ΕΜΠΕΙΡΑ ΣΥΣΤΗΜΑΤΑ (EXPERT SYSTEMS)

ή

ΣΥΣΤΗΜΑΤΑ ΕΜΠΕΙΡΟΓΝΩΜΟΣΥΝΗΣ

1. Εισαγωγή

Η αδιάκοπη απόκτηση τεχνολογικής γνώσης αναπόφευκτα οδηγεί στην εξειδίκευση και στην ανάγκη του "ειδικού-εμπειρογνώμονα". Τα τελευταία χρόνια η ανάπτυξη του κλάδου των Η/Υ που ονομάζεται "Τεχνητή Νοημοσύνη", έκανε δυνατή την αποτελεσματική υλοποίηση των **Εμπείρων Συστημάτων** (ή **Συστημάτων Εμπειρογνωμοσύνης**). Τα Εμπειρα Συστήματα (**Ε.Σ.**) είναι συστήματα προγραμμάτων, που έχουν τη δυνατότητα να προσομοιώνουν τη γνώση και τον τρόπο σκέψης ενός εμπειρογνώμονα σε κάποιο εξειδικευμένο πεδίο εφαρμογής (όπως είναι η ιατρική διάγνωση, η επισκευή πολύπλοκων συσκευών, η στήριξη οικονομικών αποφάσεων κ.α.).

Η εξέλιξη των Ε.Σ. οδήγησε στα **Συστήματα που Βασίζονται στη Γνώση** (**Knowledge Based Systems**) ή απλά **Συστήματα Γνώσης**. Πρόκειται για συστήματα προγραμμάτων τα οποία λειτουργούν σαν ολοκληρωμένα εργαλεία που παρέχουν στο χρήστη "έξυπνη" βοήθεια για την επίλυση σύνθετων προβλημάτων.

Σήμερα, ο συνδυασμός της τεχνολογίας των Εμπείρων Συστημάτων - Συστημάτων Γνώσης με τις τεχνολογίες των Βάσεων Δεδομένων και των Οντοκεντρικών Συστημάτων (Object Oriented Systems) αποτελεί ίσως το πιο σύγχρονο εργαλείο για την ανάπτυξη Πληροφοριακών Συστημάτων υψηλών απαιτήσεων.

2. Ειδικά Χαρακτηριστικά των Ε.Σ.

Εκτός από τον αφηρημένο ορισμό που δόθηκε παραπάνω για τα Ε.Σ. υπάρχει ένα σύνολο ειδικών χαρακτηριστικών τα οποία πρέπει να διαθέτει ένα Εμπειρο Σύστημα, για να ξεχωρίζει από ένα κλασσικό σύστημα λογισμικού:

- Ενα Εμπειρο σύστημα πρέπει να διαθέτει ένα μηχανισμό για την άμεση αναπαράσταση της γνώσης που σχετίζεται με το πεδίο εφαρμογής.
  Η γνώση αυτή αποτελείται από όλες τις σχετικές πληροφορίες που χρειάζονται στον εμπειρογνώμονα (άνθρωπο) για να διεκπεραιώσει την αντίστοιχη διαδικασία επίλυσης του προβλήματος και αναφέρεται σα **βάση γνώσης** (***Knowledge base***).

- Ενα Εμπειρο Σύστημα πρέπει να διαθέτει ένα μηχανισμό συλλογιστικής για την παραγωγή νέας γνώσης από την ήδη υπάρχουσα στη βάση γνώσης.
  Αυτός ο μηχανισμός ονομάζεται **μηχανή επαγωγής** (***inference engine***) και είναι υπεύθυνος για την παραγωγή συμπερασμάτων και την παροχή συμβουλών, στο χρήστη του Ε.Σ.

- Ενα Εμπειρο Σύστημα πρέπει να έχει τη δυνατότητα επεξήγησης των συλλογισμών του.
  Αυτό είναι απαραίτητο, για να βοηθήσει το χρήστη του Ε.Σ. να καταλάβει (και να πιστέψει) τους λόγους για τις αποφάσεις που πάρθηκαν κατά τη διαδικασία επίλυσης του προβλήματος, χωρίς να χρειάζεται να καταλάβει τις λεπτομέρειες της διαδικασίας επαγωγής συμπερασμάτων.

Υπάρχουν δύο βασικές μεθοδολογίες που έχουν ακολουθηθεί από τους ερευνητές για την ανάπτυξη Εμπείρων Συστημάτων. Η πρώτη μεθοδολογία βασίζεται στην υπόθεση ότι μπορούν να βρεθούν γενικές μέθοδοι επίλυσης προβλημάτων εμπειρογνωμοσύνης και στη συνέχεια να εφαρμοστούν σε διαφορετικά πεδία εφαρμογών. Η δεύτερη μεθοδολογία βασίζεται στην υπόθεση ότι ειδικά προβλήματα χρειάζονται ειδικές λύσεις. Αυτό σημαίνει ότι ένα Ε.Σ. για ένα νέο πεδίο εφαρμογής πρέπει να προγραμματιστεί από μηδενική

βάση. Η πρώτη μεθοδολογία είναι καλύτερη από θεωρητική άποψη, αλλά η δεύτερη ήταν αυτή που οδήγησε στα πρώτα πρακτικά Ε.Σ.

Αργότερα η σύνθεση των δύο παραπάνω μεθοδολογιών είχε σαν αποτέλεσμα την ανάπτυξη των λεγόμενων **Κελύφων Εμπειρων Συστημάτων** (***Expert System Shells***) Ενα τέτοιο κέλυφος αποτελείται από μία μηχανή επαγωγής και διαθέτει ένα γενικό σχήμα για την αναπαράσταση της γνώσης, που μπορούν να χρησιμοποιηθουν σαν βάση για την ανάπτυξη Ε.Σ. για συγγενικά πεδία εφαρμογών.

3. Εφαρμογές των Εμπείρων Συστημάτων

Εμπειρα Συστήματα έχουν ήδη αναπτυχθεί, για την επίλυση ενός μεγάλου εύρους προβλημάτων, σε πεδία όπως είναι η ιατρική, η μηχανολογία, η επιστήμη των υπολογιστών, τα μαθηματικά, η χημεία, η γεωλογία, η διοίκηση επιχειρήσεων, η νομική, η εκπαίδευση και η στρατιωτική άμυνα (επίθεση). Οι βασικότερες κατηγορίες προβλημάτων που αντιμετωπίστηκαν από τα συστήματα αυτά είναι οι παρακάτω:

- **Διάγνωση** (***Diagnosis***)**:**

Ο καθορισμός της αιτίας κάποιας δυσλειτουργίας, σε σύνθετες καταστάσεις, με βάση κάποια παρατηρούμενα συμπτώματα. Πολλά Εμπειρα Συστήματα έχουν αναπτυχθεί για διάγνωση βλαβών σε μηχανές, σε ηλεκτρονικές συσκευές. Επίσης ιδιαίτερη έμφαση έχει δοθεί στην ανάπτυξη Ε.Σ για τη διάγνωση ασθενειών.

- **Πρόγνωση** (***Prediction***)**:**

Η πρόβλεψη πιθανών αποτελεσμάτων, με βάση κάποια γνωστά δεδομένα Ε.Σ. έχουν αναπτυχθεί για την πρόγνωση καιρού, την πρόβλεψη πωλήσεων κ.λπ.

• **Σχεδιασμός (*Design*):**

Η διαμόρφωση των τμημάτων ενός συστήματος, ώστε αυτό με βάση κάποιους περιορισμούς, να τηρεί κάποιες προδιαγραφές. Παραδείγματα εφαρμογών είναι η σχεδίαση VLSI κυλωμάτων και ο σχεδιασμός συστημάτων Η/Υ.

• **Διερμήνευση (*Interpretation*):**

Η εξαγωγή συμπερασμάτων ή η δημιουργία περιγραφών υψηλού επιπέδου με βάση ένα σύνολο δεδομένων (συνήθως ασαφών). Τυπικές εφαρμογές αυτής της κατηγορίας είναι τα συστήματα αναγνώρισης φωνής και εικόνας

• **Κατάστρωση Πλάνων (*Planning*):**

Ο σχεδιασμός μιάς σειράς ενεργειών, οι οποίες αν ακολουθηθουν και με δεδομένες κάποιες αρχικές προϋποθέσεις, θα μπορέσουν να εκπληρώσουν κάποιο σκοπό.

• **Παρακολούθηση (*Monitoring*):**

Η σύγκριση της παρατηρούμενης με την αναμενόμενη συμπεριφορά ενός συστήματος.

• **Εκσφαλμάτωση (*Debugging*):**

Ο καθορισμός τρόπων για τη διόρθωση λαθών που οδηγούν στην κακή συμπεριφορά ενός συστήματος.

• **Επισκευή (*Repair*):**

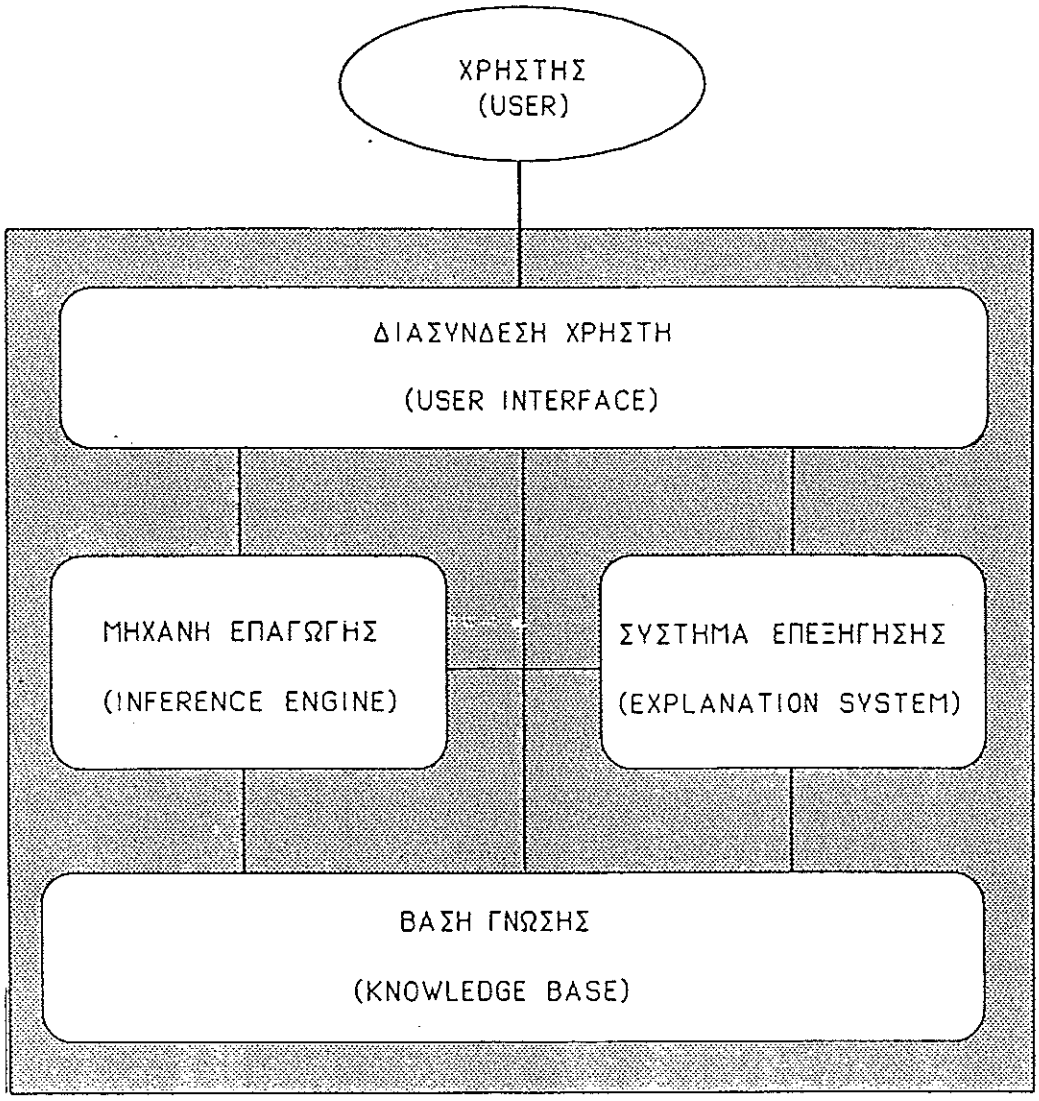
Η εκτέλεση ενός πλάνου για την διόρθωση κάποιων λαθών, που έχουν εντοπιστεί σ' ένα σύστημα.

• **Εκπαίδευση (*Instruction*):**

Σχετίζεται με την παροχή βοήθειας στην διαδικασία κατανόησης εννοιών ενός συγκεκριμένου αντικειμένου, από μαθητευόμενους.

4. Η Αρχιτεκτονική των Εμπειρων Συστημάτων

Εξ' αιτίας του μεγάλου εύρους των εφαρμογών και των διαφόρων τεχνικών που χρησιμοποιούνται για την κατασκευή των Εμπείρων Συστημάτων δεν μπορεί να υποστηριχθεί ότι υπάρχει κάποια "τυποποιημένη αρχιτεκτονική" η οποία ακολουθείται για το το σχεδιασμό τους. Παρόλα αυτά τα βασικά τμήματα που φαίνονται στο σχήμα 1 συναντώνται στα περισσότερα Εμπειρα Συστήματα.



Σχ. 1 : Η αρχιτεκτονική ενός τυπικού Ε.Σ.

- **Ελεγχος (*Control*):**

Η ρύθμιση της συμπεριφοράς ενός σύνθετου συστήματος

- **Εξυπνη Βοήθεια (*Intelligent Assistance*):**

Συστήματα τα οποία, ανάλογα με τις ανάγκες του χρήστη, μπορούν να παρέχουν ειδικές συμβουλές, να δίνουν πληροφορίες και να εκτελούν διάφορες επί μέρους εργασίες.

4.1 Η Βάση Γνώσης (*Knowledge Base*)

Η Βάση Γνώσης είναι η καρδιά του Ε.Σ., και περιλαμβάνει όλες τις πληροφορίες που σχετίζονται με την γνώση επίλυσης των προβλημάτων του συγκεκριμένου πεδίου εφαρμογής. Σε πολλά Ε.Σ. η βάση γνώσης χωρίζεται με τη σειρά της σε δύο υποτμήματα. Το πρώτο περιλαμβάνει ειδικά δεδομένα που περιγράφουν αντικείμενα, περιστατικά και καταστάσεις, όπως για παράδειγμα:

"Η τελική ταχύτητα του Χ αυτοκινήτου είναι 160Km"

Το δεύτερο περιλαμβάνει πιό γενικές αρχές, και ευρετικούς κανόνες (*heurestic rules* ή *rules of thumb*) επίλυσης προβλημάτων οι οποίοι προέρχονται από συσσωρευμένες εμπειρικές διαπιστώσεις του εμπειρογνώμονα ή/και από τεχνολογικά δεδομένα του πεδίου εφαρμογής, όπως για παράδειγμα:

"Σε περίπτωση που το Χ αυτοκίνητο έχει διανύσει περισσότερα από 80,000 Km και ξεκινά με δυσκολία σε ανηφορικούς δρόμους τότε χρειάζεται αλλαγή δίσκου συμπλέκτη"

Ο τρόπος αναπαράστασης της βάσης γνώσης, θα πρέπει να είναι τέτοιος, ώστε να διευκολύνει:

(α) την πρόσληψη της γνώσης από τον ειδικό-εμπειρογνώμονα
(β) την ανάκτηση της γνώσης αυτής από το Ε.Σ.
(γ) τη διαδικασία της συλλογιστικής του Ε.Σ.

Γιά τους λόγους αυτούς, η βάση γνώσης, δεν πρέπει να αναπαριστάνεται έμμεσα, με τη χρήση των δομών δεδομένων που χρησιμοποιούν οι καθιερωμένες γλώσσες προγραμματισμού, αλλά με τη χρήση ενός **δηλωτικού** φορμαλισμού, ο οποίος πρέπει να είναι σε θέση να μοντελοποιήσει τις έννοιες του αντίστοιχου πεδίου εφαρμογής με τον αμεσότερο δυνατό τρόπο.

Υπάρχει ένας αρκετά μεγάλος αριθμός τεχνικών αναπαράστασης γνώσης οι κυριότερες από τις οποίες είναι:

- οι κανόνες παραγωγής (*production rules*)
- η μαθηματική λογική *(mathematical logic)*
- τα σημασιολογικά δίκτυα (*semantic networks*)
- τα πλαίσια (*farmes*)

Τα πιό γνώστα Ε.Σ. χρησιμοποιούν σα μέθοδο αναπαράστασης τους κανόνες παραγωγής. Τα συστήματα αυτά αναφέρονται συνήθως απλά με τον όρο **Ε.Σ. βασιζόμενα σε κανόνες (*rule-based E.S.*)**. Η μεγάλη εξάπλωση των συστημάτων αυτών οφείλεται κύρια στο **MYCIN**, ένα Ε.Σ. για τη διάγνωση ασθενειών του αίματος. Ενας κανόνας παραγωγής έχει τη γενική μορφή:

if <προϋπόθεση(εις)> **then** <ενέργεια(ες)>

και αποτελείται από δύο βασικά τμήματα:

- το τμήμα **if** το οποίο περιγράφει τις προυποθέσεις ή καταστάσεις ή παρατηρήσεις ή συνθήκες του κανόνα που πρέπει να είναι γνωστές

- το τμήμα **then** το οποίο πειγράφει ποιές είναι ενέργειες που πρέπει να ακολουθηθούν, ή τα συμπεράσματα που προκύπτουν, στην περίπτωση που ισχύουν οι προυποθέσεις.

Συχνά οι κανόνες παραγωγής μπορούν να περιέχουν παράγοντες αβεβαιότητας είτε στο τμήμα **if** είτε στο τμήμα **then**., όπως για παράδειγμα:

if <πιθανότητα βροχής >= 50%> **then** <πάρε αδιάβροχο>

if <η μηχανή δεν γυρίζει> **and** <τα φώτα δεν ανάβουν>
then <πρόβλημα με τη μπαταρία> : 80%

Στο σχήμα 2 δίνεται ένα μικρό συνόλο κανόνων εμπειρίας γιά τον καθορισμό της βλαβής ενός αυτοκινήτου σε περίπτωση που δεν ξεκινά.

⇨ Κανόνας 1

if <η μηχανή τροφοδοτείται με βενζίνη> **and**
<η μηχανή γυρίζει>
then <πρόβλημα με τα μπουζί>

⇨ Κανόνας 2

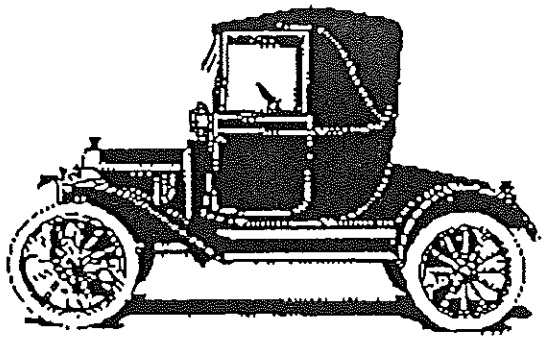
if <η μηχανή δεν γυρίζει> **and**
<τα φώτα δεν ανάβουν>
then <πρόβλημα με τη μπαταρία> **or**
<πρόβλημα με τα καλώδια>

⇨ Κανόνας 3

if <η μηχανή δεν γυρίζει> **and**
<τα φώτα ανάβουν>
then <πρόβλημα με τη μίζα>

⇨ Κανόνας 4

if <υπάρχει βενζίνη στο ντεπόζιτο> **and**
<υπάρχει βενζίνη στο καρμπυρατέρ>
then <η μηχανή τροφοδοτείται με βενζίνη>



Σχ. 2 : κανόνες παραγωγής για το πρόβλημα εκίνησης αυτοκινήτου

4.2 Η Μηχανή Επαγωγής Συμπερασμάτων (*Inference Engine*)

Η απλή πρόσβαση σε ένα μεγάλο όγκο πληροφορίας που σχετίζεται με κάποιο πεδίο εφαρμογής δεν είναι ικανή για να καταστήσει κάποιον ειδικό-εμπειρογνώμονα. Απαιτείται και κάποια μεθοδολογία για την επιλογή και την εφαρμογή της γνώσης αυτής σε κάποιο συγκεκριμένο πρόβλημα. Το ρόλο αυτό σ' ένα Ε.Σ. παίζει η μηχανή επαγωγής.

Η μηχανή επαγωγής αποφασίζει ποιά ευρετική διαδικασία αναζήτησης θα χρησιμοποιηθεί πάνω στην πληροφορία της βάσης γνώσης, για την παραγωγή κάποιου συμπεράσματος, που απαιτείται για την επίλυση κάποιου προβλήματος. Συνήθως η μηχανή επαγωγής αποτελείται από δύο επί μέρους τμήματα:

- **το διερμηνέα (*interpreter*)** ο οποίος χρησιμοποιείται για να υλοποιήσει κάποια (ή κάποιες) συγκεκριμένη τεχνική επαγωγής συμπεράσματος (*inference technique*)

- **τον επιλογέα (*dispatcher*)** ο οποίος υλοποιεί κάποια στρατηγική που καθορίζει πότε και με ποιά σειρά θα χρησιμοποιηθούν οι πληροφορίες της βάσης γνώσης.

Η τεχνική επαγωγής συμπεράσματος εξαρτάται άμεσα από τον τρόπο αναπαράστασης της γνώσης. Οι πιό γνωστές τεχνικές επαγωγής συμπεράσματος (που σχετίζονται κυρίως με τους κανόνες παραγωγής) είναι:

- η **ορθή συλλογιστική (*forward reasoning*)** και
- η **ανάστροφη συλλογιστική (*backward reasoning*)**

Η ορθή συλλογιστική ξεκινά από ήδη υπάρχουσα γνώση και εφαρμόζοντας κάποιο κανόνα παράγει καινούργια γνώση. Η παραπάνω διαδικασία εφαρμόζεται επαναληπτικά, έως ότου παραχθεί το ζητούμενο αποτέλεσμα. Σε κάθε βήμα, η εφαρμογή κάποιου κανόνα γίνεται με το ταίριασμα του **if**-τμήματος του κανόνα με κάποιο γνωστό γεγονός (ή γεγονότα). Το

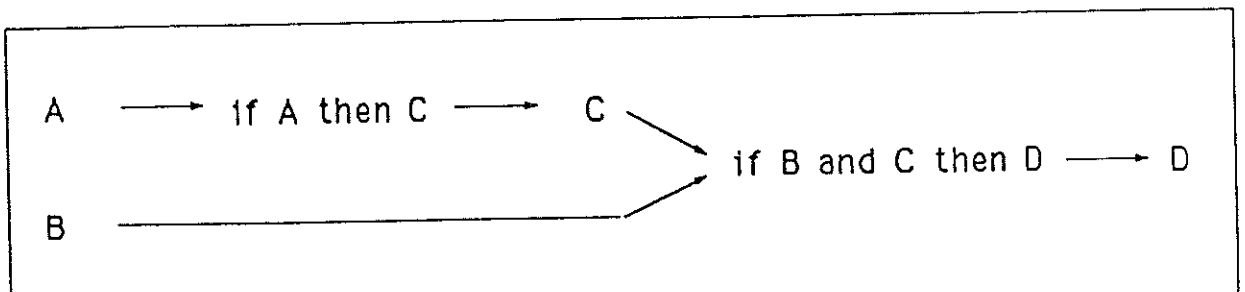
συμπεράσμα που παράγεται από το **then**-τμήμα του κανόνα, προστίθεται σαν νέο γεγονός στη βάση γνώσης.

Η ανάστροφη συλλογιστική ξεκινά από το ζητούμενο αποτέλεσμα και, εφαρμόζοντας κάποιο κανόνα, παράγει ένα νέο ζητούμενο. Η παραπάνω διαδικασία εφαρμόζεται επαναληπτικά, έως ότου κάποιο νέο ζητούμενο αποτέλεσμα που παράγεται αποτελεί κάποιο γνωστό γεγονός της βάσης γνώσης. Σε κάθε βήμα, γίνεται ταίριασμα του ζητούμενου αποτελέσματος με το **then**-τμήμα κάποιου κανόνα και στη συνέχεια το **if**-τμήμα του συγκεκριμένου κανόνα αποτελεί το νέο ζητούμενο.

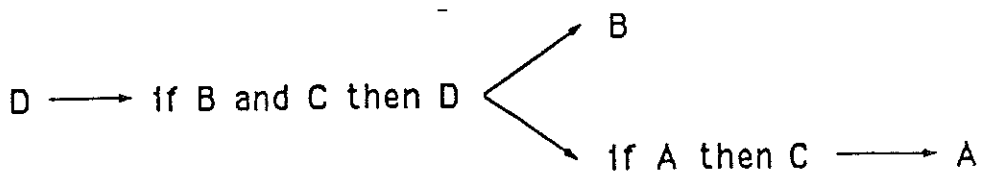
Εστω για παράδειγμα η παρακάτω βάση γνώσης (Τα γράμματα που είναι μόνα τους συμβολίζουν απλά γεγονότα):

A
B
if A then C
if B and C then D.

και έστω ότι ζητούμενο συμπέρασμα είναι το **D**. Τα σχήματα 10.3 και 10.4 δείχνουν τη διαδικασία επαγωγής συμπεράσματος, που βασίζεται αντίστοιχα, στην ορθή και την ανάστροφη συλλογιστική.



σχ. 3 : Διαδικασία συλλογισμού με βάση την Ορθή Συλλογιστική



σχ. 4 : Διαδικασία συλλογισμού με βάση την Ανάστροφη Συλλογιστική

Σε πολλές περιπτώσεις Ε.Σ. συναντάμε τεχνικής επαγωγής που βασίζονται στο συνδυασμό της ορθής με την ανάστροφη συλλογιστική.

Παρακάτω δίνονται δύο "μετα-διερμηνείς" γραμμένοι σε PROLOG οι οποίοι υλοποιούν την ορθή και ανάστροφη συλλογιστική:

Μετα-διερμηνέας για Ορθή συλλογιστική:

```
:- op(800, fx, if).
:- op(700, xfx, then).
:- op(300, xfy, or).
:- op(200, xfy,and).

forward:-
     new_derived_fact(P), !,
     write('Παραγωγή του γεγονότος: '), write(P), nl,
     assert( fact(P)),
     forward.

forward:-
     write(' Δεν παράγονται άλλα γεγονότα').

new_derived_fact(Conclusion):-
     if Condition then Conclusion,
     not fact(Conclusion),
     composed_fact(Condition).
```

```
composed_fact(Condition):-
    fact(Condition).      % Απλό γεγονός

composed_fact(Condition1 and Condition2):-
    composed_fact(Condition1),
    composed_fact(Condition2).

composed_fact(Condition1 or Condition2):-
    composed_fact(Condition1)
    ;
    composed_fact(Condition2).
```

Μετα-διερμηνέας για Ανάστροφη συλλογιστική:

```
is_true(P):-
    fact(P).

is_true(P):-
    if Condition then P,
    is_true(Condition).

is_true(P1 and P2):-
    is_true(P1),
    is_true(P2).

is_true(P1 or P2):-
    is_true(P1)
    ;
    is_true(P2).
```

4.3 Η Διασύνδεση με το Χρήστη (*User Interface*)

Το τμήμα αυτό είναι υπεύθυνο για την επικοινωνία του χρήστη με το Ε.Σ. Υπάρχουν τουλάχιστον δύο είδη επικοινωνίας που πρέπει να υλοποιήση το τμήμα διασύνδεσης με το χρήστη:

- Το πρώτο αφορά στη διαδικασία **πρόσληψης της γνώσης (*knowledge aquisition*)** από τον ειδικό-εμπειρογνώμονα για την κατασκευή από τον προγραμματιστή της βάσης γνώσης. Για το σκοπό αυτό, στο τμήμα διασύνδεσης με το χρήστη πολλών Ε.Σ. περιλαμβάνεται ένας **εκδότης βάσης γνώσης (*knowledge base editor*)**. Ο εκδότης αυτός έχει τη δυνατότητα να ελέγξει το συντακτικό του κανόνα που εισάγεται, καθώς επίσης και τυχόν ασυνέπειες του κανόνα αυτού (*cosistency checking*) με τους ήδη υπάρχοντες στη βάση γνώσης. Επίσης έχει πρόσβαση στο τμήμα επεξήγησης των συλλογισμών του Ε.Σ. και μπορεί να βοηθήσει τον προγραμματιστή να εντοπίσει τυχόν λάθη στη συμπεριφορά του Ε.Σ.

- Το δεύτερο αφορά στη διαδικασία παροχής και ζήτησης πληροφοριών από τον τελικό χρήστη του Ε.Σ. κατά την επίλυση του συγκεκριμένου προβλήματος.

Πολλές φορές υπάρχει και ένα τρίτο υπο-τμήμα στο τμήμα διασύνδεσης που ασχολείται με την εκπαίδευση του χρήστη (*training subsystem*) στη χρήση του Ε.Σ.

Οι τεχνικές υλοποίησης του τμήματος διασύνδεσης με το χρήστη περιλαμβάνουν κλασικές διαδικασίες ερωτήσεων-απαντήσεων, χρήση μενού, γραφικά και πολύ συχνά κάποια μορφή επικοινωνίας σε φυσική γλώσσα (*natural language interface*).

4.4 Η Επεξήγηση των Συλλογισμών

Το τμήμα επεξήγησης των συλλογισμών του Ε.Σ. είναι υπεύθυνο για την παροχή εξηγήσεων στο χρήστη για τη διαδικασία που ακολουθείται στην επίλυση του προβλήματος. Οι εξηγήσεις αυτές αναφέρονται:

- στο **πώς** (***how queries***) το σύστημα έφτασε σε κάποιο συμπέρασμα.
- στο **γιατί** (***why queries***) το σύστημα ζητά κάποια πληροφορία από το χρήστη

Παρακάτω δίνεται ένας "μετα-διερμηνέας" γραμμένος σε PROLOG ο οποίος υλοποιεί το μηχανισμό **why**. (θεωρούμε ότι οι κανόνες **if-then** παριστάνονταται απλά με τη βοήθεια κανόνων της PROLOG.)

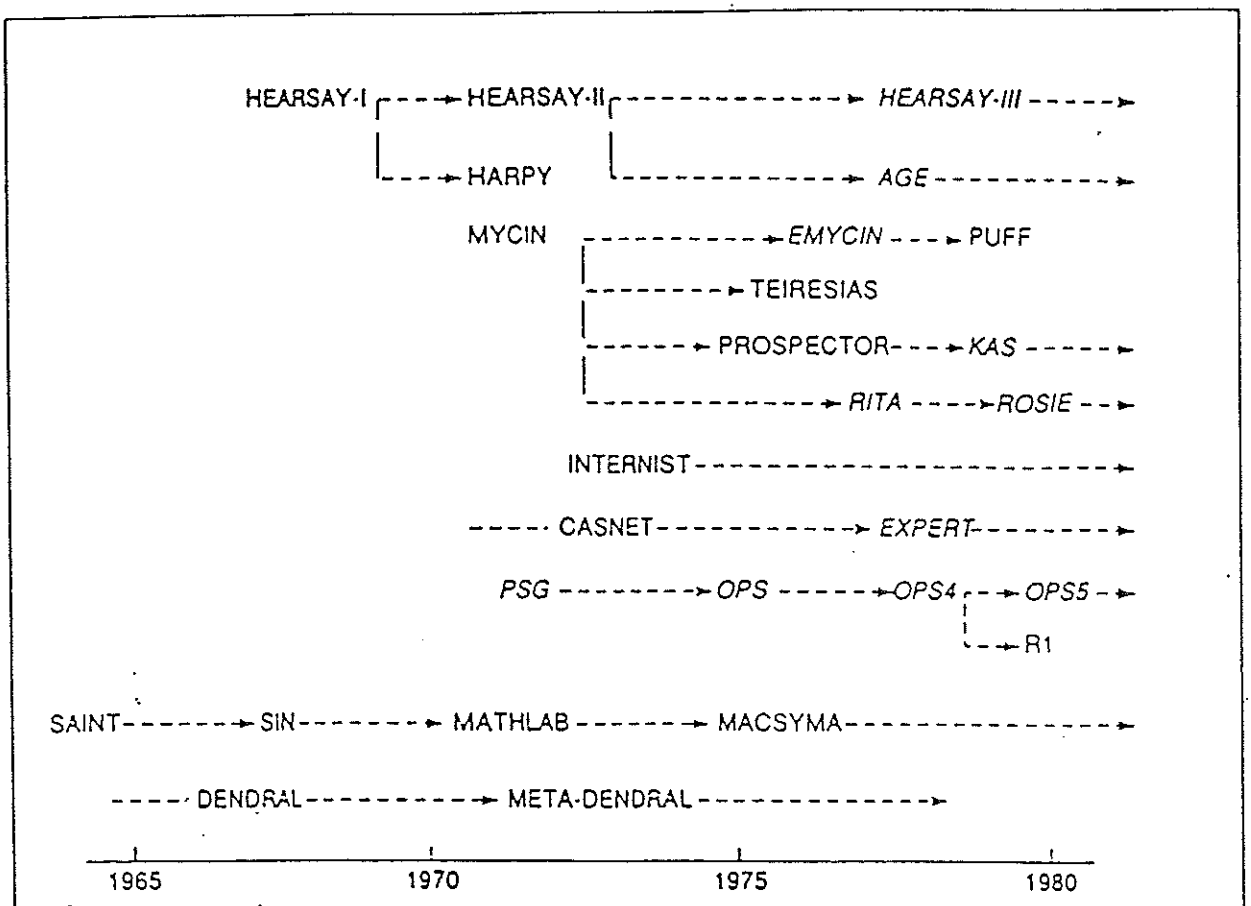
```
solve(true, _):- !.
solve(not A, Rules):-
     not solve(A, Rules).
solve((A,B), Rules):-
     solve(A, Rules),
     solve(B, Rules).
solve(A,Rules):-
     clause(A,B),
     solve(B, [(A:-B)|Rules]).
solve(A,Rules):-
     ask_user(A,Rules).

ask_user(A, Rules):-
     write(A),
     write(' Δώσε true αν η ερώτηση είναι αληθής - false άν όχι'),
     read(Answer),
     respond(Answer,A,Rules).
```

```
respond(true,_ ,_).
respond(why, A, [Rule|Rules]):-
      write(Rule),
      askuser(A,Rules).
respond(why,A, [ ]):-
      write('"No more Rules"),
      askuser(A,[ ]).
```

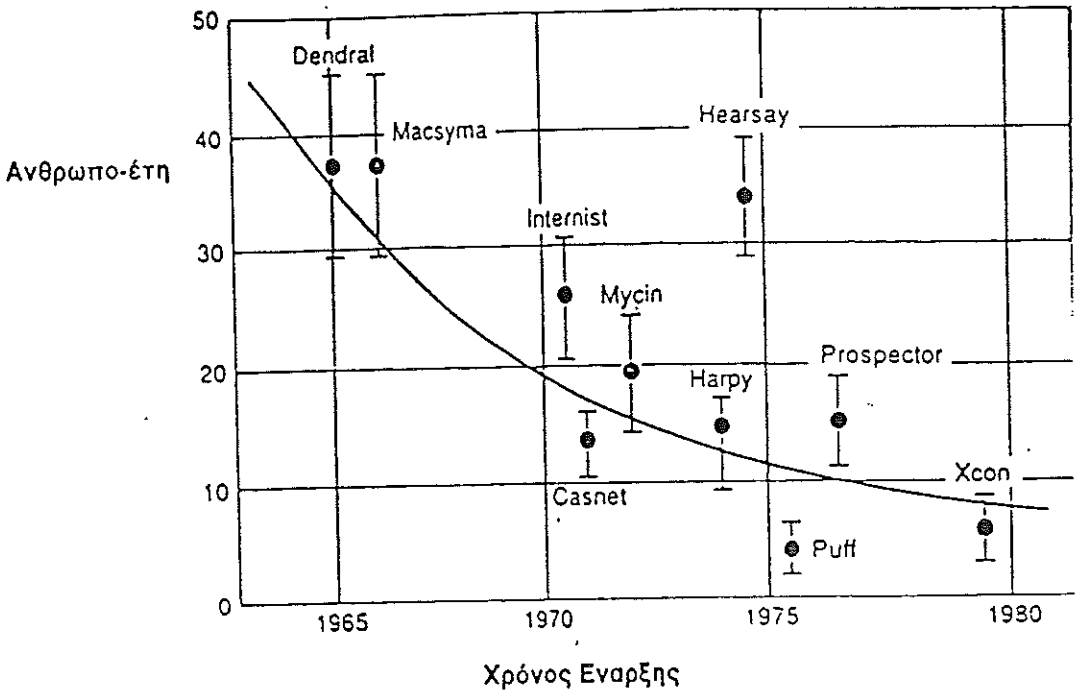
5. Η Διαδικασία Ανάπτυξης ενός Εμπειρου Συστήματος

Η διαδικασία ανάπτυξης ενός Ε.Σ περιλαμβάνει τα παρακάτω στάδια:

- Προσδιορισμός Προβλήματος (Identification)
- Καθορισμός Εννοιών (Conceptualization)
- Τυποποίηση (Formalization)
- Υλοποίηση (Implementation)
- Ελεγχος (Testing)



σχ. 5 : Η ιστορία ανάπτυξης μερικών γνωστών Ε.Σ.



σχ. 6 : Ο χρόνος που απαιτήθηκε για την υλοποίηση μερικών γνωστών Ε.Σ. (σε ανθρωπο-έτη εργασίας)

```
problem(spark_plugs) :-
     engine_gets_gas(yes),
     find_if(engine_turns_over, yes).

problem(battery) :-
     find_if(engine_turns_over, no),
     find_if(lights_come_on, no).

problem(cables) :-
     find_if(engine_turns_over, no),
     find_if(lights_come_on, no).

problem(starter) :-
     find_if(engine_turns_over, no),
     find_if(lights_come_on, yes).

engine_gets_gas(yes) :-
     find_if(gas_in_tank, yes),
     find_if(gas_in_carburator, yes).

find_if(Question,Answer) :-
     P =.. [Question,X],
     P , !,
     Answer=X.

find_if(Question,Answer) :-
     write(Question), write(' ? (yes/no)  --> '),
     read(Y), nl,
     NP =.. [Question,Y],
     assert(NP),
     Answer = Y.

start :- clear_all([ engine_turns_over(_), /* Καθαρισμός */
        lights_come_on(_),                  /* της μνήμης  */
        gas_in_tank(_),                     /* εργασίας    */
        gas_in_carburator(_) ]).

clear_all([]).
clear_all([H|T]) :- retract_all(H), clear_all(T).

retract_all(H) :- retract(H), fail.
retract_all(_).
```

σχ. .7 : Οι κανόνες Εμπειρίας για το πρόβλημα του αυτοκινήτου σε PROLOG

6. PROLOG και Εμπειρα Συστήματα

Η γλώσσα PROLOG διαθέτει χαρακτηριστικά που την καθιστούν κατάλληλη να χρησιμοποιηθεί σαν εργαλείο για την ανάπτυξη Ε.Σ.

Ο τρόπος ορισμού των φράσεων που διαθέτει προσφέρεται για την κατ' ευθείαν αναπαράσταση της γνώσης με μορφή κανόνων παραγωγής (*production rules*). Επίσης αποτελεί ένα κοινό φορμαλισμό για την αναπαράσταση της γνώσης με διάφορους άλλους τρόπους όπως είναι τα σημασιολογικά δίκτυα (*semantic networks*), τα πλαίσια (*frames*) και τα αντικείμενα (*objects*).

Ενα μεγάλο μέρος της μηχανής επαγωγής παρέχεται αυτόματα από την PROLOG, καθώς ο μηχανισμός εκτέλεσης που διαθέτει είναι ένας μηχανισμός επαγωγής συμπερασμάτων, ανάλογος με το μηχανισμό ανάστροφης συλλογιστικής (*backward reasoning*) που χρησιμποποιείται από τα Ε.Σ. που βασίζονται σε κανόνες παραγωγής.

Για τους παραπάνω λόγους η PROLOG είναι πολύ κοντά στο να θεωρηθεί σαν ένα Κέλυφος Ε.Σ. Στους λόγους αυτούς προστίθεται και η ικανότητα της να χρησιμοποιείται σαν μετα-γλώσσα. Με τη χρήση των μετα-διερμηνέων που εύκολα κατασκευάζονται, μπορούν να υλοποιηθούν πολλά επι μέρους χαρακτηριστικά των Ε.Σ., όπως είναι (α) η αλληλοεπικοινωνία χρήστη και συστήματος (β) η επεξήγηση των συλλογισμών, (γ) ο μηχανισμός αβέβαιης συλλογιστικής κ.λπ.

Στο σχήμα .7 δίνεται το πρόγραμμα της PROLOG που αντιστοιχεί στο πρόβλημα της εκίνησης του αυτοκινήτου που αναφέρθηκε στην παράγραφο .4, ενώ στο σχήμα 8 δίνονται τρείς διδικασίες εκτέλεσης για αντίστοιχες διαδικασίες διάγνωσης της βλάβης.

```
⇨ ?- start,problem(X).

gas_in_tank ? (yes/no)  --> yes.
gas_in_carburator ? (yes/no)  --> yes.
engine_turns_over ? (yes/no)  --> no.
lights_come_on ? (yes/no)  --> yes.

   X = starter

⇨ ?- start, problem(X).

gas_in_tank ? (yes/no)  --> yes.
gas_in_carburator ? (yes/no)  --> yes.
engine_turns_over ? (yes/no)  --> no.
lights_come_on ? (yes/no)  --> no.

   X = battery ;
   X = cables ;
   no

⇨ ?- start, problem(X).

gas_in_tank ? (yes/no)  --> yes.
gas_in_carburator ? (yes/no)  --> yes.
engine_turns_over ? (yes/no)  --> yes.

   X = spark_plugs ;
   no
```

σχ. 8 : Εκτέλεση του προγράμματος του σχήματος 7