ΠΡΟΛΟΓΟΣ

Η παρούσα διπλωματική εργασία πραγματεύεται την κατασκευή ενός έμπειρου συστήματος διάγνωσης και επιδιόρθωσης βλαβών σε αεροσκάφη. Στα πλαίσια της εργασίας, αρχικά καταγράφηκαν οι φάσεις των διαδικασιών διάγνωσης και επιδιόρθωσης βλαβών σε συγκεκριμένο τύπο αεροσκάφους που ακολουθούν οι τεχνικοί και μηχανικοί αεροπορικής εταιρείας, ακολούθησε η αξιολόγηση των στοιχείων που συγκεντρώθηκαν και στη συνέχεια πραγματοποιήθηκε η ανάπτυξη του έμπειρου συστήματος.

Η καταγραφή των διαδικασιών επετεύχθη μετά από πολυάριθμες επισκέψεις σε τεχνική βάση ελληνικής αεροπορικής εταιρείας καθώς και της βοήθειας των μηχανικών και των τεχνικών της που εξειδικεύονται στον τομέα της διάγνωσης και της επιδιόρθωσης βλαβών.

Αποτέλεσμα της προσπάθειας αυτής είναι να κατασκευαστεί ένα έμπειρο σύστημα, στο σύστημα κανόνων CLIPS, με την ονομασία A300-600 Troubleshooter Expert System. Το υλοποιημένο σύστημα αποσκοπεί να αποτελέσει ένα βοηθητικό εργαλείο στην λήψη αποφάσεων επιδιόρθωσης βλαβών αλλά και να εκσυγχρονίσει με νέες τεχνολογίες την φιλοσοφία που ακολουθείται σήμερα από τους μηχανικούς στην συντήρηση των αεροσκαφών.

Κλείνοντας, θα ήθελα να ευχαριστήσω τους Γουγουλάκη Γεώργιο και Ζενερτζή Σταμάτη του τμήματος Μελετών Αεροσκαφών της Ολυμπιακής Αεροπορίας, για την πολύτιμη βοήθεια τους στην διεκπεραίωση του έμπειρου συστήματος καθώς και την βιβλιοθήκη της Ολυμπιακής Αεροπορίας για την προσφορά των απαραίτητων εγχειριδίων. Επίσης, ευχαριστώ ιδιαίτερα τον καθηγητή του τμήματος Μηχανικών Παραγωγής και Διοίκησης Ματσατσίνη Νικόλαο, για την πολύτιμη βοήθεια και καθοδήγηση του στην ολοκλήρωση της παρούσας εργασίας. Τέλος, ευχαριστώ θερμά την οικογένειά μου για την υλική και ηθική συμπαράστασή της κατά τη διάρκεια των σπουδών μου.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ 1
ΠΕΡΙΕΧΟΜΕΝΑ 2
ΚΕΦΑΛΑΙΟ 1 4
Εισαγωγή 4
1.1 Γενικά 4
1.2 Φιλοσοφίες συντήρησης 6
1.3 Καθορισμός του προβλήματος 13
1.4 Στόχοι και δομή της εργασίας 13
ΚΕΦΑΛΑΙΟ 2 15
Έμπειρα Συστήματα 15
2.1 Εισαγωγή 15
2.2 Δομή έμπειρων συστημάτων 16
2.3 Χαρακτηριστικά έμπειρων συστημάτων 24
2.4 Εργαλεία ανάπτυξης 26
2.4.1 Γλώσσες προγραμματισμού ΤΝ 28
2.4.2 Κελύφη έμπειρων συστημάτων 28
2.5 Αναπαράσταση γνώσης 32
2.6 Μέθοδοι αναπαράστασης γνώσης 33
2.6.1 Σημαντικά - σημασιολογικά δίκτυα (semantic networks) 34
2.6.2 Αντικείμενο-ιδιότητα-τιμή 36
2.6.3 Κανόνες 36
2.6.4 Πλαίσια 38
2.7 Ανάπτυξη έμπειρων συστημάτων 39
2.8 Σύγκριση έμπειρων συστημάτων 42
ΚΕΦΑΛΑΙΟ 3 46
Το Σύστημα Κανόνων CLIPS 46
3.1 Εισαγωγή 46
3.2 Δομή του CLIPS 47
3.3 Σύνταξη του CLIPS 48
3.3.1 Βασικά δομικά στοιχεία 48
3.3.2 Μεταβλητές 49
3.3.3 Γεγονότα 49

3.3.4 Κανόνες **51**
3.4 Βασικές Εντολές Περιβάλλοντος **52**
3.5 Συναρτήσεις **53**
3.6 Προτεραιότητα Κανόνων και Στρατηγικές Επίλυσης Συγκρούσεων **58**
3.6.1 Προτεραιότητα Κανόνων **58**
3.6.2 Στρατηγικές Επίλυσης Συγκρούσεων **59**
3.7 Το περιβάλλον CLIPS **60**
ΚΕΦΑΛΑΙΟ 4 ..... **62**
Το Έμπειρο Σύστημα A300-600 TES **62**
4.1 Η Ανάγκη Δημιουργίας του Έμπειρου Συστήματος **62**
4.2 Ανάλυση και σχεδίαση του έμπειρου συστήματος A300-600 TES **63**
4.3 Ανάλυση και σχεδίαση της βάσης γνώσης **66**
4.4 Δομή της βάσης γνώσης **70**
4.5 Μηχανισμός εξαγωγής συμπερασμάτων **76**
4.6 Μνήμη εργασίας **77**
ΚΕΦΑΛΑΙΟ 5 ..... **79**
Παρουσίαση του συστήματος μέσω εφαρμογής **79**
ΚΕΦΑΛΑΙΟ 6 ..... **84**
Συμπεράσματα αξιολόγησης **84**
6.1 Συμπεράσματα **84**
6.2 Μελλοντική εργασία **85**
ΒΙΒΛΙΟΓΡΑΦΙΑ **86**

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή

1.1 Γενικά

Αντικειμενικός σκοπός της λειτουργίας της συντήρησης σε μια αεροπορική εταιρεία είναι η παροχή ασφαλών αεροσκαφών στο ελάχιστο δυνατό χρόνο και στο ελάχιστο δυνατό κόστος. Η ασφάλεια είναι ο πλέον σημαντικός παράγοντας στη συντήρηση αεροσκαφών. Από την άλλη, η αεροπορική εταιρεία, όντας μια επιχείρηση, έχει πάντα σαν στόχο την ελαχιστοποίηση του κόστους λειτουργίας.

Η συντήρηση των αεροσκαφών, στην πολιτική αεροπορία, απορροφά ένα σημαντικό κομμάτι των λειτουργικών εξόδων τους. Οι αεροπορικές δραστηριότητες λαμβάνουν χώρα σε ένα ιδιαίτερο δυναμικό περιβάλλον : η χρήση των αεροσκαφών, οι ποικίλες διαμορφώσεις, οι περιορισμοί στους διαθέσιμους πόρους και οι επιχειρησιακές απαιτήσεις μεταβάλλονται αρκετές φορές στη διάρκεια μιας πτητικής ημέρας. Μέσα σε αυτό το περιβάλλον, οι διαχειριστές της συντήρησης θα πρέπει να αφομοιώσουν ένα σημαντικό όγκο τεχνικού πληροφοριακού υλικού προκειμένου να εξασφαλίσουν την υψηλή ποιότητα των αεροσκαφών εν πτήση.

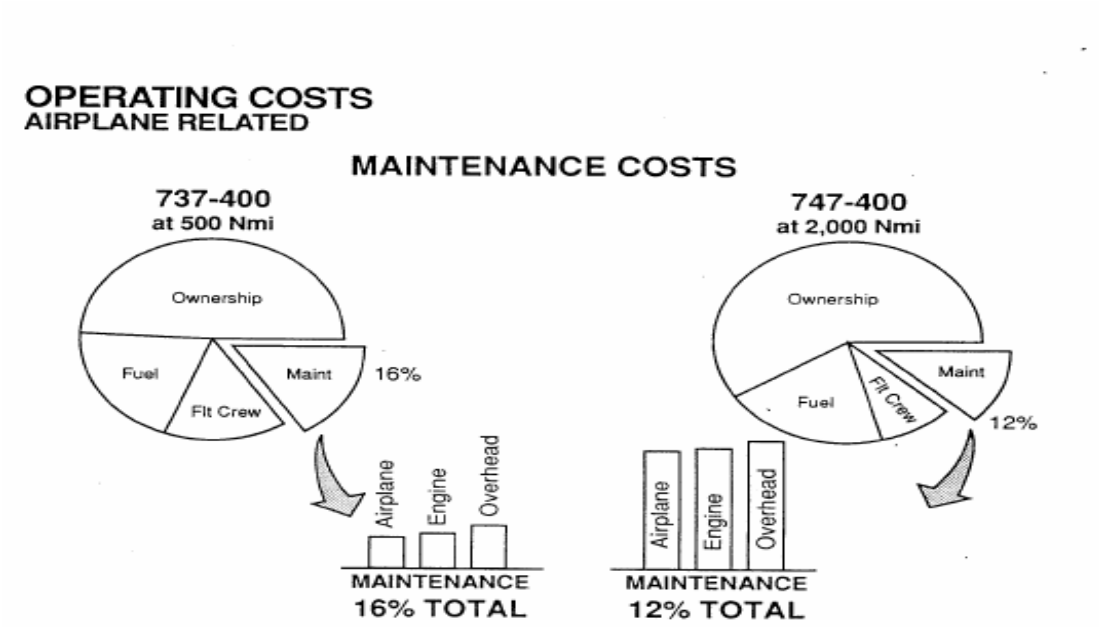
Η συντήρηση αεροσκαφών εκτελείται από τεχνικούς οι οποίοι επιβάλλεται από τους διεθνής κανονισμούς, να συμβουλεύονται εξειδικευμένους μηχανικούς για οδηγίες και έγκριση επισκευής και συντήρησης. Οι μηχανικοί εκτός από την εμπειρία που μπορεί να διαθέτουν, στηρίζονται σε εγχειρίδια συντήρησης (Aircraft Maintenance Manuals), επιδιόρθωσης βλαβών (Troubleshooting Manuals) και εξωτερικές πηγές πληροφοριών, διαδικασίες αρκετά χρονοβόρες και δαπανηρές. Η έγκαιρη ανάκτηση

των πληροφοριών είναι ζωτικής σημασίας για τους μηχανικούς προκειμένου να διαγνώσουν το είδος της βλάβης και να δώσουν τις σωστές διαδικασίες στους τεχνικούς.

Από την στιγμή που ένα αεροσκάφος θα εισέλθει στο υπόστεγο για την προκαθορισμένη ή μη συντήρηση του, εξειδικευμένα συνεργεία θα αναλάβουν να φέρουν εις πέρας το πολύπλοκο και δύσκολο έργο της συντήρησης του αεροσκάφους. Ένα από τα στάδια της συντήρησης είναι η διάγνωση και η επιδιόρθωση βλαβών. Οι βλάβες που παρουσιάζει ένα αεροσκάφος ανιχνεύονται με ποικίλους τρόπους. Πολλές από αυτές εμφανίζονται στις οθόνες του πιλοτηρίου κατά την διάρκεια μιας πτήσης, άλλες ανιχνεύονται κατά την επιθεώρηση πριν και μετά την πτήση από τους τεχνικούς και άλλες κατά την διάρκεια της συντήρησης. Υπάρχουν όμως και βλάβες οι οποίες μπορεί να οφείλονται στις καλωδιώσεις του αεροσκάφους και είναι από τις πιο πολύπλοκες, αφού τις περισσότερες φορές χρειάζεται η καθήλωση του αεροσκάφους στο έδαφος για την αποκατάσταση τους.

Αναμφίβολα η λειτουργία μιας αεροπορικής εταιρείας συμβαδίζει με εκείνη μιας οποιαδήποτε εταιρείας η οποία έχει σαν αντικειμενικό της σκοπό την αύξηση του κέρδους της. Οι παράμετροι που επηρεάζουν το κόστος λειτουργίας μιας αεροπορικής εταιρείας συνοψίζονται στο κόστος του προσωπικού και συντήρησης των εγκαταστάσεων της, το κόστος υγρών καυσίμων, το κόστος απόκτησης του στόλου των αεροσκαφών και ανανέωσης αυτού και το κόστος συντήρησης του.

Το κόστος συντήρησης του στόλου των αεροσκαφών μιας αεροπορικής εταιρείας αντιστοιχεί σε ποσοστό 10 – 15 % του συνολικού κόστους λειτουργίας της. Αυτό υποδιαιρείται ισοδύναμα στα δομικά μέρη του αεροσκάφους, τους κινητήρες και τις γενικές επισκευές εξαρτημάτων. Σημαντική παράμετρος στο κόστος συντήρησης αποτελεί ο τύπος του αεροσκάφους όπως φαίνεται και στο ακόλουθο σχήμα, όπου προκύπτει ότι το κόστος συντήρησης μεταβάλλεται από 16% στο σύνολο του λειτουργικού κόστους της εταιρείας για ένα Boeing 737 σε 12% για ένα Boeing 747.



Σχήμα 1.1. Κόστος Συντήρησης Αεροσκάφους *(Πηγή: U. S. Department of Transportation/Federal Aviation Administration, 2003)*

1.2 Φιλοσοφίες συντήρησης

Κατά τη συντήρηση των αεροσκαφών, οι αεροπορικές εταιρείες αναπτύσσουν προγράμματα λογισμικού στα οποία εμπεριέχονται όλες εκείνες οι εργασίες που τα καθιστούν ασφαλή και ικανά να ανταπεξέλθουν στις συνθήκες πτήσης. Αυτά τα προγράμματα περιλαμβάνουν τα τρία ακόλουθα αντικείμενα:

- Την ανταπόκριση στις απαιτήσεις ασφάλειας των αεροσκαφών και της λειτουργικότητας αυτών.
- Την ελαχιστοποίηση του ρυθμού αστοχίας κατά τη λειτουργία, μέσα από την προληπτική συντήρηση.
- Την ελαχιστοποίηση του κόστους συντήρησης.

Αρχικά, τα προγράμματα συντήρησης αναπτύσσονται από τους κατασκευαστές, μέσα από τη χρήση τεχνικών και αναλυτικών εργαλείων. Στη συνέχεια, αυτό το πρόγραμμα συντήρησης γίνεται αποδεκτό από τις αρμόδιες αρχές (FAA/JAR),οι οποίες αποφαίνονται για την ορθότητα του πριν αυτό αποδεσμευτεί προς τις ενδιαφερόμενες

πλευρές. Τα προγράμματα συντήρησης εμπεριέχουν όλες εκείνες τις ελάχιστες απαιτήσεις, οι οποίες θα πρέπει να ικανοποιηθούν από το χρήστη προκειμένου να υπάρξει αποτελεσματικότητα και ασφάλεια στη συντήρηση.

Με την πάροδο κάποιου χρονικού διαστήματος, η απόκτηση εμπειρίας από τη χρήση του αεροσκάφους δίνει τη δυνατότητα βελτίωσης με αναπροσαρμογές του προγράμματος συντήρησης, το οποίο γίνεται έτσι πιο ευέλικτο και οικονομικό. Έτσι, οι διάφοροι χρήστες του αεροσκάφους ή των εξαρτημάτων αυτού είναι δυνατό να προτείνουν την αναθεώρηση της πολιτικής συντήρησης που ακολουθείται γενικά ή σε κάποια επιμέρους τμήματα, με βάση στοιχεία που προκύπτουν και τα οποία αποστέλλονται για έρευνα και συζήτηση στον κατασκευαστικό φορέα. Με βάση την εμπειρία του χρήστη, είναι εφικτή η αναδιάρθρωση του προγράμματος συντήρησης των αεροσκαφών, με αποτέλεσμα τη μείωση του χρόνου καθήλωσης αυτών κατά τις επιθεωρήσεις, και κατά συνέπεια τον μεγαλύτερο χρόνο πτητικότητας αυτών γεγονός που αποφέρει σημαντική αύξηση των κερδών των αεροπορικών εταιρειών.

Οι εργασίες συντήρησης αεροσκαφών συνήθως εφαρμόζονται σύμφωνα με κύκλους πτήσης ή ώρες πτήσης ή ημερολογιακές επιθεωρήσεις, ανάλογα πάντα, με τον τύπο του αεροσκάφους. Οι εργασίες συντήρησης αεροσκαφών μπορούν να κατηγοριοποιηθούν σε τρείς κύριες ομάδες, ανάλογα με τις διαδικασίες τους:

- *Φιλοσοφία συντήρησης τύπου "Hard Time"*. Οι εργασίες αυτού του τύπου έχουν προληπτικό χαρακτήρα. Μέχρι το 1960, όλη η αεροπορική συντήρηση ήταν βασισμένη σε αυτή τη φιλοσοφία των προληπτικών αντικαταστάσεων. Η συντηρητική αυτή προσέγγιση, περιλάμβανε την αποσυναρμολόγηση και επιθεώρηση των διαφόρων τμημάτων του αεροσκάφους σε χρονική στιγμή πολύ πριν την πιθανή αστοχία αυτών.
- *Φιλοσοφία συντήρησης τύπου "On Condition"*. Η εμπειρία μέσα από την πάροδο των χρόνων στην αεροπορική βιομηχανία έδειξε ότι δεν ήταν απαραίτητη η περιοδική επιθεώρηση κάθε τμήματος του αεροσκάφους. Επιπλέον, η διασφάλιση της συντήρησης αυτών των τμημάτων μπορούσε να πραγματοποιηθεί και με επιθεώρηση αυτών, χωρίς την αποσυναρμολόγηση τους. Η φιλοσοφία αυτή, η οποία είναι

επίσης προληπτική, εφαρμόζεται σε εξαρτήματα περιοδικά και μέσα στο χρόνο αστοχίας τους.

- *Φιλοσοφία συντήρησης τύπου "Condition Based Monitoring"*. Σε αντίθεση με τους δύο προηγούμενους τύπους συντήρησης, η φιλοσοφία αυτή είναι διορθωτική και όχι προληπτική. Αφορά εξαρτήματα στα οποία η αστοχία τους δεν παρεμποδίζει τη λειτουργία της αξιοπιστίας του συστήματος.

Τα διάφορα προγράμματα συντήρησης περιλαμβάνουν και τις τρείς παραπάνω φιλοσοφίες σε κάποια αναλογία. Ωστόσο, το κόστος συντήρησης της κάθε περίπτωσης δίνει κάποιο πλεονέκτημα στις φιλοσοφίες συντήρησης τύπου "On Condition" και "Condition Based Monitoring". Ο πίνακας που ακολουθεί παρουσιάζει συγκριτικά τις τρεις προαναφερθείσες φιλοσοφίες συντήρησης αεροσκαφών.

Φιλοσοφία Συντήρησης	**Τύπος**	**Εφαρμογές**	**Πλεονεκτήματα**	**Μειονεκτήματα**
Hard Time	Προληπτική	Απλά εξαρτήματα	Ευκολία στην σχεδίαση και πρόληψη των απαιτήσεων υλικών	Υψηλό κόστος
On Condition	Προληπτική	Απλά ή πιο πολύπλοκα εξαρτήματα	Θεωρητικά η χαμηλότερου κόστους διαδικασία συντήρησης	Συνέπειες κόστους από την αστοχία εξαρτήματος
Condition Based Monitoring	Διορθωτική	Ακριβά και πολύπλοκα εξαρτήματα	Βελτιωμένη ποιότητα και μείωση του κόστους	Επενδύσεις σε συστήματα παρακολούθησης της αξιοπιστίας συστημάτων

Πίνακας 1.1. Φιλοσοφίες Συντήρησης Αεροσκαφών *(Πηγή: U. S. Department of Transportation/Federal Aviation Administration, 2003)*

Το γράφημα που ακολουθεί είναι ένα γενικό διάγραμμα αντικατάστασης αεροσκάφους και δείχνει μερικές από τις συνέπειες που θα βρεθεί αντιμέτωπη μια αεροπορική εταιρεία. Βέβαια, η διαχείριση ενός αεροπορικού στόλου είναι δύσκολη

και πολύπλοκη δουλειά με ενδεχόμενα καθυστέρησης ή ακύρωσης πτήσεων, να είναι καθημερινό φαινόμενο. Η σωστή και ορθή αντιμετώπιση τέτοιων φαινομένων εξαρτάται από το πόσο άρτια στελεχωμένη και καλά οργανωμένη είναι η επιχείρηση.

Σχήμα 1.2. Τυπικό Διάγραμμα Αντικατάστασης Αεροσκάφους *(Πηγή: U. S. Department of Transportation/Federal Aviation Administration, 2003)*

Η συχνότητα εφαρμογής της προγραμματισμένης συντήρησης των αεροσκαφών της πολιτικής αεροπορίας είναι ένας συνδυασμός των ωρών πτήσης και των απογειώσεων – κύκλων προσγειώσεων και μπορεί να εφαρμοστεί σε κάθε τμήμα που έχει την τεχνική δυνατότητα και τον κατάλληλο εξοπλισμό για την παροχή αυτής. Λειτουργώντας σύμφωνα με τους κανονισμούς που θεσπίζει η FAA (*Federal Aviation Administration* – οργανισμός θέσπισης διεθνών κανόνων και πολιτικής συντήρησης αεροσκαφών), υφίστανται τεσσάρων ειδών έλεγχοι – επιθεωρήσεις για κάθε αεροσκάφος, οι οποίες διαφέρουν ως προς το σκοπό τους, τη διάρκεια και την συχνότητα:

- *Έλεγχος Τύπου Α ("Α" Check):* Η πρώτη σημαντική επιθεώρηση, η οποία εφαρμόζεται κάθε 65-100 ώρες ή περίπου μια φορά την εβδομάδα. Ο έλεγχος αυτός περιλαμβάνει επιθεώρηση όλων των κυρίων συστημάτων, όπως το σύστημα προσγείωσης, τους κινητήρες και τις επιφάνειες ελέγχου.
- *Έλεγχος Τύπου Β ("Β" Check):* Η επόμενη σημαντική επιθεώρηση εφαρμόζεται κάθε 300 – 600 ώρες πτήσης και εμπεριέχει ενδελεχείς οπτικούς ελέγχους καθώς και λίπανση όλων των κινούμενων επιφανειών, όπως είναι οι επιφάνειες κλίσης και το ουραίο πτέρωμα. Εκτός από εξαιρετικές περιπτώσεις τόσο αυτός ο έλεγχος, όσο και ο προηγούμενος πραγματοποιούνται κατά τη διάρκεια της νύχτας, ώστε το αεροσκάφος να είναι εν ενεργεία την επόμενη μέρα.
- *Έλεγχοι Τύπου C, D ("C", "D" Checks):* Πρόκειται για ιδιαίτερα εκτεταμένους ελέγχους οι οποίοι εφαρμόζονται κάθε 1 – 4 χρόνια αντίστοιχα και απαιτούν την καθήλωση του αεροσκάφους για περίπου ένα μήνα κάθε φορά.

Η εφαρμογή των ελέγχων τύπου C, D γίνεται σε τεχνικούς οργανισμούς ειδικά εξοπλισμένους και με την κατάλληλη τεχνογνωσία και έπειτα από τη σύναψη συμβολαίου με την ενδιαφερόμενη αεροπορική εταιρεία. Αντίθετα, οι έλεγχοι Α, Β είναι μικρής έκτασης και πραγματοποιούνται σε κατάλληλα αεροδρόμια που έχουν τη δυνατότητα παροχής υποστήριξης.

Ο πίνακας που ακολουθεί δείχνει το πρόγραμμα συντήρησης αεροσκαφών της Ολυμπιακής Αεροπορία. Ο πίνακας αποτελεί ένα πλάνο προγραμματισμένων συντηρήσεων, ανάλογα τον τύπο αεροσκάφους και του αριθμού πτήσεων που έχει στο ενεργητικό του. Για παράδειγμα, το αεροσκάφος A300-600 που φαίνεται στη δεύτερη γραμμή του πίνακα όταν κλείσει 465 ώρες πτήσης πρέπει να προγραμματιστεί για την συντήρηση με κωδικό SC-3. Οι συντηρήσεις αυτές αφορούν ελέγχους και επισκευές σε διάφορα τμήματα του αεροσκάφους και χωρίζονται σε κατηγορίες όπως φαίνεται στην πρώτη στήλη του πίνακα. Από τις παρατηρήσεις που φαίνονται στην τελευταία στήλη (remarks), οι μηχανικοί αναλαμβάνουν να συγκεντρώσουν όλα τα στοιχεία που αφορούν τους ελέγχους που αναφέρονται στο αντίστοιχο κελί του πίνακα και να δώσουν τις σωστές διαδικασίες εκτέλεσης της

συντήρησης. Οι συντηρήσεις των αεροσκαφών είναι προγραμματισμένες με τέτοιο τρόπο ώστε κάθε φορά που ένα αεροσκάφος εισάγεται για προγραμματισμένη συντήρηση να ελέγχονται τα απαραίτητα κάθε φορά τμήματα του και εκείνα που καταπονούνται περισσότερο σε κάθε πτήση όπως το σύστημα προσγείωσης τα συστήματα ελέγχου κτλ.

MAINTENANCE SYSTEM FOR OA FLEET						
A/C TYPE AND INSPECTION FREQ		**-**	**B737-484 B737-300**	**A300-600**	**A340**	**REMARKS**
LINE MAINTENANCE SERVICING	**SC 1**	-	NOTE 1	NOTE 2	NOTE 3	WALK AROUND, COCKPIT AND CABIN CHECKS, SERVICING, CLEANING, RECTIFICATION OF ALL SNAGS.
	SC 2	-	300 FH	8 DAYS	7 DAYS	MORE DETAILED INSPECTION OF THE A/C, SERVICING AND CLEANING, RECTIFICATION OF SNAGS DEFERRED TO SC-2
	SC 3	-	600FH	465 FH	----------	SAME AS SC-2 PLUS ADDITIONAL INSPECTION OF STRUCTURES AND SYSTEMS WITH HIGHER STANDARDS
LIGHT MAINT.	**MI-1**	-	1200FH	930 FH	600 FH	INSPECTION OF STRUCTURES AND SYSTEMS, RECTIFICATION OF ALL SNAGS DEFERRED, INCORPORATION OF MOD'S, SERVICING AND CLEANING
HEAVY MAINTENANCE	**MI-3**	-	4800 FH	18 MOS OR 2250 FC	20 MOS	SYSTEMS AND STRUCTURES INSPECTION (SAMPLE, ACCESS), CLEANING AND SERVICING, INCORPORATION OF MOD'S, RECTIFICATION OF SNAGS AND DEFERRED ITEMS.
	MID-D	-		5 YRS	5 YRS & 6 MOS	STRUCTURAL INSPECTIONS, CORROSION PROTECTION, SYSTEMS OPERATIONAL AND FUNCTIONAL TESTS, ADJUSTMENT AND RIGGING
	MI-4	-	24000 FH	10 YRS	10 YRS	STRUCTURAL INSPECTIONS, CORROSION PROTECTION, SYSTEMS OPERATIONAL AND FUNCTIONAL TESTS, ADJUSTMENT AND RIGGING.

Πίνακας 1.2. Πρόγραμμα Συντήρησης Στόλου Αεροσκαφών Ολυμπιακής Αεροπορίας (*Πηγή: Τμήμα Μελετών Αεροσκαφών, Ολυμπιακή Αεροπορία*)

1.3 Καθορισμός του προβλήματος

Το πρόβλημα της ανάπτυξης ενός τέτοιου συστήματος είναι αρκετά δύσκολο και περίπλοκο, περιλαμβάνει μάλιστα πολλά στάδια μέχρι την υλοποίηση του. Αρχικά έγινε πιστή καταγραφή των μεθόδων που χρησιμοποιούν οι τεχνικοί και μηχανικοί συντήρησης αεροσκαφών. Στη συνέχεια αποσπάσθηκε η γνώση των εμπλεκομένων φορέων του τομέα και ακολούθησε η ανάκτηση διάφορων εγχειριδίων. Από όλη αυτήν την διαδικασία έγινα αξιολόγηση των στοιχείων για το ποια θα χρησιμοποιηθούν.

Επόμενο στάδιο είναι αυτό της επιλογής του έμπειρου συστήματος με το οποίο θα υλοποιηθεί το σύστημα διάγνωσης και επιδιόρθωσης βλαβών. Ύστερα από αναζήτηση στο διαδίκτυο αλλά και στην σχετική βιβλιογραφία του χώρου των έμπειρων συστημάτων, επιλέχθηκε το σύστημα κανόνων CLIPS και αυτό για διάφορους λόγους. Κάποιοι από τους λόγους που μας οδήγησαν στην επιλογή του συγκεκριμένου έμπειρου συστήματος είναι: 1) το σύστημα κανόνων CLIPS χρησιμοποιεί την προς τα εμπρός συλλογιστική, 2) η επαρκής βιβλιογραφία που κυκλοφορεί, 3) το τελικό προϊόν που θα προκύψει είναι μεταφέρσιμο σε όλα τα γνωστά λειτουργικά συστήματα, 4) παρέχεται δωρεάν από το διαδίκτυο μέσα από την σχετική ιστοσελίδα και τέλος 5) υπάρχει ομάδα συζητήσεων στο διαδίκτυο, η οποία παρακολουθείται από τους ίδιους τους κατασκευαστές του CLIPS.

Η ολοκλήρωση όλων των παραπάνω σταδίων είχαν σαν αποτέλεσμα την ανάπτυξη ενός έμπειρου συστήματος διάγνωσης και επιδιόρθωσης βλαβών, με την ονομασία A300-600 Troubleshooter Expert System.

1.4 Στόχοι και δομή της εργασίας

Στόχος της συγκεκριμένης εργασίας είναι η ανάπτυξη ενός έμπειρου συστήματος για διάγνωση και επιδιόρθωση βλαβών για το αεροσκάφος A300-600 της AIRBUS INDUSTRIES. Αν και η τεχνολογική ανάπτυξη στον χώρο της αεροπορικής βιομηχανίας έχει κάνει τεράστια άλματα προόδου, υπάρχουν ακόμα πολλοί τομείς οι οποίοι παρουσιάζουν ερευνητικό ενδιαφέρον όπως, ο τομέας της συντήρησης

αεροσκαφών. Η συντήρηση αεροσκαφών αποτελεί σημαντικό και νευραλγικό παράγοντα στην λειτουργία μιας αεροπορικής εταιρείας μιας και από αυτόν εξαρτάται το πτητικό μέρος της.

Στο σημείο αυτό, θα πρέπει να αναφέρουμε ότι σκοπός μας επίσης είναι, η υλοποίηση του συστήματος και η εφαρμογή του. Έτσι κατασκευάστηκε το πρόγραμμα με την ονομασία A300-600 Troubleshooting Expert System, στο σύστημα κανόνων CLIPS.

Όσον αφορά τη δομή της εργασίας, στο *δεύτερο κεφάλαιο*, παρουσιάζουμε τον επιστημονικό χώρο των εμπείρων συστημάτων και των βάσεων γνώσης και αναφέρουμε τις μεθοδολογίες που ακολουθούμε για την ανάπτυξη τους.

Στο *τρίτο κεφάλαιο*, παρουσιάζουμε το σύστημα κανόνων του CLIPS.

Στο *τέταρτο κεφάλαιο*, καταγράφουμε τα βήματα που ακολουθήσαμε για την ανάλυση και σχεδίαση της βάσης γνώσης και στο *πέμπτο κεφάλαιο* παρουσιάζεται το σύστημα μέσω εφαρμογών.

Τέλος στο έκτο κεφάλαιο αναφέρονται τα συμπεράσματα αξιολόγησης της παρούσας εργασίας.

ΚΕΦΑΛΑΙΟ 2

Έμπειρα Συστήματα

2.1 Εισαγωγή

Το υλικό που χρησιμοποιήθηκε για το θεωρητικό κομμάτι των έμπειρων συστημάτων και αναλύεται στο κεφάλαιο αυτό, προέρχεται από τις πανεπιστημιακές σημειώσεις του μαθήματος Μηχανική Γνώσεων και Αποφάσεων, Πολυτεχνείο Κρήτης (Ματσατσίνης, 2004) (παράγραφοι 2.4, 2.5, 2.6, 2.6.1, 2.6.2, 2.6.3), και από το βιβλίο 'Τεχνητή Νοημοσύνη, εκδόσεις Γαρταγάνη, Θεσσαλονίκη (Βλαχάβας Ι., Κεφαλάς Π., Βασιλειάδης Ν., Ρεφανίδης Ι., Κόκορας Φ., Σακελλαρίου Η., 2002) (παράγραφοι 2.4, 2.4.1, 2.4.2).

Τα έμπειρα συστήματα αποτελούν το γνωστότερο πεδίο εφαρμογής της τεχνητής νοημοσύνης. Είναι προγράμματα, που συνδυάζουν τη γνώση των ειδικών και προσπαθούν να επιλύσουν ειδικά προβλήματα, μιμούμενα τη διαδικασία της λογικής των.

Η ανάπτυξη τους άρχισε άρχισε από τα τέλη της δεκαετίας του 1960. Αυτή τη περίοδο οι προσπάθειες ήταν στραμμένες προς την επίλυση προβλημάτων γενικού σκοπού (general-purpose problem solver) που είχε αναπτυχθεί από τους Newell και Simon (1972) στην προσπάθεια τους για δημιουργία ενός "νοήμονος υπολογιστή". Σε ένα πρωτοπόρο άρθρο τους, οι Newell και Simon (1958), ανέπτυξαν την ιδέα ότι ένα σύστημα, που θα χρησιμοποιεί κανόνες παραγωγής (production rules), θα μπορεί να μοντελοποιήσει μερικές μορφές ανθρώπινης διαδικασίας επίλυσης προβλημάτων. Το πέρασμα από τα προγράμματα γενικού-σκοπού στα προγράμματα ειδικού-σκοπού έγινε στα μέσα της δεκαετίας του 1960 με την ανάπτυξη του DENDRAL από τον Feigenbaum στο Πανεπιστήμιο του Stanford. Η παρατήρηση, ότι η γνώση, που απαιτείται για την διεξαγωγή ειδικών εργασιών, ήταν υψηλότερου επιπέδου από την γνώση της κοινής λογικής, που εχρησιμοποιείτο μέχρι τότε στην ανάπτυξη συστημάτων Τεχνητής Νοημοσύνης, οδήγησε στη ραγδαία ανάπτυξη των Ε.Σ. (Goodman et al., 1991). Από τα μέσα του '70 άρχισε η συνεχής ανάπτυξη τέτοιων

συστημάτων, τα οποία απαιτούσαν την επεξεργασία μεγάλης μάζας εξειδικευμένης γνώσης. Η ισχύς ενός Ε.Σ. εξαρτάται κυρίως από τη γνώση που διαθέτει και όχι από τους χρησιμοποιούμενους φορμαλισμούς αναπαράστασης της γνώσης 'η τους μηχανισμούς εξαγωγής συμπερασμάτων.

Ο Feigenbaum (1982) θεωρεί ότι ένα έμπειρο σύστημα είναι ένα έξυπνο πρόγραμμα το οποίο για να λύσει κάποιο πρόβλημα, που είναι αρκετά δύσκολο ώστε να απαιτεί σημαντική εμπειρία για την επίλυσή του, χρησιμοποιεί ειδική γνώση και διαδικασίες εξαγωγής συμπερασμάτων. Η γνώση που είναι απαραίτητη για τη λειτουργία σε αυτό το επίπεδο, μαζί με τις διαδικασίες εξαγωγής συμπερασμάτων, μπορεί να θεωρηθούν σαν ένα μοντέλο εμπειρίας των καλύτερων ειδικών του χώρου.

Από τις αρχές του '80 οι τεχνολογίες ανάπτυξης Ε.Σ., που περιορίζονται σε πανεπιστημιακούς χώρους άρχισαν να εφαρμόζονται για την ανάπτυξη εμπορικών εφαρμογών (R-1, XCON, XCEL, CATS-1 κ.α.). Ταυτόχρονα άρχισαν οι προσπάθειες ανάπτυξης ειδικών εργαλείων, που θα συνέβαλαν στην ταχύτερη ανάπτυξη Ε.Σ. (EMYCIN, AGE, KAS, EURISKO, M.1, EXSYS κ.α.) και τα οποία άρχισαν να εμφανίζονται στην αγορά από το 1983.

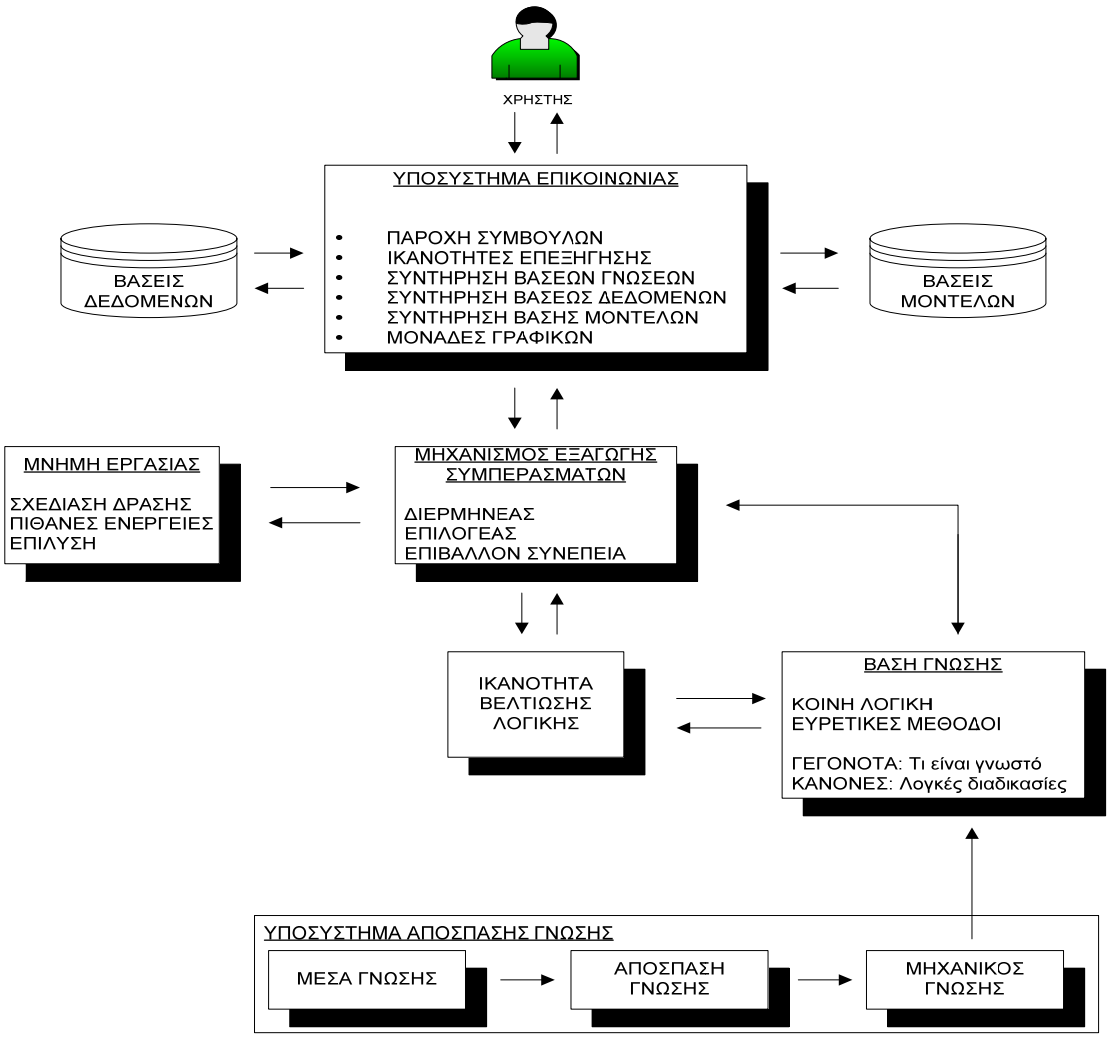
Στη ραγδαία ανάπτυξη των Ε.Σ. οδήγησε η από μέρους τους κάλυψη υπαρχόντων αναγκών της αγοράς. Η όλο και αυξανόμενη ανάγκη για όσο το δυνατόν πιο εξειδικευμένες συμβουλές, σε όλα τα επίπεδα αφενός και αφετέρου η έλλειψη και το κόστος εξειδικευμένου προσωπικού ώθησε τις επιχειρήσεις να επενδύουν τεράστια ποσά για την ανάπτυξη έμπειρων συστημάτων.

2.2 Δομή έμπειρων συστημάτων

Κάθε έμπειρο σύστημα έχει δύο όψεις. Η μία είναι αυτή που βλέπει ο κατασκευαστής του, κατά τη φάση ανάπτυξης και εισαγωγής της γνώσης σε αυτό, ενώ η άλλη είναι αυτή που αντιλαμβάνεται ο χρήστης του συστήματος κατά τη διάρκεια της λειτουργίας του όταν δέχεται τις συμβουλές του.

Μια τυπική διαδικασία χρήσης ενός έμπειρου συστήματος είναι αρχικά να διατυπωθεί ένα αίτημα, από το χρήστη, για παροχή μιας συμβουλής. Στη συνέχεια το Ε.Σ. χρησιμοποιεί το μηχανισμό εξαγωγής συμπερασμάτων για να χειρισθεί τους κανόνες της βάσης γνώσης. Αν το σύστημα έχει κάποιες απορίες σχετικά με κάποια γεγονότα (facts) τότε απευθύνεται στο χρήστη για πρόσθετες πληροφορίες. Τα γεγονότα αυτά καθώς και όσα προκύπτουν στα διάφορα ενδιάμεσα στάδια λειτουργίας του Ε.Σ. αποθηκεύονται στη μνήμη εργασίας. Η διαδικασία συνεχίζεται μέχρι το σύστημα να καταλήξει σ ένα τελικό συμπέρασμα. Το συμπέρασμα αυτό μπορεί να είναι είτε η παροχή συμβουλής στο αρχικό ερώτημα. Είτε να δηλώσει αδυναμία παροχής συμβουλής επειδή δεν υπάρχει επαρκής γνώση για αυτό.

Στη συνέχεια θα εξετασθούν αναλυτικότερα τα διάφορα τμήματα ενός αντιπροσωπευτικού έμπειρου συστήματος όπως αυτού που παρουσιάζεται στο σχήμα 2.1.



Σχήμα 2.1 Δομή ενός Έμπειρου Συστήματος *(Πηγή: Ματσατσίνης, 2004)*

Χρήστης

Ο χρήστης ενός έμπειρου συστήματος μπορεί να είναι κάποιος μη ειδικός ο οποίος χρησιμοποιεί το Ε.Σ. γιατί χρειάζεται τη συμβουλή του ώστε να βελτιώσει τη ποιότητα των αποφάσεων που παίρνει. Χρήστης μπορεί να είναι κάποιος ο οποίος χρησιμοποιεί το Ε.Σ. για να αυξήσει τη γνώση του πάνω σε ένα ειδικό θέμα ή κάποιος που να θέλει μια δεύτερη γνώμη σε κάποια απόφαση που θέλει να πάρει ή για να τον βοηθά σε κάποιες δευτερεύουσες εργασίες. Τέλος χρήστης μπορεί να είναι και ένα άτομο που θέλει να παρακολουθεί τη λογική , μέσω των σταδιακών βημάτων που ακολουθεί το σύστημα, για να οδηγηθεί σε κάποιο συμπέρασμα.

Σύστημα επικοινωνίας

Μέσω του υποσυστήματος αυτού και με τη βοήθεια διαφόρων γραφικών οθονών για καλύτερα αποτελέσματα εξασφαλίζεται η επικοινωνία του χρήστη με το σύστημα σε φυσική γλώσσα.

Ο χρήστης μπορεί να επικοινωνεί με το σύστημα και να συντηρεί τις διάφορες βάσεις (γνώσης, δεδομένων, μοντέλων) που είναι απαραίτητες για τη σωστή λειτουργία του.

Οι συμβουλές που δέχεται, λαμβάνονται μέσω του υποσυστήματος αυτού. Ο χρήστης μπορεί να εκμεταλλευτεί τις ικανότητες επεξήγησης και να λάβει απαντήσεις στα ερωτήματα:

- Πως οδηγήθηκε σε κάποιο συμπέρασμα;
- Γιατί απορρίφθηκαν κάποιες εναλλακτικές απαντήσεις;
- Ποια είναι η διαδικασία για να καταλήξει το σύστημα στο τελικό συμπέρασμα;
- Γιατί το σύστημα υποβάλλει κάποιες ερωτήσεις στη προσπάθεια του να οδηγηθεί στη τελική απάντηση;

Μηχανισμός εξαγωγής συμπερασμάτων

Η καρδία ενός Ε.Σ. είναι το τμήμα εκείνο του προγράμματος που είναι γνωστό σαν μηχανισμός εξαγωγής συμπερασμάτων. Ο μηχανισμός αυτός είναι υπεύθυνος για τη διαχείριση και τον έλεγχο της γνώσης που βρίσκεται αποθηκευμένη τόσο στη βάση γνώσης όσο και στη μνήμη εργασίας, με στόχο τη διαμόρφωση συμπερασμάτων. Τα κύρια μέρη ενός τέτοιου μηχανισμού είναι ο διερμηνέας (interpreter), ο επιλογέας ή χρονοπρογραμματιστής (scheduler) και το τμήμα επιβολής συνέπειας (consistency enforcer).

Ο διερμηνέας, στα περισσότερα συστήματα είναι γνωστός σαν διερμηνέας κανόνων, ασχολείται δε με την εκτέλεση των επιλεγέντων ενεργειών, εφαρμόζοντας στη βάση γνώσης τους αντίστοιχους κανόνες με σκοπό την παραγωγή νέας γνώσης. Για το

σκοπό αυτό εφαρμόζει διάφορες τεχνικές όπως στατιστικές μεθόδους ή μεθόδους ταυτοποίησης προτύπων (pattern matching). Με βάση αυτές τις μεθόδους εκτιμά τις πληροφορίες που βρίσκονται αποθηκευμένες στη βάση γνώσης, ώστε να ανακαλύψει τους ενδεδειγμένους κανόνες.

Ο επιλογέας είναι υπεύθυνος για τη στρατηγική ελέγχου του συστήματος, έτσι ασχολείται με τη τήρηση της σειράς εκτέλεσης των διαφόρων ενεργειών και υπολογίζει τα αποτελέσματα εφαρμογής των κανόνων υπό το φώς των προτεραιοτήτων των δεδομένων ή άλλων κριτηρίων της διάταξης ενεργειών (agenda). Αποφασίζει στην ουσία πότε και με ποια σειρά θα χρησιμοποιηθούν τα δεδομένα της βάσης γνώσης. Περιέχει αφενός μεν γνώση εκτίμησης (assessment knowledge), ώστε να μπορεί να αξιολογεί τις εναλλακτικές διαδρομές έρευνας, αφετέρου δε γνώση ελέγχου (control knowledge) για να μπορεί να συντονίζει τις διάφορες λειτουργίες. Συχνά είναι απαραίτητο να περιέχεται στο μηχανισμό και γνώση, μια και έχει ήδη αναφερθεί ότι η γνώση είναι διαχωρισμένη από το μηχανισμό εξαγωγής συμπερασμάτων. Στις περιπτώσεις αυτές, όπου δηλαδή ο μηχανισμός εξαγωγής συμπερασμάτων εμπεριέχει γνώση, θα θεωρείται ότι είναι εξαρτώμενος από τη συγκεκριμένη ειδική γνώση.

Ο επιβάλλον τη συνέπεια (consistency enforcer), επιχειρεί να διατηρεί μια συνεπή αναπαράσταση της εμφανιζόμενης λύσης.

Η μνήμη εργασίας είναι ένας χώρος ον οποίο χρησιμοποιεί ο μηχανισμός εξαγωγής συμπερασμάτων, είτε για να τοποθετεί τις απαραίτητες πληροφορίες του τρέχοντος προβλήματος, είτε για να αποθηκεύει ενδιάμεσα αποτελέσματα στη πορεία του προς αναζήτηση της τελικής λύσης. Όπως φαίνεται και στο παραπάνω σχήμα, στη μνήμη εργασίας καταχωρούνται τα ακόλουθα τρία είδη αποφάσεων:

<u>Σχεδίαση δράσης:</u> Αναφέρεται στο τρόπο επίλυσης του τρέχοντος προβλήματος.
<u>Πιθανές ενέργειες:</u> Είναι ένας κατάλογος των ενδεχόμενων επόμενων ενεργειών.
<u>Επίλυση:</u> Εδώ καταχωρούνται οι εναλλακτικές υποθέσεις και οι ενέργειες που το σύστημα έχει επιτελέσει μέχρι τώρα.

Βάσεις γνώσεις

Στη βάση γνώσης περιέχονται δύο βασικά στοιχεία, τα γεγονότα (facts) και οι κανόνες (rules). Τα γεγονότα αναφέρονται τόσο στη κατάσταση του προβλήματος, όσο και στην αντίστοιχη θεωρία. Οι κανόνες κατευθύνουν την εκμετάλλευση της υπάρχουσας γνώσης για την επίλυση του τρέχοντος προβλήματος. Συχνά αντί για κανόνες χρησιμοποιούνται ευριστικοί (heuristics). Αυτοί περιέχουν ειδική γνώση για τη μεθοδολογία αποτελεσματικής και αποδοτικής επίλυσης ειδικών και σύνθετων προβλημάτων, καθώς και κάθε άλλης σχετικής πληροφορίας.

Υποσύστημα απόσπασης γνώσης

Το υποσύστημα αποτελείται από τα μέσα γνώσης του πεδίου και τον μηχανισμό γνώσης ο οποίος εργάζεται με στόχο την απόσπαση, συγκέντρωση, κωδικοποίηση και μεταφορά της γνώσης-εμπειρίας κάποιου πεδίου, σε μια βάση γνώσης ενός Ε.Σ. Σαν πηγές γνώσης, εκτός από έναν ή περισσότερους ειδικούς, μπορούν να χρησιμοποιηθούν οι γνώσεις που συγκεντρώνονται από τη διεθνή βιβλιογραφία, τα ειδικά περιοδικά του χώρου, τα αποτελέσματα ειδικών ερευνών, τις βάσεις δεδομένων κ.α.

Μηχανική της γνώσης

Σαν γνώση μπορεί να θεωρηθεί μια συλλογή ειδικών γεγονότων (facts), διαδικασιών (procedures) και κανόνων (rules). Μερικά από τα είδη, της χρησιμοποιούμενης γνώσης στα Ε.Σ. παρουσιάζονται στο σχήμα 2.2 (Fikes και Kehler, 1985). Όλες οι ενέργειες που αναφέρονται στη γνώση είναι αποτελούν το αντικείμενο, της μηχανικής της γνώσης (Feigenbaum και McCorduck, 1983) ενώ οι ειδικοί που ασχολούνται με αυτή ονομάζονται μηχανικοί γνώσης. Η μηχανική της γνώσης εμπεριέχει και τη συνεργασία των ειδικών (experts), ενός τομέα, με το μηχανικό γνώσης με στόχο τη κωδικοποίηση της γνώσης των και το καθορισμό με σαφήνεια των λογικών διαδικασιών που ο ειδικός χρησιμοποιεί για να επιλύσει ένα πραγματικό πρόβλημα (Σχήμα 2.3). Πολλές φορές η χρησιμοποιούμενη γνώση είναι αδόμητη και συγκεχυμένη. Η διαδικασία ανάπτυξης ενός έμπειρου συστήματος και η συνεργασία του ειδικού με το μηχανισμό γνώσης τον οδηγεί (εξαναγκάζει) να βρει τρόπους να

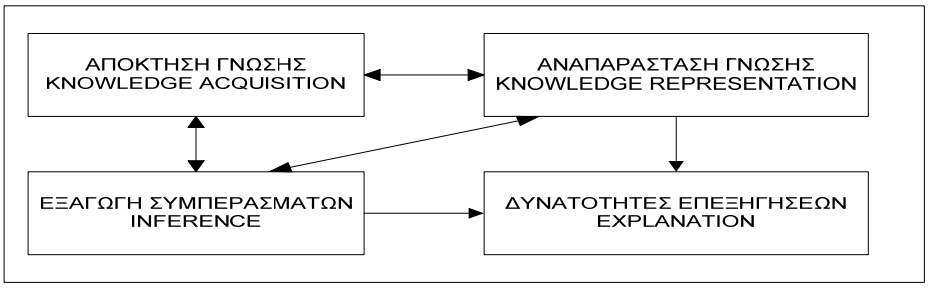
εκφράσει τη γνώση και τον τρόπο σκέψης του με μεγάλη σαφήνεια (Goodman et al., 1991).

ΕΙΔΗ ΓΝΩΣΗΣ
ΔΙΑΔΙΚΑΣΙΕΣ ΠΕΡΙΟΡΙΣΜΟΙ ΓΕΓΟΝΟΤΑ ΓΥΡΩ ΑΠΟ ΕΝΑ ΠΕΔΙΟ ΑΒΕΒΑΙΑ ΓΕΓΟΝΟΤΑ ΓΕΝΙΚΗ ΓΝΩΣΗ ΔΙΑΖΕΥΚΤΙΚΑ ΓΕΓΟΝΟΤΑ ΑΝΤΙΚΕΙΜΕΝΑ ΚΑΙ ΣΧΕΣΕΙΣ ΟΡΙΣΜΟΙ ΛΕΞΙΛΟΓΙΟΥ ΠΕΡΙΓΡΑΦΕΣ ΣΥΜΠΕΡΙΦΟΡΑΣ ΚΑΙ ΠΙΣΤΕΥΩ ΕΥΡΕΤΙΚΕΣ ΚΑΙ ΚΑΝΟΝΕΣ ΑΠΟΦΑΣΗΣ ΔΙΑΔΙΚΑΣΙΕΣ ΓΙΑ ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΩΝ ΤΥΠΙΚΕΣ ΚΑΤΑΣΤΑΣΕΙΣ ΥΠΟΘΕΣΕΙΣ ΜΕΤΑΓΝΩΣΗ

Σχήμα 2.2. Είδη Γνώσης

Ο μηχανικός της γνώσης πρέπει να δώσει μεγάλη προσοχή στο διαχωρισμό των δομών της γνώσης από τους μηχανισμούς ελέγχου. Επίσης το Ε.Σ. θα πρέπει να έχει την ικανότητα να εξηγεί τι έκανε (what), γιατί το έκανε (why) και πως το έκανε (how).

Ένα άλλο σημαντικό πρόβλημα εκτός από την απόσπαση της γνώσης είναι και ο τρόπος αναπαράστασης της ούτως ώστε να είναι δυνατή η λειτουργία του μηχανισμού εξαγωγής συμπερασμάτων του Ε.Σ.



Σχήμα 2.3. Δραστηριότητες

Απόκτηση γνώσης (knowledge acquisition)

Η πολυπλοκότητα της διαδικασίας απόκτησης γνώσης οφείλεται, κατά κύριο λόγο, στην ύπαρξη αφενός μεν πολλών μέσων πληροφόρησης και αφετέρου δε στα διαφορετικά είδη γνώσης (Boose, 1985; Shaw and Gaines, 1986; Wright and Ayton, 1987; Johnson and Johnson, 1987; McGraw and Harbison, 1989; Brute and Blount, 1989; McGovern et all., 1991). Σημαντικός παράγων της διαδικασίας απόσπασης της γνώσης είναι και το κόστος απόκτησης της (Dos Santos and Mookerjee, 1993).

Διαδικασία απόκτησης γνώσης

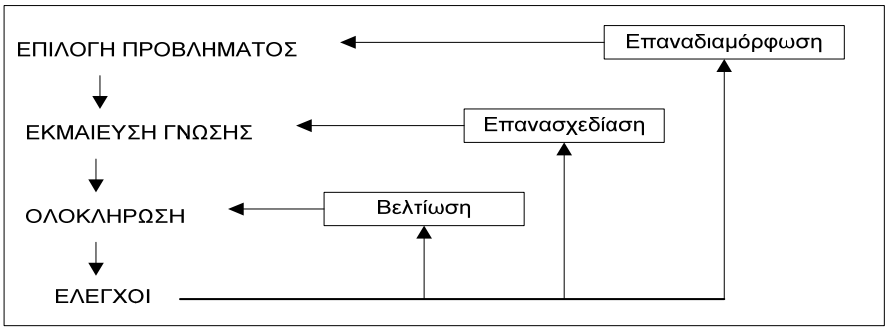
Στο σχήμα 2.4 παρουσιάζονται τα τέσσερα κύρια στάδια απόκτησης της γνώσης (Doukidis and Whitley, 1988). Κατά τη διάρκεια του πρώτου σταδίου γίνεται αρχικά η επιλογή του προς επίλυση προβλήματος, στη συνέχεια αναγνωρίζονται. Κατά το στάδιο εκμαίευσης της γνώσης από έναν ή πολλούς ειδικούς, αρχικά γίνεται η απόσπαση της γνώσης (knowledge acquisition), κατά την οποία η αποσπασθείσα γνώση οργανώνεται και αναπαρίσταται από το μηχανικό γνώσης, σε μορφή κατάλληλη για χρήση από το προς ανάπτυξη Ε.Σ.

Κατά το στάδιο της ολοκλήρωσης η αποκτηθείσα γνώση ενσωματώνεται στο Ε.Σ. αφού προηγουμένως διαμορφωθεί καταλλήλως ούτως ώστε να είναι δυνατή η χρήση της από το Ε.Σ.

Στη διαδικασία απόκτησης της γνώσης περιέχονται και οι έλεγχοι που αφορούν την εκτίμηση (evaluation), την εγκυρότητα (validation) και την επαλήθευση (verification) της αποκτηθείσης γνώσης (O'Keefe et al., 1987; Marcot, 1987; Nguyen, 1987; Parsaye, 1988) και τελικά διευκρινίζονται και καθορίζονται με σαφήνεια οι χρησιμοποιούμενες έννοιες και σχέσεις για την αναπαράσταση της γνώσης. Αυτή η τελευταία εργασία θεωρείται από πολλούς συγγραφείς σαν ένα ανεξάρτητο στάδιο της όλης διαδικασίας (Hayes-Roth, 1983; 1984; Turban, 1993).

Τα διάφορα στάδια αλληλεπιδρούν μεταξύ τους μέσω των εργασιών επαναδιαμόρφωσης, επανασχεδίασης και βελτίωσης του αναπτυσσόμενου συστήματος.

Το σύστημα βρίσκεται κάτω από τη συνεχή παρακολούθηση και εκτίμηση του ή των ειδικών, η γνώση των οποίων χρησιμοποιήθηκε για την ανάπτυξη του.



Σχήμα 2.4. Στάδια Απόκτησης Γνώσης

Σημαντικός παράγων στην ανάπτυξη ενός Ε.Σ. είναι ο αριθμός των συμμετεχόντων ειδικών (multiple experts). Η χρησιμοποίηση πολλών ειδικών επιβάλλεται κυρίως είτε για λόγους διεύρυνσης των προτεινόμενων λύσεων ενός προβλήματος, είτε για να αυξηθούν οι δυνατότητες διαφορετικών λογικών προσεγγίσεων. Αυτή η συνεργασία παρουσιάζει και μειονεκτήματα όπως αυτά που οφείλονται στις αλληλοσυγκρουόμενες απόψεις-εκτιμήσεις των ειδικών, τις οποίες πρέπει να επιλύσει ο μηχανικός της γνώσης (Boose, 1985; LeClair, 1985; Nii, 1986; Mittal, 1985; Shaw and Gaines, 1986; Alexander and Evans, 1988).

2.3 Χαρακτηριστικά έμπειρων συστημάτων

Μερικά επιθυμητά χαρακτηριστικά που πρέπει να έχουν τα έμπειρα συστήματα είναι τα ακόλουθα (Jackson et al., 1999):

Επεξήγηση και Αιτιολόγηση της πορείας συλλογισμού. Τα έμπειρα συστήματα πρέπει να επεξηγούν τη συλλογιστική πορεία που ακολούθησαν για την εύρεση της λύσης στην οποία κατέληξαν. Δεν είναι δηλαδή αποδεκτή η απλή αναφορά μιας λύσης, αλλά πρέπει και να τεκμηριώνεται με τον ίδιο τρόπο που ένας άνθρωπος-ειδικός θα τεκμηρίωνε τη δική του απόφαση. Για παράδειγμα, για ένα γιατρό που χρησιμοποιεί συμβουλευτικά ένα έμπειρο σύστημα στη διάγνωση ασθενειών, δεν αρκεί η υπόδειξη μόνο της ασθένειας αλλά και η αιτιολόγηση της υπόδειξης. Δηλαδή, με βάση ποια

στοιχεία και ποιους συλλογισμούς έκανε τη διάγνωση το έμπειρο σύστημα. Με τον τρόπο αυτό αυξάνεται η εμπιστοσύνη του χρήστη προς το σύστημα.

Δυναμικότητα. Τα έμπειρα συστήματα πρέπει να επιτρέπουν την αλλαγή της υπάρχουσας γνώσης. Επειδή πολλές φορές η γνώση για κάποιο τομέα της επιστήμης δεν είναι σταθερή, αλλά αλλάζει διαρκώς, θα πρέπει να υπάρχουν οι κατάλληλοι μηχανισμοί για τροποποίηση της υπάρχουσας γνώσης, πρόσθεση καινούριας ή αφαίρεση λανθασμένης γνώσης από το σύστημα.

Ταχύτητα Απόκρισης. Τα έμπειρα συστήματα πρέπει να φτάνουν σε αποτέλεσμα πολύ γρήγορα ή τουλάχιστον σε τόσο χρόνο όσο θα έκανε και ένας άνθρωπος-ειδικός να αποκριθεί σε ένα τυχαίο πρόβλημα. Δεν έχει νόημα το σύστημα να φτάνει σε μια τέλεια λύση και να την τεκμηριώνει, όταν αργεί πάρα πολύ για να το πετύχει αυτό. Σε περιπτώσεις μάλιστα όπου από την ταχύτητα απόκρισης του έμπειρου συστήματος εξαρτώνται άμεσα ή έμμεσα ανθρώπινες ζωές, όπως για παράδειγμα η ιατρική παρακολούθηση ασθενή, ένα αργό έμπειρο σύστημα θα ήταν άχρηστο.

Διαφάνεια του κώδικα. Τα έμπειρα συστήματα περιέχουν σαν κώδικα μια σαφή περιγραφή του προβλήματος με το οποίο ασχολούνται. Αναπαριστούν τη γνώση για το πρόβλημα σε συμβολική μορφή και δε θα έπρεπε να αναμιγνύουν τη γνώση με το μηχανισμό χειρισμού και ελέγχου της.

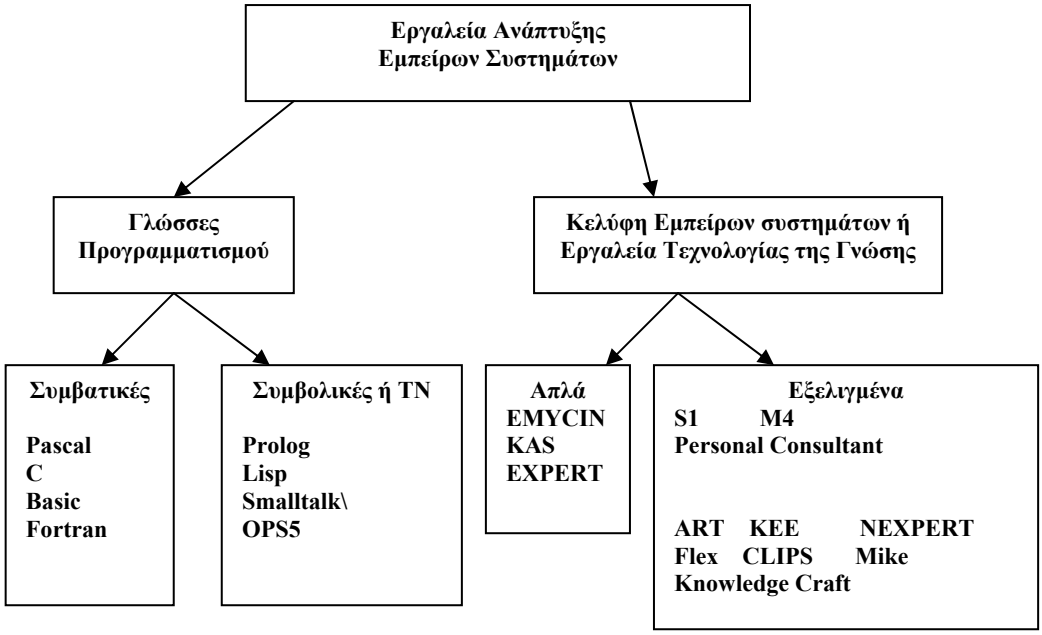
Χειρισμός αβέβαιης ή ελλιπούς γνώσης. Τα έμπειρο σύστημα θα πρέπει να είναι σε θέση να επεξεργαστεί γνώση που είναι ασαφής ή ελλιπής. Υπάρχουν προβλήματα για τα οποία δεν είναι διαθέσιμη όλη η γνώση που απαιτείται για την επίλυση τους. Σε μια τέτοια περίπτωση ένας άνθρωπος-ειδικός θα έπαιρνε μια απόφαση στηριζόμενος στη διαθέσιμη γνώση. Την ίδια συμπεριφορά θα πρέπει να είναι ικανά να επιδεικνύουν και τα έμπειρα συστήματα. Για παράδειγμα, η γνώση του ότι κάποιος ασθενής είναι “ωχρός” δεν μπορεί να καθοριστεί με απόλυτη ακρίβεια παρά μόνο με κάποιους ασαφείς χαρακτηρισμούς όπως “πολύ” ή “λίγο”. Όπως ένας γιατρός θα μπορούσε να προχωρήσει σε διάγνωση με τέτοιου είδους αβέβαιη γνώση έτσι και το έμπειρο σύστημα θα πρέπει να είναι σε θέση να αναπαραστήσει και να χειριστεί τέτοια γνώση.

2.4 Εργαλεία ανάπτυξης

Ένα έμπειρο σύστημα μπορεί να αναπτυχθεί με δύο βασικές κατηγορίες εργαλείων (Σχήμα 2.5), τις γλώσσες προγραμματισμού και τα εργαλεία τεχνολογίας της γνώσης (knowledge engineering tools) ή κελύφη εμπείρων συστημάτων (expert system shells).

Οι γλώσσες προγραμματισμού χωρίζονται σε δύο κατηγορίες, στις συμβατικές και τις συμβολικές ή γλώσσες ΤΝ. Οι συμβατικές γλώσσες δεν ενδείκνυνται για το χειρισμό γνώσης και την έκφραση εννοιών, αλλά έχουν το πλεονέκτημα ότι είναι συμβατές με την υπάρχουσα υπολογιστική υποδομή, όπως βάσεις δεδομένων και άλλα βοηθητικά προγράμματα. Τέτοιες γλώσσες είναι οι BASIC, FORTRAN, C/C++, PASCAL, κλπ. Αντίθετα οι γλώσσες ΤΝ, όπως LISP, PROLOG, κλπ είναι προσανατολισμένες στη χρήση συμβόλων και παρέχουν εξελιγμένους μηχανισμούς χειρισμού συμβολικών εκφράσεων και λιστών, μεταβλητές χωρίς προκαθορισμένο τύπο, αυτόματη διαχείριση μνήμης και ευέλικτες δομές ελέγχου. Οι γλώσσες ΤΝ είναι ευκολότερες στην εκμάθηση από τα εργαλεία τεχνολογίας της γνώσης, γιατί υποστηρίζουν ένα μόνο τρόπο αναπαράστασης γνώσης και συλλογιστικής.

Τα κελύφη είναι εξειδικευμένα εργαλεία για την ανάπτυξη εμπείρων συστημάτων. Ένα κέλυφος παρέχει στο μηχανικό της γνώσης μία ή περισσότερες μεθόδους αναπαράστασης γνώσης καθώς και έναν ή περισσότερους μηχανισμούς εξαγωγής συμπερασμάτων. Τέλος, παρέχει διάφορες ευκολίες διασύνδεσης με το χρήστη και με το περιβάλλον γενικότερα.



Σχήμα 2.5. Εργαλεία Ανάπτυξης Έμπειρων Συστημάτων

Τα εργαλεία προήλθαν από υπάρχοντα συστήματα, από τα οποία είχε απλά αφαιρεθεί η βάση γνώσης, όπως για παράδειγμα το EMYCIN. Τα απλά αυτά κελύφη συνήθως είναι προσανατολισμένα σε εφαρμογές ανάλογες με το έμπειρο σύστημα από το οποίο προήλθαν και έχουν περιορισμένες δυνατότητες αναπαράστασης γνώσης. Αν ένα από τα απλά αυτά κελύφη μετατραπεί έτσι ώστε να υποστηρίζει ευρύ φάσμα εφαρμογών καθώς και πολλούς τρόπους αναπαράστασης γνώσης και συλλογιστικής, τότε αποτελεί ένα εξελιγμένο κέλυφος.

Τα περισσότερα από τα σύγχρονα και εξελιγμένα κελύφη αναπτύχθηκαν από την αρχή σε κάποια γλώσσα προγραμματισμού και δεν προήλθαν από κάποιο έμπειρο σύστημα. Αυτά τα κελύφη παρέχουν συνήθως εξελιγμένες δυνατότητες διασύνδεσης με το χρήστη και το κυριότερο, υποστηρίζουν πολλές διαφορετικές μεθόδους αναπαράστασης της γνώσης και συλλογιστικές. Τα κελύφη αυτά είναι συνήθως δυσκολότερα στην εκμάθηση από τα απλά, γιατί περιέχουν πολλές και ετερογενείς μεταξύ τους προγραμματιστικές έννοιες.

2.4.1 Γλώσσες προγραμματισμού ΤΝ

Οι γλώσσες προγραμματισμού ΤΝ αποτελούν ένα εργαλείο για τη γρήγορη κατασκευή ενός πρωτότυπου του έμπειρου συστήματος. Συνήθως ο κώδικας μπορεί να εκτελεστεί και να ελεγχθεί την ώρα που δημιουργείται. Παρόλα αυτά η διασύνδεση αυτών των γλωσσών με το χρήστη δεν είναι αρκετά εξελιγμένη αν και τα τελευταία χρόνια έχουν δημιουργηθεί βιβλιοθήκες, οι οποίες επεκτείνουν σημαντικά τις δυνατότητες των γλωσσών αυτών.

Οι γλώσσες προγραμματισμού διαθέτουν συνήθως έναν απλό μηχανισμό ελέγχου, αλλά επειδή είναι ευέλικτες, δίνουν στον προγραμματιστή τη δυνατότητα να "δημιουργήσει" το δικό του μηχανισμό ελέγχου για το έμπειρο σύστημα (συλλογιστική), όπως και το δικό του τρόπο αναπαράστασης της γνώσης, με τις δομές δεδομένων. Έτσι οι προγραμματιστές μπορούν να πειραματιστούν με διάφορες μεθόδους και να καταλήξουν στην προσφορότερη για το συγκεκριμένο έμπειρο σύστημα.

Ενδεικτικά αναφέρουμε τις τέσσερις κατηγορίες γλωσσών προγραμματισμού ΤΝ:

- Συναρτησιακός προγραμματισμός (LISP).
- Λογικός προγραμματισμός (PROLOG).
- Αντικειμενοστραφής προγραμματισμός.
- Προγραμματισμός με κανόνες παραγωγής (OPS5).

2.4.2 Κελύφη έμπειρων συστημάτων

Τα κελύφη είναι εξειδικευμένα εργαλεία για την ανάπτυξη εμπείρων συστημάτων και ονομάζονται έτσι γιατί ουσιαστικά αποτελούν έμπειρα συστήματα χωρίς τη βάση γνώσης, δηλαδή μπορούν να παρομοιαστούν ως περιβλήματα (κελύφη) μιας βάσης γνώσης. Ο παραπάνω ορισμός δεν προϋποθέτει πως τα κελύφη προέρχονται αποκλειστικά από υπάρχοντα έμπειρα συστήματα με αφαίρεση της αρχικής βάσης γνώσης τους. Υπάρχουν κελύφη γενικού σκοπού τα οποία αναπτύσσονται από την αρχή σε κάποια γλώσσα προγραμματισμού, χωρίς να είναι προσανατολισμένα σε κάποια συγκεκριμένη εφαρμογή.

Ένας πιο ουσιαστικός διαχωρισμός των κελυφών είναι σε απλά ή εξελιγμένα και γίνεται βάσει σημαντικών χαρακτηριστικών τους, όπως οι διάφορες μορφές αναπαράστασης της γνώσης και συλλογιστικής που παρέχει στο μηχανικό της γνώσης. Επίσης, τα διάφορα κελύφη διακρίνονται για τις διάφορες ευκολίες διασύνδεσης με το χρήστη και με το περιβάλλον γενικότερα.

Απλά κελύφη

Τα απλά κελύφη εμπείρων συστημάτων προήλθαν από υπάρχοντα έμπειρα συστήματα, με αφαίρεση της βάσης γνώσης τους. Κλασσικότερο παράδειγμα το EMYCIN, το οποίο προήλθε από το έμπειρο σύστημα MYCIN για διάγνωση μικροβιολογικών μολύνσεων του αίματος. Το EMYCIN χρησιμοποιήθηκε για την κατασκευή διαφόρων άλλων εμπείρων συστημάτων, για ιατρικά και μη προβλήματα, όπως για παράδειγμα:

- CLOT (ασθένειες του αίματος)
- PUFF (πνευμονικές ασθένειες)
- HEAD MED (ψυχιατρικές διαγνώσεις)
- LITHIO (γεωλογία)
- TAX-ADVISOR (νομική)
- SACON (δομική ανάλυση μηχανολογικών σχεδίων)

Το EMYCIN είναι γραμμένο σε INTERLISP, μια δημοφιλή διάλεκτο της γλώσσας LISP στη δεκαετία του ’70. Τα γεγονότα στο EMYCIN παριστάνονται σαν τριάδες της μορφής: “*έννοια-παράμετρος-τιμή*”. Κάθε τριάδα συνοδεύεται και από ένα συντελεστή βεβαιότητας. Οι διάφορες έννοιες ιεραρχούνται σε ένα δένδρο (*context tree*) που οργανώνει τις πληροφορίες για ένα περιστατικό.

Ο τρόπος αναπαράστασης της γνώσης είναι οι κανόνες, οι οποίοι ταξινομούνται ανάλογα με τις έννοιες στις οποίες αναφέρονται. Οι κανόνες αποτελούνται από μια συνθήκη και μια ενέργεια. Ο τρόπος εκτέλεσης των κανόνων είναι η ανάστροφη ακολουθία εκτέλεσης, δηλαδή από την ενέργεια προς τη συνθήκη. Όταν επαληθεύεται η συνθήκη τότε προστίθεται στη μνήμη οι τριάδες που εμφανίζονται στην ενέργεια. Η ενέργεια του κανόνα συνοδεύεται και με συντελεστές βεβαιότητας.

Το EMYCIN υποστηρίζει και μετα-κανόνες (*meta-rules*), οι οποίοι χρησιμοποιούνται για να αποτρέψουν την εκτέλεση κάποιων κανόνων ή για να καθορίσουν τη σειρά εκτέλεσης των κανόνων. Οι μετα-κανόνες εξετάζουν τις συνθήκες των κανόνων που μπορούν να εκτελεστούν και καθορίζουν τη σειρά εκτέλεσης τους.

Ένα προφανές μειονέκτημα των απλών κελυφών είναι ότι δεν είναι δυνατόν να είναι κατάλληλα για την επίλυση όλων των προβλημάτων, εφόσον δημιουργήθηκαν από ένα έμπειρο σύστημα το οποίο επιλύει ένα συγκεκριμένο πρόβλημα. Για παράδειγμα, το MYCIN επιτελούσε διάγνωση, συνεπώς το EMYCIN είναι κατάλληλο κυρίως για διαγνωστικά προβλήματα όπου η ανάστροφη συλλογιστική βοηθά στην επικέντρωση της έρευνας γύρω από ζητούμενες δυσλειτουργίες. Παρόλα αυτά είναι δυνατόν να γενικευθεί η χρησιμότητα συγκεκριμένων κελυφών για διάφορα προβλήματα, αφού το ίδιο πρόβλημα μπορεί να αντιμετωπισθεί με πολλούς τρόπους, ενώ μια μεθοδολογία μπορεί να χρησιμοποιηθεί σε πολλά διαφορετικά προβλήματα.

Το ουσιαστικότερο μειονέκτημα των απλών κελυφών είναι η υποστήριξη ενός μόνο είδους αναπαράστασης και δόμησης της γνώσης καθώς και συλλογιστική και αβεβαιότητας. Αυτός είναι και ο λόγος για τον οποίο υποστηρίζουν μικρό εύρος εφαρμογών.

Εξελιγμένα κελύφη

Εξελιγμένα κελύφη ονομάζονται τα εργαλεία της τεχνολογίας της γνώσης τα οποία υποστηρίζουν πολλαπλούς τρόπους αναπαράστασης γνώσης και συλλογιστικής και συνεπώς μεγαλύτερη γενικότητα εφαρμογών. Υπάρχουν δύο κατηγορίες εξελιγμένων κελυφών, ανάλογα με τον τρόπο με τον οποίο αναπτύχθηκαν. Κάποια εξελιγμένα κελύφη, όπως τα PERSONAL CONSULTANT, S1 και M4, δημιουργήθηκαν αρχικά από έμπειρα συστήματα και στη συνέχεια μετατράπηκαν και επεκτάθηκαν κατάλληλα ώστε να υποστηρίζουν περισσότερα είδη αναπαράστασης γνώσης και συλλογιστικής.

Μια άλλη κατηγορία εξελιγμένων κελυφών είναι αυτά που αναπτύχθηκαν από την αρχή σε κάποια γλώσσα προγραμματισμού και δεν προήλθαν από κάποιο έμπειρο σύστημα. Και αυτού του είδους τα κελύφη υποστηρίζουν πολλές διαφορετικές

μεθόδους αναπαράστασης της γνώσης και συλλογιστικής καθώς και εξελιγμένες δυνατότητες διασύνδεσης με το χρήστη. Η κυριότερη διαφορά τους με τα απλά κελύφη είναι ότι επειδή φτιάχνονται από την αρχή, δε δεσμεύονται από τον τρόπο δόμησης και τους περιορισμούς του αρχικού έμπειρου συστήματος. Τα εργαλεία αυτά είναι συνήθως δυσκολότερα στην εκμάθηση από όλα τα προαναφερθέντα γιατί περιέχουν πολλές και ετερογενείς μεταξύ τους προγραμματιστικές έννοιες, πλην όμως δίνουν περισσότερες δυνατότητες στην κατασκευή και τη συντήρηση των εμπείρων συστημάτων. Η ανάμειξη διαφόρων προγραμματιστικών μοντέλων σε αυτά τα εργαλεία γίνεται εντελώς πειραματικά από τους προγραμματιστές, γιατί δεν υπάρχουν καθιερωμένες θεωρίες υβριδικών τρόπων αναπαράστασης της γνώσης.

Οι κυριότεροι τρόποι αναπαράστασης γνώσης που αναμιγνύονται σε αυτά τα συστήματα είναι τα πλαίσια και οι κανόνες. Τα πλαίσια πλεονεκτούν στη δομημένη αναπαράσταση σύνθετων φυσικών αντικειμένων και στο συμπαγή τρόπο χειρισμού τους. Από την άλλη, οι κανόνες πλεονεκτούν στο δηλωτικό τρόπο αναπαράστασης εμπειρικών συσχετίσεων μεταξύ παρατηρηθέντων δεδομένων (συνθηκών) και επαρκούντων δράσεων για την αντιμετώπιση των περιπτώσεων που αντιπροσωπεύουν οι συνθήκες. Ο συνδυασμός των δύο τρόπων αναπαράστασης γνώσης γίνεται ως εξής:

- Οι συνθήκες των κανόνων μπορούν να αναφέρονται σε τιμές των ιδιοτήτων (slots) των πλαισίων.
- Οι δράσεις των κανόνων μπορούν να αλλάζουν τις τιμές των ιδιοτήτων ή να δημιουργούν και να διαγράφουν πλαίσια.

Επίσης η ύπαρξη ορθής και ανάστροφης εκτέλεσης κανόνων ή/και άλλων ευριστικών τρόπων εκτέλεσης, καθώς και η δυνατότητα συνδυασμού των διαφόρων τρόπων εκτέλεσης στο ίδιο πρόγραμμα αποτελεί σημαντικό εργαλείο στα χέρια του προγραμματιστή για να επιβάλλει τον έλεγχο εκτέλεσης που κρίνει απαραίτητο σε κάθε φάση λειτουργίας του προγράμματος.

Μερικά τέτοια εξελιγμένα κελύφη είναι τα ART, KEE, και KNOWLEDGE CRAFT, τα οποία αναπτύχθηκαν σε LISP, τα FLEX και MIKE, που αναπτύχθηκαν σε PROLOG, τα NEXPERT και CLIPS, που αναπτύχθηκαν σε C/C++, κλπ.

2.5 Αναπαράσταση γνώσης

Στην καθημερινή ζωή, η αναπαράσταση της γνώσης για τον κόσμο γίνεται σχετικά εύκολα με απλή περιγραφή σε φυσική γλώσσα. Η φυσική γλώσσα θεωρείται μια από τις καλύτερες μεθόδους αναπαράστασης της γνώσης για τον άνθρωπο. Ωστόσο η φυσική γλώσσα είναι ακατάλληλη για αναπαράσταση γνώσης σε υπολογιστικά συστήματα. Ο κυριότερος λόγος είναι η *πολυπλοκότητα* (ambiguity) της γλώσσας, δηλαδή οι πολλαπλές ερμηνείες που επιδέχεται μια φράση, καθώς επίσης και η ερμηνεία με βάση τα *συμφραζόμενα* (context).

Για τα συστήματα ΤΝ πρέπει να χρησιμοποιηθεί ένας μονοσήμαντος και τυποποιημένος συμβολισμός, ο οποίος, εκτός της δυνατότητας που θα δίνει για ακριβή αναπαράσταση της γνώσης, θα πρέπει να μπορεί να συνδυαστεί κατάλληλα με *μηχανισμούς εξαγωγής συμπερασμάτων* (inference engines) που υλοποιούνται από τη *στρατηγική αναζήτησης* στη γνώση ενός προβλήματος και από τη *συλλογιστική* (reasoning). Η στρατηγική αναζήτησης υλοποιείται με διάφορους τρόπους, οι γνωστότεροι δύο από αυτούς είναι η *συλλογιστική οδηγούμενη από τους στόχους* (goal driven ή top-down) και η *συλλογιστική οδηγούμενη από τα δεδομένα* (data driven ή bottom-up). Η συλλογιστική, δηλαδή ο γενικός τρόπος παραγωγής γνώσης από ήδη υπάρχουσα γνώση, υλοποιείται με τρείς κυρίως μεθόδους, την *παραγωγή* (deduction), την *επαγωγή (induction*) και την *απαγωγή* (abduction).

Σύμφωνα με τον Winston (1990), *αναπαράσταση γνώσης είναι ένα σύνολο συντακτικών και σημασιολογικών παραδοχών, οι οποίες καθιστούν δυνατή τη περιγραφή ενός κόσμου.* Συνεπώς, τα βασικά δομικά στοιχεία που έχει κάθε μέθοδος αναπαράστασης γνώσης είναι :

- το *συντακτικό* (syntax), δηλαδή ο ορισμός των συμβόλων που χρησιμοποιεί και οι κανόνες με τους οποίους τα σύμβολα αυτά μπορούν να συνδυαστούν, και
- τη *σημασιολογική* (semantics), δηλαδή ο καθορισμός των εννοιών που αποδίδονται στα σύμβολα και συνδυασμούς συμβόλων που επιτρέπει το συντακτικό.

Είδη γνώσης

Οι Bar και Feigenbaum (1981) παραθέτουν 4 είδη γνώσης τα οποία κάθε μέθοδος αναπαράστασης γνώσης πρέπει να είναι ικανή να αναπαραστήσει σε συστήματα ΤΝ:

- *Αντικείμενα* (objects): Αναπαράσταση των αντικειμένων ενός κόσμου, της ευρύτερης κατηγορίας ή της κλάσης αντικειμένων στην οποία ανήκουν, καθώς και της σχέσης μεταξύ αυτών των κλάσεων.
- *Γεγονότα* (events): Αναπαράσταση των ενεργειών ή περιστατικών που συμβαίνουν σε έναν κόσμο, της χρονικής ακολουθίας με την οποία συμβαίνουν, καθώς και τις σχέσεις αιτίου-αποτελέσματος.
- *Εκτέλεση (*performance*)*: Αναπαράσταση των δεξιοτήτων για το πώς κάποιος κάνει πράγματα.
- *Μετα*-γνώση (meta-knowledge): Αναπαράσταση της γνώσης για το τι γνωρίζει κάποιος.

2.6 Μέθοδοι αναπαράστασης γνώσης

Η επιλογή ενός τρόπου για την αναπαράσταση της γνώσης ενός συγκεκριμένου προβλήματος είναι σημαντικότατη εργασία, δεδομένου ότι προσδιορίζει το αν θα προσεγγισθεί ικανοποιητικά η συλλογιστική του ειδικού. Πρέπει λοιπόν να επιλεγεί η κατάλληλη μέθοδος για τη συγκεκριμένη γνώση και για το συγκεκριμένο πρόβλημα.

Οι μέθοδοι αναπαράστασης της γνώσης, αν και έχουν μεγάλες διαφορές μεταξύ τους εν τούτοις, έχουν δυο κοινά χαρακτηριστικά:

- Υλοποιούνται με τη βοήθεια μιας από τις υπάρχουσας γλώσσες προγραμματισμού,
- Χρησιμοποιούν τα γεγονότα και τη γνώση που περιέχουν, στη προσπάθεια τους για αναζήτηση λογικών συμπερασμάτων με τη βοήθεια των ειδικών τεχνικών αναζήτησης (search) και ταυτοποίησης (pattern-matching) του μηχανισμού εξαγωγής συμπερασμάτων στη βάση γνώσης του Ε.Σ.

Οι δημοφιλέστερες μέθοδοι αναπαράστασης γνώσης είναι

- Λογική
  - *Προτασιακή λογική (propositional logic)*

- *Κατηγορηματική λογική (predicate logic)*
- *Διαζευκτική μορφή της λογικής (clausal form of logic)*

- Δομημένες αναπαραστάσεις γνώσης
  - *Σημασιολογικά Δίκτυα (semantic networks)*
  - *Πλαίσια (frames)*
  - *Εννοιολογική εξάρτηση (conceptual dependency)*
  - *Σενάρια (scripts)*
- Κανόνες (*if-then rules*)

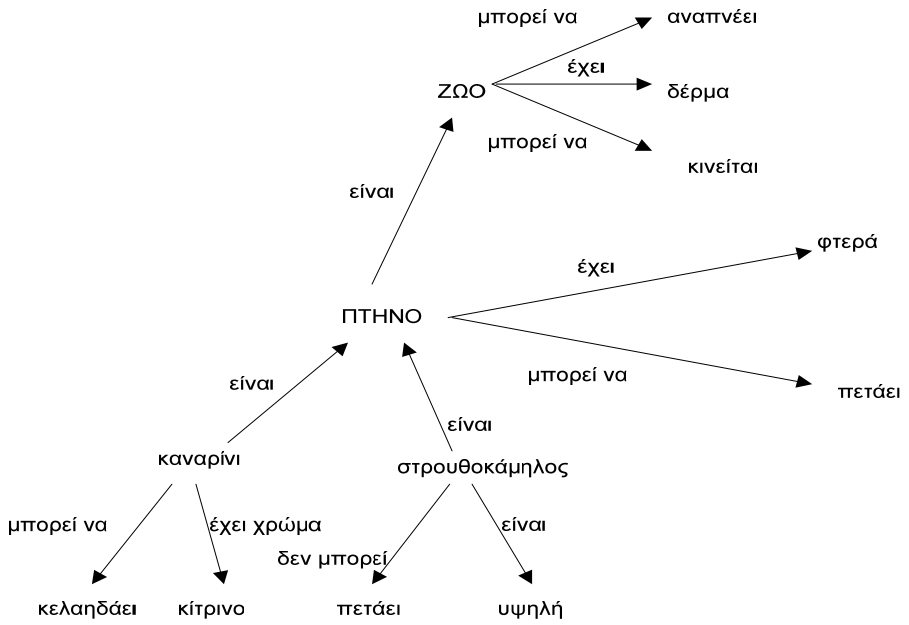
Κάθε μια από τις υπάρχουσες μεθόδους αναπαριστά καλύτερα κάποιες ειδικές μορφές γνώσης. Έτσι, για να μπορέσουμε να έχουμε την καλύτερη δυνατή αναπαράσταση της γνώσης ενός ειδικού προβλήματος, δεν θα πρέπει να μιλάμε πάντοτε για την αποκλειστική εφαρμογή μιας συγκεκριμένης μεθόδου αλλά για την εύρεση του αποτελεσματικότερου συνδυασμού μεθόδων αναπαράστασης.

Κατά τη διαδικασία επιλογής των μεθόδων αυτών θα πρέπει να συνεκτιμηθεί το γεγονός των διαφορετικών μορφών μηχανισμών εξαγωγής συμπερασμάτων που χρησιμοποιείται σε κάθε μορφή αναπαράστασης της γνώσης.

Ακολούθως θα περιγραφούν μερικές από τις ευρύτερα χρησιμοποιούμενες μεθόδους αναπαράστασης της γνώσης.

2.6.1 Σημαντικά - σημασιολογικά δίκτυα (semantic networks)

Με τα σημαντικά δίκτυα γίνεται προσπάθεια να οργανωθεί και να αναπαρασταθεί η αποσπασθείσα γνώση του ειδικού με τρόπο ανάλογο με αυτόν που χρησιμοποιούν οι άνθρωποι για να αναπαριστούν τη γνώση τους. Το σημαντικό δίκτυο αποτελείται από ένα δίκτυο κόμβων (nodes) και συνδέσμων (links) όπως αυτό του παρακάτω σχήματος:



Σχήμα 2.6. Σχηματική παράσταση Σημασιολογικού Δικτύου

Τα Σημασιολογικά Δίκτυα (semantic networks) είναι ένα από τους παλαιότερους τρόπους αναπαράστασης της γνώσης. Τα Σημασιολογικά Δίκτυα είναι ένα σύνολο από αντικείμενα που αντιστοιχούν στους κόμβους του δικτύου. Οι κόμβοι συνδέονται μεταξύ τους με συνδέσμους και έτσι αποτελούν δίκτυο. Με την σύνδεση δύο κόμβων ορίζεται και μια συσχέτιση μεταξύ τους. Στους κόμβους και στους συνδέσμους αναγράφεται το αντικείμενο και τι είδος σύνδεσης αντίστοιχα.

Οι κόμβοι μπορεί να είναι:

α. αντικείμενα που είναι αντιληπτά από τον άνθρωπο με τις αισθήσεις του.

β. αντικείμενα που είναι έννοιες αφηρημένες ή μη.

γ. περιγραφικά στοιχεία αντικειμένων

Οι σχέσεις που συνδέουν μπορεί να είναι της μορφής:

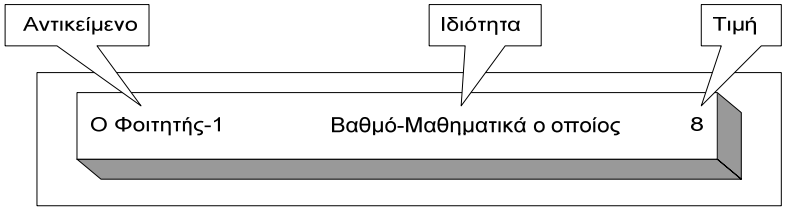
is_a

instance_of

a_kind_of

2.6.2 Αντικείμενο-ιδιότητα-τιμή

Οι τρίπλετες και τα πλαίσια θεωρούνται σαν μερικές περιπτώσεις των σημαντικών δικτύων. Έτσι ο σύνδεσμος Αντικείμενο-Ιδιότητα μπορεί να θεωρηθεί σαν ένας σύνδεσμος 'έχει' ενώ ο σύνδεσμος Ιδιότητα-Τιμή μπορεί να θεωρηθεί σαν ένας σύνδεσμος 'είναι'



Στο κατηγορικό λογισμό, τα νέα συμπεράσματα που προκύπτουν, προστίθενται στα ήδη υπάρχοντα χωρίς να επιτρέπεται η αναθεώρηση της γνώσης (αθροιστική λογική).

Με τη Τιμή καθορίζεται η συγκεκριμένη τιμή μιας ιδιότητας ενός αντικειμένου.

2.6.3 Κανόνες

Τα πρώτα συστήματα αναπτύχθηκαν από τους Newell και Simon το 1972 για να περιγράψουν την ανθρώπινη γνώση. Στα συστήματα αυτά η γνώση αναπαρίσταται με τη βοήθεια κανόνων που έχουν τη μορφή:

IF	συνθήκες(υποθέσεις)
THEN	ενέργειες(συμπεράσματα)

Αυτό σημαίνει ότι όταν οι συνθήκες είναι αληθείς τότε συνεπάγεται ότι αληθεύουν οι ενέργειες και αντιστρόφως. Παρομοίως όταν οι συνθήκες είναι ψευδείς συνεπάγεται ότι και οι ενέργειες είναι ψευδείς και αντιστρόφως. Τα χαρακτηριστικά των κανόνων είναι (Curry and Moutinho, 1991):

- Η σύνταξη των κανόνων πρέπει να περιέχει περιορισμένο κείμενο με σαφή έννοια και να είναι κατανοητό σε κάθε στάδιο εργασιών (π.χ. στάδια παροχής συμβουλών και αιτιολόγησης της συλλογιστικής του).
- Οι κανόνες μπορούν να παρίστανται με σύνθετη δομή ή με διασυνδεόμενες ομάδες κανόνων.
- Για το χειρισμό τους και την κατάλληλη επεξεργασία τους, απαιτείται η χρήση ειδικών γλωσσών, όπως Lisp. Prolog, κ.λπ., διαφόρων shells ή ειδικά κατασκευασμένου λογισμικού για το χειρισμό των διαδικασιών 'λογικής', του συστήματος.
- Η γνώση που χρησιμοποιείται εχει ποιοτικές διαστάσεις.
- Παρέχει τη δυνατότητα στα συστήματα να αποκτούν (αντλούν) γνώση (automated knowledge acquisition) από παραδείγματα ή εφαρμογές (αλγόριθμος ID3 του Quinlan, 1986).

Οι κανόνες χρησιμοποιούνται για τη περιγραφή ευρετικών διαδικασιών επίλυσης προβλημάτων. Ο ορισμός των συνθηκών των κανόνων γίνεται με μεγάλη σαφήνεια με αποτέλεσμα να μειώνονται οι αλληλεπιδράσεις μεταξύ των. Κάθε κανόνας περιγράφει μια συγκεκριμένη περίπτωση γνώσης με αποτέλεσμα την ανεξαρτησία και τη καθαρότητα της αναπαριστώμενης γνώσης. Οι κανόνες χρησιμοποιούνται τόσο για την αναπαράσταση της δηλωτικής (declarative) όσο και της διαδικαστικής (procedural) γνώσης.

Κατά την ανάπτυξη ενός Ε.Σ. δημιουργούνται δύο είδη κανόνων ανάλογα με το είδος της γνώσης που αναπαριστούν. Οι δηλωτικοί κανόνες γνώσης είναι αυτοί που περιέχουν τη γνώση για το πρόβλημα, ενώ οι διαδικαστικοί κανόνες περιέχουν τη γνώσω για τον τρόπο επίλυσης του. Οι πρώτοι αποτελούν τη βάση γνώσης ενώ οι δεύτεροι περιέχονται στο μηχανισμό εξαγωγής συμπερασμάτων. Υπάρχει άλλο ένα είδος κανόνων, οι μετα-κανόνες (metarules) οι οποίοι περιέχουν γνώση για το χειρισμό τόσο των άλλων δύο τύπων κανόνων όσο και για το χειρισμό των ίδιων. Οι μετα-κανόνες περιέχονται και αυτοί στο μηχανισμό εξαγωγής συμπερασμάτων.

Στη συνέχεια δίνονται τα σημαντικότερα πλεονεκτήματα και μειονεκτήματα των κανόνων παραγωγής.

Πλεονεκτήματα:

- Έχουν 'φυσική' μορφή απεικόνισης της γνώσης, απλή σύνταξη και είναι εύκολη η κατανόηση της γνώσης που αναπαριστούν.
- Ανεξαρτησία κανόνων.
- Παρέχουν ευκολίες μεταβολών.
- Η ανάπτυξη και η συμπλήρωση της βάσης γνώσης μπορεί να γίνει σταδιακά.
- Η διαδικασία εξαγωγής συμπερασμάτων και η παροχή επεξηγήσεων είναι σχετικά εύκολη.

Μειονεκτήματα:

- Δυσκολίες στην αναπαράσταση γενικής γνώσης όσον αφορά έννοιες και αντικείμενα.
- Δυσκολία στην αναπαράσταση σύνθετης γνώσης όσον αφορά έννοιες και αντικείμενα.
- Δυσκολία κατανόησης του καθολικού αλγορίθμου.
- Περιορισμοί στην αναζήτηση λύσης λόγω του μεγάλου αριθμού κανόνων.

2.6.4 Πλαίσια

Η τεχνική της αναπαράστασης της γνώσης με τη μέθοδο των πλαισίων (frames) θυμίζει λίγο τη δομή μιας λογικής εγγραφής (record) και αποτελεί την αντικειμενοστραφή εκδοχή της αναπαράστασης της γνώσης. Η μέθοδος αυτή προτάθηκε από τον Minsky (1975)και ο βασικός της στόχος είναι η ομαδοποίηση των χαρακτηριστικών ενός αντικειμένου.

Ένα πλαίσιο περιέχει πληροφορίες που συνοδεύουν ένα αντικείμενο. Κάθε πληροφορία που συνοδεύει το αντικείμενο τοποθετείται σε μια υποδοχή (slot). Κάθε υποδοχή περιέχει τιμές ή δείκτες προς άλλα πλαίσια ή κανόνες ή ακόμη και διαδικασίες από όπου αντλούνται οι τιμές που αντιστοιχίζονται στην πληροφορία που σχετίζεται με το αντικείμενο.

Με τα πλαίσια μπορούμε να έχουμε μια περισσότερο ισχυρή αναπαράσταση από αυτή με κανόνες και με τριάδα αντικείμενο-ιδιότητα-τιμή.

Με τα πλαίσια η προσέγγιση και αναπαράσταση της γνώσης γίνεται τόσο με διαδικαστική μορφή (procedural) όσο και με δηλωτική μορφή (declarative).

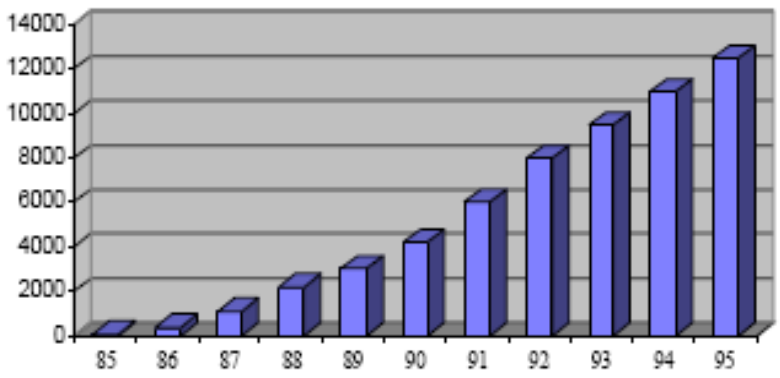
Frame Name	**Slot**	**Value**
Αρκούδα	*Είναι*	*Θηλαστικό*
	Ευφυΐα	*Υψηλή*
	Χρώμα	*ΑΝ είναι πολική* *Λευκό* *Αλλιώς* *Καφέ*
	Βάρος	*Διαδικασία προσδ. βάρους με (ηλικία, εποχή, γεωγραφική περιοχή)*

Σχήμα 2.7. Μορφή Πλαισίου

2.7 Ανάπτυξη έμπειρων συστημάτων

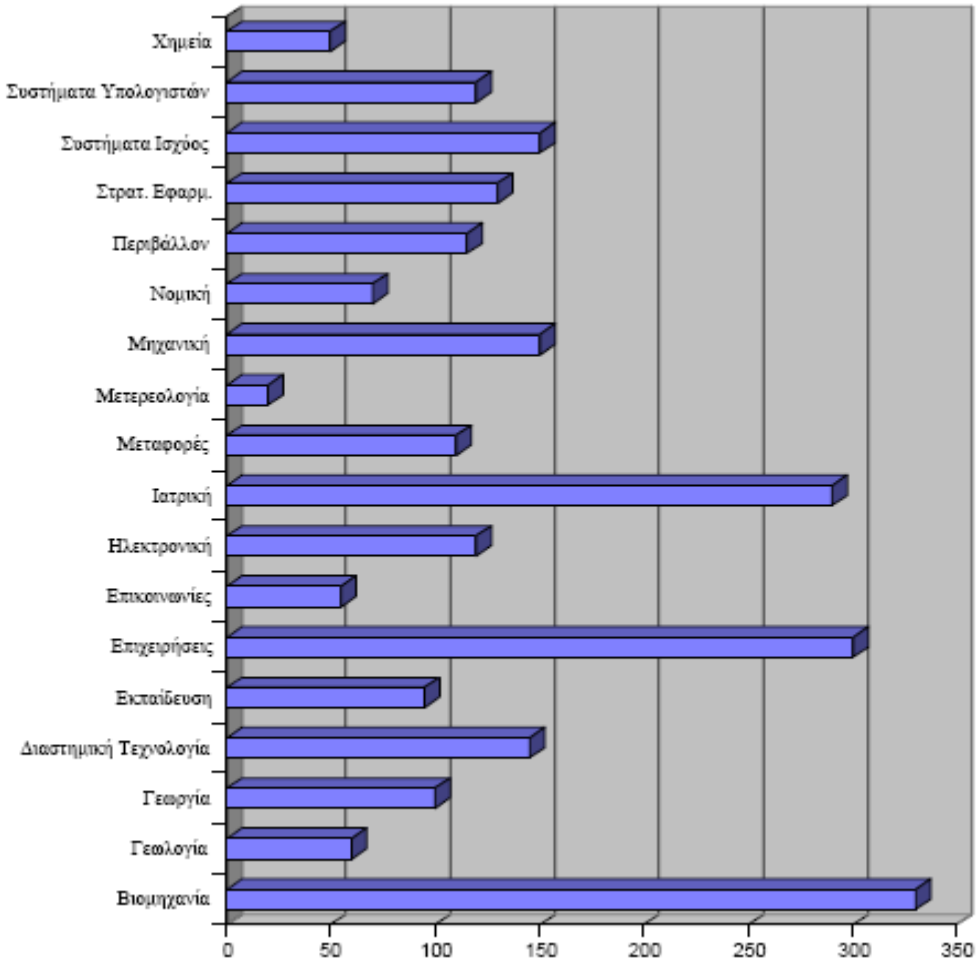
Ενδιαφέρον παρουσιάζει ο ρυθμός ανάπτυξης των εφαρμογών εμπείρων συστημάτων. Έτσι, κατά τη διάρκεια του ’70 που οι ερευνητές είχαν εστιάσει την προσοχή τους στην ανάπτυξη ‘έξυπνων προγραμματιστικών τεχνικών’ (intelligent programming techniques), δημιουργήθηκαν ελάχιστα συστήματα, ενώ αντίθετα κατά τη δεκαετία του ΄80, που υπήρξε καλύτερη κατανόηση, η κατάσταση άλλαξε ραγδαία. Ο εντυπωσιακός ρυθμός ανάπτυξης (Walker 1996), των έμπειρων συστημάτων μπορεί να θεωρηθεί σαν ένας δείκτης αποδοχής τους από τη βιομηχανία, τις επιχειρήσεις και ερευνητικά κέντρα (σχήμα 2.7).

Τα Ε.Σ. έχουν ήδη εφαρμοστεί σύμφωνα με τους Jayaraman και Srivastava (1996), με αρκετή επιτυχία στη παραγωγή και την διοίκηση των επιχειρήσεων.



Σχήμα 2.8. Ο ρυθμός ανάπτυξης των έμπειρων συστημάτων (*Πηγή: Walker, 1996*)

Στο επόμενο σχήμα 2.8 παρουσιάζεται ο αριθμός των συστημάτων σε κάθε μια από περιοχές ανάπτυξης εφαρμογών των έμπειρων συστημάτων (Walker, 1996). Το 1986 στα πλαίσια μιας έρευνας του Waterman, έδειχνε ότι η πλειοψηφία (30%) των εφαρμογών ήταν στο πεδίο της ιατρικής. Το διάγραμμα δείχνει ότι αυτό το πεδίο παραμένει ελκυστικό, μέχρι και σήμερα, για όσους αναπτύσσουν έμπειρα συστήματα. Ακόμη πιο αποκαλυπτική είναι η δραστηριότητα στους κλάδους της βιομηχανίας και των επιχειρήσεων τα τελευταία χρόνια. Σύμφωνα με την έρευνα του Waterman περίπου 10% των εφαρμογών αφορούσαν τους κλάδους αυτούς το 1986. Σήμερα αυτοί οι κλάδοι περιλαμβάνουν το 60% των εφαρμογών. Επίσης το διάγραμμα αποδεικνύει καθαρά ότι ο τομέας των έμπειρων συστημάτων έχει ωριμάσει και έχει γίνει αποδεκτός από τους τομείς της παραγωγής.



Σχήμα 2.9. Έμπειρα Συστήματα στις διάφορες περιοχές εφαρμογών (*Πηγή: Walker, 1996*).

Στο χώρο της συντήρησης αεροσκαφών και ειδικότερα στο τομέα της διάγνωσης και επιδιόρθωσης βλαβών σε αεροσκάφη, αν και εξαιρετικά εξειδικευμένοι χώροι, κάποιες προσπάθειες ανάπτυξης έμπειρων συστημάτων έχουν υλοποιηθεί.

Οι Goker και Kuru (1990) του πανεπιστημίου Bogazici, ανέπτυξαν ένα πρόγραμμα διάγνωσης και επιδιόρθωσης βλαβών σε αεροσκάφη για λογαριασμό αεροπορικής εταιρείας. Το πρόγραμμά τους βασίζεται στο εγχειρίδιο επιδιόρθωσης βλαβών της κατασκευάστριας εταιρείας, σε αρχεία ιστορικού συντήρησης συγκεκριμένου τύπου αεροσκάφους αλλά και σε εγχειρίδια και αρχεία που κρατά η κάθε αεροπορική εταιρεία στη βάση δεδομένων τους.

Το υλοποιημένο έμπειρο σύστημα αποτελείται όπως και όλα τα Ε.Σ., από τη βάση γνώσης, τον μηχανισμό εξαγωγής συμπερασμάτων και την διασύνδεση με τον χρήστη.

Η βάση γνώσης αποτελείται από τέσσερα ξεχωριστά είδη γνώσης: δένδρα επιδιόρθωσης βλαβών, δεδομένα αντικατάστασης εξαρτημάτων, δεδομένα ταξινόμησης αεροσκαφών και δεδομένα ταξινόμησης εξαρτημάτων.

2.8 Σύγκριση έμπειρων συστημάτων

Στους πίνακες που ακολουθούν έγινε μια προσπάθεια να συγκεντρωθούν τα περισσότερα κελύφη εμπείρων συστημάτων, που έχουν αναπτυχθεί και εξακολουθούν να υφίστανται έως τις μέρες μας. Κυριότερες μέθοδοι καταγραφής τους είναι με αναζήτηση στο διαδίκτυο και συγκεκριμένα στην ιστοσελίδα http://www.kbsc.com/rulebase.html , όπου περιέχεται μεγάλος αριθμός Ε.Σ. αλλά και με αναζήτηση τόσο στην εγχώρια (Τεχνητή νοημοσύνη, Α' έκδοση) όσο και στην ξένη βιβλιογραφία (Jackson, 1998;Durkin, 1994).

Η σύγκριση θα γίνει βάση της μεθόδου που χρησιμοποιούν για να αναπαραστήσουν την γνώση και του μηχανισμού εξαγωγής συμπερασμάτων. Όπως θα δούμε και στους πίνακες, η γνώση μπορεί να αναπαρασταθεί πέρα από τις γνωστές μεθόδους και με άλλες πιο εξελιγμένες. Κυρίαρχη μέθοδος και πιο διαδεδομένη πάραυτα δεν παύει να αποτελεί η αναπαράσταση με κανόνες και η αναπαράσταση με πλαίσια.

ΚΕΛΥΦΟΣ ΕΜΠΕΙΡΟΥ ΣΥΣΤΗΜΑΤΟΣ	ΑΝΑΠΑΡΑΣΤΑΣΗ ΓΝΩΣΗΣ						ΜΗΧΑΝΙΣΜΟΣ ΕΞΑΓΩΓΗΣ ΣΥΜΠΕΡΑΣΜΑΤΩΝ					
	rules	table form	frame based	decision tree	optional frame extension	object oriented	forward chaining	backward chaining	induction	set logic	inexact logic	mixed forward & backward chaining
1st-CLASS		✓							✓			
ACE	✓						✓	✓				
ADS	✓		✓				✓	✓				
Advisor-2	✓						✓	✓				
AIM	✓						✓	✓				
Arity EXPERT			✓				✓	✓				
ART	✓		✓				✓					
ART-IM	✓		✓				✓					
Auto-Intelligence	✓							✓				
CLIPS	✓						✓					
Crystal	✓						✓	✓	✓			
Decision Expert		✓							✓			
ESP Advisor	✓						✓	✓				
ESP Frame Engine			✓				✓	✓				
Expert Controller			✓				✓	✓		✓		
EXSYS EL	✓				✓		✓	✓				

EXSYS Professional	✓				✓		✓	✓				
Flex	✓		✓				✓	✓				
Goldworks	✓		✓			✓	✓	✓				
Guru First Step	✓						✓	✓	✓			✓
Guru	✓						✓	✓	✓			✓
Intelligence Compiler							✓	✓			✓	
Kappa_PC	✓		✓			✓	✓	✓				
KBMS	✓		✓				✓	✓				
KDS	✓		✓				✓	✓	✓			
KEE	✓					✓	✓	✓				
KES			✓				✓	✓				
Knowledge Craft			✓				✓	✓				
Knowledger Maker		✓							✓			
KnowldgerPro	✓						✓	✓				
LASER			✓				✓	✓				
Level5	✓							✓				
Level5 Object	✓		✓				✓	✓				
LogicTree		✓							✓			
M.1	✓											

M.4	✓						✓	✓				
MacSMARTS		✓							✓			
MacSMARTS Professional		✓							✓			
Nexpert Object	✓		✓				✓	✓				
OPS83	✓						✓					
Personal Consultant				✓				✓				
Personal Consultant Easy	✓						✓	✓				
pLogic		✓							✓			
ProKappa	✓		✓			✓	✓	✓				
ReMind	✓		✓			✓	✓	✓				
Superexpert		✓							✓			
TIMM		✓							✓			
VP-Expert	✓						✓	✓	✓			

Πίνακας 2.1: Συγκριτικός πίνακας έμπειρων συστημάτων

ΚΕΦΑΛΑΙΟ 3

Το Σύστημα Κανόνων CLIPS

3.1 Εισαγωγή

Η βιβλιογραφία που χρησιμοποιήθηκε για τη συγγραφή του κεφαλαίου αυτού προέρχεται στο μεγαλύτερο μέρος της από το βιβλίο 'Τεχνητή Νοημοσύνη, εκδόσεις Γαρταγάνη, Θεσσαλονίκη (Βλαχάβας Ι., Κεφαλάς Π., Βασιλειάδης Ν., Ρεφανίδης Ι., Κόκορας Φ., Σακελλαρίου Η., 2002) και από το βιβλίο 'EXPERT SYSTEMS: Principles and Programming' (Giarratano, Riley, 2005).

Το CLIPS (C Language Integrated Production System) είναι ένα περιβάλλον που προσφέρει δυνατότητες για προγραμματισμό με κανόνες, αντικείμενα και συναρτήσεις. Αναπτύχθηκε από τη NASA με σκοπό να αποτελέσει μια χαμηλού κόστους πλατφόρμα ανάπτυξης έμπειρων συστημάτων, αντικαθιστώντας τα ήδη υπάρχοντα συστήματα τα οποία βασίζονταν στη γλώσσα LISP.

Η ανάπτυξη του CLIPS άρχισε το 1984 και η πρώτη έκδοση ήταν έτοιμη την άνοιξη του 1985. για την ανάπτυξη του χρησιμοποιήθηκε η γλώσσα C, έτσι ώστε το τελικό προϊόν να είναι μεταφέρσιμο σε όλα τα γνωστά λειτουργικά συστήματα (DOS, WINDOWS, UNIX, VMS) και να είναι εύκολα επεκτάσιμο. Η πρώτη πλήρης έκδοση του CLIPS ήταν η 3.0, η οποία ήταν έτοιμη το καλοκαίρι του 1986. Την έκδοση αυτή ακολούθησαν οι εκδόσεις 4.0, 4.1 και 4.2, οι οποίες περιείχαν σημαντικές βελτιώσεις στον αρχικό κώδικα, χωρίς να προσθέτουν όμως σημαντικά στοιχεία. Η έκδοση 5.0, που παρουσιάστηκε το 1991, επέκτεινε το αρχικό σύστημα σημαντικά, υποστηρίζοντας εκτός από προγραμματισμό με κανόνες, διαδικαστικό (Procedural Programming) και αντικειμενοστραφή προγραμματισμό (Object Oriented Programming). Η αντικειμενοστραφής γλώσσα προγραμματισμού που παρέχεται με το CLIPS ονομάζονται COOL (CLIPS Object-Oriented Language). Η έκδοση 5.1 υποστήριζε καινούρια διεπαφή χρήστη (user interface) για τα νέα λειτουργικά

συστήματα και εμφανίστηκε το 1991. Η τελευταία έκδοση είναι η 6.3 και παρουσιάστηκε το καλοκαίρι του 2007.

3.2 Δομή του CLIPS

Το CLIPS είναι ένα διερμηνευόμενο σύστημα παραγωγής (interpreted production system), το οποίο υποστηρίζει την ορθή ακολουθία εκτέλεσης (forward chaining). Τα κύρια μέρη του συστήματος είναι:

- Η λίστα γεγονότων (fact list), η οποία αντιστοιχεί στη μνήμη εργασίας (working memory) των συστημάτων παραγωγής. Όπως δηλώνει και το όνομα της, είναι ο χώρος στον οποίο αποθηκεύονται τα γεγονότα (facts), τόσο εκείνα που ορίζονται κατά την εκκίνηση του συστήματος, όσο και εκείνα που δημιουργούνται κατά την εκτέλεση του.
- Η βάση κανόνων (rule base / knowledge base), όπου περιέχονται οι κανόνες. Αν και οι κανόνες μπορούν να ορισθούν μέσα από το περιβάλλον του συστήματος, συνήθως είναι αποθηκευμένοι σε κάποιο αρχείο απλού κειμένου (text document), το οποίο φορτώνεται στο σύστημα.
- Ο μηχανισμός εξαγωγής συμπερασμάτων (inference engine), ο οποίος ελέγχει τη λειτουργία ολόκληρου του συστήματος. Ο μηχανισμός αυτός προσφέρει ένα πλήθος από στρατηγικές επίλυσης συγκρούσεων (conflict resolution strategies) για την επιλογή του κανόνα που θα πυροδοτηθεί. Το σύνολο των υποψηφίων κανόνων για πυροδότηση αποτελεί το σύνολο σύγκρουσης, ή την ατζέντα (conflict set ή agenda) του συστήματος.

Ένα πρόγραμμα στο CLIPS είναι ένα σύνολο από κανόνες και γεγονότα και η εκτέλεση του συνίσταται σε μια ακολουθία από πυροδοτήσεις κανόνων, των οποίων οι συνθήκες ικανοποιούνται. Η ικανοποίηση των συνθηκών γίνεται μέσω ταυτοποίησης τους με τα γεγονότα που υπάρχουν στη λίστα γεγονότων. Η εκτέλεση τερματίζεται όταν δεν υπάρχουν άλλοι κανόνες προς πυροδότηση ή όταν κληθεί συγκεκριμένη εντολή τερματισμού. Ο κύκλος λειτουργίας του συστήματος είναι ο τυπικός κύκλος λειτουργίας ενός συστήματος παραγωγής:

1. Εύρεση όλων των κανόνων των οποίων οι συνθήκες ικανοποιούνται και προσθήκη τους στην ατζέντα (agenda – conflict set).
2. Αν η ατζέντα είναι κενή τότε η εκτέλεση τερματίζεται.
3. Επιλογή ενός κανόνα με βάση τη στρατηγική επίλυσης ανταγωνισμού (conflict resolution) και εκτέλεση του
4. Επιστροφή στο βήμα 1, εκτός αν υπάρχει εντολή τερματισμού (halt).

3.3 Σύνταξη του CLIPS

Η σύνταξη του CLIPS είναι απλή και θυμίζει εκείνη της γλώσσας προγραμματισμού LISP. Στην ενότητα που ακολουθεί παρουσιάζεται η σύνταξη της γλώσσας που προσφέρει το σύστημα, ξεκινώντας από τα βασικά δομικά στοιχεία της και ακολουθεί η παρουσίαση των μεταβλητών, των γεγονότων και των κανόνων. Θα πρέπει να σημειωθεί ότι στο CLIPS υπάρχει διαχωρισμός κεφαλαίων και πεζών χαρακτήρων (case-sensitive).

3.3.1 Βασικά δομικά στοιχεία

Τα βασικά δομικά στοιχεία του CLIPS είναι τα ακόλουθα:

Σύμβολα (symbols): Σύμβολο μπορεί να είναι οποιαδήποτε ακολουθία χαρακτήρων η οποία ξεκινά με οποιονδήποτε χαρακτήρα εκτός από τους ακόλουθους :

< | \& ( ) $? + -

και δεν περιέχει τους χαρακτήρες:

< | \& ( ) ;

Για παράδειγμα: cat-and-dog, airway_32, memory_relocation_fail.

Αλφαριθμητικά (strings): Όπως συνηθίζεται, ένα αλφαριθμητικό ξεκινά και τελειώνει με διπλά εισαγωγικά.

Αριθμοί (numbers): Το CLIPS υποστηρίζει τις ακόλουθες αναπαραστάσεις αριθμών (η σημειολογία θεωρείται γνωστή. Καθώς είναι κοινή σε πολλές γλώσσες προγραμματισμού):

11, 54, 97, -32, +53, 21.45, 78.32, 2e10, 2E10

Σχόλια: Το CLIPS θεωρεί σχόλιο ότι ακολουθεί το χαρακτήρα “;” μέχρι το τέλος μιας γραμμής. Επίσης, ορισμένες εντολές περιέχουν στη δομή της σύνταξης τους ειδικό όρισμα για εισαγωγή σχολίων, τα οποία συνήθως περικλείονται σε διπλά εισαγωγικά.

3.3.2 Μεταβλητές

Οι μεταβλητές στο CLIPS είναι σύμβολα, τα οποία ξεκινούν απαραίτητα με τους χαρακτήρες ? ή $? και με τον περιορισμό ο πρώτος χαρακτήρας που ακολουθεί να μην είναι αριθμός. Υπάρχουν δύο είδη μεταβλητών, οι μονότιμες και οι πολλαπλοί τιμών.

- Οι μονότιμες (singlevalue) μεταβλητές μπορούν να πάρουν σαν τιμή μόνο ένα σύμβολο, αριθμό ή αλφαριθμητικό, όπως άλλωστε δηλώνει και το όνομα τους. Αυτές ξεκινούν με το χαρακτήρα ?. Για παράδειγμα, το σύμβολο ?var1 είναι μια μονότιμη μεταβλητή με παραδείγματα επιτρεπτών τιμών τα symbol1, flight530, Friday_meeting, 45, 867, κλπ.
- Οι μεταβλητές πολλαπλών τιμών (multivalue) παίρνουν σαν τιμές ένα ή περισσότερα σύμβολα. Οι μεταβλητές αυτές ξεκινούν με τους χαρακτήρες &?.

Οι μεταβλητές εμφανίζονται τόσο στις συνθήκες όσο και στις ενέργειες ενός κανόνα, αλλά παίρνουν τιμές κυρίως στις συνθήκες των κανόνων μέσω της διαδικασίας ταυτοποίησης. Ανάθεση τιμής σε μεταβλητή στις ενέργειες ενός κανόνα είναι βέβαια δυνατή με τη χρήση κατάλληλης συνάρτησης, αλλά καλό είναι να αποφεύγεται. Τέλος θα πρέπει να τονισθεί ότι η εμβέλεια των μεταβλητών περιορίζεται στον κανόνα που αυτές εμφανίζονται.

3.3.3 Γεγονότα

Τα γεγονότα αποτελούν την πληροφορία την οποία το σύστημα “γνωρίζει” και στη οποία βασίζεται ώστε να εξάγει συμπεράσματα. Τα γεγονότα είναι λίστες από σύμβολα τα οποία περικλείονται σε παρενθέσεις όπως για παράδειγμα:

(person Akis Saltas)
(day Friday)
(flight_time_arrival 19:55)
(list of days (mon tue wen thu fri sat) list of months (Jan Feb Mar))

Όπως προαναφέρθηκε τα γεγονότα αποθηκεύονται στη λίστα γεγονότων και είναι δυνατό να εμφανιστούν στο χρήστη με δύο τρόπους:

- Με την εντολή (facts) από τη γραμμή εντολών του CLIPS.
- Μέσω του αντίστοιχου παράθυρου (facts window).

Τα γεγονότα που υπάρχουν στη μνήμη εργασίας, δεν είναι στατικά και μπορούν να εισαχθούν ή να διαγραφούν από αυτή κατά την εκτέλεση του προγράμματος.

Τέλος, κάθε γεγονός το οποίο είναι αποθηκευμένο στη λίστα γεγονότων έχει ένα χαρακτηριστικό αριθμό (fact index), ο οποίος ορίζεται αυτόματα από το σύστημα κατά την καταχώρηση του γεγονότος στη λίστα. Ο αριθμός αυτός χαρακτηρίζει μοναδικά το γεγονός και χρησιμεύει τόσο στη διαγραφή των γεγονότων όσο και στο να γνωρίζει το σύστημα και ο χρήστης ποια γεγονότα ενεργοποίησαν έναν κανόνα.

Εισαγωγή και διαγραφή γεγονότων

Δύο είναι οι βασικοί τρόποι εισαγωγής γεγονότων στη λίστα γεγονότων: με τη εντολή assert και την εντολή deffacts. Η διαγραφή ενός γεγονότος από τη λίστα γίνεται με την εντολή retract. Παρακάτω δίνεται η σύνταξη των εντολών αυτών:

Σύνταξη: (assert <fact>)

Η εντολή assert εισάγει ένα γεγονός στη λίστα γεγονότων. Χρησιμοποιείται συνήθως στο δεξιό μέρος των κανόνων.

Σύνταξη: (deffacts <fact-set-name>
"comment"
(fact 1)
(fact 2)

……
(fact n))

Η εντολή χρησιμοποιείται για τη μαζική εισαγωγή γεγονότων κατά την εκκίνηση ενός προγράμματος στο CLIPS. Για να εισαχθούν τα γεγονότα που ορίζονται με την παραπάνω εντολή στη λίστα γεγονότων θα πρέπει όχι μόνο να φορτωθεί το αντίστοιχο αρχείο αλλά και να εκτελεστεί η εντολή (reset), η οποία παρουσιάζεται σε επόμενη παράγραφο.

Σύνταξη: (retract <fact-index>)

Η διαγραφή ενός γεγονότος από τη λίστα γίνεται με την εντολή retract. Το όρισμα fact-index είναι ο μοναδικός αριθμός που αντιστοιχεί το CLIPS στο συγκεκριμένο γεγονός. Η ίδια εντολή μπορεί να χρησιμοποιηθεί τόσο από τη γραμμή εντολών του CLIPS όσο και στις ενέργειες των κανόνων.

3.3.4 Κανόνες

Οι κανόνες στο CLIPS, όπως και σε όλα τα συστήματα παραγωγής, αποτελούνται από δύο μέρη, τις συνθήκες και τις ενέργειες, είναι δηλαδή της μορφής:

if (συνθήκες) *then* (ενέργειες)

Οι συνθήκες είναι τις περισσότερες φορές ένα σύνολο από γεγονότα τα οποία θα πρέπει να υπάρχουν στη λίστα γεγονότων για να ικανοποιείται ο κανόνας. Οι συνθήκες ενός κανόνα μπορούν να περιέχουν μεταβλητές και έτσι να ταυτοποιούνται με περισσότερα του ενός γεγονότα της λίστας γεγονότων. Αυτό επιτρέπει να ικανοποιείται ο κανόνας με περισσότερους από έναν τρόπους, έχοντας κάθε φορά διαφορετικές αναθέσεις τιμών στις μεταβλητές του. Στην περίπτωση αυτή στην ατζέντα εισάγεται ο κανόνας τόσες φορές, όσοι είναι οι διαφορετικοί τρόποι με τους οποίους ικανοποιείται και φυσικά με διαφορετικές αναθέσεις στις αντίστοιχες μεταβλητές.

Στις ενέργειες του κανόνα περιγράφεται το τι θα λάβει χώρα κατά την πυροδότηση του (firing). Οι ενέργειες μπορούν να περιλαμβάνουν οποιαδήποτε συνάρτηση του CLIPS, όπως για παράδειγμα την εισαγωγή ή διαγραφή ενός γεγονότος από τη λίστα γεγονότων, την εκτύπωση αποτελεσμάτων, υπολογισμό τιμών, κλπ. Οι κανόνες ορίζονται μέσω της συνάρτησης defrule του CLIPS.

Σύνταξη: (defrule <rule-name>
"<comment>"
(condition 1)
(condition 2)
…….
(condition n)
=>
(command 1)
(command 2)
……..
(command n)

Το σύμβολο "=>" διαχωρίζει τις συνθήκες από τις ενέργειες του κανόνα. Το όνομα του κανόνα πρέπει να είναι μοναδικό. Αν και είναι δυνατό εντολές της παραπάνω μορφής να εισαχθούν από τη γραμμή εντολών του συστήματος, συνήθως αποθηκεύονται σε κάποιο αρχείο κειμένου (text document) και φορτώνονται στο CLIPS.

3.4 Βασικές Εντολές Περιβάλλοντος

Στα επόμενα δίνονται μερικές βασικές εντολές του περιβάλλοντος του CLIPS, οι οποίες είναι απαραίτητες για την εκτέλεση ενός προγράμματος.

Σύνταξη: (load "<File_Name>")

Η εντολή αυτή φορτώνει στο σύστημα ένα αρχείο με ορισμούς κανόνων, γεγονότων και συναρτήσεων. Στο παραθυρικό περιβάλλον η ίδια λειτουργία γίνεται μέσο του File→Load.

Σύνταξη: (reset)

Η εντολή αυτή διαγράφει από τη λίστα γεγονότων τα γεγονότα που έχουν πιθανόν δημιουργηθεί κατά την εκτέλεση του προγράμματος και επαναφέρει σε αυτή τα αρχικά γεγονότα που ορίζονται στο αρχείο που έχουμε φορτώσει στο σύστημα. Επίσης εισάγει στη λίστα γεγονότων και το γεγονός (initial-fact). Θα πρέπει να σημειωθεί ότι ακόμη και αν φορτωθούν γεγονότα από αρχείο, αυτά δεν εισάγονται αυτόματα στη λίστα γεγονότων εάν δεν προηγηθεί η εντολή (reset).

Σύνταξη: (run)

Εκκινεί την εκτέλεση των κανόνων που έχουν φορτωθεί στη μνήμη. Η εντολή μπορεί να συνταχθεί και σαν (run N), όπου N είναι ο αριθμός των κύκλων λειτουργίας που θα εκτελεστούν. Μετά την εκτέλεση N κύκλων λειτουργίας το σύστημα θα σταματήσει. Η τελευταία αυτή δυνατότητα χρησιμοποιείται κυρίως κατά τη διαδικασία αποσφαλμάτωσης.

Σύνταξη: (clear)

Η εντολή αυτή καθαρίζει τελείως το περιβάλλον της γλώσσας, διαγράφοντας όλα τα γεγονότα από τη λίστα γεγονότων και όλους τους κανόνες που τυχόν έχουν φορτωθεί στο σύστημα.

3.5 Συναρτήσεις

Οι συναρτήσεις στο CLIPS έχουν παρόμοια μορφή με εκείνη των συναρτήσεων στην LISP. Η συνάρτηση είναι συντακτικά μια δομή που περικλείεται μέσα σε παρενθέσεις, όπου το πρώτο στοιχείο της παρένθεσης είναι το όνομα της συνάρτησης και τα επόμενα στοιχεία τα ορίσματα της. Το διαχωριστικό σύμβολο μεταξύ των ορισμάτων είναι το κενό. Η κλήση μιας συνάρτησης γίνεται με την δήλωση:

(<όνομα συνάρτησης> όρισμα1 όρισμα2 όρισμαN)

Για παράδειγμα με την έκφραση (+ 2 3) θα κληθεί η συνάρτηση + με ορίσματα 2 και 3. Το αποτέλεσμα της συνάρτησης θα επιστραφεί στο σημείο που αυτή εμφανίζεται. Στα επόμενα δίνονται βασικές συναρτήσεις της γλώσσας, όπως οι αριθμητικές συναρτήσεις, οι συναρτήσεις ελέγχου και οι συναρτήσεις για τη σύγκριση αριθμών.

Βασικές αριθμητικές συναρτήσεις

Το CLIPS υποστηρίζει όλες τις βασικές αριθμητικές συναρτήσεις. Θα πρέπει να σημειωθεί ότι οι συναρτήσεις δέχονται περισσότερα από δύο ορίσματα.

Συνάρτηση	**Σύνταξη**
Πρόσθεση αριθμών	(+ <ορίσματα>)
Αφαίρεση αριθμών	(- <ορίσματα>)
Πολλαπλασιασμός αριθμών	(* <ορίσματα>)
Διαίρεση αριθμών	(/ <ορίσματα>)

Πίνακας 3.1.Βασικές αριθμητικές συναρτήσεις.

Σύγκριση αριθμών

Οι συναρτήσεις σύγκρισης αριθμών στο CLIPS ακολουθούν και αυτές τη σύνταξη των αντίστοιχων εντολών στη LISP. Και στην περίπτωση αυτή οι συναρτήσεις δέχονται περισσότερα από δύο ορίσματα. Ο πίνακας 3.2 παραθέτει τις διαθέσιμες συναρτήσεις της κατηγορίας.

Συνάρτηση	**Σύνταξη**	**Επιστρέφει TRUE**
Αριθμητική ισότητα	(= <αριθμητικά ορίσματα>)	αν όλα τα ορίσματα είναι ίσα.
Μεγαλύτερο	(> < αριθμητικά ορίσματα>)	αν τα ορίσματα είναι κατά φθίνουσα σειρά.
Μικρότερο	(< <αριθμητικά ορίσματα>)	αν τα ορίσματα είναι κατά αύξουσα σειρά.
Μεγαλύτερο ή ίσο	(>= <αριθμητικά ορίσματα>)	αν τα ορίσματα είναι κατά φθίνουσα σειρά (όχι απόλυτη).
Μικρότερο ή ίσο	(<= <αριθμητικά ορίσματα>)	αν τα ορίσματα είναι κατά αύξουσα σειρά (όχι απόλυτη).
Διάφορο	(<> <αριθμητικά ορίσματα>)	αν τα ορίσματα δεν είναι ίσα.

Πίνακας 3.2. Συναρτήσεις σύγκρισης αριθμών.

Λογικές συναρτήσεις

Το CLIPS δίνει τη δυνατότητα να εκφραστούν πολύπλοκες συνθήκες κανόνων με τη χρήση των κλασσικών λογικών συναρτήσεων. Οι λογικές συναρτήσεις επιτρέπουν το συνδυασμό γεγονότων και κατά συνέπεια την πιο συμπαγή διατύπωση κανόνων.

Συνάρτηση	**Σύνταξη**	**Επιστρέφει TRUE**
Λογικό ΚΑΙ	(and <ορίσματα>)	αν όλα τα ορίσματα της είναι TRUE.
Λογικό Ή	(or <ορίσματα>)	αν ένα τουλάχιστον όρισμα είναι TRUE.
Λογική άρνηση	(not <ορίσματα>)	αν το όρισμα είναι FALSE.
Ισότητα	(eq <ορίσματα>)	αν τα ορίσματα είναι ίσα κατά τύπο και τιμή. Τα ορίσματα είναι οποιεσδήποτε συμβολοσειρές.
Ανισότητα	(neq <ορίσματα>)	αν τα ορίσματα δεν είναι ίσα κατά τύπο και τιμή. Τα ορίσματα είναι οποιεσδήποτε συμβολοσειρές.

Πίνακας 3.3. Λογικές συναρτήσεις.

Συναρτήσεις ελέγχου τύπου

Σε πολλές περιπτώσεις είναι δυνατό να πρέπει να εξακριβωθεί ο τύπος της τιμής μιας μεταβλητής, ώστε να γίνει η κατάλληλη επεξεργασία. Ο παρακάτω πίνακας παρουσιάζει τις διαθέσιμες συναρτήσεις για έλεγχο τύπων.

Συναρτήσεις ελέγχου	**Σύνταξη**	**Επιστρέφει TRUE**
Αριθμών	(numberp <όρισμα>)	αν το όρισμα είναι αριθμός.
Κινητής Υποδιαστολής	(floatp <όρισμα>)	αν το όρισμα είναι αριθμός κινητής υποδιαστολής.
Ακεραίου	(integerp <όρισμα>)	αν το όρισμα είναι ακέραιος αριθμός.
Αλφαριθμητικό	(stringp <όρισμα>)	αν το όρισμα είναι αλφαριθμητικό
Σύμβολο	(symbolp <όρισμα>)	αν το όρισμα είναι σύμβολο

Πίνακας 3.4. Συναρτήσεις ελέγχου τιμών.

Συναρτήσεις εισόδου – εξόδου

Σύνταξη: (printout <device> <expression>).

Αποστέλλει την έκφραση <expression> στη συγκεκριμένη συσκευή <device>. Η συσκευή μπορεί να είναι οποιαδήποτε συσκευή I/O, για παράδειγμα ένα αρχείο ή η οθόνη. Στη περίπτωση που η συσκευή είναι η οθόνη, η εντολή συντάσσεται με τιμή t (terminal) στην παράμετρο <device>.

Σύνταξη: (read)

Η εντολή εισάγει από την προκαθορισμένη συσκευή εισόδου (standard input) το επόμενο σύμβολο. Συνήθως χρησιμοποιείται σε συνδυασμό με την εντολή bind, η οποία όπως θα δούμε στα επόμενα αναθέτει τιμή σε μια μεταβλητή στις ενέργειες ενός κανόνα.

Συνάρτηση ανάθεσης τιμής σε μεταβλητή

Σύνταξη: (bind <variable> <value>)

Η εντολή χρησιμοποιείτε για να ανατεθεί <value> σε μια μεταβλητή <variable> στις ενέργειες των κανόνων.

Συναρτήσεις ελέγχου ροής προγράμματος

Σύνταξη: (while (condition) do
(command 1)
(command 2)
……
(command n))

Η συνάρτηση έχει την ίδια ακριβώς λειτουργία με τις αντίστοιχες συναρτήσεις των άλλων γλωσσών προγραμματισμού, δηλαδή όσο ικανοποιείται μια συνθήκη, εκτελείται ένα σύνολο συναρτήσεων. Πολλές φορές στην περίπτωση είναι απαραίτητη η χρήση της εντολής bind για να αλλάξει τις τιμές μεταβλητών που συμμετέχουν στη συνθήκη.

Σύνταξη: (if (condition)
then
(command 1)

(command 2)

…….

else

(command A)

……..

(command N))

Η κλασσική συνάρτηση ελέγχου η οποία είναι κοινή σε όλες σχεδόν τις γλώσσες προγραμματισμού. Εκτελεί υπό συνθήκη κάποια σύνολα εντολών.

Ορισμός συναρτήσεων στο CLIPS

Εκτός από τις προκαθορισμένες συναρτήσεις, μέρος των οποίων παρουσιάστηκε στις προηγούμενες παραγράφους, η γλώσσα δίνει τη δυνατότητα να ορίσει ο χρήστης τις δικές του συναρτήσεις. Ο ορισμός αυτών των συναρτήσεων γίνεται μέσω μιας ειδικής συνάρτησης, της deffunction.

Σύνταξη: (deffunction <function name)

(<variables>)

(command 1)

…….

(command n))

Θα πρέπει εδώ να σημειωθεί ότι η συνάρτηση επιστρέφει την τιμή της τελευταίας εντολής που εκτελείται. Η εντολή αυτή μπορεί να είναι μια συνάρτηση, ένα σύμβολο, ή μια μεταβλητή της οποίας η τιμή επιστρέφεται. Οι ορισμοί των συναρτήσεων, όπως και οι ορισμοί γεγονότων και κανόνων, μπορούν να περιέχονται σε ένα αρχείο κειμένου και να φορτώνονται μαζικά στο σύστημα.

3.6 Προτεραιότητα Κανόνων και Στρατηγικές Επίλυσης Συγκρούσεων

Όπως έχει αναφερθεί στην εισαγωγή, όλοι οι κανόνες των οποίων οι συνθήκες ικανοποιούνται εισάγονται στην ατζέντα (agenda), η οποία αντιστοιχεί στο σύνολο συγκρούσεων (conflict set) των κλασσικών συστημάτων παραγωγής. Από το σύνολο των κανόνων αυτών επιλέγεται κάθε φορά ένας κανόνας, ο οποίος και πυροδοτείται με βάση δύο κριτήρια : την προτεραιότητα των κανόνων και τη στρατηγική επίλυσης συγκρούσεων. Στην πράξη, η ατζέντα συμπεριφέρεται σαν μια στοίβα (stack) όπου όσο μεγαλύτερη προτεραιότητα έχει ένας κανόνας τόσο πιο ψηλά βρίσκεται στην κορυφή της στοίβας. Ένας νέος κανόνας τοποθετείται στην ατζέντα σύμφωνα με τα ακόλουθα κριτήρια :

1. Οι νέοι κανόνες μπαίνουν "πάνω" από όλους τους κανόνες με μικρότερη ή ίση προτεραιότητα (salience) και "κάτω" από όλους τους κανόνες με μεγαλύτερη προτεραιότητα.
2. Στους κανόνες με ίδια προτεραιότητα χρησιμοποιείται η τρέχουσα στρατηγική επίλυσης συγκρούσεων για να καθοριστεί η σειρά τους.
3. Εάν κάποιοι κανόνες ενεργοποιήθηκαν από το ίδιο σύνολο γεγονότων και τα προηγούμενα βήματα δεν μπόρεσαν να ορίσουν μια σειρά, τότε δίνεται σε αυτούς μια αυθαίρετη σειρά (όχι τυχαία), η οποία εξαρτάται από την υλοποίηση του συστήματος.

Στις επόμενες παραγράφους θα δοθεί μια σύντομη παρουσίαση της προτεραιότητας των κανόνων και των διαθέσιμων στρατηγικών επίλυσης συγκρούσεων.

3.6.1 Προτεραιότητα Κανόνων

Η προτεραιότητα ενός κανόνα ορίζεται μέσω της δήλωσης :

(declare (salience <number>)

στη συνάρτηση ορισμού του κανόνα. Για παράδειγμα :

```
(defrule Cartesian
(declare (salience 30))
(element ?a)
(element ?b)
=>
(printout t "Elements:   "?a"  "?b   crlf))
```

Η αριθμητική τιμή καθορίζει την προτεραιότητα του κανόνα. Όσο μεγαλύτερη είναι η τιμή αυτή τόσο μεγαλύτερη είναι και η προτεραιότητα του συγκεκριμένου κανόνα. Οι επιτρεπτές τιμές της προτεραιότητας είναι από -10000 έως 10000. Εάν δεν υπάρχει δήλωση, ο κανόνας θεωρείται ότι έχει την προκαθορισμένη τιμή μηδέν.

3.6.2 Στρατηγικές Επίλυσης Συγκρούσεων

Το CLIPS διαθέτει επτά διαφορετικές στρατηγικές επίλυσης συγκρούσεων. Σε κάθε χρονική στιγμή μόνο μια από αυτές είναι ενεργή και χρησιμοποιείται για την επιλογή του κανόνα από την ατζέντα. Η τρέχουσα στρατηγική δίνεται από τη συνάρτηση:

```
(get-strategy)
```

η οποία επιστρέφει το όνομα της στρατηγικής.

Ο ορισμός της επιθυμητή στρατηγικής επίλυσης συγκρούσεων γίνεται με τη χρήση της συνάρτησης:

```
(set-strategy <strategy>)
```

όπου <strategy>) είναι μια από τις τιμές : depth, breadth, simplicity, complexity, lex, mea, random. Η σημασία των τιμών αυτών είναι εξής:

depth: Σύμφωνα με αυτή τη στρατηγική οι νέοι κανόνες μπαίνουν "πάνω" από τους "παλαιούς". Η σύγκριση γίνεται με βάση το πότε εισήχθηκαν τα γεγονότα που ικανοποιούν τον κανόνα στη λίστα γεγονότων.

breadth: Αντίθετα με την προηγούμενη στρατηγική, οι νέοι κανόνες μπαίνουν "κάτω" από τους "παλαιούς".

simplicity: Οι κανόνες με απλούστερες συνθήκες κατατάσσονται "πάνω" από τους κανόνες με τις περισσότερο πολύπλοκες. Η πολυπλοκότητα ενός κανόνα εξαρτάται από τον αριθμό των συνθηκών και από τις συγκρίσεις (περιορισμούς) που λαμβάνουν χώρα στις συνθήκες του κανόνα.

complexity: Αντίθετα με την προηγούμενη στρατηγική, οι πιο πολύπλοκοι κανόνες μπαίνουν "πάνω" από τους "απλούστερους".

lex: Η στρατηγική αυτή πρωτοεμφανίστηκε στο σύστημα OPS5. Οι κανόνες οι οποίοι ενεργοποιούνται από "νεότερα" γεγονότα κατατάσσονται υψηλότερα στη agenda. Για τους κανόνες οι οποίοι κατατάσσονται στην "ίδια ομάδα" με βάση το προηγούμενο κριτήριο, υψηλότερα κατατάσσονται οι κανόνες οι οποίοι έχουν περισσότερες συνθήκες. Ουσιαστικά αποτελεί συνδυασμό των στρατηγικών depth και complexity.

mea: Και αυτή η στρατηγική εμφανίστηκε στο σύστημα OPS5. Εξετάζεται το γεγονός το οποίο αντιστοιχεί στην πρώτη συνθήκη και οι κανόνες διατάσσονται με βάση το πότε εισήχθηκε αυτό στη λίστα. Όσο νεότερο είναι το γεγονός τόσο "ψηλότερα" μπαίνει ο κανόνας. Για κανόνες οι οποίοι έχουν την ίδια σειρά με βάση αυτό το κριτήριο χρησιμοποιείται η στρατηγική lex.

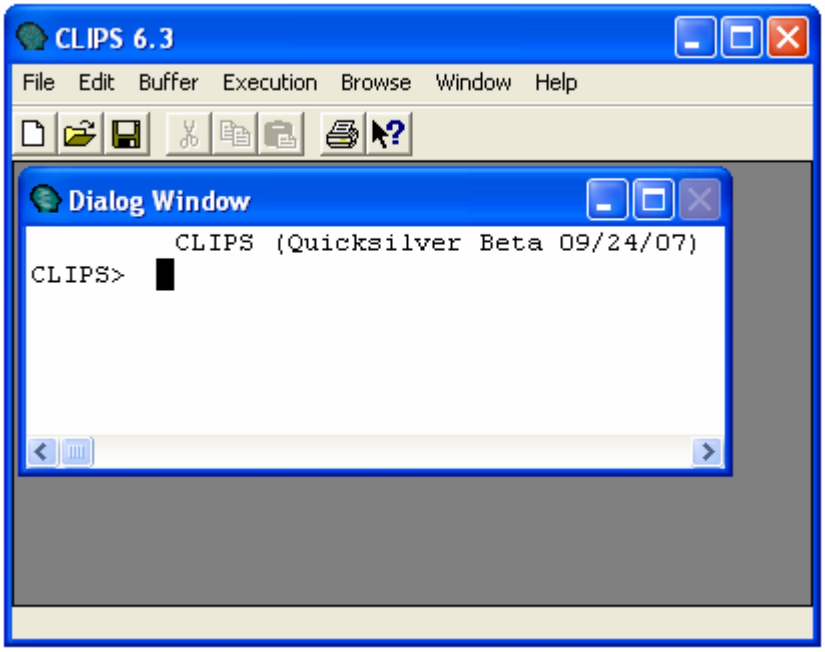
random: Οι κανόνες μπαίνουν στην ατζέντα με τυχαίο τρόπο.

Υπενθυμίζεται ότι η στρατηγική επίλυσης συγκρούσεων εφαρμόζεται σε κανόνες ίδιας προτεραιότητας.

3.7 Το περιβάλλον CLIPS

Το περιβάλλον του CLIPS εκκινεί σε περιβάλλον MS-Windows με την εκτέλεση του αρχείου CLIPSWIN.EXE και μπορεί κάποιος να το προμηθευτεί δωρεάν (freeware)

από το διαδίκτυο στην ιστοσελίδα http://www.ghg.net/clips/CLIPS.html . Την περίοδο αυτήν κυκλοφορεί η έκδοση CLIPS 6.3 Beta for Windows Release 2.



Σχήμα 3.1. Το περιβάλλον CLIPS.

ΚΕΦΑΛΑΙΟ 4

Το Έμπειρο Σύστημα A300-600 TES

4.1 Η Ανάγκη Δημιουργίας του Έμπειρου Συστήματος

Το έμπειρο σύστημα A300-600 TES αναπτύχθηκε στο σύστημα κανόνων CLIPS. Η εργασία διαπραγματεύεται όπως έχει προαναφερθεί με την διάγνωση (diagnosis) και την επιδιόρθωση βλαβών (troubleshooting) σε αεροσκάφη και συγκεκριμένα με το αεροσκάφος A300-600 της AIRBUS INDUSTRIES. Η μοντελοποίηση του συστήματος βασίστηκε στο Troubleshooting Manual (TSM) της εταιρείας και αποτελεί το εγκεκριμένο εγχειρίδιο για την διάγνωση και επιδιόρθωση βλαβών στον συγκεκριμένο τύπο του αεροσκάφους. Μάλιστα, χρησιμοποιείτε ευρέως από όλες τις αεροπορικές εταιρείες που διαθέτουν στο στόλο τους αεροσκάφη τέτοιου τύπου.

Η διάγνωση και επιδιόρθωση βλαβών σε πολύπλοκα συστήματα δεν μπορεί να επιτευχθεί χωρίς την συμβολή των απαραίτητων εγχειριδίων του προμηθευτή αλλά και της εμπειρίας των μηχανικών και τεχνικών. Η επιδιόρθωση μιας βλάβης απαιτεί εξειδικευμένες γνώσεις, και κατανόηση της λειτουργίας των συστημάτων. Βέβαια, ακόμα και οι πιο εξειδικευμένοι μηχανικοί και τεχνικοί δεν έχουν γνώσεις για το κάθε σύστημα του αεροσκάφους, αλλά χωρίζονται σε τμήματα ανάλογα με την ειδικότητα και το αντικείμενο εργασίας τους. Εδώ όμως παρουσιάζεται ένα μεγάλο πρόβλημα, αυτό της διακίνησης πληροφοριών και εγγράφων μέσα σε μια αεροπορική εταιρεία, πράγμα το οποίο είναι εξαιρετικά χρονοβόρο.

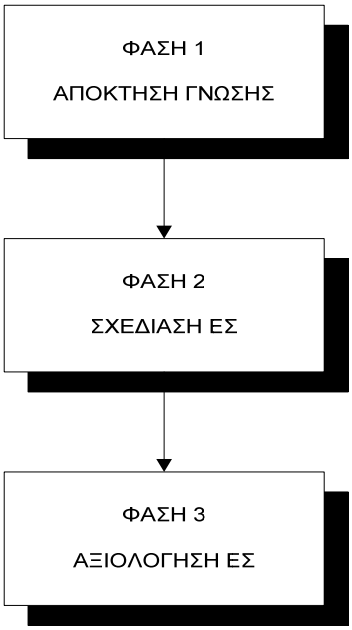
Από την στιγμή που ένα αεροσκάφος μπαίνει για προγραμματισμένη ή μη συντήρηση–επισκευή, εξειδικευμένα συνεργεία αναλαμβάνουν ανάλογα με τη φύση της δουλειάς που έχει να κάνει το καθένα, την αποπεράτωση του έργου τους. Στις αεροπορικές εταιρείες υπάρχουν ειδικά συνεργεία για την διάγνωση και επισκευή βλαβών. Μάλιστα υπάρχουν και αρκετά συστήματα εντός του αεροσκάφους που εμφανίζουν μηνύματα για βλάβες που παρουσιάστηκαν σε διάφορα υποσυστήματα με σκοπό να βοηθήσουν και να κάνουν ευκολότερη την δουλειά των τεχνικών. Αρκετές όμως βλάβες ανακαλύπτονται και κατά την προγραμματισμένη συντήρηση. Από

αυτές, κάποιες καμιά φορά επείγουν και άλλες όχι. Μόλις λοιπόν συγκεντρωθούν τα στοιχεία για την αποκατάσταση των βλαβών, στέλνονται σε ειδικευμένους μηχανικούς, οι οποίοι με την σειρά τους θα προτείνουν τις εγκεκριμένες διαδικασίες που πρέπει να ακολουθηθούν για τη αποκατάσταση των βλαβών. Πολλές φορές όμως οι μηχανικοί βρίσκονται σε αδιέξοδα όσον αφορά την επιδιόρθωση αφού μπορεί να μην έχουν ξανασυναντήσει παρόμοιο περιστατικό. Αυτό έχει σαν αποτέλεσμα την τηλεφωνική ή προσωπική συνεργασία με εξειδικευμένους ανθρώπους κατασκευαστριών αεροπορικών εταιρειών, κάτι το οποίο ίσως να πάρει αρκετό καιρό με όλα τα αρνητικά επακόλουθα για την προγραμματισμένη πτητικότητα του αεροσκάφους.

Στις μέρες μας γίνεται προσπάθεια από όλους, ώστε να δημιουργηθούν συστήματα τα οποία να αποτελούν αναπόσπαστο κομμάτι της δουλειάς των μηχανικών και των τεχνικών, σε μια προσπάθεια να βελτιστοποιηθεί ο χρόνος της συντήρησης των αεροσκαφών και να ελαχιστοποιηθεί ο χρόνος καθήλωσης τους στα υπόστεγα συντήρησης. Καμιά αεροπορική εταιρεία δεν θέλει αεροσκάφη αρκετά μεγάλα διαστήματα καθηλωμένα στο έδαφος, αφού δεν χάνουν μόνο χρήματα και επιβάτες αλλά χάνουν και την αξιοπιστία τους.

4.2 Ανάλυση και σχεδίαση του έμπειρου συστήματος A300-600 TES

Για την ανάλυση και σχεδίαση του έμπειρου συστήματος διάγνωσης και επιδιόρθωσης βλαβών που παρουσιάζεται στην παρούσα εργασία, χρειάστηκε η βοήθεια και η εμπειρία πολλών ανθρώπων. Αρχικά έπρεπε να αποκτηθεί η γνώση για το πώς οι τεχνικοί και μηχανικοί αεροσκαφών επιδιορθώνουν μια βλάβη. Σε μια πολύπλοκη μηχανή όπως είναι το αεροσκάφος, αρχικά έπρεπε να καταγραφούν τα επιμέρους βήματα που ακολουθούνται για την αποκατάσταση των βλαβών. Σε δεύτερη φάση σχεδιάστηκε το σύστημα και η βάση γνώσης του και τέλος έγινε η αξιολόγηση του συστήματος σύμφωνα των αποτελεσμάτων του.



Σχήμα 4.1. Στάδια ανάλυσης και σχεδίασης ΕΣ

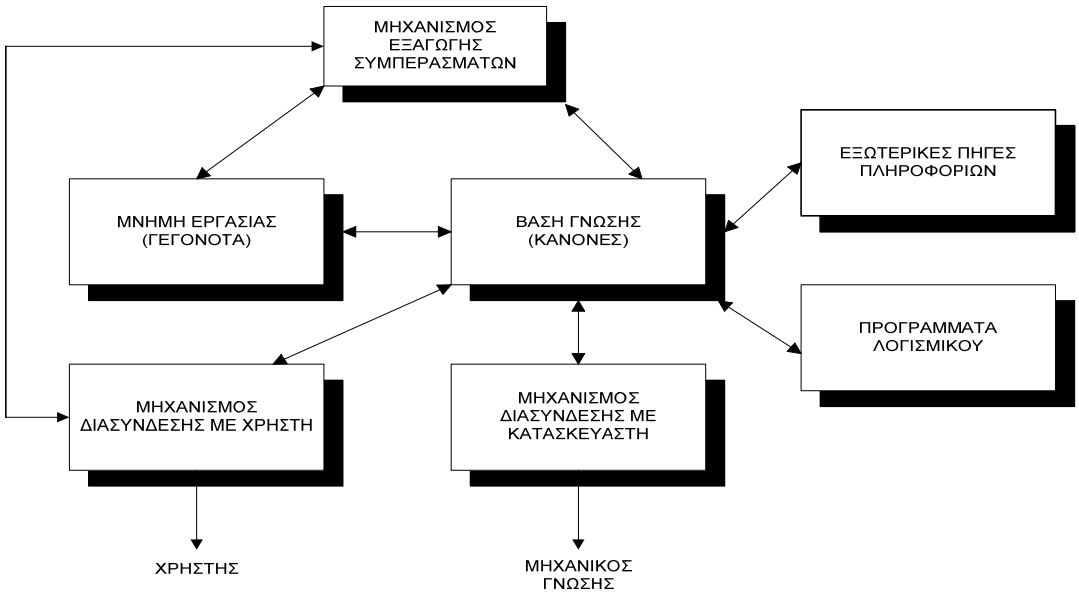
Για να ολοκληρωθεί το πρώτο στάδιο αυτό της απόκτησης της γνώσης, χρειάστηκε η συμβολή εξειδικευμένων ανθρώπων, οι οποίοι με την εμπειρία τους κατέδειξαν τις μεθόδους που χρησιμοποιούνται για την διάγνωση και επιδιόρθωση βλαβών.

Σχήμα 4.2. Διαδικασία απόσπασης γνώσης.

Η σχεδίαση του έμπειρου συστήματος αποτελείται από διάφορα μέρη. Τα τρία κύρια τα οποία αποτελούν την καρδιά του συστήματος είναι:

- ✓ Η Βάση Γνώσης
- ✓ Ο Μηχανισμό Εξαγωγής Συμπερασμάτων
- ✓ Η Μνήμη εργασίας

Ο αρχιτεκτονικός σχεδιασμός του υλοποιημένου συστήματος φαίνεται παρακάτω:



Σχήμα 4.3. Αρχιτεκτονική του έμπειρου συστήματος A300-600 TES

Η βάση γνώσης και ο μηχανισμός εξαγωγής συμπερασμάτων θα αναλυθούν διεξοδικά σε επόμενες παραγράφους. Όσον αφορά τώρα τα υπόλοιπα υποσυστήματα του Ε.Σ. :

Υποσύστημα επικοινωνίας χρήστη συστήματος

Είναι ο μηχανισμός μέσω του οποίου χρήστης βλέπει και επικοινωνεί με το σύστημα. Ο χρήστης μπορεί να βλέπει γραφικές παραστάσεις ή και κείμενα στην οθόνη του υπολογιστή του, ανάλογα των πληροφοριών που ζητάει και των δυνατοτήτων που προσφέρει το σύστημα.

Μηχανισμός διασύνδεσης κατασκευαστή

Ο μηχανισμός αυτός χρησιμοποιείται από τον κατασκευαστή για να υλοποιηθεί το υπό ανάπτυξη σύστημα και αφορά τη χρήση κάποιου πακέτου λογισμικού για την σύνταξη του πηγαίου κώδικα.

Εξωτερικές πηγές πληροφοριών

Το κομμάτι αυτό του Ε.Σ. περιέχει την γνώση που θα χρησιμοποιηθεί για την κατασκευή της βάσης γνώσης. Η γνώση μπορεί να είναι κάποιο εγχειρίδιο ή ανθρώπινη γνώση πάνω σε ένα τομέα (π.χ. παθολογία).

Προγράμματα λογισμικού

Τα περισσότερα προγράμματα ανάπτυξης Ε.Σ. που κυκλοφορούν στο εμπόριο είναι 'ανοικτής αρχιτεκτονικής' και επιτρέπουν την αλληλεπίδραση του υπό ανάπτυξη έμπειρου συστήματος με προγράμματα λογισμικού. Η δυνατότητα αυτή είναι πολύ σημαντική, διότι πληροφορίες οι οποίες είναι αποθηκευμένες σε άλλα μέσα μπορούν πολύ εύκολα να συνεργαστούν με το έμπειρο σύστημα.

Τρίτο και τελευταίο στάδιο της ανάλυσης και σχεδίασης είναι το στάδιο της αξιολόγησης του υλοποιημένου Ε.Σ. Το σύστημα πρέπει να δίνει σωστά αποτελέσματα τα οποία βασίζονται σε κάποια συλλογιστική.

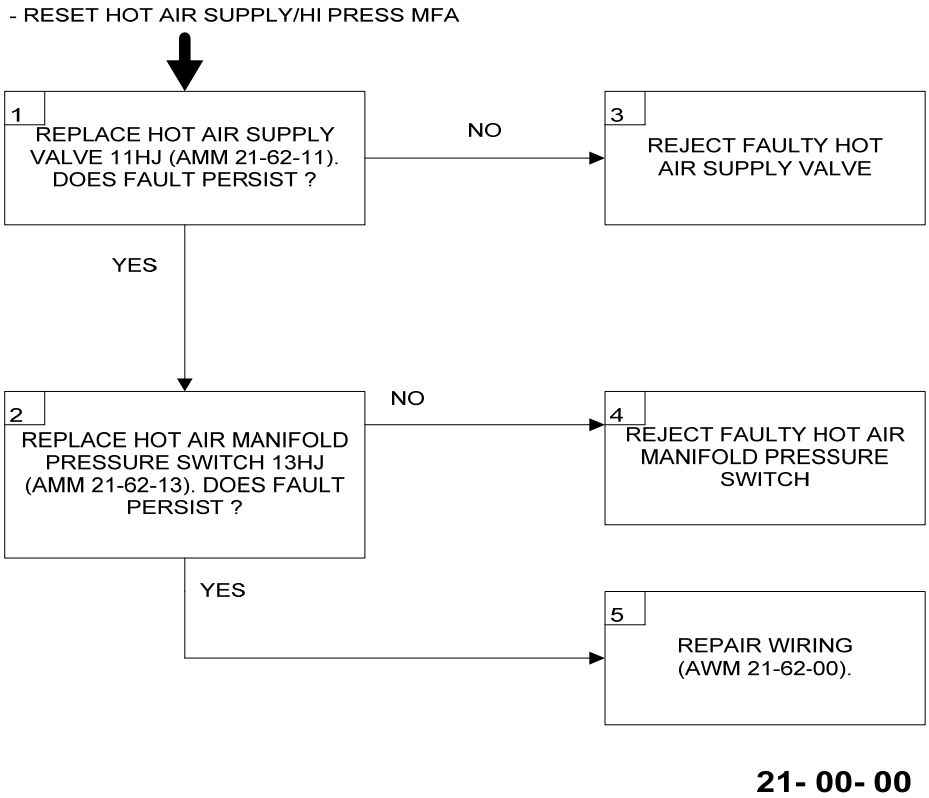
4.3 Ανάλυση και σχεδίαση της βάσης γνώσης

Κύρια πηγή για την ανάπτυξη της βάσης γνώσης αποτέλεσε το εγχειρίδιο Troubleshooting Manual (TSM) της AIRBUS. Η βάση γνώσης αποτελείται από ένα σύνολο από έτοιμα πλάνα διάγνωσης και επισκευής βλαβών αναπαριστώντας προκαθορισμένες διαδικασίες και πρωτοκόλλα. Η βάση γνώσης είναι ουσιαστικά ένας χάρτης ροής (flow chart) που περιγράφει τα βήματα που πρέπει να ακολουθήσει ένας τεχνικός για να διαγνώσει και να επιδιορθώσει μια βλάβη.

Το σχήμα 4.4 δείχνει ένα απόσπασμα χάρτη ροής του έμπειρου συστήματος, και καθοδηγεί τον τεχνικό σε μια συγκεκριμένη διαδικασία επιδιόρθωσης. Ένας χάρτης ροής έχει κόμβους, οι οποίοι αναπαριστούν διαγνώσεις, αποφάσεις με βάση τα αποτελέσματα, καθώς και ακολουθίες ενεργειών επιδιόρθωσης. Το υλοποιημένο σύστημα διαθέτει ένα σύστημα διασύνδεσης με το χρήστη, τόσο για την εκτέλεση

ενός πλάνου διάγνωσης και επιδιόρθωσης, όσο και για τον εμπλουτισμό της βάσης γνώσης με νέα πλάνα.

Το βασικό πλεονέκτημα της χρήσης ενός υπολογιστή για την παρουσίαση-εκτέλεση ενός πλάνου είναι η αλληλεπίδραση του τεχνικού με τον υπολογιστή. Το σύστημα παρουσιάζει στον τεχνικό κάθε φορά μόνο το επόμενο που αυτός πρέπει να εκτελέσει, απαλλάσσοντας τον από τον κόπο να ψάχνει μέσα στις σελίδες του εγχειριδίου. Επιπλέον το σύστημα μπορεί να καταγράφει τις ενέργειες που εκτέλεσε ο τεχνικός για μελλοντική αναφορά.



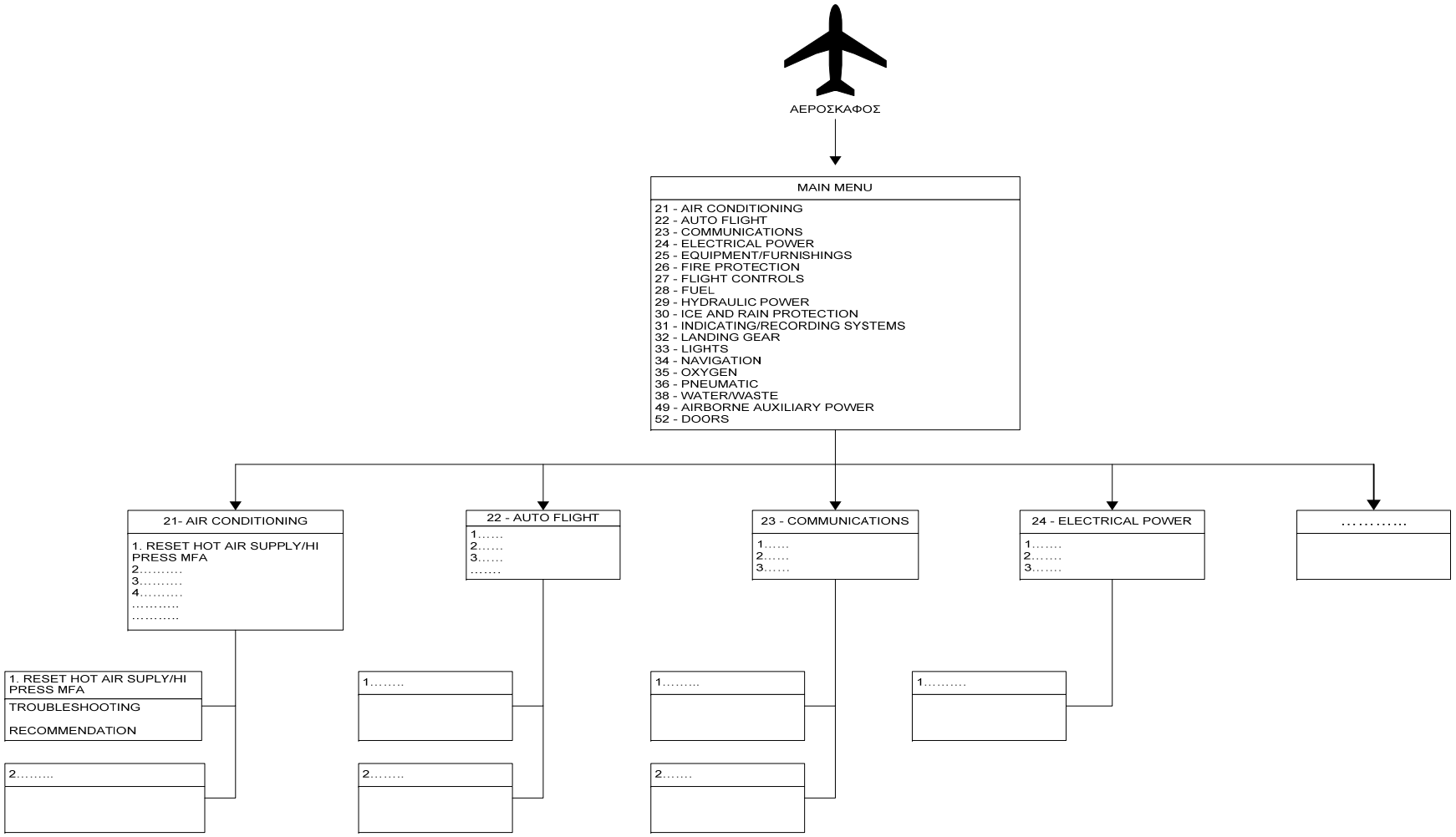
Σχήμα 4.4. Σχηματική αναπαράσταση επιδιόρθωσης βλάβης (*Πηγή: A300-600 Troubleshooting Manual, Airbus Industries*)

Η βάση γνώσης περιέχει 66 βλάβες και αποτελεί ένα μικρό κομμάτι του εγχειριδίου, αν αναλογιστούμε ότι περιέχονται σε αυτό πάνω από 3000 έτοιμα πλάνα διάγνωσης και επιδιόρθωσης βλαβών.

Το εγχειρίδιο είναι χωρισμένο στα διάφορα συστήματα του αεροσκάφους και για τον διαχωρισμό τους χρησιμοποιείται ένας διψήφιος αριθμός ταξινόμησης. Ο αριθμός που

βλέπουμε κάτω δεξιά στο σχήμα 4.4 είναι ο αντιπροσωπευτικός αριθμός ταξινόμησης της συγκεκριμένης βλάβης. Τα πρώτα δύο ψηφία αφορούν το σύστημα του αεροσκάφους και τα υπόλοιπα της βλάβης.

Η βάση γνώσης σχεδιάστηκε με βάση την συλλογιστική αυτήν, ώστε ο τεχνικός που θα χρησιμοποιήσει το σύστημα να μην χρειαστεί να εκπαιδευτεί επιπλέον σε καινούργιες μεθοδολογίες διάγνωσης και επιδιόρθωσης. Σχηματικά η βάση γνώσης παρουσιάζεται στο σχήμα 4.5:



Σχήμα 4.5. Η βάση γνώσης.

4.4 Δομή της βάσης γνώσης

Η βάση γνώσης αποτελείται στο σύνολο της από γεγονότα και κανόνες. Ειδικότερα η βάση γνώσης περιέχει 196 κανόνες και είναι χωρισμένοι σε κατηγορίες. Οι τρείς κύριες κατηγορίες κανόνων είναι αυτές που αφορούν την διάγνωση βλαβών, την επιδιόρθωση τους και τέλος η κατηγορία κανόνων για τα διάφορα μενού. Στην παράγραφο αυτήν θα γίνει μια λεπτομερής ανάλυση της δομής της βάσης γνώσης.

Γεγονότα

Όπως έχουμε αναφέρει σε προηγούμενο κεφάλαιο τα γεγονότα αποτελούν την πληροφορία την οποία το σύστημα 'γνωρίζει' και στην οποία βασίζεται ώστε να εξάγει συμπεράσματα. Τα γεγονότα είναι λίστες από σύμβολα τα οποία περικλείονται σε παρενθέσεις.

(initial – fact): Δημιουργείται από το CLIPS κάθε φορά που εκκινεί.

Τα γεγονότα που υπάρχουν στη μνήμη εργασίας δεν είναι στατικά και μπορούν να εισαχθούν ή να διαγραφούν από αυτή κατά την εκτέλεση του προγράμματος.

Υπάρχουν ακόμα δύο γεγονότα τα οποία είναι αποθηκευμένα στη λίστα γεγονότων με χαρακτηριστικό αριθμό για το καθένα:

(troubleshoot – mode aircraft) και (menu – level aircraft main)

Τα γεγονότα αυτά τα έχουμε ορίσει εμείς και χαρακτηρίζουν μοναδικά ένα γεγονός. Χρησιμεύουν τόσο στη διαγραφή γεγονότων, όσο και στο να βοηθά το σύστημα να γνωρίζει ποια γεγονότα πυροδότησαν ένα κανόνα.

Κανόνες

Οι κανόνες του έμπειρου συστήματος της παρούσης εργασίας, είναι της μορφής

if (συνθήκες) *then* (ενέργειες)

Οι συνθήκες είναι τις περισσότερες φορές ένα σύνολο από γεγονότα τα οποία θα πρέπει να υπάρχουν στη λίστα γεγονότων για να ικανοποιείται ο κανόνας. Οι ενέργειες του κανόνα περιγράφουν το τι θα λάβει χώρα κατά την πυροδότηση του. Οι συνθήκες αποτελούν το αριστερό μέρος του κανόνα (left handside – LHS), ενώ οι ενέργειες το δεξί μέρος του κανόνα (right handside – RHS) και διαχωρίζονται μεταξύ τους με το σύμβολο '=>'.

Από το σημείο αυτό θα εισάγουμε αντιπροσωπευτικά δείγματα κώδικα, γραμμένο στο CLIPS, για να καταλάβει ο αναγνώστης τη δομή του έμπειρου συστήματος που κατασκευάστηκε σε αυτήν την εργασία.

Ο πρώτος κανόνας που φαίνεται παρακάτω είναι μια συνάρτηση η οποία επιστρέφει την τιμή της εντολής. Ανάλογα δηλαδή της τιμής που θα δώσει ο χρήστης yes/no ή y/n η συνάρτηση η συνάρτηση επιστρέφει την τιμή TRUE ειδάλλως επιστρέφει την τιμή FALSE.

Κανόνας 1

```
(deffunction ask-yes-or-no (?question)
        (bind ?response unknown)
        (while(and(neq ?response n)(neq ?response no)(neq ?response y)(neq ?response yes))
        (printout t ?question "  ")
        (bind ?response (read)))
        (if(or(eq ?response yes)(eq ?response y))
        then(return TRUE)
        else(return FALSE)))
```

Οι κανόνες που ακολουθούν στη συνέχεια είναι ένας κανόνας καλωσορίσματος του έμπειρου συστήματος προς τον χρήστη και τρείς κανόνες ερωτήσεων που αφορούν τον αριθμό της πτήσης, την ημερομηνία εκδήλωσης της βλάβης και τον αριθμό νηολογίου του αεροσκάφους και χρησιμεύουν για αρχειοθέτηση και μελλοντική αναφορά σε περίπτωση παρόμοιας κατάστασης.

Κανόνας 2

```
(defrule system-banner
        (declare(salience 900))
=>
(printout t crlf crlf)
(printout t "The A300-600 Troubleshooting Expert System")
(printout t crlf crlf crlf crlf))
```

Κανόνες 3 έως 5

```
(defrule ask-acreg
        (declare(salience 800))
=>
(printout t
"Please give aircraft's registration number :")
(bind ?response (read))
(assert(acreg ?response))
(printout t crlf))
;;
(defrule ask-date
        (declare(salience 700))
=>
(printout t
"Please enter the current date that fault occured :")
(bind ?response (read))
(assert(date ?response))
(printout t crlf ))
;;
(defrule ask-flt
        (declare(salience 600))
=>
(printout t
"Please give the flight number :")
(bind ?response (read))
(assert(flt ?response)))
```

Όπως φαίνεται στον παραπάνω κώδικα, οι κανόνες έχουν οριστεί μέσω της προτεραιότητας τους με την εντολή

(declare (salience <number>))

Αυτό έγινε πιο πολύ για λόγους καλής λειτουργικής εμφάνισης του προγράμματος.

Εν συνεχεία ακολουθεί ο κανόνας εμφάνισης του κύριου μενού από το οποίο ο χρήστης επιλέγει το σύστημα του αεροσκάφους που θέλει να διαγνώσει μια ή περισσότερες βλάβες:

Κανόνας 6

```
(defrule main-menu
        (declare (salience 500))
        (troubleshoot-mode aircraft)
        ?ml<-(menu-level aircraft main)
=>
        (retract ?ml)
        (printout t crlf crlf)
        (printout t
"       Choose one of the sections listed below" crlf
"       by typing a number and pressing the return key." crlf crlf
"           1.)Air conditioning." crlf
"           2.)Auto flight." crlf
```

```
"            3.)Communications." crlf
"            4.)Electrical power." crlf
"            5.)Equipment/furnishings." crlf
"            6.)Fire protection." crlf
"            7.)Flight controls." crlf
"            8.)Fuel." crlf
"            9.)Hydraulic power." crlf
"            10.)Ice and rain protection." crlf
"            11.)Indicating/recoding systems." crlf
"            12.)Landing gear." crlf
"            13.)Lights." crlf
"            14.)Navigation." crlf
"            15.)Oxygen." crlf
"            16.)Pneumatic." crlf
"            17.)Water/waste." crlf
"            18.)Airborne auxiliary power." crlf
"            19.)Doors." crlf
"            20.)Quit the program." crlf crlf
"        Choise:")
(bind ?response(read))
(assert(problem-response aircraft ?response))
(printout t crlf))
```

Ακολουθεί ο κανόνας τερματισμού του προγράμματος με την επιλογή 20 του χρήστη από το κύριο μενού

Κανόνας 7

```
(defrule user-quits
        (troubleshoot-mode aircraft)
        (problem-response aircraft 20)
=>
(printout t "You have QUIT the A300-600 Troubleshooting Expert System." crlf)
(halt))
```

Εν συνεχεία ακολουθούν μερικοί κανόνες, οι οποίοι είναι πιο πολύ βοηθητικής μορφής και χρειάζονται για να εξάγουν την επιλογή του χρήστη από το κύριο μενού:

Κανόνας 8

```
(defrule air-conditioning
        (troubleshoot-mode aircraft)
        ?pr<-(problem-response aircraft 1)
=>
(retract ?pr)
(assert(menu-level aircraft possible-faults-1-1))
(assert(problem aircraft Air-conditioning)))
```

Οι επόμενοι κανόνες αποτελούν τους κανόνες διάγνωσης βλαβών του συστήματος και ένα αντιπροσωπευτικό δείγμα κανόνα γραμμένο στο CLIPS φαίνεται παρακάτω:

Κανόνας 9

```
(defrule possible-faults-1-1
        (troubleshoot-mode aircraft)
        (menu-level aircraft possible-faults-1-1)
        (problem aircraft ?problem&Air-conditioning)
=>
(printout t crlf crlf crlf)
(printout t
"You selected "?problem" as the section you would" crlf
"like to view. If you change your mind and would like" crlf
"to review the main menu, press 0 now, otherwise select one" crlf
"of the possible faults of "?problem" listed below" crlf
"to begin the fault isolation."
crlf crlf
"0.)Return to main menu." crlf
"1.)RESET HOT AIR SUPPLY/HI PRESS MFA (TSM task 21-00-00)." crlf
"2.)INBD FLOWBAR ON ON GROUND WITH ONE OR BOTH ENGINES NOT
RUNNING (TSM task 21-20-00)." crlf
"3.)IN FLIGHT INBD FLOWBAR IS ON. FAULT LEGEND ON OVBD VALVE P/BSW IS
ON (TSM task 21-20-00)." crlf
"4.)ON GROUND WITH BOTH ENGINES RUNNING. INBD FLOWBAR IS ON. 'FAULT'
LEGEND ON OVBD VALVE P/BSW IS ON (TSM task 21-20-00)." crlf
"5.)IN FLIGHT INBD FLOWBAR IS ON. WITH EXTRACT P/BSW PRESSED (IN),
'FLOW' LEGEND IS ON AND DOES NOT GO OFF WHEN EXTRACT P/BSW IS
RELEASED (OUT) (TSM task 21-20-00)." crlf
"6.)LAMP & AC PWR TEST NEGATIVE (TSM task 21-30-00)." crlf
"7.)DC PWR TEST NEGATIVE (TSM task 21-30-00)." crlf
"8.)RATE LIMIT TEST NEGATIVE (TSM task 21-30-00)." crlf
"9.)LANDING ELEVATION TEST NEGATIVE (TSM task 21-30-00)." crlf
"10.)BARO SET TEST NEGATIVE (TSM task 21-30-00)." crlf
"11.)FWD VALVE TEST NEGATIVE (TSM task 21-30-00)." crlf
"12.)AFT VALVE TEST NEGATIVE (TSM task 21-30-00)." crlf
"13.)PACK TEMPERATURE CONTROL FAULT (TSM task 21-61-00)." crlf crlf
"Choise:")
(bind ?response(read))
(assert(possible-fault ?problem ?response))
(printout t crlf))
```

Ο παραπάνω κανόνας έχει άμεση σχέση με τον προηγούμενο και αυτό γιατί, ο κανόνας 8 'αφαιρεί' την επιλογή του χρήστη από το κύριο μενού και εμφανίζει το μενού του κανόνα 9.

Επόμενη κατηγορία κανόνων είναι αυτοί οι οποίοι αφαιρούν την επιλογή του χρήστη από το μενού του κανόνα 9 και είναι της μορφής:

Κανόνας 10

```
(defrule numeric-to-text-reset-hot-air-supply-hi-press-mfa-1-1
        (troubleshoot-mode aircraft)
        ?ml<-(menu-level aircraft possible-faults-1-1)
        ?pc<-(possible-fault ?problem 1)
=>
(retract ?ml)
(assert(menu-level aircraft possible-faults-1-1-1))
```

```
(retract ?pc)
(assert(possible-fault ?problem reset-hot-air-supply-hi-press-mfa)))
```

Σειρά τώρα έχουν οι κανόνες επιδιόρθωσης βλάβης.

Κανόνας 11

```
(defrule possible-fault-reset-hot-air-supply-hi-press-mfa
        (troubleshoot-mode aircraft)
        (menu-level aircraft possible-faults-1-1-1)
        ?pc<-(possible-fault ?problem reset-hot-air-supply-hi-press-mfa)
=>
(retract ?pc)
(printout t crlf)
(printout t
"You choose RESET HOT AIR SUPPLY/HI PRESS MFA (TSM task 21-00-00) to
troubleshoot." crlf
"Answer with yes/no or y/n until the Recommendation.") crlf crlf
(printout t crlf crlf)
(printout t
"PREREQUISITES" crlf
"EXTERNAL POWER (AMM 24-41-00, P. BLOCK 301)." crlf
"C/B 'S IN : 324/L78, L79.") crlf
(printout t crlf crlf)
(if(ask-yes-or-no "REPLACE HOT AIR SUPPLY VALVE 11HJ (AMM 21-62-11) .DOES
FAULT PERSIST ?")
        then
        (if(ask-yes-or-no "REPLACE HOT AIR MANIFOLD PRESSURE SWITCH
13HJ(AMM 21-62-13).DOES FAULT PERSIST ?")
        then(assert(Recommendation "REPAIR WIRING (AWM 21-62-00)."))
        else(assert(Recommendation "REJECT FAULTY HOT AIR MANIFOLD
PRESSURE SWITCH.")))
        else
        (assert(Recommendation "REJECT FAULTY HOT AIR SUPPLY VALVE."))))
```

Όπως φαίνεται στον παραπάνω κώδικα, ο κανόνας αυτός αφορά την επιδιόρθωση συγκεκριμένης βλάβης του αεροσκάφους η οποία έχει επιλεχθεί μέσα από τα μενού που εμφανίζονται κατά την διάρκεια 'τρεξίματος' του προγράμματος. Ο κανόνας αυτός κατασκευάστηκε σύμφωνα με το πλάνο του εγχειριδίου της κατασκευάστριας αεροπορικής εταιρείας (σχήμα 4.4). Μέσα από μια διαδικασία ερωτοαπαντήσεων ναι/όχι ο χρήστης φτάνει στο σημείο της συμβουλής (recommendation) επιδιόρθωσης της βλάβης.

Ακολουθούν κανόνες με τους οποίους κινούμαστε στα διάφορα μενού και είναι της μορφής:

Κανόνας 12

```
(defrule ascend-to-main-menu-1
```

```
        ?ml<-(menu-level aircraft possible-faults-1-1)
        ?pc<-(possible-fault ?problem 0)
=>
(retract ?ml)
(retract ?pc)
(assert(menu-level aircraft main)))
```

Τέλος υπάρχει ο κανόνας με τον οποίο το σύστημα αφαιρεί την εντολή (recommendation) από τον κανόνα και μας την τυπώνει στην οθόνη του υπολογιστή του χρήστη.

Κανόνας 13

```
(defrule print-recommendation
(Recommendation ?r)
=>
(printout t "Recommendation is " ?r crlf))
```

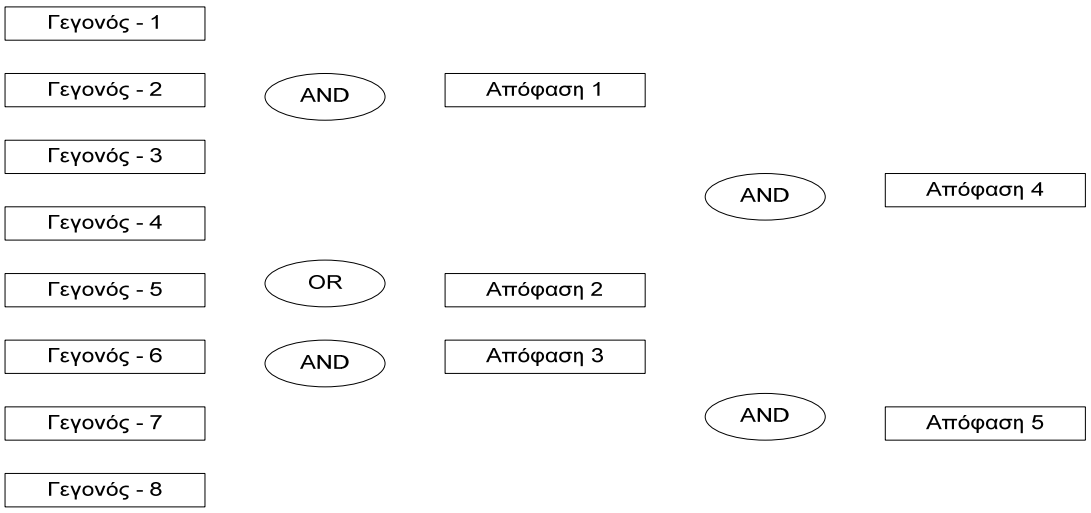
4.5 Μηχανισμός εξαγωγής συμπερασμάτων

Ο μηχανισμός εξαγωγής συμπερασμάτων είναι υπεύθυνος για τη διαχείριση και τον έλεγχο της γνώσης που βρίσκεται αποθηκευμένη τόσο στη βάση γνώσης όσο και στη μνήμη εργασίας, με στόχο τη διαμόρφωση συμπερασμάτων.

Το έμπειρο σύστημα που υλοποιείται στην παρούσα εργασία χρησιμοποιεί την προς τα εμπρός συλλογιστική (forward reasoning or chaining) για να εξάγει τα αποτελέσματα.

Η προς τα εμπρός συλλογιστική ξεκινά, από τα δεδομένα που διαθέτει και προσπαθεί να καταλήξει σε συμπεράσματα. Είναι δηλαδή μια προσέγγιση καθοδηγούμενη από τα δεδομένα (data – driven). Η αναζήτηση ξεκινά, με τη δημιουργία του αρχικού επιπέδου του δένδρου εξαγωγής συμπερασμάτων, με τη στρατηγική της προς τα εμπρός συλλογιστικής. Σαν ρίζα τοποθετείται η αρχική πληροφορία (η υπόθεση του κανόνα: IF υπόθεση THEN συμπέρασμα). Στη συνέχεια δημιουργείται το επόμενο επίπεδο. Αρχικά ανευρίσκονται όλοι οι κανόνες των οποίων τα αριστερά μέρη ταυτίζονται με τη πληροφορία της ρίζας. Στη συνέχεια τοποθετούνται σαν κόμβοι του επιπέδου αυτού, τα δεξιά μέρη τους (συμπεράσματα). Για τη δημιουργία κάθε επόμενου επιπέδου επαναλαμβάνεται η διαδικασία αυτή για κάθε κόμβο του

προηγούμενου επιπέδου. Η διαδικασία τερματίζεται μόλις επιτευχθεί ο στόχος αναζήτησης.



Σχήμα 4.6. Προς τα εμπρός συλλογιστική

Οι ενέργειες που γίνονται είναι:

- ✓ Αρχικά δίνονται κάποια γεγονότα (facts). Εργαζόμαστε προχωρώντας προς τα εμπρός κατά μήκος του δικτύου της συλλογιστικής (σχήμα 4.6).
- ✓ Διερευνώνται τι συμπεράσματα μπορούν να προκύψουν από τα δεδομένα.
- ✓ Εάν οι κανόνες που ταυτοποιούνται ότι ισχύουν είναι περισσότεροι του ενός είτε ενεργοποιούνται όλοι οι κανόνες και γίνεται η παράλληλη επεξεργασία τους είτε επιλέγεται μόνο ένας από αυτούς.

4.6 Μνήμη εργασίας

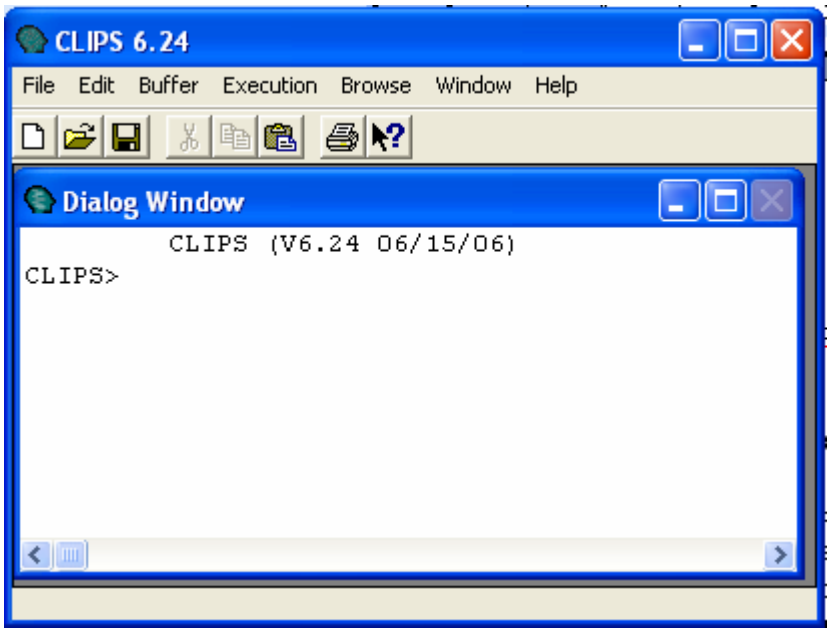
Η μνήμη εργασίας περιέχει όλες τις πληροφορίες που είτε αφορούν το πρόβλημα, είτε απορρέουν από το σύστημα. Η πληροφορία που αποκτάται κατά την διάρκεια της εργασίας ονομάζεται περιεχόμενο (context). Η μνήμη εργασίας (working memory) είναι εκείνο το μέρος του έμπειρου συστήματος το οποίο περιέχει τα γεγονότα (facts) του προβλήματος που προκύπτουν κατά τη διάρκεια χρήσης του προγράμματος (Durkin, 1994). Στη διάρκεια της συνεργασίας του, με το έμπειρο σύστημα, ο χρήστης εισάγει στη μνήμη εργασίας πληροφορίες για το συγκεκριμένο πρόβλημα. Το σύστημα συνδυάζει τις πληροφορίες αυτές με την γνώση που ήδη υπάρχει στην

βάση γνώσης και εξάγει καινούργια γεγονότα. Στη συνέχεια εισάγει τα καινούργια γεγονότα στην μνήμη εργασίας και η διαδικασία ταύτισης συνεχίζεται. Τελικά το σύστημα καταλήγει σε κάποιο συμπέρασμα το οποίο, με τη σειρά του, εισάγεται στην μνήμη εργασίας.

Παρουσίαση του συστήματος μέσω εφαρμογής

Το περιβάλλον του CLIPS εκκινεί σε περιβάλλον MS-Windows με την προϋπόθεση ότι έχουμε "κατεβάσει" το αρχείο από τον δικτυακό τόπο http://www.ghg.net/clips/CLIPS.html και το έχουμε εγκαταστήσει στον υπολογιστή μας. Η διαδικασία είναι πολύ απλή αφού το μόνο που έχουμε να κάνουμε είναι να κάνουμε extract το αρχείο zip σε ένα φάκελο της επιλογής μας. Η χρήση του είναι δωρεάν (freeware) και εκκινεί σε όλα τα γνωστά λειτουργικά συστήματα (DOS, WINDOWS, UNIX, VMS).

Αμέσως μετά εμφανίζεται στην οθόνη το γραφικό περιβάλλον που βλέπουμε στο παρακάτω σχήμα 5.1.

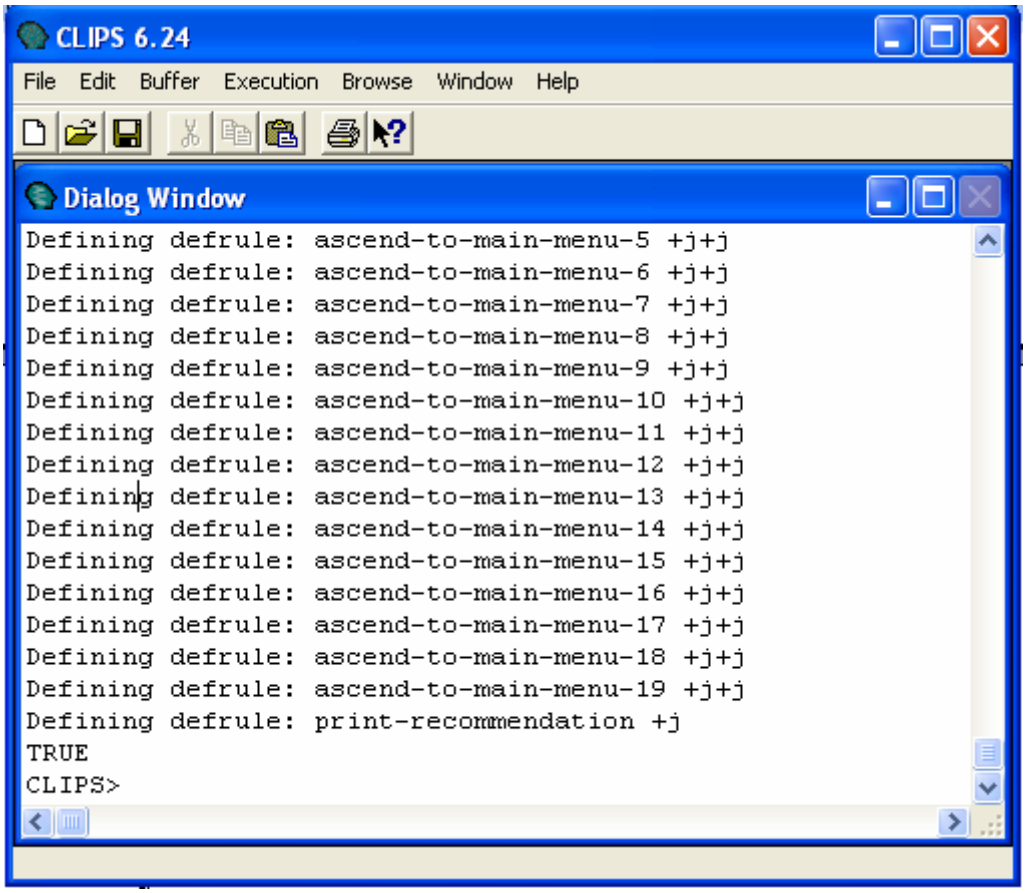


Σχήμα 5.1. Το Περιβάλλον του CLIPS

Για να φορτωθεί το αρχείο με την ονομασία A300-600_TES.clp επιλέγεται από το μενού File η εντολή Load και μέσα από το σχετικό παράθυρο Load CLIPS Constructs. Επιλέγεται το επιθυμητό αρχείο.

Στο σημείο αυτό, η μεταγλώττιση (compile) του προγράμματος αρχίζει να εμφανίζεται στη οθόνη. Με την προϋπόθεση ότι ο κώδικας μας είναι σωστά

γραμμένος το σύμβολο TRUE εμφανίζεται στο τέλος της μεταγλώττισης, διαφορετικά θα εμφανιστεί το σύμβολο FALSE. Τα “+j” και “=j” που εμφανίζονται στο τέλος της γραμμής κάθε κανόνα είναι πληροφορίες του CLIPS για την εσωτερική δομή των μεταγλωττισμένων κανόνων.



Σχήμα 5.2.Φόρτωση του A300-600 TES

Για να “τρέξουμε” το πρόγραμμα, από το μενού Execution επιλέγουμε πρώτα Reset και έπειτα Run. Αμέσως εμφανίζεται στην οθόνη ένα μήνυμα καλωσορίσματος και η πρώτη ερώτηση του συστήματος :

The A300-600 Troubleshooting Expert System

Please give aircraft’s registration number:

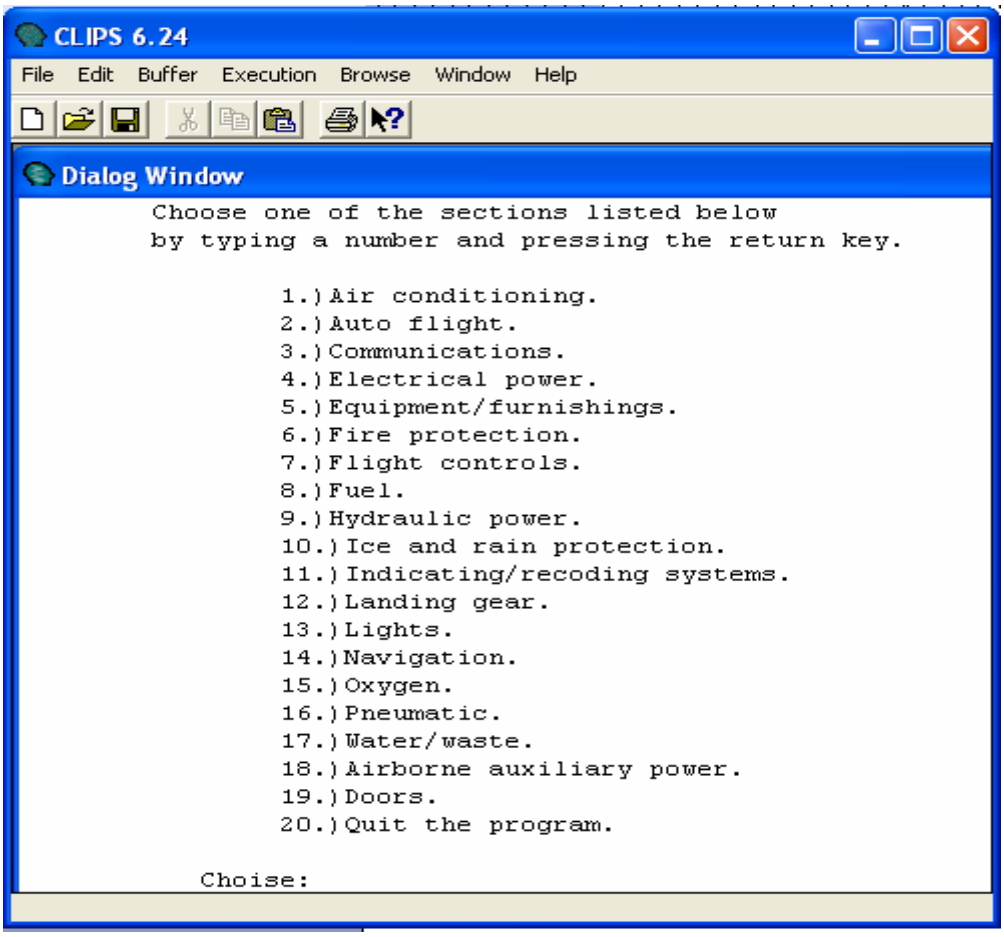
Αφού δώσουμε τον τριψήφιο αριθμό του αεροσκάφους το σύστημα προχωράει στις επόμενες δύο ερωτήσεις.

Please give the current date that fault occurred:

Please give the flight number:

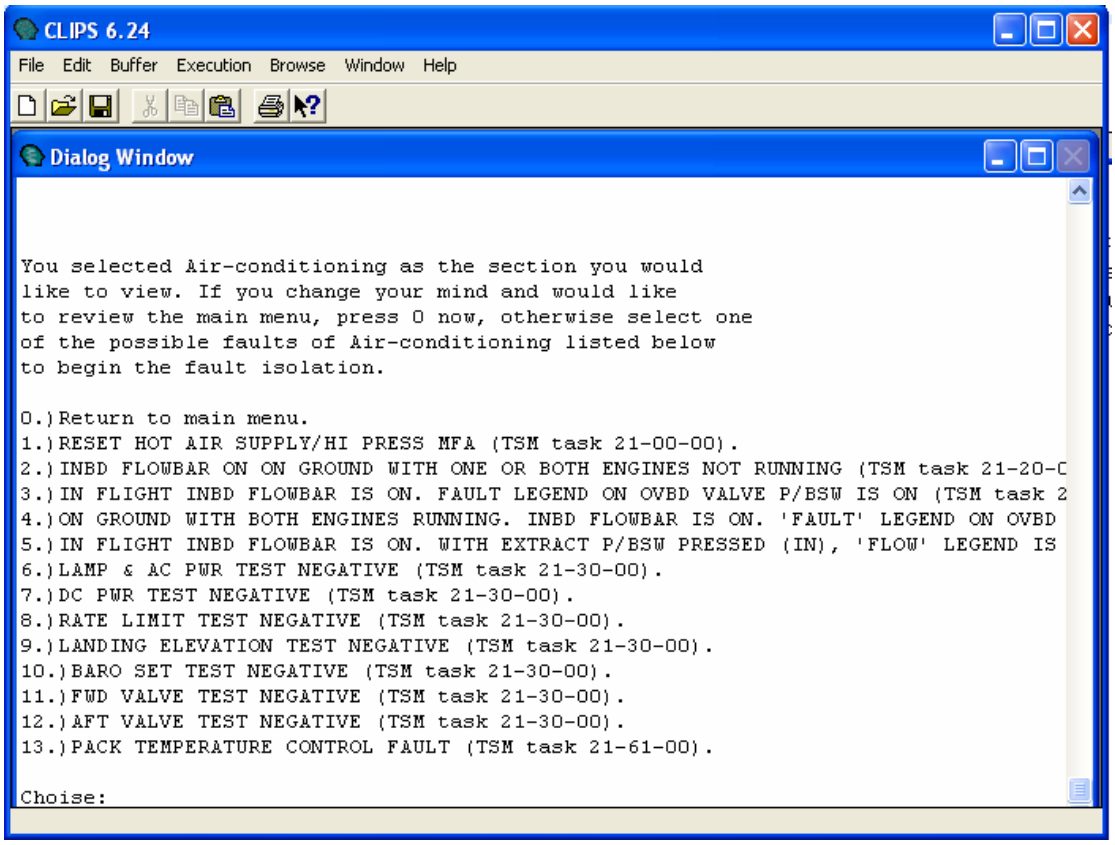
Οι ερωτήσεις αυτές γίνονται από το σύστημα, ώστε να δημιουργείται ένα ιστορικό βλαβών βάση του αριθμού του αεροσκάφους, της ημερομηνίας που εμφανίστηκε η βλάβη και τον αριθμό πτήσης. Στην περίπτωση που ο χρήστης δώσει λάθος απάντηση το σύστημα ρωτάει ξανά μέχρι να πάρει σωστή απάντηση.

Το βήμα τώρα που ακολουθεί είναι και το πιο σημαντικό. Με το πέρας των ερωτοαπαντήσεων, εμφανίζεται ένα μενού με όλους τους τομείς του αεροσκάφους. Ο χρήστης δεν έχει παρά να επιλέξει τον τομέα που τον ενδιαφέρει για να δει τις βλάβες που περιέχονται όπως δείχνει το παρακάτω σχήμα 5.3.



Σχήμα 5.3. Μενού Επιλογής Τομέα

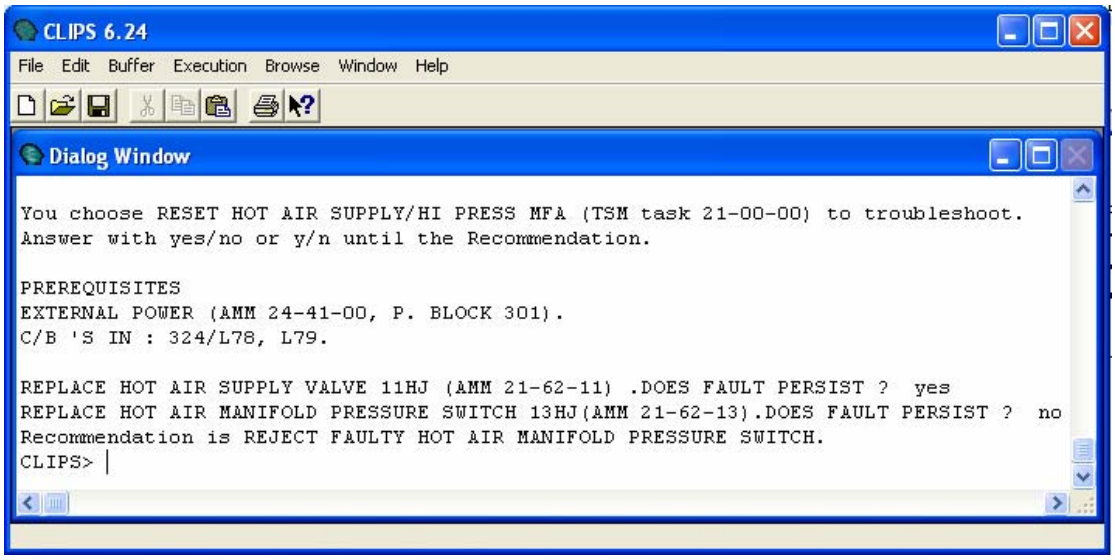
Επιλέγοντας έναν αριθμό από το 1 έως το 19, εμφανίζεται το επόμενο μενού με το περιεχόμενο όλων των βλαβών για τον τομέα που επέλεξε. Στην περίπτωση που επιλέξει το 20 το πρόγραμμα τερματίζεται. Έστω ότι επιλέγουμε 1 δηλαδή τον τομέα AIR CONDITIONING, τότε εμφανίζεται το υπομενού του τομέα :



Σχήμα 5.4. Μενού Επιλογής Βλάβης

Στο παράθυρο αυτό βλέπουμε το εισαγωγικό μήνυμα το οποίο μας πληροφορεί ότι επιλέξαμε τον τομέα AIR CONDITIONING και τις πιθανές βλάβες του. Με την επιλογή ενός αριθμού οδηγούμαστε στο επόμενο βήμα, αυτό της επισκευής της επιλεχθείσας βλάβης. Με την επιλογή 0 οδηγούμαστε πάλι στο αρχικό μενού.

Αφού επιλέξουμε την βλάβη που μας ενδιαφέρει για να προβούμε στην επισκευή της, εμφανίζεται το παράθυρο του παρακάτω σχήματος :



Σχήμα 5.5. Διαδικασία Επιδιόρθωσης Βλάβης

Το σημείο αυτό είναι το τελικό στάδιο του προγράμματος, όπου το σύστημα μας προτείνει πώς να αντιμετωπίσουμε την βλάβη για την αποκατάσταση της. Να τονίσουμε βέβαια ότι για την αντιμετώπιση της κάθε βλάβης, απαιτείται από τον τεχνικό που εργάζεται για την αποκατάσταση της να πάρει τα απαραίτητα μέτρα προφύλαξης (*PREREQUISITES*) για αποφυγή ατυχήματος του ίδιου, αλλά και προς αποφυγή επέκτασης της βλάβης σε διπλανά υποσυστήματα.

ΚΕΦΑΛΑΙΟ 6

Συμπεράσματα αξιολόγησης

6.1 Συμπεράσματα

Έπειτα λοιπόν από την αξιολόγηση των φάσεων της διαδικασίας συντήρησης και της διάγνωσης και επιδιόρθωσης βλαβών και λαμβάνοντας υπόψη τόσο το εργασιακό περιβάλλον όσο και τις ιδιαίτερες συνθήκες και τις ποικίλες παραμέτρους που επηρεάζουν την όλη διαδικασία, προκειμένου να είναι σε όσο το δυνατόν μεγαλύτερο βαθμό πληρέστερη η παρούσα έρευνα, θα πρέπει να αναφερθούν οι ακόλουθες διαπιστώσεις:

1. Η διαδικασία διάγνωσης και επιδιόρθωσης βλαβών του συγκεκριμένου τύπου αεροσκάφους είναι μια πολύπλοκη διαδικασία, με πολλές παραμέτρους και αρκετούς αστάθμητους παράγοντες, που επηρεάζουν το χρόνο ολοκλήρωσης της διαδικασίας σε μεγάλο βαθμό.
2. Ο πιο σημαντικός ίσως από τους αστάθμητους παράγοντες είναι η διαφορετικότητα και η μοναδικότητα του κάθε αεροσκάφους, που έγκειται όχι μόνο στην παλαιότητα του, αλλά και στις συνθήκες κάτω από τις οποίες έχει πετάξει μέχρι να πραγματοποιήσει την προγραμματισμένη συντήρηση.
3. Τα ευρήματα που παρουσιάζει κάθε αεροσκάφος κατά την επιθεώρηση του ή και κατά την προγραμματισμένη συντήρηση του είναι διαφορετικά, ανάλογα με την πληθώρα αλλά κυρίως ανάλογα με τη σοβαρότητα των ευρημάτων, επηρεάζεται αρχικά ο χρόνος αποκατάστασης τους, σε δεύτερο όμως επίπεδο η χρονική διάρκεια όλης της διαδικασίας διάγνωσης και επιδιόρθωσης βλαβών.
4. Η έλλειψη ολοκληρωμένων προγραμμάτων λογισμικού διάγνωσης και επιδιόρθωσης βλαβών καθιστά πιο χρονοβόρο το έργο των τεχνικών και μηχανικών του τομέα αυτού, με αρκετά αρνητικά επακόλουθα στην λειτουργία μιας αεροπορικής επιχείρησης.
5. Ο δυναμικός χαρακτήρας του αριθμού των εργαζομένων που απασχολούνται στη διαδικασία της συντήρησης και της διάγνωσης και επιδιόρθωσης βλαβών είναι ακόμα μια παράμετρος που επηρεάζει τη συνολική διάρκεια της

διαδικασίας. Η μεταβλητότητα του αριθμού εργαζομένων στη συγκεκριμένη διαδικασία επηρεάζει τη χρονική διάρκεια της διάγνωσης και επιδιόρθωσης, ανάλογα με το εάν αυξάνεται ή μειώνεται το τεχνικό δυναμικό.

6.2 Μελλοντική εργασία

Το σύστημα που υλοποιήθηκε στην παρούσα εργασία αποτελεί μια πρώτη προσπάθεια ενός ολοκληρωμένου συστήματος διάγνωσης και επιδιόρθωσης βλαβών.

Μελλοντικό σχέδιο είναι η επιθυμία κατασκευής ενός έμπειρου συστήματος διάγνωσης και επιδιόρθωσης βλαβών το οποίο θα περιλαμβάνει στην βάση γνώσης του όλα τα εγχειρίδια του αεροπορικού κατασκευαστή όπως είναι τα Aircraft Maintenance Manual (AMM), Troubleshooting Manual (TSM), Aircraft Wiring Manual (AWM) αλλά και όλο το ιστορικό βλαβών της κάθε εταιρείας που θα ενδιαφερθεί να το εντάξει στο σταθμό εργασίας συντήρησης αεροσκαφών της.

Το έμπειρο σύστημα θα μπορεί να το δουλεύει ο κάθε τεχνικός μέσω ενός μικρού υπολογιστή χειρός (PDA), το οποίο θα είναι συνδεδεμένο με τα ανάλογα τμήματα συντήρησης και επιδιόρθωσης βλαβών της εταιρείας με σκοπό να αποφευχθούν διαδικασίες όπως είναι αυτή της διακίνησης εγγράφων μέσα σε μια αεροπορική εταιρεία ή της αναζήτησης στα εγκεκριμένα εγχειρίδια.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Alexander, S.M. G.W. Evans (1988), The integration of multiple experts: A review of methodologies, in: E. Turban and P. Watkins (eds.), Applied Expert Systems North-Holland.

Airbus Industries, "*Troubleshooting Manual A300-600*", Rev. Jun 01/04

Air Transport Association of America (1986), "ATA *Specification for Manufacturers' Technical Data – ATA Specification No. 100* "

Boose, J.H. (1986a), Expertise transfer for expert systems design, Elsevier, New York.

Boose, J.H. (1986b), ETS: A system for the transfer of human expertise, in: J. Kowalik (ed.), Knowledge based problem solving, Englewood Cliffs, NJ: Prentice Hall.

Boose, J.H. (1986c), Rapid acquisition and combination of knowledge from multiple experts in the same domain, Future Computer Systems, vol. 1, pp. 191-216.

Boose, J.H. (1986d), A survey of knowledge acquisition tools and techniques, Knowledge Aqcuisition, vol. 1, no. 1, pp. 3-37.

Brute, J.F., A. Blount (1989), Knowledge acquisition, McGraw-Hill, New York.

CLIPS: A Tool for Building Expert Systems, http://www.ghg.net/clips/CLIPS.html

Curry, B., L. Moutinhon (1994), Intelligent Computer Models for Marketing Decisions, Management Decision, Vol. 32, No. 4, pp. 30-35.

Dos Santos, B.L., Z.T.S. Mookerjee (1993), Expert system design: Minimizing information acquisition costs, Decision Support Systems, no. 9, pp. 161-181.

Doukidis, G.I., E. Whitley (1988), Developing expert systems, Chartwell-Bratt.

Durkin J. (1994), EXPERT SYSTEMS: Design and Development, Macmillan Publishing Company.

Feigenbaum, E. (1982), Knowledge engineering for the 1980's, Department of Computer Science, Stanford University, California.

Feigenbaum, E., P. McCorduck (1983), The fifth generation, Reading, MA: Addison-Wesley.

Fikes, R., T. Kehler (1985), The role of frame-based representation in reasoning, Communication of ACM, September.

Gaines, B.R. (1985), Knowledge acquisition: Developments and advances, in M.D. Oliif (ed.), Expert systems and intelligent manufacturing, Elsevier Science, New York.

Giarratano J.C., Riley G. D. (2005), "EXPERT *SYSTEMS: Principles and Programming*", Fourth Edition, Thomson Course Technology

Goker M. H., Kuru S. (1990), THE TSE_310 TROUBLESHOOTING EXPERT PROTOTYPE FOR THE AIRBUS A-310 COMMERCIAL AIRCRAFT, pages: 692-696, Proceedings of the 3rd international conference on industrial and engineering of artificial intelligence and expert systems – Vol. 2

Goodman, I.R., M.M. Gupta, H.T. Ngueyen, G.S. Rogers (1991), Conditional logic in expert systems, North-Holland.

Hayes-Roth, F., D. Waterman, D. Lenat (1983), Building expert systems, Reading, MA: Addison-Wesley.

Hayes-Roth, F. (1984), The knowledge-based expert system: A tutorial IEEE Computer, September, pp. 11-28.

Jackson P. (1999), "*Introduction to EXPERT SYSTEMS*", Addison Wesley Longman Limited

Jayaraman, V., R. Srivastava (1996), Expert systems in production and operations management – Current applications and future prospects, International Journal of Operations & Production Management, vol. 16, No. 12, pp.27-44.

Johnson, L., N.E. Johnson (1987), A knowledge elicitation method for expert systems design, System research and info. Science, vol. 2, pp. 153-166.

LeClair, S.R. (1985), A multi-expert knowledge system architecture for manufacturing decision analysis, Unpublished Ph.D. Dissertation, Arizona State University.

Marcot, B. (1987), Testing your knowledge base, AI Expert, August.

McGovern, J., D. Samson, A. Wirth (1991), Knowledge acquisition for intelligent decision systems, Decision Support Systems no. 7,pp. 263-272.

McGraw, K.L., B.K. Harbison (1989), Knowledge acquisition, priciples and guidelines, Englewood Cliffs, NJ: Prentice-Hall.

Mittal, S. (1985), Knowledge acquisition from multiple experts, AI Magazine, Summer.

Newell, A., H.A. Simon (1958), Element of a theory of human problem solving, Psychological Review, no. 65, pp. 151-166.

Newell, A.,H.A. Simon (1972), Human problem solving, NJ: Prentice-Hall, Englewood.

Ngyuen, T.A. et al. (1987), Knowledge base verification, AI Magazine, Summer.

Nii, P. (1986), Blackboard systems: The blackboard model of problem solving and the evolution of blackboard architectures AI Magazine, vol. 7 no. 2.

O'Keefe, R.M., O. Balci, E.P. Smith (1987), Validation expert performance, IEEE Expert, Winter.

Shaw, M.L.G., B.R. Games (1986), Interactive elicitation of knowledge from experts, Future Computing Systems, vol. 1, no. 2, pp. 151-190.

Turban , E. (1993), Decision support and expert systems: Management support systems, 3rd ed., Macmillan, New York.

Walker, A. (1996), Knowledge Systems: Principles and practice, McGraw-Hill.

Winston, P.H., S.a. Shellard (eds.) (1990), Artificial intelligence at MIT expanding frontiers, vol. 1and 2, MIT Press, Cambridge.

Βλαχάβας Ι., Κεφαλάς Π., Βασιλειάδης Ν., Ρεφανίδης Ι., Κόκκορας Φ., Σακελλαρίου Η. (2002), "ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ" Εκδόσεις Γαρταγάνη, Θεσσαλονίκη

Ματσατσίνης Ν. Φ. (2004), Μηχανική Γνώσεων και Αποφάσεων, Πανεπιστημιακές Σημειώσεις, Τμήμα Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης.