ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
«ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΙΣΜΟΥ»

Μεταπτυχιακή Εργασία

ΑΝΑΠΤΥΞΗ ΕΛΕΓΚΤΗ ΕΥΕΛΙΚΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΚΑΤΕΡΓΑΣΙΩΝ

ΜΕ ΧΡΗΣΗ ΠΡΑΚΤΟΡΩΝ ΚΑΙ ΔΙΚΤΥΩΝ ΤΟΥ PETRI

Σωτήριος Χ. Μεσσήνης

Επιβλέπων: Γ.-Χ. Βοσνιάκος

Αθήνα 2017

Περίληψη

Ο πρωταρχικός σχεδιασμός και η μετέπειτα υλοποίηση ελεγκτών στη βιομηχανία αποτελεί ένα εξαιρετικά ενδιαφέρον, ερευνητικά και βιομηχανικά, πεδίο. Η αναδυόμενη σημασία του σωστού σχεδιασμού και προσομοίωσης των εν λόγω ελεγκτών έγκειται στη σωστή εγκατάσταση, λειτουργία και εν γένει χρήση του βιομηχανικού/παραγωγικού εξοπλισμού της εκάστοτε βιομηχανίας με οφέλη τόσο οικονομικά, όσο και περιβαλλοντικά αλλά και ασφάλειας του ανθρώπινου παράγοντα.

Στην παρούσα εργασία εστιάζουμε και προχωρούμε στην ανάλυση μίας συγκεκριμένης τάξης συστημάτων κατεργασιών που καλούνται ευέλικτα συστήματα κατεργασιών (Flexible Manufacturing Systems – FMS). Πιο συγκεκριμένα, επιδιώκεται ο αποτελεσματικός σχεδιασμός και η προσομοίωση ελεγκτή για ένα ευέλικτο σύστημα κατεργασιών εργαστηριακής κλίμακας που βρίσκεται στο Εργαστήριο Τεχνολογίας των Κατεργασιών του Εθνικού Μετσόβιου Πολυτεχνείου.

Στα πλαίσια της παραπάνω ανάπτυξης, βασιζόμενοι στην Θεωρία Πολλαπλών Πρακτόρων (Multi Agent Systems), από τον κλάδο της Επιστήμης των Υπολογιστών και πιο συγκεκριμένα της Κατανεμημένης Τεχνητής Νοημοσύνης, προχωράμε στο σχεδιασμό και ανάλυση του ελεγκτή του ευέλικτου συστήματος κατεργασιών του εργαστηρίου. Η επιλογή της εν λόγω προσέγγισης βασίζεται στο ολοένα αυξανόμενο, κυρίως βιομηχανικά, συγκριτικό της πλεονέκτημα σε σχέση με ελεγκτές που βασίζονται καθαρά εντός του πλαισίου της Θεωρίας Συστημάτων Διακριτού Γεγονότος (Discrete Event Systems). Η εφαρμογή της μεθοδολογίας DACS από τη διεθνή βιβλιογραφία μας οδήγησε στην τελική επιλογή/διαμόρφωση των πρακτόρων του ευέλικτου συστήματος κατεργασιών. Στη συνέχεια, χρησιμοποιήθηκε το Contract Net Protocol για την υλοποίηση της επικοινωνίας μεταξύ των πρακτόρων και ενσωματώθηκαν κατάλληλα σε αυτό στοιχεία της Θεωρίας Δικτύων Petri για την αποφυγή ανεπιθύμητων καταστάσεων του συστήματος.

Η βασική συνεισφορά της παρούσας εργασίας βασίζεται στην ανάλυση σκοπιμότητας του ελεγκτή που σχεδιάστηκε, με την εφαρμογή τεχνικών από μία ειδική ομάδα συστημάτων, γνωστά στη διεθνή βιβλιογραφία ως Συστήματα Κατανομής Πόρων, διατυπωμένων σε Δίκτυα Petri. Οι τεχνικές αυτές οδήγησαν στην ανάπτυξη Αλγεβρικής Πολιτικής Αποφυγής Αδιεξόδου του Ευέλικτου Συστήματος Κατεργασιών και, εν συνεχεία, στην ενσωμάτωση αυτής της πολιτικής στον βασισμένο σε Πράκτορες Ελεγκτή.

by Sotirios Ch. Messinis, Postgraduate Student, NTUA

Abstract

The primary design and the subsequent implementation of industrial controllers steadily constitute an exciting research and industrial field. The emerging importance of correct design and simulation of these controllers is implied by the necessity of the correct installation and general use of the manufacturing/production machinery in a range of industries. The main advantages rely on financial/economic development, environmental protection and human safety.

In this work, we focus on the analysis of a special class of manufacturing systems, broadly known as Flexible Manufacturing Systems (FMS). More specifically, we pursue the efficient design and simulation of a controller for the Flexible Manufacturing System of the Manufacturing Technology Lab of National Technical University of Athens. In this facility framework, based on Multi Agent Systems theory – the descendant of Distributed Artificial Intelligence - we moved onto the design and analysis of the FMS controller. The choice of the latter approach is heavily based on its ever increasing industrial advantage comparing with the broadly prevalent discrete event controllers, which are based on techniques and methods from Discrete Event Systems theory. The application of DACS methodology from the international research bibliography let us choose and define the agents of our flexible system. Subsequently, we applied the Contract Net Protocol for the implementation of communication among the agents and we effectively incorporated in it structural details of the Petri Net theory. This provided the opportunity to avoid undesirable states of the system.

The main contribution of this work relies on the feasibility analysis of the resulting controller, based on implementation techniques from a special class of Discrete Event Systems, broadly known as Resource Allocation Systems. These systems, being defined in the framework of Petri Net theory, lead us to the development of an Algebraic Deadlock Avoidance Policy for the Flexible Manufacturing System under study. Ultimately, this policy was incorporated into the final Agent – Based controller.

Supervisor: George - Ch. Vosniakos, Professor of Manufacturing Systems, NTUA

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον Καθηγητή Γ.-Χ. Βοσνιάκο της Σχολής Μηχανολόγων Μηχανικών του ΕΜΠ, που μου έδωσε την ευκαιρία να εκπονήσω υπό την επίβλεψή του τη συγκεκριμένη μεταπτυχιακή εργασία, εντός ενός επιστημονικού πεδίου που με ενδιαφέρει πάρα πολύ.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Εισαγωγή

Ε1. Σκοπός της Εργασίας

Ο επαρκής και αποτελεσματικός σχεδιασμός βιομηχανικών ελεγκτών αποτελεί το πρώτο αναγκαίο στάδιο δημιουργίας και κατόπιν υλοποιήσεως αυτών. Γενικότερα, η αυξανόμενη ανάγκη προσαρμοστικότητας των μοντέρνων συστημάτων παραγωγής στις ολοένα και μεταβαλλόμενες εξωτερικές απαιτήσεις της αγοράς που φθάνουν σαν παραγγελίες σε αυτά, επιβάλει τη δημιουργία εύρωστων και πρακτικά εφικτών ελεγκτών για την ομαλή λειτουργία και παραγωγικότητα αυτών.

Τα ευέλικτα συστήματα κατεργασιών (FMS) είναι τα πλέον κατάλληλα από πλευρά δομής και προσαρμοστικότητας στις τρέχουσες αλλά και μελλοντικές απαιτήσεις της αγοράς προϊόντων διακριτής παραγωγής. Η βιομηχανία διακριτής παραγωγής (Discrete Manufacturing) περιλαμβάνει προϊόντα όπως: αυτοκίνητα, εξαρτήματα, αεροπλάνα, παιχνίδια, ηλεκτρονικές συσκευές και άλλα. Η ανάγκη δημιουργίας/σχεδιασμού συστημάτων που αποφεύγουν ανεπιθύμητες καταστάσεις, όπως τα αναδυόμενα deadlock κατά την λειτουργία αυτών, είναι, συνεπώς, πρωταρχικής σημασίας για όλα τα εμπλεκόμενα μέρη σε αυτά (μηχανήματα, άνθρωπος, περιβάλλον, οικονομία). Η ανάλυση και ο σχεδιασμός των απαιτούμενων λογικών ελεγκτών αυτών των συστημάτων θα πρέπει να περιλαμβάνει την αποφυγή τέτοιων πιθανών καταστάσεων, αλλά και να δίνει τη δυνατότητα στις μοντέρνες βιομηχανίες για την εφικτή και σχετικά άμεση υλοποίηση αυτών.
Στην παρούσα εργασία, επιδιώκεται η αποτελεσματική ενσωμάτωση ισχυρών τεχνικών αποφυγής deadlock από το επιστημονικό πεδίο των Συστημάτων Διακριτού Γεγονότος, στην ανάπτυξη ελεγκτή του ευέλικτου συστήματος κατεργασιών με βάση τα Συστήματα Πολλαπλών Πρακτόρων. Τα τελευταία συστήματα έχουν αποδειχθεί σαν μία πολλά υποσχόμενη προσέγγιση ευφυούς ελέγχου διαφόρων τεχνολογικών συστημάτων και εφαρμογών μεταξύ των οποίων είναι : αυτόνομα ρομποτικά συστήματα, ηλεκτρονικό εμπόριο, συστήματα μεταφορών, γραφικές εφαρμογές σε παιχνίδια Η/Υ και άλλα. Τις τελευταίες δεκαετίες έχει επιδιωχθεί με επιτυχία η εφαρμογή ευφυούς ελέγχου πραγματικού χρόνου, με βάση τα Συστήματα Πολλαπλών Πρακτόρων, και σε ποικίλα συστήματα παραγωγής. Η εφαρμογή κατάλληλης μεθοδολογίας και η επιλογή του κατάλληλου πρωτόκολλου επικοινωνίας των πρακτόρων (εμπλεκόμενων πόρων/ροών εργασίας) παρουσιάζεται λεπτομερώς στο δεύτερο μέρος της παρούσας εργασίας.

Ε2. Βιβλιογραφική Ανασκόπηση

Για την εκπόνηση της παρούσας εργασίας μελετήθηκαν συγκεκριμένες πηγές και επιστημονικά άρθρα από τη διεθνή βιβλιογραφία. Ειδικότερα, επιδιώχθηκε η όσο το δυνατόν αποτελεσματικότερη σύνδεση πηγών από δύο διαφορετικές διεθνείς επιστημονικές κοινότητες : της Θεωρίας Ελέγχου με έμφαση στα Συστήματα Διακριτού Γεγονότος και της Τεχνητής Νοημοσύνης με έμφαση στα Συστήματα Πολλαπλών Πρακτόρων και εν γένει της Κατανεμημένης Τεχνητής Νοημοσύνης. Για τον λόγο αυτό, αντλήθηκαν πληροφορίες πού προσέγγισαν το έργο της παρούσας εργασίας από διαφορετικές σκοπιές και οι οποίες επισημαίνονται επιλεκτικά και συνοπτικά ως ακολούθως :

Τα επιστημονικά άρθρα [9]-[13] μελετήθηκαν πρωτίστως για την κατανόηση της προσέγγισης των Συστημάτων Κατανομής Πόρων που υιοθετήσαμε στο Τρίτο μέρος της εργασίας. Συγκεκριμένα, στο άρθρο [9] οι J.Park και S.Reveliotis αναπτύσσουν συγκεκριμένη Αλγεβρική Πολιτική Αποφυγής Αδιεξόδου κάνοντας χρήση δομικής ανάλυσης Δικτύων Petri. Επιπλέον, το συγκεκριμένο άρθρο μας εισάγει σε καινούργιες ευέλικτες εφαρμογές της αναπτυχθείσας πολιτικής αποφυγής αδιεξόδου. Στο άρθρο [10] οι S. A. Reveliotis and J. Y. Choi προτείνουν μία αναλυτική μέθοδο δημιουργίας εποπτών (supervisors) που επιβάλουν την αντιστρεψιμότητα σε φραγμένα δίκτυα Petri. Αυτό επιτυγχάνεται κάνοντας χρήση στοιχείων από τη Θεωρία των Περιοχών (Theory of Regions) και την επιβολή γενικευμένων περιορισμών αμοιβαίου αποκλεισμού (GMECs) στη συμπεριφορά ενός δικτύου με χρήση θέσεων monitor. Οι S. Reveliotis, E. Roszkowska, και J. Choi [11] παρουσιάζουν μία γενίκευση συγκεκριμένης τάξεως πολιτικών αποφυγής αδιεξόδου (PK-DAPs) εστιάζοντας σε συμπεριφορές δικτύων που παράγουν μη κυρτή αναπαράσταση του χώρου καταστάσεών τους και που οι τεχνικές της Θεωρία των Περιοχών δεν μπορούν να αντιμετωπίσουν.

Επιπλέον τα βιβλία [13] και [14] μελετήθηκαν εκτενώς για την κάλυψη επιπρόσθετων γνώσεων στο ευρύτερο επιστημονικό πεδίο των Συστημάτων Διακριτού Γεγονότος και της σχετικής τεχνολογίας ελέγχου αυτών με σκοπό την αποφυγή αδιεξόδων. Ειδικότερα, από τα συγκεκριμένα βιβλία αντλήθηκαν πολύτιμες πληροφορίες που αξιοποιήθηκαν στο τρίτος μέρος της παρούσας εργασίας για την ανάπτυξη συγκεκριμένης Πολιτικής Αποφυγής Αδιεξόδου, εφαρμοσμένης στο ευέλικτο σύστημα κατεργασιών του Εργαστηρίου στο οποίο εκπονείται η εργασία.

Εν συνεχεία, τα κύρια άρθρα και βιβλία μελέτης για την δημιουργία ελεγκτή με βάση τα Συστήματα Πολλαπλών Πρακτόρων και την επιλογή κατάλληλου πρωτόκολλου επικοινωνίας, όπως αυτή αναπτύχθηκε στο δεύτερο μέρος της εργασίας είναι τα εξής:

Στο άρθρο [1] οι E.H. Durfee , V.R. Lesser και D.D. Corkill μας εισάγουν στη συνεργατική κατανεμημένη επίλυση προβλήματος, ένα επιστημονικό πεδίο συνδυασμού τεχνητής νοημοσύνης και κατανεμημένης επεξεργασίας. Από το άρθρο αντλούμε θεμελιώδεις γνώσεις της Θεωρίας Πολλαπλών Πρακτόρων που δημιουργήθηκε από τον παραπάνω συνδυασμό.

Από το άρθρο [4] αντλήσαμε σημαντικές πληροφορίες που μας οδήγησαν στην ανάπτυξη του τελικού ελεγκτή της παρούσας εργασίας. Ένα Σύστημα Κατεργασιών προσεγγίζεται σαν Σύστημα Πολλαπλών Πρακτόρων και γίνεται εφαρμογή Contract Nets για την αναγκαία επικοινωνία και συνεργασία μεταξύ των μερών του συστήματος. Για την αντιμετώπιση ανεπιθύμητων καταστάσεων του συστήματος γίνεται επιπλέον χρήση Δικτύων Petri.

Στο άρθρο [5] αναπτύσσεται επαρκής μηχανισμός συνεργασίας μεταξύ των μερών ενός Συστήματος Πολλαπλών Πρακτόρων με βάση τη Θεωρία των Δικτύων Petri. Πιο συγκεκριμένα, τα Δίκτυα Petri εισάγονται με κατάλληλο τρόπο σε ένα ΣΠΠ για να αποφευχθούν και αντιμετωπιστούν ενδεχόμενα αδιέξοδα του συστήματος.

Από τη μελέτη του βιβλίου[3], καταλήξαμε στην υιοθέτηση της προτεινόμενης μεθοδολογίας DACS για την αναγνώριση και διαμόρφωση των τελικών πρακτόρων του συστήματος μας. Μετέπειτα, από το βιβλίο [2] ανακτήθηκαν όλες οι απαραίτητες γνώσεις σχετικά με τον προγραμματισμό των διαμορφωμένων πρακτόρων σε περιβάλλον JAVA. Το πλαίσιο ανάπτυξης πρακτόρων JADE, που παρουσιάζεται στο βιβλίο, είναι το μοναδικό συμμορφούμενο με τις προδιαγραφές και απαιτήσεις του Οργανισμού Ευφυών Φυσικών Πρακτόρων (FIPA).

Η επιδίωξη συνδυασμού των Αλγεβρικών DAP – διατυπωμένων σε Δίκτυα Petri – για την ανάλυση σκοπιμότητας, και εν συνεχεία απαιτούμενου συντονισμού και συνεργασίας των αναπτυχθέντων πρακτόρων του τελικού ελεγκτή, δεν έχει μελετηθεί εκτενώς στη προαναφερόμενη βιβλιογραφία.

Ε3. Δομή Εργασίας

Η παρούσα εργασία δομείται ως ακολούθως :

- Στο πρώτο κεφάλαιο γίνεται η εισαγωγή βασικών όρων σχετικά με τα ευέλικτα συστήματα κατεργασιών, καθώς και η επεξήγηση αυτών με σκοπό να εισάγει τον αναγνώστη στα προς μελέτη και έλεγχο συστήματα. Στο ίδιο κεφάλαιο περιγράφουμε το Ευέλικτο Σύστημα Κατεργασίας του Εργαστηρίου Τεχνολογίας των Κατεργασιών του ΕΜΠ με το οποίο θα ασχοληθούμε στο υπόλοιπο της εργασίας.
- Στο δεύτερο κεφάλαιο γίνεται η εισαγωγή και επεξήγηση θεμελιωδών και βασικών όρων της θεωρίας των Δικτύων Petri και η αναγκαιότητα μοντελοποίησης των μοντέρνων συστημάτων κατεργασιών με τα διάφορα είδη Δικτύων που υπάρχουν στη βιβλιογραφία.
- Στο τρίτο κεφάλαιο γίνεται η παρουσίαση των λογισμικών PIPE2 και TINA που χρησιμοποιήθηκαν για την απεικόνιση και προσομοίωση του προς μελέτη συστήματος.
- Στο τέταρτο κεφάλαιο εισάγεται ο αναγνώστης στις θεμελιώδεις έννοιες των Συστημάτων Πολλαπλών Πρακτόρων και τις εφαρμογές αυτών, με έμφαση στον έλεγχο συστημάτων κατεργασιών.
- Στο πέμπτο κεφάλαιο αναπτύσσεται ο τελικός agent-based ελεγκτής με εφαρμογή συγκεκριμένου πρωτόκολλου επικοινωνίας στο κατάλληλο πλαίσιο ανάπτυξης πρακτόρων JADE.
- Στο έκτο κεφάλαιο γίνεται εισαγωγή στα Συστήματα Κατανομής Πόρων και της διατύπωσης αυτών με Δίκτυα Petri. Επιπλέον εισάγεται ο αναγνώστης σε συγκεκριμένη ομάδα Πολιτικών Αποφυγής Deadlock.
- Στο έβδομο κεφάλαιο γίνεται εφαρμογή της Πολιτικής Αποφυγής Deadlock που αναπτύχθηκε στο έκτο κεφάλαιο για την εξασφάλιση της ζωτικότητας και αντιστρεψιμότητας του δικτύου.
- Στο όγδοο κεφάλαιο γίνεται λεπτομερή ανάλυση των αποτελεσμάτων και επιβεβαίωση της λειτουργίας του ελεγκτή.
- Στο ένατο κεφάλαιο παραπέμπονται τα συμπεράσματα και οι μελλοντικές βελτιώσεις της παρούσας εργασίας.

ΜΕΡΟΣ ΠΡΩΤΟ

ΚΕΦΑΛΑΙΟ 1: Ευέλικτα Συστήματα Παραγωγής (FMS)

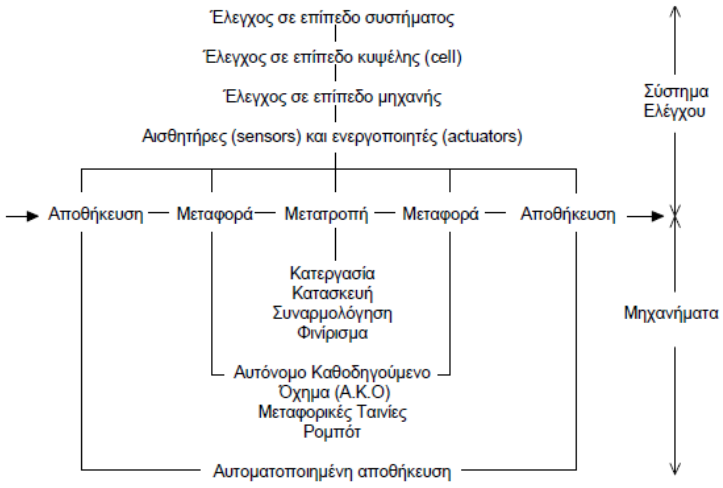
1.1 Εισαγωγή και Βασικές Έννοιες

1.1.1 Εισαγωγή

Το βασικό χαρακτηριστικό των **Ευέλικτων Συστημάτων Παραγωγής** (ή συνώνυμα σε αυτή την εργασία: Κατεργασιών) είναι η αυτοματοποίηση δηλαδή ο συνδυασμός των υπολογιστών με τις τεχνολογίες αυτομάτου ελέγχου μηχανών για την δημιουργία συστημάτων παραγωγής. Δύο βασικές διακρίσεις αυτοματοποίησης είναι :

- Αυτοματοποίηση πρώτου βαθμού (μηχανοποίηση εργασίας)
- Αυτοματοποίηση δευτέρου βαθμού (μεγάλα κέντρα κατεργασίας)

Οι αυξανόμενες ανάγκες για συστήματα παραγωγής με υψηλή παραγωγικότητα και ευελιξία, οδήγησαν στην ανάπτυξη μιας νέας γενιάς αυτοματοποιημένων συστημάτων παραγωγής. Ο αυτοματισμός ασχολείται όχι μόνο με τη μηχανοποίηση απλών και σύνθετων μεθόδων παραγωγής, αλλά και με τα συστήματα μεταφοράς και ελέγχου που τις συνδέουν. Η ευέλικτη αυτοματοποίηση βρίσκει εφαρμογή στους τομείς της επεξεργασίας, της κατασκευής, της συναρμολόγησης, του φινιρίσματος, της αποθήκευσης, της διαχείρισης και της μεταφοράς πρώτων υλών. Τα ευέλικτα συστήματα αυτοματοποίησης περιλαμβάνουν εξελιγμένες μηχανές παραγωγής και επιθεώρησης (ποιοτικού ελέγχου), συστήματα ελέγχου με υπολογιστές, συστήματα επικοινωνίας καθώς και συστήματα διαχείρισης υλικών [17].



Σχήμα 1.1 Σύνθεση και Λειτουργία Ευέλικτου Συστήματος Κατεργασιών

1.1.2 Ευελιξία συστήματος παραγωγής (εσωτερική και εξωτερική ευελιξία)

Η εσωτερική ευελιξία αφορά την ικανότητα του συστήματος παραγωγής να ανταποκρίνεται στις αλλαγές που προκαλούνται από την ίδια την επιχείρηση (π.χ. βελτίωση στο σχεδιασμό του προϊόντος) ενώ η εξωτερική ευελιξία αφορά την αντίδραση

του συστήματος σε ερεθίσματα προερχόμενα από το ευρύτερο περιβάλλον (π.χ. την αγορά). Η εσωτερική ευελιξία μπορεί περαιτέρω να διακριθεί ως εξής :

- Ευελιξία προϊόντος

Σχετίζεται με την ικανότητα ενός συστήματος παραγωγής να προσαρμόζεται στις απαιτήσεις ενός συγκεκριμένου προϊόντος.

- Ευελιξία ποικιλίας προϊόντων
  Η ευελιξία ποικιλίας επικεντρώνεται:

❖ Στον αριθμό των διαφορετικών προϊόντων που παράγονται από το σύστημα σε οποιαδήποτε χρονική στιγμή.
❖ Στη δυνατότητα του συστήματος παραγωγής να εναλλάσσεται μεταξύ δύο προϊόντων γρήγορα και με μικρό κόστος.
❖ Στις διάφορες παραλλαγές μεγέθους, σχήματος, πρώτων υλών, απαιτούμενων κατεργασιών κτλ.

Είναι σημαντικό, τα συστήματα παραγωγής να αντεπεξέρχονται δυναμικά στην εισαγωγή νέων προϊόντων και στην απόσυρση παλαιοτέρων με τις λιγότερες δυνατές διαταραχές. Η ικανότητα ταυτόχρονης διεκπεραίωσης πολλών εργασιών αναφέρεται και ως **"ευελιξία έργου".**

- Ευελιξία μεθόδων παραγωγής

Η ευελιξία μεθόδων παραγωγής είναι μέτρο της ευκολίας σχετικά με το πόσο μπορεί ένα σύστημα παραγωγής να ξεπερνάει τις δυσκολίες που προκαλούνται εκ των έσω. Η ικανότητα κατασκευής εξαρτημάτων και συναρμολογήσεων με διαφορετικούς τρόπους αποτελεί απαραίτητη προϋπόθεση (διευκολύνει τον προγραμματισμό, ελαχιστοποιεί τις επιδράσεις από τις βλάβες του συστήματος).

- Ευελιξία περιβάλλοντος

Επιδράσεις του ευρύτερου περιβάλλοντος στο παραγωγικό σύστημα:

❖ μεταβολές της ζήτησης και του κύκλου ζωής ενός προϊόντος
❖ οι τεχνολογικές αλλαγές: ανάπτυξη νέων υλικών και τεχνικών
❖ νέες συνήθειες και γενικότερη κατάσταση της οικονομίας

1.1.3 Δομή Ευέλικτου Συστήματος Παραγωγής.

Σχήμα 1.2 Δομή Ευέλικτου Συστήματος Κατεργασιών

Μονάδα (Module)

Μία αριθμητικά ελεγχόμενη εργαλειομηχανή με ενσωματωμένο μικροϋπολογιστή (CNC = Computerised Numerical Control). Περιλαμβάνει συσκευή αυτόματης αλλαγής εργαλείων, αποθήκη εργαλείων και ενδεχομένως αυτόματο σύστημα φόρτωσης /εκφόρτωσης και αποθήκευσης των προς κατεργασία κομματιών (μεταφορική ταινία, ρομποτικό βραχίονα, «ολοκληρωμένο» ρομπότ).

Κυψέλη (cell)

Μία κυψέλη αποτελείται από μικρό αριθμό συνδεδεμένων CNC μηχανών, με αυτόματη φόρτωση και εκφόρτωση (π.χ. ημικυκλική διάταξη γύρω από ένα ρομπότ ή γραμμική διάταξη με συνδεδεμένα π.χ. κέντρα κατεργασίας). Μία κυψέλη είναι απαραίτητο να περιλαμβάνει *σύστημα παλετών*, *αποθήκη εργαλείων* και σύστημα ελέγχου που επιτρέπει στην εγκατάσταση να είναι αυτό-ελεγχόμενη.

Σύστημα (system)

Ένα ευέλικτο σύστημα παραγωγής περιλαμβάνει αυτόματα συστήματα διακίνησης υλικών και ελέγχου, τα οποία λειτουργούν συμπληρωματικά των εργασιών των κυψελών ή και των μονάδων. Το σύστημα είναι μεγαλύτερο της κυψέλης και περιέχει επιπροσθέτως αυτόματα συστήματα αποθήκευσης. Λόγω των μεγαλύτερων αποστάσεων που πιθανόν να υπάρχουν στο επίπεδο του συστήματος, τα υλικά και τα εξαρτήματα μεταφέρονται με *αυτόματα καθοδηγούμενα οχήματα(AGV)* ή μεταφορικές ταινίες (conveyors).

Ολοκληρωμένη μέθοδος ελέγχου

Συντονίζει τις λειτουργίες των εργαλειομηχανών και των συστημάτων διαχείρισης υλικού ώστε να επιτύχει ευελιξία. Παράλληλα, αντικείμενο του σχεδιασμού ελέγχου του συστήματος είναι η κατανομή των εργασιών στο κύτταρο στοχεύοντας στην ελαχιστοποίηση του συνολικού μη παραγωγικού χρόνου, τη μεγιστοποίηση του βαθμού χρήσης των μηχανών και άλλα κριτήρια. Ο έλεγχος συστημάτων κατεργασιών πραγματοποιείται με την υλοποίηση κάποιας λογικής η οποία, για αυτοματοποιημένα συστήματα, ενσωματώνεται σε ένα πρόγραμμα λογισμικού και εκτελείται σε Προγραμματιζόμενους Λογικούς Ελεγκτές (PLC) ή σε βιομηχανικού τύπου PC. Για την επικοινωνία του ελεγκτή με τα υπόλοιπα στοιχεία του κυττάρου χρησιμοποιούνται τεχνολογίες δικτύων ή διαύλων.

Διατάξεις αυτόματης αλλαγής και διαχείρισης εργαλείων.

Η σημασία της επιλογής του σωστού εργαλείου, της παρακολούθησης της φθοράς , του πλήθους, της εναλλαξιμότητας και του κόστους των εργαλείων ενός ΕΣΚ δημιουργεί την ανάγκη για ένα ξεχωριστό σύστημα διαχείρισης εργαλείων. Σε ένα τέτοιο σύστημα είναι δυνατόν η αυτόματη αλλαγή των εργαλείων με κάποιο κριτήριο π.χ. την ελαχιστοποίηση αλλαγών εργαλείων. Η αυτόματη καταγραφή εργαλείων μπορεί να γίνεται με αισθητήρες ανάγνωσης barcode ή EPROMS με ραδιο-ζεύξη. Η καταγραφή της κατάστασης των εργαλείων μπορεί να γίνεται εκτός γραμμής από προσωπικό ή αυτόματα βάσει μέτρησης των διαστάσεων από ειδικούς αισθητήρες επαφής ή με έμμεση εκτίμηση των διαστάσεων μέσω μέτρησης δυνάμεων κατεργασίας ή απορροφώμενης ισχύος.

Επιπλέον σημαντικό στοιχείο ενός ευέλικτου συστήματος παραγωγής αποτελούν και οι ***Προγραμματιζόμενες ιδιοσυσκευές συγκράτησης των τεμαχίων.*** Με τις ιδιοσυσκευές επιτυγχάνεται συντόμευση, στο ελάχιστο δυνατό, του συνολικού χρόνου που είναι απαραίτητος για την εκτέλεση των βοηθητικών σταδίων κατεργασίας. Αυτό καθιστά την πορεία κατασκευής των διαφόρων στοιχείων μηχανών και προιόντων πιο σύντομη και πιο οικονομική.

1.1.4 Ευελιξία συστήματος παραγωγής και τύποι ευέλικτων συστημάτων

Η *ευελιξία* ενός συστήματος παραγωγής συνδέεται τόσο με την δυνατότητα παραγωγής πολλών διαφορετικών τύπων προϊόντων, όσο και με την ταχύτητα εφαρμογής των απαιτούμενων αλλαγών κατά τη μετάβαση από έναν τύπο προϊόντος σε έναν άλλο. Η ταχύτητα αλλαγής συνδέεται με τον χρόνο που απαιτείται για τον επανα-προγραμματισμό των μηχανών αλλά και για την φυσική προετοιμασία του εξοπλισμού με την αλλαγή εργαλείων, διατάξεων συγκράτησης κλπ. Γενικά, για να θεωρηθεί ευέλικτο ένα σύστημα παραγωγής πρέπει να ικανοποιεί τα εξής κριτήρια:

1. ικανότητα παραγωγής σχετικά μικρών παρτίδων, ακόμα και μιας μόνο μονάδας, με σχετικά μικρό κόστος,
2. ικανότητα ταχείας προσαρμογής σε αλλαγές στο πρόγραμμα παραγωγής (μίγμα προϊόντων, όγκος παραγωγής),
3. ικανότητα αναγνώρισης και διόρθωσης σφαλμάτων/βλαβών κατά την παραγωγή και γενικότερα ανάκαμψης του συστήματος, και
4. ικανότητα εύκολης εισαγωγής νέων προϊόντων στην παραγωγή.

Τα δύο πρώτα κριτήρια είναι και τα σημαντικότερα για να χαρακτηριστεί ένα σύστημα ευέλικτο, ενώ τα άλλα δύο αφορούν σε παράγοντες που αυξάνουν το γενικό βαθμό ευελιξίας του συστήματος. Από τεχνική άποψη το ιδανικά ευέλικτο σύστημα παραγωγής ενσωματώνει εξοπλισμό που μπορεί να τροποποιηθεί και να προγραμματιστεί εύκολα και γρήγορα για την παραγωγή μιας μεγάλης ποικιλίας προϊόντων, για αυτό και τα περισσότερα FMS διαθέτουν εξοπλισμό CNC και ρομπότ που είναι εξ ορισμού γενικής χρήσης (ευέλικτες) μηχανές.

Μια πρώτη ταξινόμηση των ευέλικτων συστημάτων βασίζεται στον αριθμό των μηχανών και το επίπεδο ευελιξίας που ενσωματώνουν. Με βάση τον αριθμό των μηχανών διακρίνονται, συνήθως, τρεις τύποι: το κύτταρο μιας μηχανής, το ευέλικτο κύτταρο παραγωγής και το ευέλικτο σύστημα παραγωγής [17].

Ένα ευέλικτο ***κύτταρο μιας μηχανής*** (single-machine cell) αποτελείται συνήθως από μια εργαλειομηχανή CNC και ένα αυτόματο σύστημα αποθήκευσης των τελικών προϊόντων. Το σύστημα είναι σχεδιασμένο να λειτουργεί χωρίς επιτήρηση και απαιτείται μόνο η περιοδική παραλαβή των έτοιμων κομματιών από τον αποθηκευτικό χώρο. Από τα τέσσερα κριτήρια που αναφέρθηκαν παραπάνω δεν ικανοποιείται το τρίτο, δεν διαθέτει δηλαδή μηχανισμούς επανόρθωσης και ανάκαμψης.

Ένα ***ευέλικτο κύτταρο παραγωγής*** (Flexible Manufacturing Cell - FMC) αποτελείται από 2-3 σταθμούς επεξεργασίας και ένα σύστημα μεταφοράς των κομματιών μεταξύ των σταθμών, το οποίο διαθέτει, συνήθως, και ένα περιορισμένο αποθηκευτικό χώρο. Το FMC ικανοποιεί σε μεγάλο βαθμό και τα τέσσερα κριτήρια ευελιξίας.

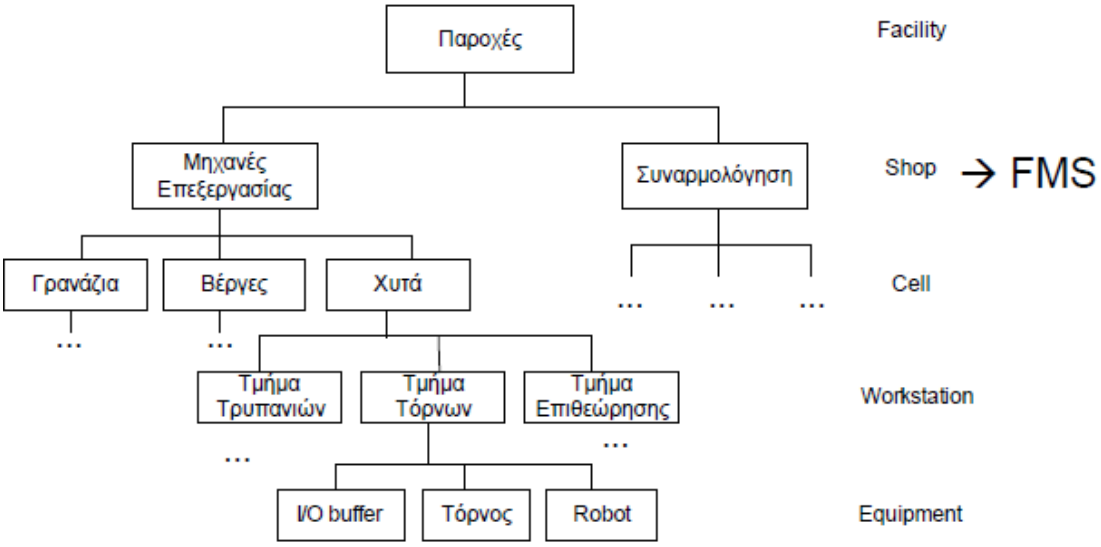
Σε σύγκριση με ένα ευέλικτο κύτταρο, το **ευέλικτο σύστημα παραγωγής** (FMS) διαθέτει συνήθως περισσότερους σταθμούς επεξεργασίας (πάνω από 3), εξοπλισμό ποιοτικού ελέγχου (π.χ. μετρητικές μηχανές), καθώς και υποστηρικτικό εξοπλισμό, όπως σταθμούς καθαρισμού των κομματιών ή των παλετών μεταφοράς των κομματιών.

1.1.5 Σχεδιασμός και Έλεγχος των FMS

Σε αυτό το στάδιο μας απασχολούν τα κάτωθι ερωτήματα:

- «τι» πρέπει να επικοινωνεί (από τα στοιχεία του συστήματος)
- «πως» επικοινωνούν τα στοιχεία αυτά μεταξύ τους

Το Εθνικό Ινστιτούτο Τυποποίησης και Τεχνολογίας της Αμερικής (National Institute of Standards and Technology, NIST) παρουσίασε ένα μοντέλο το οποίο διαιρεί τις παραγωγικές δραστηριότητες σε πέντε επίπεδα:



Σχήμα 1.3 Επίπεδα Παραγωγικών Δραστηριοτήτων κατά NIST

Ο σχεδιασμός εκτελείται σε κάθε επίπεδο αλλά ο ορίζοντας σχεδιασμού διαφέρει. Στον παρακάτω πίνακα διατυπώνονται βασικές διαφορές σχετικά με τον ορίζοντα σχεδιασμού.

Επίπεδο	**Ορίζοντας σχεδιασμού**	**Αποφάσεις**
Παροχές (facility)	Μήνες - χρόνια	Cad (manufacturing eng), σχεδιασμός παραγωγής, λογιστικά, ετήσια απόδοση, δυναμικότητα, συνολική παραγωγή, ποιότητα
Κατάστημα (shop)	Εβδομάδες - μήνες	Ομαδοποίηση παραγγελιών και ταξινόμηση, προληπτική συντήρηση, έλεγχος αποθεμάτων
Κυψέλη (cell)	Ώρες - εβδομάδες	Ακολουθία παρτίδων, δρομολόγηση εργασιών
Σταθμός κατεργασίας (workstation)	Λεπτά - ώρες	Εκκίνηση εργασιών και προσαρμογή, επιθεώρηση
Εξοπλισμός (equipment)	Χιλιοστά του δευτερολέπτου - λεπτά	Έλεγχος σε επίπεδο μηχανής, Συλλογή δεδομένων από αισθητήρες

Πίνακας 1.1 Ορίζοντας Σχεδιασμού Παραγωγής

Κλασικά ερωτήματα που αναδύονται κατά τον σχεδιασμό ευέλικτου συστήματος κατεργασιών είναι τα ακόλουθα [17]:

(Α) Επιλογή υλικού και λογισμικού (hardware and software) του συστήματος.

Υλικό του συστήματος: κέντρα κατεργασίας, κέντρα αλλαγής εργαλείων, συστήματα φόρτωσης/εκφόρτωσης παλετών, εργαλεία κοπής, παλέτες, εξαρτήματα, συστήματα αποθήκευσης, μηχανές προσωρινής αποθήκευσης, συστήματα μετακίνησης υλικών και συστήματα ελεγκτών.

(Β) Θεώρηση οικονομικού κριτηρίου: Για κάθε τύπο κομματιών υπολογίζεται το κόστος κατασκευής τους στο FMS, και το κόστος κατασκευής τους με εναλλακτικές διαδικασίες (εσωτερικά με συμβατικές μηχανές ή NC μηχανές ή με υπεργολαβίες). Κάθε κομμάτι που έχει χαμηλότερο κόστος παραγωγής με το FMS είναι υποψήφιο να μπει στη λίστα των κομματιών που θα παράγονται με το FMS. Το πρόβλημα επιλογής των κομματιών που θα παράγονται με το FMS ανάγεται στο κλασικό πρόβλημα του γυλιού (knapsack problem): παραγωγή κομματιών με το FMS με στόχο τη μεγιστοποίηση του κέρδους κάτω από τον περιορισμό της χωρητικότητας του FMS.

(Γ) Το τελευταίο θέμα στο σχεδιασμό του συστήματος είναι η χωροταξική διάταξη των μηχανών, των χώρων προσωρινής αποθήκευσης και της κεντρικής αποθήκης, καθώς επίσης και η σχεδίαση διακίνησης υλικών.

1.1.6 Εφαρμογές και Οφέλη

Τα ευέλικτα συστήματα παραγωγής ιστορικά βρίσκουν εφαρμογή κυρίως στην παραγωγή προϊόντων πρισματικής γεωμετρίας, μέσω κατεργασιών κοπής με φρέζα και δράπανο (π.χ. μπλοκ κυλίνδρων μιας μηχανής εσωτερικής καύσης). Σε σύγκριση με κυλινδρικής μορφής προϊόντα, τα κομμάτια αυτά είναι συνήθως βαρύτερα, πράγμα που καθιστά πιο δύσκολη τη διαχείρισή τους από έναν εργαζόμενο ή ρομπότ [17]. Το σχετικό κόστος παραγωγής τείνει να είναι αρκετά υψηλό, λόγω της πιο πολύπλοκης και χρονοβόρας διαδικασίας παραγωγής. Λόγω των παραπάνω η επένδυση σε ένα ευέλικτο σύστημα παραγωγής με δυνατότητα μεγάλης ποικιλίας προϊόντων μπορεί να θεωρηθεί οικονομικά δικαιολογημένη. Τα αξονο-συμμετρικά κομμάτια αντίθετα, έχουν συνήθως απλούστερη διαδικασία κατασκευής και είναι αρκετά ελαφρύτερα. Στην περίπτωση αυτή είναι σχετικά ευκολότερο να επιτευχθεί η απαιτούμενη ευελιξία με την χρήση ενός ρομπότ τοποθέτησης και ενός κέντρου κατεργασιών τόρνου.

Τα κύρια εκτιμώμενα οφέλη από την υλοποίηση ενός FMS μπορούν να συνοψιστούν ως εξής.

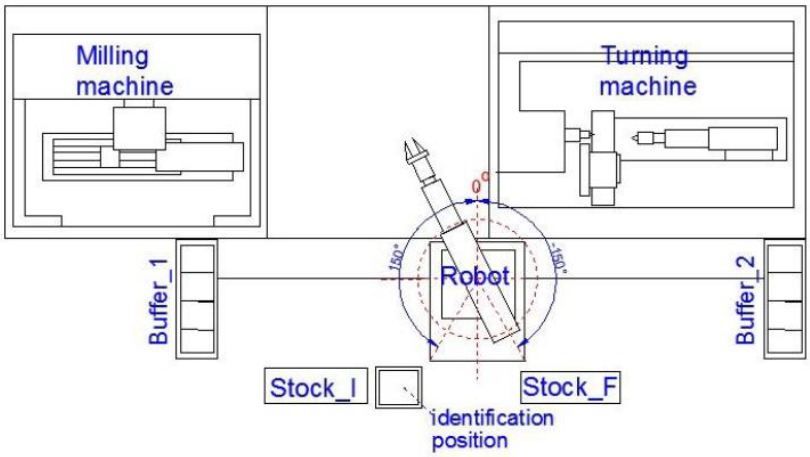
- Μεγαλύτερος βαθμός εκμετάλλευσης των μηχανών λόγω καλύτερου προγραμματισμού, καλύτερης χρήσης των περιοχών ενδιάμεσης αποθήκευσης και δυναμικού προγραμματισμού.
- Απαιτούνται λιγότερες μηχανές κυρίως λόγω του μεγαλύτερου βαθμού εκμετάλλευσης.
- Καλύτερη εκμετάλλευση χώρου. Σε σύγκριση με ένα τυπικό κατάστημα εργασιών εκτιμάται ότι ένα FMS απαιτεί περίπου τον μισό χώρο.
- Μεγαλύτερος βαθμός ανταπόκρισης σε αλλαγές. Η εισαγωγή νέων προϊόντων καθώς και οι αλλαγές στο πρόγραμμα παραγωγής είναι σχετικά ευκολότερες από ότι σε ένα σύστημα με χαρακτηριστικά αντίστοιχα της γραμμής παραγωγής.
- Μείωση αποθεμάτων/μονάδων υπό επεξεργασία. Επειδή κατασκευάζεται ένα μίγμα προϊόντων και όχι οι τυπικές ομοιογενείς παρτίδες προϊόντων, τόσο τα

αποθέματα πρώτων υλών και έτοιμων προϊόντων όσο και τα ενδιάμεσα αποθέματα (μονάδες υπό επεξεργασία) μπορούν να περιοριστούν σημαντικά, συνήθως σε ποσοστά 60-80%.

- Ταχύτερη παράδοση. Καθώς μειώνονται οι μονάδες προϊόντος υπό επεξεργασία μειώνεται αντίστοιχα ο χρόνος διεκπεραίωσης μιας εντολής παραγωγής ή παραγγελίας
- Υψηλότερη παραγωγικότητα εργασίας. Ο υψηλότερος ρυθμός παραγωγής σε συνδυασμό με τη χρήση τεχνολογιών αυτοματοποίησης και ψηφιακής τεχνολογίας αυξάνει την παραγωγικότητα της εργασίας και μπορεί να οδηγήσει σε εξοικονόμηση κόστους εργασίας.
- Δυνατότητα λειτουργίας χωρίς επίβλεψη. Το υψηλό επίπεδο αυτοματοποίησης επιτρέπει θεωρητικά τη λειτουργία του συστήματος συνεχώς (24 ώρες) και με ελάχιστη επίβλεψη.

1.2 Περιγραφή του Ευέλικτου Συστήματος Κατεργασιών της Εργασίας

Η διάταξη του Ευέλικτου Συστήματος Κατεργασιών του Εργαστηρίου Τεχνολογίας των Κατεργασιών του ΕΜΠ , απεικονίζεται σχηματικά ως ακολούθως.



Σχήμα 1.4 Κάτοψη Ευέλικτου Συστήματος Κατεργασιών Εργαστηρίου των Κατεργασιών ΕΜΠ [16]

1.2.1 Εξοπλισμός Ευέλικτου Συστήματος Κατεργασιών

Ο εξοπλισμός του FMS αποτελείται από:

- το κέντρο τόρνευσης cnc (M1)
- τη φρέζα cnc (M2)
- το ρομπότ (R)
- μια αρχική αποθήκη τεμαχίων (Stock_Input)
- μια θέση δίπλα στην αρχική αποθήκη των τεμαχίων όπου γίνεται ταυτοποίηση του είδους του τεμαχίου από ειδική κάμερα.
- Δύο ενδιάμεσες αποθήκες τεμαχίων στις οποίες τοποθετούνται κατεργασμένα τεμάχια που απαιτούν περαιτέρω κατεξεργασία (Buffer). Στη διάταξη υπάρχουν

δύο διαφορετικές ενδιάμεσοι χώροι αποθήκευσης B1 για τεμάχια με επόμενη κατεργασία στη φρέζα και B2 για τεμάχια με επόμενη κατεργασία στο κέντρο τόρνευσης

- μια τελική αποθήκη (Stock_Output) όπου τοποθετούνται τα πλήρως κατεργασμένα τεμάχια.

1.2.2 Περιγραφή Παραγωγικής Διαδικασίας

Στο FMS που μελετάμε στην παρούσα εργασία γίνεται παραγωγή τεσσάρων διαφορετικών προϊόντων με διαφορετικά βήματα κατεργασίας. Όπως συμβαίνει γενικότερα στον τομέα της διακριτής παραγωγής, υπάρχουν συγκεκριμένες εντολές παραγωγής που δέχεται το σύστημα σαν αποτέλεσμα συγκεκριμένων παραγγελιών (παρτίδες). Οι εντολές αυτές παίρνουν την μορφή φασεολογίου (συγκεκριμένος αριθμός και αλληλουχία κατεργασιών) για το κάθε είδος τεμαχίου.

Στην παρούσα εργασία θεωρούμε ότι στο ευέλικτο σύστημα κατεργασιών λαμβάνουν χώρα τέσσερις εργασίες (Jobs) που αντιστοιχούν στα τέσσερα διαφορετικά είδη τεμαχίων προς επεξεργασία. Πιο συγκεκριμένα, οι τέσσερις εργασίες είναι :

- ΠΡΟΙΟΝ Α : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΟΝ ΤΟΡΝΟ
- ΠΡΟΙΟΝ Β : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΗ ΦΡΕΖΑ
- ΠΡΟΙΟΝ C : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΟΝ ΤΟΡΝΟ → ΣΤΗ ΦΡΕΖΑ
- ΠΡΟΙΟΝ D : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΗ ΦΡΕΖΑ → ΣΤΟ ΤΟΡΝΟ

Η ενδιάμεση αποθήκευση διαθέτει συγκεκριμένη χωρητικότητα, έτσι ώστε να μεγιστοποιείται η απόδοση του συστήματος. Επιπλέον, η συγκεκριμένη πολιτική δίνει μεγαλύτερη ευελιξία στην αυτοματοποίηση του συγκεκριμένου συστήματος.

Η φόρτωση και εκφόρτωση των τεμαχίων στον τόρνο, τη φρέζα και τους αποθηκευτικούς χώρους γίνεται αυτόματα από το ρομπότ.

Οι βασικοί περιορισμοί του συστήματος σχετικά με την μεταφορά όλων των τεμαχίων (ρομπότ) και εν γένει τη λειτουργία του συστήματος είναι οι εξής :

- Το σύστημα διαθέτει δύο ‘εισόδους’. Μεταφορά τεμαχίου από την αρχική αποθήκη προς τον τόρνο και μεταφορά τεμαχίου από την αρχική αποθήκη προς την φρέζα.
- Όλες τις μεταφορές στο σύστημα τις πραγματοποιεί εξ ολοκλήρου το ρομπότ.
- Προτεραιότητα επεξεργασίας έχουν πάντοτε τα τεμάχια στα buffers που αναμένουν κατεργασία στις αντίστοιχες μηχανές. Με αυτόν τον τρόπο επιτυγχάνεται η ελαχιστοποίηση του συνολικού χρόνου παραγωγής των τεμαχίων.
- Όταν μία μηχανή είναι κατειλημμένη από τεμάχιο που υφίσταται κατεργασία, το επόμενο τεμάχιο προς κατεργασία σε αυτήν αναμένει στο αντίστοιχο buffer.
- Η χωρητικότητα στις ενδιάμεσες αποθήκες, στα πλαίσια της υπόλοιπης εργασίας, θεωρείται ίση με τέσσερα.
- Η ευφυΐα που επιδιώκεται να δοθεί στο σύστημα καλύπτει πρώτα την εκτέλεση εργασιών που βρίσκονται πιο κοντά σε πιθανό deadline από πλευράς παραγγελιών (orders).

- Η πιθανή μη διαθεσιμότητα μίας ενδιάμεσης αποθήκης σε κάποιο στάδιο της παραγωγής οδηγεί το ρομπότ σε αναμονή (idle state) για την αποδέσμευση της αντίστοιχης μηχανής κατεργασίας.

Οι παραπάνω απαιτήσεις είναι εκείνες που μας οδηγούν στη διερεύνηση αποτελεσματικού σχεδιασμού ελεγκτή του συγκεκριμένου συστήματος προς αποφυγή ανεπιθύμητων καταστάσεων όπως θα επισημάνουμε παρακάτω.

Επισημαίνεται σε αυτό το σημείο, ότι κατά τη μοντελοποίηση του συστήματος δεν λαμβάνονται υπ' όψη:
i) ο τρόπος λήψης και τοποθέτησης από το ρομπότ που είναι πιθανόν να διαφοροποιείται σε κάθε τεμάχιο
ii) οι διαφορετικές θέσεις τοποθέτησης των τεμαχίων στις ενδιάμεσες αποθήκες και η ενημέρωση του ελεγκτή σε σχέση με τη δέσμευση/ελευθέρωση αυτών.
Οι απλοποιήσεις αυτές δεν επηρεάζουν την ακρίβεια της μοντελοποίησης καθώς οι διαφοροποιημένες ενδεχομένως κινήσεις μιας εργασίας του ρομπότ αντιπροσωπεύουν την ίδια κατάσταση για το σύστημα.

ΚΕΦΑΛΑΙΟ 2: Θεωρία Δικτύων Petri

2.1.Εισαγωγή

Στο επιστημονικό πεδίο των Δυναμικών Συστημάτων Διακριτού Γεγονότος (DEDS) γίνεται εκτενής χρήση δύο μεθόδων μοντελοποίησης, κυρίως για την απεικόνιση φαινομένων ταυτοχρονισμού, παραλληλίας και σύγκρουσης σε διαφορετικού τύπου εγκαταστάσεις. Αυτές οι μέθοδοι μοντελοποίησης είναι τα Δίκτυα Petri και τα Finite State Automata.

Στο πεδίο εφαρμογών των Συστημάτων Κατεργασιών έχει εφαρμοστεί ευρέως ο πρώτος τύπος μοντελοποίησης ο οποίος έχει αποδειχθεί εξαιρετικά αποδοτικός από πλευράς απεικονίσεως λεπτομερών καταστάσεων στα πλαίσια της λειτουργίας του συστήματος.

Η επινόηση των Δικτύων Petri έγινε το 1962 από τον Carl Adam Petri και συναντάται για πρώτη φορά στην διδακτορική του διατριβή με τίτλο «*Επικοινωνία με Αυτόματα*». Στην άνω διατριβή, έγινε χρήση των Δικτύων Petri ως μαθηματικού εργαλείου με σκοπό αυτά να αποτελέσουν τη βάση μιας θεωρίας επικοινωνίας (συσχετίσεων και γεγονότων) μεταξύ των συστατικών μερών ενός υπολογιστικού συστήματος. Η κύρια ιδέα την οποία πραγματεύεται στην εργασία του ο Petri ήταν η διαπίστωση ότι η πιο αποτελεσματική μέθοδος για την τυποποιημένη ανάλυση ενός συνόλου από επικοινωνούντα αυτόματα, ήταν να δηλωθούν με τον ίδιο τρόπο οι αλλαγές κατάστασης στα αυτόματα και η επικοινωνία μεταξύ τους.

Τα δίκτυα Petri είναι ένα θεωρητικό μοντέλο ροής πληροφορίας. Ο σκοπός για τον οποίο αναπτύχθηκαν οι έννοιες, τα χαρακτηριστικά, οι ιδιότητες, τα εργαλεία ανάλυσης και οι τεχνικές που σχετίζονται με αυτά, ήταν η εύρεση απλών και αποτελεσματικών μεθόδων προκειμένου να γίνει η περιγραφή και ανάλυση της ροής πληροφορίας και ο έλεγχος συστημάτων. Τα δίκτυα Petri απαιτούν το συνδυασμό μαθηματικών και γραφικών εργαλείων για την πραγμάτωση της μοντελοποίησης συστημάτων που προσδιορίζονται ως παράλληλα, ασύγχρονα, κατανεμημένα, στοχαστικά ή μη αιτιοκρατικά.

Μερικές από τις πλέον κοινές καταστάσεις συστημάτων διακριτών γεγονότων για την αναπαράσταση των οποίων χρησιμοποιούνται δίκτυα Petri είναι αμοιβαία αποκλειόμενα γεγονότα, κατανομή κοινών πόρων σε ένα σύστημα, περιορισμοί προαπαιτούμενων και ακολουθίες γεγονότων. Τα βασικότερα αντικείμενα για τα οποία χρησιμοποιούνται τα Δίκτυα Petri είναι η μοντελοποίηση, προσομοίωση, αξιολόγηση αποδοτικότητας, ανάλυση συμπεριφοράς, επιβεβαίωση δομικών ιδιοτήτων, παρακολούθηση, χρονοπρογραμματισμός, εποπτικός έλεγχος και ο έλεγχος σε πραγματικό χρόνο συστημάτων [14].

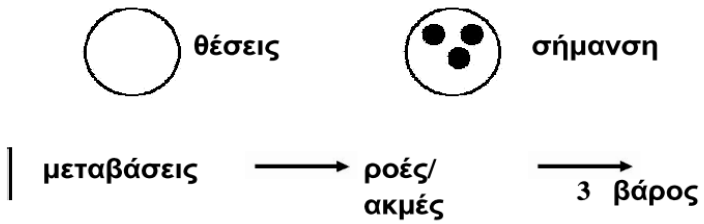
Τέλος, ενδεικτικά αναφέρεται ότι τα Δίκτυα Petri έχουν ένα ευρύ φάσμα εφαρμογών σε πολλούς τομείς, μεταξύ των οποίων συμπεριλαμβάνεται η μοντελοποίηση και αξιολόγηση αποδοτικότητας συστημάτων πολλών εξυπηρετητών -πολλαπλών ουρών, η μοντελοποίηση και μελέτη συστημάτων παραγωγής, οι εφαρμογές ρομποτικού ελέγχου, χημικών εργοστασίων, προβλήματα χρονοπρογραμματισμού και ελέγχου σιδηροδρομικών δικτύων, ενεργειακών συστημάτων, εφοδιαστικών αλυσίδων, η αυτοματοποίηση εργοστασίων, συστημάτων ελέγχου κυκλοφορίας (εναέριας ή επίγειας), συστήματα στρατιωτικών εντολών και ελέγχου, η επιλογή συμπεριφοράς πλοήγησης, η προσομοίωση πολλαπλών μη επανδρωμένων αεροσκαφών και η διαχείριση έργων.

Στο επιστημονικό πεδίο των Συστημάτων Κατεργασιών, τα δίκτυα Petri χρησιμοποιήθηκαν πρωταρχικά για την αναπαράσταση απλών γραμμών παραγωγής με ουρές, μηχανουργείων, αυτοματοποιημένων συστημάτων παραγωγής, και στη συνέχεια χρησιμοποιήθηκαν για την μοντελοποίηση ευέλικτων συστημάτων παραγωγής (FMS) , αυτοματοποιημένων γραμμών συναρμολόγησης, συστημάτων με κοινή χρήση-διαχείριση πόρων και σε συστήματα παραγωγής με μεθόδους just-in-time και Kanban.
Βασικά πλεονεκτήματα των Δικτύων Petri είναι [14] :

- Ευκολία στη μοντελοποίηση χαρακτηριστικών πολύπλοκων βιομηχανικών συστημάτων όπως : παραλληλία, σύγχρονες και ασύγχρονες λειτουργίες, συγκρούσεις, αμοιβαίος αποκλεισμός, σχέσεις προτεραιότητας, μη ντετερμινισμός και αδιέξοδα (deadlocks)
- Δυνατότητα δημιουργίας κώδικα κατευθείαν από τη γραφική παράσταση του δικτύου Petri. Ένας εκτελέσιμος κώδικας δικτύου Petri μπορεί επίσης να δημιουργηθεί σε εφαρμογές πραγματικού χρόνου χρησιμοποιώντας προγραμματιζόμενους λογικούς ελεγκτές ή υπολογιστές.
- Δυνατότητα διερεύνησης βασικών ιδιοτήτων του συστήματος (όπως η ύπαρξη κολλημάτων μέσω ανάλυσης σε υπολογιστικά προγράμματα βασισμένα στα δίκτυα Petri αποφεύγοντας πολυάριθμες προσομοιώσεις για πολλά σενάρια λειτουργίας του συστήματος
- Δυνατότητα ανάλυσης λειτουργίας και απόδοσης του συστήματος αξιολογώντας χρόνους παραγωγής, χρόνους αναμονής, βαθμό χρήσης πόρων.

2.2 Θεωρία, Έννοιες, Ορισμοί

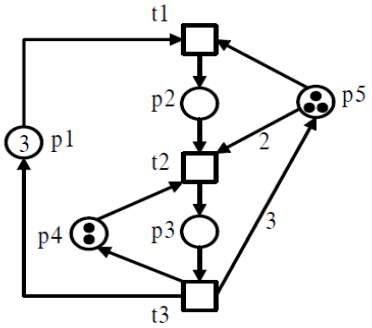
Τα Δίκτυα Petri είναι ένα είδος προσανατολισμένων διμερών γράφων. Αποτελούνται από δύο μέρη : μία δομή δικτύου και μία αρχική σήμανση. Ένα δίκτυο συμπεριλαμβάνει δύο είδη κόμβων : τις θέσεις και τις μεταβάσεις, οι οποίες συνδέονται μεταξύ τους με τόξα. Οι θέσεις απεικονίζονται γραφικά με κύκλους και οι μεταβάσεις με μπάρες. Οι θέσεις μπορούν να έχουν στο εσωτερικό τους μικρές μαύρες κουκίδες, ή ένα θετικό ακέραιο αριθμό που υποδηλώνει αυτές, οι οποίες είναι γνωστές στη διεθνή βιβλιογραφία ως tokens.



Σχήμα 2.1 Στοιχεία Δικτύων Petri

Η κατανομή των κουπονιών (tokens) στις θέσεις ενός δικτύου καλείται σήμανση (marking) και ανταποκρίνεται στην κατάσταση του μοντελοποιημένου συστήματος. Η αρχική σήμανση μας δίνει την αρχική κατανομή των tokens σε ένα δίκτυο. Γενικά ισχύει ο ακόλουθος ορισμός:
Ένα γενικευμένο Δίκτυο Petri είναι η τετράδα N = (P, T, F, W) όπου P και T είναι πεπερασμένα , όχι – κενά και αποσυνδεδεμένα σύνολα. Το P είναι το σύνολο των θέσεων και το T το σύνολο των μεταβάσεων ενός δικτύου με $P \cup T$ διαφορετική του μηδενός και $P \cap T = 0$. Το σύνολο $F = (P \times T) \cup (T \times P)$ καλείται ως σχέση ροής του

δικτύου απεικονίζοντας προσανατολισμένους κλάδους με τόξα - βέλη από τις θέσεις στις μεταβάσεις και αντίστροφα. Επιπλέον, στις μεταβάσεις σε κάποια δίκτυα μπορούν να υπάρχουν και βάρη $W:(P\times T)\cup(T\times P) \rightarrow N$,όπου N το σύνολο των μη αρνητικών ακεραίων.
Στην τελευταία περίπτωση γίνεται ανάθεση βάρους: $W(x,y)>0$ αν και μόνο αν (x,y) ανήκει στο F αλλιώς $W(x,y)=0$ (x,y ανήκουν στο $(P\times T)\cup(T\times P)$).
Μία σήμανση M ενός Δικτύου Petri είναι μία απεικόνιση από το P στο N. Το M(p) υποδηλώνει τον αριθμό των κουπονιών σε μία θέση p η οποία γεμίζει από μία σήμανση M εάν και μόνο αν $M(p) > 0$. Η ένδειξη $N = (P, T, F, W)$ καλείται σύνηθες δίκτυο (ordinary Petri Net) και έχει την ίδια ισχύ μοντελοποίησης όπως τα Γενικευμένα. Στο ακόλουθο σχήμα απεικονίζεται ένα σχετικά απλό Δίκτυο Petri που απεικονίζει όλα τα προαναφερόμενα.



Σχήμα 2.2 Δίκτυο Petri

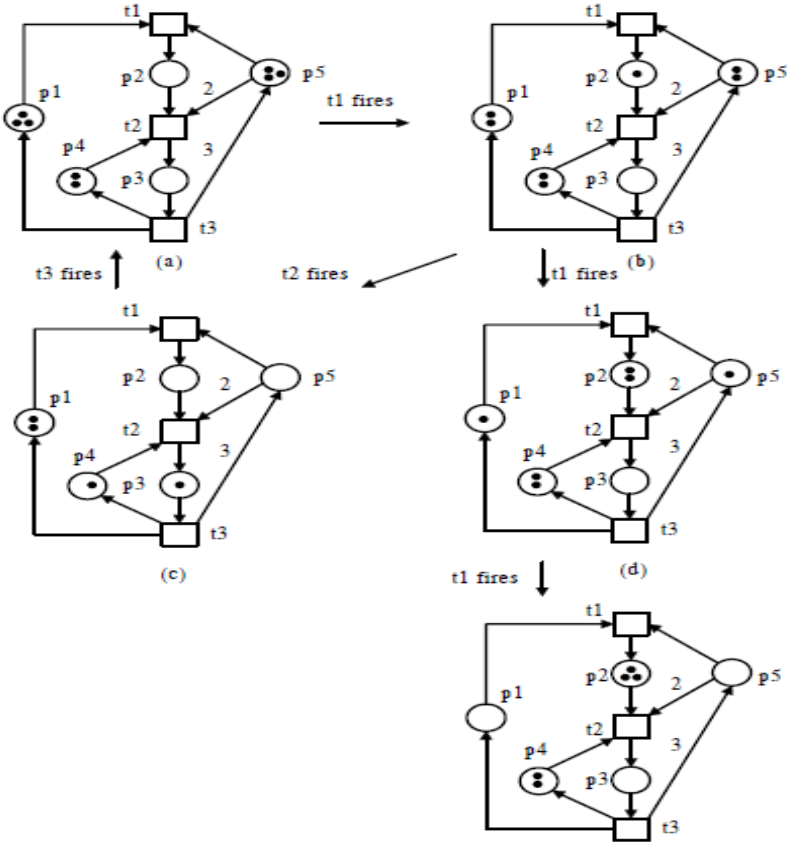
Για την σήμανση ενός δικτύου Petri χρησιμοποιείται ένας ακέραιος μη αρνητικός αριθμός για κάθε θέση, ο οποίος δείχνει τον αριθμό των τελειών που βρίσκονται στην θέση αυτή την χρονική στιγμή. Η σήμανση ενός δικτύου Petri συμβολίζεται με το γράμμα M και είναι ένα διάνυσμα μεγέθους ίσου με τον αριθμό θέσεων του. Η αρχική σήμανση ενός δικτύου Petri είναι υπεύθυνη για όλες τις σημάνσεις που εμφανίζονται κατά την εκτέλεση των διαδοχικών ενεργοποιήσεων. Οι σημάνσεις αυτές συμβολίζονται ως M_i όπου ο δείκτης i είναι θετικός ακέραιος αριθμός που δείχνει τον αύξοντα αριθμό της τρέχουσας ενεργοποίησης.
Όταν σε ένα Δίκτυο Petri η κατάσταση του οποίου περιγράφεται αρχικά από τη σήμανση M_i ενεργοποιηθεί μια μετάβαση t σε ετοιμότητα, προκύπτει η νέα του σήμανση M_{i+1} που περιγράφεται από την εξίσωση:

$$M_{i+1}(p) = M_i(p) - W(p, t) + W(t, p),$$

όπου W(p, t) τα βάρη μεταβάσεων από μία θέση εισόδου του δικτύου και W(t, p) τα βάρη μεταβάσεων θέσεων εξόδου αυτής.

Η σήμανση M_i+1 λέγεται προσεγγίσιμη από την αρχική σήμανση M_i. Η αλλαγή κατάστασης του Δικτύου Petri από M_i σε M_{i+1} λόγω ενεργοποίησης της t συμβολίζεται ως:
$M_i \rightarrow M_{i+1}$
Το σύνολο των σημάνσεων που επιτυγχάνεται από το M στο N ονομάζεται σύνολο προσβασιμότητας (reachability set) του δικτύου Petri (N, M) και δηλώνεται ως R(N,M). Το αντίστοιχο γράφημα, όπως θα δούμε παρακάτω, που απεικονίζει το σύνολο R

ονομάζεται γράφημα προσβασιμότητας. Στα παρακάτω σχήματα απεικονίζεται ενδεικτικά η εξέλιξη των σημάνσεων του δικτύου Petri του σχήματος 2.2.



Σχήμα 2.3 Εξέλιξη Σημάνσεων Δικτύου Petri [14]

Στη σήμανση M_1, οι μεταβάσεις t_1 και t_2 «καίγονται». Όταν «καεί» η t_2 στη M_1 οδηγεί στη M_2 όπως δείχνει το Σχήμα 2.3c. Όταν «καίγεται» η μετάβαση t_1 στη M_1 οδηγεί στη M_3 όπως απεικονίζεται στο Σχήμα 2.3d. Μόνο η μετάβαση t_1 «καίγεται» στη M_3. Στο σχήμα 2.3e απεικονίζεται η σήμανση M_4 αφού «καεί» η t_1 από τη M_3. Μετέπειτα, στη σήμανση M_2 μόνο η t_3 «καίγεται». Όταν «καεί», οδηγεί πίσω στο M_0.

Το σύνολο προσβασιμότητας του δικτύου είναι :

$M_0 = 3p_1+2p_4+3p_5$

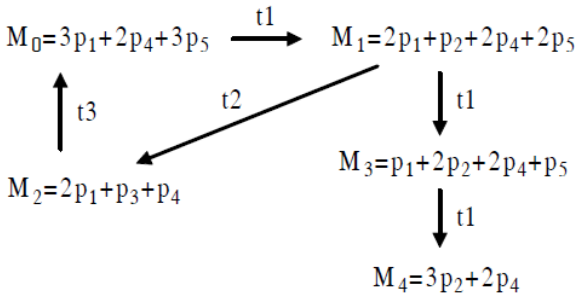
$M_1 = 2p_1 + p_2 +2p_4 +2p_5$

$M_2 = 2p_1 + p_3 + p_4$

$M_3 = p_1 +2p_2 +2p_4 + p_5$

$M_4 = 3p_2+2p_4$ (στη σήμανση M_4 καμία άλλη μετάβαση δεν ενεργοποιείται)

Το γράφημα προσβασιμότητας απεικονίζεται ως εξής :



Σχήμα 2.4 Γράφημα Προσβασιμότητας (Reachability Graph)

Ένα δίκτυο Petri N =(P,T,F,W) μπορεί να αντιπροσωπευτεί από τον πίνακα επίπτωσης [N], όπου [N] = |P|×|T| είναι ακέραιος πίνακας με [N](p, t)=W(t, p)−W(p, t). Για μία θέση p (μετάβαση t), το διάνυσμα σύμπτωσης, που μπορεί να είναι μία σειρά (ή στήλη), υποδηλώνεται ως [N](p, ·) ή ([N](·, t))[14].

2.3 Δυναμικές Καταστάσεις και Ιδιότητες Δικτύων Petri

Οι βασικότερες καταστάσεις που συναντάμε κατά τη μελέτη Συστημάτων Διακριτών Γεγονότων είναι η ακολουθία γεγονότων, η παραλληλία, ο αμοιβαίος αποκλεισμός, ο συγχρονισμός, το αδιέξοδο και η σύγκρουση [15]. Στο σημείο αυτό παρουσιάζεται ο τρόπος με τον οποίο μοντελοποιούνται οι καταστάσεις αυτές με τα δίκτυα Petri.

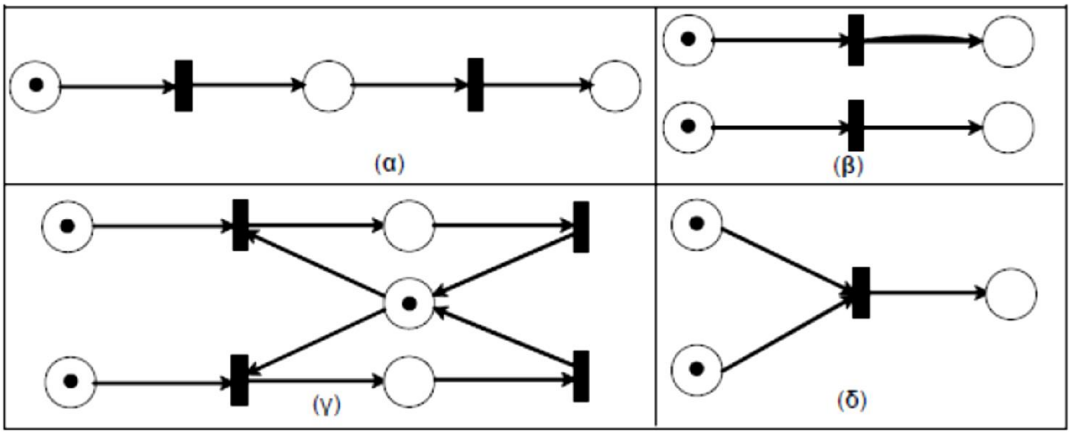
Α.) Η ακολουθία αναφέρεται σε ένα δίκτυο που αποτελείται από δύο μεταβάσεις, όπου η θέση εξόδου της πρώτης αποτελεί τη θέση εισόδου της δεύτερης. Στην περίπτωση αυτή η δεύτερη μετάβαση ενεργοποιείται μόνο εάν ενεργοποιηθεί η πρώτη μετάβαση.

Β.) Η παραλληλία συμβαίνει όταν δύο μεταβάσεις είναι σε ετοιμότητα και δεν αλληλεπιδρούν μεταξύ τους, οπότε μπορούν να πραγματοποιηθούν ταυτόχρονα.

Γ.) Ο αμοιβαίος αποκλεισμός συμβαίνει όταν έχουμε κοινούς πόρους σε ένα σύστημα. Σε αυτόν, δύο μεταβάσεις είναι παράλληλα σε ετοιμότητα σε κάποια σήμανση, όμως δεν μπορούν να ενεργοποιηθούν κι οι δύο εξαιτίας της ύπαρξης κοινής θέσης εισόδου η οποία περιέχει ένα μόνο κουπόνι. Η ενεργοποίηση της μιας μετάβασης αφαιρεί την ετοιμότητα της άλλης, η οποία έχει μια τουλάχιστον κενή θέση εισόδου.

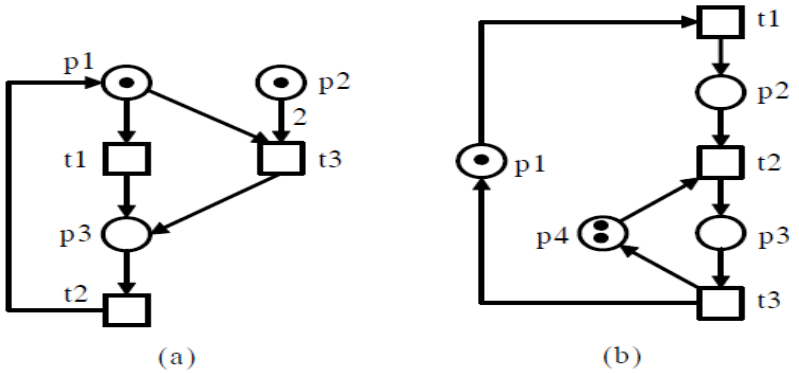
Ο αμοιβαίος αποκλεισμός αποτελεί περίπτωση σύγκρουσης. Σε περιπτώσεις συγκρούσεων, η επιλογή της μετάβασης που θα ενεργοποιηθεί, μπορεί να γίνει με διαφορετικά κριτήρια, όπως για παράδειγμα την ύπαρξη προτεραιοτήτων.

Δ.) Ο συγχρονισμός παρατηρείται όταν μια μετάβαση έχει περισσότερες από μια θέσεις εισόδου. Στην περίπτωση αυτή, η μετάβαση δεν μπορεί να τεθεί σε ετοιμότητα μέχρι να βρεθούν κουπόνια σε όλες της τις θέσεις εισόδου. Χαρακτηριστική περίπτωση κατεργασίας που είναι αναγκαίος ο συγχρονισμός αποτελούν οι συναρμολογήσεις στα συστήματα παραγωγής.



Σχήμα 2.5 Περιπτώσεις Δυναμικών Καταστάσεων Δικτύων Petri [15]

Αδιέξοδο εμφανίζεται σε ένα δίκτυο Petri όταν αυτό φτάσει σε μια κατάσταση όπου καμία μετάβαση δεν μπορεί να τεθεί σε ετοιμότητα και να ενεργοποιηθεί και συνεπώς η εκτέλεση του δικτύου διακόπτεται. Με λίγες εξαιρέσεις (π.χ. όταν μελετάται η εξυπηρέτηση συγκεκριμένου αριθμού πελατών σε ένα σύστημα), πρόκειται για ανεπιθύμητη κατάσταση που οφείλεται σε λάθος σχεδιασμού, κι απαιτεί τον επανασχεδιασμό μέρους του συστήματος ή του μοντέλου του. Ένα δίκτυο μπορεί να χαρακτηριστεί ως ελεύθερο αδιεξόδων αλλά όχι ζωντανό (live) όταν κάποιες από τις μεταβάσεις του είναι ζωντανές και κάποιες όχι Σχήμα 2.5(a.). Αντίθετα, ένα σύστημα που όλες οι μεταβάσεις είναι ζωντανές μπορεί να χαρακτηριστεί ως ζωντανό και κατά συνέπεια και ελεύθερο αδιεξόδων Σχήμα 2.5 (b).



Σχήμα 2.6 Ζωτικότητα και Deadlock Δικτύου Petri

Τα δίκτυα Petri, ως μαθηματικό εργαλείο, έχουν μια σειρά από ιδιότητες, τις οποίες διαχωρίζουμε σε ιδιότητες συμπεριφοράς και δομικές ιδιότητες. Οι πρώτες εξαρτώνται από την αρχική κατάσταση του δικτύου ενώ οι τελευταίες περιγράφουν το δίκτυο σχετικά με την δομή και την τοπολογία ανεξαρτήτως της αρχικής του κατάστασης. Στο σύνολό τους οι ιδιότητες αυτές, επιτρέπουν την αναγνώριση της παρουσίας ή μη μιας σειράς από λειτουργικά χαρακτηριστικά στα υπό εξέταση συστήματα. Με τον τρόπο αυτό, γίνεται δυνατή η κατηγοριοποίηση τους και η εξαγωγή συμπερασμάτων για τη συμπεριφορά των μοντέλων που υλοποιούνται.

Οι πιο χαρακτηριστικές ιδιότητες συμπεριφοράς είναι η προσβασιμότητα (reachability), η k-φραγή (k-boundedness) και η ασφάλεια (safety), η ζωτικότητα (liveness), η αντιστρεπτότητα (reversibility) και η επιμονή (persistence) [15].

Η έννοια της προσβασιμότητας όπως έχει προαναφερθεί, παρατηρείται όταν υπάρχει μια ακολουθία μεταβάσεων σ, η ενεργοποίηση των οποίων οδηγεί από την αρχική σήμανση M_0 στην M_i.
Η k-φραγή έχει έννοια τόσο για συγκεκριμένες θέσεις όσο και για ολόκληρα Δίκτυα Petri. Μια θέση p είναι k-φραγμένη με αναφορά στην M_0, αν ο αριθμός κουπονιών της θέσης αυτής είναι το πολύ ίσος με τον πεπερασμένο αριθμό k για όλες τις σημάνσεις που ανήκουν στο $R(M_0)$. Ομοίως, το Δίκτυο Petri είναι φραγμένο αν όλες του οι θέσεις είναι k-φραγμένες.
Η ασφάλεια είναι η ειδική περίπτωση της k-φραγής, όπου το k είναι ίσο με τη μονάδα. Η επιβεβαίωση της k-φραγής σε ένα μοντέλο συστήματος παραγωγής εγγυάται εξάλειψη της περίπτωσης υπέρβασης της χωρητικότητας των πεπερασμένων αποθηκευτικών χώρων του. Η ασφάλεια έχει την επιπλέον έννοια διαθεσιμότητας ενός πόρου κάθε χρονική στιγμή, ειδικά για την περίπτωση κοινών πόρων. Στα συστήματα παραγωγής η έννοια της φραγής συσχετίζεται στην πράξη με την έννοια της ευστάθειας, ειδικά αν το σύστημα μοντελοποιείται ως δίκτυο αναμονής.
Η έννοια της ζωτικότητας συνδέεται με την μη εμφάνιση αδιεξόδων. Ένα μοντέλο είναι ζωτικό αν κάθε μετάβαση του είναι ζωτική. Μια μετάβαση t είναι ζωτική αν υπάρχει η δυνατότητα ενεργοποίησής της μετά από την ενεργοποίηση μιας ακολουθίας μεταβάσεων. Όταν σε ένα δίκτυο Petri έστω και μια μετάβαση είναι ζωτική, αυτό δεν μπορεί να οδηγηθεί σε αδιέξοδο. Μια μετάβαση που δεν είναι ζωτική αναφέρεται ως νεκρή. Ένα Δίκτυο Petri του οποίου μόνο κάποιες από τις μεταβάσεις είναι ζωτικές, ονομάζεται μερικά ζωτικό.
Ένα Δίκτυο Petri ονομάζεται αντιστρεπτό αν για κάθε σήμανση M που ανήκει στο $R(M_0)$ ισχύει και M_0 να ανήκει στο R(M). Επομένως, για να είναι ένα Δίκτυο Petri αντιστρεπτό, πρέπει η αρχική του σήμανση να μπορεί να προσεγγιστεί από όλες τις προσεγγίσιμες σημάνσεις. Η αντιστρεπτότητα συνεπάγεται κυκλική συμπεριφορά του συστήματος. Επίσης, συνεπάγεται ότι το μοντέλο μπορεί από μόνο του να κάνει τις αναγκαίες ρυθμίσεις για την επανεκκίνηση του συστήματος, ιδιότητα σημαντική για την ανάκαμψη από σφάλματα (π.χ. βλάβες μηχανών) σε πεπερασμένο αριθμό βημάτων.
Τέλος, αναφέρονται ενδεικτικά οι δομικές ιδιότητες των Δικτύων Petri οι οποίες είναι: η δομική ζωτικότητα και ο δομικός περιορισμός, που αποτελούν επεκτάσεις των αντίστοιχων ιδιοτήτων συμπεριφοράς και ισχύουν για οποιαδήποτε πεπερασμένη σήμανση m_j. Επιπλέον, στην κατηγορία αυτή ιδιοτήτων εντάσσονται η συντηρητικότητα (conservativeness), η επαναληψιμότητα (repetitiveness) και η συνέπεια (consistency).

2.4 Κατηγορίες Δικτύων Petri

2.4.1 Δίκτυα Petri Χαμηλού Επιπέδου

Η διαδικασία κατηγοριοποίησης ενός δικτύου σε μια συγκεκριμένη δομική κατηγορία είναι εξαιρετικά χρήσιμη, διότι οι ιδιότητες που παρουσιάζουν τα δίκτυα κάθε κατηγορίας χρησιμοποιούνται για την διευκόλυνση της ανάλυσης. Ωστόσο, στην περίπτωση μοντελοποίησης πολύπλοκων ρεαλιστικών συστημάτων, ενδέχεται να προκύψουν σύνθετα δίκτυα, εάν επιλεγεί ένα μοντέλο χαμηλού επιπέδου. Για τη διευκόλυνση, ως ένα βαθμό, της συστηματικής δόμησης δικτύων έχουν προταθεί στην παρούσα εργασία, τεχνικές σύνθεσης και αποσύνθεσης με βάση συνιστώσες (τμήματα δικτύων) που ανήκουν σε μια συγκεκριμένη κάθε φορά κατηγορία, όπου μπορούν εύκολα

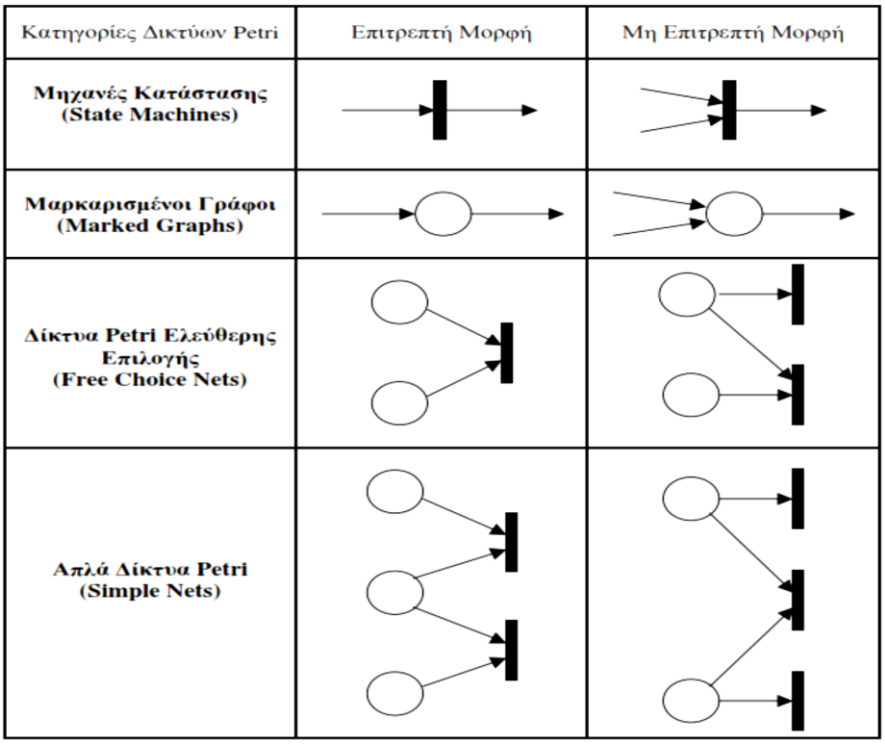
να εφαρμοστούν μέθοδοι ανάλυσης. Οι κατηγορίες των δικτύων Petri χαμηλού επιπέδου είναι οι εξής :

1) Μηχανές Κατάστασης (State Machines) Μηχανή κατάστασης είναι ένα δίκτυο Petri με την ιδιότητα ότι κάθε μετάβαση έχει ακριβώς μία θέση εισόδου και ακριβώς μία θέση εξόδου. Αυτό σημαίνει πως κάθε μετάβαση έχει μόνο ένα εισερχόμενο και μόνο ένα εξερχόμενο τόξο. Οι μηχανές κατάστασης επιτρέπουν την αναπαράσταση των συγκρούσεων αλλά δεν μπορούν να περιγράψουν συγχρονισμό ανάμεσα σε ταυτόχρονες ενέργειες.

2) Σημασμένοι Γράφοι (Marked Graphs) Μαρκαρισμένος γράφος είναι ένα δίκτυο Petri με την ιδιότητα ότι κάθε θέση έχει ακριβώς μία μετάβαση εισόδου και ακριβώς μία μετάβαση εξόδου. Αυτό σημαίνει πως κάθε θέση έχει μόνο ένα εισερχόμενο και μόνο ένα εξερχόμενο τόξο. Όλοι οι μαρκαρισμένοι γράφοι είναι επίμονοι και επιτρέπουν την αναπαράσταση ταυτόχρονων ενεργειών, αλλά αδυνατούν να περιγράψουν δομές συγκρούσεων. Επομένως, οι μαρκαρισμένοι γράφοι κρίνονται κατάλληλα για τη μοντελοποίηση ταυτόχρονων συστημάτων που είναι ελεύθερα αποφάσεων (decision-free).

3) Δίκτυα Petri Ελεύθερης Επιλογής (Free Choice Nets) Το Δίκτυο Petri ελεύθερης επιλογής έχει την ιδιότητα ότι στην τελική σχέση, κάθε θέση με κάθε μετάβαση συνδέονται ως εξής : είτε η μετάβαση είναι η μοναδική μετάβαση εξόδου αυτής της θέσης είτε η θέση είναι η μοναδική θέση εισόδου αυτής της μετάβασης. Δηλαδή κάθε τόξο, είτε είναι το μοναδικό εξερχόμενο τόξο μιας θέσης, είτε είναι το μοναδικό εισερχόμενο τόξο μιας μετάβασης. Αυτά τα δίκτυα δεν αναπαριστούν καταστάσεις σύγχυσης.

4) Απλά Δίκτυα Petri (Simple Nets) Ένα απλό δίκτυο Petri είναι ένα δίκτυο με την ιδιότητα ότι όλες οι μεταβάσεις έχουν το πολύ μία θέση εισόδου που οδηγεί σε άλλες μεταβάσεις. Δηλαδή, δεν υπάρχουν στο δίκτυο, δύο μεταβάσεις οι οποίες να έχουν τις ίδιες θέσεις εισόδου και ταυτόχρονα τις ίδιες θέσεις εξόδου.



Σχήμα 2.7 Κατηγορίες Δικτύων Petri [15]

2.4.2 Δίκτυα Petri Υψηλού Επιπέδου

Βασικός περιορισμός όλων των μοντέλων δικτύων Petri χαμηλού επιπέδου είναι η πολυπλοκότητα των χρησιμοποιούμενων δικτύων, για την περιγραφή εφαρμογών μεσαίου βαθμού πολυπλοκότητας. Με αυτόν τον τρόπο καθίσταται δύσκολη, τόσο η διαδικασία υπολογισμού των αμετάβλητων διανυσμάτων όσο και η ανάλυση του γραφήματος προσβασιμότητας, αφού σε μη πεπερασμένα δίκτυα παρουσιάζεται το λεγόμενο πρόβλημα της έκρηξης του χώρου καταστάσεων (state explosion problem). Ένα επιπλέον πρόβλημα είναι, η μη επαρκής υποστήριξη των χαμηλού επιπέδου μοντέλων για τον προσδιορισμό των διακεκριμένων οντοτήτων ενός συστήματος (individuals), των ιδιοτήτων και των συσχετισμών μεταξύ τους. Τέλος, ο φορμαλισμός των περισσότερων μοντέλων δεν υποστηρίζει ρητά τον προσδιορισμό συγκεκριμένων μηχανισμών δόμησης, όπως είναι οι τελεστές σύνθεσης.

Ορίζονται τέσσερις γενικές κατηγορίες για τα υψηλού επιπέδου δίκτυα Petri :

- επεκτάσεις που εμφανίζουν διακεκριμένες μάρκες (individual tokens), οι οποίες καλούνται και "καθαρά" δίκτυα Petri υψηλού επιπέδου (pure HPNs) [πχ. Χρωματισμένα δίκτυα Petri (Coloured PNs - CPNs), τα δίκτυα Κατηγορήματος - Μετάβασης (Predicate-Transition nets / PrT-nets) και τα δίκτυα με Διακεκριμένες Μάρκες (Individual Token Nets - ITNs)]
- υψηλού επιπέδου δίκτυα που εμφανίζουν τροποποιημένη σημασιολογία (high level PNs with modified semantics),
- επεκτάσεις που προτείνουν συγκεκριμένους μηχανισμούς δόμησης, γνωστές και ως ιεραρχικά δίκτυα Petri υψηλού επιπέδου (hierarchical high-level PNs / HHPNs) μοντέλα που υιοθετούν χαρακτηριστικά από άλλες τυπικές μέθοδες προσδιορισμού προδιαγραφών.

2.4.3 Η εισαγωγή του χρόνου στα Δίκτυα Petri

Σε πολλές εφαρμογές σχεδιασμού σύγχρονων συστημάτων κρίνεται αναγκαία η λήψη και η ενσωμάτωση του χρόνου στην μοντελοποίηση τους. Για το λόγο αυτό στη διεθνή βιβλιογραφία συναντά κανείς και χρονισμένες εκδόσεις δικτύων Petri (Timed Petri Nets, Simple Time PNs, Time PNs, Communicating Time PNs, Timed Place PNs,Timing Constraint PNs) με τα ακόλουθα δομικά στοιχεία[15]:

Χρονισμένες μεταβάσεις (timed transitions) : Ο χρόνος συνδέεται με τις μεταβάσεις ενός δικτύου. Η εκκίνηση της δράσης που σχετίζεται με μια μετάβαση αντιστοιχεί στην ενεργοποίηση της μετάβασης, ενώ ο τερματισμός της δράσης αντιστοιχεί στην πυροδότηση της μετάβασης.

Χρονισμένες θέσεις (timed places) : Ο χρόνος συνδέεται με τις θέσεις ενός δικτύου. Οι μάρκες στη θέση εισόδου μιας μετάβασης επιτρέπεται να πυροδοτήσουν τη μετάβαση, μόνο μετά το τέλος της χρονικής καθυστέρησης που σχετίζεται με τη συγκεκριμένη θέση εισόδου.

Χρονισμένοι κλάδοι (timed arcs) : Ο χρόνος συνδέεται με τα τόξα ενός δικτύου. Μια καθυστέρηση ταξιδιού (traveling delay) συνδέεται με κάθε τόξο. Οι μάρκες μπορούν να πυροδοτήσουν μια μετάβαση μόνο όταν φτάσουν στη μετάβαση.

Χρονισμένα κουπόνια (timed tokens) : Στην περίπτωση ενός χρονικού μοντέλου υψηλού επιπέδου, οι χρονικές προδιαγραφές συνήθως σχετίζονται με τις μάρκες του δικτύου, υπό

τον τύπο χρονο-σφραγίδων (time stamps). Οι μάρκες φέρουν μια χρονική σφραγίδα, η οποία υποδεικνύει πότε μπορούν να πυροδοτήσουν μια μετάβαση. Αυτή η χρονική σφραγίδα μπορεί να προσαυξηθεί σε κάθε πυροδότηση μετάβασης.

Επιπλέον, στο ευρύ πεδίο εφαρμογών της Θεωρίας των Δικτύων Petri μπορούν να εντοπισθούν και στοχαστικές εκδόσεις αυτών. Αντιπροσωπευτικές στοχαστικές εκδόσεις των δικτύων Petri είναι : τα Στοχαστικά δίκτυα Petri (Stochastic Petri nets - SPNs), τα Γενικευμένα Στοχαστικά δίκτυα Petri (Generalized Stochastic Petri nets - GSPNs), τα Ημι-μαρκοβιανά Στοχαστικά δίκτυα Petri (Semi-Markov Stochastic Petri nets - Semi-Markov SPNs), τα Στοχαστικά δίκτυα Petri τύπου Φάσης (Phase Type Stochastic Petri nets - PHSPNs) , τα Ντετερμινιστικά Στοχαστικά δίκτυα Petri (Deterministic Stochastic Petri nets - DSPNs) και τα Στοχαστικά δίκτυα Petri Αναγέννησης Markov (Markov Regenerative Stochastic Petri nets - MRSPNs).
Ένα στοχαστικό μοντέλο προτείνει μια συγκεκριμένη κατανομή για τις χρονικές καθυστερήσεις των μεταβάσεων. Οι χρονικές καθυστερήσεις των μεταβάσεων περιγράφονται με κατάλληλες συναρτήσεις πυκνότητας πιθανότητας (probability density functions).

2.5 Δομικά Αναλλοίωτα (P-Invariants / T-Invariants)

Ένα σημαντικό χαρακτηριστικό των Δικτύων Petri είναι ότι οι δομικές τους ιδιότητες μπορούν να ληφθούν από γραμμικές αλγεβρικές τεχνικές. Αυτές οι ιδιότητες που εξαρτώνται μόνο από την δομή της τοπολογίας ενός δικτύου και είναι ανεξάρτητες από την αρχική σήμανση αυτού, καλούνται ***αναλλοίωτα***. Τα αναλλοίωτα είναι ένα σημαντικό μέσο ανάλυσης της συμπεριφοράς ενός δικτύου από δομική οπτική.

Ένα P-διάνυσμα είναι ένα διάνυσμα στήλης I: P→ Z διαμορφούμενο από ένα διάνυσμα θέσης P, και ένα T-διάνυσμα είναι ένα διάνυσμα στήλης J:T→ Z διαμορφούμενο από ένα διάνυσμα μετάβασης T, όπου Z είναι το σύνολο των ακεραίων [14].
Το P-διάνυσμα I ονομάζεται P-αναλλοίωτο εάν και μόνο εάν το I διαφέρει του 0 και $I^T[N]=0^T$, όπου [N] = |P|×|T|
Το T-διάνυσμα J ονομάζεται T-αναλλοίωτο εάν και μόνο εάν το J διαφέρει του 0 και J[N]=0, όπου [N] = |P|×|T|
Επιπλέον, ένα P-αναλλοίωτο καλείται και P- ημι-ροή εάν κάθε στοιχείο του πίνακα I δεν είναι αρνητικό. Ομοίως, ένα T-αναλλοίωτο είναι και T- ημι-ροή εάν κάθε στοιχείο του πίνακα J δεν είναι αρνητικό.

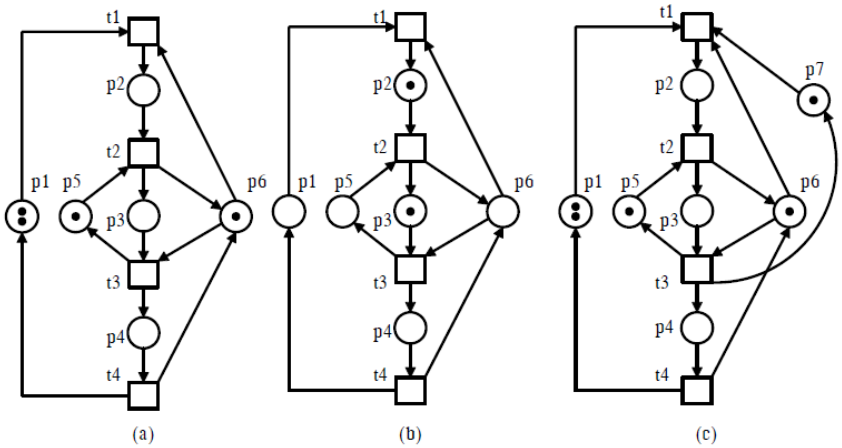
2.6 Σιφώνια (Siphons) και Παγίδες (Traps)

Στην τελευταία ενότητα του παρόντος κεφαλαίου θα αναφερθούμε σε δύο επιπλέον δομικά χαρακτηριστικά που εντοπίζονται ευρέως στη διεθνή βιβλιογραφία σχετικά με την εφαρμογή των Δικτύων Petri και παίζουν πολύ σημαντικό ρόλο στην ανάλυση ζωτικότητας ενός δικτύου.
Τα P-αναλλοίωτα που μπορούν να εξαχθούν από την εξίσωση κατάστασης ενός δικτύου Petri είναι αναλλοίωτα σήμανσης. Ο υπολογισμός των tokens στις αντίστοιχες θέσεις παραμένει σταθερός. Σε ένα δίκτυο Petri ,τα σιφώνια και οι παγίδες είναι επίσης δομικά αντικείμενα που εμπλέκουν αναλλοίωτα σήμανσης. Ωστόσο, οι κανόνες αναλλοίωτων που σχετίζονται με αυτά δεν ισχύουν σε κάθε εφικτή σήμανση, αλλά από τη στιγμή που

αυτοί επαληθεύονται παραμένουν αληθείς σε κάθε επόμενη εφικτή σήμανση. Ένα σιφώνιο παραμένει άδειο όταν χάσει όλα τα tokens. Μία παγίδα παραμένει μαρκαρισμένη όταν διαθέτει τουλάχιστον ένα κουπόνι σε κάποια θέση της.

Ένα μη-κενό σύνολο S που ανήκει στο P είναι ένα σιφώνιο εάν και μόνο εάν το •S ανήκει στο S•. Το S που ανήκει στο P είναι παγίδα αν και μόνο αν το S• ανήκει στο •S, όπου S• και •S είναι τα σύνολά των μεταβάσεων εξόδου και εισόδου αντίστοιχα σε ένα σιφώνιο S.
Ένα σιφώνιο (παγίδα) είναι ελάχιστο αν και μόνο αν δεν υπάρχει άλλο σιφώνιο (παγίδα) που εμπεριέχεται σε αυτό ως κατάλληλο υποσύνολο. Το ελάχιστο σιφώνιο είναι απόλυτο όταν το •S εμπεριέχεται ,αλλά δεν είναι ίσο, του S•.

Η σημασία των σιφωνίων έγκειται στη αναζήτηση ύπαρξης deadlock σε ένα δίκτυο και την γενικότερη ανάλυση ζωτικότητας (liveness) αυτού. Εάν δεν υπάρχει κανένα (ελάχιστο) σιφώνιο που αδειάζει κατά την εξέλιξη των σημάνσεων ενός δικτύου Petri τότε το δίκτυο χαρακτηρίζεται ως deadlock-free.
Για τον λόγο αυτό, στα πλαίσια εφαρμογών του επιστημονικού πεδίου της μηχανικής ελέγχου Συστημάτων Διακριτού Γεγονότος, διατυπωμένα σε δίκτυα Petri, συνήθως αναζητείται η ύπαρξη άδειου σιφωνίου καθώς αυτό το φαινόμενο οδηγεί το σύστημα σε καταστάσεις αδιεξόδου (deadlock), όπως θα δούμε σε επόμενο κεφάλαιο.
Στο ακόλουθο σχήμα απεικονίζονται τρεις διαφορετικές καταστάσεις ενός δικτύου Petri.



Σχήμα 2.8 Σιφώνια και Παγίδες σε Δίκτυα Petri [14]

Στην κατάσταση (a) το σύστημα βρίσκεται στην αρχική σήμανσή του. Στην (b) αναδύεται φαινόμενο deadlock καθώς αδειάζει το σιφώνιο p1,p4,p5,p6 και στην (c) αντιμετωπίζεται το φαινόμενο deadlock με την εφαρμογή κατάλληλης τεχνικής όπως θα δούμε στο τρίτο μέρος της παρούσας εργασίας.

ΚΕΦΑΛΑΙΟ 3: Εργαλεία Προσομοίωσης Δικτύων Petri

3.1 Εισαγωγή

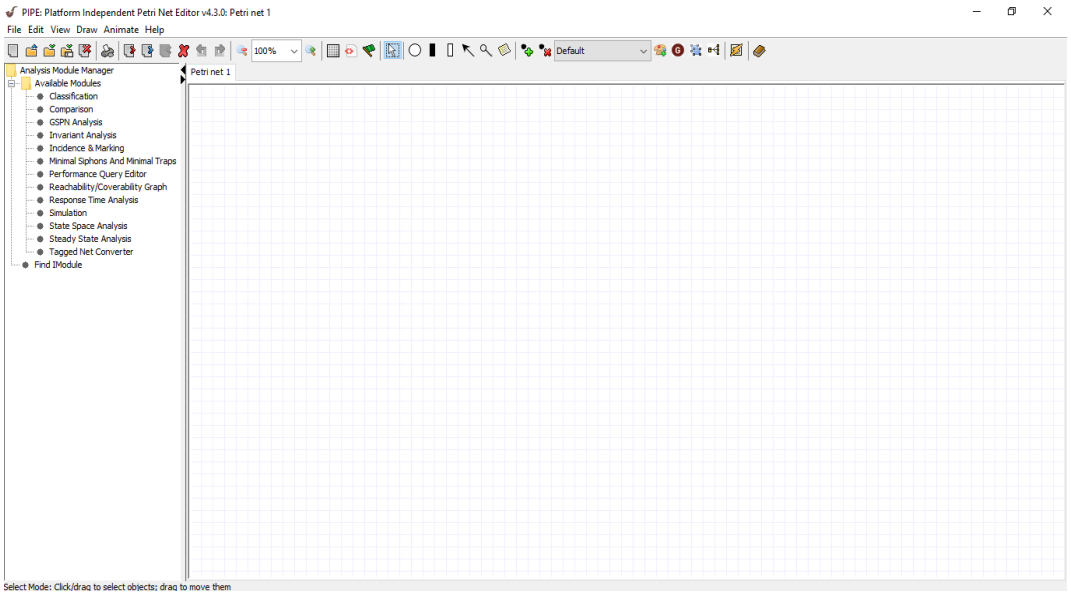
Τα συστήματα που μοντελοποιούνται με δίκτυα Petri μπορούν να προσομοιωθούν με μία πληθώρα υπολογιστικών εργαλείων που έχουν αναπτυχθεί για τους διάφορους τύπους δικτύων (Γενικευμένα, Στοχαστικά, Χρονικά, Αντικειμενοστραφή κ.α.). Έτσι, ο σχεδιαστής και αναλυτής του κάθε συστήματος, έχει τη δυνατότητα να ελέγξει ένα σύστημα ως προς τη δομή, αλλά και να επιβεβαιώσει φαινόμενα που μπορεί να ανακύψουν κατά τη λειτουργία ενός συστήματος, όπως παραλληλία, ταυτοχρονισμός και άλλα. Επιπλέον, μπορεί να γίνει εκτεταμένη διερεύνηση της συμπεριφοράς και των ιδιοτήτων ενός μοντελοποιημένου συστήματος

Στην παρούσα εργασία έγινε χρήση δύο εργαλείων προσομοίωσης των αναπτυχθέντων δικτύων του συστήματος. Τα εργαλεία αυτά επιλέχθηκαν κυρίως ως προς : τη λειτουργικότητα τους, τη σαφήνεια του δικτύου, το χρόνο υπολογισμού (κυρίως σχετικά με την ανάλυση του χώρου καταστάσεων, των αναλλοίωτων, των σιφωνίων, και του γραφήματος προσβασιμότητας) αλλά και της διαθεσιμότητας των απαραίτητων ενοτήτων/επιλογών για την κάλυψη των απαιτήσεων της παρούσας εργασίας.

3.2 PIPE2 (Platform Independent Petri Net Editor 2)

Το PIPE2 είναι ένα open source εργαλείο δημιουργίας και ανάλυσης δικτύων Petri , συμπεριλαμβανομένου και των Γενικευμένων Στοχαστικών Δικτύων Petri. Το συγκεκριμένο εργαλείο ξεκίνησε από μία ομάδα μεταπτυχιακών εργασιών που διεκπεραιώθηκαν στο Τμήμα Πληροφορικής του Imperial College London από το 2002/3 έως το 2006/7.

Λεπτομέρειες σχετικά με την δημιουργία του συγκεκριμένου προγράμματος μπορούν να αντληθούν από την ιστοσελίδα http://pipe2.sourceforge.net/index.html



Σχήμα 3.1 Γραφικό Περιβάλλον PIPE2

Στο εργαλείο προσομοίωσης PIPE2 έχουν ενσωματωθεί η πλειοψηφία των δυνατοτήτων και χαρακτηριστικών άλλων υπαρχόντων εργαλείων Δικτύων Petri, με βελτιωμένες συντομεύσεις αυτών. Επιπλέον, το συγκεκριμένο εργαλείο σχεδιάστηκε ούτως ώστε να είναι πλήρως συμμορφωμένο με τη Petri Net MarkUp Language και να μπορεί έτσι να είναι συμβατό και με άλλα εργαλεία. Το πρόγραμμα αποτελείται από ένα πάνελ Επεξεργασίας/Animation παρέχοντας βασικές λειτουργίες στο σχεδιασμό απλών Δικτύων Θέσεων-Μεταβάσεων και δίνοντας τη δυνατότητα απεικόνισης ενεργοποιήσεων των μεταβάσεων. Συμπληρωματικά, αριστερά του χώρου εργασίας του προγράμματος δίνεται μία λίστα στην οποία αναγράφονται διάφορες ενότητες ανάλυσης των δικτύων. Οι ενότητες δίνουν τη δυνατότητα εξαγωγής αναλλοίωτων, σιφωνίων, προσομοίωσης, ταξινόμησης και ανάλυσης του χώρου καταστάσεων καθώς και εξαγωγής γραφήματος προσβασιμότητας (reachability/coverability graph) συγκεκριμένου μεγέθους.
Στο Toolbar επεξεργασίας των δικτύων φαίνονται όλες οι απαραίτητες ενδείξεις σχημάτων για την κατασκευή δικτύων Petri καθώς και εικονίδια σήμανσης των δυνατοτήτων αποθήκευσης, διαγραφής, μεγέθυνσης/σμίκρυνσης και άλλες δυνατότητες σχετικές αυτών. Επιπλέον, παρέχεται η δυνατότητα δημιουργίας χρωματισμένων δικτύων Petri από τις σχετικές σημάνσεις στα δεξιά του Toolbar. Στο ίδιο σημείο υπάρχουν και τα εικονίδια ομαδοποίησης και από-ομαδοποίησης συγκεκριμένων μεταβάσεων. Οι μεταβάσεις που μπορούν να εισαχθούν σε ένα δίκτυο είναι άμεσες (μαύρες) και χρονισμένες (λευκές), ενώ τα τόξα μεταξύ θέσεων και μεταβάσεων μπορεί να είναι κανονικά αλλά και αποτρεπτικά (inhibitor arcs). Σε κάθε σημείο του δικτύου μπορεί να εισαχθεί λεζάντα για την περιγραφή και επισήμανση μίας συγκεκριμένης πληροφορίας.
Οι επιλογές των ενοτήτων γραφικά μας δίνουν τα παρακάτω ενδεικτικά αποτελέσματα.

Classification (Ταξινόμηση Δικτύου)

Στην ενότητα αυτή το εργαλείο PIPE2 μας δείχνει το είδος του δικτύου που έχουμε αναπτύξει.

Petri net classification results

State Machine	False
Marked Graph	False
Free Choice Net	False
Extended Free Choice Net	False
Simple Net	True
Extended Simple Net	True

GSPN Analysis (Ανάλυση Γενικευμένου Στοχαστικού Δικτύου)

Στην ενότητα αυτή πραγματοποιείται ανάλυση του δικτύου Petri ως προς συγκεκριμένα χαρακτηριστικά του όπως : σύνολο των καταστάσεων του δικτύου, κατανομές σχετικά με τις καταστάσεις του δικτύου και την κατανομή των tokens σε αυτό, εξέλιξη των χρονισμένων μεταβάσεων του δικτύου. Επιπλέον, παίρνουμε συγκεκριμένες πληροφορίες σχετικά με τον μέσο αριθμό token σε κάθε θέση κατά την ενεργοποίηση των μεταβάσεων του δικτύου. Στην περίπτωση όπου το δίκτυό μας δεν εμπεριέχει χρονισμένες μεταβάσεις η συγκεκριμένη ενότητα παραμένει ανενεργή.

GSPN Steady State Analysis Results

Set of Tangible States

	P19	P20	…	P9	P18
M0	4	0		1	0
M1	4	0		0	0
M2	4	0		1	0
M3	4	0		0	0

Steady State Distribution of Tangible States

Marking	Value
M0	0,25
M1	0,25
M2	0,25
M3	0,25

Average Number of Tokens on a Place

Place	Number of Tokens
P19	4
P20	0
…	
P9	0,5
P18	0

Token Probability Density

	μ=0	μ=1	μ=2	μ=3	μ=4	μ=5	μ=6
P19	0	0	0	0	1	0	0
P20	1	0	0	0	0	0	0
…							
P9	0,5	0,5	0	0	0	0	0
P18	1	0	0	0	0	0	0

Throughput of Timed Transitions

Transition	Throughput
T10	0
T14	0
T2	0
T6	0

Sojourn times for tangible states

Marking	Value
M0	∞
M1	∞
M2	∞
M3	∞

State space exploration took 1,079s
Solving the steady state distribution took 0,083s
Total time was 2,211s

Invariant Analysis (Ανάλυση Αναλλοίωτων)

Στην ενότητα αυτή μας δίνονται αναλυτικά τα P- και T- αναλλοίωτα του διαμορφωμένου δικτύου καθώς και οι εξισώσεις των P- αναλλοίωτων. Επιπλέον, μας δίνεται πληροφόρηση σχετικά με το αν το δίκτυο είναι δομικά φραγμένο.

Petri net invariant analysis results

T-Invariants

T0	T1	T2
1	1	1

P-Invariants

P0	P1	P2	P3	P4
1	1	1	0	0
0	0	1	1	0
0	1	3	0	1

P-Invariant equations

M(P0) + M(P1) + M(P2) = 3
M(P2) + M(P3) = 2
M(P1) + 3M(P2) + M(P4) = 3

The net is covered by positive T-Invariants, therefore it might be bounded and live

The net is covered by positive P-Invariants, therefore it is bounded

Analysis time: 0.0s

Incidence Marking (Σήμανση Συμβάντων)

Στην ενότητα αυτή εξάγουμε όλη την εξέλιξη των σημάνσεων του δικτύου και τις σχετικές με αυτή μήτρες συμβάντων (συνδυαστικές και μεμονωμένες).

Forwards incidence matrix

	T0	T1	T2
P0	0	0	1
P1	1	0	0
P2	0	1	0

Backwards incidence matrix

	T0	T1	T2
P0	1	0	0
P1	0	1	0
P2	0	0	1

Combined incidence matrix I

	T0	T1	T2
P0	-1	0	1
P1	1	-1	0
P2	0	1	-1

P3	0	0	1
P4	0	0	3

P3	0	1	0
P4	1	2	0

P3	0	-1	1
P4	-1	-2	3

Inhibition matrix H

	T0	T1	T2
P0	0	0	0
P1	0	0	0
P2	0	0	0
P3	0	0	0
P4	0	0	0

Marking

	P0	P1	P2	P3	P4
Initial	3	0	0	2	3
Current	3	0	0	2	3

Enabled transitions

T0	T1	T2
yes	no	no

Minimal Siphons and Traps (Ελάχιστα Σιφώνια και Παγίδες)

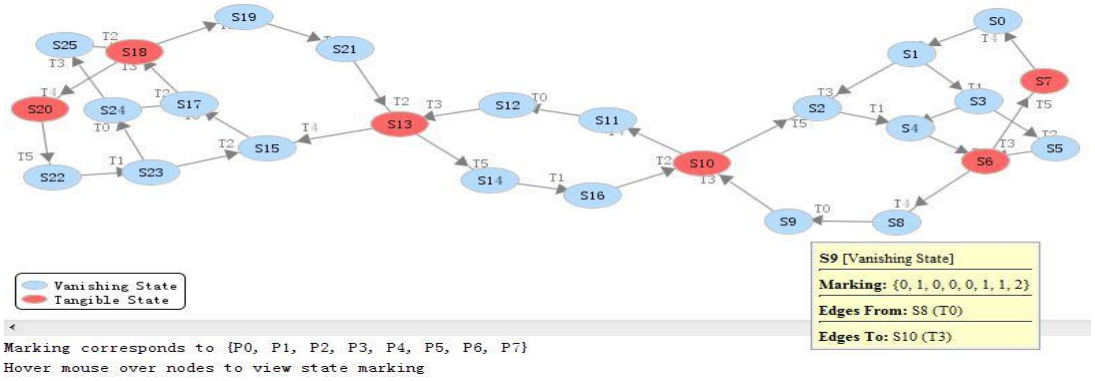
Τα ελάχιστα σιφώνια που μπορεί να υπάρχουν στο δίκτυο και οι ελάχιστες παγίδες εξάγονται από τη συγκεκριμένη ενότητα του εργαλείου PIPE2. Τα σιφώνια που δεν είναι και παγίδες υποδηλώνουν τα ***απόλυτα ελάχιστα σιφώνια***.

Minimal siphons	Minimal traps
{P2, P4 }	{P1, P2, P4 }
{P2, P3 }	{P2, P3 }
{P0, P1, P2 }	{P0, P1, P2 }

Analysis time: 0.003s

Reachability/Coverability Graph

Στην ενότητα αυτή εξάγεται γραφικά το γράφημα Προσβασιμότητας του δικτύου Petri.



Σχήμα 3.2 Γράφημα Προσβασιμότητας

Simulation (Προσομοίωση)

Η επιλογή της ενότητας αυτής θα εξάγει λεπτομερή αναφορά σχετικά με το μέσο αριθμό tokens και το διάστημα εμπιστευτικότητας αυτού σε κάθε θέση του δικτύου.

Petri net simulation results

Place	Average number of tokens	95% confidence interval (+/-)
P0	1,5	0,27999
P1	1,5	0,39199
P2	0	0,112
P3	2	0,112
P4	1,5	0,056

State Space Analysis (Ανάλυση Χώρου Καταστάσεων)

Η ενότητα αυτή είναι που μας δίνει τις σημαντικές πληροφορίες σχετικά με το αν είναι φραγμένο το δίκτυο, συνέπεια των P- αναλλοίωτων από την αντίστοιχη ενότητα, αν είναι ασφαλές και αν παρουσιάζει deadlock.

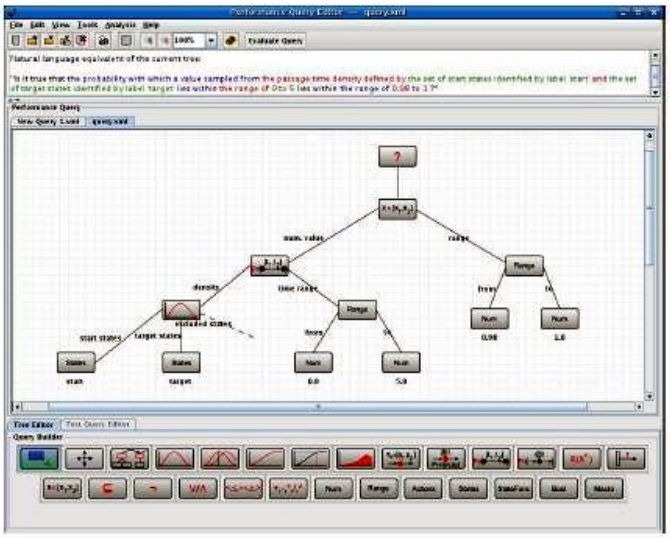
Petri net state space analysis results

Bounded true
Safe false
Deadlock true
Shortest path to deadlock: T0 T1 T8 T9 T0

Τέλος, επιπρόσθετες ενότητες μας δίνουν ειδικά γραφήματα απεικονίσεως (performance trees) της απόδοσης του δικτύου και των σχετικών κατανομών tokens που αναπτύσσονται σε αυτό, αλλά και πιο εξειδικευμένες απεικονίσεις σχετικά με στατιστικά στοιχεία της εξέλιξης σημάνσεως των δικτύων Petri που έχουν αναπτυχθεί (Performance Query Editor).

Textual	Graphical	Description
?		The result of a performance query.
Mult		Concurrent evaluation of multiple independent queries.
PTD		Passage time density, calculated from a given set of start and target states.
Dist		Passage time distribution obtained from a passage time density.
Perctl		Percentile of a passage time density or distribution.
Conv*		Convolution of two passage time densities.
ProbInInterval		Probability with which a passage takes place in a certain amount of time.
ProbInStates*		Transient probability of a system being in a given set of states at a given point in time.
Moment		Raw moment of a passage time density or distribution.
FR		Mean occurrence of an action (mean firing rate of a transition).
SS:P		Probability mass function yielding the steady-state probability of each possible value taken on by a StateFunc when evaluated over a given set of states.
SS:S*		Set of states that have a certain steady-state probability.
StatesAtTime*		Set of states that the system can occupy at a given time.
InInterval		Boolean operator that determines whether a numerical value is within an interval.
Macro*		User-defined performance concept composed of other operators.
⊆		Boolean operator that determines whether a set is included in or corresponds to another set.
∨/∧		Boolean disjunction or conjunction of two logical expressions.
¬		Boolean negation of a logical expression.
⋈		Arithmetic comparison of two numerical values.
⊕		Arithmetic operation on two numerical values.
Num		A real number.
Range		A range of real numbers, defined by a lower and an upper bound.
Bool		A Boolean value.
Actions		A system action.
States		A set of system states.
StateFunc		A real-valued function on a set of states.

Σχήμα 3.3 Επιλογές Διαγραμμάτων Performance Query Editor του PIPE2[8]

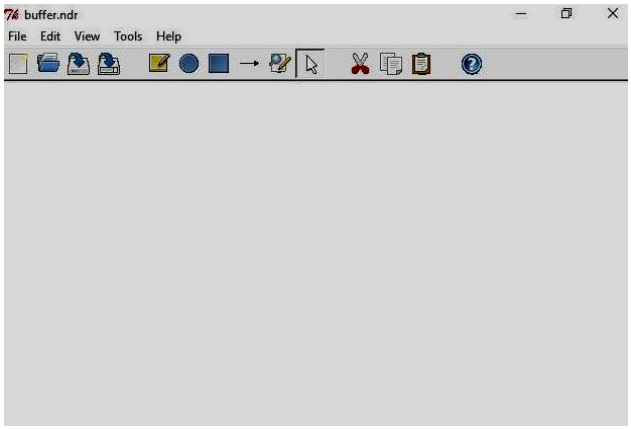


Σχήμα 3.4 Γραφικό Περιβάλλον Performance Query Editor [8]

3.3 Το εργαλείο προσομοίωσης TINA (Time petri Net Analyzer)

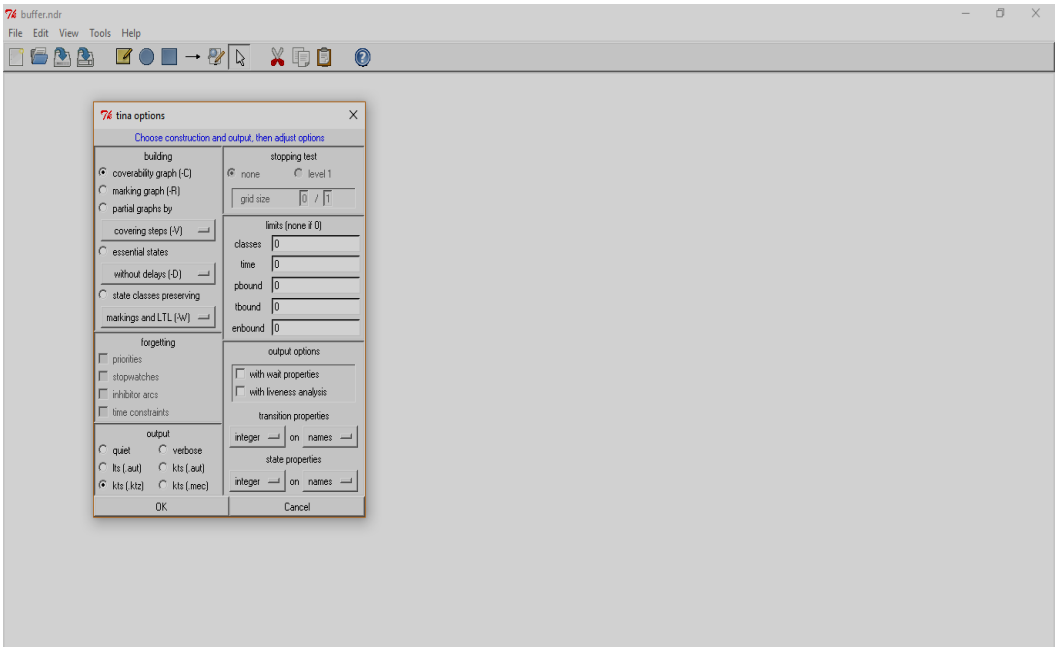
Το εργαλείο προσομοίωσης TINA αναπτύχθηκε από το Εργαστήριο Ανάλυσης και Αρχιτεκτονικής των Συστημάτων του Γαλλικού Εθνικού Κέντρου Επιστημονικής Έρευνας (LAAS – CNRS). Το εργαλείο εστιάζει σε χρονισμένα δίκτυα Petri και δίνει στο χρήστη πολλές επιλογές σχετικά με την προσομοίωση του δικτύου αλλά και την ανάλυση προσβασιμότητας. Το συγκεκριμένο εργαλείο παρέχει δυνατότητα ανάλυσης της ζωτικότητας και αντιστρεψιμότητας του διαμορφωμένου δικτύου Petri και λεπτομερή ανάλυση του χώρου καταστάσεων αυτού. Στα πλαίσια της συγκεκριμένης εργασίας, το εργαλείο TINA χρησιμοποιήθηκε για την επαλήθευση της ζωτικότητας και αντιστρεψιμότητας του δικτύου που αναπτύξαμε στο Τρίτο Μέρος (Κεφάλαιο 7). Από πλευράς γραφικού περιβάλλοντος το εργαλείο TINA είναι αρκετά πιο απλό συγκριτικά με το PIPE2.

Στη γραμμή εργασιών περιέχει όλα τα δομικά στοιχεία των δικτύων Petri και επιλογές διαχείρισης των αρχείων. Στην επιλογή δημιουργίας του μοντέλου, ο χρήστης έχει την δυνατότητα ανάπτυξης χρονισμένου δικτύου Petri αλλά και αυτόματου (automaton). Οι βασικές επιλογές ελέγχου του δικτύου συνοψίζονται στην ανάλυση προσβασιμότητας, δομική ανάλυση, προσομοιωτής βήματος, και έλεγχος σύνταξης του δικτύου. Ενδεικτικά, ο χώρος εργασίας του TINA απεικονίζεται παρακάτω.



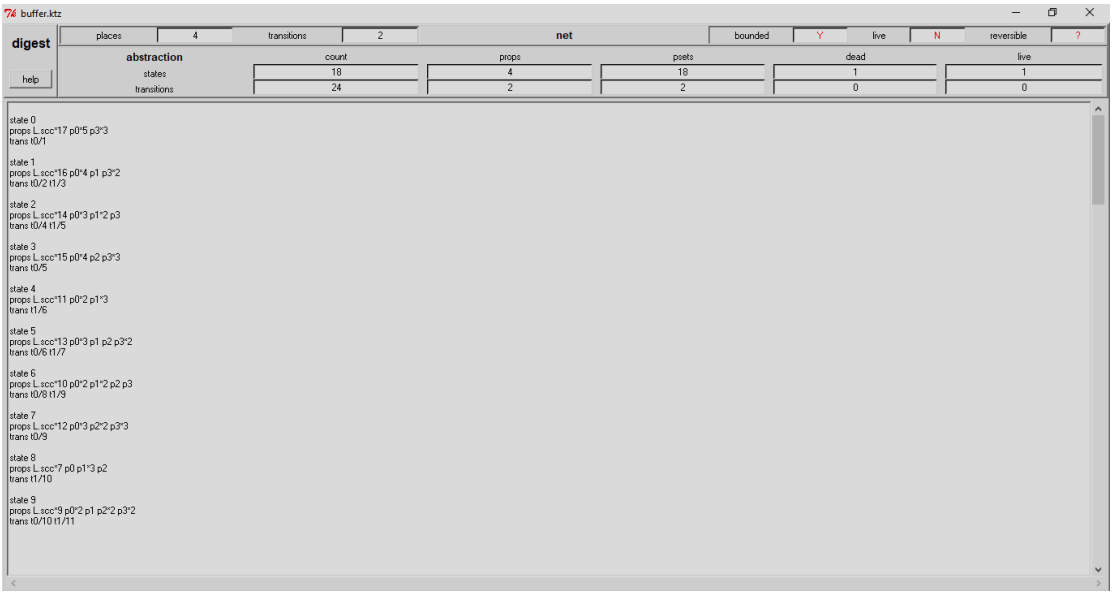
Σχήμα 3.5 Γραφικό Περιβάλλον εργαλείου προσομοίωσης TINA

Οι επιλογές σχετικά με την εξαγωγή της ανάλυσης προσβασιμότητας απεικονίζεται στο ακόλουθο παράθυρο της εφαρμογής.



Σχήμα 3.6 Παράθυρο Επιλογών Ανάλυσης Προσβασιμότητας

και το τελικό παράθυρο εξαγωγής αυτής είναι το ακόλουθο:



Σχήμα 3.7 Εξαγωγή Ανάλυσης Προσβασιμότητας στο TINA toolbox

Στο δεξί πάνω μέρος του παράθυρου αντλούμε πληροφορίες σχετικά με τον περιορισμό/φραγή, την ζωτικότητα και αντιστρεψιμότητα του δικτύου (http://projects.laas.fr/tina/).

ΜΕΡΟΣ ΔΕΥΤΕΡΟ

ΚΕΦΑΛΑΙΟ 4: Συστήματα Πολλαπλών Πρακτόρων

4.1 Εισαγωγή

Οι πράκτορες λογισμικού (software agents) προσφέρουν μία καινούργια, καινοτόμο προσέγγιση στο σχεδιασμό και τη δημιουργία πολύπλοκων κατανεμημένων συστημάτων και επεκτείνουν σημαντικά προηγούμενες προσεγγίσεις όπως αντικειμενοστραφή ή κατανεμημένο υπολογισμό. Αντί να μοντελοποιούνται τα κατανεμημένα συστήματα σαν λογισμικά προγράμματα που ανταλλάζουν δεδομένα και εντολές, η τεχνολογία πολλαπλών πρακτόρων δημιουργεί αυτόνομες οντότητες λήψης αποφάσεων που επικοινωνούν για τις προτιμήσεις τους, διαπραγματεύονται μερικούς στόχους και συντονίζουν τους σκοπούς τους για να επιτύχουν το στόχο του συστήματος. Η προσέγγιση υπολογισμού τους, βασισμένη σε απόφαση και αλληλεπίδραση, κάνει εφικτή τη δημιουργία συστημάτων που μπορούν να αντιδρούν δυναμικά σε απρόβλεπτα γεγονότα, να ενσωματώνουν διαφορετικές προτιμήσεις και συμπεριφορές, να εκμεταλλεύονται διαφορετικές δυνατότητες των στοιχείων τους και να προσαρμόζονται σε αλλαγές του περιβάλλοντος τους [3]. Η ικανότητα των πρακτόρων να προσαρμόζουν τη συμπεριφορά τους κατά τον υπολογισμό μειώνει τις ανάγκες ενός σχεδιαστή συστήματος να λαμβάνει υπόψη όλα τα πιθανά σενάρια και αλλαγές που μπορεί να συναντήσει. Επιπλέον, ένας σχεδιασμός συστήματος βασισμένο σε πράκτορες ταιριάζει απόλυτα στην κατανεμημένη φύση της λήψης αποφάσεων σε πολλούς τομείς εφαρμογής και έτσι αυξάνει την κατανόηση και ικανότητα διατήρησης ενός λογισμικού συστήματος.
Τα πλεονεκτήματα της τεχνολογίας πρακτόρων έχουν επίσημα αναγνωριστεί και έχουν οδηγήσει σε μελέτες εφαρμογών διαφορετικών τομέων. Κάποιες εφαρμογές που έχουν αναφερθεί περιλαμβάνουν βιομηχανικές εφαρμογές (έλεγχος διεργασιών, έλεγχος εναέριας κυκλοφορίας), εμπορικές εφαρμογές (διαχείριση πληροφορίας, ηλεκτρονικό εμπόριο, διαχείριση επιχειρησιακών διεργασιών), ιατρικές εφαρμογές (παρακολούθηση ασθενών κ.α.) και διασκέδαση (παίγνια, θέατρο, σινεμά). Ένας ακόμα τομέας όπου τα τελευταία έτη είναι ‘στόχος’ πολλών εφαρμογών πρακτόρων είναι ο έλεγχος παραγωγής. Η συγκεκριμένη περιοχή εφαρμογών είναι εξαιρετικά σημαντική από πλευράς βιομηχανικής δραστηριότητας και συνεισφοράς στην εγχώρια οικονομία μίας βιομηχανικής χώρας. Επιπλέον, τα συστήματα παραγωγής είναι από τη φύση τους κατανεμημένα με δυναμική φύση αντιμετωπίζοντας πολλές αλλαγές και διαταραχές κατά τη λειτουργία τους.

4.2 Τεχνολογία Πρακτόρων Λογισμικού (Software Agents)

Η τεχνολογία πρακτόρων λογισμικού είναι ένας πολύ ενεργός τομέας έρευνας ξεκινώντας στις αρχές τις δεκαετίας του 1980 σαν υπο-περιοχή της Τεχνητής Νοημοσύνης, ευρέως γνωστή ως Κατανεμημένη Τεχνητή Νοημοσύνη. Το κομμάτι αυτό της Τεχνητής Νοημοσύνης εστιάζει στο σχεδιασμό Πολλαπλών Πρακτόρων (multi - agent planning) και στην Κατανεμημένη Επίλυση Προβλήματος (distributed problem solving) (Steffan Bussman, 2004). Η έρευνα των συστημάτων πολλαπλών πρακτόρων ασχολείται με το πώς θα μοντελοποιηθούν και θα εφαρμοστούν πρότυπα ατομικής και κοινωνικής συμπεριφοράς σε κατανεμημένα συστήματα. Όροι, όπως αυτονομία, αντιδραστικότητα, συντονισμός, συνεργασία, διαπραγμάτευση, σύγκρουση, ανάθεση ρόλου και αυτό-οργάνωση, είναι στον πυρήνα της έρευνας αυτής της κατηγορίας συστημάτων που

αναζητά τρόπους να εφαρμοστούν αποτελεσματικά η ατομική και κοινωνική συμπεριφορά στα προς μελέτη συστήματα.
Γενικότερα, ως πράκτορας θα μπορούσε να οριστεί μία διεργασία λογισμικού με τις ακόλουθες ιδιότητες:

Αυτονομία → δράσεις ανεξάρτητες της μεσολάβησης ανθρώπων και έλεγχος πάνω σε αυτές

Κοινωνική Ικανότητα → οι πράκτορες αντιδρούν μεταξύ τους (μέσω μίας συγκεκριμένης γλώσσας επικοινωνίας αυτών)

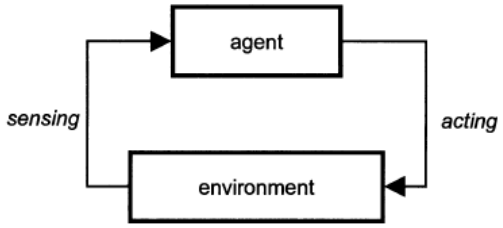
Αντιδραστικότητα → οι πράκτορες αντιλαμβάνονται το περιβάλλον τους και αντιδρούν σε αλλαγές που συμβαίνουν σε αυτό

Προ-δραστικότητα → οι πράκτορες πέραν της αντίληψης και ανταπόκρισης στο περιβάλλον τους, επιπλέον είναι ικανοί να αποκτήσουν συμπεριφορά σκοπού με συγκεκριμένο κίνητρο.

Οι ομάδες αλληλεπιδρώντων πρακτόρων συντελούν Σύστημα Πολλαπλών Πρακτόρων (Multi Agent Systems – MAS).

4.3 Μοντέλα Πρακτόρων

Στην ενότητα αυτή θα αναφερθούν συγκεκριμένες τεχνικές μοντελοποίησης και εφαρμογής Συστημάτων Πολλαπλών Πρακτόρων (MAS).
Βασισμένοι στην αυτονομία ενός πράκτορα, μπορούμε να πούμε ότι σε ένα δεδομένο περιβάλλον ένας πράκτορας αισθάνεται και αντιδρά. Στο ακόλουθο σχήμα απεικονίζεται η συγκεκριμένη ιδιότητα.



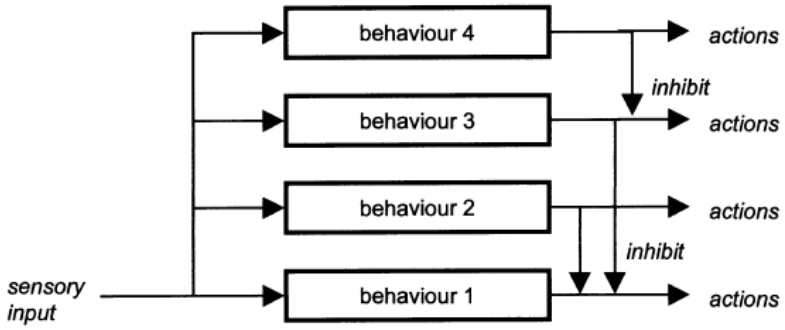
Σχήμα 4.1 Δομή Πράκτορα [3]

Η ιδιότητα της αυτόνομης δράσης, μας οδηγεί στην ακόλουθη κατηγοριοποίηση των πρακτόρων με βάση τις επιλογές δράσεων αυτών. Οι τρεις βασικοί τύποι αρχιτεκτονικής πρακτόρων είναι :

1. Reactive agents (Αντιδρώντες)
2. Deliberative Agents (Συμβουλευτικοί)
3. Hybrid Agents (Υβριδικοί)

Reactive agents

Στην κατηγορία αυτής της αρχιτεκτονικής πρακτόρων, η αισθητηριακή είσοδος συνδέεται με τις ικανότητες δράσης ενός πράκτορα. Δηλαδή, για κάθε πιθανή είσοδο πρέπει να ορίζεται ποια δράση θα εκτελεστεί άμεσα. Κατά τη διάρκεια της εκτέλεσης, ο πράκτορας χρειάζεται επαναλαμβανόμενα μόνο την αίσθηση εισόδου ώστε να την ταιριάξει με τις συνθήκες της κάθε δράσης. Το πρόβλημα που ανακύπτει είναι όταν πολλές συνθήκες ταιριάξουν με την ίδια είσοδο δημιουργώντας **σύγκρουση δράσεων**.
Το συγκεκριμένο πρόβλημα προσεγγίζεται επιτυχώς με την αρχιτεκτονική 'Περίληψης' . Κάθε συνθήκη-συμπεριφορά σχετίζεται με μία λίστα, χαμηλής προτεραιότητας, συμπεριφορών που αυτή απαγορεύει . Κάθε συμπεριφορά ενεργοποιείται όταν δεν υπάρχει άλλη που την απαγορεύει όπως φαίνεται στο σχήμα.



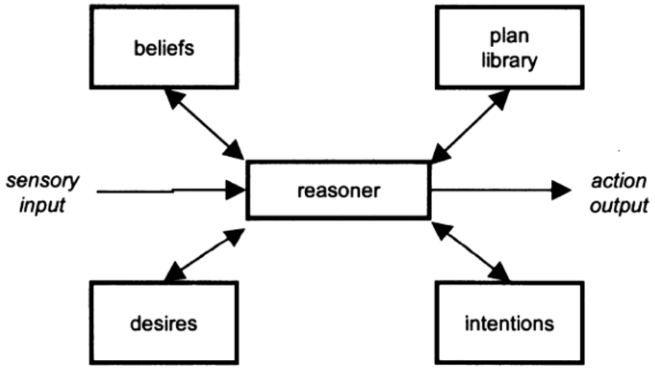
Σχήμα 4.2 Αρχιτεκτονική Περίληψης [3]

Βασικό πλεονέκτημα αυτής της αρχιτεκτονικής είναι η εξασφάλιση της αντιδραστικότητας. Από την άλλη, είναι δύσκολη η εφαρμογή προ-δραστικότητας και συμπεριφοράς στόχου, καθώς οι πράκτορες εστιάζουν μόνο στην τρέχουσα κατάσταση και αντιδρούν , αν και μόνο αν, συμβεί κάποια αλλαγή στο περιβάλλον τους.

Deliberative Agents

Οι αρχιτεκτονικές συμβουλευτικών πρακτόρων αναπαριστούν στόχους αλλά και σχέδια για το πώς ένας πράκτορας θέλει να συμπεριφερθεί στο μέλλον για να επιτύχει αυτούς τους στόχους. Πιθανότατα η επικρατέστερη αρχιτεκτονική πρακτόρων αυτής της κατηγορίας είναι η αρχιτεκτονική BDI (belief-desire-intention). Στη BDI αρχιτεκτονική ο πράκτορας λαμβάνει τις πεποιθήσεις του (αισθητηριακή είσοδος συσσωρευμένη στο χρόνο), τις επιθυμίες του (στόχους του) και σχηματίζει τους σκοπούς του σχετικά με το τι θα κάνει στο μέλλον. Οι σκοποί θεωρούνται 'μαθήματα ενεργειών' που δεσμεύει ο πράκτορας, εκτελούμενες μέχρι να ανακύψουν συγκεκριμένες καταστάσεις που θα τον αναγκάσουν να τις εγκαταλείψει.
Εν αντιθέσει με τους αντιδρώντες πράκτορες, οι BDI πράκτορες είναι ικανοί να ακολουθούν τους στόχους τους προ-δραστικά και παράλληλα να αντιδρούν και με το περιβάλλον. Στην περίπτωση που οι πεποιθήσεις που λαμβάνονται από την είσοδο δεν υποστηρίζουν πια ένα σκοπό, τότε αυτός είτε επιδέχεται αλλαγή ή εγκαταλείπεται και έτσι η συμπεριφορά του πράκτορα προσαρμόζεται. Η συγκεκριμένη αρχιτεκτονική μπορεί να εφαρμοστεί σε αυτόνομους, προ-δραστήριους και αντιδρώντες πράκτορες. Το βασικό μειονέκτημα που συναντάμε εδώ είναι ότι οι πράκτορες αυτής της αρχιτεκτονικής αντιδρούν μόνο αφού η πληροφορία από την είσοδο περάσει όλα τα αναγκαία

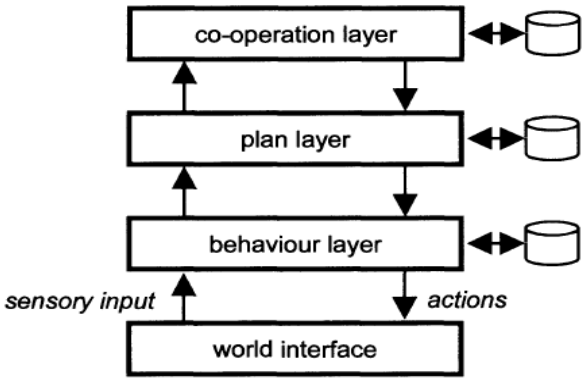
διαφορετικά βήματα για να σχηματιστεί ένας σκοπός (σχηματισμός πεποίθησης και προσαρμογής των σκοπών πριν τις εκτελούμενες ενέργειες). Όσο αυξάνει η πολυπλοκότητα των πρακτόρων, αυτή η διαδικασία αυξάνει το χρόνο εκτέλεσης των απαιτούμενων ενεργειών.



Σχήμα 4.3: BDI αρχιτεκτονική [3]

Hybrid Agents

Οι υβριδικές αρχιτεκτονικές πρακτόρων ενσωματώνουν τους μηχανισμούς αντιδρώντων και συμβουλευτικών αρχιτεκτονικών. Χαρακτηριστικό παράδειγμα αρχιτεκτονικής είναι η InteRRaP που αποτελείται από τα εξής τρία στρώματα. Ένα στρώμα συμπεριφοράς για κανόνες ενεργειών αντιδρώντων πρακτόρων, ένα στρώμα κατευθυνόμενων στόχων προ-δραστήριου σχεδιασμού και ένα στρώμα συνεργασίας για τις λειτουργίες μοντελοποίησης και χειρισμού αλληλεπιδράσεων με άλλους πράκτορες. Το σήμα εισόδου περνάει πρώτα στο στρώμα συμπεριφοράς. Εάν κάποιος κανόνας ενέργειας είναι εφαρμόσιμος στην είσοδο στη φάση αυτή τότε ενεργοποιείται, αλλιώς το σήμα εισόδου περνάει στο επόμενο στρώμα. Όταν ένα συγκεκριμένο στρώμα επιλέξει κάποια ενέργεια τότε η πληροφορία αυτή κατεβαίνει ιεραρχικά σε χαμηλότερα επίπεδα εκτέλεσης με συνέπεια το στρώμα συμπεριφοράς να είναι εκείνο που εκτελεί στο τέλος τις αντίστοιχες ενέργειες.
Οι υβριδικές αρχιτεκτονικές καλύπτουν όλους τους τύπους πρακτόρων. Ένας υβριδικός πράκτορας μπορεί να είναι αυτόνομος, αντιδρών, προ-δραστήριος και σε πολλές περιπτώσεις να εκδηλώνει κοινωνική συμπεριφορά. Μοναδικό μειονέκτημα είναι η δυσκολία συντονισμού των διαφορετικών στρωμάτων κατά τη διάρκεια του σχεδιασμού.



Σχήμα 4.4 Υβριδική Αρχιτεκτονική Πρακτόρων [3]

4.4 Αλληλεπίδραση Πρακτόρων

Η δεύτερη στοιχειώδης ιδιότητα ενός πράκτορα είναι η ικανότητα του να αντιδρά με τους άλλους πράκτορες. Για να συναντήσει τους στόχους του ένας πράκτορας θα πρέπει να αποφεύγει αρνητικές επιδράσεις ή εναλλακτικά να μπορεί να εκμεταλλευθεί άλλες με άλλους πράκτορες. Πιο συγκεκριμένα, κάποιοι στόχοι συγκεκριμένης πολυπλοκότητας μπορούν να επιτευχθούν από ένα σύνολο πρακτόρων που δουλεύουν συνεργατικά και όχι από έναν που δρα μόνος του. Γενικότερα η έννοια της αλληλεπίδρασης μπορεί να ερμηνευτεί σαν κάθε είδος ανταλλαγής πληροφορίας που επηρεάζει τις ενέργειες άλλων πρακτόρων. Οι δύο βασικοί τύποι αλληλεπίδρασης είναι ο **συντονισμός και η διαπραγμάτευση**.

Συντονισμός

Συντονισμός είναι η διαδικασία κατά την οποία οι πράκτορες επιβεβαιώνουν ότι το σύνολό τους ενεργεί με ένα καλά ορισμένο τρόπο . Το πρόβλημα του συντονισμού γίνεται εξαιρετικά δύσκολο σε ένα σύστημα πολλαπλών πρακτόρων όπου δεν υπάρχει κεντρικός έλεγχος και κάθε πράκτορας ενεργεί αυτόνομα επιλέγοντας τις ενέργειές του.
Ο συντονισμός εν γένει είναι εξαιρετικά δύσκολο να επιτευχθεί για τους ακόλουθους λόγους :

1. Οι ενέργειες του πράκτορα μπορεί να συγκρούονται. Για παράδειγμα δύο ρομπότ μπορεί να συγκρούονται για το ίδιο τεμάχιο.
2. Μπορεί ολικοί περιορισμοί να πρέπει να καλυφθούν. Για παράδειγμα, η επεξεργασία ενός τεμαχίου σε διαφορετικές μηχανές μπορεί να πρέπει να προγραμματιστεί έτσι ώστε το τεμάχιο να τελειώσει σε συγκεκριμένες ημερομηνίες.
3. Κανένας πράκτορας να μην έχει επαρκείς δυνατότητες ή πόρους για να επιτύχει τους στόχους του συστήματος. Για παράδειγμα, μία μηχανή συνήθως δεν είναι επαρκής να εκτελέσει όλες τις λειτουργίες επεξεργασίας ενός τεμαχίου.

Όλες οι παραπάνω περιπτώσεις, έχουν σαν κοινό την ύπαρξη ενός είδους **εξάρτησης** μεταξύ των πρακτόρων. Οι εξαρτήσεις αυτές με την σειρά τους έχουν κατηγοριοποιηθεί στη διεθνή βιβλιογραφία ως εξής :
Μονομερής εξάρτηση : ένας πράκτορας εξαρτάται από έναν άλλο αλλά όχι αντίστροφα.
Κοινή εξάρτηση : Δύο πράκτορες εξαρτώνται ο ένας από τον άλλο για τον ίδιο σκοπό.
Αμοιβαία εξάρτηση : Ίδια με την Κοινή αλλά εδώ οι πράκτορες εξαρτώνται για διαφορετικό σκοπό.

Επιπροσθέτως , από **οργανωτική σκοπιά** (βασισμένοι σε μοντέλο εργασίας/πόρου) ,οι εξαρτήσεις μπορούν να ομαδοποιηθούν και ως εξής :

Κατάλληλη εξάρτηση : πολλαπλές δραστηριότητες επιλεκτικά παράγουν ένα πόρο
Εξάρτηση Ροής : μία δραστηριότητα παράγει πόρο για άλλη δραστηριότητα.
Κοινή εξάρτηση : δύο η περισσότερες δραστηριότητες χρησιμοποιούν τον ίδιο πόρο.

Πολλές έρευνες έχουν εστιάσει κατά το παρελθόν στη διαχείριση των αναδυόμενων **σχέσεων στόχου και σχεδιασμού** που αναδύονται στα συστήματα πολλαπλών πρακτόρων. Μία βασική προσέγγιση διαχείρισης τέτοιων σχέσεων είναι ο Μερικός Ολικός Σχεδιασμός (Partial Global Planning) [1].Σε αυτή την προσέγγιση, κατανεμημένοι πράκτορες που σχεδιάζουν, συντονίζουν τις ενέργειές τους αφαιρώντας

στοιχεία από τα σχέδια τους και στη συνέχεια ανταλλάζοντας αυτά με άλλους. Δίνοντας τις τοπικές αφαιρέσεις σχεδίου, ο κάθε πράκτορας μπορεί να αναγνωρίσει κοινούς στόχους στους οποίους συνεισφέρουν οι τοπικοί στόχοι αυτών. Από την ώρα που οι κοινοί στόχοι είναι μερικώς γνωστοί από τους πράκτορες, καλούνται μερικοί-ολικοί στόχοι.
Τέλος, σχετικές έρευνες για τους **μηχανισμούς συντονισμού** των πρακτόρων έχουν αναδείξει συγκεκριμένες τεχνικές όπως :

- Αμοιβαία Απόκλιση βασισμένη σε Πλειοδοσία
- Κατανεμημένη Επίλυση Προβλήματος
- Αλγόριθμοι Αναζήτησης
- Κοινωνικοί Νόμοι

Διαπραγμάτευση

Διαπραγμάτευση είναι η διαδικασία με την οποία μία κοινή απόφαση εξάγεται μεταξύ δύο ή περισσότερων μερών. Τα μέρη πρώτα συζητάνε για συγκρουόμενες απαιτήσεις και μετά προχωράνε σε συμφωνία από μία διαδικασία εκχώρησης ή αναζήτησης νέων εναλλακτικών.
Στα συστήματα λογισμικού, μεταξύ των αυτόνομων πρακτόρων συχνά προκύπτουν συγκρούσεις. Ένα από τα βασικά αντικείμενα της έρευνας πολλαπλών πρακτόρων είναι η ενεργοποίηση των πρακτόρων στην διεξαγωγή διαπραγμάτευσης για να αποφευχθούν οι ανακύπτουσες συγκρούσεις. Οι επικρατέστερες ομάδες μεθόδων για διαπραγμάτευση βασίζονται :

- στη θεωρία παιγνίων (game based negotiation)
- σε δημιουργία πλάνων
- στην κατανεμημένη ικανοποίηση περιορισμών (distributed constraint satisfaction)

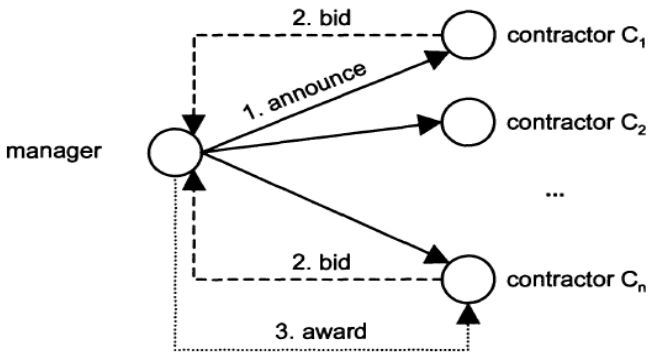
Εν συνεχεία, οι δημοφιλέστερες ομάδες πρωτοκόλλων διαπραγμάτευσης είναι :

- Auctions (Δημοπρασίες)
- Μηχανισμοί Ισοζυγίου Αγοράς
- Τεχνικές Επίλυσης Συγκρούσεων

Οι δημοπρασίες είναι μηχανισμοί εμπορίου για την ανταλλαγή εμπορευμάτων. Διακρίνονται σε μονόπλευρες , δίπλευρες και συνεχείς δημοπρασίες, ενώ για την διεκπεραίωσή τους υπάρχει ένα πιθανό σύνολο πρωτοκόλλων δημοπρασίας. Τα κύρια πρωτόκολλα δημοπρασίας είναι τα : English Auctions, Continuous Double Auctions, και το Contract Net Protocol με βάση και το οποίο θα αναπτυχθεί το μοντέλο της παρούσας εργασίας[3].

Πρωτόκολλο Contract Net (CNP)

Το συγκεκριμένο πρωτόκολλο παραδοσιακά εμπλέκεται κυρίως στην έρευνα σχετικά με τη Συνεργατική Κατανεμημένη Επίλυση Προβλήματος (Cooperative Distributed Problem Solving- CDPS). Στο CNP υπάρχουν δύο τύποι πρακτόρων : οι διαχειριστές (managers) και οι πλειοδότες (bidders). Εργολάβοι μπορούν να χαρακτηριστούν οι πιθανοί πλειοδότες. Ο διαχειριστής ξεκινάει το πρωτόκολλο και το συνεχίζει ως ακολούθως : Ανακοινώνει την εργασία στους ενδεχόμενους εργολάβους (contractors). Ο εργολάβος απαντάει με μία προσφορά. Η αναγκαία πληροφορία της προσφοράς ορίζεται από το διαχειριστή στο μήνυμα ανακοίνωσης. Ο διαχειριστής συγκρίνει τις προσφορές και επιλέγει την καλύτερη σύμφωνα με τις προτιμήσεις του. Ο εργολάβος που έστειλε την καλύτερη προσφορά λαμβάνει μήνυμα δημιουργίας συμβολαίου με το διαχειριστή για την εκτέλεση συγκεκριμένης εργασίας. Το πρωτόκολλο CNP είναι αδιάφορο για το είδος της εργασίας ή του αγαθού που ανταλλάζεται. Επιπλέον, σταματάει αφού ο εκτελών τη δημοπρασία λαμβάνει την πρώτη προσφορά από κάθε πλειοδότη, αντί να επαναλαμβάνει τη δημοπρασία μέχρις ότου κανένας πλειοδότης να μην αλλάζει προσφορές. Τέλος, η προσφορά είναι ‘κρυφή’ στους άλλους πλειοδότες.



Σχήμα 4.5 Contact Net Protocol (Steffan Bussman, 2004)

Οι τεχνικές σύναψης συμβολαίων του συγκεκριμένου πρωτόκολλου μπορούν να συνοψιστούν στα εξής :

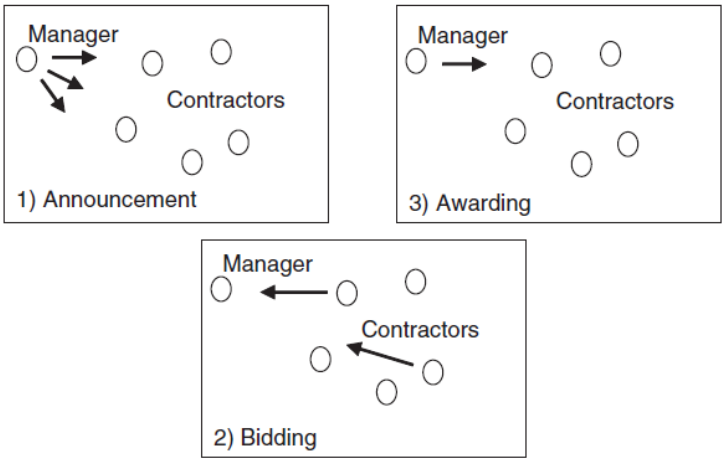
Ο διαχειριστής (manager) :

1. Χωρίζει το πρόβλημα σε υποπροβλήματα
2. Αναλαμβάνει να τα αναθέσει στους εργολάβους (contractors)
3. Επιβλέπει την πορεία της λύσης

Ο εργολάβος αναλαμβάνει να λύσει ένα υπο-πρόβλημα. Οι εργολάβοι μπορούν να χωρίσουν το πρόβλημα σε περισσότερα υπο-προβλήματα και να το αναθέσουν σε άλλους.
Τέλος, επισημαίνεται ότι κάθε πράκτορας μπορεί να είναι ταυτόχρονα διαχειριστής και εργολάβος.
Η διαδικασία ανάθεσης των υπο-προβλημάτων συνοπτικά περιλαμβάνει:

- Τη δημοσιοποίηση τους
- Αξιολόγηση από τους αποδέκτες των δημοσιοποιημένων υπο-προβλημάτων
- Αποστολή των αποδεκτών στο διαχειριστή είτε προσφορών (bids) είτε απορρίψεων (declination)

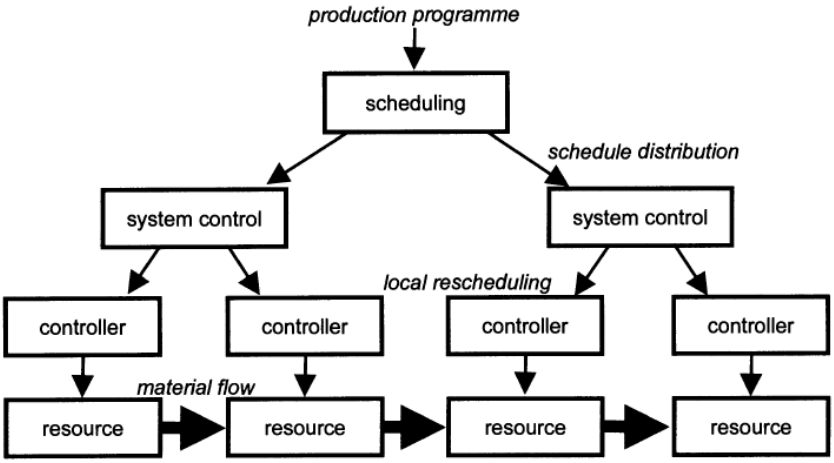
- Συλλογή προσφορών από το διαχειριστή
- Αξιολόγηση προσφορών
- Κατάλληλες αναθέσεις (awards)



Σχήμα 4.6: Φάσεις του Πρωτοκόλλου Contract Net [2]

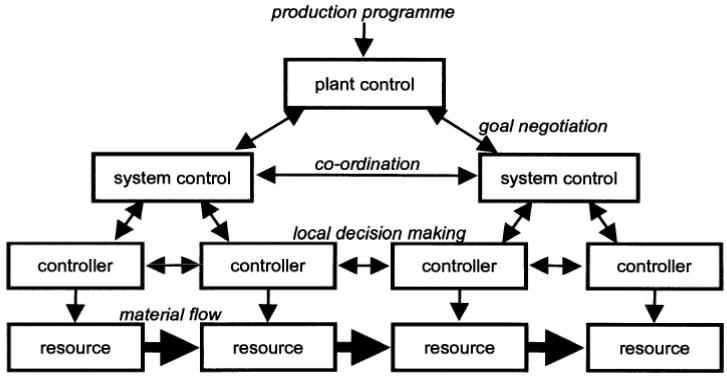
4.5 Τεχνολογία Συστημάτων Πολλαπλών Πρακτόρων στον Έλεγχο Παραγωγής

Η κλασική προσέγγιση ελέγχου παραγωγής χαρακτηρίζεται καλύτερα ως ιεραρχική και βασισμένη σε χρονοδιάγραμμα. Πρώτα από όλα, τα συστήματα ελέγχου οργανώνονται σε μία ιεραρχία εντολών όπου υφιστάμενες μονάδες υποστηρίζονται μόνο από τα υφιστάμενα επίπεδα. Κάθε επίπεδο ιεραρχίας δημιουργεί προγράμματα για τις υφιστάμενες μονάδες με ελάχιστη ανάδραση από τα χαμηλότερα επίπεδα. Μία προσαρμογή του προγράμματος δύναται να πραγματοποιηθεί μόνο από κάποια γειτονική ή ανώτερη μονάδα. Σφάλματα που προκύπτουν διορθώνονται διαμέσου της εφαρμογής του επόμενου κύκλου προγραμματισμού. Με αυτή τη δομή, στην περίπτωση διαταραχής του συστήματος, ένας ελεγκτής δεν είναι ικανός να εκτελέσει τις ενέργειές του. Από την άλλη, από την στιγμή που οι λειτουργίες παραγωγής βελτιστοποιούνται για την αύξηση της παραγωγικότητας και την ελαχιστοποίηση του κόστους, οι χωρητικότητες των διαθέσιμων πόρων χρησιμοποιούνται πλήρως και τα μεγέθη των ενδιάμεσων αποθηκών μειώνονται στο ελάχιστο δυνατό. Σαν συνέπεια όλων αυτών, οποιαδήποτε απόκλιση από το πρόγραμμα επηρεάζει άμεσα τις γειτονικές μονάδες προκαλώντας cascading φαινόμενα της διαταραχής. Η έλλειψη system-wide επανα-προγραμματισμού καθιστά αδύνατο τον περιορισμό των διαταραχών.



Σχήμα 4.7 Κλασική Προσέγγιση Ελέγχου [3]

Για να προσπελαστούν τέτοιου είδους προβλήματα που δημιουργεί η κλασική προσέγγιση ελέγχου θα πρέπει η κάθε μονάδα να αποκτήσει την ελευθερία να επιλέγει τις κατάλληλες δράσεις ανάλογα με την πρόσφατη κατάσταση που βρίσκεται. Το πρόγραμμα παραγωγής θα πρέπει να αποσυντεθεί και να κατανεμηθεί με την μορφή υπο-στόχων στις υφιστάμενες μονάδες του συστήματος. Η επιδίωξη των υπο - στόχων των μονάδων και η συνεργασία μεταξύ αυτών για την ολοκλήρωση τους, οδηγεί τελικά στην ολοκλήρωση του προγράμματος παραγωγής. Γενικότερα, η στοχο-καθοδηγούμενη και συνεργατική προσέγγιση στον έλεγχο παραγωγής μας οδηγεί στην εξασφάλιση της απαραίτητης ευρωστίας, ευελιξίας και επανα-συγκρότησης της παραγωγικής διαδικασίας. Ο ελεγκτής που σχεδιάζεται με την τελευταία ευέλικτη προσέγγιση θα μπορεί να αντιληφθεί από πού προέρχονται οι οποιεσδήποτε αλλαγές και έτσι να αντιμετωπίσει τις διαταραχές, με την αποτελεσματική διαχείριση αυτών.



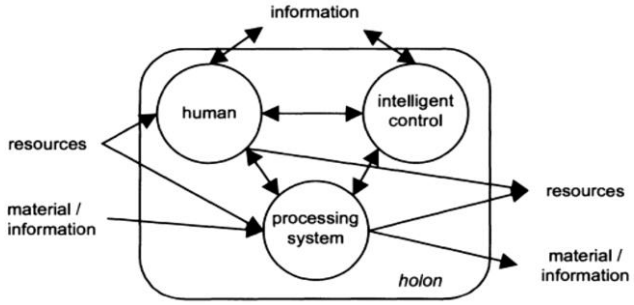
Σχήμα 4.8 Στοχο – Καθοδηγούμενη Προσέγγιση Ελέγχου [3]

Οι καινούργιες απαιτήσεις σχεδιασμού συστημάτων ελέγχου παραγωγής που προκύπτουν από την τελευταία προσέγγιση είναι οι ακόλουθες :

- Εξασφάλιση της κατανομής ελέγχου στα φυσικά στοιχεία του συστήματος παραγωγής
- Ένας τοπικός ελεγκτής θα πρέπει να εφοδιαστεί με ικανότητες λήψης απόφασης αντίδρασης και κατεύθυνσης στόχου
- Οι τοπικοί ελεγκτές θα πρέπει να συνεργάζονται με ευέλικτο τρόπο

Συνοψίζοντας, μπορούμε να συμπεράνουμε ότι ο καινούργιος καινοτόμος ελεγκτής θα πρέπει να παρουσιάζει στοιχεία αυτονομίας και συνεργασίας στη λήψη των αποφάσεων που καλείται να εκτελέσει.

Κλείνοντας αυτή την ενότητα, αναφέρουμε συνοπτικά τις επικρατούσες τάσεις στην έρευνα της προσέγγισης ελέγχου συστημάτων παραγωγής. Αυτές είναι ο ***Ετεραρχικός Έλεγχος*** **και τα *Holonic Συστήματα Κατεργασιών***. Στην πρώτη περίπτωση ελέγχου, εξαλείφεται οποιοδήποτε κεντρικό στοιχείο προερχόμενο από τις παραδοσιακές τεχνικές κεντρικού και ιεραρχικού ελέγχου συστημάτων παραγωγής. Αυτό σημαίνει ότι όλοι οι τοπικοί ελεγκτές συνδέονται διαμέσου ενός δικτύου επικοινωνίας και είναι αποκλειστικά υπεύθυνοι για εργασίες ελέγχου εντός της περιοχής δράσης τους. Από την άλλη, τα Holonic Συστήματα Κατεργασιών αποτελούνται από αυτόνομες και αυτοδύναμες μονάδες (holons) που λειτουργούν σε ευέλικτη ιεραρχία. Κάθε σύστημα μπορεί να είναι holon, εφόσον είναι ικανό να ελέγχει την εκτέλεση των σχεδίων και στρατηγικών του. Πολλά στοιχεία της παραγωγής όπως οι μηχανές, οι ταινιόδρομοι, τα αυτόματα καθοδηγούμενα οχήματα, ο ανθρώπινος παράγοντας αλλά και οι παραγγελίες που φτάνουν σε ένα σύστημα παραγωγής μπορεί να είναι holon. Ένα σύστημα από holons που συνεργάζεται για την κάλυψη στόχων παραγωγής ονομάζεται ολο-αρχία (holarchy) [6]. Ένα holon μπορεί από μόνο του να συντελεί και μία ολο-αρχία και να συνεργάζεται με άλλες καθώς και με την κύρια ολο-αρχία του συστήματος. Επιπλέον, έχουν την ικανότητα εμπλοκής σε πολλές διαφορετικές ιεραρχίες, ενώ μπορούν να δημιουργούν και αλλάζουν ιεραρχίες εντός μίας ολο-αρχίας. Στο παρακάτω σχήμα απεικονίζονται όλες οι πιθανές διεπαφές μεταξύ των holons.



Σχήμα 4.9 Αλληλεπιδράσεις Holons [3]

Σχετικά με το σχεδιασμό και τον έλεγχο της παραγωγής τα Holonic Manufacturing Systems έχουν αντιμετωπίσει με επιτυχία τα ακόλουθα :

- Κατανεμημένη αποσύνθεση παραγγελιών σε εργασίες κατεργασιών
- Κατανεμημένο προγραμματισμό εργασιών κατεργασίας μεταξύ αυτόνομων και συνεργατικών μονάδων
- Αυτόνομη εκτέλεση εργασιών κατεργασίας που αλληλεπιδρούν με τη διεργασία προγραμματισμού
- Αρχιτεκτονικές ελέγχου μηχανών αποτελούμενες από συνεργαζόμενες συσκευές.

Η τεχνολογία ελέγχου βασισμένη σε πράκτορες, που αναπτύσσεται στην παρούσα εργασία αφορά κυρίως τα Holonic Συστήματα Κατεργασιών.

ΚΕΦΑΛΑΙΟ 5: Μοντελοποίηση-Σχεδιασμός Ευέλικτου Συστήματος Κατεργασιών Εργαστηρίου Τεχνολογίας των Κατεργασιών

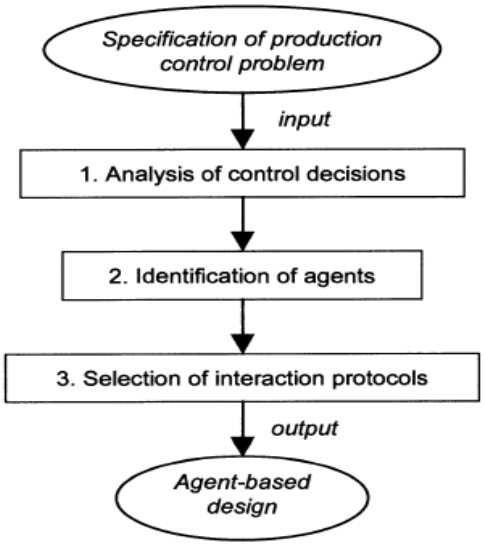
5.1 Εισαγωγή

Για την ανάπτυξη και εφαρμογή συστήματος ελέγχου, βασισμένου στη τεχνολογία πρακτόρων, για το ευέλικτο σύστημα κατεργασιών της παρούσας εργασίας στηριχθήκαμε σε συγκεκριμένη μεθοδολογία σχεδιασμού [10] και η επακόλουθη ανάπτυξη του συστήματος ελέγχου σχετικά με την ενσωμάτωση των Δικτύων Petri στον agent-based ελεγκτή του FMS του εργαστηρίου για την αποφυγή ανεπιθύμητων καταστάσεων βασίστηκε στην αναφορά [4].

5.2 Η μεθοδολογία DACS (Design of Agent-based Production Control Systems)

Η συγκεκριμένη μεθοδολογία ακολουθεί την κοινή προσέγγιση αναγνώρισης των πρακτόρων της εφαρμογής και μετά προχωράει στο σχεδιασμό των αλληλεπιδράσεων μεταξύ τους. Γενικότερα, ακολουθεί δύο φάσεις, της ανάλυσης και του σχεδιασμού. Αντλεί στοιχεία από μοντέλα της Θεωρίας Αποφάσεων και της Θεωρίας Ελέγχου των Κατεργασιών για να δημιουργήσει μοντέλα αποφάσεων ελέγχου που θα χρησιμοποιηθούν στην αναγνώριση πρακτόρων ελέγχου. Τέλος, θα δημιουργήσει συγκεκριμένες έννοιες βασισμένες σε πράκτορες, όπως τις εμπλεκόμενες εξαρτήσεις μεταξύ αυτών για το σχηματισμό των τελικών πραγματικών πρακτόρων και των αλληλεπιδράσεων αυτών [3].
Πιο συγκεκριμένα, η μεθοδολογία αποτελείται από τα εξής σημαντικά βήματα :

- Ανάλυση Λήψης Αποφάσεων (Αποφάσεις Ελέγχου και εξαρτήσεις μεταξύ αυτών)
- Αναγνώριση Πρακτόρων
- Επιλογή Πρωτόκολλου Αλληλεπίδρασης Πρακτόρων



Σχήμα 5.1 Στάδια Μεθοδολογίας DACS

5.2.1 Προσδιορισμός Συστήματος Ελέγχου Ευέλικτου Συστήματος Παραγωγής

Α.) Προσδιορισμός Φυσικών Στοιχείων του Ευέλικτου Συστήματος

Το προς μελέτη, μοντελοποίηση και έλεγχο ευέλικτο σύστημα του εργαστηρίου όπως αυτό επισημάνθηκε στο πρώτο κεφάλαιο της παρούσας εργασίας αποτελείται από δύο μηχανές (φρέζα και τόρνο) , δύο ενδιάμεσες αποθήκες, ένα ρομπότ μεταφοράς των εμπλεκόμενων τεμαχίων, μία είσοδο και μία έξοδο του συστήματος.

Β.) Προσδιορισμός Συνθηκών Λειτουργίας του Συστήματος

Το σύστημα εκτελεί 4 εργασίες όπως αυτές επισημάνθηκαν και στο πρώτο κεφάλαιο της παρούσας εργασίας ήτοι:

- ΠΡΟΙΟΝ Α : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΟΝ ΤΟΡΝΟ (JOB 1) – με είσοδο τεμαχίου τύπου Α
- ΠΡΟΙΟΝ Β : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΗ ΦΡΕΖΑ (JOB 2) – με είσοδο τεμαχίου τύπου Γ
- ΠΡΟΙΟΝ C : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΟΝ ΤΟΡΝΟ → ΣΤΗ ΦΡΕΖΑ (JOB 3) – με είσοδο τεμαχίου τύπου Β
- ΠΡΟΙΟΝ D : ΕΠΕΞΕΡΓΑΣΙΑ ΣΤΗ ΦΡΕΖΑ → ΣΤΟ ΤΟΡΝΟ (JOB 4) – με είσοδο τεμαχίου τύπου Δ

Γ.) Τέλος , ο προσδιορισμός των στόχων της παραγωγής, είναι η ολοκλήρωση των παραγγελιών με βέλτιστη απόδοση του συστήματος και την απαραίτητη ευελιξία εξασφαλίζοντας ευρωστία στην περίπτωση μηχανολογικών σφαλμάτων, την απαραίτητη ποιότητα, την συντήρηση και άλλους παράγοντες.

5.2.2 Ανάλυση Αποφάσεων Ελέγχου

Για την αποφυγή απεριόριστων υποθέσεων σχετικά με την επίλυση ελέγχου του συστήματος, θα πρέπει να εστιάσουμε στην αρχή της φάσεως σχεδιασμού του ελεγκτή στην αναγνώριση αυτών των καταστάσεων που αναπαριστούν τις ενέργειες των αποφάσεων που ο ελεγκτής πρέπει να αναγνωρίσει για την επίτευξη των στόχων της παραγωγής. Αυτές οι ενέργειες αποφάσεων στα πλαίσια της συγκεκριμένης μεθοδολογίας καλούνται στη διεθνή βιβλιογραφία «effectoric» (αποφάσεις με τουλάχιστον μία φυσική ενέργεια ως αποτέλεσμα). Στη συνέχεια κρίνεται αναγκαία η αναγνώριση των εξαρτήσεων μεταξύ αυτών των αποφάσεων. Η ανάγκη αυτή μπορεί να γίνει αντιληπτή αν σκεφτούμε την ανάθεση συγκεκριμένου τεμαχίου κάποιας εργασίας σε μηχανή συγκεκριμένης χωρητικότητας η οποία μπορεί να επεξεργαστεί και άλλα τεμάχια άλλων εργασιών. Συνεπώς, οι αναθέσεις διαφορετικών τεμαχίων σε κάποια μηχανή θα πρέπει να συντονιστούν για την επίτευξη του στόχου του συστήματος.
Για την αναγνώριση των παραπάνω αποφάσεων πρέπει να εστιάσουμε στα στοιχεία της παραγωγής και να συλλέξουμε όλες εκείνες τις αποφάσεις που μας δίνουν τουλάχιστον δύο εναλλακτικές ενέργειες. Ειδικότερα, μπορούμε να εστιάσουμε στους πόρους του

συστήματος και τους διαφορετικούς τύπους τεμαχίων προς επεξεργασία σε αυτούς για να αναγνωρίσουμε αποτελεσματικά όλες τις απαραίτητες, εμπλεκόμενες αποφάσεις. Μετά την αναγνώριση όλων των σχετικών αποφάσεων, μπορούμε να προχωρήσουμε στο χαρακτηρισμό αυτών με βάση τον γενικό ακόλουθο πίνακα.

Attribute	Description	
id	unique identifier	mandatory
title	short description of the decision task	optional
parameters	Optional parameters that function as variables in the specification of the following attributes. With the help of parameters, similar decision tasks can be generalised into decision types covering different situations during the production process.	optional
control interface	The control interface that is necessary to enact the physical actions of the decision space.	mandatory
trigger	Specification of situations in the production process in which the effectoric decision becomes necessary.	mandatory
decision space	The set of decision alternatives available in this decision situation. At least one of the decision alternatives must be a physical action.	mandatory
local decision rule	Any constraints and preferences on the decision space that are induced by local aspects of the decision situation.	optional

Πίνακα 5.1 Χαρακτηρισμός Διαμορφώσεως Αποφάσεων [3]

Η δι-επαφή ελέγχου (control interface), ο προσδιορισμός των καταστάσεων της παραγωγικής διαδικασίας (trigger) και ο χώρος αποφάσεων (decision space) θεωρούνται απαραίτητα στοιχεία για τον χαρακτηρισμό των αποφάσεων του συστήματος.
Στην περίπτωση του ευέλικτου συστήματος κατεργασιών που μελετάται στην παρούσα εργασία θέτουμε τους εξής συμβολισμούς :
W_i → τα τεμάχια που εισέρχονται προς κατεργασία στο σύστημα
D_i → οι εμπλεκόμενες αποφάσεις στη λειτουργία του συστήματος
R → το ρομπότ
M_i → οι μηχανές κατεργασίας
B_i → οι ενδιάμεσες αποθήκες
In → η αποθήκη εισόδου
Out →η αποθήκη εξόδου

Στην αρχή του συστήματος γίνεται από το ρομπότ η επιλογή τεμαχίου από την αρχική αποθήκη. Έτσι η αρχική απόφαση του συστήματος είναι η D_1 και διαμορφώνεται ως ακολούθως.

Id : D_1

Title : Επίλεξε ένα τεμάχιο στο In

Parameters: -

Control Interface : Στο Robot

Trigger : Η M_i είναι ελεύθερη

Decision Space : W_1-W_4 (4 διαφορετικά τεμάχια για 4 διαφορετικές εργασίες)

Local Decision Space: Επίλεξε ένα τεμάχιο.

Η επόμενη απόφαση είναι η D_2 και σχετίζεται με την μεταφορά τεμαχίου στη μηχανή M_1. Σύμφωνα με τον παραπάνω πίνακα έχουμε :

Id : D_2

Title: Μετάφερε το τεμάχιο στη Μηχανή M_i

Parameters: W_1 ,W_2, W_3, W_4

Control Interface: Στη Μηχανή M_i

Trigger: W_1,W_2, W_3, W_4 φθάνουν στη M_i

Decision Space: {Είσοδος στη M_i}

Local Decision Space: {W_1,W_3} στη M_1 και {W_2,W_4} στη M_2

Εν συνεχεία, προχωράμε στον προσδιορισμό των αποφάσεων σχετικά με την είσοδο των επεξεργασμένων τεμαχίων στις ενδιάμεσες αποθήκες B_1 και B_2.

Id : D_3

Title: Μετάφερε το τεμάχιο στην B_i

Parameters: W_2, W_4

Control Interface: B_i

Trigger: W_2, W_4 φθάνει στη B_i

Decision Space: {Είσοδος στη B_i}

Local Decision Space: W_2 στη B_2, W_4 στη B_1

Η επόμενη απόφαση του συστήματος σχετίζεται με την κατεργασία των τεμαχίων στην κάθε μηχανή. Έτσι έχουμε ,

Id : D_4

Title: Επεξεργασία στη M_i

Parameters: W_1, W_2, W_3, W_4 και M_i

Control Interface: M_i

Trigger: W_1 – W_4 φθάνουν στη M_i

Decision Space: Σύνολο κατεργασιών στη M_i

Local Decision Space: W_1, W_3 στη M_1 και W_2, W_4 στη M_2

Σχετικά με την αναμονή του τεμαχίου στις ενδιάμεσες αποθήκες έχουμε:

Id : D_5

Title: Αναμονή τεμαχίου στην B_i

Parameters: W_2, W_4

Control Interface: B_i

Trigger: -

Decision Space: -

Local Decision Space: -

Η απόφαση που σχετίζεται με την έξοδο ολοκληρωμένου προιόντος από το σύστημα είναι
Id : D_6

Title: Μετέφερε το τεμάχιο στην έξοδο

Parameters: -

Control Interface: Στην αποθήκη εξόδου

Trigger: -

Decision Space: Σύνολο επεξεργασμένων τεμαχίων

Local Decision Space: Επέλεξε όποιο επεξεργασμένο τεμάχιο είναι έτοιμο

Τελευταία απόφαση είναι εκείνη που σχετίζεται με την επιλογή της επόμενης μηχανής προς επεξεργασία :
Id : D_7

Title: Επέλεξε την επόμενη μηχανή επεξεργασίας

Parameters: -

Control Interface: M_i

Trigger: -

Decision Space: {W_2, W_4}

Local Decision Space: -

Στη συνέχεια της μεθοδολογίας προχωράμε στη ***διαμόρφωση των εξαρτήσεων*** που δημιουργούνται μεταξύ των αποφάσεων.
Οι εξαρτήσεις που δημιουργούνται μεταξύ των αποφάσεων μπορούν να συνοψιστούν στις εξής 3 ομάδες.

- Εξάρτηση Δρομολόγησης Τεμαχίων (ένα τεμάχιο μπορεί να δρομολογείται είτε σε μία μηχανή επεξεργασίας, είτε σε μία ενδιάμεση αποθήκη προς περαιτέρω επεξεργασία)
- Εξάρτηση Επιλογής Μηχανών Επεξεργασίας (ένα τεμάχιο μπορεί να επεξεργαστεί είτε στη μηχανή M_1 στη μηχανή M_2)
- Εξάρτηση Φορτώσεως Τεμαχίων (υπάρχουν 4 διαφορετικά είδη τεμαχίων προς επεξεργασία με 4 διαφορετικές δρομολογήσεις επεξεργασίας)

Ο αντίστοιχος πίνακας χαρακτηρισμού των εν λόγω εξαρτήσεων θα έχει τη γενική μορφή

Attribute	Description	
id	unique identifier	mandatory
title	short description of the dependency	optional
decision tasks	decision tasks or information sources involved in the dependency	mandatory
constraints	Any combination of decision alternatives that should not be chosen.	optional
preferences	Preference function on the combination of decision alternatives that are not forbidden by the constraints.	optional

Πίνακας 5.2 Χαρακτηρισμός Εξαρτήσεως Αποφάσεων [3]

Στο ευέλικτο σύστημα κατεργασιών που μελετάμε στην παρούσα εργασία οι υπάρχουσες εξαρτήσεις αποφάσεων μπορούν να οριστούν ως εξής:

Καταρχήν, οι Εξαρτήσεις Δρομολόγησης Τεμαχίων είναι:

Id: DP_1

Title: Δρομολόγηση τεμαχίων στη (επόμενη) Μηχανή M_i

Decision Tasks: D_2 και δυνατότητες επεξεργασίας μηχανής

Constraints: Κατεύθυνε ένα κομμάτι στη μηχανή για να επεξεργαστεί

Preferences: -

Εν συνεχεία , η εξάρτηση επιλογής μηχανής επεξεργασίας μπορεί να φανεί ως ακολούθως:

Id: DP_2

Title: Ελαχιστοποίηση αναγκαίας δρομολόγησης για κάθε τεμάχιο

Decision Tasks: D_2, D_3

Constraints: Επιλογή της κατάλληλης σειράς μηχανών για να ληφθούν όλες οι απαραίτητες κατεργασίες ενός τεμαχίου

Preferences: Επιλογή της ακριβούς σειράς μηχανών και ενδιάμεσων αποθηκών

Τέλος , η εξάρτηση φορτώσεως τεμαχίων διαμορφώνεται ως εξής:

Id: DP_3

Title: Περιορισμός στην απαραίτητη συνολική επεξεργασία τεμαχίων

Decision Tasks: D_1, D_2, D_3, D_4, D_5

Constraints: Εισαγωγή του απαραίτητου αριθμού τεμαχίων κάθε στιγμή στο σύστημα

Preferences: -

5.2.3 Βελτίωση Μοντέλου Αποφάσεων και Αναγνώριση Τελικών Πρακτόρων

Στο στάδιο αυτό επιδιώκεται η διαίρεση των διαμορφούμενων έως τώρα αποφάσεων, με βάση τους εμπλεκόμενους πόρους του συστήματος σε αυτές, και η δημιουργία υποχώρων αποφάσεων που θα μας επιτρέψουν την ακριβή δημιουργία των τελικών πρακτόρων του συστήματος ελέγχου.
Η απόφαση **D_1** αφορά μόνο στην επιλογή τεμαχίου από το ρομπότ από την αρχική αποθήκη. Συνεπώς, δεν μπορεί να υποδιαιρεθεί σε άλλες αποφάσεις. Το ίδιο ισχύει και για την απόφαση **D_6.**
Η απόφαση **D_2** που αφορά στη δρομολόγηση τεμαχίου σε μία μηχανή επεξεργασίας και υπάγεται στην εξάρτηση **DP_1**, μπορεί να υποδιαιρεθεί στις εξής αποφάσεις:

- **D_2_ma** (σχετικά με την επιλογή της μηχανής)
- **D_2_wp** (σχετικά με τον τύπο του τεμαχίου προς επεξεργασία)
- **D_2_rb** (σχετικά με την μεταφορά του τεμαχίου από το ρομπότ)

Αντίστοιχα η απόφαση **D_3** υποδιαιρείται σε :

- **D_3_rb** (σχετικά με την μεταφορά του τεμαχίου από το ρομπότ)
- **D_3_wp** (σχετικά με τον τύπο του τεμαχίου προς επεξεργασία)

Η απόφαση **D_4** υποδιαιρείται σε :

- **D_4_ma**
- **D_4_wp**

Η απόφαση **D_5** δεν υποδιαιρείται καθώς αφορά μόνο την αναμονή του τεμαχίου.
Τέλος, η απόφαση **D_7** διαιρείται σε:

- **D_7_ma**
- **D_7_wp**

Κατόπιν των παραπάνω διαμορφώσεων των εμπλεκόμενων αποφάσεων, και της υποδιαίρεσης αυτών σε σχέση με τους πόρους του συστήματος, οδηγούμαστε στην τελική αναγνώριση των πρακτόρων.

Machine Agent :

- **D_4_ma (process at Mi)**
- **D_7_ma (Choose next machine)**

Robot Agent:

- **D_1 (Load Workpiece)**
- **D_2_rb (Move to Mi)**
- **D_3_rb (Move to Bi)**
- **D_6 (Unload the product)**

Buffer Agent:

- **D_5 (Waiting for Process)**

Workpiece Agent:

- **D_2_wp (Workpiece Moving at Mi)**
- **D_3_wp (Workpiece Moving at Bi)**
- **D_4_wp (Workpiece Processing at Mi)**
- **D_7_wp (Choose next Machine)**

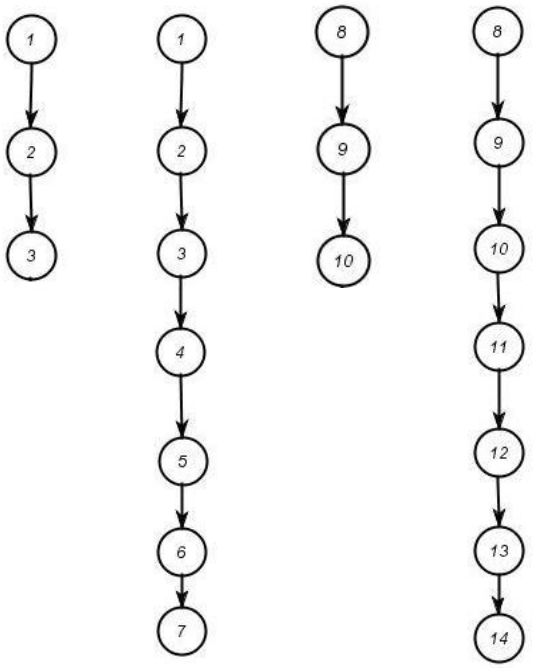
Έχοντας καταλήξει με βάση τη συγκεκριμένη μεθοδολογία στη διαμόρφωση των τελικών πρακτόρων, θα προχωρήσουμε στην ανάπτυξη του agent-based ελεγκτή με βάση το πρωτόκολλο Contract Net. Το συγκεκριμένο πρωτόκολλο επιλέγεται με βάση τις αποτελεσματικές εφαρμογές του σε πληθώρα συστημάτων στη διεθνή βιβλιογραφία.

5.3 Ανάπτυξη Τελικού Ελεγκτή με βάση το Πρωτόκολλο Contract Net (CNP)

Στην ενότητα αυτή θα αναπτύξουμε ένα **μηχανισμό συντονισμού** των πρακτόρων για να διεκπεραιώσουν τις εμπλεκόμενες εργασίες στο ευέλικτο σύστημα κατεργασιών που μελετάμε. Η ανάπτυξη αυτού του μηχανισμού θα βασιστεί στο πρωτόκολλο Contract Net. Η παράλληλη ενσωμάτωση των Δικτύων Petri σε αυτό θα γίνει προς επιδίωξη αποφυγής ανεπιθύμητων καταστάσεων του συστήματος που μελετάμε [5].

5.3.1 Σχηματισμός Συνεργατικού Δικτύου με το CNP

Όπως έχει προαναφερθεί, στο CNP υπάρχουν δύο τύποι πρακτόρων : οι διαχειριστές και οι πλειοδότες. Το κάθε έργο ανατίθεται σε ένα διαχειριστή για επεξεργασία. Η ροή εργασιών ενός έργου περιγράφεται από ένα απεριοδικό σημασμένο γράφο.
Στο παρακάτω Σχήμα απεικονίζονται οι ροές εργασιών του ευέλικτου συστήματος κατεργασιών του εργαστηρίου. Κάθε κόμβος στη ροή εργασιών υποδηλώνει μία λειτουργία του έργου.



Manager 1 Manager 2 Manager 3 Manager 4

Σχήμα 5.2 Ροές εργασιών FMS

Οι διαμορφούμενοι πράκτορες που μας έδωσε η μεθοδολογία DACS είναι οι 4 πράκτορες του συστήματος. Κάθε πλειοδότης έχει μία σειρά από πόρους υπό τον έλεγχό του. Ο κάθε τύπος πλειοδότη εκτελεί το ίδιο σύνολο λειτουργιών. Στο ευέλικτο σύστημα κατεργασιών που μελετάμε στην παρούσα εργασία οι πλειοδότες που σχετίζονται με τους πόρους του συστήματος είναι :

- Bidder 1: Machine1, Machine2
- Bidder 2: Buffer1, Buffer2
- Bidder 3: Robot

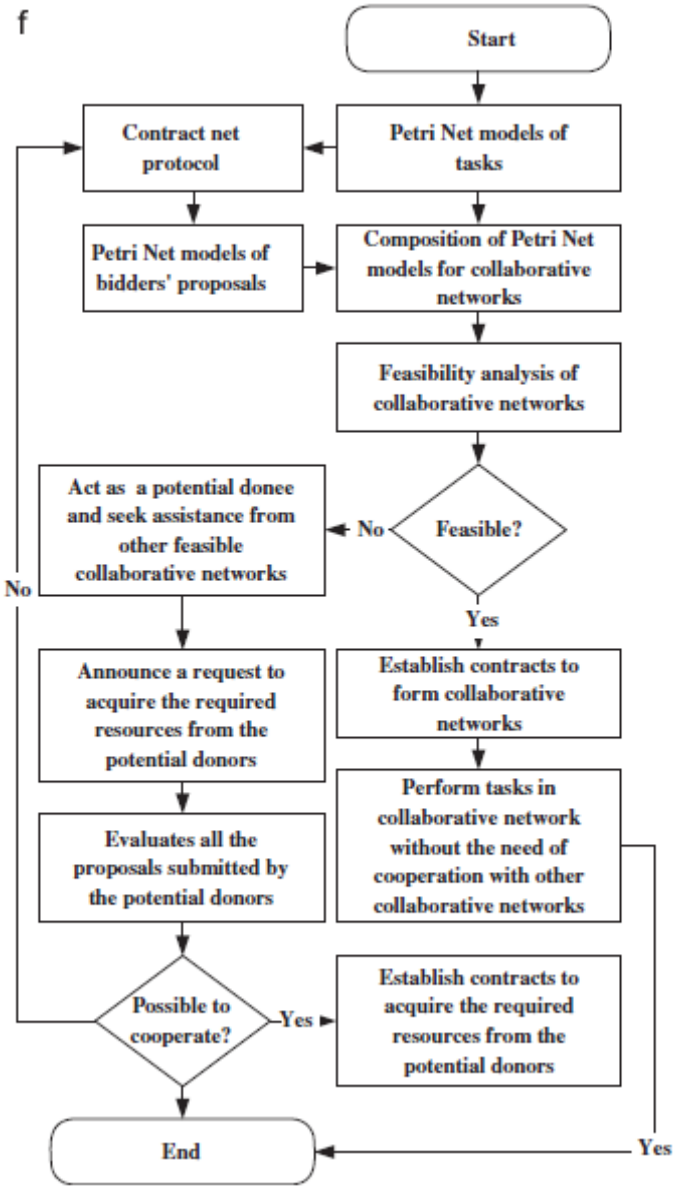
Σημειώνεται ότι οι διαχειριστές του παραπάνω σχήματος εμπεριέχουν τις 4 ροές εργασιών του ευέλικτου συστήματος οι οποίες αντιστοιχούν στα 4 διαφορετικά τεμάχια προς επεξεργασία :

- Workpiece 1 1→2→3
- Workpiece 2 1→2→3→4→5→6→7
- Workpiece 3 8→9→10
- Workpiece 4 8→9→10→11→12→13→14

Κάθε λειτουργία μπορεί να εκτελείται από έναν η περισσότερους πλειοδότες. Οι πλειοδότες επιπλέον μπορούν να συμμετέχουν σε μία ή περισσότερες δραστηριότητες

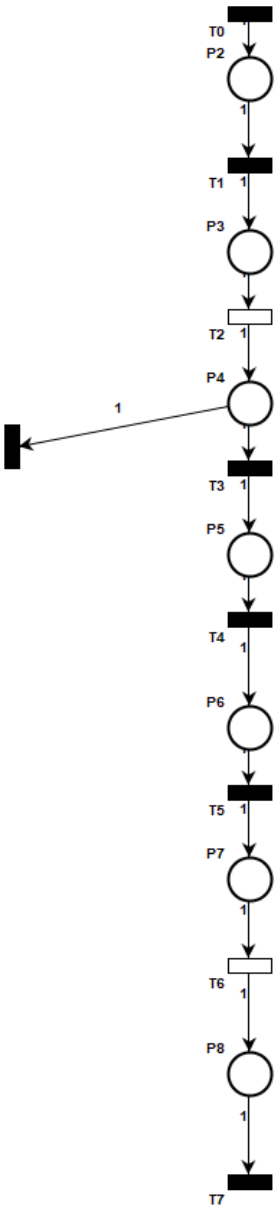
μίας ροής εργασιών δημιουργώντας συμβόλαια (contracts) με τους διαχειριστές. Ένας διαχειριστής διαπραγματεύεται με τους ενδεχόμενους πλειοδότες και αποφασίζει το σύνολο των πλειοδοτών που μπορούν να σχηματίσουν ένα συνεργατικό δίκτυο για την διεκπεραίωση συγκεκριμένου έργου. Εν γένει, το σύνολο των διαχειριστών και των πλειοδοτών μπορούν να αποτελέσουν ένα (ολικό) συνεργατικό δίκτυο. Η εφαρμογή του CNP δεν οδηγεί πάντα σε συνεργατικά δίκτυα που διεκπεραιώνουν όλες τις εργασίες αποτελεσματικά. Αυτό μπορεί να συμβεί εξαιτίας της αποδοχής πλεοναζόντων πόρων για την εκτέλεση της ανατιθέμενης εργασίας. Με αυτό τον τρόπο δυσκολεύεται η πρόοδος των έργων που έχουν ανατεθεί σε άλλους διαχειριστές.

Στη συνέχεια, υποθέτουμε ότι όλοι οι πράκτορες στο σύστημα είναι πρόθυμοι να εκτελέσουν όλα τα έργα που τους ανατίθενται και να συνεισφέρουν και στην διεκπεραίωση άλλων έργων. Στο ακόλουθο σχήμα απεικονίζεται ο μηχανισμός συντονισμού που επιδιώκουμε.

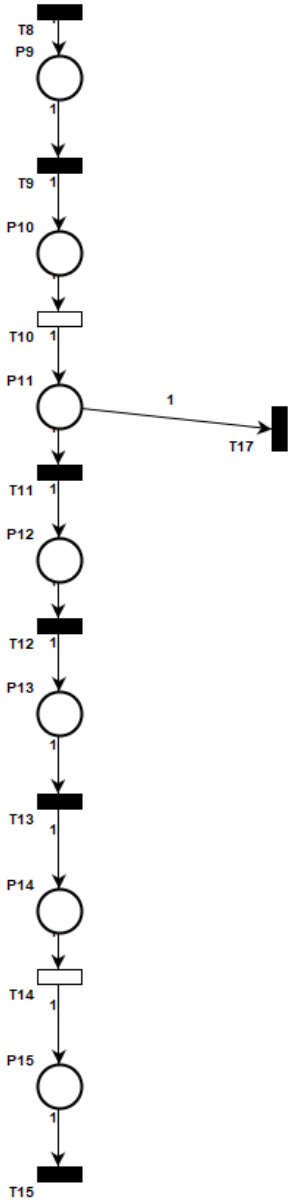


Σχήμα 5.3 Μηχανισμός Συντονισμού Πρακτόρων [5]

Ο διαχειριστής εξάγει ανάλυση σκοπιμότητας του συνεργατικού του δικτύου βασισμένος στη αντίστοιχη σύνθεση της ροής εργασίας με τα μοντέλα των πλειοδοτών, διατυπωμένα σε Δίκτυα Petri. Όταν ένας διαχειριστής δεν διαθέτει τους απαιτούμενους πόρους για την εκτέλεση του έργου ζητάει από άλλους διαχειριστές να του δώσουν πόρους. Οι διαχειριστές που μπορούν να δώσουν πόρους ονομάζονται δωρητές (donators) ενώ εκείνοι που ζητούν αποδέκτες (donees). Ο ενδεχόμενος δωρεοδόχος πρώτα ανακοινώνει μία αίτηση παραχώρησης πόρων και μετά αξιολογεί την πιθανότητα συνεργασίας με τους ενδεχόμενους δωρητές.
Οι ροές εργασιών (διαχειριστές), συνολικά, μπορούν να απεικονιστούν με δίκτυα Petri ως ακολούθως.

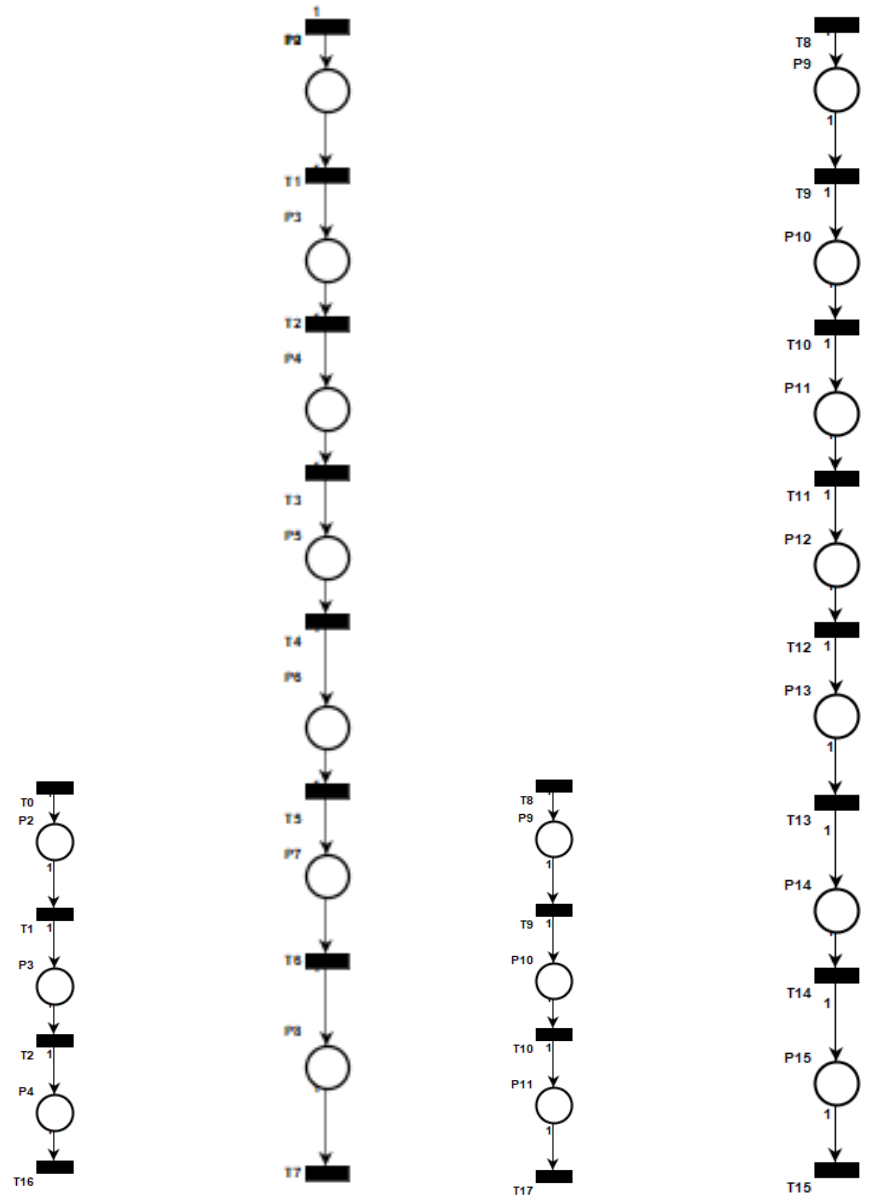


Manager 1&2



Manager 3&4

Σχήμα 5.4 Συνολική Απεικόνιση Ροών Εργασιών με Δίκτυα Petri



Manager 1 Manager 2 Manager 3 Manager 4

Σχήμα 5.5 Μεμονωμένες Ροές Εργασιών με Δίκτυα Petri

Με τις μεταβάσεις υποδηλώνονται οι λειτουργίες του κάθε έργου ενώ με τις θέσεις υποδηλώνονται οι καταστάσεις του έργου. Πιο συγκεκριμένα οι μεταβάσεις και οι θέσεις ορίζονται ως εξής :

Μεταβάσεις

- T0= Ξεκινάει η εκφόρτωση νέου τεμαχίου από την αρχική αποθήκη στο τόρνο
- T1= Ολοκληρώνεται η εκφόρτωση νέου τεμαχίου στο τόρνο – Ξεκινά η πρώτη κατεργασία
- T2= Ολοκληρώνεται η κατεργασία - Ξεκινά η εκφόρτωση του έτοιμου τεμαχίου από τη μηχανή κατεργασίας (τόρνος) στη τελική αποθήκη
- T3= Ολοκληρώνεται η κατεργασία – Ξεκινά η εκφόρτωση του τεμαχίου από τη μηχανή κατεργασίας (τόρνος) στην ενδιάμεση αποθήκη
- T4= Ολοκληρώνεται η φόρτωση του τεμαχίου από τον τόρνο στην ενδιάμεση αποθήκη
- T5= Ξεκινά η εκφόρτωση του αποθηκευμένου τεμαχίου από την ενδιάμεση αποθήκη στην φρέζα
- T6= Ολοκληρώνεται η κατεργασία - Ξεκινά η εκφόρτωση του έτοιμου τεμαχίου από τη μηχανή κατεργασίας (φρέζα) στη τελική αποθήκη
- T8= Ολοκληρώνεται η φόρτωση του έτοιμου τεμαχίου τύπου 2 από τη φρέζα στην τελική αποθήκη
- T9= Ξεκινάει η εκφόρτωση νέου τεμαχίου από την αρχική αποθήκη στη φρέζα
- T10= Ολοκληρώνεται η εκφόρτωση νέου τεμαχίου στη φρέζα– Ξεκινά η πρώτη κατεργασία
- T11= Ολοκληρώνεται η κατεργασία – Ξεκινά η εκφόρτωση του τεμαχίου από τη μηχανή κατεργασίας (φρέζα) στην ενδιάμεση αποθήκη
- T12= Ολοκληρώνεται η φόρτωση του τεμαχίου από τη φρέζα στην ενδιάμεση αποθήκη
- T13= Ξεκινά η εκφόρτωση του αποθηκευμένου τεμαχίου από την ενδιάμεση αποθήκη στον τόρνο
- T14= Ολοκληρώνεται η κατεργασία - Ξεκινά η εκφόρτωση του έτοιμου τεμαχίου από τη μηχανή κατεργασίας (φρέζα) στη τελική αποθήκη
- T15= Ολοκληρώνεται η φόρτωση του έτοιμου τεμαχίου τύπου 4 από τον τόρνο στην τελική αποθήκη
- T16= Ολοκληρώνεται η φόρτωση του έτοιμου τεμαχίου τύπου 1 από τον τόρνο στην τελική αποθήκη
- T17= Ολοκληρώνεται η φόρτωση του έτοιμου τεμαχίου τύπου 3 από τη φρέζα στην τελική αποθήκη

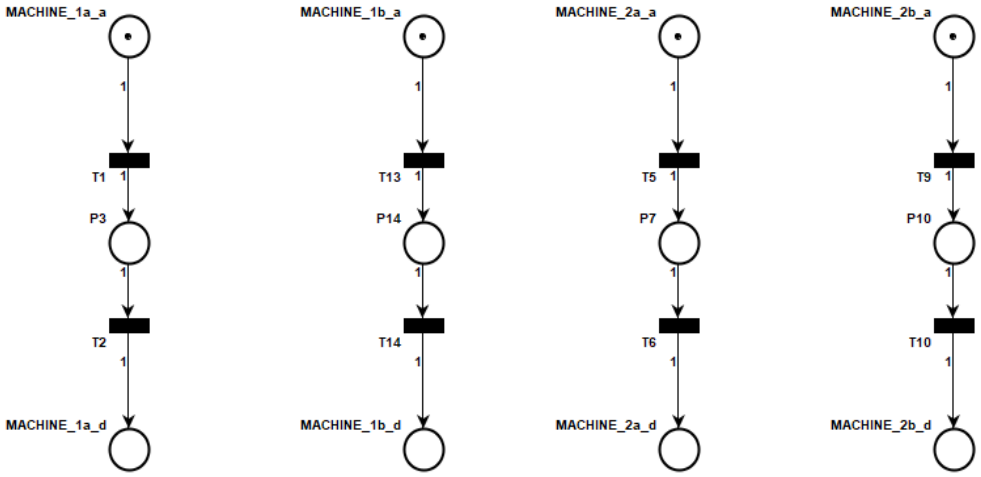
Θέσεις

- P2= Το ρομπότ ξεφορτώνει το τεμάχιο από την αποθήκη εισόδου και το φορτώνει στον τόρνο
- P3= Ο τόρνος κατεργάζεται το τεμάχιο τύπου 1
- P4= Το ρομπότ ξεφορτώνει το τεμάχιο από τον τόρνο και το φορτώνει σε επόμενο σταθμό
- P5= Ενδιάμεση αποθήκη για επεξεργασία στη φρέζα
- P6= το ρομπότ ξεφορτώνει το τεμάχιο από την ενδιάμεση αποθήκη και το φορτώνει στην φρέζα

- P7= Η φρέζα κατεργάζεται το τεμάχιο τύπου 3
- P8= το ρομπότ ξεφορτώνει το τεμάχιο από τη φρέζα και το φορτώνει στην τελική αποθήκη
- P9= το ρομπότ ξεφορτώνει το τεμάχιο από την αποθήκη εισόδου και το φορτώνει στη φρέζα
- P10=Η φρέζα κατεργάζεται το τεμάχιο τύπου 2
- P11= Το ρομπότ ξεφορτώνει το τεμάχιο από τη φρέζα και το φορτώνει σε επόμενο σταθμό
- P12= Ενδιάμεση αποθήκη για επεξεργασία στον τόρνο
- P13= το ρομπότ ξεφορτώνει το τεμάχιο από την ενδιάμεση αποθήκη και το φορτώνει στον τόρνο
- P14= Ο τόρνος επεξεργάζεται το τεμάχιο τύπου 4
- P15= το ρομπότ ξεφορτώνει το τεμάχιο από τον τόρνο και το φορτώνει στην τελική αποθήκη

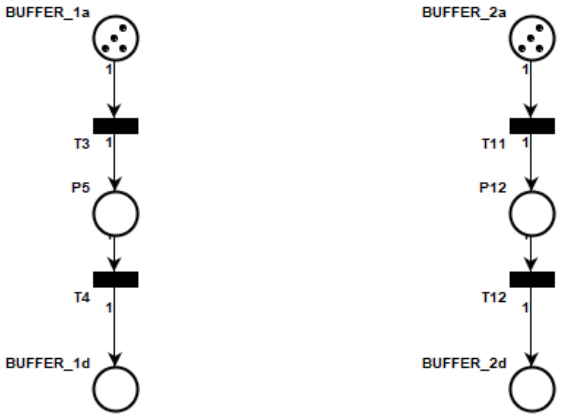
Οι πλειοδότες, υποδηλώνοντας τους πόρους του συστήματος, θα απεικονισθούν με την αρχική θέση τους στο δίκτυο Petri να περιέχει τη χωρητικότητα αυτών, πριν κατανέμουν κάποιο πόρο, και την τελική θέση τους αφού αποσπάσουν τις διατιθέμενες ποσότητες αυτών. Αυτή η διαδικασία λαμβάνει μέρος εμπλέκοντας την κατάλληλη θέση αλλά και τις μεταβάσεις του δικτύου Petri των διαχειριστών όπως αυτές περιγράφηκαν παραπάνω.
Οι δραστηριότητες των πόρων (πλειοδοτών) μπορούν να αναπαρασταθούν με δίκτυα Petri ως εξής:

Machines:



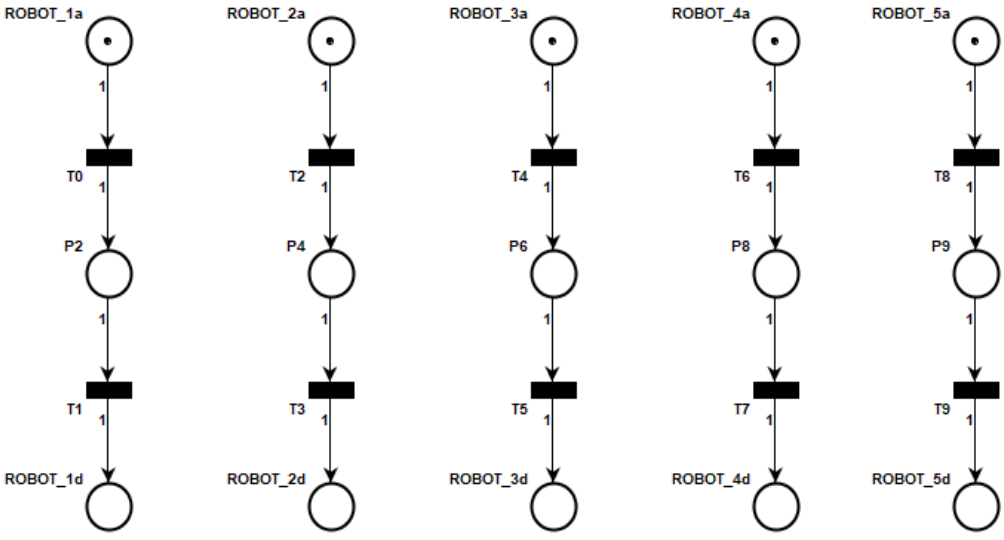
Σχήμα 5.6 Οι μηχανές-πλειοδότες του FMS σε Δίκτυα Petri

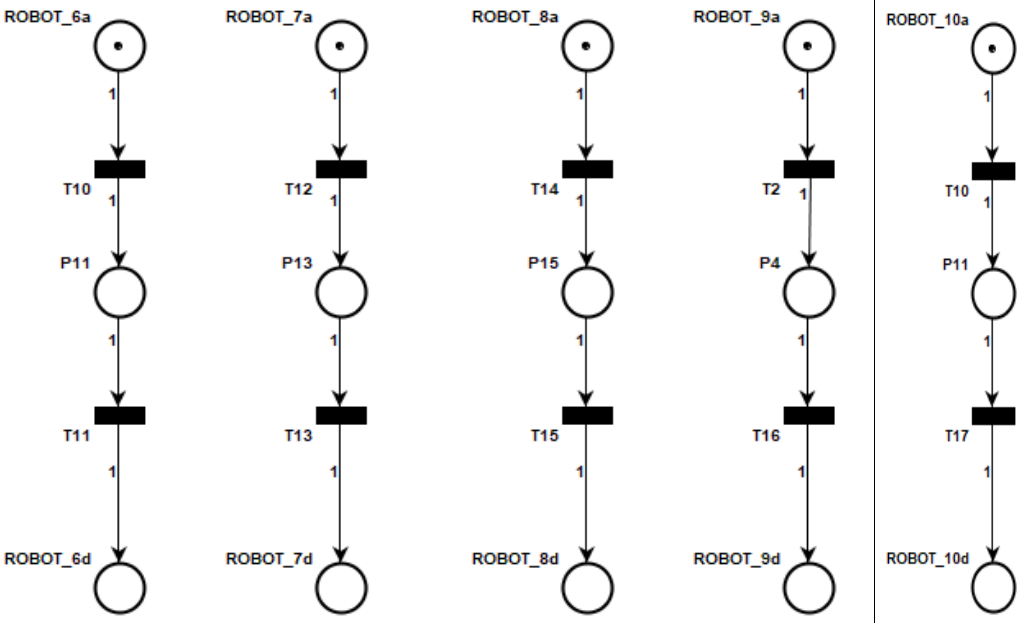
Buffers:



Σχήμα 5.7 Οι αποθήκες-πλειοδότες του FMS σε Δίκτυα Petri

Robot:

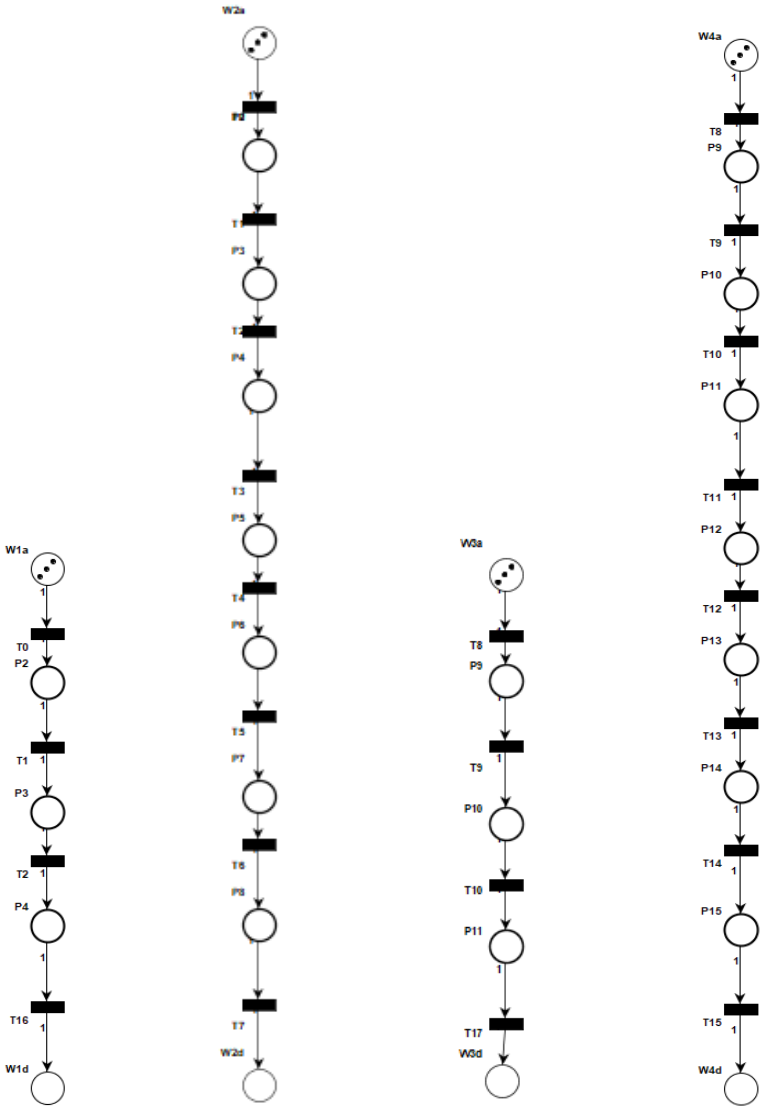




Σχήμα 5.8 Οι εργασίες του Ρομπότ – πλειοδότη του FMS σε Δίκτυα Petri

Διαχειριστές 1,2,3,4 (Workpieces) :

Τα τεμάχια προς επεξεργασία, όπως έχουμε προαναφέρει, θεωρούμε ότι είναι τεσσάρων διαφορετικών τύπων με 4 διαφορετικές διαδρομές επεξεργασίας. Για τον λόγο αυτό, οι ροές των εργασιών-διαχειριστές, θα απεικονίζονται σε 4 διαφορετικά είδη δικτύων Petri, υποδηλώνοντας τις διαδρομές των τεμαχίων και τις αρχικές ποσότητες αυτών. Θεωρούμε πως για κάθε τύπο τεμαχίου υπάρχουν διαθέσιμες στην αρχική αποθήκη 3 μονάδες.



Manager 1 (Workpiece 1) | Manager 2 (Workpiece 2) | Manager 3 (Workpiece 3) | Manager4 (Workpiece 4)

Σχήμα 5.9 : Απεικονίσεις Επεξεργασίας Τεμαχίων (Διαχειριστές) σε Δίκτυα Petri

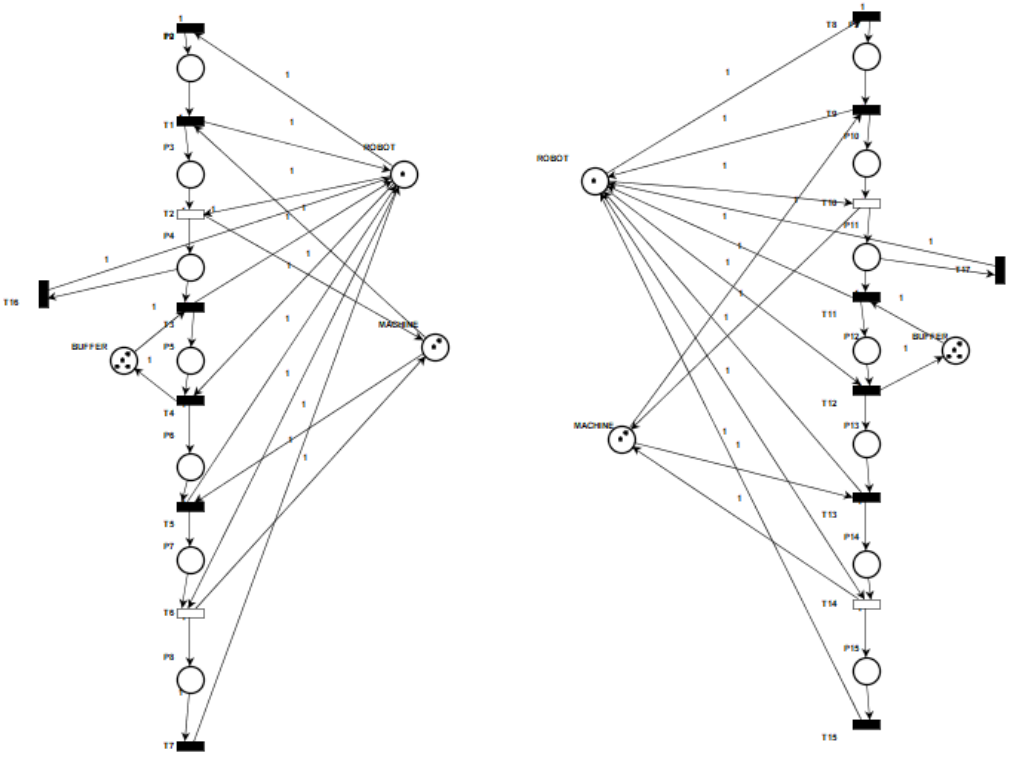
Μεμονωμένα, στις απεικονίσεις των πλειοδοτών με δίκτυα Petri, οι πρώτες και οι τελευταίες μεταβάσεις και θέσεις απεικονίζουν την κατανομή (allocation) και την απόσπαση (deallocation) των διατιθέμενων πόρων αντίστοιχα.

Εν συνεχεία, συγχωνεύοντας τα δίκτυα Petri των διαχειριστών και των διαμορφούμενων πλειοδοτών, αλλά και τις μεταβάσεις και θέσεις κατανομής-απόσπασης των πλειοδοτών των παραπάνω δικτύων, καταλήγουμε στους ακόλουθους απεριοδικούς σημασμένους γράφους (Acyclic Marked Graphs – AMG). Οι συγχωνεύσεις που λαμβάνουν χώρα είναι :

{Machine_1a, Machine_2a, Machine_3a, Machine_4a}∪{Machine_1d, Machine_2d, Machine_3d, Machine_4d} = ***Machine***
{Buffer_1a, Buffer_2a}∪{ Buffer_1d, Buffer_2d} = ***Buffer***

{Robot_a1, Robot_a2, Robot_a3, Robot_a4, Robot_a5, Robot_a6, Robot_a7, Robot_a8, Robot_a9, Robot_a10}∪{Robot_d1, Robot_d2, Robot_d3, Robot_d4, Robot_d5, Robot_d6, Robot_d7, Robot_d8, Robot_d9, Robot_d10} = ***Robot***

{Wa1, Wa2, Wa3, Wa4}∪{Wd1, Wd2, Wd3, Wd4} = ***P_idle***



Task 1 ($N_1(m_0)$) Task 2 ($N_2(m_0)$)

Σχήμα 5.10 Εργασίες Διαχειριστών με τους εμπλεκόμενους πόρους

Εδώ σημειώνουμε ότι, ο πλειοδότης Bidder 1 κατέχει συνολικά δύο πόρους στο σύστημα (φρέζα και τόρνος), ο πλειοδότης Bidder 2 κατέχει συνολικά 8 πόρους σε όλο το σύστημα (Buffer 1 , Buffer 2 από 4 θέσεις στην κάθε μία), ο πλειοδότης Bidder 3 (Ρομπότ) κατέχει συνολικά 1 θέση σε όλο το σύστημα καθώς αποτελείται από έναν και μόνο πόρο. Τέλος, τα αρχικά τεμάχια είναι στο σύνολό τους 12 (3 από τον κάθε τύπο) και θα συμπεριληφθούν στο τελικό δίκτυο Petri.
Ένα έργο w, μπορεί να διεκπεραιωθεί αν και μόνο αν το συνεργατικό δίκτυο, έστω $N_w(m_0)$, που διαμορφώθηκε είναι εφικτό [5]. Αυτό θα επιβεβαιωθεί (για το ολικό συνεργατικό δίκτυο) με συγκεκριμένη μέθοδο στο Κεφάλαιο 7 της εργασίας.

5.3.2 Συνθήκη Συνεργασίας Συνεργατικών Δικτύων

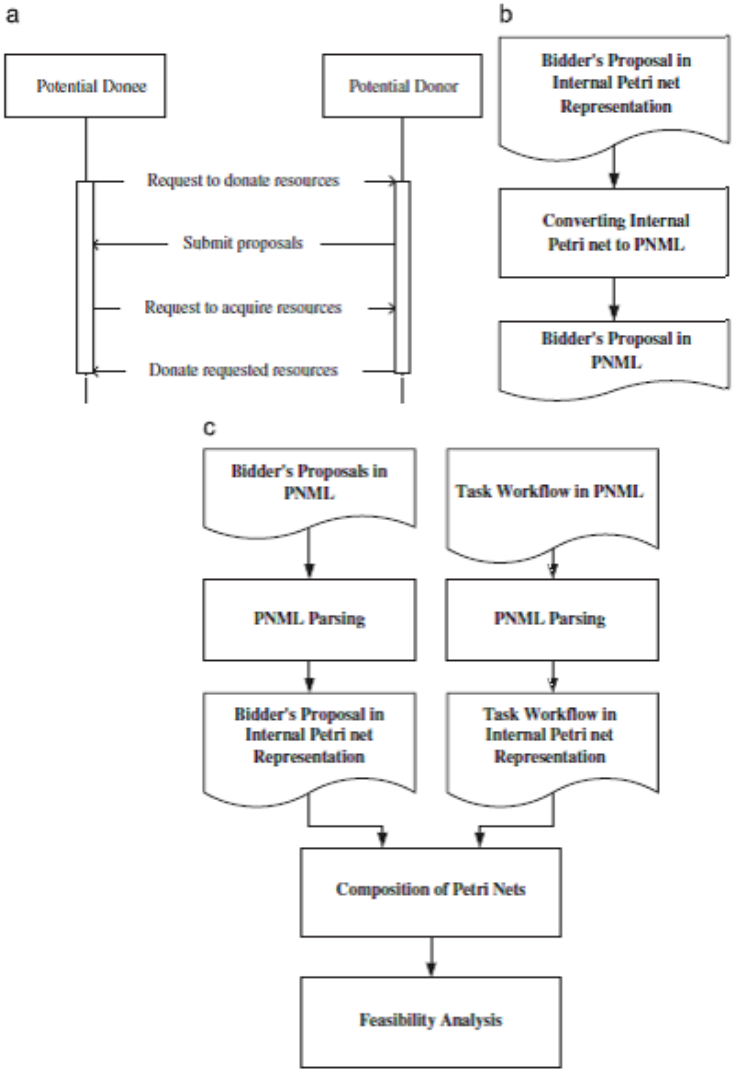
Ένας διαχειριστής μπορεί να σχηματίσει συνεργατικό δίκτυο αποκτώντας τους απαιτούμενους πόρους για την ολοκλήρωση των έργων που του έχουν ανατεθεί. Πλεονάζοντες πόροι μπορεί να υπάρξουν σε ένα συνεργατικό δίκτυο ακόμα και αν ο διαχειριστής εφαρμόσει συγκεκριμένες τεχνικές βελτιστοποίησης για να ελαχιστοποιήσει τους απαιτούμενους πόρους για ένα έργο [5]. Αυτό μπορεί να συμβεί κυρίως για τους ακόλουθους δύο λόγους:

- Τα έργα μπορεί να φθάνουν στο σύστημα δυναμικά και στοχαστικά. Εάν ο χρόνος αφίξεως μεταξύ δύο πόρων είναι μεγάλος, η κατανομή των ελάχιστων πόρων επιβάλει ότι οι μη κατανεμημένοι πόροι είναι διαθέσιμοι στο χρόνο αυτό. Κάτι τέτοιο όμως δεν είναι αποτελεσματικό καθώς μειώνει την απόδοση του συστήματος. Στην περίπτωση αυτή ένας διαχειριστής που λαμβάνει πλεονάζοντες πόρους, μπορεί να δώσει αυτούς σε άλλους διαχειριστές που τους έχουν ανάγκη και έτσι να βελτιωθεί η απόδοση.
- Πλεονάζοντες πόροι μπορούν ακόμα να υπάρξουν και με την πρόοδο εκτέλεσης του έργου. Αυτό μπορεί να συμβεί όταν με την ολοκλήρωση μίας εργασίας οι εμπλεκόμενοι πόροι αποδεσμευτούν και αποτελέσουν πλεονάζοντες, Στην περίπτωση αυτή πάλι ο διαχειριστής θα πρέπει να δώσει αυτούς σε άλλους διαχειριστές.

Γενικότερα, όταν ένας διαχειριστής δεν διαθέτει τους απαραίτητους πόρους για την εκτέλεση ενός έργου, ζητάει από άλλους διαχειριστές που χρησιμοποιούν τους ίδιους πόρους να του δώσουν. Αντίθετα, όταν ένας διαχειριστής διαθέτει πλεονάζοντες πόρους, δίνει αυτούς σε άλλους διαχειριστές που τους αιτούνται. Στην πρώτη περίπτωση το συνεργατικό δίκτυο χαρακτηρίζεται ανέφικτο και στη δεύτερη εφικτό. Στην περίπτωση του ευέλικτου συστήματος κατεργασιών του Εργαστηρίου και θεωρώντας δύο ομάδες προϊόντων, με βάση τις κοινές τους φάσεις κατεργασίας σε κάθε μηχανή (Workpiece 1,2,3,4), μπορούμε να διακρίνουμε τα εξής:
Για την εκτέλεση των εργασιών 1 (ΠΡΟΙΟΝ Α) και 2 (ΠΡΟΙΟΝ Γ) που εμπεριέχονται στην πρώτη και τρίτη ροή εργασίας αντίστοιχα απαιτούνται 4 πόροι. Το ρομπότ για την μεταφορά του τεμαχίου, ο τόρνος σαν πρώτος σταθμός εργασίας, η ενδιάμεση αποθήκη για την περίπτωση της εργασίας 2 που το τεμάχιο απαιτεί περαιτέρω κατεργασία και η φρέζα που είναι ο δεύτερος σταθμός εργασίας. Από την άλλη πλευρά, οι εργασίες 3 (ΠΡΟΙΟΝ Β) και 4 (ΠΡΟΙΟΝ Δ) της δεύτερης και τέταρτης ροής εργασιών απαιτούν τους ίδιους πόρους με διαφορετική σειρά μηχανών στην περίπτωση του Προϊόντος Δ και την χρήση της φρέζας αντί του τόρνου στην περίπτωση του Προϊόντος Γ. Συνεπώς, δημιουργείται η ανάγκη αίτησης, των διαχειριστών 3 και 4, για πλεονάζοντες πόρους από τους διαχειριστές 1 και 2. Αυτό μπορεί να συμβεί όταν στις διάφορες φάσεις του πρώτου έργου υπάρχουν διαθέσιμοι πόροι (μηχανής, του ρομπότ, ενδιάμεσης αποθήκης) και έτσι μπορούν δοθούν για την διεκπεραίωση εργασιών του δεύτερου έργου. Οι διαχειριστές των εργασιών της ίδιας ομάδας προϊόντων, θα χρειαστεί επιπλέον να αιτηθούν πλεονάζοντες πόρους και μεταξύ τους καθώς ο κάθε διαχειριστής αφορά σε μία ξεχωριστή εντολή παραγωγής/προϊόντος.
Τέλος, μπορούμε να επισημάνουμε από τα προαναφερθέντα ότι, ένα συνεργατικό δίκτυο είναι δωρητής (donator) εάν είναι εφικτό και διαθέτει τουλάχιστον ένα τύπο πλεοναζόντων πόρων. Από την άλλη ένα συνεργατικό δίκτυο είναι αποδέκτης (donee) όταν είναι ανέφικτο.

5.3.3 Αρχιτεκτονική Εφαρμογής

Στην ενότητα αυτή εφαρμόζουμε ένα μηχανισμό για να διευκολυνθεί η συνεργασία μεταξύ ενός συνόλου ενδεχόμενων δωρητών και ενός αποδέκτη. Οι αλληλεπιδράσεις τους απεικονίζονται από το διάγραμμα ροής **a** του παρακάτω σχήματος, όπου χρησιμοποιούνται 4 μηνύματα (request to donate resources, submit resource donation proposals, request to acquire resources, donate the requested resources).

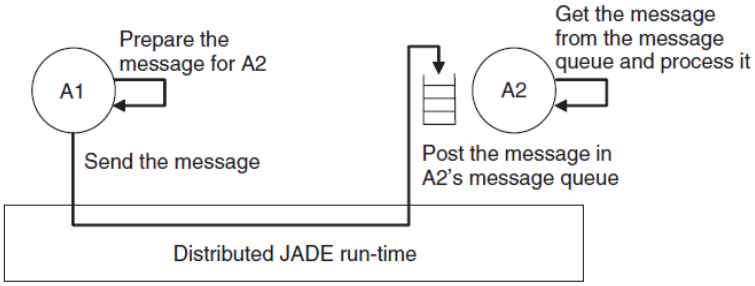


Σχήμα 5.11 a.) Διάγραμμα ροής, b.) Διαδικασία Κωδικοποίησης Πλειοδότη, c.) Διαδικασία Διαχειριστή [5]

Ο **μηχανισμός συντονισμού** θα εφαρμοστεί διαμέσου μίας συμβατής πλατφόρμας κατά τον FIPA (www.fipa.org) που υποστηρίζει την ανάπτυξη συστήματος πολλαπλών πρακτόρων. Η επιλεγμένη πλατφόρμα είναι η JADE. Τα μηνύματα που ανταλλάζονται από πράκτορες στη JADE έχουν τη μορφή που προσδιορίζεται από την ACL language κατά τα διεθνή πρότυπα του FIPA για τη δια-λειτουργικότητα των πρακτόρων. Η μορφή αυτή περιέχει τον ακόλουθο αριθμό πεδίων:

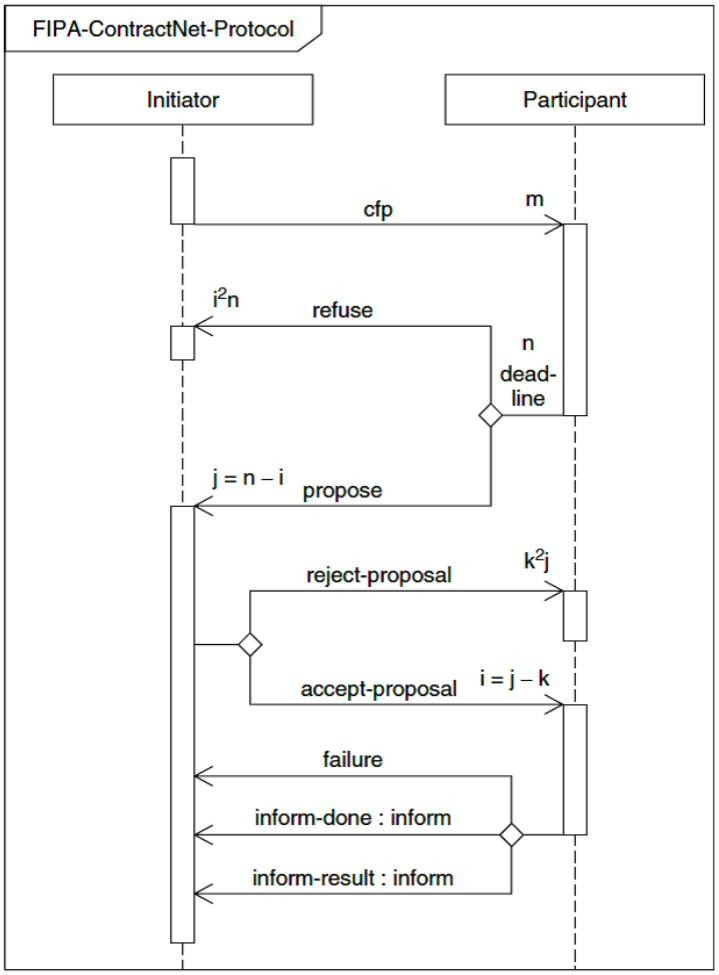
- αποστολέα του μηνύματος
- λίστα των αποδεκτών

- επικοινωνιακό σκοπό (τι σκοπεύει ο αποστολέας να αποκτήσει με την αποστολή του μηνύματος)
- περιεχόμενο (η πραγματική πληροφορία που περιλαμβάνεται στο μήνυμα)



Σχήμα 5.12 : Ασύγχρονη Αποστολή μηνυμάτων της JADE [2]

Επειδή οι πλειοδότες (bidders) έχουν απεικονιστεί με ένα δίκτυο Petri θα χρησιμοποιήσουμε την **Petri Net Mark Up Language** για να συμπληρώσουμε το περιεχόμενο των μηνυμάτων των προτάσεων των πλειοδοτών. Η **PNML** εξάγεται κατευθείαν από τα δίκτυα Petri συγκεκριμένων συμβατών εργαλείων προσομοίωσης όπως το PIPE2 που χρησιμοποιήθηκε στην παρούσα εργασία. Ο πλειοδότης, συνεπώς, μετατρέπει το δίκτυο Petri σε μορφή **PNML** και το συμπληρώνει στο αντίστοιχο πεδίο σύμφωνα με το διάγραμμα **b** του παραπάνω σχήματος. Από την άλλη, ο διαχειριστής πρέπει να αποκωδικοποιήσει τα μηνύματα που δέχεται από τους πλειοδότες και στη συνέχεια να τα «μετατρέψει» σε δίκτυα Petri, αποδεχόμενος την κατάλληλη πρόταση που συνθέτει αυτά μαζί του. Παράλληλα, αποκωδικοποιεί και την σχετική ροή εργασίας των έργων για να συνθέσει το τελικό δίκτυο Petri προκειμένου να γίνει ανάλυση σκοπιμότητας.



Σχήμα 5.13: FIPA –Contract Net Protocol [2]

5.3.4 JADE (Java Agent-Based Development Framework)

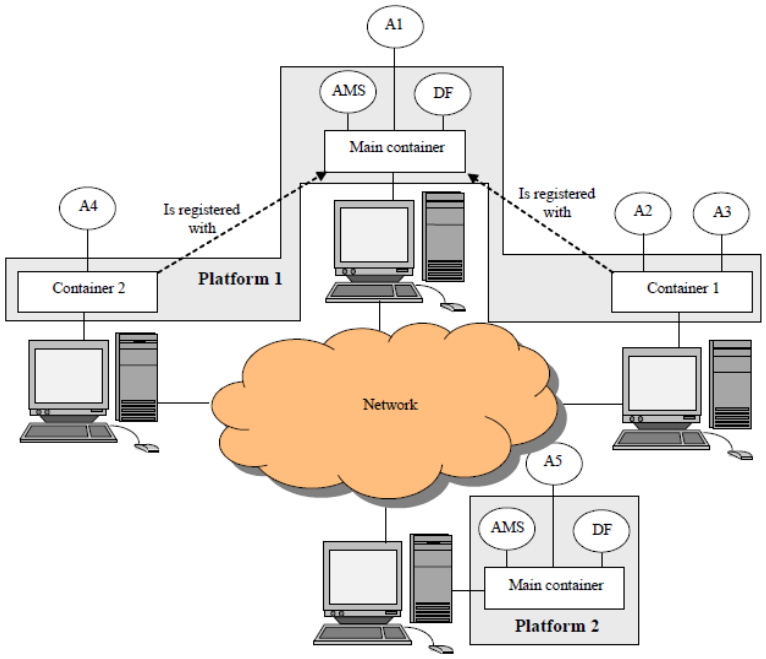
Η JADE είναι μία πλατφόρμα βασισμένη σε γλώσσα προγραμματισμού Java για την ανάπτυξη συστημάτων πολλαπλών πρακτόρων συμβατών με τον FIPA. Η πλατφόρμα εμπεριέχει ένα σύνολο τάξεων (classes) που επιτρέπουν στο χρήστη να παράγει ένα σύστημα πρακτόρων φύσεως πολλαπλών νημάτων (multi-threaded). Επιπλέον περιέχει ένα σύνολο εργαλείων για την δοκιμή της λειτουργικότητας των επιμέρους πρακτόρων και γενικότερα του συστήματος σαν σύνολο.

Εν αντιθέσει με τη Java όπου τα αντικείμενα διαθέτουν μεθόδους για να εκτελέσουν δράσεις, η JADE παρέχει ένα σύνολο συμπεριφορών όπου μπορούν να αξιοποιηθούν για εκτελέσει ένας ή πολλοί πράκτορες την επιθυμητή συμπεριφορά. Συνοπτικά, τα επιμέρους κύρια στοιχεία της εν λόγω πλατφόρμας είναι :

<u>Containers και Πλατφόρμες</u>

Κάθε παράδειγμα εκτέλεσης ενός κώδικα Java στο περιβάλλον JADE πραγματοποιείται σε Container από την στιγμή που μπορεί να εμπεριέχονται πολλοί πράκτορες σε μία εφαρμογή. Το σύνολο των ενεργών Containers καλείται Πλατφόρμα. Ένα ειδικό Main Container πρέπει πάντα να είναι ενεργό και όλα τα άλλα κατοχυρώνονται στη συνέχεια από την στιγμή που ενεργοποιείται η εκκίνηση τους από τον χρήστη. Τα δευτερεύοντα Containers πρέπει να είναι σε θέση, μέσω του χρήστη, να επικοινωνούν με το Κύριο (Main) Container που θα κατοχυρωθούνε.

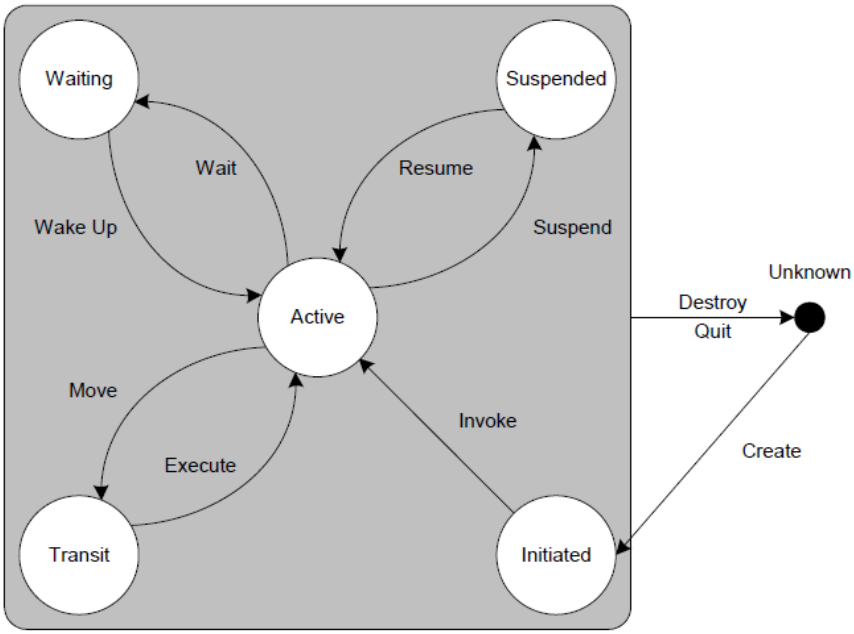
Το ακόλουθο σχήμα απεικονίζει δύο πλατφόρμες JADE που αποτελούνται από 3 και 1 Containers αντίστοιχα. Οι πράκτορες αναγνωρίζονται από ένα μοναδικό όνομα και μπορούν να επικοινωνούν μεταξύ τους άσχετα με την πραγματική τους τοποθεσία (ίδιο ή διαφορετικό Container).



Σχήμα 5.14: Containers και Πλατφόρμες [2]

Αναγνώριση Πρακτόρων

Ο κάθε πράκτορας αναγνωρίζεται από έναν agent identifier όπου αναπαριστάται με ανάκληση της τάξης jade.core.AID. Η μέθοδος getAID() της τάξης του πράκτορα επιτρέπει την ανάκτηση του agent identifier. Ένα αντικείμενο AID συμπεριλαμβάνει ένα μοναδικό όνομα συν έναν αριθμό διευθύνσεων. Οι διευθύνσεις αυτές είναι της πλατφόρμας που ο πράκτορας υπάρχει και ενεργεί και χρησιμοποιούνται στην περίπτωση που ένας πράκτορας πρέπει να επικοινωνήσει με κάποιον άλλο σε άλλη Πλατφόρμα.



Σχήμα 5.15 : Κύκλος ζωής Πράκτορα κατά FIPA [2]

Κάθε πράκτορας ενεργοποιείται πρωτίστως από τη γραμμή εντολών, ανεξαρτήτως ανάπτυξης κάποιου κώδικα, αλλά και διαμέσου αλληλεπίδρασης με το χρήστη από το GUI (Graphical User Interface) που διαθέτει κατά την εκτέλεση του προγράμματος.
Οι αναπτυχθέντες πράκτορες μπορούν να τερματίσουν τη δράση τους διαμέσου της μεθόδου doDelete(). Επιπλέον σε κάθε αναπτυσσόμενο σύστημα πρακτόρων ο χρήστης μπορεί να ορίσει συγκεκριμένες «συζητήσεις» και δράσεις μεταξύ τους για την εκτέλεση της εφαρμογής. Στην περίπτωση αυτή γίνεται χρήση της μεθόδου getArguments().
Κατά την ανάθεση εργασιών στους πράκτορες σε εφαρμογές σύνθετης και ανακύπτουσας πολυπλοκότητας γίνεται χρήση ειδικών τάξεων Συμπεριφορών. Είναι δυνατό στα πλαίσια της JADE να τρέχουνε διάφορες συμπεριφορές ταυτόχρονα. Ωστόσο, το πόσο αποτελεσματικά θα εκτελεστεί και προγραμματιστεί μία εφαρμογή με βάση ταυτόχρονες διαφορετικές συμπεριφορές τίθεται στην ευχέρεια του χρήστη κάνοντας χρήση μεθόδων JAVA τόσο για την ενεργοποίηση όσο και για το μπλοκάρισμα των εφαρμοσμένων συμπεριφορών.
Παρακάτω, περιγράφονται συνοπτικά οι σημαντικότερες εφαρμογές συμπεριφορών.

<u>One Shot Behaviour</u>
Η συγκεκριμένη συμπεριφορά υποδηλώνει δράσεις που εκτελούνται μία φορά και τερματίζονται. Μπορεί να χρησιμοποιηθεί πρωτίστως για την εισαγωγή παραμέτρων σε πράκτορες όταν αυτοί ξεκινάνε τις δράσεις τους, ή για το σχηματισμό υπο – ρουτίνων της Σειριακής συμπεριφοράς όπως αυτή θα αναπτυχθεί.

<u>Cyclic Behaviour</u>
Η συγκεκριμένη συμπεριφορά είναι μία από τις πιο κοινά χρησιμοποιούμενες συμπεριφορές σχετικά με την επικοινωνία πρακτόρων. Για την αποδοχή μηνύματος, μία κυκλική συμπεριφορά μπορεί να εφαρμόζεται ακόμα και όταν δεν εμφανίζεται η δράση τους, αναμένοντας ένα νέο μήνυμα. Από την στιγμή αποδοχής του μηνύματος, ενεργοποιείται και επεξεργάζεται την πληροφορία του μηνύματος.

Ticker Behaviour

Άλλη μία κοινά χρησιμοποιούμενη συμπεριφορά είναι η TickerBehaviour η οποία εκτελεί περιοδικά μία δράση συγκεκριμένης διάρκειας. Αυτή η συμπεριφορά χρησιμεύει στην αποστολή μηνυμάτων σε άλλους πράκτορες, ενημερώνοντάς τους για την τρέχουσα κατάσταση του συστήματος.

Sequential Behaviour

Ο συγκεκριμένος τύπος συμπεριφοράς εφαρμόζεται όταν ο τύπος της δράσης που πρέπει να εκτελεστεί εμφανίζει συγκεκριμένη πολυπλοκότητα και απαιτείται μία σειρά εκτέλεσης μικρότερων δράσεων εντός αυτής. Στην περίπτωση που δύο πράκτορες χρειάζεται να συνεργαστούν για μία εργασία αλλά αναμένουν την ολοκλήρωση προηγούμενων σταδίων εκτέλεσης της εργασίας αυτής, η χρήση της SequentialBehaviour ενδείκνυται.

Επιπλέον, στα πλαίσια εφαρμογής κάποιας από τις παραπάνω συμπεριφορές μπορούν να εκτελεστούν συγκεκριμένα πρωτόκολλα επικοινωνίας όπως το Contract Net Protocol. **Στην περίπτωση της παρούσας εργασίας, με χρήση της SequentialBehaviour, εκτελούνται σειριακά πολλαπλά Contract Nets προκειμένου να έχουμε την επιθυμητή αλληλουχία επικοινωνιών και δράσεων μεταξύ του ρομπότ, των μηχανών και των ενδιάμεσων αποθηκών του Ευέλικτου Συστήματος Κατεργασιών.**

Ένα από τα πιο χρήσιμα στοιχεία της JADE είναι ο χειρισμός των μηνυμάτων που περνάνε μεταξύ των πρακτόρων χωρίς την ανάγκη γνώσης της απευθείας τοποθεσίας αυτών, καθώς και ο χειρισμός ανάλυσης πολύπλοκων Strings. Ο χειρισμός της τοποθεσίας του πράκτορα γίνεται από το AMS (Agent Management System) και το MTS (Message Transport System). Το περιεχόμενο του μηνύματος απαιτεί κωδικοποίηση και αποκωδικοποίηση σε μορφή string που γίνεται αυτόματα από τη JADE, εν αντιθέσει με τη JAVA, παρέχοντας ένα σύνολο οντολογιών όπου μπορεί να κωδικοποιήσει αυτό και να το εξάγει από strings.

Τέλος, η δοκιμή του συνολικού συστήματος πρακτόρων, αλλά και μεμονωμένα μπορεί να γίνει με τη χρήση συγκεκριμένων εργαλείων debugging, τα οποία εφαρμόζονται με τη μορφή πρακτόρων εντός της πλατφόρμας ώστε να συνυπάρχουν στο σύστημα με τους προς δοκιμή πράκτορες.

Συνοπτικά αναφέρονται τα ακόλουθα εργαλεία-πράκτορες.

Sniffer Agent

Το συγκεκριμένο εργαλείο επιτρέπει στο σχεδιαστή του συστήματος να δει συνολικά όλες τις επικοινωνίες μεταξύ των πρακτόρων της πλατφόρμας συμπεριλαμβανομένης της ροής επικοινωνίας και το περιεχόμενο των μηνυμάτων.

Introspector Agent

Το συγκεκριμένο εργαλείο παρέχει λεπτομέρειες σχετικές με τη λειτουργικότητα των πρακτόρων και τις συμπεριφορές που είναι ενεργές καθ' όλη τη διάρκεια της ύπαρξής τους στο σύστημα. Παρέχει επιπλέον χρήσιμες πληροφορίες σχετικές με το περιεχόμενο των εισερχόμενων μηνυμάτων που αναμένονται να επεξεργαστούν από τους πράκτορες.

Dummy Agent

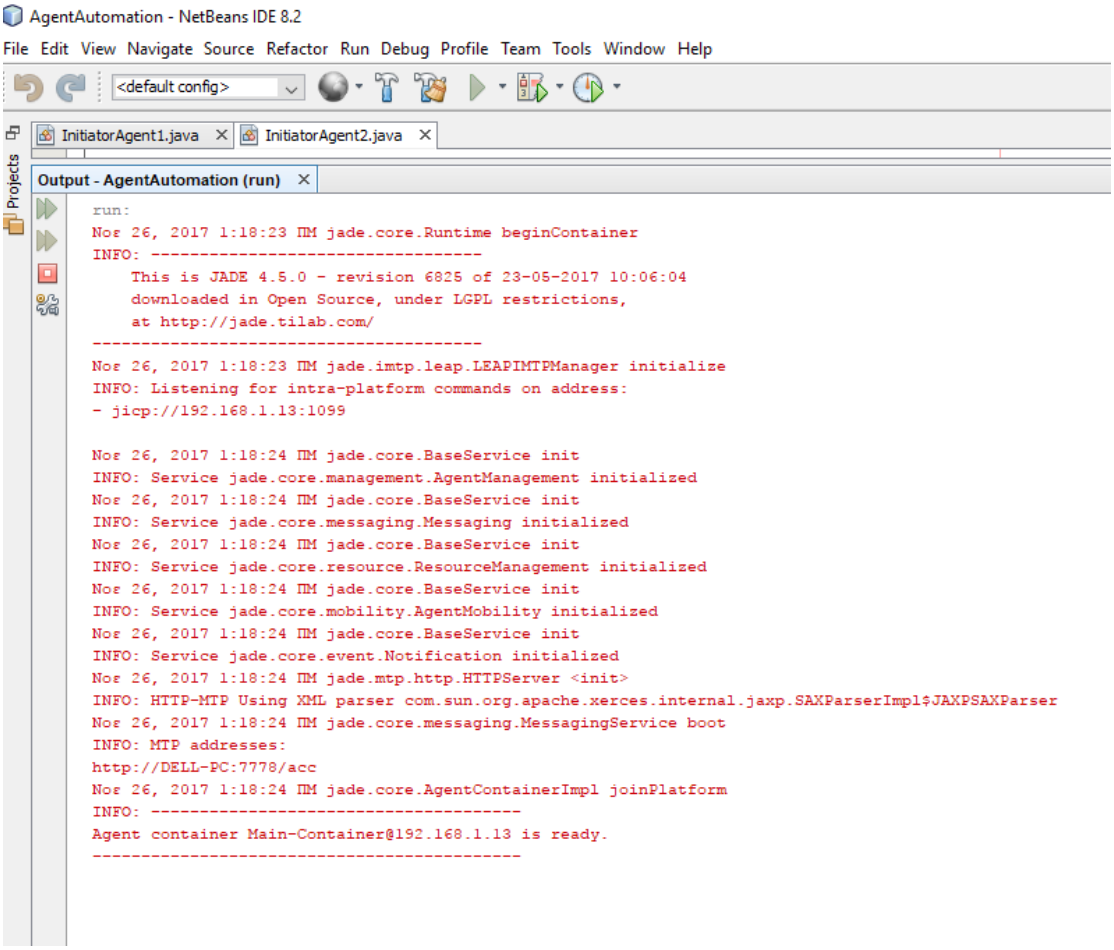
Για μία απλούστερη ανάλυση των ανταποκρίσεων ενός πράκτορα, το συγκεκριμένο εργαλείο παρέχει τη δυνατότητα δημιουργίας ερεθίσματος από το χρήστη με τη μορφή

μηνύματος, ούτως ώστε να δοκιμάσει και ελέγξει την εκδήλωση ανεπιθύμητων συμπεριφορών από τους πράκτορες.

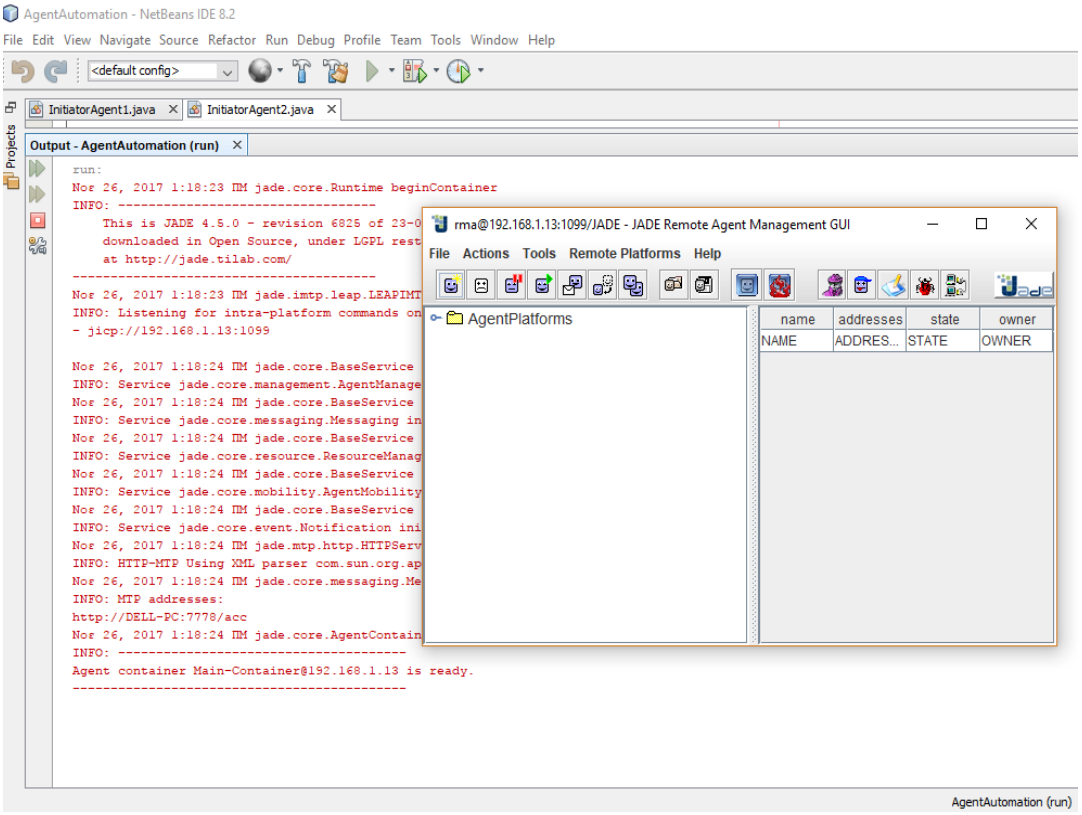
5.4 Υλοποίηση Εφαρμογής Πλαισίου JADE

Στην υλοποίηση της εφαρμογής όπως αυτή περιγράφηκε στην προηγούμενη παράγραφο καταλήγουμε στον προγραμματισμό μίας αρχικής αλληλεπίδρασης των πρακτόρων με τυχαίο τρόπο ούτως ώστε να καταλήξουμε στο σωστό ταίριασμα μεταξύ τους (πλειοδότες και διαχειριστές). Το περιεχόμενο των μηνυμάτων επικοινωνίας των πρακτόρων είναι σε μορφή XML,όπως αυτό μας δόθηκε από το πρόγραμμα PIPE2.
Για να επιτευχθεί όμως η σωστή αλληλουχία των απαραίτητων εργασιών του φυσικού συστήματος με επαναλαμβανόμενο τρόπο, ενσωματώσαμε τις αναπτυχθείσες τάξεις Java του πρωτόκολλου Contract Net σε σειριακή συμπεριφορά (Sequential Behaviour).
Ειδικότερα, το γραφικό περιβάλλον της εφαρμογής στο πλαίσιο της JADE και η αλληλεπίδρασή της με το χρήστη φαίνονται ακολούθως:
Θεωρούμε τα ονόματα των εμπλεκόμενων πρακτόρων A (Manager1), B (Manager2), C (Manager3), D (Manager4) για τους διαχειριστές και B1 (Bidder1), B2 (Bidder2), B3 (Bidder3) για τους πλειοδότες. Οι Διαχειριστές A και C αφορούν την επεξεργασία των τεμαχίων A και B που έχουν και την συντομότερη διαδρομή επεξεργασίας στο σύστημα με δύο εμπλεκόμενους πόρους (ρομπότ, μηχανή). Οι άλλοι δύο διαχειριστές B και D αφορούν την επεξεργασία των τεμαχίων Γ και Δ με την εμπλοκή των ενδιάμεσων αποθηκών Buffer2 (για αναμονή κατεργασίας στη φρέζα) και Buffer1 (για αναμονή κατεργασίας στο τόρνο) αντίστοιχα.

Κατά την επιλογή εκτέλεσης των τάξεων της JAVA ανοίγει το παράθυρο Remote Agent Management GUI της JADE όπως παρακάτω.

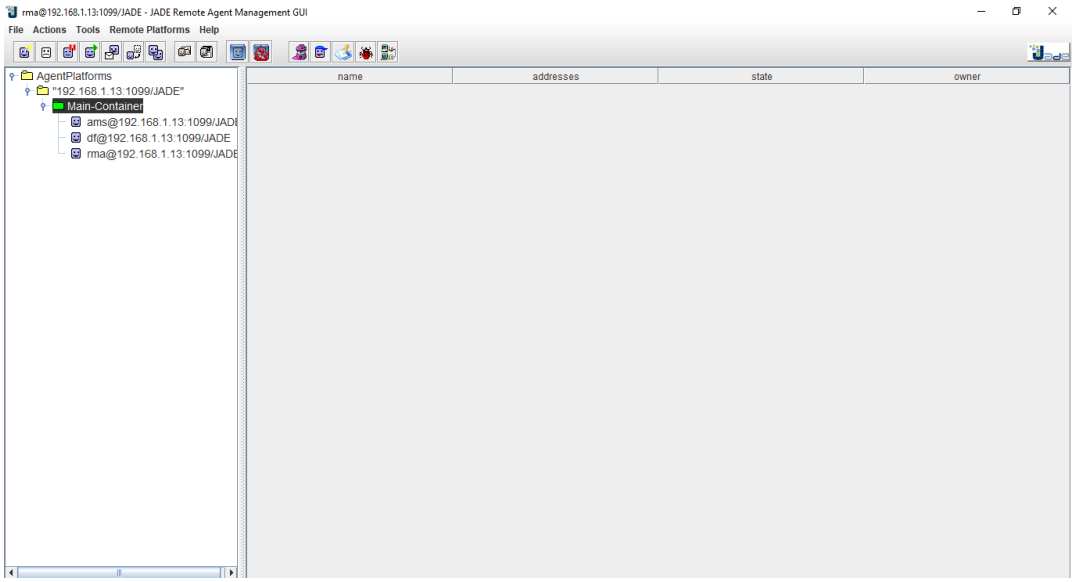


Σχήμα 5.16 : Εκκίνηση JADE



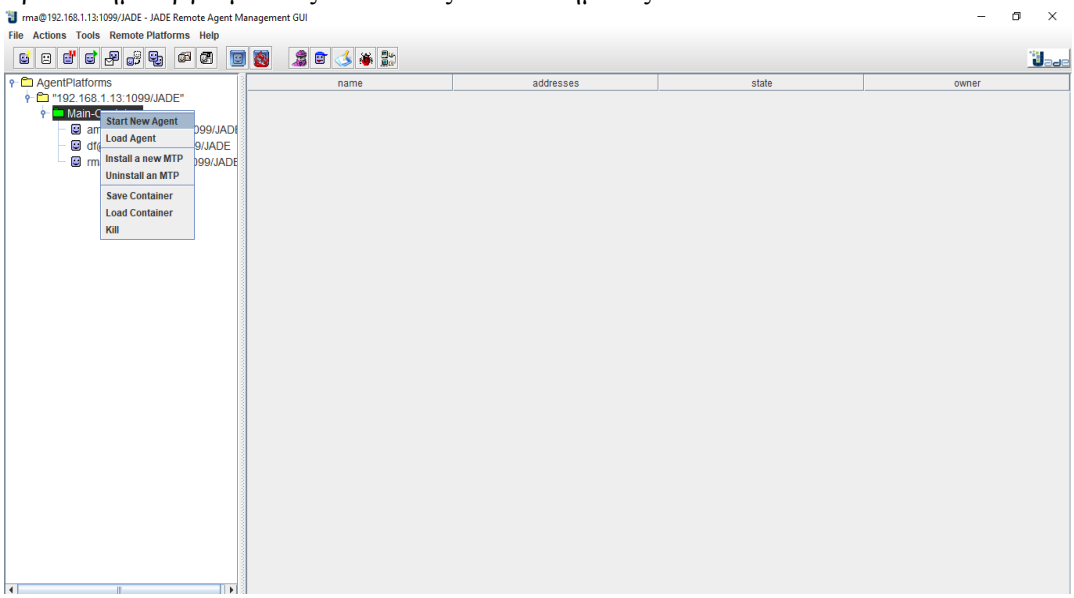
Σχήμα 5.17: JADE Remote Agent Management GUI

Από τη δεξιά στήλη διακρίνουμε τις πλατφόρμες των πρακτόρων καθώς και την τοπική πλατφόρμα και το Main Container που εμπεριέχεται σε αυτή.

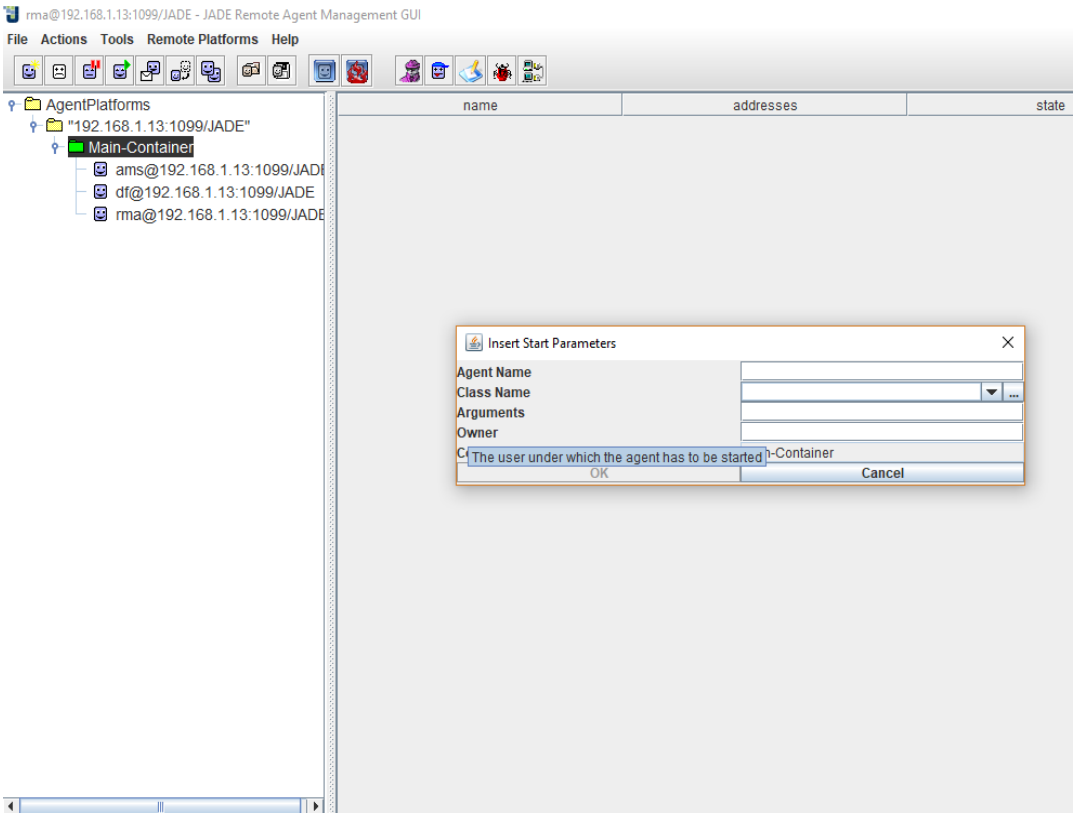


Σχήμα 5.18 : JADE Remote Agent Management GUI (b)

Στη φάση αυτή αρχίζει η δημιουργία των πρακτόρων μας, εντός του Main Container.
Στο Main Container εμπεριέχονται επίσης οι : Directory Facilitator (DF), Remote Management Agent (rma), Agent Management System (ams) που δεν θα μας απασχολήσουν στα πλαίσια της παρούσας εργασίας.
Πρώτα δημιουργούμε τους πλειοδότες του συστήματος

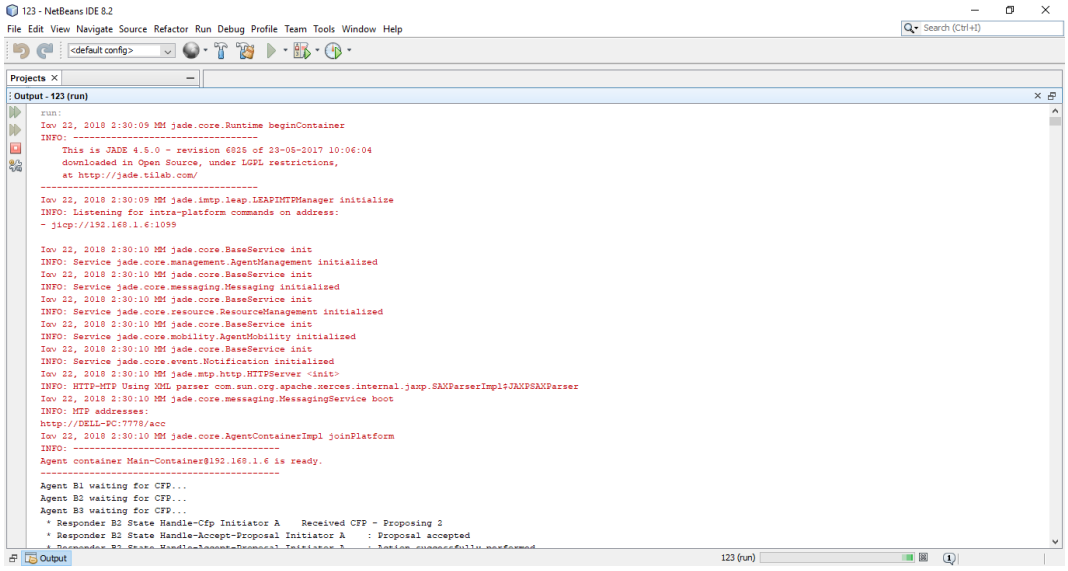


Σχήμα 5.19: JADE Remote Agent Management GUI (c)



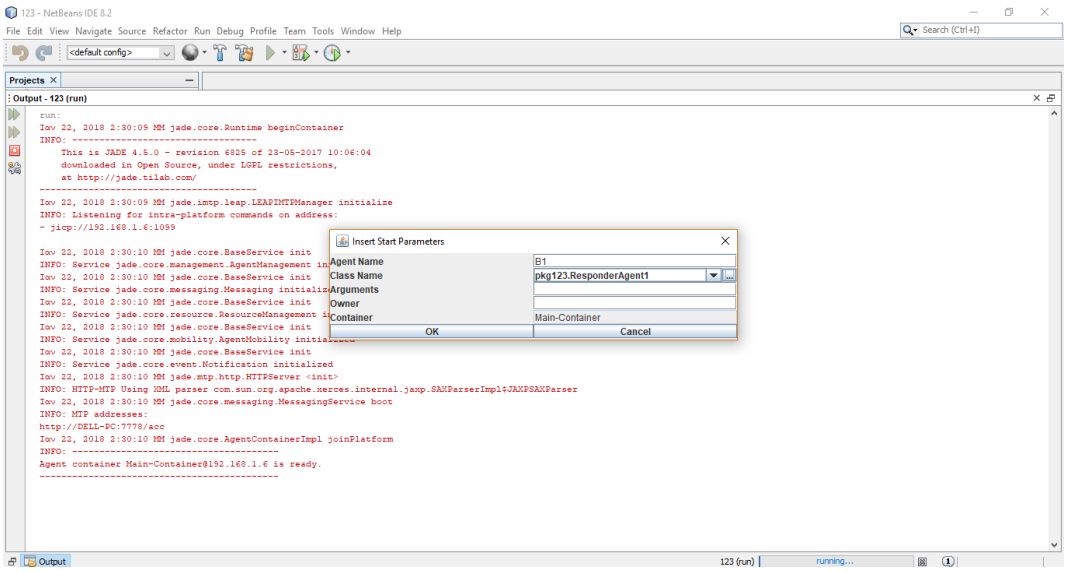
Σχήμα 5.20 : JADE Remote Agent Management GUI (d)

επιλέγοντας τις αντίστοιχες τάξεις της Java προς εκτέλεση.



Σχήμα 5.21: JADE Remote Agent Management GUI (e)

Στη συνέχεια δημιουργούμε και τους διαχειριστές του συστήματος επιλέγοντας τις αντίστοιχες τάξεις Java.

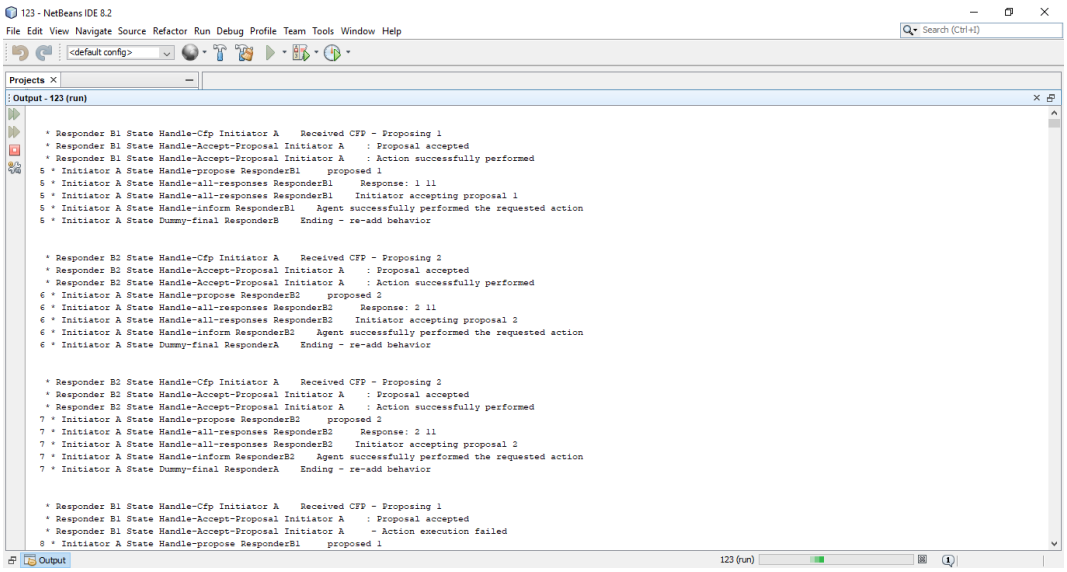


Σχήμα 5.22 : JADE Remote Agent Management GUI – Αντιστοιχία Πρακτόρων FMS

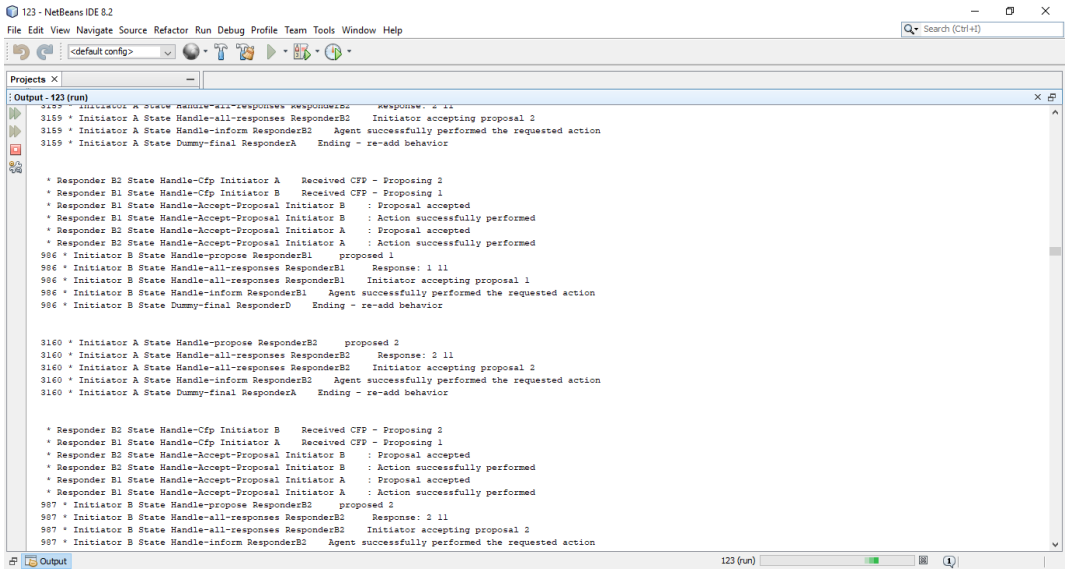
και τέλος δηλώνουμε τα arguments των διαχειριστών, ήτοι :

- Manager1(A): Bidder1(B1), Bidder(B2)
- Manager2(B): Bidder1(B1), Bidder2(B2), Bidder3(B3)
- Manager3(C): Bidder1(B1), Bidder2 (B2)
- Manager4(D): Bidder1(B1), Bidder2(B2), Bidder3(B3)

Με τη δήλωση των arguments ξεκινάει η εκτέλεση όλων των πρακτόρων και τρέχει επαναλαμβανόμενα επ' άπειρον μέχρις ότου διακοπεί από το χρήστη. Εναλλακτικά, θα πρέπει να δηλωθεί προγραμματιστικά σε ποια επανάληψη θα διακοπεί η λειτουργία όλου του συστήματος ή κάποιας ροής εργασίας αυτού (διαχειριστής).



Σχήμα 5.23: JADE Remote Agent Management GUI - Output



Σχήμα 5.24 : JADE Remote Agent Management GUI – Output (b)

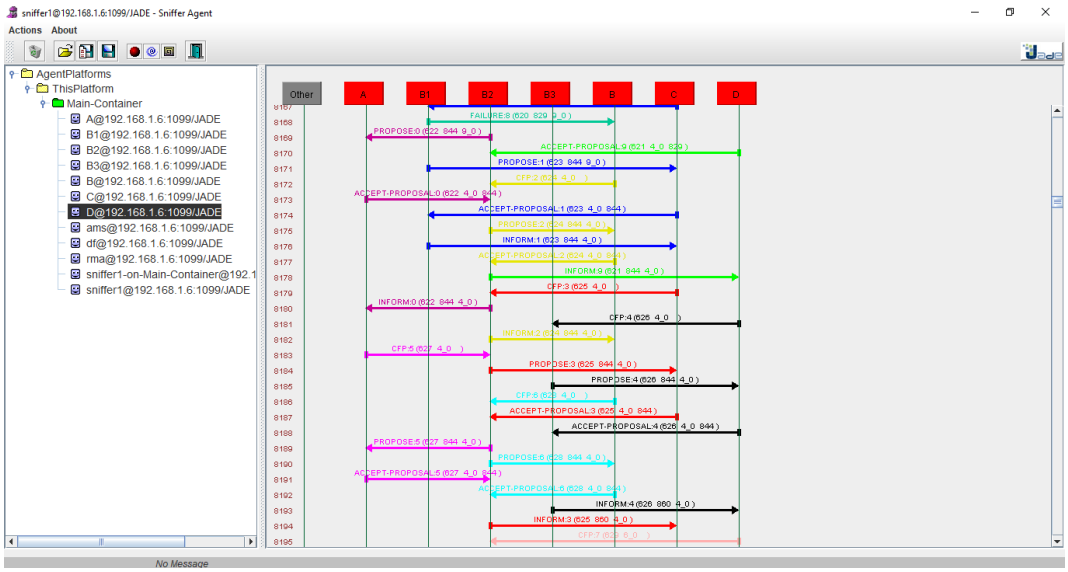
Τελικές αλληλεπιδράσεις\επικοινωνίες πρακτόρων συστήματος:
Manager1(A)→Bidder2(B2)→Manager1(A)→Bidder1(B1)→Manager1(A)
→Bidder2(B2)

Manager2(B)→Bidder2(B2)→Manager2(B)→Bidder1(B1)→Manager2(B)→
Bidder2(B2)→Manager2(B)→Bidder3(B3)→Manager2(B)→Bidder2(B2)→
Manager2(B)→Bidder1(B1)→Manager2(B) → Bidder2(B2)

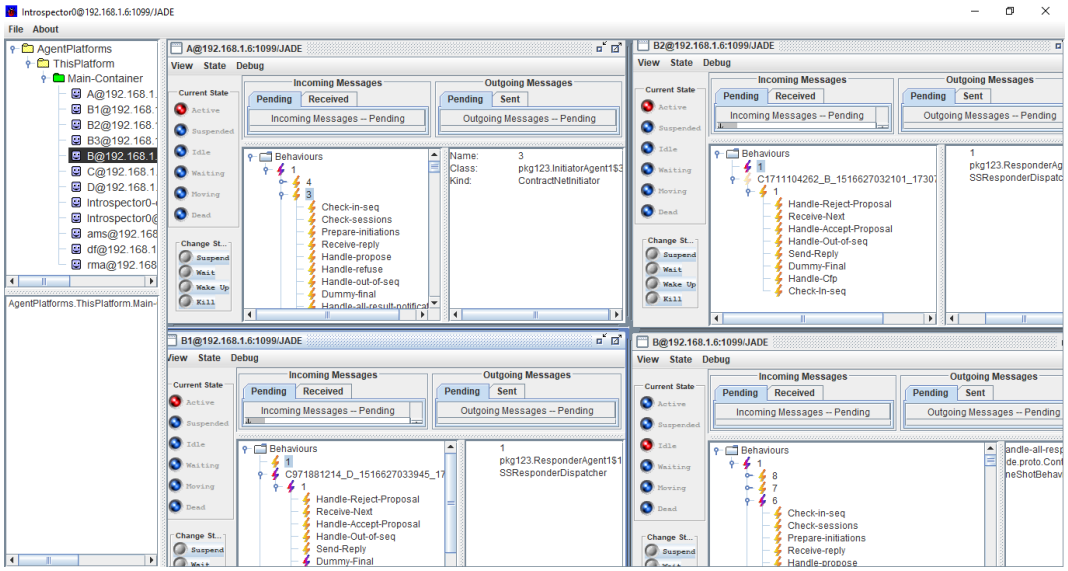
Manager3(C)→Bidder2(B2)→Manager3(C)→Bidder1(B1)→Manager3(C)
→Bidder2(B2)

Manager4(D)→Bidder2(B2)→Manager4(D)→Bidder1(B1)→Manager4(D)→
Bidder2(B2)→Manager4(D)→Bidder3(B3)→Manager4(D)→Bidder2(B2)→
Manager4(D)→Bidder1(B1)→Manager4(D) → Bidder2(B2)

Τέλος, ενδεικτικά παρουσιάζονται οι απεικονίσεις των βασικών εργαλείων-πρακτόρων debugging του αναπτυχθέντος συστήματος πολλαπλών πρακτόρων με βάση το CNP.



Σχήμα 5.25: Sniffer Agent



Σχήμα 5.26 : Introspector Agent

ΜΕΡΟΣ ΤΡΙΤΟ

ΚΕΦΑΛΑΙΟ 6 Συστήματα Κατανομής Πόρων

6.1 Εισαγωγή

Το πρόβλημα της διαχείρισης κατανομής πόρων αναδύεται σε μία πληθώρα σύγχρονων τεχνολογικών εφαρμογών, συμπεριλαμβανομένων των ευέλικτα αυτοματοποιημένων συστημάτων παραγωγής, των αυτοματοποιημένων σιδηροδρομικών και μονογραμμικών συστημάτων μεταφορών, των ηλεκτρονικών συστημάτων διαχείρισης ροής εργασιών αλλά και πιο πρόσφατα, των πολυπύρηνων υπολογιστικών αρχιτεκτονικών και των ανερχόμενων κβαντικών υπολογιστών. Βασικό χαρακτηριστικό αρκετών από τις προαναφερθείσες εφαρμογές είναι η ολοένα αυξανόμενη ανάγκη ελέγχου και συντονισμού σε πραγματικό χρόνο μέσα από ένα υπολογιστικό σύστημα ελέγχου. Αυτή η ανάπτυξη δικαιολογείται από έναν αριθμό τεχνικών και οικονομικών παραγόντων, συμπεριλαμβανομένης της αναγκαίας ασφάλειας, και διευκολύνεται από τις μοντέρνες υπολογιστικές και αισθητηριακές τεχνολογίες [12].

Η σύνθεση της λογικής ελέγχου που θα διαχειρίζεται την κατανομή των πόρων των παραπάνω εφαρμογών, διατηρώντας την απαραίτητη ευελιξία στις ποικίλες εκτελούμενες διεργασίες αυτών, είναι μία συνεχή πρόκληση, ερευνητικά και βιομηχανικά. Στη διεθνή βιβλιογραφία, έχουν προταθεί μία σειρά από συγκεκριμένες πολιτικές για την επαρκή και αποτελεσματική προσέγγιση της συγκεκριμένης αναδυόμενης τάξης λογικών προβλημάτων.

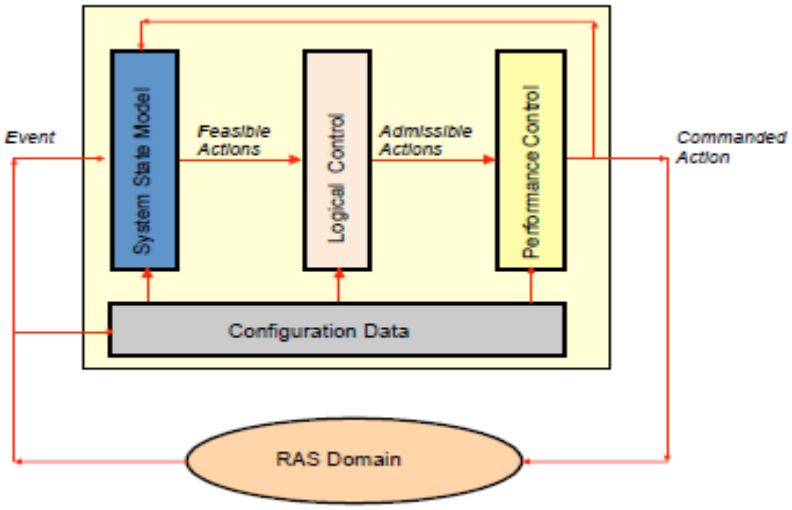
Η αποτελεσματική και συστηματική επίλυση των εμπλεκόμενων προβλημάτων αδιεξόδου στα εν λόγω συστήματα, πρέπει να βασιστεί σε λεπτομερή μελέτη των σειρών γεγονότων που λαμβάνουν χώρα κατά τη λειτουργία αυτών. Η ανάλυση και επίλυση αυτών των προβλημάτων απαιτεί ένα μεθοδολογικό πλαίσιο με έμφαση στην ανάλυση των σειρών γεγονότων του συστήματος, που διαφοροποιείται από τον παραδοσιακά απαιτούμενο έλεγχο απόδοσης αυτού. Αυτό το πλαίσιο έχει διαμορφωθεί από μία περιοχή της μοντέρνας θεωρίας ελέγχου που είναι γνωστή ως *ποιοτική ή λογική ανάλυση και έλεγχος Συστημάτων Διακριτού Γεγονότος (ΣΔΓ)*.

Χρησιμοποιώντας ευρέως τον όρο ‘Σειριακά Συστήματα Κατανομής Πόρων’ αλλά και άλλες τυποποιημένες παρουσιάσεις από την Θεωρία Συστημάτων Διακριτού Γεγονότος οι προαναφερθέντες αναπτύξεις έχουν οδηγήσει στα εξής :

- Ένα αναλυτικό χαρακτηρισμό της συγκεκριμένης τάξης προβλημάτων
- Την τυποποίηση μίας έννοιας βέλτιστου ελέγχου της αντίστοιχης δυναμικής
- Το χαρακτηρισμό της υπολογιστικής πολυπλοκότητας των βέλτιστων λύσεων
- Αποτελεσματικούς. και επαρκείς αλγορίθμους που μπορούν να παρέχουν βέλτιστη και υπο-βέλτιστη επίλυση προβλημάτων αδιεξόδου για κάθε πρακτική εκδήλωση αυτών από τις τάξεις των Συστημάτων Κατανομής Πόρων και παρά την ανακύπτουσα υπολογιστική πολυπλοκότητα
- Μεθοδολογική βάση που μπορεί να συνεισφέρει στον περαιτέρω αποτελεσματικό προγραμματισμό των ΣΚΠ.

Επιπλέον, με βάση τη θεωρία εποπτικού ελέγχου των ΣΔΓ, οι συνεισφορές όλων των παραπάνω οδηγούν σε ένα είδος αποτρεπτικού ελέγχου των τάξεων Συστημάτων Κατανομής Πόρων (περιορισμός δυναμικής συστήματος προκειμένου να μην δημιουργηθεί αδιέξοδο). Εν γένει, γίνεται προσπάθεια εντοπισμού ενός υποσυνόλου

αποδεκτής συμπεριφοράς από το αρχικό σύνολο της εφικτής συμπεριφοράς του συστήματος και κατόπιν έλεγχος αυτού με μέγιστα επιτρεπτό τρόπο.
Στο ακόλουθο σχήμα απεικονίζεται ολόκληρη η διαδικασία ελέγχου- βασισμένη σε γεγονότα, στα πλαίσια των Συστημάτων Κατανομής Πόρων.



Σχήμα 6.1 Διαδικασία Ελέγχου (Event-Driven) στα Συστήματα Κατανομής Πόρων [12]

Στο σχήμα μπορούμε να διακρίνουμε τον ελεγκτή να ανταποκρίνεται στα διάφορα γεγονότα που λαμβάνουν μέρος στο περιβάλλον του ελεγχόμενου ΣΚΠ αναβαθμίζοντας το μοντέλο κατάστασης που ορίζει την εφικτή συμπεριφορά του συστήματος. Αυτή η συμπεριφορά φιλτράρεται από τον λογικό ελεγκτή έτσι ώστε να πάρουμε την επιθυμητή συμπεριφορά – επιβαλλόμενη από την λειτουργία του συστήματος και την αποφυγή αδιεξόδου. Τέλος, η επιθυμητή συμπεριφορά επεξεργάζεται από τον ελεγκτή απόδοσης του συστήματος για να επιλεχθούν έτσι οι ειδικές δράσεις που θα επιβληθούν στο σύστημα.

6.2 Μοντελοποίηση Συστημάτων Κατανομής Πόρων

Ένα Σειριακό Σύστημα Κατανομής Πόρων ορίζεται από την 5-άδα Φ = {R, C, P, A, D} όπου:

- R = { R_1, R_2, R_3, ……R_m} είναι το σύνολο των τύπων των πόρων του συστήματος
- C:R → Z+ (σύνολο θετικών ακεραίων) – είναι η συνάρτηση χωρητικότητας του συστήματος, υποδηλώνοντας τον αριθμό των μονάδων του κάθε πόρου που είναι διαθέσιμα στο σύστημα. Οι πόροι θεωρείται ότι είναι επαναχρησιμοποιήσιμοι. Δηλαδή $C(R_i) = C_i$ αποτελεί ένα σύστημα αναλλοίωτο για κάθε i.
- P = {Π_1, Π_2, ……Π_n} υποδηλώνει τον τύπο των διεργασιών που υποστηρίζονται από το Σύστημα Κατανομής Πόρων. Ο κάθε τύπος επεξεργασίας Π_j είναι ένα σύνθετο στοιχείο αποτελούμενο από α.) το σύνολο των σταδίων επεξεργασίας Δ_j = { Ξ_{j1},Ξ_{j2},……Ξ_{jn}} που εμπλέκονται στον τύπο διεργασίας Π_j και β.) μία επιπρόσθετη δομή δεδομένων G_j που κωδικοποιεί τη σειριακή λογική που ολοκληρώνει το σύνολο των σταδίων επεξεργασίας Δ σε ένα σύνολο ενδεχομένων ροών επεξεργασίας.

- A: $\Delta \rightarrow \Pi_{im}$ { 0,……,C_i} είναι η συνάρτηση κατανομής πόρων σχετιζόμενη με κάθε στάδιο επεξεργασίας Ξ_{jk} και το διάνυσμα κατανομής πόρων $A(\Xi_{jk})$ διάφορο του μηδενός.
- D είναι η συνάρτηση καθορισμού κάθε σταδίου επεξεργασίας Ξ_{jk} στο Δ, σε μία κατανομή που χαρακτηρίζει το χρόνο επεξεργασίας του αντίστοιχου σταδίου επεξεργασίας.

Τέλος, θα μπορούσαμε να δηλώσουμε $\xi = |\Delta|$ και να ορίσουμε το μέγεθος Φ του ΣΚΠ ως $|\Phi| = |R| + \xi + \Sigma_i^m C_i$ για λόγους πολυπλοκότητας .

6.3 Ταξινόμηση Συστημάτων Κατανομής Πόρων

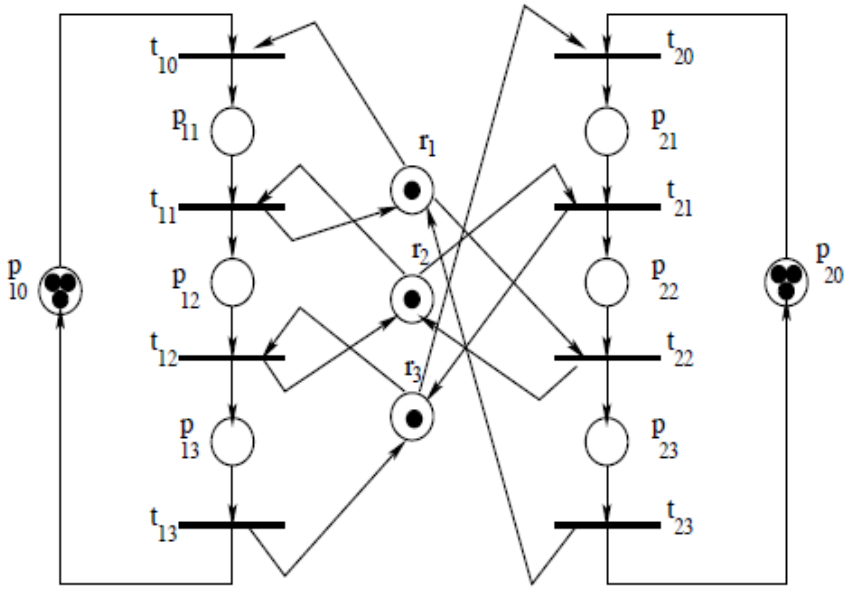
Τα κυρίαρχα από πλευράς εφαρμογών ΣΚΠ μπορούν να ταξινομηθούν ανάλογα με τον προσδιορισμό των συστατικών τους μερών ως ακολούθως.

Based on the structure of the Process Sequential Logic	Based on the structure of the Requirement Vectors
Linear: Each process is defined by a linear sequence of stages **Disjunctive:** A number of alternative process plans encoded by an acyclic digraph **Merge-Split:** Each process is a fork-join network **Complex:** A combination of the above behaviors	**Single-Unit:** Each stage requires a single unit from a single resource **Single-Type:** Each stage requires an arbitrary number of units, but all from a single resource **Conjunctive:** Stages require different resources at arbitrary levels

Πίνακας 6.1 Κατάταξη Συστημάτων Κατανομής Πόρων [12]

6.3.1 Συζευγμένα – Από-συζευγμένα ΣΚΠ (Conjunctive – Disjunctive RAS)

Στο σημείο αυτό θα εστιάσουμε πρωταρχικά σε μία τάξη ΣΚΠ γνωστή ως Συζευγμένα-Από-συζευγμένα ΣΚΠ, ιδιότητες των οποίων θα αναφερθούν και στο υπόλοιπο κομμάτι της παρούσας ενότητας. Με από-συζευγμένο τρόπο, στον τύπο αυτών των συστημάτων η συνάρτηση κατανομής πόρων επιβάλει περιπτώσεις διεργασίας του ίδιου τύπου πόρου να εκτελούνται σε διαφορετικές σειρές των σταδίων επεξεργασίας και η εκτελούμενη σειρά να συμβαίνει δυναμικά με βάση τους διαθέσιμους πόρους. Με συζευγμένο τρόπο, στις απαιτήσεις των πόρων των διάφορων σταδίων επεξεργασίας, επιβάλλεται ότι αυτές θα είναι ένα αυθαίρετα δομημένο (πολύ-) υποσύνολο του R.
Η δομή δεδομένων G_j των Σ-Α ΣΚΠ που ορίζει τη σειριακή λογική των τύπων επεξεργασίας Π_j , θα είναι ένα μη κυκλικό διπλό γράφημα με το σύνολο των κόμβων να είναι ίσο με το αντίστοιχο σύνολο των σταδίων επεξεργασίας Δ_j. Η κάθε κατευθυνόμενη άκρη (Ξ_{jk}, Ξ_{jq}) του γραφήματος επιβάλει ότι το στάδιο επεξεργασίας Ξ_{jq} μπορεί να είναι ένα άμεσο διαδοχικό στάδιο του σταδίου επεξεργασίας Ξ_{jk} και το αντίστροφο [12].
Ένα ενδεικτικό Σ-Α ΣΚΠ , διατυπωμένο σε δίκτυα Petri, απεικονίζεται στο ακόλουθο σχήμα.



Σχήμα 6.2 Σύστημα Κατανομής Πόρων

Γενικά , ένα ΣΚΠ χωρίζεται σε θέσεις λειτουργίας p_s (p_{11},p_{12},p_{13},p_{21},p_{22},p_{23}) πόρων p_R (r_1, r_2, r_3) και εισόδων p_i (p_{10},p_{20}) , όπως διακρίνεται και στο παραπάνω δίκτυο. Επιπλέον, τα εν λόγω δίκτυα Petri συναντώνται στη βιβλιογραφία και ως S^3PR (System of Simple Sequential Processes).

6.4 Αδιέξοδο Συστήματος Κατανομής Πόρων και Πολιτικές Αποφυγής Αδιεξόδου (DAPs)

Στα πλαίσια των Συστημάτων Κατανομής Πόρων ο κύριος λόγος της μη ασφάλειας αυτών είναι ο σχηματισμός μερικών αδιεξόδων (partial deadlocks). Αυτό συμβαίνει συνήθως όταν κάποια στάδια επεξεργασίας του συστήματος απαιτούν πόρους για την εκτέλεση αυτής, αναμένοντας την διαθεσιμότητά τους από κάποιο/α άλλο/α στάδιο/α επεξεργασίας. Στο Ευέλικτο Σύστημα Κατεργασιών που μελετάμε στην παρούσα εργασία αυτό μπορεί να συμβεί όταν, για παράδειγμα, με γεμάτες τις μηχανές επεξεργασίας το ρομπότ πάει να φορτώσει τεμάχιο σε μία από αυτές. Εναλλακτικό σενάριο αποτελεί η γεμάτη ενδιάμεση αποθήκη λόγω επεξεργασίας τεμαχίου στην αντίστοιχη μηχανή και η εμπλοκή του ρομπότ στην μεταφορά τεμαχίου προς και από αυτήν.
Για την αποφυγή τέτοιων ανεπιθύμητων καταστάσεων, στη διεθνή βιβλιογραφία, έχουν προταθεί συγκεκριμένες πολιτικές αποφυγής αδιεξόδου (DAP). Στην παρούσα εργασία εστιάζουμε σε μία συγκεκριμένη ομάδα DAP που είναι γνωστές ως (Αλγεβρικές) Πολιτικές Αποφυγής Αδιεξόδου Πολυωνυμικού Πυρήνα (Algebraic PK-DAPs) [9][12].
Στο πλαίσιο μοντελοποίησης ενός ΣΚΠ με δίκτυα Petri, ο σχηματισμός deadlock έγκειται στην έλλειψη αντιστρεψιμότητας του δικτύου. Επιπλέον, στο χώρο προσβασιμότητας $R(N,M_0)$, αυτή η έλλειψη γραφικά απεικονίζεται με το σχηματισμό ισχυρά συνδεδεμένων στοιχείων που δεν είναι συν-προσβάσιμα, δηλαδή η αρχική σήμανση M_0 δεν είναι προσβάσιμη από αυτά διαμέσου κάποιων εφικτών μεταβάσεων. Με βάση αυτό, μία ορθή πολιτική αποφυγής deadlock Δ θα πρέπει να περιορίσει τη λειτουργία του συστήματος σε ένα ισχυρά συνδεδεμένο στοιχείο του υποκείμενου χώρου προσβασιμότητας που θα περιλαμβάνει την αρχική σήμανση M_0.

Ο χώρος προσβασιμότητας είναι κατά βάση πεπερασμένος και έτσι η βέλτιστη πολιτική είναι καλά ορισμένη. Το υποκείμενο πρόβλημα ασφάλειας είναι NP-complete και η ανάπτυξη υπο-βέλτιστων DAPs που είναι εφαρμόσιμες με πολυώνυμη πολυπλοκότητα είναι επαρκείς. Από πλευράς σχεδιασμού αυτών των πολιτικών γίνεται η αναγνώριση μίας ιδιότητας H(M) , όπου M είναι η σήμανση που ανήκει στο R, τέτοια ώστε α.) η πολυπλοκότητα της δοκιμής H() στις σημάνσεις του ΣΚΠ είναι πολυώνυμη με βάση το μέγεθος του ΣΚΠ, β.) $H(M_0)$ = TRUE και γ.) ο υποχώρος {M που ανήκει στο $R(N,M_0)$: H(M) = TRUE} είναι ισχυρά συνδεδεμένος[11].

6.4.1 Αλγεβρικές DAPs και η αναπαράστασή τους με Δίκτυα Petri

Οι αλγεβρικές PK-DAPs ορίζονται σαν μία ειδική τάξη PK-DAPs όπου η ιδιότητα H(M) αποτελεί ένα σύστημα γραμμικών ανισοτήτων στις σημάνσεις M του ΣΚΠ που είναι πολυώνυμες, σε μέγεθος σύμφωνα με το μέγεθος του ΣΚΠ [12]. Οι ανισότητες αυτές θα περιορίζουν τη σήμανση M ακριβώς στα στοιχεία του ΣΚΠ, εν αντιστοιχία με τις θέσεις του δικτύου που αναπαριστούν τις λειτουργίες του συστήματος. Μία τυπική αλγεβρική PK-DAP έχει τη μορφή

$$A \times M_s \leq b$$

όπου A είναι μη αρνητική ακέραιη μήτρα με K γραμμές, το b είναι K- διαστάσεων μη αρνητικό ακέραιο διάνυσμα, και το K είναι πολυωνυμικά σχετικό με το μέγεθος του ΣΚΠ. Το M_s είναι οι σημάνσεις των θέσεων p_s του δικτύου που απεικονίζουν τις λειτουργίες του συστήματος.

Η αναπαράσταση της DAP στο δίκτυο Petri γίνεται με την επιβολή των ανισοτήτων

$$A(k, .) \times M_s \leq b(k) , \qquad k= 1,2,3,……..K$$

στην συμπεριφορά του συστήματος τοποθετώντας στο αρχικό δίκτυο μία θέση ελέγχου $p_c(k)$ (εικονική θέση) . Η συνδεσιμότητα της θέσης $p_c(k)$ με το υπόλοιπο δίκτυο ορίζεται από την μήτρα ροής

$$\theta_{pc(k)} = - A (k, .) \times \Theta_s$$

που Θ_s υποδηλώνει την υπο – μήτρα ροής του μη ελέγξιμου δικτύου Φ εν αντιστοιχία με τις θέσεις λειτουργίας p_s. Η αρχική σήμανση της θέσης ελέγχου $p_c(k)$ ορίζεται ως

$$M_0 (p_c(k)) = b(k)$$

και ο ελεγκτής που προκύπτει επιβάλει στην αρχική συμπεριφορά του συστήματος το σύστημα ανισοτήτων διαμέσου του P-αναλλοίωτου

$$A(k, .) \times M_s + M_0 (p_c(k)) = b(k)$$

6.4.2 Ανάλυση και Σχεδιασμός PK–DAPs μέσω δομικής ανάλυσης Δικτύων Petri

Σε αυτή την ενότητα εστιάζουμε στο δομικό χαρακτηρισμό της ορθότητας της DAP και του ΣΚΠ που χαρακτηρίζεται ελεύθερο από deadlock. Αυτό θα επιδιωχθεί με την αρίθμηση όλου του υποκείμενου χώρου καταστάσεων. Βασική έννοια που θα μας

απασχολήσει ακολούθως είναι η αντιστρεψιμότητα του δικτύου που χαρακτηρίζεται από τα εξής:

- Ένα δίκτυο Petri N είναι αντιστρέψιμο εάν και μόνο αν είναι ζωντανό
- Το N είναι ζωντανό εάν και μόνο αν ο χώρος R(N, M_0) των προσβάσιμων σημάνσεων δεν περιέχει άδειο σιφώνιο S.

Η αξιολόγηση των όρων της ζωτικότητας (liveness) και αντιστρεψιμότητας (reversibility) που αναφέρονται στα παραπάνω δύο κριτήρια μπορεί να γίνει με την εφαρμογή τυποποίησης μαθηματικού προγραμματισμού. Δεδομένου ότι ένα δίκτυο Petri N είναι δομικά φραγμένο, η τυποποίηση παίρνει τη μορφή μεικτού ακέραιου προγραμματισμού. Θεωρώντας ότι S είναι το μέγιστο άδειο σιφώνιο στο δίκτυο N η εύρεση αυτού στο N είναι η επίλυση του προβλήματος μεικτού ακέραιου προγραμματισμού [14] .
Θέτουμε v_p=1 για κάθε θέση p που δεν ανήκει στο S και z_t=1 για κάθε μετάβαση t που δεν ανήκει στο S. Εάν για κάθε μετάβαση εξόδου t μιας θέσης p έχουμε v_p=0 τότε έχουμε και z_t=0. Αντίστοιχα ,εάν για κάθε θέση εξόδου p μίας μετάβασης t έχουμε z_t=1, συνεπάγεται ότι και v_p=1 . Αυτά οδηγούν στην ακόλουθη τυποποίηση :

$$z_t \geq \sum_{p \in {}^\bullet t} v_p - |{}^\bullet t| + 1, \forall t \in T,$$

$$v_p \geq z_t, \forall (t,p) \in F,$$

$$v_p, z_t \in \{0,1\}.$$

όπου F είναι τα βάρη του δικτύου N. Για δομικά φραγμένο δίκτυο επιπλέον ισχύει ότι

$$v_p \geq M(p)/\pi(p), \forall p \in P$$

όπου π(p) είναι το δομικό όριο/φραγή της θέσης p με

$$\pi(p) = max\{M(p) | M = M_0 + [N]Y, M \geq 0, Y \geq 0\}$$

Το μέγιστο μη σημασμένο σιφώνιο μπορεί να εντοπιστεί από το ακόλουθο πρόβλημα μεικτού ακέραιου προγραμματισμού αντικειμενικής συνάρτησης

$$G^{MIP}(M_0) = min \sum v_p \quad (p \epsilon P)$$

με περιορισμούς

$$z_t \geq \sum_{p \in {}^\bullet t} v_p - |{}^\bullet t| + 1, \forall t \in T,$$

$$v_p \geq z_t, \forall (t,p) \in F,$$

$$v_p, z_t \in \{0,1\}.$$

$$M = M_0 + [N]Y, M \geq 0, Y \geq 0$$

Στο δίκτυο Petri θα υπάρχουν σιφώνια μη σημασμένα κάθε φορά που

$$G^{MIP}(M_0) < |P|$$

Η συγκεκριμένη τυποποίηση θα εφαρμοστεί στο δίκτυο N ούτως ώστε να εντοπιστεί ένα μέγιστα άδειο σιφώνιο. Εφόσον αυτό υπάρχει υπολογίζουμε το ελάχιστο σιφώνιο αυτού και κατόπιν το συμπληρωματικό αυτού, εισάγοντας έναν επιπλέον περιορισμό στον επόμενο κύκλο εφαρμογής της τυποποίησης. Ο καινούργιος περιορισμός θα έχει τη μορφή

$$M([S]) \leq M_0(S) - 1$$

όπου $M_0(S)$ είναι η αρχική σήμανση του ελάχιστου άδειου σιφωνίου που μας έδωσε ο πρώτος κύκλος εφαρμογής του μεικτού ακέραιου προγραμματισμού και [S] το συμπληρωματικό σιφώνιο αυτού. Ο υπολογισμός των ελάχιστων άδειων σιφωνίων εξαγόμενων από τα μέγιστα αυτών μας δίνεται από συγκεκριμένα υπολογιστικά εργαλεία όπως το PIPE2 που χρησιμοποιείται στην παρούσα εργασία.

Ο υπολογισμός των συμπληρωματικών σιφωνίων δίνεται στο παράρτημα της παρούσας εργασίας.

Εισάγοντας, κάθε φορά που η τυποποίηση μας δίνει ένα μέγιστο άδειο σιφώνιο, έναν επιπλέον περιορισμό, θα καταλήξουμε σε κάποιο κύκλο εφαρμογής της που η αντικειμενική συνάρτηση θα είναι ίση με τον αριθμό των θέσεων P του δικτύου. Στο στάδιο αυτό η περαιτέρω εφαρμογή της τυποποίησης είναι ανέφικτη και έχουμε οδηγηθεί στην εξαγωγή όλων των αναγκαίων σιφωνίων που πρέπει να ελεγχθούν με μία επιπλέον θέση στο δίκτυο (monitors).

Με αυτό τον τρόπο, οδηγούμαστε στη διαμόρφωση ενός συγκεκριμένου ζευγαριού (A,b), των προαναφερθέντων γραμμικών ανισοτήτων στην εισαγωγή αυτής της ενότητας, που καθιστούν την παραπάνω τυποποίηση ανέφικτη.

Κατά συνέπεια όλων των παραπάνω, θα οδηγηθούμε στην ανάπτυξη συγκεκριμένης APK-DAP με συγκεκριμένες γραμμικές ανισότητες της μορφής :

$$A \times M_s \leq b$$

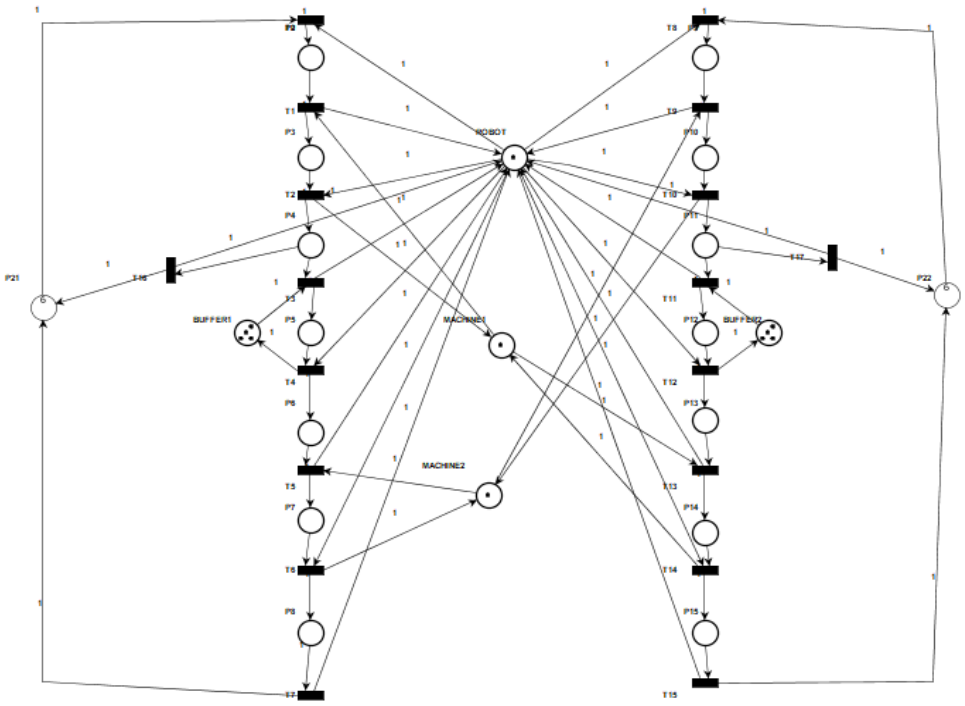
εξασφαλίζοντας τις απαιτήσεις ζωτικότητας και αντιστρεψιμότητας του δικτύου.

ΚΕΦΑΛΑΙΟ 7: Ανάπτυξη αλγεβρικής πολιτικής αποφυγής αδιεξόδου για το FMS του Εργαστηρίου

Στο κεφάλαιο αυτό επιδιώκουμε την απαιτούμενη ανάλυση σκοπιμότητας των συνεργατικών δικτύων που διαμορφώσαμε στο Κεφάλαιο 5 της παρούσας εργασίας, για να ολοκληρωθεί ο agent – based ελεγκτής. Θα προσεγγίσουμε τα συνεργατικά δίκτυα του 5ου Κεφαλαίου ως Συστήματα Κατανομής Πόρων. Στη συνέχεια, θα γίνει εφαρμογή τεχνικών του προηγούμενου κεφαλαίου για την ανάπτυξη συγκεκριμένης Αλγεβρικής Πολιτικής Αποφυγής Αδιεξόδου που θα εξασφαλίζει την απαιτούμενη ζωτικότητα και αντιστρεψιμότητα του τελικού δικτύου.

7.1 Προσέγγιση Συνεργατικών Δικτύων ως Σύστημα Κατανομής Πόρων

Ένα συνεργατικό δίκτυο είναι εφικτό όταν εξασφαλίζεται η ζωτικότητα αυτού. Στο Κεφάλαιο 5 οι πλειοδότες υποδηλώνανε τα σύνολα των διατιθέμενων πόρων του συστήματος και οι διαχειριστές τις ροές εργασιών των εμπλεκόμενων έργων. Στη θεωρία των Συστημάτων Κατανομής Πόρων, όπως αναλύθηκε και στο προηγούμενο κεφάλαιο, υπάρχουν οι εργασίες (Jobs) του συστήματος διατυπωμένες σε δίκτυα Petri και οι διατιθέμενοι πόροι κατάλληλα ενταγμένοι σε αυτά με τη μορφή θέσεων. Οι ροές εργασιών των έργων που αναπτύξαμε στο Κεφάλαιο 5 με δίκτυα Petri είναι ακριβώς, σε αντιστοιχία, οι εργασίες του συστήματος με βάση την θεωρία Συστημάτων Κατανομής Πόρων. Οι μεταβάσεις και οι θέσεις που υποδηλώθηκαν μπορούν να είναι ακριβώς οι ίδιες και με την προσέγγιση αυτής της θεωρίας. Οι εμπλεκόμενοι πόροι μπορούν επίσης να αναπαρασταθούν όπως οι bidders στα συγχωνευμένα δίκτυα του κεφαλαίου 5, με μοναδική διαφορά την διάσπαση του Bidder 1 σε 2 θέσεις δικτύου απεικονίζοντας τις δύο διαφορετικές μηχανές. Με βάση τη δομή των **Συζευγμένων-Από-συζευγμένων ΣΚΠ με τη μορφή Δικτύων Petri** το ευέλικτο σύστημα κατεργασιών του εργαστηρίου μπορεί να απεικονισθεί ως εξής :



Σχήμα 7.1 Ευέλικτο Σύστημα Κατεργασιών του Εργαστηρίου σαν ΣΚΠ

Όπου $\{p_{21}, p_{22}\}$ οι αποθήκες εισόδου και $\{p_2\text{-}p_{15}\}$ οι θέσεις λειτουργίας του συστήματος κατεργασίας. Η ακριβής αντιστοιχία προσέγγισης του ευέλικτου συστήματος κατεργασιών σαν Σύστημα Κατανομής Πόρων με τα αναπτυγμένα, από-συζευγμένα, συνεργατικά δίκτυα εφαρμογής του Πρωτόκολλου Contract Net, μας δίνει τη δυνατότητα να εξασφαλίσουμε τη ζωτικότητα του συστήματος με χρήση εργαλείων από το επιστημονικό πεδίο της Θεωρίας Διακριτού Γεγονότος.

7.2 Ανάπτυξη Αλγεβρικής Πολιτικής Αποφυγής Αδιεξόδου (Deadlock)

Η ανάγκη εξασφάλισης ζωτικότητας του αναπτυχθέντος Συστήματος Κατανομής Πόρων μπορεί να επιτευχθεί με την επιβολή Αλγεβρικής PK- DAP. Για να γίνει αυτό αναγνωρίζουμε πρώτα τα σιφώνια του ΣΚΠ. Το εργαλείο PIPE2 μπορεί να μας δώσει όλα τα ελάχιστα σιφώνια και παγίδες του συστήματος.

Minimal Siphons and Minimal Traps

Minimal siphons

- {P5_Default, BUFFER_1_Default }
- {ROBOT_Default, MACHINE_1_Default, P11_Default, P15_Default, P4_Default, P6_Default,
- P8_Default, P9_Default }
- {MACHINE_2_Default, P10_Default, P7_Default }
- {P3_Default, MACHINE_1_Default, P14_Default }

- {P11_Default, P10_Default, P12_Default, P13_Default, P14_Default, P15_Default, P22_Default, P9_Default }
- {P12_Default, BUFFER_2_Default }
- {P2_Default, ROBOT_Default, P11_Default, P13_Default, P15_Default, P4_Default, P6_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, P11_Default, MACHINE_2_Default, P13_Default, P15_Default, P4_Default, P8_Default }
- {ROBOT_Default, MACHINE_1_Default, P11_Default, MACHINE_2_Default, P15_Default, P4_Default, P8_Default }
- {P2_Default, P21_Default, P3_Default, P4_Default, P5_Default, P6_Default, P7_Default, P8_Default }

Minimal traps

- {MACHINE_2_Default, P10_Default, P7_Default }
- {P11_Default, P10_Default, P12_Default, P13_Default, P14_Default, P15_Default, P22_Default, P9_Default }
- {P3_Default, MACHINE_1_Default, P14_Default }
- {P5_Default, BUFFER_1_Default }
- {P12_Default, BUFFER_2_Default }
- {P2_Default, P21_Default, P3_Default, P4_Default, P5_Default, P6_Default, P7_Default, P8_Default }
- {P2_Default, ROBOT_Default, P11_Default, P13_Default, P15_Default, P4_Default, P6_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, P11_Default, BUFFER_2_Default, P15_Default, P4_Default, P6_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, P11_Default, BUFFER_2_Default, P6_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, P11_Default, BUFFER_2_Default, BUFFER_1_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, P11_Default, BUFFER_2_Default, P15_Default, P4_Default, BUFFER_1_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, P11_Default, P13_Default, P6_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, P11_Default, P13_Default, BUFFER_1_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, P11_Default, P13_Default, P15_Default, P4_Default, BUFFER_1_Default, P8_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_2_Default, P13_Default, P15_Default, P4_Default, P6_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_2_Default, BUFFER_2_Default, P15_Default, P4_Default, P6_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, MACHINE_2_Default, BUFFER_2_Default, P6_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, MACHINE_2_Default, BUFFER_2_Default, BUFFER_1_Default, P9_Default }

- {P2_Default, ROBOT_Default, MACHINE_2_Default, BUFFER_2_Default, P15_Default, P4_Default, BUFFER_1_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, MACHINE_2_Default, P13_Default, P6_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_1_Default, MACHINE_2_Default, P13_Default, BUFFER_1_Default, P9_Default }
- {P2_Default, ROBOT_Default, MACHINE_2_Default, P13_Default, P15_Default, P4_Default, BUFFER_1_Default, P9_Default }

Analysis time: 0.443s

Εν συνεχεία, εφαρμόζουμε την τυποποίηση μαθηματικού προγραμματισμού όπως αυτή αναπτύχθηκε στο προηγούμενο κεφάλαιο. Για τη εφαρμογή και επίλυση της μαθηματικής τυποποίησης χρησιμοποιήσαμε το εργαλείο LINDO 6.1. Αφού το παραπάνω ΣΚΠ είναι δομικά φραγμένο (σύμφωνα με το PIPE2), προχωράμε στην ακόλουθη εφαρμογή της τυποποίησης μεικτού ακέραιου προγραμματισμού του Κεφαλαίου 6 για την εύρεση του μέγιστου άδειου σιφωνίου.

MIN
VP2+VP3+VP4+VP5+VP6+VP7+VP8+VP9+VP10+VP11+VP12+VP13+VP14+VP15+VP16+VP17+VP18+VP19+VP20+VP21+VP22

SUBJECT TO

ZT0-VP21-VP18>=-1	ZT14-VP14-VP18>=-1	VP11-ZT10>=0
ZT1-VP2-VP16>=-1	ZT15-VP15>=0	VP12-ZT11>=0
ZT2-VP3-VP18>=-1	ZT16-VP4>=0	VP13-ZT12>=0
ZT3-VP4-VP19>=-1	ZT17-VP11>=0	VP14-ZT13>=0
ZT4-VP5-VP18>=-1	VP21-ZT7>=0	VP15-ZT14>=0
ZT5-VP6-VP17>=-1	VP2-ZT0>=0	VP16-ZT2>=0
ZT6-VP7-VP18>=-1	VP3-ZT1>=0	VP16-ZT14>=0
ZT7-VP8>=0	VP4-ZT2>=0	VP17-ZT6>=0
ZT8-VP22-VP18>=-1	VP5-ZT3>=0	VP17-ZT10>=0
ZT9-VP9-VP17>=-1	VP6-ZT4>=0	VP19-ZT4>=0
ZT10-VP10-VP18>=-1	VP7-ZT5>=0	VP20-ZT12>=0
ZT11-VP11-VP20>=-1	VP8-ZT6>=0	VP22-ZT15>=0
ZT12-VP12-VP18>=-1	VP9-ZT8>=0	VP22-ZT17>=0
ZT13-VP13-VP16>=-1	VP10-ZT9>=0	VP21-ZT16>=0

VP18-ZT1>=0

VP18-ZT3>=0

VP18-ZT5>=0

VP18-ZT7>=0

VP18-ZT9>=0

VP18-ZT11>=0

VP18-ZT13>=0

VP18-ZT15>=0

VP18-ZT16>=0

VP18-ZT17>=0

6VP21-MP21>=0

VP2-MP2>=0

VP3-MP3>=0

VP4-MP4>=0

VP5-MP5>=0

VP6-MP6>=0

VP7-MP7>=0

VP8-MP8>=0

VP9-MP9>=0

VP10-MP10>=0

VP11-MP11>=0

VP12-MP12>=0

VP13-MP13>=0

VP14-MP14>=0

VP15-MP15>=0

VP16-MP16>=0

VP17-MP17>=0

VP18-MP18>=0

4VP19-MP19>=0

4VP20-MP20>=0

6VP22-MP22>=0

MP21+Y0-Y7-Y16=6

MP2+Y1-Y0=0

MP3+Y2-Y1=0

MP4+Y3-Y2=0

MP5+Y4-Y3=0

MP6+Y5-Y4=0

MP7+Y6-Y5=0

MP8+Y7-Y6=0

MP9+Y9-Y8=0

MP10+Y10-Y9=0

MP11+Y11-Y10=0

MP12+Y12-Y11=0

MP13+Y13-Y12=0

MP14+Y14-Y13=0

MP15+Y15-Y14=1

MP16+Y1+Y13-Y2-Y14=1

MP17+Y5+Y9-Y6-Y10=1

MP19+Y3-Y4=4

MP20+Y11-Y12=4

MP22+Y8-Y15-Y17=6

MP18+Y0+Y2+Y4+Y6+Y8+Y10
+Y12+Y14-Y1-Y3-Y5-Y7-Y9-
Y11-Y13-Y15-Y16-Y17=1

END

INT VP2

INT VP3

INT VP4

INT VP5

INT VP6

INT VP7

INT VP8

INT VP9

INT VP10

INT VP11

INT VP12

INT VP13

INT VP14

INT VP15

INT VP16

INT VP17

INT VP18

INT VP19

INT VP20

INT VP21

INT VP22

INT ZT1

INT ZT2

INT ZT3

INT ZT4

INT ZT5

INT ZT6

INT ZT7

INT ZT8

INT ZT9

INT ZT10

INT ZT11

INT ZT12

INT ZT13

INT ZT14

INT ZT15

INT ZT16

INT ZT17

η οποία μας δίνει το μέγιστο άδειο σιφώνιο:

S = {p_2, p_4, p_5, p_6, p_7, p_8, p_{11}, p_{12}, p_{14}, p_{15}, p_{16} (MACHINE_1), p_{17} (MACHINE_2), p_{18}(ROBOT)}.

Από το μέγιστο άδειο σιφώνιο εξάγουμε το ελάχιστο σιφώνιο : S_1 = {p_4, p_8, p_{11}, p_{15}, p_{16} (MACHINE_1), p_{17} (MACHINE_2), p_{18} (ROBOT)}, το οποίο αντιστοιχεί σε ένα από τα απόλυτα ελάχιστα σιφώνια που εξάγαμε με το εργαλείο PIPE2, και υπολογίζουμε το συμπληρωματικό αυτού [S_1] = {p_2, p_3, p_6, p_7, p_9, p_{10}, p_{13}, p_{14}} .

M_0 (S_1) = 3 (από τη σήμανση των πόρων p_{16} (MACHINE_1), p_{17} (MACHINE_2), p_{18} (ROBOT))

και

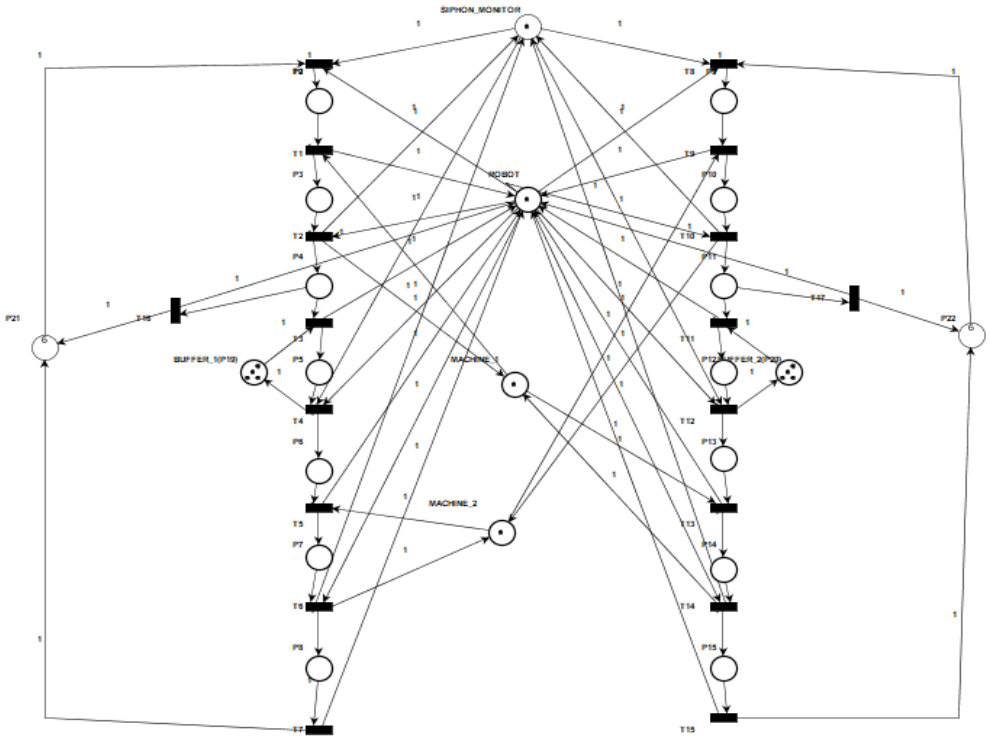
M ([S_1]) = Mp_2+Mp_3+Mp_6+Mp_7+Mp_9+Mp_{10}+Mp_{13}+Mp_{14}

Συνεπώς, εφαρμόζοντας την ανάπτυξη Αλγεβρικής Πολιτικής Αποφυγής Αδιεξόδου του Κεφαλαίου 6 έχουμε :

$M([S_1]) \leq M_0(S_1) - 1$

$Mp_2+Mp_3+Mp_6+Mp_7+Mp_9+Mp_{10}+Mp_{13}+Mp_{14} \leq 2$

Εφαρμόζοντας πάλι την τυποποίηση του μεικτού ακέραιου προγραμματισμού με επιπλέον περιορισμό την παραπάνω ανισότητα καταλήγουμε σε εξίσωση της αντικειμενικής συνάρτησης με το σύνολο των θέσεων του δικτύου. Συνεπώς, η περαιτέρω εφαρμογή της τυποποίησης είναι ανέφικτη και το σύστημα χαρακτηρίζεται αντιστρέψιμο και ζωντανό αν και μόνο αν ελεγχθεί το σιφώνιο S_1. Το σιφώνιο S_1 μπορεί να ελεγχθεί με την τοποθέτηση κατάλληλου monitor στο δίκτυο του ΣΚΠ.
Το άθροισμα των μεταβάσεων εισόδου (θετικές) και εξόδου (αρνητικές) όλων των θέσεων του σιφωνίου μας οδηγεί στην εξαγωγή όλων των μεταβάσεων που θα εισέρχονται και εξέρχονται της θέσης του monitor. Καταλήγουμε στην ακόλουθη ενσωμάτωση του monitor στο δίκτυο Petri του ΣΚΠ του ευέλικτου συστήματος κατεργασιών του εργαστηρίου.



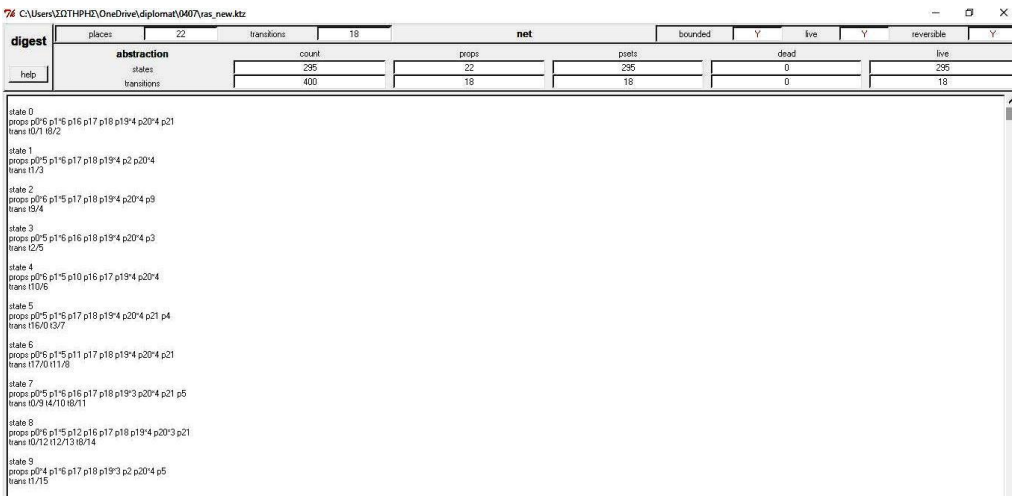
Σχήμα 7.2 Ευέλικτο Σύστημα Κατεργασιών σαν Ελεγχόμενο ΣΚΠ (Siphon-based Control)

Το οποίο μας διασφαλίζει την ζωτικότητα και αντιστρεψιμότητα του δικτύου διαμέσου της επιβολής PK –DAP που έχει την ακόλουθη μορφή :

$$[\,1\,1\,0\,0\,1\,1\,0\,1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,] \times Mp_i \leq 2 \ ,$$

με $Mp_i = [Mp_2, \ldots\ldots\ldots Mp_{22}]^T$

Στο ακόλουθο σχήμα επιβεβαιώνεται η εξασφάλιση της ζωτικότητας και αντιστρεψιμότητας του τελικού ελεγχόμενου δικτύου.



Σχήμα 7.3 Επιβεβαίωση Ζωτικότητας και Αντιστρεψιμότητας του Δικτύου του σχήματος 7.2

ΚΕΦΑΛΑΙΟ 8: Ανάλυση Αποτελεσμάτων

Στο κεφάλαιο αυτό θα παρουσιαστεί αναλυτικά η εξέλιξη των επικοινωνιών των διαμορφωμένων πρακτόρων των προηγούμενων κεφαλαίων προς επιβεβαίωση της λειτουργίας του τελικού ελεγκτή. Οι επικοινωνίες μεταξύ των πρακτόρων παρουσιάζονται στο Output της εφαρμογής που αναπτύχθηκε στη Java, όπως αυτή έτρεξε στο περιβάλλον NetBeans IDE 8.2 με χρήση βιβλιοθηκών του πλαισίου ανάπτυξης πρακτόρων JADE (Java Agent Development).

Για την πιο αποτελεσματική εκτέλεση του κώδικα και προς αποφυγή συγκρούσεων κατά τη δημιουργία των σχετικών επικοινωνιών ανάμεσα στους πράκτορες δημιουργήσαμε μία ενιαία τάξη της Java που αφορά τον πυρήνα της λειτουργίας όλων των Διαχειριστών (InitiatorAgent.java) σύμφωνα με το πρωτόκολλο Contract Net. Επιπλέον, δημιουργήθηκε μία ενιαία τάξη και για την λειτουργία των Πλειοδοτών (ResponderAgent.java) του Συστήματος Πολλαπλών Πρακτόρων που αναπτύχθηκε.

Η εκτέλεση και αλληλεπίδραση των παραπάνω ενιαίων τάξεων θα ολοκληρωθεί με μία τρίτη τάξη της Java η οποία εμπεριέχει την main μέθοδο και δημιουργήθηκε για καθαρά προγραμματιστικούς λόγους.

Όπως προαναφέρθηκε στο προηγούμενο κεφάλαιο Πολλαπλά Πρωτόκολλα Contract Net θα εφαρμοσθούν σειριακά διαμέσου της εφαρμογής συγκεκριμένης συμπεριφοράς στα πλαίσια της JADE. Πιο συγκεκριμένα θα γίνει χρήση της SequentialBehaviour class της JADE μέσω της οποίας μπορούν να εφαρμοσθούν διαδοχικά πολλαπλά πρωτόκολλα Contract Net με τη μορφή υπο-συμπεριφοράς (subBehaviour). Με τον τρόπο αυτό επιδιώκεται διαδοχικά το σωστό ταίριασμα Διαχειριστή – Πλειοδότη σύμφωνα με τις απαιτήσεις των εργασιών του φυσικού συστήματος.

Η εφαρμογή των παραπάνω σε συνδυασμό με την ενσωμάτωση των αναγκαίων, για τη διαπραγμάτευση πλειοδότη-διαχειριστή, σύνθετων συνθηκών στην ενιαία τάξη των Διαχειριστών μας οδηγεί στην έκβαση των ακόλουθων επικοινωνιακών αποτελεσμάτων.

Προς καλύτερη απεικόνιση και οπτική κατανόηση των αποτελεσμάτων θεωρούμαι ότι οι Διαχειριστές είναι οι:

InitiatorAgent A για τον Manager 1,

InitiatorAgent B για τον Manager 2,

InitiatorAgent F για τον Manager 3, και

InitiatorAgent G για τον Manager 4

Καθώς και

ResponderAgent 1: για τον Bidder 1

ResponderAgent 2: για τον Bidder 2

ResponderAgent 3: για τον Bidder 3

Οι κύκλοι επεξεργασίας του κάθε τεμαχίου (Α,Β,Γ,Δ) που πρέπει να ολοκληρωθούν επαναλαμβανόμενα είναι :

Initiator A αφορά το Τεμάχιο Α →Αποδοχή των Proposals : 2 - 1 - 2

Initiator B αφορά το Τεμάχιο Γ →Αποδοχή των Proposals : 2 - 1 - 2 - 3 - 2 - 1 - 2

Initiator F αφορά το Τεμάχιο Β →Αποδοχή των Proposals : 2 - 1 - 2

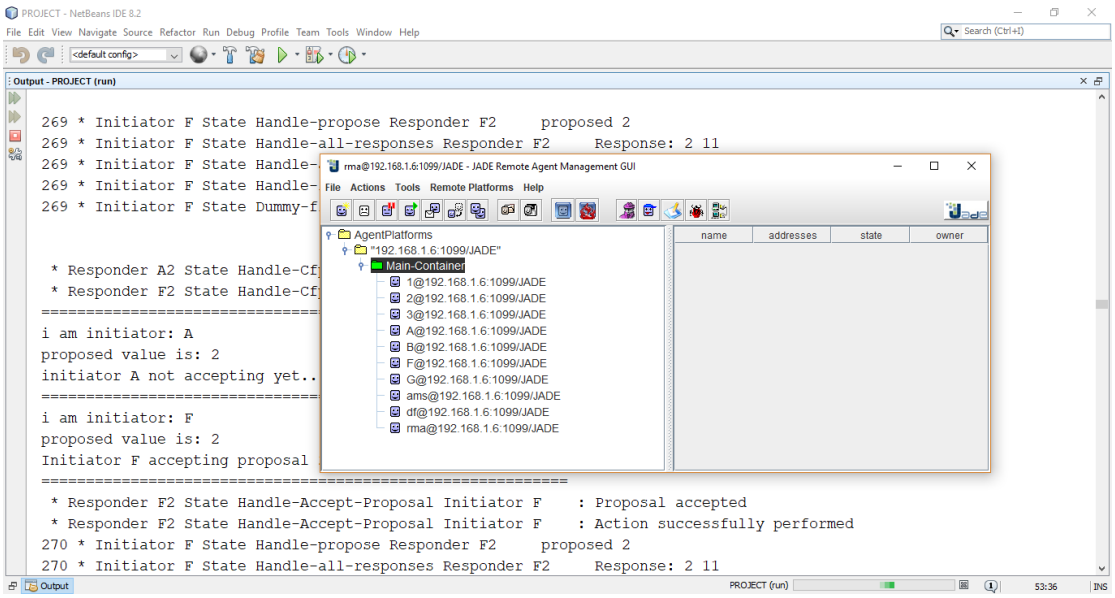
Initiator G αφορά το Τεμάχιο Δ →Αποδοχή των Proposals : 2 - 1 - 2 - 3 - 2 - 1 - 2

Αποδοχή της Proposal 1 : Αποδοχή Φρέζας ή Τόρνου

Αποδοχή της Proposal 2 : Αποδοχή του Ρομπότ

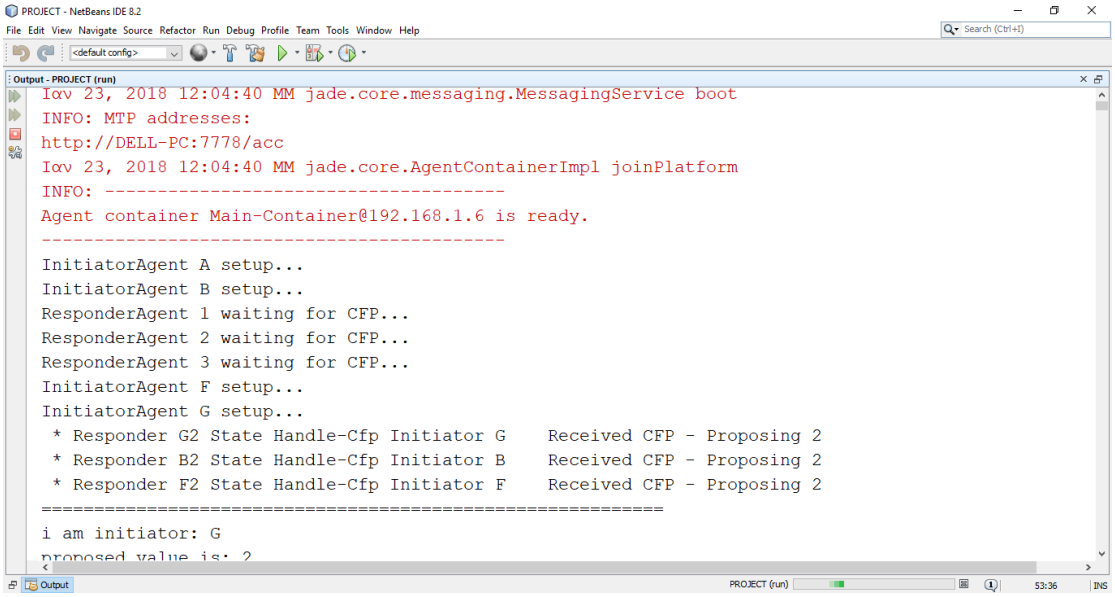
Αποδοχή της Proposal 3 : Αποδοχή Θέσεως στην Ενδιάμεση Αποθήκη

- *Εκκίνηση JADE (εκτέλεση της main method)*



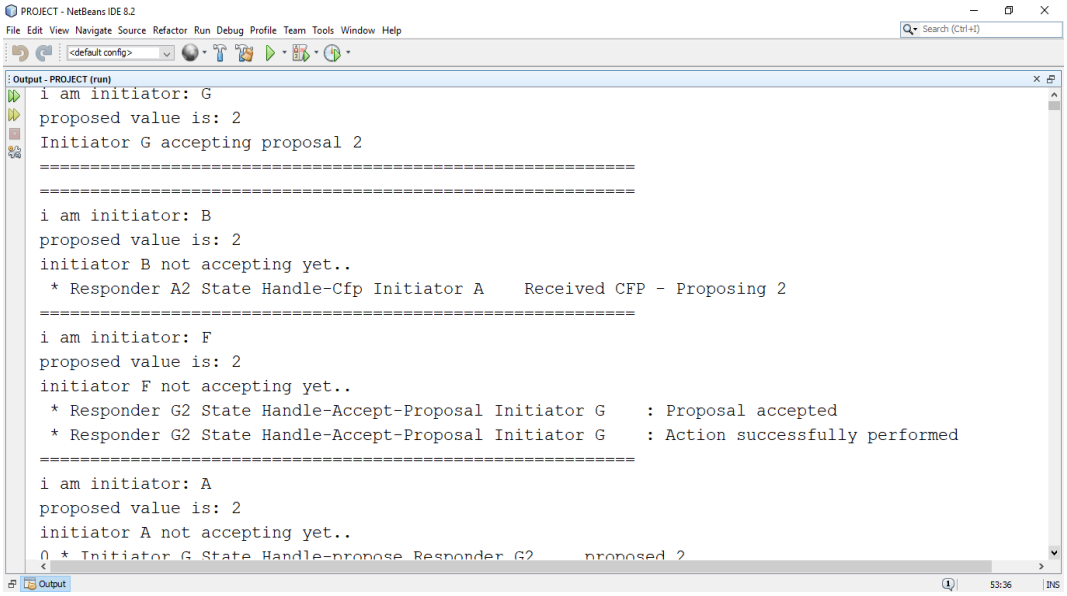
Σχήμα 8.1 Εκκίνηση JADE

- *Εκκίνηση των Πρακτόρων (Initiators και Responders) και αποστολή των πρώτων CFPs (Call For Proposals)*



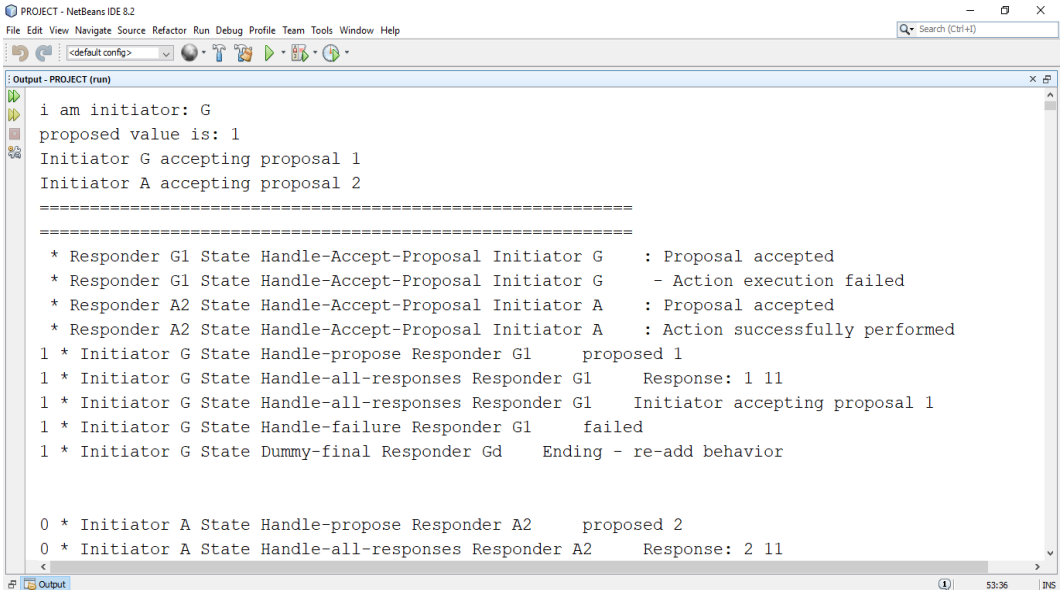
Σχήμα 8.2 Εκκίνηση Πρακτόρων

- *Διαμόρφωση 1της Επικοινωνίας :*
  Ο G αποδέχεται την proposal 2 – το ρομπότ
  Οι A,B και F αναμένουν τη σύνδεση με το ρομπότ



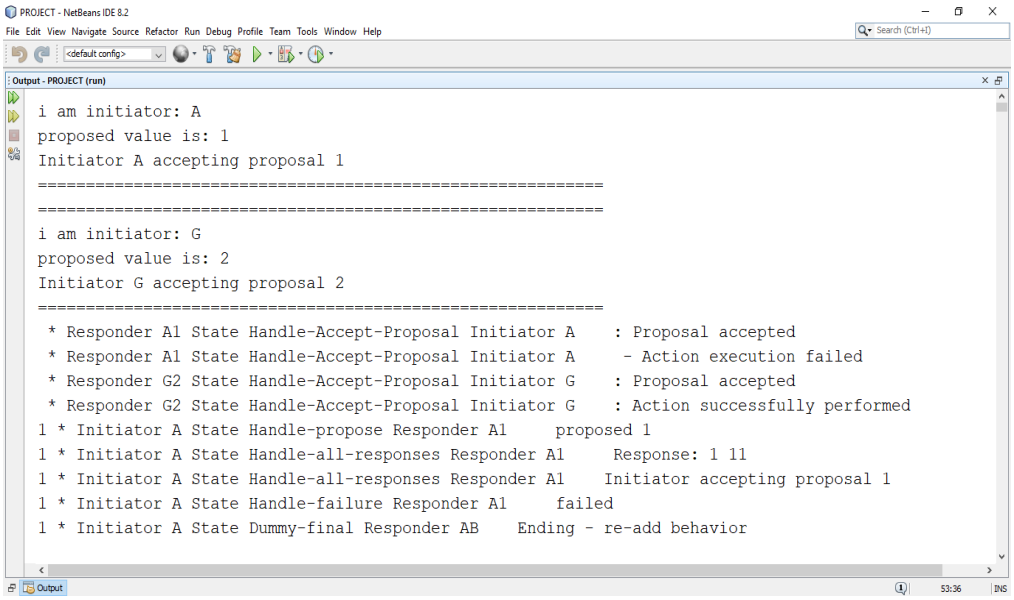
Σχήμα 8.3 Διαμόρφωση 1ης Επικοινωνίας

- *Διαμόρφωση 2ης Επικοινωνίας*
  Ο G αποδέχεται την proposal 1 (Μηχανή)
  Ο A αποδέχεται την proposal 2 (Ρομπότ)



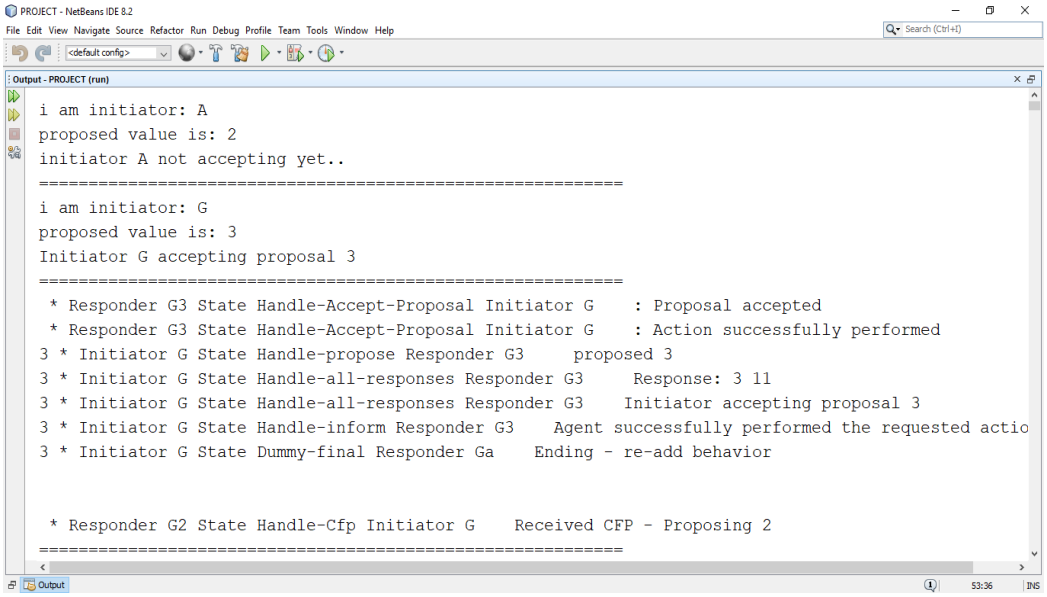
Σχήμα 8.4 Διαμόρφωση 2ης Επικοινωνίας

- *Διαμόρφωση 3ης Επικοινωνίας*
  Ο Α αποδέχεται την proposal 1
  Ο G αποδέχεται την proposal 2



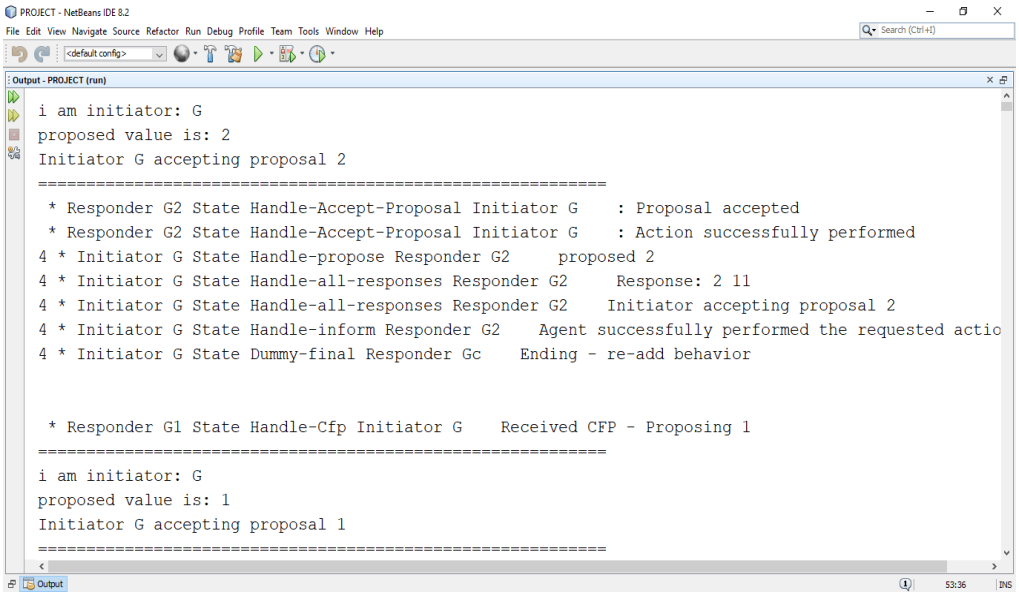
Σχήμα 8.5 Διαμόρφωση 3ης Επικοινωνίας

- *Διαμόρφωση 4ης Επικοινωνίας*
  Ο G αποδέχεται την proposal 3
  Ο Α αναμένει την proposal 2



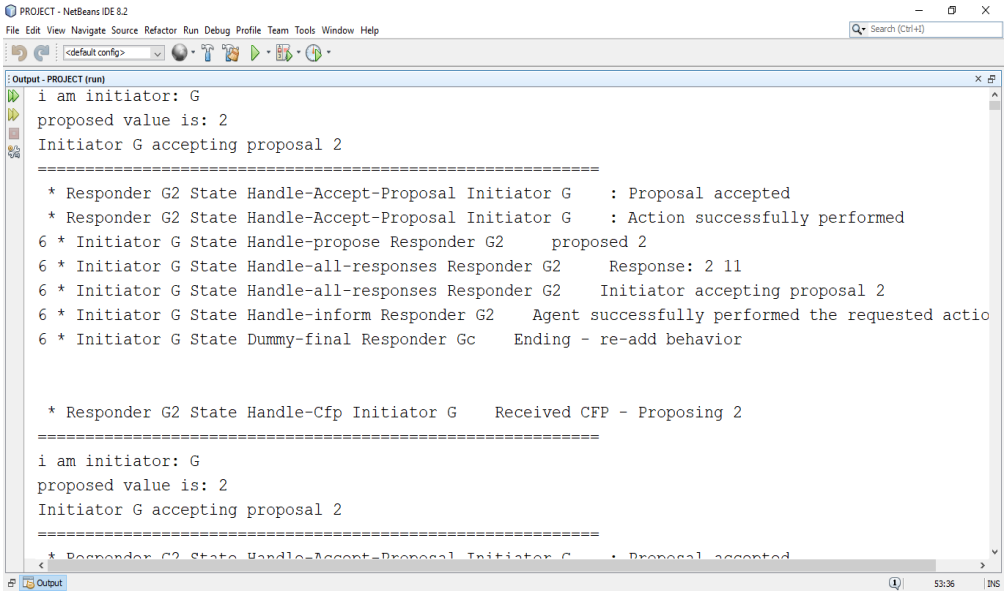
Σχήμα 8.6 Διαμόρφωση 4ης Επικοινωνίας

- *Διαμόρφωση 5ης και 6ης Επικοινωνίας*
  Ο G αποδέχεται την proposal 2
  Ο G αποδέχεται την proposal 1



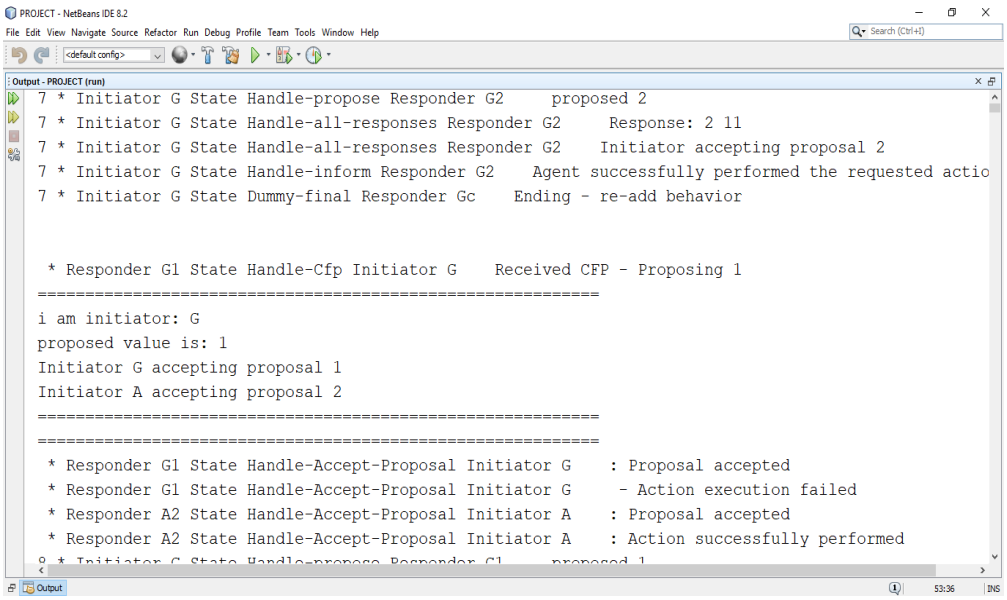
Σχήμα 8.7 Διαμόρφωση 5ης και 6ης Επικοινωνίας

- *Διαμόρφωση 7ης και 8ης Επικοινωνίας*
  Ο G αποδέχεται την proposal 2 και ολοκληρώνεται ένας κύκλος επεξεργασίας τεμαχίου Δ.
  Ο G αποδέχεται την proposal 2 και ξεκινάει με την είσοδο στο σύστημα ένας καινούργιος κύκλος επεξεργασίας τεμαχίου Δ



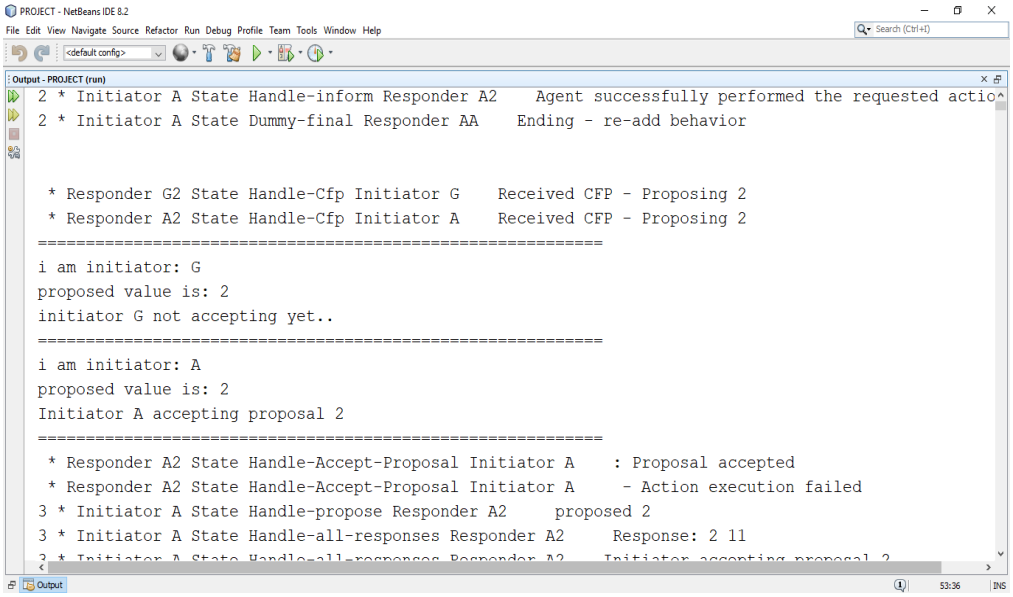
Σχήμα 8.8 Διαμόρφωση 7ης και 8ης Επικοινωνίας

- *Διαμόρφωση 9ης και 10ης Επικοινωνίας*
  Ο G αποδέχεται την proposal 1
  Ο A αποδέχεται την proposal 2 και ολοκληρώνεται ένας κύκλος επεξεργασίας τεμαχίου A



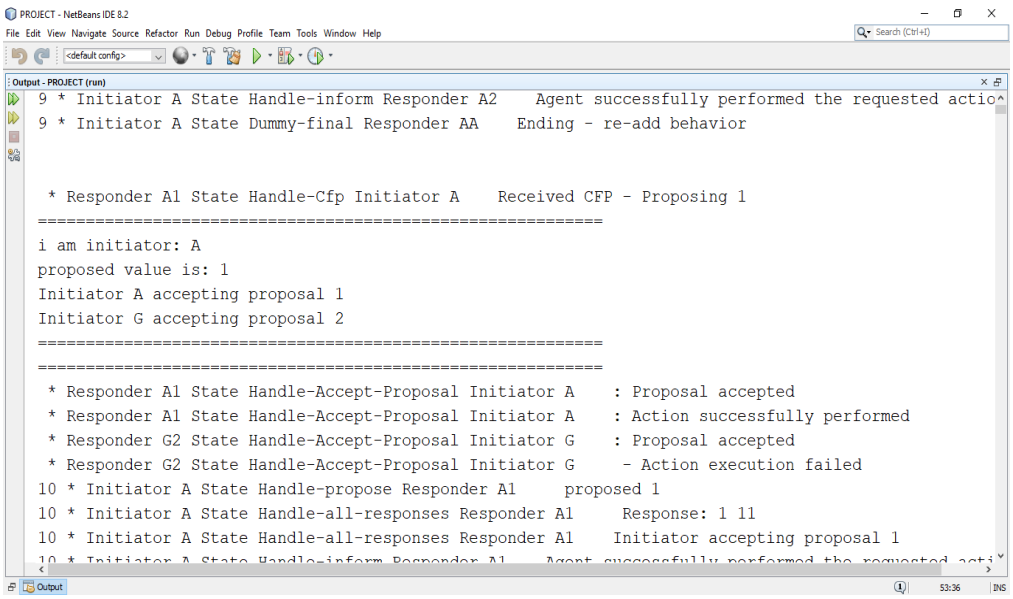
Σχήμα 8.9 Διαμόρφωση 9ης και 10ης Επικοινωνίας

- *Διαμόρφωση 11ης Επικοινωνίας*
  Ο A αποδέχεται την proposal 2 και ξεκινάει ένας καινούργιος κύκλος επεξεργασίας τεμαχίου A
  Ο G αναμένει την proposal 2



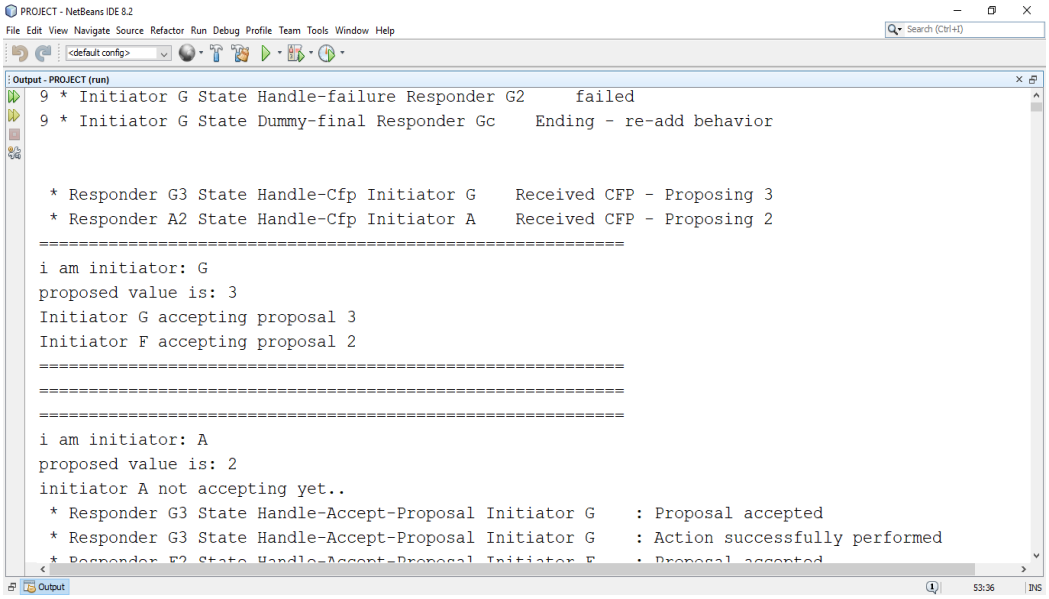
Σχήμα 8.10 Διαμόρφωση 11ης Επικοινωνίας

- *Διαμόρφωση 12ης και 13ης Επικοινωνίας*
  Ο Α αποδέχεται την proposal 1
  Ο G αποδέχεται την proposal 2



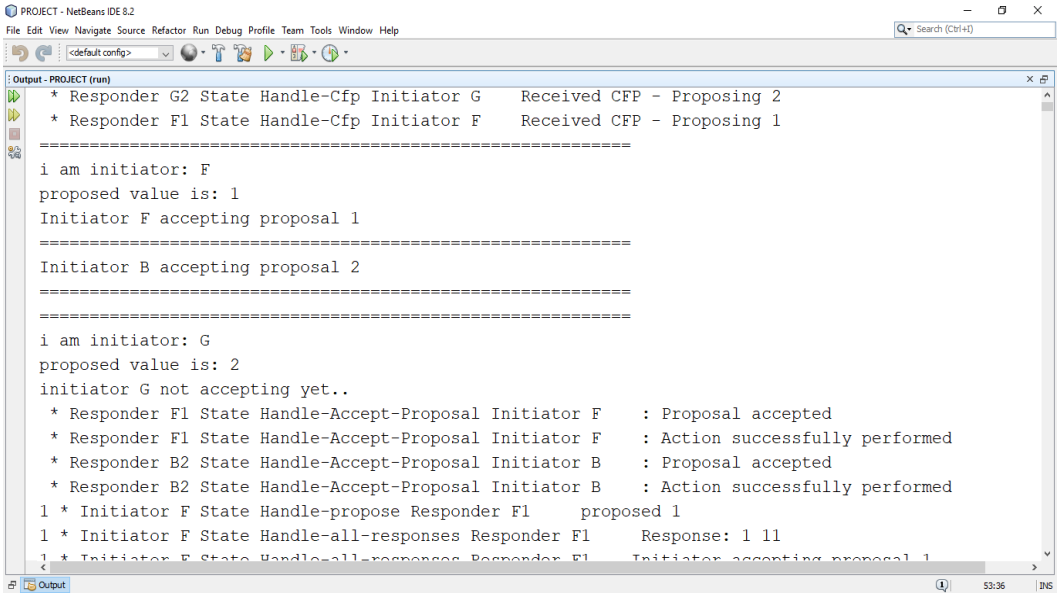
Σχήμα 8.11 Διαμόρφωση 12ης και 13ης Επικοινωνίας

- *Διαμόρφωση 14ης και 15ης Επικοινωνίας*
  Ο G αποδέχεται την proposal 3 (αποθήκη)
  Ο F αποδέχεται την proposal 2 και ξεκινάει ο κύκλος επεξεργασίας του τεμαχίου Β



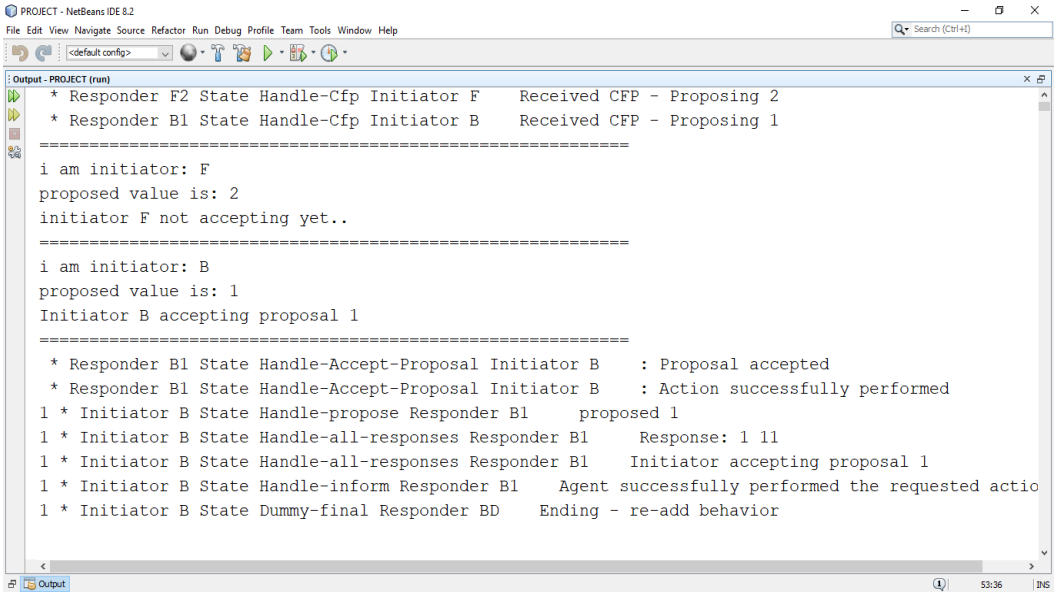
Σχήμα 8.12 Διαμόρφωση 14ης και 15ης Επικοινωνίας

- *Διαμόρφωση 16ης και 17ης Επικοινωνίας*
  Ο F αποδέχεται την proposal 1
  Ο B αποδέχεται την proposal 2 και ξεκινάει ο κύκλος επεξεργασίας του τεμαχίου Γ



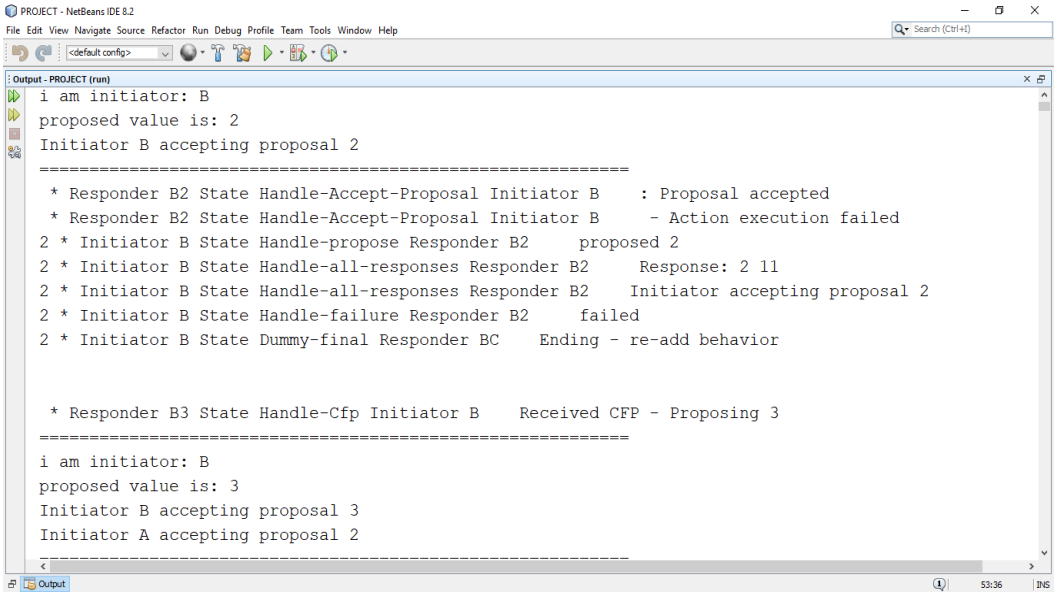
Σχήμα 8.13 Διαμόρφωση 16ης και 17ης Επικοινωνίας

- *Διαμόρφωση 18ης Επικοινωνίας*
  Ο F αναμένει την proposal 2
  Ο B αποδέχεται την proposal 1



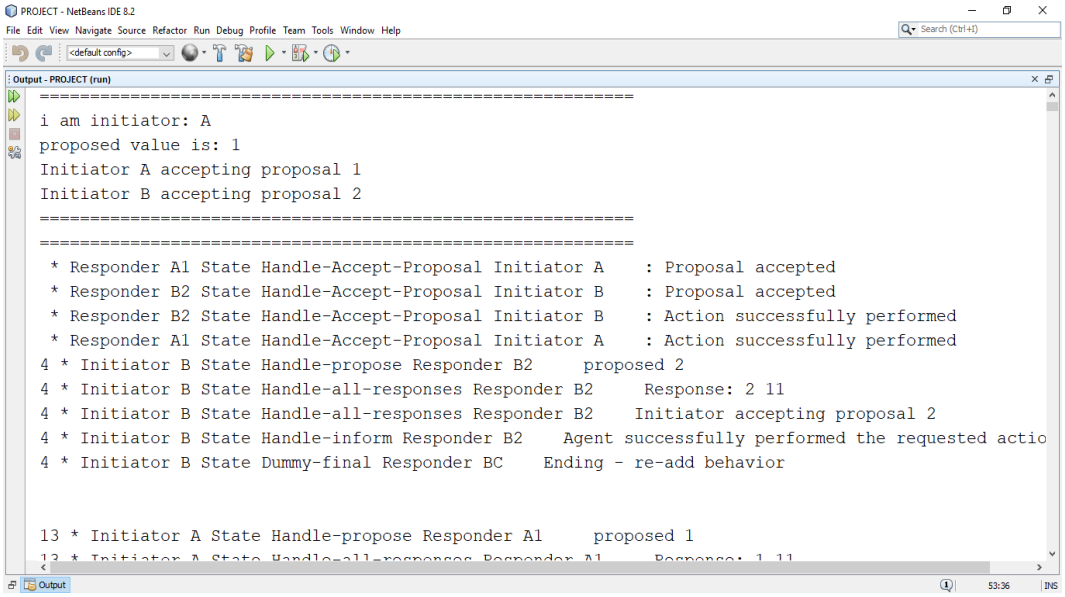
Σχήμα 8.14 Διαμόρφωση 18ης Επικοινωνίας

- *Διαμόρφωση 19ης , 20ης και 21ης Επικοινωνίας*
  Ο Β αποδέχεται την proposal 2 και μετά την proposal 3(αποθήκη)
  Ο Α αποδέχεται την proposal 2



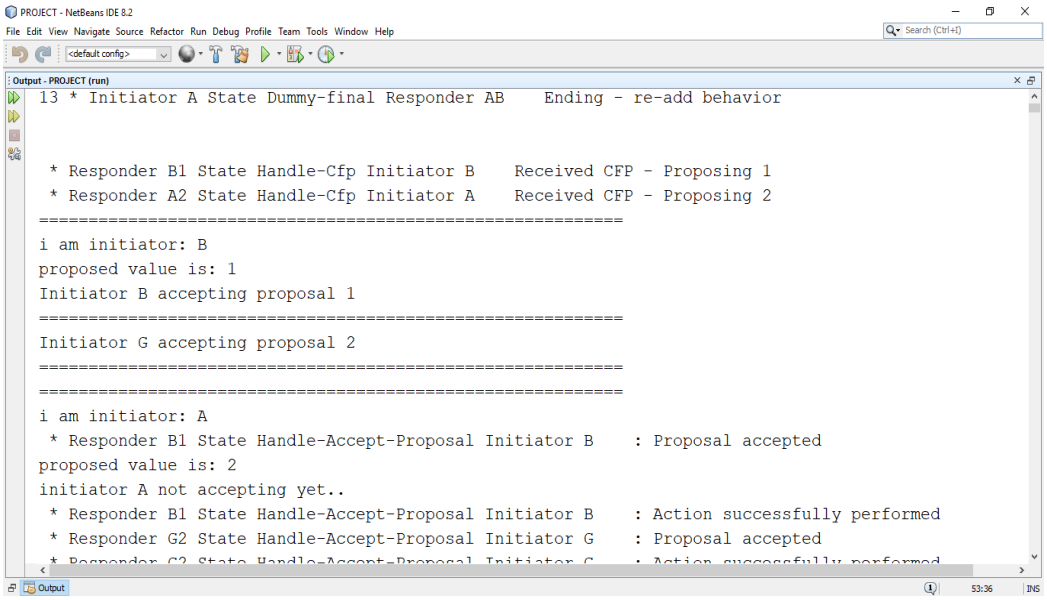
Σχήμα 8.15 Διαμόρφωση 19ης , 20ης και 21ης Επικοινωνίας

- *Διαμόρφωση 22ης και 23ης Επικοινωνίας*
  Ο Β αποδέχεται την proposal 2
  Ο Α αποδέχεται την proposal 1



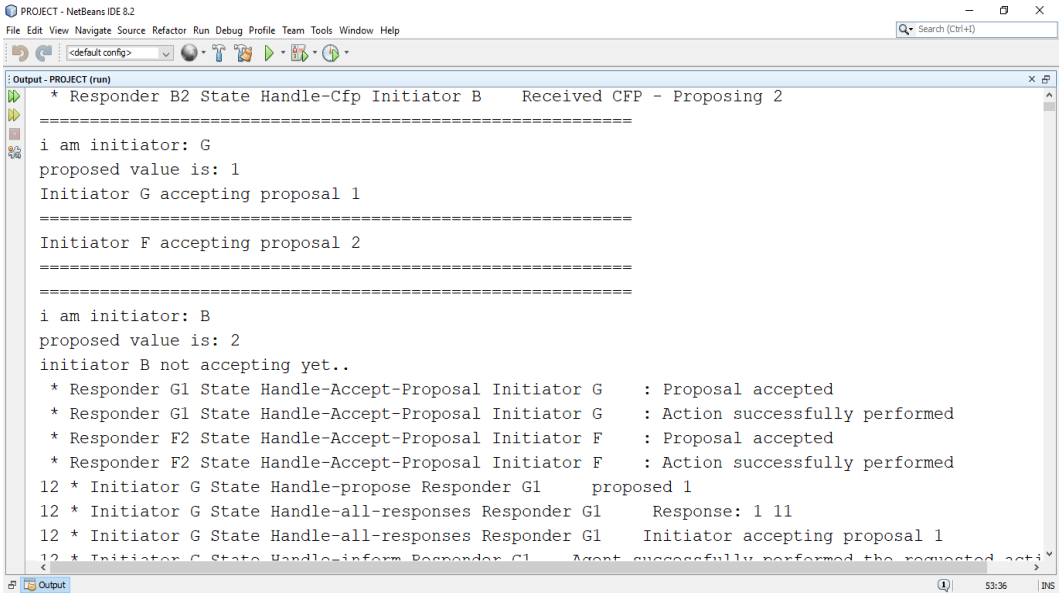
Σχήμα 8.16 Διαμόρφωση 22ης και 23ης Επικοινωνίας

- *Διαμόρφωση 24ης Επικοινωνίας*
  Ο B αποδέχεται την proposal 1
  Ο G αποδέχεται την proposal 2



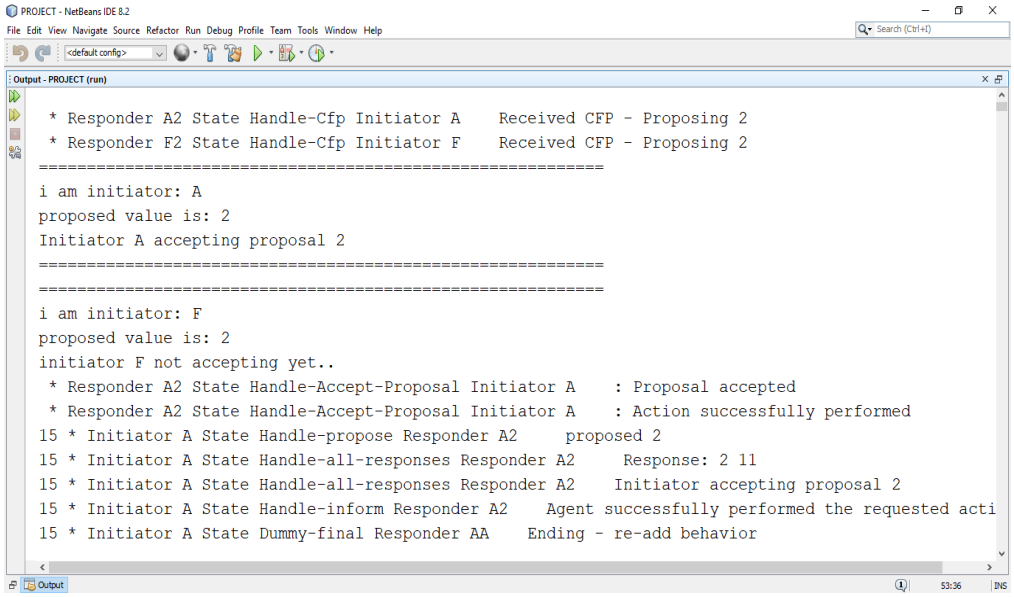
Σχήμα 8.17 Διαμόρφωση 24ης Επικοινωνίας

- *Διαμόρφωση 25ης και 26ης Επικοινωνίας*
  Ο F αποδέχεται την proposal 2 και κλείνει ο πρώτος κύκλος επεξεργασίας του τεμαχίου B
  Ο G αποδέχεται την proposal 1



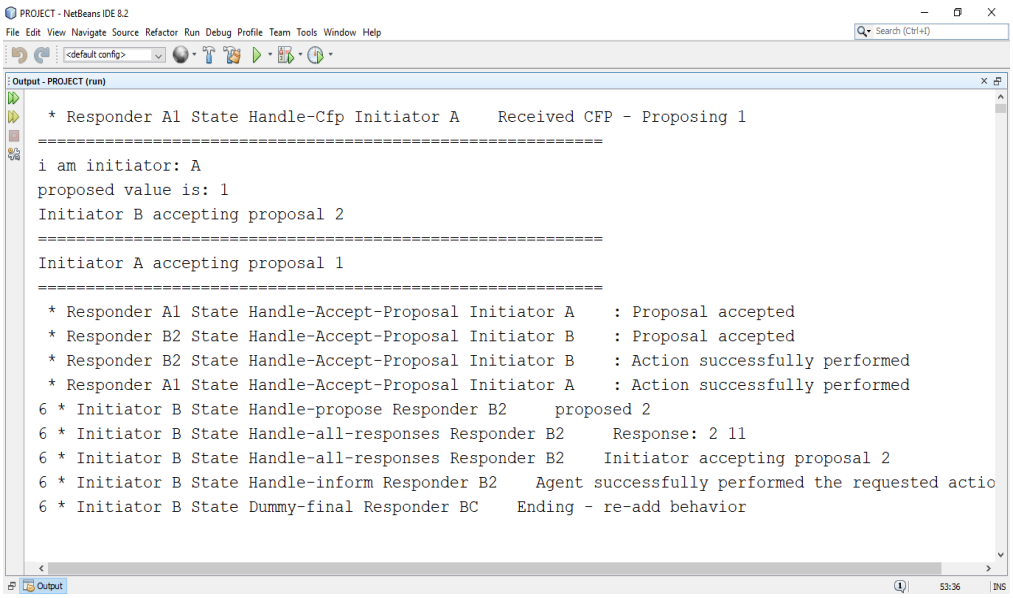
Σχήμα 8.18 Διαμόρφωση 25ης και 26ης Επικοινωνίας

- *Διαμόρφωση 27ης Επικοινωνίας*
  Ο Α αποδέχεται την proposal 2 και κλείνει άλλος ένας κύκλος επεξεργασίας του τεμαχίου Α



Σχήμα 8.19 Διαμόρφωση 27ης Επικοινωνίας

- *Διαμόρφωση 28ης Επικοινωνίας*
  Ο Β αποδέχεται την proposal 2 και ολοκληρώνεται ο 1τος κύκλος επεξεργασίας του τεμαχίου Γ



Σχήμα 8.20 Διαμόρφωση 28ης Επικοινωνίας

Στο στάδιο αυτό και τα 4 τεμάχια προς επεξεργασία έχουν ολοκληρώσει από τουλάχιστον ένα κύκλο επεξεργασίας.

Οι επικοινωνίες μεταξύ των πρακτόρων του συστήματος εκτελούνται επαρκώς καλύπτοντας τις απαιτήσεις της σωστής αλληλουχίας ολοκλήρωσης των επιμέρους επεξεργασιών των τεμαχίων του ευέλικτου συστήματος κατεργασιών.

ΚΕΦΑΛΑΙΟ 9: Συμπεράσματα – Βελτιώσεις

Στην παρούσα εργασία επιδιώξαμε την ανάπτυξη ελεγκτή βασισμένου σε πράκτορες για το Ευέλικτο Σύστημα Κατεργασιών του Εργαστηρίου Τεχνολογίας των Κατεργασιών του ΕΜΠ. Με εφαρμογή συγκεκριμένης μεθοδολογίας από τη διεθνή βιβλιογραφία έγινε η αναγνώριση των πρακτόρων που ενεπλάκησαν στην διαμόρφωση του τελικού ελεγκτή του συστήματος. Η επιλογή του πρωτόκολλου επικοινωνίας Contract Net για την αποτελεσματική επικοινωνία των πρακτόρων του συστήματος μας οδήγησε στην επιτυχή ανάπτυξη του τελικού ελεγκτή. Σαν αποτέλεσμα είχαμε την διαμόρφωση ελέγχου υψηλού επιπέδου (high level control) για τη σωστή λειτουργία του συστήματος κατεργασιών με παράλληλη αποφυγή ανεπιθύμητων καταστάσεων όπως τα αδιέξοδα (deadlock). Η διαμόρφωση ελεγκτή με βάση τη Θεωρία Συστημάτων Πολλαπλών Πρακτόρων είναι μία πολλά υποσχόμενη προσέγγιση που μπορεί να καλύψει με επιτυχία ένα μεγάλο εύρος συστημάτων κατανεμημένης φύσεως, όπως τα συστήματα παραγωγής. Η επικοινωνία και αλληλεπίδραση των πρακτόρων έγινε σε περιβάλλον Java με χρήση του πλαισίου ανάπτυξης JADE. Οι πράκτορες απεικονίσθηκαν με Δίκτυα Petri και το περιεχόμενο των μηνυμάτων τους ήταν σε μορφή XML.

Επιπλέον, για να ενισχυθεί η αποτροπή ανεπιθύμητων φαινομένων κατά τη λειτουργία του συστήματος αναπτύχθηκε Αλγεβρική (Πολυώνυμη Kernel) Πολιτική Αποφυγής Deadlock – PK-DAP η οποία ενσωματώθηκε με κατάλληλο τρόπο στο διαμορφωμένο σύστημα πολλαπλών πρακτόρων. Η ενσωμάτωση Αλγεβρικής DAP σε σύστημα πολλαπλών πρακτόρων είναι κάτι το οποίο δεν έχει μελετηθεί διεξοδικά στη σχετική βιβλιογραφία.
Η ανάπτυξη της εν λόγω PK-DAP μας βοήθησε επιπροσθέτως στην ανάλυση σκοπιμότητας του τελικού ελεγκτή που αναπτύχθηκε στο πλαίσιο JADE.

Το σύστημα της παρούσας εργασίας αναπτύχθηκε με δεδομένες 4 ροές εργασίας για την αντίστοιχη επεξεργασία 4 διαφορετικών τεμαχίων. Η επέκταση της εφαρμογής με επιπλέον ροές εργασίας διαφορετικών διαδρομών και πολυπλοκότητας, αλλά και επιπλέον πόρους, μπορεί να υλοποιηθεί επαρκώς με βάση την προτεινόμενη μεθοδολογία και σχεδιασμό ελέγχου του συστήματος όπως παρουσιάσθηκε στα προηγούμενα κεφάλαια. Η παράλληλη ανάπτυξη αλγεβρικής πολιτικής αποφυγής αδιεξόδου με βάση τα συστήματα κατανομής πόρων μπορεί εν συνεχεία να ενσωματωθεί σε πιο σύνθετες εφαρμογές συστημάτων παραγωγής.

Τέλος, αναδύονται ερευνητικές προκλήσεις σχετικά με την ενσωμάτωση Αλγεβρικών Πολιτικών Αποφυγής Αδιεξόδων σε άλλα πρωτόκολλα επικοινωνίας πρακτόρων, όπως το BDI (Belief – Desire – Intentions), παράλληλα με την επέκταση εφαρμογής ευφυούς ελέγχου πραγματικού χρόνου σε συστήματα συγχρονισμένων δικτύων μεταφοράς αλλά και εν γένει σύγχρονων συστημάτων φραγμένων πόρων (Resource – bounded Systems).

Βιβλιογραφία

[1] Durfee, E.H., Lesser, V.R., Corkill, D.D., (1989). Trends in cooperative distributed problem solving. IEEE Transactions on Knowledge and Data Engineering 1 (1), 63–83.

[2] Bellifemine, F. L. (2007), Caire, G., Greenwood, D.-Developing Multi-Agent Systems with JADE - Wiley Series in Agent Technology,Wiley, England

[3]Bussmann, S., Jennings, N.R., Wooldridge, M., (2004). Multiagent Systems for Manufacturing Control: A Design Methodology'' Series on Agent Technology. Springer, Berlin.

[4] Hsieh, F.S., (2004). Model and control holonic manufacturing systems based on fusion of contract nets and Petri nets. Automatica 40, 51–57.

[5] Hsieh, F.S., (2009). Developing co-operation mechanism for multi-agent systems with Petri Nets. Engineering Applications of Artificial Intelligence, Volume 22 issue 4-5, 2009

[6] Hsieh, F. S. (2008). Holarchy formation and optimization in holonic manufacturing systems with contract net. Automatica, Volume 44 Issue 4, Pages 959-970

[7]Wolsey, L. A. (1998). Integer Programming. Wiley Series in Discrete Mathematics and Optimisation,Wiley, London

[8]Dinghe, N. (2009). PIPE2: A Tool for the Performance Evaluation of Generalised Stochastic Petri Nets, Department of Computing, Imperial College London

[9]Park, J. and Reveliotis, S. A. (2000). Algebraic synthesis of efficient deadlock avoidance policies for sequential resource allocation systems. IEEE Trans. on R&A, 16:190–195

[10]Reveliotis, S. A. and Choi, J. Y. (2006). Designing reversibility-enforcing supervisors of polynomial complexity for bounded Petri nets through the theory of regions. In Proceedings of ATPN 2006, pages 322–341

[11] Reveliotis, S. A., Roszkowska, E., and Choi, J. Y. (2007). Generalized algebraic deadlock avoidance policies for sequential resource allocation systems. IEEE Transactions on Automatic Control, 52:2345–2350

[12] Reveliotis, S. A. (2017). Logical Control of Complex Resource Allocation Systems. Foundations and Trends® in Systems and Control Vol. 4, No. 1-2 (2017) 1–223 ,NOW Publications, USA

[13]Reveliotis, S. A. (2005) Real-Time Management of Resource Allocation Systems. International Series in Operations Research&Management Science, Springer , USA

[14] Li, Z., Zhou, MC (2009). Deadlock Resolution in Automated Manufacturing Systems - A Novel Petri Net Approach. Springer Series: Advances in Industrial Control,Springer-Verlag, London

[15] Δημητριάδου - Γιαννέλη Χριστίνα (2011). Δίκτυα Petri. Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, Τμήμα Πληροφορικής και Οικονομικών (http://ikee.lib.auth.gr/record/132267/files/GRI-2013-10806.pdf)

[16] Παπάζογλου Κατερίνα (2016), Ανάπτυξη προγράμματος ελέγχου plc για ευέλικτο κύτταρο κατεργασιών με βάση χρωματιστά δίκτυα Petri, Διπλωματική Εργασία ,ΕΜΠ

[17] Γιαννάτσης Ι., Δεδούσης Β., Κανελίδης Β. (2015) Σύγχρονες Τεχνολογίες Κατασκευής με τη βοήθεια Η/Υ, Κεφάλαιο 4, ΣΥΝΔΕΣΜΟΣ ΕΛΛΗΝΙΚΩΝ ΑΚΑΔΗΜΑΪΚΩΝ ΒΙΒΛΙΟΘΗΚΩΝ, Εκδόσεις Κάλλιπος, Αθήνα

[18] Βλαχάβας Ι., Κεφαλάς Π., Βασιλειάδης Ν. (2011), Πολυπρακτορικά Συστήματα (Διαφάνειες), Τεχνητή Νοημοσύνη Β' Έκδοση, Εκδόσεις Πανεπιστημίου Μακεδονίας, Θεσσαλονίκη

[19] Ευέλικτα Συστήματα Παραγωγής (2016) , (Διαφάνειες) ΤΕΙ Δυτικής Μακεδονίας (https://eclass.teiwm.gr/modules/document/file.php/TUCMECH116/2-FMS.pdf)

[21] www.lindo.com (LINDO Systems Optimisation Software)

[22] www.tilab.com (JADE Agent Development Framework)

[23] pipe2.sourceforge.net/ (PIPE2 : Platform Independent Petri net Editor 2)

[24] www.laas.fr/tina (TINA - TIme petri Net Analyzer)

[25] http://www.iro.umontreal.ca/~dift6802/jade/src/examples/protocols/ContractNetResponderAgent.java

[26] http://www.iro.umontreal.ca/~dift6802/jade/src/examples/protocols/ContractNetInitiatorAgent.java

Παράρτημα

A.

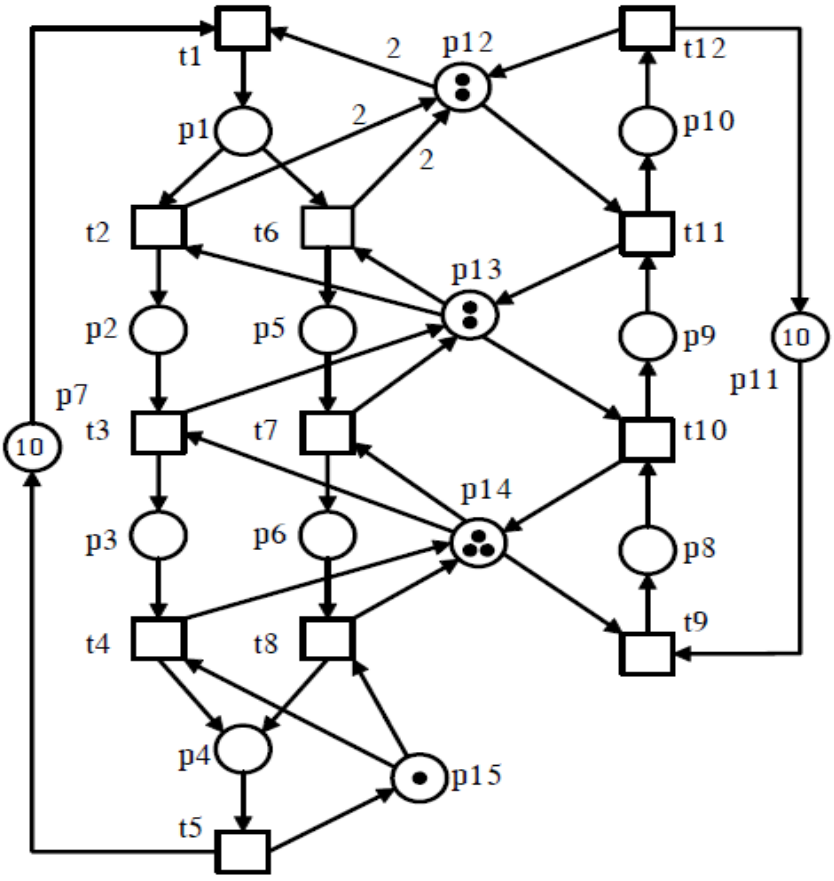
Έστω r ότι είναι θέση πόρου και S ένα απόλυτα ελάχιστο σιφώνιο σε ένα δίκτυο τύπου S3PR. Η διαφορά δύο πολυ-συνόλων I_r και r μας δίνει τον 'κάτοχο' του πόρου r :

$H(r) = I_r - r$

όπου I_r είναι η ελάχιστη P – ημιρροή του δικτύου.

Σαν πολυ-σύνολο η εξίσωση $Th(S) = \Sigma_{r \in S_R} H(r) - \Sigma_{r \in S_R, p \in S_A} I_r(p).p$ ονομάζεται συμπληρωματικό σύνολο του S. Επιπλέον, $||Th(S)||=\{p|p \in \cup_{r \in S_R} H(r), p /\in S\}$ ονομάζεται υποστήριξη του συμπληρωματικού συνόλου Th(S) του σιφωνίου S.

Στο ακόλουθο παράδειγμα δίνεται η λεπτομερής δημιουργία συμπληρωματικών σιφωνίων.



Τα ελάχιστα απόλυτα σιφώνια του δικτύου είναι:

S1 = {p3, p6, p9, p13, p14},

S2 = {p2, p5, p10, p12, p13},

S3 = {p3, p6, p10, p12, p13, p14}.

Έχουμε :

Ip_{12} = 2p1+ p10+ p12,

Ip_{13} = p2+ p5+ p9+ p13,

Ip_{14} = p3+ p6+ p8+ p14,

Ip_{15} = p4+ p15,

H(p12)=2p1+ p10,
H(p13)= p2+ p5+ p9,
H(p14)= p3+ p6+ p8.

Τα συμπληρωματικά σιφώνια και οι υποστηρίξεις των σιφωνίων S1-S3 είναι :
Th(S1)=(H(p13)+H(p14))−(p3+ p6+ p9)= p2+ p5+ p8,
Th(S2)=(H(p12)+H(p13))−(p2+ p5+ p10)= 2p1+ p9,
Th(S3)= $\sum$[H(p12) + H(p13)+ H(p14)] −(p3+ p6+ p10) = 2p1+ p2+ p5+ p8+ p9,

και
||Th(S1)|| = {p2, p5, p8},
||Th(S2)|| = {p1, p9},
||Th(S3)|| = {p1, p2, p5, p8, p9}

B.

ContractNetInitiator.java

```
/**
 * *****************************************************************
 * JADE - Java Agent DEvelopment Framework is a framework to develop
 * multi-agent systems in compliance with the FIPA specifications.
 * Copyright (C) 2000 CSELT S.p.A.
 *
 * GNU Lesser General Public License
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation,
 * version 2.1 of the License.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA  02111-1307, USA.
 * *****************************************************************
 */
package examples.protocols;

import jade.core.Agent;
import jade.core.AID;
import jade.lang.acl.ACLMessage;
import jade.proto.ContractNetInitiator;
import jade.domain.FIPANames;

import java.util.Date;
import java.util.Vector;
import java.util.Enumeration;

/**
  This example shows how to implement the initiator role in
  a FIPA-contract-net interaction protocol. In this case in particular
  we use a <code>ContractNetInitiator</code>
  to assign a dummy task to the agent that provides the best offer
  among a set of agents (whose local
  names must be specified as arguments).
  @author Giovanni Caire - TILAB
 */
```

```
public class ContractNetInitiatorAgent extends Agent {
		private int nResponders;

 protected void setup() {
		// Read names of responders as arguments
		Object[] args = getArguments();
		if (args != null && args.length > 0) {
				nResponders = args.length;
		System.out.println("Trying to delegate dummy-action to one out of "+nResponders+" responders.");

				// Fill the CFP message
				ACLMessage msg = new ACLMessage(ACLMessage.CFP);
				for (int i = 0; i < args.length; ++i) {
						msg.addReceiver(new AID((String) args[i], AID.ISLOCALNAME));
				}
						msg.setProtocol(FIPANames.InteractionProtocol.FIPA_CONTRACT_NET);
		// We want to receive a reply in 10 secs
		msg.setReplyByDate(new Date(System.currentTimeMillis() + 10000));
		msg.setContent("dummy-action");

addBehaviour(new ContractNetInitiator(this, msg) {



protected void handlePropose(ACLMessage propose, Vector v) {

System.out.println("Agent "+propose.getSender().getName()+" proposed "+propose.getContent());

     }

protected void handleRefuse(ACLMessage refuse) {

System.out.println("Agent "+refuse.getSender().getName()+" refused");

    }

protected void handleFailure(ACLMessage failure) {
				if (failure.getSender().equals(myAgent.getAMS())) {
// FAILURE notification from the JADE runtime: the receiver
// does not exist
      System.out.println("Responder does not exist");
				}
				else {
				System.out.println("Agent "+failure.getSender().getName()+" failed");
				}
// Immediate failure --> we will not receive a response from this agent
						nResponders--;
}

protected void handleAllResponses(Vector responses, Vector acceptances) {

			if (responses.size() < nResponders) {
// Some responder didn't reply within the specified timeout
		System.out.println("Timeout expired: missing "+(nResponders - responses.size())+" responses");
			}
// Evaluate proposals.
								int bestProposal = -1;
								AID bestProposer = null;
								ACLMessage accept = null;
								Enumeration e = responses.elements();
						while (e.hasMoreElements()) {
						ACLMessage msg = (ACLMessage) e.nextElement();
						if (msg.getPerformative() == ACLMessage.PROPOSE) {
						ACLMessage reply = msg.createReply();
						reply.setPerformative(ACLMessage.REJECT_PROPOSAL);
						acceptances.addElement(reply);
						int proposal = Integer.parseInt(msg.getContent());
									if (proposal > bestProposal) {
											bestProposal = proposal;
											bestProposer = msg.getSender();
											accept = reply;
```

```
                                                                                }
                                                                        }
                                                                }
                                                        // Accept the proposal of the best proposer
                                                                if (accept != null) {
System.out.println("Accepting proposal "+bestProposal+" from responder "+bestProposer.getName());

                accept.setPerformative(ACLMessage.ACCEPT_PROPOSAL);
                                                                }
                                                        }

protected void handleInform(ACLMessage inform) {

System.out.println("Agent "+inform.getSender().getName()+" successfully performed the requested action");

}
                                                } );
                }
                else {
                                System.out.println("No responder specified.");
                }
 }
}
```

ContractNetResponder.java

```
 * ***************************************************************
 * JADE - Java Agent DEvelopment Framework is a framework to develop
 * multi-agent systems in compliance with the FIPA specifications.
 * Copyright (C) 2000 CSELT S.p.A.
 *
 * GNU Lesser General Public License
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation,
 * version 2.1 of the License.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA  02111-1307, USA.
 * ***************************************************************
 */
package examples.protocols;

import jade.core.Agent;
import jade.core.AID;
import jade.lang.acl.ACLMessage;
import jade.lang.acl.MessageTemplate;
import jade.proto.ContractNetResponder;
import jade.domain.FIPANames;
import jade.domain.FIPAAgentManagement.NotUnderstoodException;
import jade.domain.FIPAAgentManagement.RefuseException;
import jade.domain.FIPAAgentManagement.FailureException;

/**
  This example shows how to implement the responder role in
  a FIPA-contract-net interaction protocol. In this case in particular
  we use a <code>ContractNetResponder</code>
  to participate into a negotiation where an initiator needs to assign
  a task to an agent among a set of candidates.
  @author Giovanni Caire - TILAB
 */
public class ContractNetResponderAgent extends Agent {
```

```
 protected void setup() {
		System.out.println("Agent "+getLocalName()+" waiting for CFP...");
		MessageTemplate template = MessageTemplate.and(
				MessageTemplate.MatchProtocol(FIPANames.InteractionProtocol.FIPA_CONTRACT_NET),
				MessageTemplate.MatchPerformative(ACLMessage.CFP) );

				addBehaviour(new ContractNetResponder(this, template) {

protected ACLMessage prepareResponse(ACLMessage cfp) throws NotUnderstoodException, RefuseException {

System.out.println("Agent "+getLocalName()+": CFP received from "+cfp.getSender().getName()+". Action is
"+cfp.getContent());
                                                int proposal = evaluateAction();
                                                if (proposal > 2) {
                                                        // We provide a proposal
                                System.out.println("Agent "+getLocalName()+": Proposing "+proposal);
                                                ACLMessage propose = cfp.createReply();
                                                propose.setPerformative(ACLMessage.PROPOSE);
                                                propose.setContent(String.valueOf(proposal));
                                                return propose;
                                                }
                                                else {
                                                        // We refuse to provide a proposal
                                                System.out.println("Agent "+getLocalName()+": Refuse");
                                                throw new RefuseException("evaluation-failed");
                                                }
                                        }

protected ACLMessage prepareResultNotification(ACLMessage cfp, ACLMessage propose,ACLMessage accept) throws
FailureException {
                                System.out.println("Agent "+getLocalName()+": Proposal accepted");

                                if (performAction()) {

                               System.out.println("Agent "+getLocalName()+": Action successfully performed");

                        ACLMessage inform = accept.createReply();
                            inform.setPerformative(ACLMessage.INFORM);
                            return inform;
                                                }
                                                else {


                                System.out.println("Agent "+getLocalName()+": Action execution failed");

                                            throw new FailureException("unexpected-error");
                                                }
                                        }

protected void handleRejectProposal(ACLMessage reject) {

                 System.out.println("Agent "+getLocalName()+": Proposal rejected");
                                        }
                                } );
 }

 private int evaluateAction() {
		// Simulate an evaluation by generating a random number
		return (int) (Math.random() * 10);
 }

 private boolean performAction() {
		// Simulate action execution by generating a random number
		return (Math.random() > 0.2);
 }
}
```