ΕΝΟΤΗΤΑ 2

ΔΙΕΡΓΑΣΙΕΣ: ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΕΛΕΓΧΟΣ

Περιεχόμενα

1. Ανάγκη για ύπαρξη διεργασιών

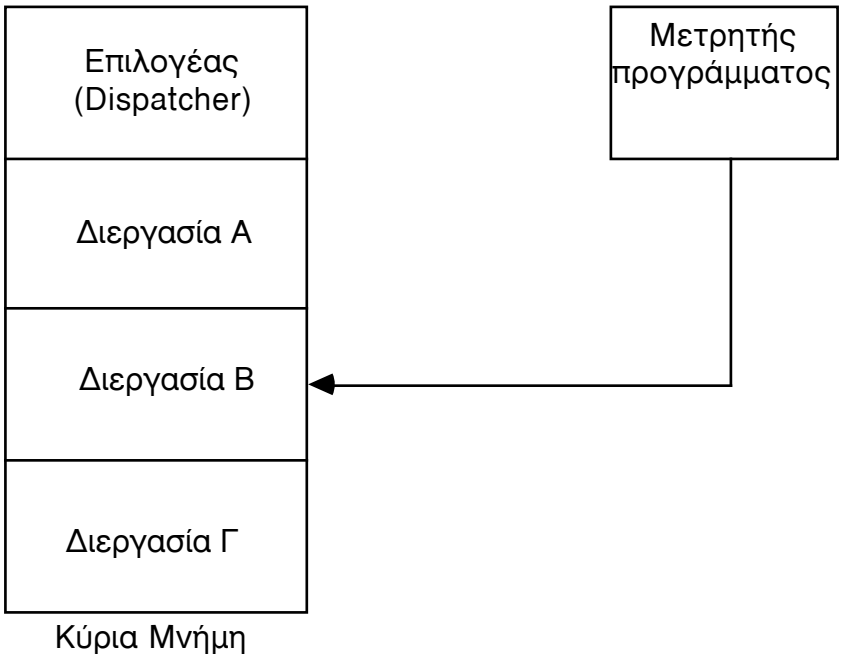
- Ένας σύγχρονος Η/Υ επιτελεί πολλές διαφορετικές εργασίες στο ίδιο χρονικό διάστημα. Η εκτέλεση ενός προγράμματος λ.χ. μπορεί να γίνεται ταυτόχρονα με το τύπωμα κάποιου αρχείου ή το διάβασμα δεδομένων από κάποιο σκληρό δίσκο. Επίσης, σε περιβάλλον πολυπρογραμματισμού, η ΚΜΕ εκτελεί ταυτόχρονα πολλά προγράμματα, αφιερώνοντας μερικά δέκατα ή εκατοστά του χιλιοστοδευτερολέπτου (millisecond) για κάθε ένα από τα προγράμματα αυτά.

- Για τη διαχείρηση πολλών ταυτόχρονων ή παράλληλων δραστηριοτήτων επινοήθηκε το μοντέλο της *διεργασίας* (process) που αποτελεί την πλέον κεντρική ιδέα σε κάποιο μοντέρνο λειτουργικό σύστημα. Ένα λειτουργικό σύστημα χρησιμοποιεί διεργασίες για κάποιες λειτουργίες όπως:
  - κατακερματισμό των προγραμμάτων σε έναν αριθμό διεργασιών οι οποίες εκτελούνται ταυτόχρονα με σκοπό τη μεγιστοποίηση του χρόνου χρήσης της ΚΜΕ και την ελαχιστοποίηση του χρόνου απόκρισης (response time)·
  - επιμερισμό των διαθέσιμων πόρων μεταξύ των εκτελούμενων διεργασιών με βάση κάποιες πολιτικές (policies) όπως προτεραιότητα, κλπ.·
  - υποστήριξη μηχανισμών επικοινωνίας μεταξύ ταυτόχρονα εκτελούμενων διεργασιών αλλά και δημιουργία νέων διεργασιών από τον χρήστη.

- Ανάγκη για *διαχείρηση διεργασιών* (process management), δηλαδή:
  - δημιουργία νέων διεργασιών, τερματισμό εκτέλεσης μίας διεργασίας, αναστολή (suspension) και επανεκκίνηση εκτέλεσης μίας διεργασίας·
  - υποστήριξη μηχανισμών επικοινωνίας και συγχρονισμού μεταξύ ταυτόχρονα εκτελούμενων διεργασιών και αντιμετώπιση *αδιεξόδων* (deadlocks).

2. Καταστάσεις διεργασιών

- Το βασικό μοντέλο των *σειριακών διεργασιών* (sequential processes) ή πιο απλά διεργασιών αναφέρεται σε ένα αριθμό διεργασιών που βρίσκονται στην κύρια μνήμη και εκτελούνται με κάποια σειρά. Υπεύθυνο για να καθορίζει ποια επεξεργασία εκτελείται ανά πάσα στιγμή είναι ένα μέρος του λειτουργικού συστήματος που λέγεται *επιλογέας* (dispatcher).



- Αν μία διεργασία εκτελείται για 6 μόνο εντολές, τότε ένα πιθανό σενάριο είναι το ακόλουθο (—> δηλώνει τέλος χρόνου, => δηλώνει χρήση Ε/Ε):

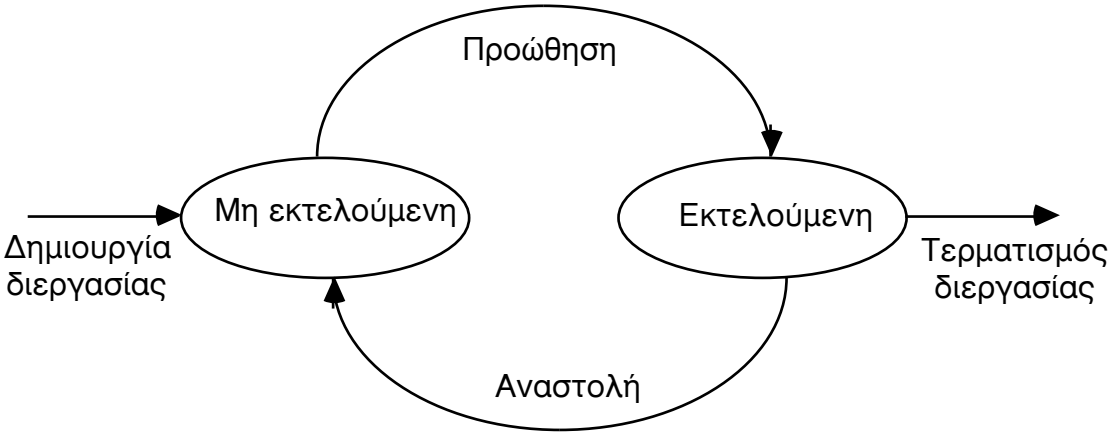
  α+0…α+5 —> δ+0…δ+5, β+0…β+3 => δ+0…δ+5, γ+0…γ+5 —> …

  όπου α+2 λόγου χάριν δηλώνει την εκτέλεση της δεύτερης εντολής και δ είναι ο επιλογέας.

- Επομένως δ+0…δ+5 είναι το κόστος (overhead) εναλλαγής των διεργασιών στην Κεντρική Μονάδα Επεξεργασίας.

2. Καταστάσεις διεργασιών (συνέχεια)

- Έτσι μία διεργασία βρίσκεται ανά πάσα στιγμή σε μία από τουλάχιστον δύο καταστάσεις: εκτελούμενη ή μη εκτελούμενη.



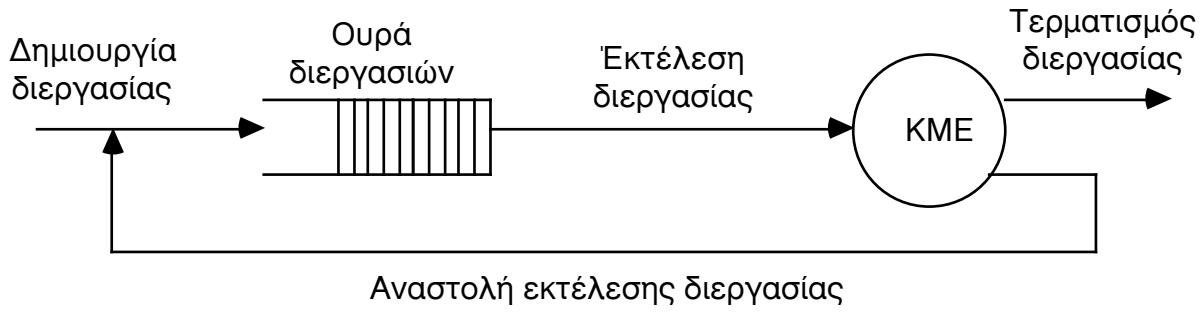
- Για να μπορεί να πραγματοποιηθεί η αναστολή και επανεκκίνηση της εκτέλεσης μίας διεργασίας πρέπει οι διεργασίες να παριστάνονται στο σύστημα με τέτοιο τρόπο ώστε όλες οι πληροφορίες αναφορικά με κάποια διεργασία να είναι στη διάθεση της ΚΜΕ. Οι πληροφορίες αυτές αποτελούν το *περιβάλλον* (environment) της διεργασίας και περιλαμβάνουν λ.χ. τις τιμές των καταχωρητών, το χώρο στη μνήμη που καταλαμβάνει, κλπ.

- Λόγοι δημιουργίας νέας διεργασίας:
  - Αρχή νέας εργασίας (job).
  - Κάποιος χρήστης “μπαίνει” στο σύστημα.
  - Το Λ.Σ. δημιουργεί μία νέα διεργασία για την εξυπηρέτηση κάποιας ανάγκης (π.χ. εκτύπωση).
  - Μία υφιστάμενη διεργασία δημιουργεί νέες διεργασίες (η διεργασία αυτή λέγεται *γονέας* και οι διεργασίες που δημιουργεί *θυγατρικές*).

- Λόγοι τερματισμού μίας διεργασίας:
  - Τερματισμός της εργασίας που τη δημιούργησε.
  - Ο χρήστης “βγαίνει” από το σύστημα.
  - Ο γονέας μίας διεργασίας τερματίζει και το Λ.Σ. αποφασίζει να τερματίσει επίσης και τυχόν θυγατρικές διεργασίες.
  - Κάποιο σοβαρό λάθος προέκυψε κατά την εκτέλεση της διεργασίας.
  - Η διεργασία υπερέβει το μέγιστο χρόνο εκτέλεσής της ή το μέγιστο μέγεθος μνήμης που μπορούσε να δεσμεύσει.

2. Καταστάσεις διεργασιών (συνέχεια)

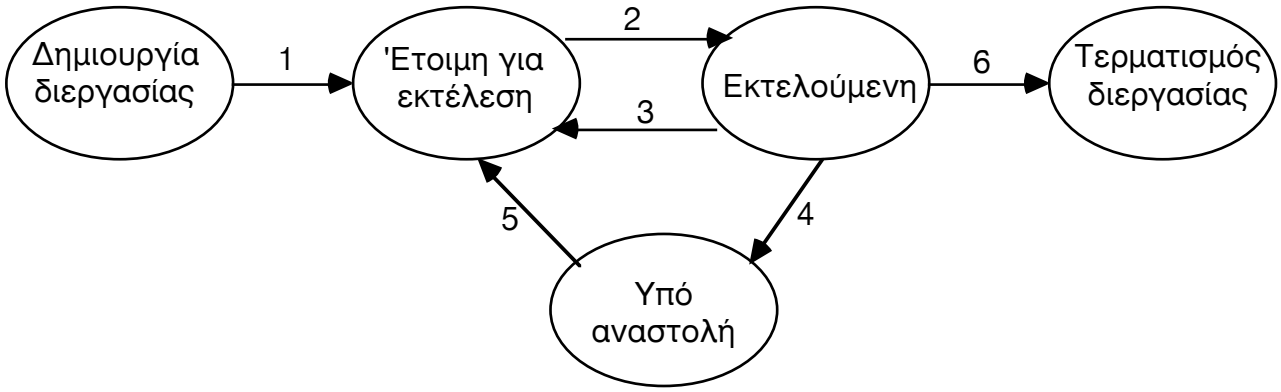
- Στο απλό σύστημα των δύο καταστάσεων, μία πιθανή υλοποίηση του συστήματος διαχείρησης διεργασιών κάνοντας χρήση ουρών FIFO είναι η ακόλουθη (όπου η ουρά υλοποιείται με πίνακα δεικτών σε διεργασίες ή συνδεδεμένη λίστα εγγραφών, μία για κάθε διεργασία):



- Το βασικό μειονέκτημα του απλού αυτού μοντέλου είναι ότι η κατάσταση “μη εκτελούμενη” δεν διαχωρίζει τις διεργασίες που είναι έτοιμες να ξαναρχίσουν εκτέλεση από αυτές που δεν είναι (γιατί λ.χ. περιμένουν την αποπεράτωση μίας εντολής εισόδου/εξόδου). Έτσι ο επιλογέας αντί να διαλέξει απλά την πρώτη διεργασία στην κορυφή της ουράς είναι υποχρεωμένος να ψάξει για την πρώτη *έτοιμη να εκτελεστεί* διεργασία. Σε ένα πραγματικό σενάριο όπου είναι ρεαλιστικό να υπάρχουν εκατοντάδες ή και χιλιάδες διεργασίες μία τέτοια συνεχής σάρωση της λίστας είναι πολύ χρονοβόρα.

2. Καταστάσεις διεργασιών (συνέχεια)

- Είναι επομένως καλύτερα να αντικαταστήσουμε την κατηγορία μη εκτελούμενη με δύο: *έτοιμη για εκτέλεση* (ready) και *υπό αναστολή* (blocked). Επιπλέον δε, ορίζουμε τη δημιουργία και τον τερματισμό μίας διεργασίας σαν ακόμα δύο κατηγορίες και δημιουργείται έτσι ένα σύστημα 5 κατηγοριών. Οι επιτρεπόμενες *μεταβάσεις* (state transitions) από τη μία κατάσταση στην άλλη φαίνονται στο ακόλουθο σχήμα:

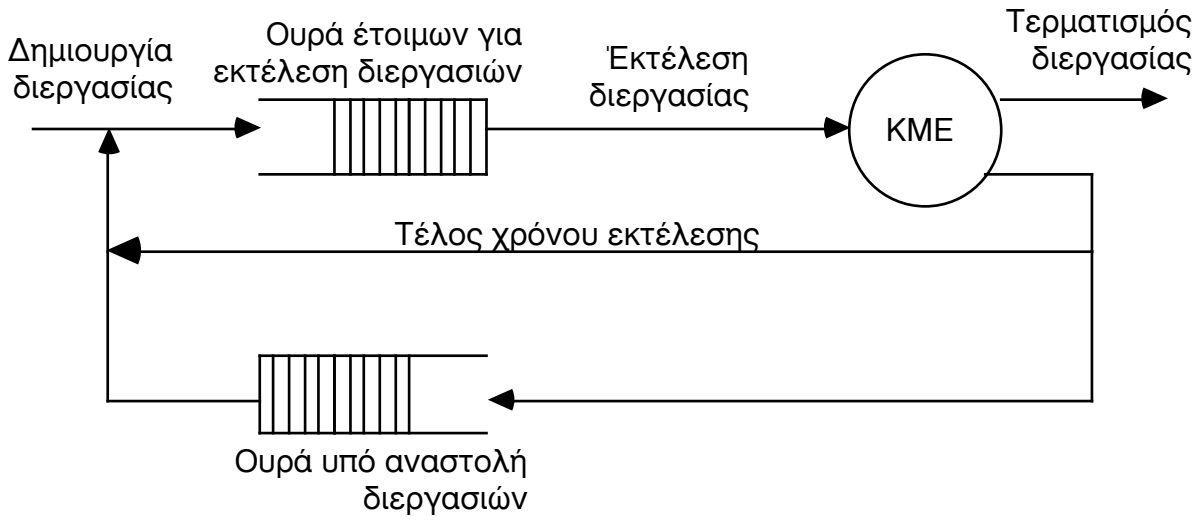


- Οι πιθανές μεταβάσεις από μία κατάσταση σε άλλη έχουν ως εξής:
  1. Μία διεργασία που έχει δημιουργηθεί αλλά δεν είναι υποψήφια για εκτέλεση τώρα συμπεριλαμβάνεται στην ομάδα των υποψήφιων.
  2 & 3. Το Λ.Σ. αποφασίζει με βάση κάποια κριτήρια (τα οποία αποτελούν τις πολιτικές χρονοδρομολόγησης που έχει οιοθετήσει το σύστημα) να διακόψει την εκτέλεση της τρέχουσας διεργασίας που είναι ενεργοποιημένη και να διαλέξει κάποια άλλη από τη λίστα των υποψήφιων διεργασιών. Η νέα αυτή διεργασία ενεργοποιείται ενώ η άλλη επιστρέφει στην ομάδα των υποψήφιων για εκτέλεση περιμένοντας να έρθει πάλι η σειρά της.
  4 & 5. Μία διεργασία που εκτελείται ζητάει από το Λ.Σ. μία υπηρεσία και μέχρις ότου το αίτημά της ικανοποιηθεί βρίσκεται υπό αναστολή, μετά δε γίνεται μέλος της ομάδας των διεργασιών που είναι υποψήφιες για εκτέλεση. Ή, μία διεργασία αναστέλλει τη λειτουργία της περιμένοντας να λάβει χώρα κάποιο εξωτερικό συμβάν.
  6. Η εκτελούμενη διεργασία τερματίζει την εκτέλεσή της αλλά οι δομές της δεν εξαφανίζονται αμέσως από το σύστημα.

2. Καταστάσεις διεργασιών (συνέχεια)

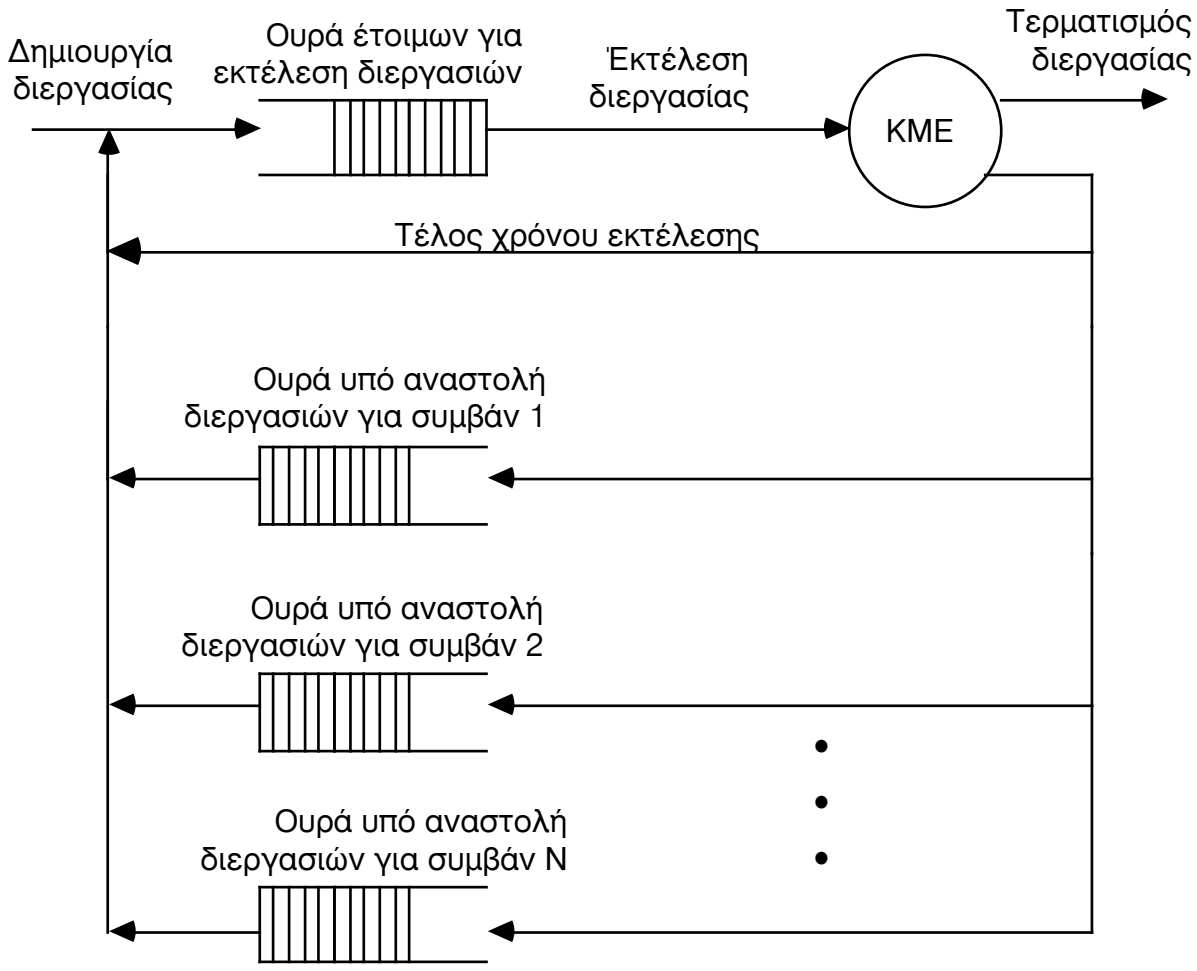
- Αν και δεν φαίνεται στο προηγούμενο σχήμα, υπάρχουν επίσης οι καταστάσεις:
  'Ετοιμη για εκτέλεση —> Τερματισμός, και
  Υπό αναστολή —> Τερματισμός.
  Π.χ. όταν ο τερματισμός μίας διεργασίας προκαλεί και τον τερματισμό όλων των θυγατρικών της.

- Ένας πιθανός τρόπος υλοποίησης του συστήματος διαχείρησης διεργασιών με 5 καταστάσεις είναι η ακόλουθη:



2. Καταστάσεις διεργασιών (συνέχεια)

- Το πρόβλημα είναι ότι κάθε φορά που μία αίτηση προς το Λ.Σ. ικανοποιείται ή κάποιο εξωτερικό συμβάν λαμβάνει χώρα, το Λ.Σ. πρέπει να ψάξει όλη την ουρά για να βρει ποιες από τις διεργασίες μπορούν να ενεργοποιηθούν. Λαμβάνοντας πάλι υπ' όψη ότι σε ένα σύστημα υπάρχουν εκατοντάδες (ή ακόμα και χιλιάδες) διεργασίες, η αναζήτηση αυτή δυνατόν να είναι πολυέξοδη σε χρόνο. Μία παραλλαγή της ανωτέρω υλοποίησης είναι να έχουμε μία ουρά για κάθε διαφορετικό εξωτερικό συμβάν ή αίτημα:

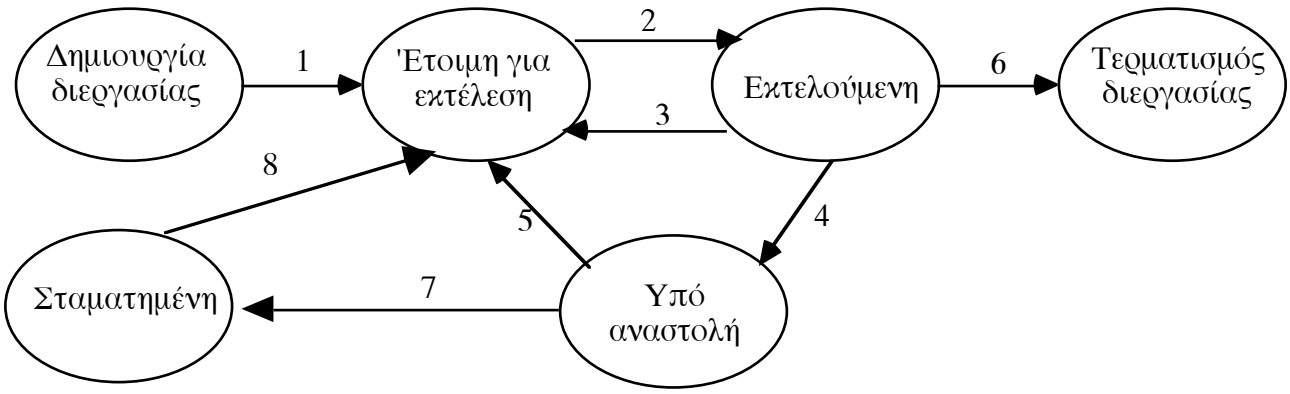


- Κάθε φορά που κάποιο συμβάν λαμβάνει χώρα, οι διεργασίες που βρίσκονται υπό αναστολή στην αντίστοιχη ουρά μεταφέρονται στην ουρά των έτοιμων για εκτέλεση διεργασιών.

2. Καταστάσεις διεργασιών (συνέχεια)

- Ακόμα και έτσι, λόγω της μεγάλης διαφοράς σε ταχύτητα μεταξύ ΚΜΕ και συσκευών Ε/Ε, κάποια στιγμή όλες οι διεργασίες θα είναι υπό αναστολή σε κάποια ουρά. Πιθανές λύσεις:
  - — Αυξάνουμε την κύρια μνήμη και κατ' επέκταση τον αριθμό των διεργασιών που μπορεί να βρίσκονται εκεί. Αλλά: (i) η κύρια μνήμη κοστίζει, (ii) πολύ συχνά, περισσότερη μνήμη συνεπάγεται *μεγαλύτερες* διεργασίες, όχι *περισσότερες* διεργασίες.
  - — Χρησιμοποιούμε το δίσκο για να μεταφέρουμε εκεί ομάδες ολόκληρες διεργασιών υπό αναστολή (σε κάποια ουρά) και να φέρουμε από εκεί στην κύρια μνήμη για εκτέλεση άλλες.

- Επομένως δημιουργείται η ανάγκη να επεκτείνουμε το μοντέλο με ακόμα μία κατάσταση και δύο μεταβάσεις.

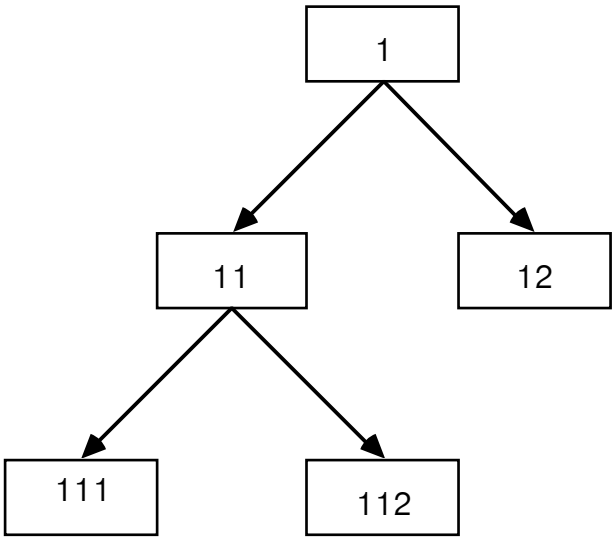


- Αν και η χρήση του δίσκου είναι ουσιαστικά πράξη Ε/Ε, λόγω τού ότι αυτός, σε σχέση με τις άλλες συσκευές Ε/Ε είναι πιο γρήγορος, η μέθοδος αυτή έχει ωφέλιμα αποτελέσματα.

3. Αναπαράσταση των διεργασιών στο λειτουργικό σύστημα

- Το μοντέλο διεργασιών πρέπει να υλοποιηθεί με τέτοιο τρόπο έτσι ώστε να μπορεί το Λ.Σ. να ελέγχει τις διεργασίες και να διαχειρίζεται τους πόρους που κάθε μία από τις διεργασίες χρειάζεται.

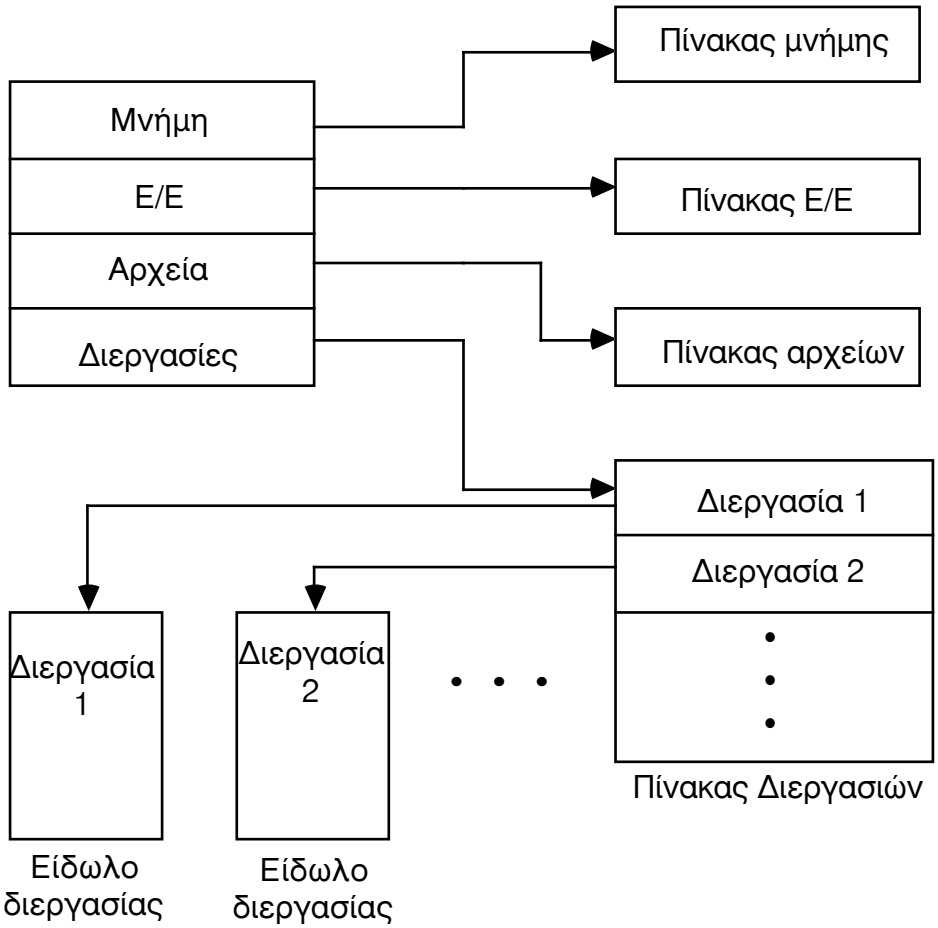
- Κάθε διεργασία έχει τη δυνατότητα να δημιουργήσει άλλες *θυγατρικές διεργασίες* (child processes). Αυτές, ανάλογα με την περίπτωση, μπορούν να ζητήσουν καινούργιους πόρους από το Λ.Σ. ή είναι υποχρεωμένες να περιοριστούν στους πόρους που έχουν δοθεί στη *γονική διεργασία* (parent process), δηλαδή αυτή που τις δημιούργησε.



- Μία διεργασία που δημιουργεί θυγατρικές διεργασίες, ανάλογα με την περίπτωση, μπορεί να συνεχίσει να εκτελείται ταυτόχρονα με τις θυγατρικές διεργασίες ή να παραμείνει υπό αναστολή μέχρις ότου η εκτέλεσή τους τερματιστεί.

3. Αναπαράσταση των διεργασιών στο λειτουργικό σύστημα (συνέχεια)

- Το Λ.Σ. δημιουργεί και διατηρεί *δομές ελέγχου* (control structures), μία για κάθε βασική οντότητα του συστήματος: διεργασίες, αρχεία, μνήμη και συσκευές εισόδου/εξόδου. Συγκεκριμένα, κάθε μία από τις δομές αυτές είναι ένας πίνακας με μία σειρά από εγγραφές. Στην περίπτωση των διεργασιών, κάθε εγγραφή αποτελείται από όλες τις πληροφορίες που σχετίζονται με την εγγραφή αυτή και τις οποίες χρειάζεται το Λ.Σ.



3. Αναπαράσταση των διεργασιών στο λειτουργικό σύστημα (συνέχεια)

- Ο πίνακας μνήμης έχει πληροφορίες όπως:
  - Κατανομή κύριας μνήμης στις διεργασίες.
  - Κατανομή περιφερειακής μνήμης στις διεργασίες.
  - Πληροφορίες για τυχόν περιορισμούς στην προσπέλαση στη μνήμη.
  - Πληροφορίες για διαχείρηση ιδεατής μνήμης.

- Ο πίνακας εισόδου/εξόδου έχει πληροφορίες όπως:
  - Ποιες συσκευές είναι διαθέσιμες.
  - Ποιες συσκευές είναι δεσμευμένες και από ποιες διεργασίες.
  - Αν κάποιες λειτουργίες εισόδου/εξόδου βρίσκονται εν εξελίξει.

- Ο πίνακας αρχείων έχει πληροφορίες όπως:
  - Ὑπαρξη αρχείων.
  - Την περιοχή που βρίσκονται στην περιφερειακή μνήμη.
  - Ἀλλες πληροφορίες για το είδος κάθε αρχείου, κλπ.

- Ο πίνακας διεργασιών έχει τις ακόλουθες κατηγορίες πληροφοριών για κάθε διεργασία:
  - Τα δεδομένα που χρειάζονται για την εκτέλεση του προγράμματος που σχετίζεται με τη διεργασία.
  - Το πρόγραμμα καθ' εαυτό.
  - Τους δείκτες στοιβών του προγράμματος και των δεδομένων.
  - Ἀλλα χαρακτηριστικά αναφορικά με το είδος και το ρόλο της διεργασίας.

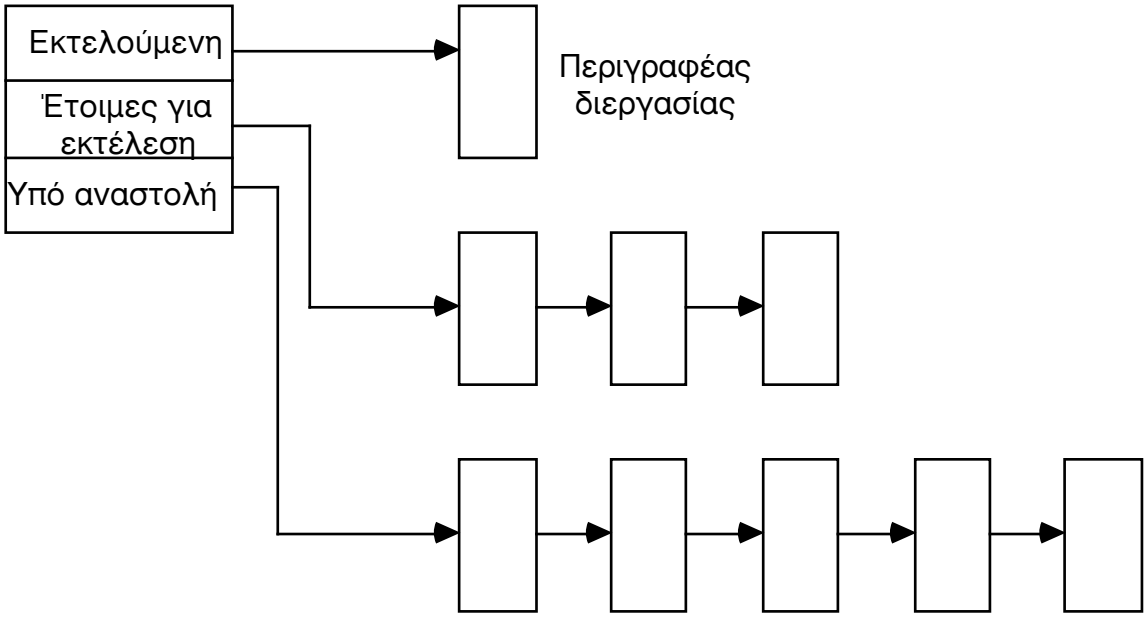
  Ὁλες αυτές οι πληροφορίες αποτελούν το *είδωλο* μίας διεργασίας (process image), ένα μόνο μέρος του οποίου βρίσκεται ανά πάσα στιγμή στην κύρια μνήμη και το υπόλοιπο είναι αποθηκευμένο στο δίσκο.

3. Αναπαράσταση των διεργασιών στο λειτουργικό σύστημα (συνέχεια)

- Τα χαρακτηριστικά μίας διεργασίας αποτελούν το *μπλοκ ελέγχου* της διεργασίας (process control block) ή τον *περιγραφέα* της διεργασίας (process descriptor) και περιέχουν τις ακόλουθες πληροφορίες:
  - Ταυτότητες:
    - Της διεργασίας.
    - Της διεργασίας που δημιούργησε αυτή τη διεργασία.
    - Του ιδιοκτήτη της διεργασίας.
  - Πληροφορίες για την κατάσταση της διεργασίας:
    - Τιμές καταχωρητών.
    - Τιμές δεικτών στοιβών για κάλεσμα συναρτήσεων, κλπ.
  - Πληροφορίες για τον έλεγχο και τη διαχείρηση της διεργασίας:
    - Τρέχουσα κατάσταση και προτεραιότητα της διεργασίας.
    - Για ποιο λόγο βρίσκεται υπό αναστολή (αν βρίσκεται).
    - Πληροφορίες για επικοινωνία με άλλες διεργασίες.
    - Δικαιώματα της διεργασίας στην προσπέλαση στη μνήμη, χρήση συσκευών και υπηρεσιών που προσφέρει το Λ.Σ., κλπ.
    - Δείκτες στους χώρους της μνήμης που χρησιμοποιεί η διεργασία.
    - Κατάσταση ανοικτών αρχείων που τυχόν χρησιμοποιεί η διεργασία.
    - Οι πληροφορίες αυτές είναι χρήσιμες για τον καθορισμό της κατάστασης (mode), αναφορικά με τα δικαιώματα που έχει, στην οποία τρέχει μία διεργασία: σε κατάσταση συστήματος (kernel/system/control mode) μία διεργασία μπορεί να χρησιμοποιεί ειδικές ή προνομοιούχες εντολές (privileged instructions) ενώ σε κατάσταση χρήστη (user mode) μόνο ένα υποσύνολο των διαθέσιμων από το σύστημα εντολών μπορεί να χρησιμοποιηθεί.

3. Αναπαράσταση των διεργασιών στο λειτουργικό σύστημα (συνέχεια)

- Η υλοποίηση των λιστών διεργασιών μπορεί επομένως να είναι στην πραγματικότητα συνδεδεμένες λίστες περιγραφέων διεργασιών:



- Οι περιγραφείς διεργασιών είναι οι πιο σημαντικές κεντρικές δομές δεδομένων και ελέγχου σε ένα Λ.Σ. γιατί περιέχουν όλες τις πληροφορίες που χρειάζεται το Λ.Σ. για κάθε διεργασία. Σχεδόν όλες οι ρουτίνες λειτουργιών του Λ.Σ. διαβάζουν τις πληροφορίες που βρίσκονται αποθηκευμένες στους περιγραφείς διεργασιών και ανάλογα με την περίπτωση τις τροποποιούν.

4. Έλεγχος διεργασιών

Ο *πυρήνας* (kernel) του Λ.Σ. είναι υπεύθυνος για τη δημιουργία, αλλαγή κατάστασης, χρονοδρομολόγηση και τερματισμό των διεργασιών.

- *Δημιουργία* μίας διεργασίας:
  - Ο πίνακας διεργασιών επεκτείνεται με μία επιπλέον εγγραφή για τη νέα διεργασία στην οποία δίνεται σαν *προσδιοριστής* (process identifier) ένας μοναδικός αριθμός.
  - Παρέχεται μνήμη για τις ανάγκες της διεργασίας (αποθήκευση κώδικα και δεδομένων, στοίβες, κλπ.).
  - Δημιουργία του μπλοκ ελέγχου της διεργασίας με όλες τις σχετικές πληροφορίες μερικές από τις οποίες έχουν εκ των προτέρων μία προκαθορισμένη αρχική τιμή (π.χ. η πρώτη κατάσταση μίας καινούργιας διεργασίας είναι συνήθως “έτοιμη για εκτέλεση”).
  - Δημιουργούνται οι κατάλληλες διασυνδέσεις της διεργασίας με τις σχετικές δομές ελέγχου με τη χρήση δεικτών (π.χ. η εισαγωγή της διεργασίας στη λίστα για διεργασίες που είναι έτοιμες για εκτέλεση).
  - Δημιουργία άλλων σχετικών δομών (π.χ. για λογιστικούς ή στατιστικούς λόγους).

- *Εναλλαγή* διεργασιών (process switching):
  - Λαμβάνει χώρα όταν υπάρξει κάποια διακοπή (interrupt), λάθος (π.χ. διαίρεση με το 0), ή κλίση από τον επιτηρητή (supervisor call).
  - Πολλές φορές η εμφάνιση μίας διακοπής δεν είναι απαραίτητο να οδηγήσει και σε εναλλαγή διεργασιών και μετά το πέρας της διακοπής μπορεί να συνεχίσει η εκτέλεση της τρέχουσας διεργασίας. Αυτή η απλούστερη περίπτωση λέγεται *μεταγωγή περιβάλλοντος* (context switching) και διαφέρει από την πιο πολύπλοκη εναλλαγή διεργασιών.

4. Έλεγχος Διεργασιών (συνέχεια)

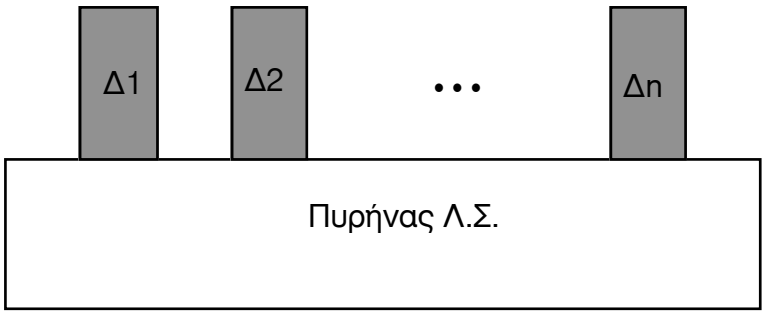
- Στην περίπτωση της μεταγωγής περιβάλλοντος, και πριν την εκτέλεση της ρουτίνας που σχετίζεται με τη διακοπή, αποθηκεύονται εκείνες οι πληροφορίες που θα μπορούσαν τυχόν να αλλοιωθούν κατά την εκτέλεση της ρουτίνας· αυτές περιλαμβάνουν το μέρος εκείνο του μπλοκ ελέγχου διεργασίας (process control block) που σχετίζεται με την κατάσταση της ΚΜΕ: μετρητής προγράμματος, τιμές άλλων καταχωρητών και περιεχόμενα των στοιβών.

- Στην περίπτωση της εναλλαγής διεργασίας οι διαδικασίες που πρέπει να ακολουθηθούν είναι πολύ περισσότερες:
  - Αποθήκευση των πληροφοριών που σχετίζονται με την κατάσταση της ΚΜΕ.
  - Ενημέρωση του μπλοκ ελέγχου της διεργασίας που σταμάτησε να εκτελείται για τη νέα της κατάσταση, τους λόγους που σταμάτησε η εκτέλεσή της και τυχόν άλλες πληροφορίες.
  - Σύνδεση του μπλοκ ελέγχου της διεργασίας με την κατάλληλη λίστα διεργασιών (των έτοιμων για εκτέλεση, υπό αναστολή, κλπ.).
  - Με βάση κάποια κριτήρια και πολιτική μία άλλη διεργασία επιλέγεται για εκτέλεση.
  - Ενημέρωση του μπλοκ ελέγχου της διεργασίας αυτής (π.χ. αλλαγή της κατάστασής της σε εκτελούμενη).
  - Ενημέρωση των δομών δεδομένων που σχετίζονται με τη διαχείριση της κύριας μνήμης.
  - Αλλαγή της κατάστασης της ΚΜΕ σε αυτή που ήταν την τελευταία φορά που εκτελέστηκε η τρέχουσα διεργασία.

5. Εκτέλεση του λειτουργικού συστήματος

Αφού το Λ.Σ. είναι το ίδιο ένα ακόμα πρόγραμμα, εκτελείται από την ΚΜΕ όπως ένα οποιοδήποτε άλλο πρόγραμμα. Τίθεται επομένως το ερώτημα αν το σύστημα βλέπει το Λ.Σ. σαν μία διεργασία, καθώς επίσης και με ποιό τρόπο το Λ.Σ. αποκτά και απελευθερώνει τον έλεγχο της ΚΜΕ. Υπάρχουν οι ακόλουθες εναλλακτικές απόψεις:

- Το Λ.Σ. είναι μία ξεχωριστή οντότητα και δεν αντιμετωπίζεται σαν διεργασία.
  - Παραδοσιακή άποψη που εφαρμόστηκε στα παλαιότερα Λ.Σ.
  - Το Λ.Σ. έχει το δικό του χώρο μνήμης, δομές, στοίβες, κλπ.
  - Εκτελείται σε προνομιούχα κατάσταση (privileged mode), δηλαδή έχει περισσότερα δικαιώματα από μία συνηθισμένη διεργασία.
  - Όταν για οποιονδήποτε λόγο μία διεργασία αναστείλει την εκτέλεσή της, ο έλεγχος δίνεται στο Λ.Σ. το οποίο αφού επιτελέσει τις λειτουργίες που απαιτούν οι λόγοι αναστολής εκτέλεσης της διεργασίας, αποφασίζει επίσης κατά πόσο η διεργασία θα συνεχίσει την εκτέλεσή της ή η ΚΜΕ θα δοθεί σε κάποια άλλη διεργασία.



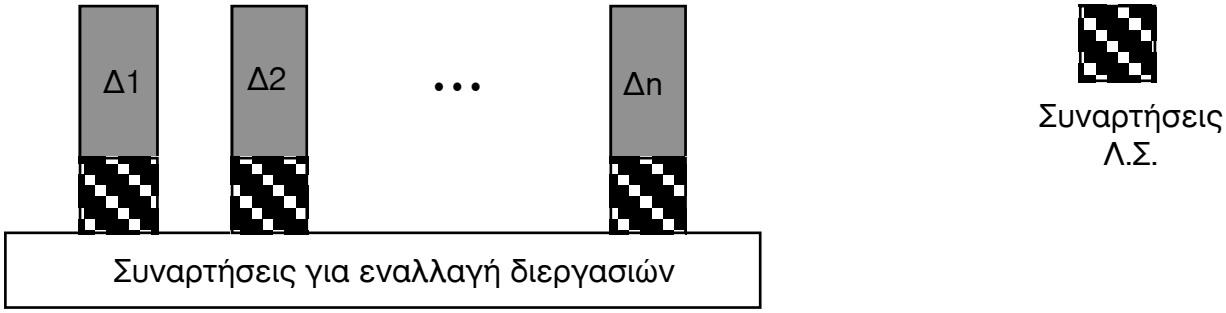
- Οι περισσότερες από τις λειτουργίες του Λ.Σ. αποτελούν μέρος των εκτελούμενων διεργασιών.
  - Χρησιμοποιείται κυρίως σε Λ.Σ. μικρών συστημάτων Η/Υ (π.χ. PCs).
  - Το Λ.Σ. είναι απλά μία ομάδα από ρουτίνες που μπορούν να καλεσθούν από οποιαδήποτε διεργασία.
  - Επιτρέπει και εκμεταλλεύεται την έννοια της μεταγωγής περιβάλλοντος. Μία διεργασία, ανάλογα με την περίσταση, εκτελείται σε κατάσταση χρήστη (user mode) ή σε κατάσταση

5. Εκτέλεση του λειτουργικού συστήματος (συνέχεια)

πυρήνα (kernel mode), όπου στη τελευταία αυτή δικαιούται να εκτελέσει ειδικές εντολές.

— Όταν μία διεργασία πρέπει να αναστείλλει την εκτέλεσή της, τότε αποθηκεύονται οι πληροφορίες που σχετίζονται με την τρέχουσα κατάστασή της και ο έλεγχος περνάει σε μία κατάλληλη ρουτίνα του Λ.Σ. Αυτό όμως γίνεται με μεταγωγή περιβάλλοντος, δηλαδή η ρουτίνα αυτή τρέχει μέσα στο χώρο μνήμης της διεργασίας που υπέστη αναστολή. Όταν ολοκληρωθεί η εκτέλεση της ρουτίνας, ο έλεγχος επανέρχεται στην υπό αναστολή διεργασία. Μόνο όταν χρειάζεται να γίνει εναλλαγή διεργασίας έχουμε πλήρη απομάκρυνση της διεργασίας από την ΚΜΕ και την εισγωγή σε αυτήν άλλης διεργασίας. Με άλλα λόγια σε σύγκριση με την προηγούμενη προσέγγιση γλυτώνουμε το κόστος της εναλλαγής διεργασίας όταν ο έλεγχος πρέπει να επανέλθει στη διεργασία που είχε ανασταλλεί.

— Το γεγονός ότι η ευθύνη εναλλαγής των διεργασιών φαίνεται να ανήκει στις διεργασίες καθεαυτές και όχι σε κάποια άλλη ουδέτερη οντότητα, δεν δημιουργεί προβλήματα δικαιοσύνης (fairness) διότι οι ρουτίνες του Λ.Σ. εκτελούνται σε κατάσταση πυρήνα όπου ο χρήστης (και ιδιοκτήτης της διεργασίας) δεν έχει δικαίωμα πρόσβασης στον κώδικα των ρουτινών του Λ.Σ. (παρόλο που αυτές εκτελούνται στο περιβάλλον της διεργασίας του) και επομένως δεν μπορεί να τον τροποποιήσει για το συμφέρον του.



5. Εκτέλεση του λειτουργικού συστήματος (συνέχεια)

- Το Λ.Σ. αποτελεί μία ομάδα από διεργασίες.
  - — Οι περισσότερες από τις σημαντικές λειτουργίες του Λ.Σ. εκτελούνται σαν ξεχωριστές διεργασίες.
  - — Διευκολύνει την ανάπτυξη δομημένης οργάνωσης στο Λ.Σ.
  - — Το Λ.Σ. γίνεται πιο ευέλικτο και μπορεί να χρησιμοποιήσει τα ίδια κριτήρια οργάνωσης των διεργασιών των χρηστών στις δικές του διεργασίες (π.χ. μπορεί να εκτελεί τις διεργασίες του με διαφορετικούς βαθμούς προτεραιότητας ανάλογα με τη σημασία της κάθε ρουτίνας που έχει καλεσθεί).
  - — Ένα Λ.Σ. βασισμένο σε αυτή την προσέγγιση μπορεί πιο εύκολα να επεκταθεί για παράλληλα και κατανεμημένα περιβάλλοντα.

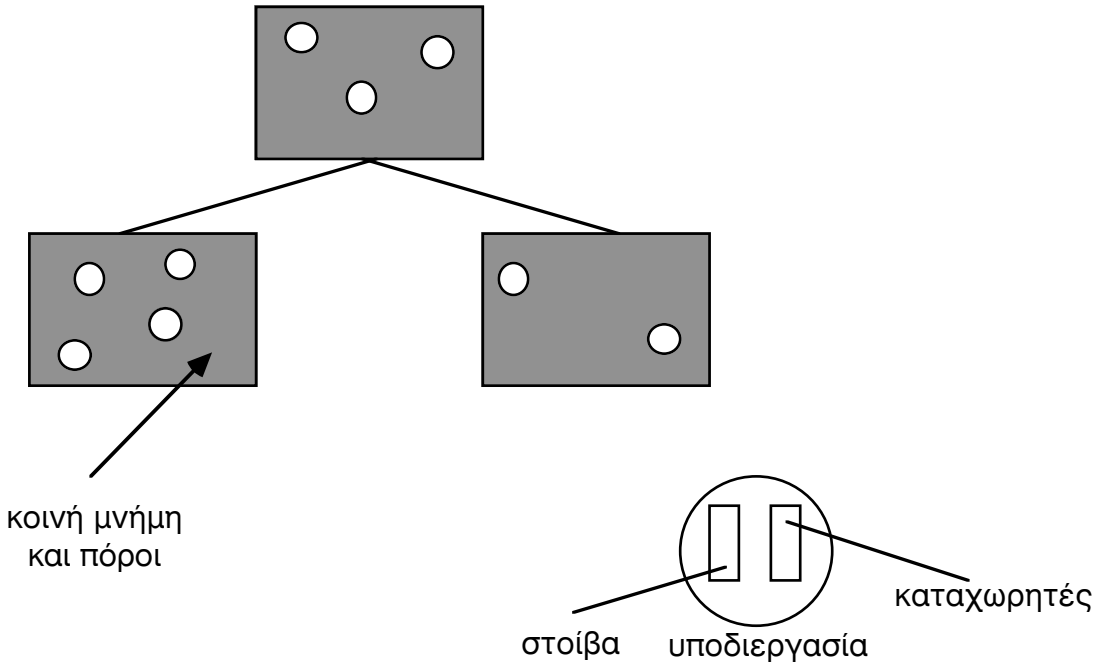
6. Η έννοια του μικροπυρήνα

- Σε πολλά μοντέρνα λειτουργικά συστήματα έχει αρχίσει να αναπτύσσεται ή έννοια του *μικροπυρήνα* (microkernel). Η ιδέα εδώ είναι ότι ο πυρήνας του Λ.Σ. αποτελείται μόνο από την άκρως απαραίτητη λειτουργικότητα που χρειάζεται ένα Λ.Σ. Όλα τα υπόλοιπα υλοποιούνται με βάση τον πυρήνα αυτόν, αλλά εκτός αυτού ως διαφορετικά προγράμματα.

- Αυτό επιτρέπει την πιο εύκολη υποστήριξη ενός Λ.Σ. από πολλές πλατφόρμες (portability) αφού μόνο ο (σχετικά μικρός) πυρήνας χρειάζεται να τροποποιηθεί ανάλογα με τις απαιτήσεις της κάθε αρχιτεκτονικής. Τα Λ.Σ. Mach και Windows NT βασίζονται σε αυτή τη φιλοσοφία.

7. Διάσπαση των διεργασιών σε υποδιεργασίες

- Μέχρι τώρα η διεργασία παρουσιάστηκε σαν η μικρότερη οντότητα που μπορεί να δεσμεύσει μνήμη και να εκτελεσθεί. Σε πολλά Λ.Σ. όμως (π.χ. OS/2, Mach) η διεργασία μπορεί να αποτελείται από *υποδιεργασίες* (threads) οι οποίες μοιράζονται το χώρο μνήμης της διεργασίας στην οποία ανήκουν αλλά εκτελούνται σαν ξεχωριστές οντότητες. Σε αυτή την περίπτωση η μικρότερη οντότητα που εκτελείται σε ένα σύστημα Η/Υ είναι η υποδιεργασία.



- Κάθε υποδιεργασία έχει τη δικιά της κατάσταση, στοίβα με τιμές του περιβάλλοντός της, μνήμη για τοπικές μεταβλητές και δικαίωμα προσπέλασης και χρήσης στη μνήμη και πόρους αντίστοιχα της διεργασίας στην οποία ανήκει.

- Η ύπαρξη υποδιεργασιών σε ένα Λ.Σ. επηρεάζει θετικά την απόδοση του συστήματος γιατί είναι πιο γρήγορη η δημιουργία, τερματισμός, αναστολή και επανεκκίνηση εκτέλεσης μίας υποδιεργασίας από την αντίστοιχη λειτουργία για διεργασία. Επίσης, η επικοινωνία μεταξύ υποδιεργασιών είναι πιο απλή από την αντίστοιχη μεταξύ διεργασιών γιατί δεν χρειάζεται να καλεσθούν οι σχετικές συναρτήσεις του πυρήνα.

7. Διάσπαση των διεργασιών σε υποδιεργασίες (συνέχεια)

- Σε γενικές γραμμές, όταν μία λειτουργία μπορεί να υλοποιηθεί σαν μία ομάδα από άμεσα συσχετιζόμενες μονάδες, είναι πολλές φορές προτιμότερο να δημιουργήσουμε μία διεργασία για τη λειτουργία αυτή, η οποία να αποτελείται από μία ομάδα από υποδιεργασίες (μία για κάθε μονάδα), αντί για μία ομάδα διεργασιών.

- Κλασσικά παραδείγματα είναι σε μία εφαρμογή να έχουμε μία υποδιεργασία που να διαβάζει δεδομένα από τον χρήστη ενώ άλλες να εκτελούν τα ήδη υπάρχοντα, περιοδική αυτόματη αποθήκευση των δεδομένων μίας εφαρμογής την ώρα που αυτή εκτελείται, ταυτόχρονη εκτέλεση εντολών εισόδου/εξόδου, κλπ.

- Ανάλογα με τον τρόπο που υποστηρίζονται από το Λ.Σ., οι υποδιεργασίες μπορεί να υλοποιηθούν σε επίπεδο χρήστη (user-level) ή Λ.Σ. (kernel-level).
  - Στο επίπεδο χρήστη, η υλοποίησή τους γίνεται μέσω μιας βιβλιοθήκης που παρέχει ρουτίνες για τη διαχείρησή τους (δημιουργία, τερματισμό, κλπ.) και οι οποίες καλούνται μέσα από το πρόγραμμα της εφαρμογής. Από το σημείο αναφοράς του Λ.Σ. δεν υπάρχει καμμία διαφορά από ένα περιβάλλον που δεν υποστηρίζει υποδιεργασίες· το Λ.Σ. συνεχίζει να βλέπει μόνο μία διεργασία.
  - Στο επίπεδο του Λ.Σ., η υλοποίησή τους γίνεται από το ίδιο το σύστημα, το οποίο είναι υπεύθυνο για τη διαχείρησή τους· ο κώδικας της εφαρμογής απλά χρησιμοποιεί ένα API για τη διασύνδεσή του με το κομμάτι του Λ.Σ. που διαχειρίζεται τις υποδιεργασίες. Τα Λ.Σ. Windows2000, Linux και OS/2 οϊοθετούν αυτή την προσέγγιση.

- Τα πλεονεκτήματα της υλοποίησης των υποδιεργασιών στο επίπεδο του χρήστη είναι:
  - Η μεταφορά της εκτέλεσης του προγράμματος από μία υποδιεργασία σε άλλη δε χρειάζεται εναλλαγή διεργασιών αλλά μόνο μεταγωγή περιβάλλοντος (μικρότερο κόστος).
  - Η χρονοδρομολόγηση των υποδιεργασιών μπορεί να καθορισθεί από κάθε εφαρμογή ξεχωριστά και όχι να επιβάλλεται από το Λ.Σ. (ευελιξία).
  - Μπορεί να χρησιμοποιηθεί οποιοδήποτε Λ.Σ. (ανεξαρτησία).

7. Διάσπαση των διεργασιών σε υποδιεργασίες (συνέχεια)

- Τα μειονεκτήματα της υλοποίησης των υποδιεργασιών στο επίπεδο του χρήστη είναι:
  - Δεδομένου ότι πολλές ρουτίνες του Λ.Σ. (system calls) όταν καλεσθούν από μία εφαρμογή οδηγούν στην αναστολή εκτέλεσής της, αν μία υποδιεργασία καλέσει μια τέτοια ρουτίνα θα ανασταλεί η εκτέλεση συνολικά της διεργασίας στην οποία ανήκει η υποδιεργασία και όχι μόνο της υποδιεργασίας που έκανε το κάλεσμα.
  - Στην περίπτωση που ο Η/Υ έχει περισσότερους από έναν επεξεργαστές, μία διεργασία δεν μπορεί να εκμεταλλευτεί περισσότερους από έναν, έστω και αν αποτελείται από πολλές υποδιεργασίες.

- Στην περίπτωση που οι υποδιεργασίες υποστηρίζονται σε επίπεδο Λ.Σ., η ανωτέρω κατάσταση αντιστρέφεται: το Λ.Σ. έχει τον έλεγχο της διαχείρησής τους· έτσι μπορεί να τις εκτελέσει σε διαφορετικούς επεξεργαστές ή αν κάποια πρέπει να αναστείλει την εκτέλεσή της, μπορεί να αρχίσει εκτέλεση κάποια άλλη υποδιεργασία της ίδιας διεργασίας. Όμως το Λ.Σ. πρέπει να υποστηρίζει υποδιεργασίες και το κόστος μεταφοράς του ελέγχου από το επίπεδο της εφαρμογής σε αυτό του Λ.Σ. είναι σημαντικό.

- Η σχέση μεταξύ διεργασιών και υποδιεργασιών μπορεί να είναι:
  - 1:1, όπου ουσιαστικά δεν υποστηρίζεται η έννοια της υποδιεργασίας (τυπικό Λ.Σ. εδώ είναι το Unix Σύστημα V).
  - 1:M, όπως περιγράφηκε ανωτέρω, όπου κάθε διεργασία καθορίζει τα όρια μνήμης και πόρων των υποδιεργασιών M από τις οποίες αποτελείται (τυπικά Λ.Σ. είναι τα OS/2, Mach και MVS).
  - M:1, όπου κατά τη διάρκεια της ζωής της, μία υποδιεργασία μπορεί να μεταναστεύσει από τη διεργασία στην οποία αρχικά δημιουργήθηκε σε άλλη διεργασία. Σε αυτή την περίπτωση, μεταφέρει μαζί της και κάποιες αναγκαίες πληροφορίες. Η ιδέα αυτή εφαρμόζεται κυρίως σε κατανεμημένα Λ.Σ. (όπως τα Clouds και Emerald) όπου είναι χρήσιμη η έννοια μίας υποδιεργασίας να εκτελείται σε διαφορετικούς χώρους μνήμης και να μεταναστεύει από μία μηχανή σε άλλη.
  - M:M, όπου γίνεται συνδυασμός των ανωτέρω δύο περιπτώσεων (τυπικό παράδειγμα εδώ είναι το πειραματικό Λ.Σ. Trix).

8. Αναπαράσταση και έλεγχος διεργασιών σε ορισμένα Λ.Σ.

Εδώ αναφέρουμε μερικά μόνο από τα κύρια χαρακτηριστικά μερικών Λ.Σ. αναφορικά με τις έννοιες που καλύφθηκαν σε αυτή την ενότητα.

- Στο Unix μία διεργασία δημιουργείται με την εντολή `fork`. Η εντολή

  ```
  pid=fork();
  ```

  δημιουργεί μία θυγατρική διεργασία που είναι ακριβές αντίγραφο της γονικής διεργασίας. Οι δύο διεργασίες εκτελούνται ταυτόχρονα. Για να ξεχωρίζει η γονική από τη θυγατρική διεργασία, η πρώτη έχει σαν προσδιοριστή μία θετική τιμή ενώ η δεύτερη το 0. Έτσι σε ένα πρόγραμμα που κάνει χρήση διεργασιών υπάρχουν εντολές του τύπου

  ```
  if (pid>0)   δηλαδή, είμαι η γονική διεργασία
    …          κώδικας για τη γονική διεργασία
  else         δηλαδή, είμαι θυγατρική διεργασία
    …          κώδικας για τη θυγατρική διεργασία
  ```

- Άλλες σημαντικές εντολές είναι οι ακόλουθες (όπου στη μεταβλητή `s` εκχωρείται η τιμή 0 αν η κλήση της συνάρτησης ήταν επιτυχής και η τιμή -1 σε περίπτωση σφάλματος):
  - `s=waitpid(pid, …)` αναμονή μέχρι τον τερματισμό της θυγατρικής διεργασίας `pid`.
  - `s=execve(name, …)` αντικατάσταση της τρέχουσας διεργασίας από το πρόγραμμα `name`.
  - `s=kill(pid, …)` αποστολή σήματος τερματισμού της διεργασίας `pid`.
  - `exit(status)` τερματισμός εκτέλεσης της διεργασίας και επιστροφή κωδικού εξόδου.

- Μία διεργασία στο Unix μπορεί να είναι σε μία από 9 καταστάσεις. Υπάρχουν 2 καταστάσεις για την περίπτωση μίας διεργασίας που εκτελείται, ανάλογα με τον αν εκτελείται σαν διεργασία πυρήνα (kernel mode) ή χρήστη (user mode). Επίσης διαχωρίζεται η περίπτωση διεργασιών υπό αναστολή που όμως βρίσκονται στην κύρια μνήμη από αυτές που έχουν μεταφερθεί στην περιφερειακή μνήμη. Τέλος, στο Unix υπάρχει 1:1 αντιστοιχία μεταξύ διεργασιών και υποδιεργασιών.

8. Αναπαράσταση και έλεγχος διεργασιών σε ορισμένα Λ.Σ. (συνέχεια)

- Ένα από τα χαρακτηριστικά του OS/2 είναι ότι επιπλέον της έννοιας της υποδιεργασίας, υποστηρίζεται επίσης η *περίοδος* (session) επαφής χρήστη-μηχανής που ορίζεται σαν ένα σύνολο από διεργασίες. Αυτό επιτρέπει την ύπαρξη και χρήση πολλών παραθύρων ταυτόχρονα και τη διοχέτευση της ροής των δεδομένων Ε/Ε από και προς το κατάλληλο παράθυρο.

- Στο Λ.Σ. MVS (Multiple Virtual Storage) υπάρχουν μόνο 3 καταστάσεις διεργασιών: έτοιμη (για εκτέλεση), εκτελούμενη και υπό αναστολή. Σε κάθε διεργασία παρέχεται ένα μέρος της διαθέσιμης μνήμης και οποιαδήποτε άλλη θυγατρική διεργασία περιορίζεται στη χρήση μόνο αυτής της μνήμης.