

**Κεφάλαιο Πέντε**  
-  
**Τα Όρια του Προγραμματισμού**

## 5.1 Διαφορετικές Μεθόδους Προγραμματισμού για Διαφορετικούς Προγραμματιστές

- Πρότυπες δραστηριότητες που δεν συνταυτίζονται με τον καθ' αυτό προγραμματισμό αλλά είναι σχετικές.
  - Σύνταξη απαιτήσεων.
    - Εισαγωγή των οντοτήτων μίας εφαρμογής.
    - Περιοδικές σχέσεις και μεταβολές.
  - Κατασκευή οπτικών εφαρμογών και αρμολόγηση συστατικών.
    - Εισδοχή και λειτουργία προκατασκευασμένων συστατικών, βάση σταθερών εξαρτήσεων.
    - Εστιασμός στη λειτουργικότητα. Αυτό είναι καλό για μικρό αριθμό συστατικών. Για μεγαλύτερους αριθμούς χρειάζεται η προσθήκη οπτικών ιδιοτήτων για βελτίωση της αντίληψης του συστήματος.
  - Γραφή.
    - Στις οπτικές εφαρμογές τα σύμβολα μπορούν να χρησιμοποιούνται σαν λέξεις.
    - Εισαγωγή συμπεριφοράς αλλά όχι νέων θέσεων. Στοχεύει στο να δέσει τα υπάρχοντα συστατικά (σε ψηλό επίπεδο). Αυτό είναι πιθανό να καταλήξει σε 'συστατικά' γραφής.
- Για κάθε μία από τις πρότυπες δραστηριότητες υπάρχει και το κατάλληλο πρότυπο προγραμματισμού.

## 5.2 Προγραμματισμός για ένα Σύστημα

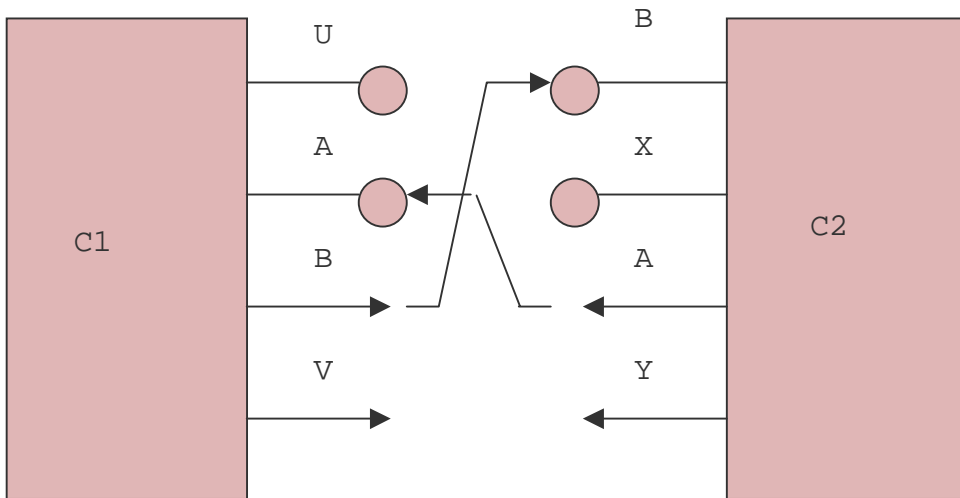
- Οι κατάλληλες θεωρητικές έννοιες για υποστήριξη του προγραμματισμού εξαρτώνται από το πρότυπο προγραμματισμού.
- Στο παραδοσιακό πρότυπο υπάρχει σύνθεση από βιβλιοθήκες εννοιών επεξεργασίας, συν η αυστηρή και άμεση αξιολόγηση (πρότυπο καλέσματος επεξεργασίας). Επίσης το ίδιο συμβαίνει με τον αντικειμενοστρεφή προγραμματισμό με το κάλεσμα μεθόδων. Παρομοίως και σε ψηλότερες βαθμίδες προγραμματισμού όπου δεν υπάρχουν στατικές αναφορές σε συγκεκριμένες επεξεργασίες ή μεθόδους. Ο καλών και ο καλούμενος δεν ελέγχουν το σύνδεσμό τους και οι διασυνδέσεις περιγράφουν και τους δύο.
- Τα καινούργια πρότυπα κοιτούν για συνδέσμους παρά για κλήσεις. Οι σύνδεσμοι είναι συμμετρικοί σε αντίθεση με τις ασύμμετρες κλήσεις.
  - Στις διασυνδέσεις εισαγωγής υπάρχουν κλήσεις σημείων.
  - Οι διασυνδέσεις εξαγωγής δίδουν τις δυναμικές διεργασίες (πιθανά συμβάντα).

## 5.3 Συνδεμοστρεφής Προγραμματισμός

- Στις κλήσεις του σπονδυλωτού προγραμματισμού υπάρχει αλληλοδιαδοχή συστατικών. Επίσης μπορεί ένα συστατικό να κληθεί από άλλο που έχει παρατηρήσει ένα συγκεκριμένο συμβάν (event). Σε αυτή τη μεθοδολογία με την εξάρτηση παρατηρητών από συμβάντα οι πληροφορίες ωθούνται προς τα έξω αμέσως μετά τη λήψη τους (πρότυπο ωθήσεως). Στόν παραδοσιακό προγραμματισμό η κάθε επεξεργασία ξέρει πια άλλη καλεί και υπάρχει σύνδεσμος ή παράκαμψη (indirection) μεταξύ καλούσης και καλούμενης (πρότυπο έλξεως).
- Στο πρότυπο ωθήσεως, οι παρακάμψεις είναι αναπόφευκτες λόγω άγνοιας της πηγής του συμβάντος για τους ενδιαφερόμενους παραλήπτες. Και στις δύο περιπτώσεις παρακάμψης τα πρότυπα τα χειριζόμαστε με τον ίδιο τρόπο. Όταν οι παρακάμψεις (με ρύθμιση κατά την εκτέλεση μίας διεργασίας) αντικαθιστούν τις εξαρτήσεις κλήσεων, αυτό καλείται *συνδεμοστρέφεια* (connection-oriented programming). Αυτοί οι μηχανισμοί συνδέσμων μπορούν να αποτελέσουν ένα ενδιάμεσο συστημα επικοινωνιών, με βάση τη διανομή μηνυμάτων. Αυτό καλείται *μηνυματοστρεφής ενδιάμεσος* (message-oriented middleware). Σε αντικειμενοστρεφείς ενδιάμεσους όπως η DCOM και η CORBA τα συμβάντα και τα μηνύματα είναι το ίδιο σημαντικά.
- Ο συνδεμοστρεφής προγραμματισμός εφαρμόζεται κυρίως στο συνδυασμό προκατασκευασμένων συστατικών ή παραγώγων τους αντικειμένων. Αυτό μπορεί να παρομοιαστεί με την περίπτωση ενώσεως ολοκληρωμένων κυκλωμάτων στα οποία υπάρχει συμμετρία συνδέσμων για εισαγωγή και εξαγωγή πληροφοριών.
- Η διαφορά στις διασυνδέσεις εισαγωγής και εξαγωγής έγκειται στη σχέση τους με ένα συστατικό. Στη περίπτωση διασύνδεσης εξαγωγής, αυτή καθορίζει τα δυνατά συμβάντα και μηνύματα. Στη περίπτωση του ασύνδετου συμβατού προτύπου, καθορίζονται άμεσοι ή στατικοί συνδέσμοι, δηλ. υπάρχει άμεση παραμετροποίηση με αναφορές διεργασιών και μεθόδων.

## Συνδεοστρεφής Προγραμματισμός

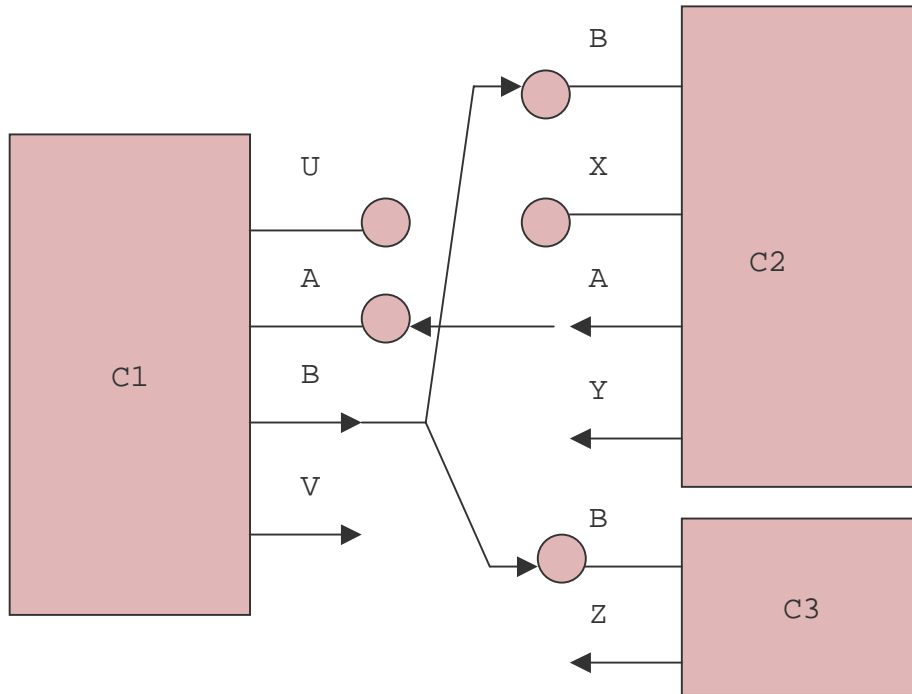
- Οι ίδιες οι συνδέσεις είναι ανεξάρτητες από τις έννοιες της εισαγωγής, εξαγωγής και επιπέδων. Ένα συστατικό υποδομής μπορεί να δίνει σε ένα κοινό επίπεδο πολλές διασυνδέσεις. Ένα συστατικό γι' αυτή την υποδομή εισάγει το κοινό επίπεδο και δηλώνει ποιές διασυνδέσεις χρησιμοποιεί. Τέτοιες συνδέσεις υπάρχουν *οριζόντια* σε αυτό το επίπεδο των συστατικών ή λογικά σ' ένα ψηλότερο επίπεδο που εδραιώνει τις συνδέσεις.
- Στο σχήμα 5.1 επεξηγούνται οι συνδέσμοι δύο αυτοκειμένων. Το συστατικό Ψ1 έχει διασυνδέσεις εξαγωγής B και V και εισαγωγής A και U. Σύνδεσμοι έχουν αποκατασταθεί μεταξύ των διασυνδέσεων A και B.



Σχ. 5.1 Συνδέσεις, διασυνδέσεις εισαγωγής - εξαγωγής.

## 5.4 Ανώτερος Συνδεσμολογίας Προγραμματισμός

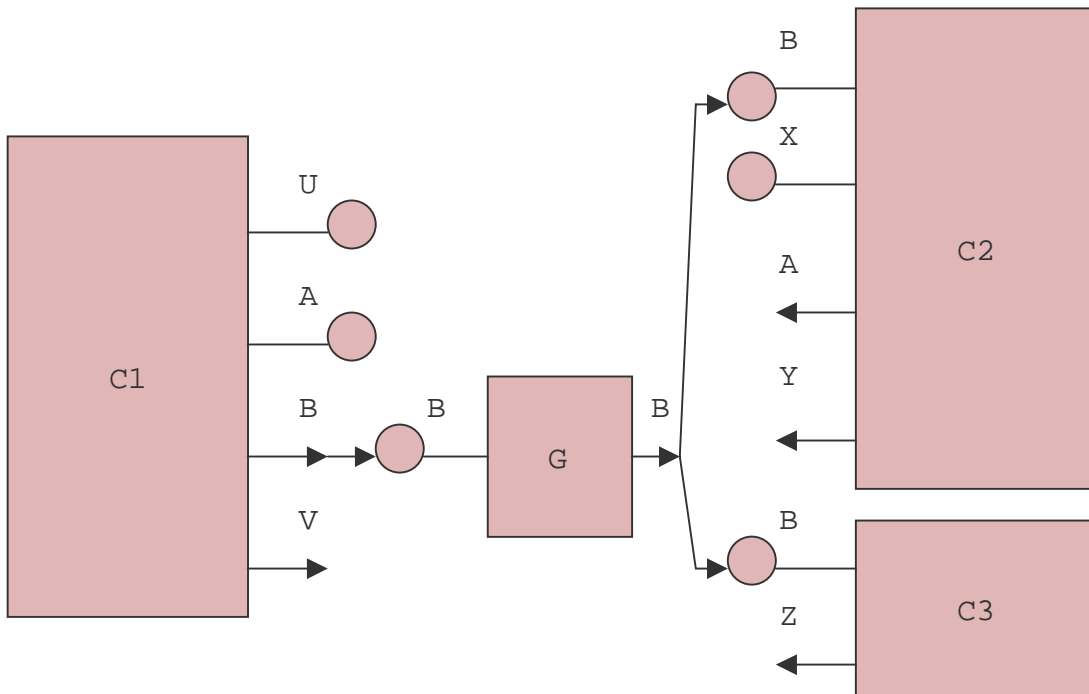
- Πολλαπλές συνδέσεις γίνονται από μία διασύνδεση εξαγωγής σε μία ομάδα εισαγωγής. Στο σχ. 5.2 η εξαγωγή Β συνδέεται με δύο διασυνδέσεις εισαγωγής Β των συστατικών C2 και C3. Φυσικά μπορεί να συμβεί και το αντίθετο.



Σχ. 5.2 Πολλαπλές συνδέσεις.

## Ανώτερος Συνδεσμολογικός Προγραμματισμός

- Η συσχέτιση διασυνδέσεων μπορεί να γίνεται μέσω άλλων συστατικών-αντικειμένων, όπως φαίνεται από το σχ. 5.3. Τέτοια συστατικά-διόδοι διηθούν, μετρούν, καταχωρούν για επανάληψη και γενικά διαχειρίζονται κλήσεις.



Σχ. 5.3 Ο διαχωρισμός της ομάδας των παραληπτών από τις πηγές και υποδοχές των μηνυμάτων (ή της διόδου των συμβάντων από τις πηγές και τις υποδοχές των συμβάντων).

## Ανώτερος Συνδεσμωστρεφής Προγραμματισμός

Αν μία ομάδα υποστηρίζει συγκεκριμένες διασυνδέσεις όπως η Β (σχ. 5.3), τότε απλά η διαχείριση της ομάδας περνά μηνύματα στα μέλη.

- Αν οι κλήσεις γίνονται δια μέσου των ιδίων μεθόδων ή διαδικασιών (μεταβιβαζόμενες μεταξύ συστατικών), τότε η δημιουργία ομάδων διόδων για την μετάδοσή τους είναι εύκολη. Τέτοια τυποποιημένα μηνύματα αποκαλούνται *αντικείμενα συμβάντων*.
- Ο κώδικας που ακολουθεί εξηγεί την μετάδοση μηνυμάτων σε ομάδες. Η σύνταξη είναι Component Pascal.

DEFINITION MessageGroups ;

TYPE

```
Sink(A)=POINTER TO ABSTRACT RECORD
  (s:Sink(A)) Receive (VAR message: A), NEW, ABSTRACT
END
```

```
Group(A)= POINTER TO ABSTRACT RECORD
  (g:Group(A)) Register (s:Sink(A)),NEW,ABSTRACT;
  (g:Group(A)) Unregister (s:Sink(A)),NEW,ABSTRACT;
  (g:Group(A)) Send(VAR message: A),NEW,ABSTRACT
END;
```

END MessageGroups.

Η παράμετρος NEW ορίζει την μέθοδο σαν νέα. Η παράμετρος ABSTRACT ορίζει την μέθοδο σαν μη συγκεκριμένη. Η χρησιμοποίηση παραμέτρων στους διαμορφωτές μηνυμάτων εξασφαλίζει τη συμβατότητά τους με τους υποδοχείς.

- Σε αυτό το παράδειγμα η παραμετροποίηση δείχνει ότι οι τύποι ομάδων και υποδοχέων μπορούν να εφαρμοσθούν με απόλυτους τύπους μηνυμάτων. Μία υποδοχή που δέχεται μόνο μηνύματα τύπου MyMessage θα δηλωθεί σαν:

TYPE

```
MyMessage=RECORD ... END;
MySink=RECORD (MessageGroups, Sink(MyMessage)) ...END;
```

Και για να δέχεται υπο-τύπους του MyMessage:

TYPE

```
MyMessage=EXTENSIBLE RECORD ... END;
```

Η ιδιότητα EXTENSIBLE δείχνει ότι ο τύπος του MyMessage δεν είναι τελικός.

## Ανώτερος Συνδεσμωτικός Προγραμματισμός

- Ο παρακάτω κώδικας δείχνει την ομαδοποίηση μετάδοσης από το διαμορφωτή Broadcast.

```

MODULE Broadcast;
  IMPORT Sets, MessageGroups;
  TYPE
    Bcaster(a)=POINTER TO RECORD(MessageGroups.Group(A))
      sinks: Sets.Set(MessageGroups.Sink(A))
  END;
  PROCEDURE<A> SendOne(s:MessageGroups.Sink(a);VAR message:
A);
  (* aux for send *)
  BEGIN s.Send(message)
  END SendOne;
  PROCEDURE<A> (g:Bcaster(A))
Register(s:MessageGroups.Sink(A));
  BEGIN g.sinks.Include(s)
  END Register;
  PROCEDURE<A> (g:Bcaster(A)) Unregister
(s:MessageGroups.Sink(A));
  BEGIN g.sinks.Exclude(s)
  END Unregister;
  PROCEDURE<A> (g:Bcaster(A)) Send(VAR message:A);
  BEGIN Sets.Do(g.sinks,SendOne,m)
  END Send;
  PROCEDURE<A> New*():MessageGroups.Group(A); (*simple
factory*)
  VAR bc:Bcaster(A);
  BEGIN
  NEW(bc);bc.sinks:=Sets.New();
  RETURN bc
  END New;
END Broadcast.

```

## Ανώτερος Συνδεσμωτικός Προγραμματισμός

Στη Component Pascal ο αστερίσκος χρησιμοποιείται για να προσδιορίζει διεργασίες που δύνανται να εισαχθούν και σε άλλους διαμορφωτές. Αυτό συμβαίνει με τη διεργασία Broadcast.New. Οι διεργασίες δεν παραμετροποιούνται άμεσα.

Υπονοούνται στις παραμέτρους τη στιγμή της κλήσης.

Ο ορισμός του διαμορφωτή Sets:

```
DEFINITION Sets;
```

```
  TYPE
```

```
    Set(A)=POINTER TO LIMITED RECORD
```

```
      (s:Set(A)) Include (elem: POINTER TO A);
```

```
      (s:Set(A)) Exclude (elem:POINTER TO A);
```

```
      (s:Set(A)) Contains(elem:POINTER TO A):BOOLEAN;
```

```
      (s:Set(A)) Size():INTEGER
```

```
  END;
```

```
Op(A,B)=PROCEDURE (elem:POINTER TO A;VAR arg: B);
```

```
  PROCEDURE<A> New():Set(A);
```

```
  PROCEDURE<A> Do(s:Set(A);op:Op(A<B);VAR arg:B);
```

```
  (*other definitions deleted*)
```

```
END Sets.
```

- Η ιδιότητα LIMITED εμποδίζει την επέκταση του καταχωρητή και περιορίζει την ενεργοποίησή του στο συγκεκριμένο διαμορφωτή.
- Ο διαμορφωτής Sets μπορεί να διαθέσει το αντικείμενο Set.
- Η μέθοδος Send (του Broadcast) χρησιμοποιεί τη διεργασία Sets.Do για να καλέσει την SendOne άπαξ για κάθε στοιχείο του Set.
- Αντίθετα με την Java η Component Pascal μπορεί να ορίσει απλαισιώτους τύπους αντικειμένων. Η γενική παράμετρος μπορεί να δοθεί στο πλαίσιο της καλούσης διεργασίας.

## 5.5 Συμβάντα και Μηνύματα

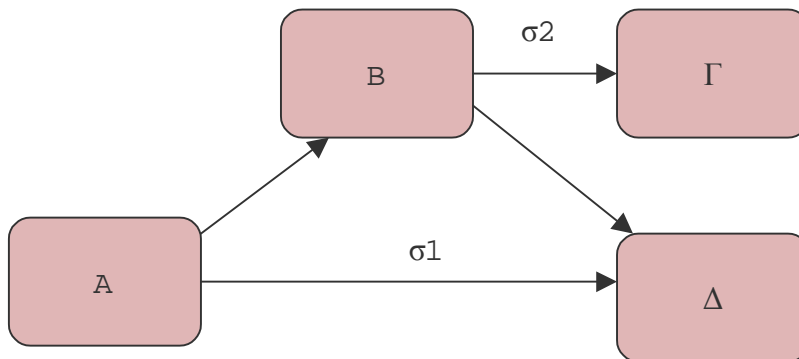
- Δήλωση και μετάδοση μηνυμάτων. Με μικρές διαφοροποιήσεις έτσι δουλεύει η JavaBeans, η CORBA Event Service, η COM connection points, και η Microsoft messaging service. Στο παράδειγμα της Connection Pascal ο καλών μία διεργασία πρέπει να περάσει από δύο στάδια.
  - Δημιουργία του αντικειμένου (μηνύματος) και παροχή αρχικών τιμών. Αντίθετα με τη Component Pascal (απλαισίοι τύποι αντικειμένων) η δημιουργία αντικειμένων-τάξεων στην JavaBeans είναι δαπανηρή. Επίσης απαραίτητη είναι η χρησιμοποίηση κατασκευαστών συμβάντων-αντικειμένων που να εξασφαλίζουν την παροχή αρχικών τιμών.
  - Αποστολή μηνύματος στις υποδοχές. Ο παραμετρικός πολυμορφισμός της Component Pascal υποστηρίζει τον προσδιορισμό του τύπου μηνύματος που πρέπει να περασθεί. Ενώ με τις JavaBeans και CORBA Event Service η χρησιμοποίηση συγκεκριμένων τύπων συμβάντων-αντικειμένων, αφήνει την πιθανότητα χρησιμοποίησης αγνώστων τύπων μηνυμάτων στις πηγές και τις υποδοχές.
- Οι δίοδοι μετάδοσης:
  - Μπορούν να ορίζονται από το Event Service (CORBA).
  - Μπορούν να διασυνδέονται με παρακάμψεις από το σύστημα (ή τα μηνύματα θα ταξιδεύουν κατ'ευθείαν από την πηγή στην υποδοχή -JavaBeans, COM).
- Σχετικά με τις υποδοχές οι υποδοχές:
  - Μπορεί να υπάρχει ξεχωριστή υποδοχή-αντικείμενο για κάθε σύνδεση στο ίδιο αντικείμενο (COM).
  - Μπορεί να χρησιμοποιηθεί αποδιανομέας-μετασχηματιστής για ξεχωρισμό διαφορετικών καλεσμάτων από διαφορετικές πηγές (JavaBeans).
- Τα δυνατά συμβάντα ή διασυνδέσεις είναι γνωστά αν καθορίζονται στον ορισμό του συστατικού. Στη JavaBeans η διασύνδεση των αντικειμένων περιέχει μία μέθοδο για κάθε δυνατό συμβάν που μπορεί να 'ακουστεί'. Μέ τη δυναμική απόσπαση των μεθόδων αυτών γίνονται γνωστά και τα αντίστοιχα συμβάντα.

## 5.6 Συμβάντα Έναντι Κλήσεων

- Τα συμβάντα αντιστοιχούν με κλήσεις διαδικασιών με τη διαφορά ότι
  - Έχουν άγνωστες υποδοχές.
  - Δεν αναμένονται αποτελέσματα.
  - Ειδοποιούν άλλες διεργασίες.
- Η CORBA EventServices και η JavaBeans δεν επιτρέπουν επιστροφή τιμών με συμβάντα.
- Σε περίπτωση που απαιτείται ανταπόκριση τότε το συμβάν-αντικείμενο μπορεί να μεταφέρει κάποια αναφορά στη διεργασία που το συλλέγει, που να προκαλεί την αποστολή απαντητικού συμβάντος.
- Άλλοι μηχανισμοί όπως το COM Connectable Objects υποστηρίζουν απ' ευθείας επιστροφή αποτελεσμάτων σε αντικείμενα-συμβάντα.

## 5.7 Διαδοχή Συμβάντων – Αιτιότητα, Συναγωνισμός και Αστάθεια Καταστάσεων

- Η επικοινωνία βασιζόμενη σε συμβάντα είναι ασύγχρονη. Η σχετική σειρά με την οποία τα συμβάντα φθάνουν στις υποδοχές είναι ακαθόριστη.
- Στο σχ. 5.4 παρατείνεται ένα παράδειγμα. Αν η πηγή A πυροδοτεί το συμβάν  $\sigma_1$  ενώ οι υποδοχές B και Γ προσέχουν, η υποδοχή B δέχεται το  $\sigma_1$  και αμέσως πυροδοτεί το  $\sigma_2$ , για το οποίο προσέχουν η Γ και η Δ. Η σειρά με τη οποία τα συμβάντα παρατηρούνται από τη Γ δεν καθορίζεται. Σε μονονηματικό (single-threaded) σύστημα ευνοείται η άφιξη του  $\sigma_2$  πριν από το  $\sigma_1$  στη Γ. Το πέρασμα του συμβάντος ελέγχεται από τη B που προκαλεί την παρατήρηση του  $\sigma_2$  από τη Γ. Συνήθως η παρατήρηση των συμβάντων με τη σειρά που πηγάζουν εξασφαλίζεται μόνο όταν αυτά έχουν την ίδια πηγή.



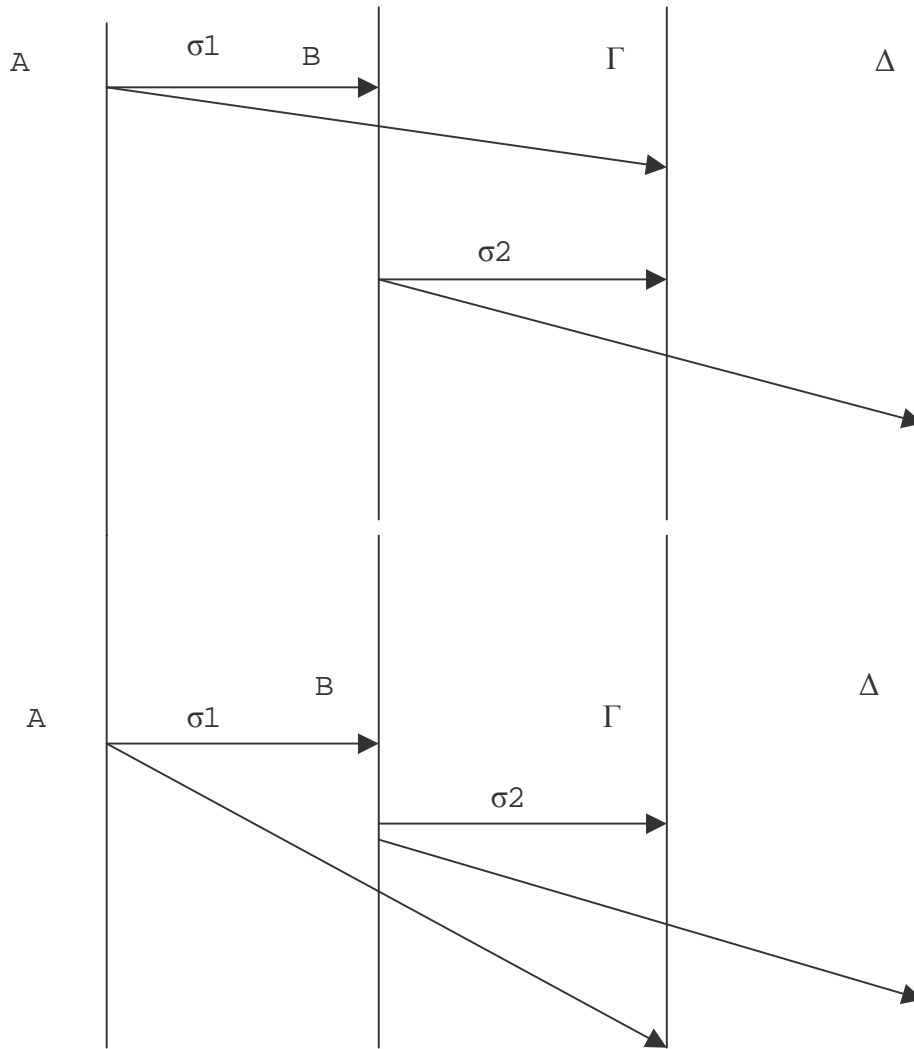
Σχ. 4 Προώθηση συμβάντων

Το ακόλουθο σχήμα 5.5 περιγράφει τη φυσική σειρά δηλ. αυτήν που είναι πλιό κοντά στην αιτιατή κατανομή των συμβάντων. Το κάτω του σχήματος περιγράφει την επαναληπτική σειρά κλήσεων, που είναι και η φυσική σειρά σε συστήματα που δεν χρησιμοποιούν ενδιάμεσους καταχωρητές και ξεχωριστά νήματα κατανομής συμβάντων. Η σειρά κατανομής είναι τυχαία.

Ένα ακόμη πιο σοβαρό θέμα στην υποδοχή σημάτων είναι αυτό της ασταθούς κατάστασης την ώρα της λήψης του συμβάντος. Αυτή η αστάθεια γίνεται αισθητή σε άλλα αντικείμενα για ένα μικρό χρονικό διάστημα. Αυτό διορθώνεται με μηχανισμό καθυστέρησης.

Αν η σχετική σειρά της άφιξης γεγονότων εξαρτάται από δυναμικές συνθήκες, τότε τα συμβάντα μπορεί να φθάσουν στη σωστή σειρά μόνο σε ορισμένες περιπτώσεις. Κατά κάποιο τρόπο υπάρχει συναγωνισμός στα συμβάντα για το πιο θα φθάσει πρώτο στο στόχο του. Η μη αυτοκρατία στο χρόνο αντικαθίσταται από ένα πολύ μεγάλο αριθμό σειρών καταστάσεων, σε ένα πολύπλοκο σύστημα συστατικών.

## Διαδοχή Συμβάντων - Αιτιότητα, Συναγωνισμός και Αστάθεια Καταστάσεων



Σχ. 5.5 Σειρά συμβάντων. Φυσική σειρά (πάνω) έναντι σειράς επαναληπτικής κλήσης (κάτω).

## 5.8 Αργή Σύνδεση: Απόσπαση Διασυνδέσεων και Μεταπρογραμματισμός

- Οι διασυνδέσεις, ειδοποιήσεις και συμβάντα είναι περιπτώσεις αργής σύνδεσης (*very late binding*).
  - Οι τύποι διασύνδεσης είναι γνωστοί αφού γίνει η μεταγλώττιση.
  - Η σύνδεση καλούντως και καλουμένου συμβαίνει αφού αρχίσει η εκτέλεση.
  - Συμβαίνει εντούτοις οι τύποι των διασυνδέσεων να γίνονται γνωστοί την ώρα της εκτέλεσης (περιπτώσεις μεταφραστών- *Web browser-applets*).
  - Φύλαξη πληροφοριών από μεταγλώττιση για έλεγχο την ώρα της εκτέλεσης (διαδικασία *επαναπόθεσης - reification*) - *αντανακλαστικά* λέγονται τα συστήματα που χρησιμοποιούν τέτοιες πληροφορίες. Ο χειρισμός τέτοιων πληροφοριών λέγεται *μεταπρογραμματισμός*.
  - Το *IBM's system object model (SOM)* είναι ένα τέτοιο σύστημα που υποστηρίζει μεταπρογραμματισμό - ορισμός νέων διασυνδέσεων και σύνθεση νέων τάξεων κατά την εκτέλεση καθώς και δημιουργία νέων περιπτώσεων. Το *SOM* επομένως μπορεί να επεμβαίνει στο εκτελεστικό πρότυπο.
  - Το *Java Core Reflection Service*: δέν επιτρέπει μεταπρογραμματισμό. Αφήνει στοιχειώδεις εκτελεστικού τυπου πληροφορίες στη *Java Language*, δηλ. Επιτρέπει τη διαδικασία επαναπόθεσης για όλες τις πλευρές ενός αντικειμένου, ή μίας τάξης, ή μίας διασύνδεσης και δυναμική ενεργοποίηση του.
  - Το *COM type library* που εξοπλίζει συστατικά με τύπους διασυνδέσεων αποσπάσεως (*dispatch interface*) για τους οποίους επιτρέπει τον έλεγχο (σύμφωνα με τους τύπους που περιέχει). Οι διασυνδέσεις απόσπασης μπορούν να ενεργοποιηθούν δυναμικά.
  - Η παροχή διασυνδέσεων αποσπάσης έχει μεγαλύτερο κόστος από την απ' ευθείας κλήση διαδικασιών. Η παροχή διαδικασιών σε συστατικά και με τους δύο τρόπους καλείται *διπλή διασύνδεση*.
  - Τέλος στην *CORBA* αποθήκες διασυνδέσεων επαναυποθέτουν όλες τις *Object Management Group / Interface Definition Language* πληροφορίες (αν συγκεκριμένα αντικείμενα μπορούν να εφαρμόσουν συγκεκριμένες διασυνδέσεις). Οι κλήσεις σε τέτοιες διασυνδέσεις γίνονται με δυναμικές ενεργοποιήσεις διασυνδέσεων και δυναμικές διασυνδέσεις προσαρμογής.

## Αργή Σύνδεση: Απόσπαση Διασυνδέσεων και Μεταπρογραμματισμός

- Οι αποσπασμένες διασυνδέσεις μπορούν εύκολα να κατασκευάσουν διεργασίες αναμετάδοσης και προσαρμογής χωρίς εξειδικευμένο κώδικα. Αυτό βέβαια υπονοεί τη χρησιμοποίηση αντανακλαστικών διαδικασιών ακόμα και αν η διασύνδεση είναι στατικά γνωστή. Για διεργασίες που στοιχίζουν πολύ απομόνες τους αυτό μπορεί να δικαιολογηθεί. Αλλιώςίτικα εμποδίζει τον αποτελεσματικό ευθύ προγραμματισμό του ως συστατικού και πρέπει να αποφεύγεται. Επίσης τα στατικά είναι λιγότερο ασφαλούς τύπου από τις συμβατικές διασυνδέσεις. Οι διπλές διασυνδέσεις είναι συμβιβασμός των δύο.
- Βλέποντας τις διαφορετικές μεθόδους κλήσεων και συνδέσεων:
  - Κλήσεις διεργασιών και στατικές μεθόδοι κλήσεων: στατικός έλεγχος και σύνδεση στη μεταγλώττιση.
  - Αντανακλαστικές κλήσεις: δυναμικός έλεγχος και σύνδεση κατά την εκτέλεση.
  - Μεταβλητές διεργασίες και απόσπαση μεθόδων: στο ενδιάμεσο των προηγούμενων βρίσκεται ο στατικός έλεγχος με δυναμική σύνδεση.

## 5.9 Διαβάθμιση Ελευθερίας: Απομόνωση Λαθών και Στατική Ασφάλεια

- Το θέμα της ασφάλειας λαθών συνήθως το εξασφαλίζει η γλώσσα προγραμματισμού. Αν όχι τότε η ασφάλεια αφήνεται σε χαμηλότερα επίπεδα. Παραδοσιακά, στο επίπεδο του λειτουργικού συστήματος εφαρμόζεται η *απομόνωση διεργασίας* (*process isolation*), με τη βοήθεια του μηχανικού εξοπλισμού. Στην περίπτωση των συστατικών αυτό κοστίζει πολύ. Η μέθοδος απομόνωσης λαθών λογισμικού συνίσταται στην ενίσχυση του κώδικα συστατικού κατά τη φόρτωση του.

## 5.10 Καταγραφή και Γραφή

- Πρότυπα συμβάντων και αποσπασμένες διασυνδέσεις υποστηρίζουν την καταγραφή δραστηριοτήτων. Η αναμετάδοση όταν όλα τα σχετικά αντικείμενα είναι δραστήρια μπορεί να προκαλέσει αστάθεια αναλόγως της κατάστασης των αντικειμένων.
- Η καταχώρηση συμβάντων δεν είναι αρκετή στην περίπτωση συστημάτων αυτόματης γραφής για επανεκτέλεση. Αυτή η γραφή θα πρέπει να είναι εφαρμόσιμη σε παρόμοιες παρά σε ίδιες καταστάσεις. Θα ήταν δηλαδή υπερβολικό να καταγράφεται επακριβώς η ταυτότητα των σχετικών αντικειμένων. Θα ήταν βασικό να χρησιμοποιείται μία συμβολική αναφορά αντικειμένων που είναι κατανοητή στο πλαίσιο του αντικειμένου. Στην επανάληψη μία γραφή θα μπορεί να ερμηνευθεί στο τρέχων πλαίσιο και στις συμβολικές αναφορές.