάσταση απορρόφησης*.

Ένα σύνολο C καταστάσεων λέμε ότι είναι *κλειστό*, αν καμία κατάσταση εκτός του C δεν είναι προσιτή από κατάσταση εντός του C. Αν το C περιλαμβάνει ακριβώς μία κατάσταση, τότε προφανώς αυτή είναι κατάσταση απορρόφησης.

Το σύνολο των καταστάσεων μιας αλυσίδας Markov είναι ένα κλειστό σύνολο και, αν επιπλέον αυτό δεν περιέχει άλλα κλειστά υποσύνολα καταστάσεων, τότε η συγκεκριμένη αλυσίδα Markov λέμε ότι είναι *αδιαχώριστη*.

Αν $p_{ii}^{(k)} = 1$ για κάποια κατάσταση *i*, τότε, αν η αλυσίδα έχει επισκεφθεί την κατάσταση *i* μία φορά, είναι σίγουρο ότι θα επιστρέψει σε αυτήν. Τέτοιες καταστάσεις ονομάζονται *επαναληπτικές*. Καταστάσεις, που δεν είναι επαναληπτικές ονομάζονται *παροδικές* και η πιθανότητα στατιστικής ισορροπίας αυτών είναι 0. Έχει αποδειχθεί ([VAS89]) ότι μία αλυσίδα Markov είναι αδιαχώριστη, αν και μόνο αν, όλες οι καταστάσεις αυτής είναι προσιτές μεταξύ τους. Επίσης, έχει αποδειχθεί ([VAS89]) ότι, αν δύο καταστάσεις είναι προσιτές η μία από την άλλη, τότε αυτές είναι του ιδίου τύπου, δηλαδή, είτε είναι και οι δύο επαναληπτικές, είτε είναι παροδικές. Κατά συνέπεια, αν μία αλυσίδα Markov είναι αδιαχώριστη, τότε όλες οι καταστάσεις της είναι είτε παροδικές, είτε επαναληπτικές.

Ας θεωρήσουμε τώρα μία αδιαχώριστη και επαναληπτική αλυσίδα Markov *X*. Τότε, κάθε κατάσταση της *X* προσεγγίζεται με πιθανότητα 1 απεριόριστο αριθμό φορών. Αν $n_1^j, n_2^j, ...$ συμβολίζουν τις διαδοχικές επισκέψεις της αλυσίδας στην κατάσταση *j*, τότε, με m_j συμβολίζουμε τον αναμενόμενο χρόνο μεταξύ αυτών των επισκέψεων,

$$m_j = E(n_{k+1}^j - n_k^j) \qquad k = 1, 2, \ldots .$$

Μία κατάσταση *j* ονομάζεται *περιοδική* με περίοδο m_j ($m_j > 1$), αν οι διαδοχικές επισκέψεις στην κατάσταση αυτή μπορούν μόνο να συμβούν σε αριθμό μεταβάσεων πολλαπλάσιων του m_j, δηλαδή,

$$P(X_{n+k\cdot m_j} = j / X_n = j) = 1 \qquad \text{για κάποιο } k \geq 1$$

Αν μία αλυσίδα Markov είναι αδιαχώριστη, τότε είτε όλες οι καταστάσεις αυτής είναι περιοδικές με την ίδια περίοδο, είτε όλες είναι μη περιοδικές.

Λαμβάνοντας λοιπόν υπόψη όλα τα παραπάνω, έχει αποδειχθεί ότι:

> **Θεώρημα Α.1** Αν η διαδικασία X είναι μία αδιαχώριστη, μη περιοδική και επαναληπτική αλυσίδα Markov, τότε οι πιθανότητες p_j στατιστικής ισορροπίας υπάρχουν και δίνονται από τη σχέση
>
> $$p_j = \frac{1}{m_j} \qquad \text{j} = 0, 1, \ldots$$

Αν τα διαστήματα μεταξύ των επιστροφών στην κατάσταση *j* είναι απεριόριστα μεγάλα, τότε η *j* ονομάζεται *ασαφώς επαναληπτική* και $p_j = 0$. Οι καταστάσεις, που ο μέσος χρόνος επιστροφής τους στις ίδιες είναι πεπερασμένος, ονομάζονται *θετικά επαναληπτικές* καταστάσεις. Γενικά, έχει αποδειχθεί ([VAS89]) ότι αν μία αλυσίδα Markov είναι αδιαχώριστη με επαναληπτικές καταστάσεις, τότε αυτές είναι είτε θετικά επαναληπτικές είτε ασαφώς επαναληπτικές. Το παρακάτω θεώρημα αποτελεί τη βάση μιας μεγάλης μερίδας αποτελεσμάτων, που αφορούν την ανάλυση της απόδοσης υπολογιστικών συστημάτων.

> **Θεώρημα Α.2** Μία αδιαχώριστη και μη περιοδική αλυσίδα Markov *X*, με πίνακα πιθανοτήτων μετάβασης $P = (p_{ij})$, είναι θετικά επαναληπτική, αν και μόνο αν το σύστημα των εξισώσεων
>
> $$p_j = \sum_i p_i \cdot p_{ij} \qquad j = 0, 1, \ldots$$

και

$$\sum_j p_j = 1$$

έχει λύση. Η λύση αυτή είναι τότε μοναδική και αποτελεί την κατανομή της X σε κατάσταση στατιστικής ισορροπίας.

Οι εξισώσεις της πρώτης μορφής ονομάζονται *εξισώσεις ισορροπίας*, ενώ η τελευταία εξίσωση είναι η *εξίσωση κανονικοποίησης*.

Παρόλα αυτά, είναι προφανές ότι οι μεταβάσεις από μία κατάσταση σε κάποια άλλη, στα υπολογιστικά συστήματα, δε συμβαίνουν σε διακριτές χρονικές στιγμές. Η λύση στο πρόβλημα δεν είναι η μελέτη μιας αλυσίδας Markov, αλλά η μελέτη μιας *διαδικασίας Markov* (Markov process), που είναι ουσιαστικά η αντίστοιχη μιας αλυσίδας Markov διαδικασία, σε συνεχή όμως χρόνο. Όσο για το χώρο καταστάσεων μιας τέτοιας διαδικασίας, αυτός παραμένει διακριτός.

Α.4 Διαδικασίες Markov

Μια διαδικασία Markov λοιπόν λέμε ότι είναι *ομογενής* αν οι πιθανότητες μετάβασης

$$p_{ij}(s) = P(X_{t+s} = j / X_t = i) \qquad i, j = 0, 1, \ldots \qquad s \geq 0$$

είναι ανεξάρτητες της παραμέτρου t. Τότε, οι συναρτήσεις $p_{ij}(s)$ ονομάζονται *συναρτήσεις πιθανότητας μετάβασης* της διαδικασίας Markov και είναι το αντίστοιχο των μετά από n βήματα πιθανοτήτων μετάβασης μιας αλυσίδας Markov.

Ας υποθέσουμε τώρα ότι μια διαδικασία Markov X παραμένει σε κάθε κατάσταση i για μία τυχαία χρονική περίοδο, εκθετικά κατανεμημένη με παράμετρο λ_i. Στο τέλος κάθε τέτοιας περιόδου, η διαδικασία μετακινείται σε μία διαφορετική κατάσταση j με πιθανότητα p_{ij}. Τότε, η διαδικασία Markov X περιγράφεται πλήρως από τα γινόμενα,

$$a_{ij} = \lambda_i \cdot p_{ij} \qquad i, j = 0, 1, \ldots. \qquad i \neq j$$

Επειδή όμως το άθροισμα όλων των πιθανοτήτων μετάβασης p_{ij} ως προς όλα τα j πρέπει να είναι 1, μπορούμε να εκφράσουμε τις παραμέτρους λ_i ως εξής:

$$\lambda_i = \sum_{\substack{j \\ j \neq i}} a_{ij} \qquad i = 0, 1, \ldots .$$

Αν ορίσουμε, $a_{ii} = -\lambda_i$ ($i = 0, 1, \ldots .$), τότε μπορεί να σχηματιστεί ο ακόλουθος πίνακας,

$$A = \begin{bmatrix} a_{00} & a_{01} & \ldots & a_{0n} \\ a_{10} & a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n0} & a_{n1} & \ldots & a_{nn} \end{bmatrix}$$

γνωστός και ως *πίνακας τάσεων*. Στον παραπάνω πίνακα, είναι προφανές ότι το άθροισμα κάθε γραμμής του είναι ίσο με 0.

Αγνοώντας τις χρονικές στιγμές κατά τις οποίες η διαδικασία X μετακινείται από κατάσταση σε κατάσταση και απλά αριθμώντας αυτές τις μεταβάσεις, τότε η αλληλουχία καταστάσεων που προκύπτει $\{X_n, n = 0, 1, \ldots . \}$ είναι μία αλυσίδα Markov. Η αλυσίδα αυτή καλείται

ενσωματωμένη στη διαδικασία X και οι πιθανότητες άμεσης μετάβασης p_{ij} δίνονται από τη σχέση

$$p_{ij} = \frac{a_{ij}}{\lambda_i} = \frac{a_{ij}}{-a_{ii}} \qquad i \neq j = 0,1,.....$$

Μία διαδικασία Markov *X* λέμε ότι είναι αδιαχώριστη, παροδική, ασαφώς επαναληπτική ή θετικά επαναληπτική, αν οι ιδιότητες αυτές χαρακτηρίζουν την αντίστοιχη ενσωματωμένη αλυσίδα Markov. Τέλος, το παρακάτω είναι μία επέκταση του θεωρήματος Α.2.

Θεώρημα Α.3 Μία αδιαχώριστη διαδικασία Markov *X* με πίνακα τάσεων $A=(\alpha_{ij})$, είναι θετικά επαναληπτική, αν και μόνο αν, το σύστημα των εξισώσεων

$$\sum_i p_i \cdot a_{ij} = \sum_i p_i \cdot \lambda_i \cdot p_{ij} = 0 \qquad j = 0, 1, \ldots$$

$$\sum_j p_j = 1$$

έχει λύση. Η λύση αυτή είναι τότε μοναδική και αποτελεί την κατανομή της *X*, σε κατάσταση στατιστικής ισορροπίας.