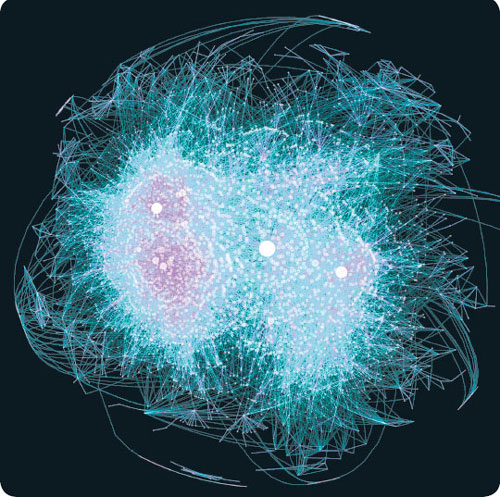
**ΑΠΑΛΛΑΚΤΙΚΗ ΕΡΓΑΣΙΑ ΣΤΟΝ ΣΗΜΑΣΙΟΛΟΓΙΚΟ ΙΣΤΟ.**



ΑΝΑΘΕΣΗ:ΒΑΣΙΛΕΙΑΔΗΣ ΝΙΚΟΛΑΟΣ

ΕΠΙΒΛΕΨΗ:ΚΟΝΤΟΠΟΥΛΟΣ ΣΤΡΑΤΟΣ

ΥΛΟΠΟΙΗΣΗ:ΧΑΤΖΗΒΑΓΙΑΣ ΧΡΗΣΤΟΣ

**Περιεχόμενα**

1.Εισαγωγή............................................................3

2.Εργαλεία υλοποίησης.........................................4

3.Η γλώσσα Rule ML............................................5

4.Απαιτήσεις..........................................................7

5.Περιγραφή του Αλγορίθμου...............................7

6.Επεξήγηση των συμβόλων του γραφου.............8

7.Παραδείγματα Εκτέλεσης..................................9

Α.Πρώτο παράδειγμα.....................................9

Β.Δεύτερο παράδειγμα...................................10

8.Επεξήγηση του κώδικα.....................................13

**Α.Οι κλάσεις που υλοποιήθηκαν για την δημιουργία της εφαρμογής.**

**Β.Συνάρτησεις για την δημιουργία της web εφαρμογής.**

**9.Οδηγίες Εγκατάστασης.....................................17**

**10.Επισκόπηση Web Εφαρμογής.........................18**

**11.Επισκόπηση Window Εφαρμογής...................19**

12.Βιβλιογραφία..................................................20

1. Εισαγωγή.

Το αντικείμενο της εργασίας είναι η δημιουργία μιας web εφαρμογής η οποία διαβάζει ένα XML έγγραφο και δημιουργεί έναν γράφο Το XML έγγραφο το οποίο επεξεργάζεται η εφαρμογή θα πρέπει να είναι μιας συγκεκριμένης γλώσσας σήμανσης, της RuleML. Η RuleML όπως υποδεικνύει το όνομα της είναι μια γλώσσα η οποία εκφράζει κανόνες σε σύνταξη XML. Ο στόχος της εργασίας είναι η ανάλυση των κανόνων και η δημιουργία ενός γράφου που θα παρουσιάζει τους κανόνες σαν κόμβους και τις συνδέσεις μεταξύ των κόμβων. Μια σύνδεση μεταξύ των κόμβων σημαίνει ότι οι κανόνες που αντιστοιχούν στους κόμβους αυτούς έχουν εξάρτηση μεταξύ τους.

Οι Κινέζοι λέγανε ότι μια εικόνα είναι όσο χίλιες λέξεις γι’αύτο και η δημιουργία μια εικόνας που παρουσιάζει έναν γράφο εξαρτήσεων μεταξύ τους μπορεί να χρησιμεύσει στην εξαγωγή χρήσιμων συμπερασμάτων για τις εξαρτήσεις που μπορεί να έχει ένας κανόνας με κάποιους άλλους. Έχοντας έναν γράφο μπορεί εύκολα κάποιος να δει αν ένας κανόνας εξαρτάται από την ενεργοποίηση ενός άλλου και με ποιον τρόπο, κατ επέκταση μπορεί να καταλάβει από την αλληλουχία των συνδέσεων ποιοι κανόνες μπορούν με την ενεργοποίησή τους να αποτελέσουν ένα “ντόμινο” ενεργοποιήσεων άλλων κανόνων.

Σε ένα μικρό έγγραφο μερικών κανόνων αυτό μπορεί να είναι εφικτό ακόμα και με την ανάγνωση τους, όμως σε περίπτωση που οι κανόνες είναι δεκάδες η εκατοντάδες τότε είναι δύσκολη η πρώτη ανάγνωση του έγγραφου ακόμα και κατανόηση του ύφους των κανόνων. Γι’αύτο το λόγο είναι επιβεβλημένη η οπτικοποίηση των εξαρτήσεων των κανόνων σε έναν κατευθυνόμενο γράφο εκθέτοντας όσο το δυνατόν περισσότερη πληροφορία στον γράφο ο οποίος θα παραχθεί. Ο γράφος θα πρέπει να είναι εύκολα αναγνώσιμος, αυτό σημαίνει ότι οι συνδέσεις δε θα πρέπει να μπλέκονται μεταξύ τους όπως επίσης και οι επεξηγήσεις των συνδέσεων θα πρέπει είναι ξεκάθαρο σε ποια σύνδεση ανήκουν.

2. Εργαλεία Yλοποίησης.

Για την ανάπτυξη του κώδικα που δημιουργεί το Visualization χρησιμοποιήθηκε η γλώσσα προγραμματισμού Python. Η **Python** δημιουργήθηκε από τον Ολλανδό Γκουίντο βαν Ρόσσουμ (Guido van Rossum) το 1990. Ο κύριος στόχος της είναι η αναγνωσιμότητα του κώδικά της και η ευκολία χρήσης της. Διακρίνεται λόγω του ότι έχει πολλές βιβλιοθήκες που διευκολύνουν ιδιαίτερα αρκετές συνηθισμένες εργασίες και για την ταχύτητα εκμάθησης της. Οι λόγοι για τους οποίους επιλέχθηκε η Python είναι οι εξής**:**

1.Η Python αναπτύσσεται ως ανοιχτό λογισμικό (open source) και η διαχείρισή της γίνεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation.Ο κώδικας διανέμεται με την άδεια Python Software Foundation License η οποία είναι συμβατή με την [*GPL*](http://el.wikipedia.org/wiki/GNU_General_Public_License).

2.Η Python είναι αντικειμενοστρεφής γλώσσα.

3.Η Python χρησιμοποιεί μεταγλωττιστή (compiler) για την δημιουργία του εκτελέσιμου κώδικα που σημαίνει ότι τα λάθη τα οποία τυχόν προκύψουν στον κώδικα είναι πολύ πιο εύκολα διαχειρίσιμα.

4.Η Python μπορεί να τρέξει και από web browsers , γεγονός που ευνοούσε ακόμα περισσότερο την χρήση της στην παρούσα εργασία, γιατί η εργασία επέβαλλε η εφαρμογή να τρέχει από web browser.

5.Η python έχει πολλές βιβλιοθήκες που ευνοούν την ανάπτυξη διάφορων εφαρμογών. Λόγου χάρη για την παρούσα εφαρμογή χρειάστηκε να χρησιμοποιηθούν τέσσερις βιβλιοθήκες.

* Βιβλιοθήκη για την ανάλυση των XML αρχείων.
* Βιβλιοθήκη για την δημιουργία του γράφου.
* Βιβλιοθήκη για την έναρξη του python web server.
* Βιβλιοθήκη για την δημιουργία παραθυρικής εφαρμογής.

Αναλυτικά κάποιες λεπτομέρειες για τις βιβλιοθήκες που χρησιμοποιήθηκαν

1.Η πρώτη και ίσως πιο ζωτική για την λειτουργιά της εφαρμογής είναι η βιβλιοθήκη pydot.Το pydot ( <http://code.google.com/p/pydot> ) είναι ένα python interface για το graphviz.

2.Το graphviz ( <http://www.graphviz.org/> ) είναι ένα πακέτο εργαλείων ανοιχτού κωδικά για την δημιουργία γράφων Προσφέρει πλήθος δυνατοτήτων για την δημιουργία γράφων όπως διάφορα σχήματα και χρώματα που μπορούν να πάρουν οι κόμβοι, διάφοροι τύποι γραμμών και διάφοροι τύποι γραμματοσειρών για την δημιουργία κατευθυνόμενων και μη γράφων με διάφορες επιγραφές πάνω στους κόμβους και στις γραμμές τους.

3.Η δεύτερη βιβλιοθήκη η οποία χρησιμοποιήθηκε είναι η βιβλιοθήκη bottle ( <http://bottlepy.org/docs/dev> ). Η bottle αποτελεί περισσότερο ένα web framework το οποίο τρέχει έναν server σε όποια port ορίσει ο προγραμματιστής. Ο προγραμματιστής μπορεί να γράψει ρουτίνες στις οποίες ορίζει τον html κώδικα της σελίδας δυναμικά η στατικά. Ενώ παράλληλα μπορεί να έχει όλες τις δυνατότητες της γλώσσας Python.

4.Η τρίτη βιβλιοθήκη και αυτή ζωτικής σημασίας είναι η minidom ( <http://code.google.com/p/minidom/> ). Η βιβλιοθήκη minidom προσφέρει λειτουργίες parsing και την εγγραφή ενός XML έγγραφου.

5.Η τέταρτη βιβλιοθήκη που χρησιμοποιήθηκε είναι η wxPython, η οποία δίνει την δυνατότητα σε έναν προγραμματιστή να δημιουργήσει υψηλού επιπέδου παραθυρικές εφαρμογές. Η χρήση της wxPython δεν απαιτούνταν από τις λειτουργιές της εργασίας, παρόλα αυτά για την ευκολία ανάπτυξης της κρίθηκε σκόπιμο αρχικά η εφαρμογή να αναπτύχθει σε παραθυρικό περιβάλλον και μετά να μεταφερθεί και σε web browser.

3.Η γλώσσα Rule ML.

**Η Rule-ML είναι μια γλώσσα σήμανσης που αναπτύχθηκε για να εκφράσει XML κανόνες τύπου “εάν ισχύει αυτό τότε ισχύει αυτό”. Σκοπός της Rule-ML είναι η επαγωγή , ο μετασχηματισμός και επανασύνταξη κανόνων.**

**Γλώσσες βασισμένες στην Rule-ML είναι οι έξης:**

* **Mathematica Markup Language (Math-ML) → είναι μια γλώσσα σήμανσης η οποία είναι ιδανική περισσότερο για να περιγραφεί μαθηματικές εξισώσεις παρά κανόνες.**
* **Predictive Model Markup Language (**[**PMML**](http://en.wikipedia.org/wiki/Predictive_Model_Markup_Language)**) →** **είναι μια γλώσσα σήμανσης η οποία μπορεί να καθορίσει διάφορα μοντέλα για την άντληση δεδομένων, συμπεριλαμβανομένων και των κανόνων συσχέτισης.**
* **Attribute Grammars in XML (**[**AG-markup**](http://en.wikipedia.org/w/index.php?title=AG-markup&action=edit&redlink=1)**) → είναι μια γλώσσα σήμανσης η οποία περιγράφει γραμματικές ιδιότητες.**
* **Extensible Stylesheet Language Transformations (**[XSLT](http://en.wikipedia.org/wiki/XSLT)**) → είναι μια γλώσσα σήμανσης η οποία έχει ένα σύνολο απο αυστηρούς κανόνες και χρησιμεύει στην μετατροπή εγγράφων XML σε άλλα έγγραφα κειμένου.**

**Ένας κάνονας Rule-ML αποτελείτε από τα δυο μέρη 1.την κεφαλή και 2.το σώμα του κανόνα.**

**Η κεφαλή του κανόνα πρέπει περιέχει μια τουλάχιστον μια δήλωση και το σώμα μια δήλωση. Η κάθε δήλωση μπορεί να αποτελείτε από πολλαπλά Variables , ένα Relation και πολλαπλά Individuals.Μια δήλωση χαρακτηρίζετε με το tag <Atom> και τα Relation Variable και Individual τα tags <Rel> <Var> και <Ind> αντίστοιχα.**

Κάθε κανόνας στην Rule-ML περικλείεται από το tag <Implies> ενώ το σώμα και η κεφαλή αντίστοιχα από τα tags <if> και <then>.

4.Απαιτήσεις.

1. Δημιουργία εφαρμογής που εκτελείτε μέσω web browser.
2. Η εφαρμογή αυτή θα πρέπει να αναλύει ένα RULE-ML έγγραφο.
3. Δημιουργία γράφου με κόμβους τους κανόνες της RULE-ML και συνδέσεις τις εξαρτήσεις.
4. Αποθήκευσης του γράφου σε μορφή αρχείου εικόνας.
5. Δημιουργία παραθυρικής διεπαφής.

5.Περιγραφή του Αλγορίθμου.

1. Συλλογή των δηλώσεων από όλους τους κανόνες του εγγράφου.
2. Απόδοση μοναδικού σειριακού αριθμού σε κάθε κανόνα. Και αποθήκευση του αριθμού στα εσωτερικά δεδομένα της δήλωσης.
3. Αποθήκευση των δηλώσεων σε δυο διαφορετικά σύνολα.Στο πρώτο σύνολο αποθηκεύονται οι δηλώσεις που υπάρχουν στα σώματα των κανόνων και στο δεύτερο σύνολο αποθηκεύονται οι δηλώσεις που υπάρχουν στις κεφαλές.
4. Οι κανόνες που ανήκουν των σωμάτων διατρέχονται σειριακά και για τον καθένα γίνετε έλεγχος αν υπάρχει κάποιος κανόνας ο οποίος είναι ίδιος με κάνονα που βρίσκετε στην κεφαλή άλλου κανόνα
5. Για κάθε ταυτοποίηση κανόνων μεταξύ κεφαλής και σώματος δημιουργούνται στον γράφο 2 κόμβοι και μια σύνδεση μεταξύ των, με κατεύθυνση από τον κανόνα του οποίου η δήλωση βρίσκετε στο σώμα επαληθεύει την δήλωση της κεφαλής του κανόνα προς τον οποίον κατευθύνετε η σύνδεση.
6. Εάν στην κεφαλή ενός κανόνα υπάρχουν παραπάνω από μια δηλώσεις τότε η εξάρτηση θα είναι μερική ενώ αν έχει έναν κανόνα τότε η εξάρτιση είναι ολική.

6.Επεξήγηση των συμβόλων του γραφου.

* + 1. Ο Κανόνας ισοδυναμεί με κόμβο στον γράφο και απεικονίζετε ως εξής.
    2. Η ολική εξάρτηση απεικονίζετε με σύνδεση που έχει διπλή ακίδα στο βέλος της.
    3. Η μερική εξάρτηση απεικονίζετε με σύνδεση που έχει μονή ακίδα στο βέλος της.
    4. Η σύνδεση αντιπροσωπεύει μια εξάρτηση, η εξάρτηση αυτή οφείλετε σε μια δήλωση αυτή η δήλωση αποτελεί και την ετικέτα της σύνδεσης – εξάρτησης.

7.Παραδείγματα Εκτέλεσης.

Α.Πρώτο παράδειγμα.

Στο πρώτο παράδειγμα θα μοντελοποιηθούν δυο κανόνες. Οι κανόνες αυτοί έχουν ποδοσφαιρικό χαρακτήρα και προσπαθούν να μοντελοποιήσουν ένα πολύ μικρό υποσύνολο των κανόνων ενός ποδοσφαιρικού αγώνα.

1. Rule 0 → Ο πρώτος κανόνας υποδεικνύει ότι εάν ένας παίκτης πάρει κόκκινη κάρτα τότε αποβάλλεται.
2. Rule 1 → Ο δεύτερος κανόνας υποδεικνύει εάν ένας παίκτης πάρει δεύτερη κίτρινη κάρτα τότε παίρνει κόκκινη.

Είναι εύκολα αντιληπτό ότι ο Rule 0 έχει ολική εξάρτηση από τον Rule 1 και η εξήγηση είναι ότι εάν εκπληρωθεί ο Rule 1 δηλαδή ένας παίκτης πάρει κόκκινη κάρτα τότε αυτόματα εκπληρώνετε και ο Rule 0 που επιβάλει την έξοδο του παίκτη εάν αυτός πάρει κόκκινη κάρτα.

Β.Δεύτερο παράδειγμα.

Το δεύτερο παράδειγμα είναι μια επέκταση των κανόνων του πρώτου παραδείγματος.

1)Rule 2 → Ο τρίτος κανόνας υποδεικνύει ότι εάν ένας παίκτης κάνει σκληρό φάουλ, τότε παίρνει κίτρινη κάρτα.

2)Rule 3 → Ο τέταρτος κανόνας υποδεικνύει ότι εάν ένας παίκτης κάνει δολοφονικό φάουλ, τότε παίρνει κόκκινη κάρτα.

3)Rule 4 → Ο πέμπτος κανόνας υποδεικνύει ότι εάν ένας παίκτης κάνει σκληρό φάουλ, και έχει κίτρινη κάρτα, τότε παίρνει δεύτερη κίτρινη κάρτα.

4)Rule 5 → Ο έκτος κανόνας υποδεικνύει ότι εάν ένας παίκτης κάνει θέατρο φάουλ, τότε παίρνει κίτρινη κάρτα.

Παρατηρήσεις:

1.Παρατηρουμε ότι ο Rule 4 εξαρτάται μερικώς από τους Rule 2 και Rule 5.Οι Rules 2 και Rule 5 υποδηλώνουν ότι κάποιος παίκτης θα πάρει κίτρινη κάρτα αν υποπέσει σε ανάλογο σφάλμα. Όταν λοιπόν πάρει την κίτρινη κάρτα τότε εκπληρώνετε ένα μέρος του Rule 4.

2. Παρατηρούμε ότι ο Rule 1 εξαρτάται ολικώς από τον Rule 4.Ο Rule 4 υποδηλώνει ότι κάποιος παίκτης θα πάρει δεύτερη κίτρινη κάρτα αν έχει ήδη μια και υποπέσει σε σκληρό φάουλ . Όταν λοιπόν πάρει την δεύτερη κίτρινη κάρτα τότε εκπληρώνετε ο Rule 1.

3.Παρατηρούμε ότι ο Rule 0 εξαρτάται ολικώς από τους Rule 3 και Rule 1.Οι Rules 1 και Rule 3 υποδηλώνουν ότι κάποιος παίκτης θα πάρει κόκκινη κάρτα αν υποπέσει σε ανάλογο σφάλμα η πάρει δεύτερη κίτρινη κάρτα Όταν λοιπόν πάρει την κόκκινη κάρτα τότε εκπληρώνετε ο Rule 0 και ο παίκτης αποβάλλετε.

8. Επεξήγηση του Κώδικα.

Α.Οι κλάσεις που υλοποιήθηκαν για την δημιουργία της εφαρμογής:

1. Η κλάση Variable → είναι η κλάση που αντιπροσωπεύει το Var που μπορεί να περιέχει ένα Atom.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Variable τα οποία είναι η τιμή του element μαζί με τα tags π.χ. <Var>5 percent</Var>.
   2. Methods →
      1. getData → επιστρέφει τα εσωτερικά data του Variable.
      2. isSame → δέχεται σαν όρισμα ένα άλλο Variable και το συγκρίνει με το παρών. instance και επιστρέφει true εάν έχουν ίδια data διαφορετικά επιστρέφει false.
2. H κλάση Individual → είναι η κλάση που αντιπροσωπεύει το Ind που μπορεί να περιέχει ένα Atom.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Variable τα οποία είναι η τιμή του element μαζί με τα tags π.χ. <Ind>5 percent</Ind>.
   2. Methods →
      1. getData → επιστρέφει τα εσωτερικά data του Individual.
      2. isSame → δέχεται σαν όρισμα ένα άλλο Individual και το συγκρίνει με το παρών instance και επιστρέφει true εάν έχουν ίδια data διαφορετικά επιστρέφει false.
3. H κλάση Relation → είναι η κλάση που αντιπροσωπεύει το Rel που μπορεί να περιέχει ένα Atom.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Relation τα οποία είναι η τιμή του element μαζί με τα tags π.χ. <Rel>5 percent</Rel>.
   2. Methods →
      1. getData → επιστρέφει τα εσωτερικά data του Relation.
      2. isSame → δέχεται σαν όρισμα ένα άλλο Relation και το συγκρίνει με το παρών instance και επιστρέφει true εάν έχουν ίδια data διαφορετικά επιστρέφει false.
4. H κλάση Atom: είναι η κλάση η οποία αντιπροσωπεύει την προϋπόθεση αν είναι στο if μέρος του κανόνα και την ενέργεια όταν βρίσκετε στο then μέρος του κανόνα.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Atom τα οποία είναι η τιμή του element μαζί με τα tags.
      2. m\_vars → Set στο οποίο αποθηκεύονται τα instances των Variables όταν καλεστεί η μέθοδος parse του Atom.
      3. m\_inds → Set στο οποίο αποθηκεύονται τα instances των Individuals όταν καλεστεί η μέθοδος parse του Atom.
      4. m\_rel → αποθηκεύεται το instance του Relation όταν καλεστεί η method parse του Atom.
      5. m\_ruleId → είναι το μοναδικό id που δίνει η εφαρμογή στους σε κάθε κανόνα και η μεταβλητή αυτή συγκρατεί το id του κανόνα στον οποίο ανήκει.
      6. m\_rule → αποθηκεύει ένα αντίγραφο του instance του κανόνα στον οποίο βρίσκετε το παρών Atom.
   2. Methods →
      1. getVars → επιστρέφει ένα αντίγραφο του Set των Variables m\_vars του Atom.
      2. getInds → επιστρέφει ένα αντίγραφο του Set των Individuals m\_inds του Atom.
      3. getRel → επιστρέφει το Relation m\_rel του Atom.
      4. setRuleId → παίρνει σαν όρισμα έναν αριθμό και τον αναθέτει στην μεταβλητή m\_ruleId.
      5. ruleId → επιστρέφει την μεταβλητή m\_ruleId.
      6. setRule → παίρνει σαν όρισμα ένα instance του κανόνα στον οποίο βρίσκετε και το αντιγραφεί στο m\_rule.
      7. getRule → επιστρέφει ένα αντίγραφο του m\_rule.
      8. hasSameInds → παίρνει σαν όρισμα ένα Atom και ελέγχει το Set m\_inds αν είναι ίδιος με το Set του παρόντα Atom, εάν ισχύει τότε επιστρέφει true, αλλιώς επιστρέφει false.
      9. hasSameVars → παίρνει σαν όρισμα ένα Atom και ελέγχει το Set m\_vars αν είναι ίδιος με το Set του παρόντα Atom, εάν ισχύει τότε επιστρέφει true, αλλιώς επιστρέφει false.
      10. hasSameNumVars → παίρνει σαν όρισμα ένα Atom και ελέγχει αν το Set m\_vars έχει ίδιο αριθμό εγγραφών με το Set m\_vars του παρόντος Atom.
      11. isSame → παίρνει σαν όρισμα ένα Atom και ελέγχει αν έχει τα ίδια m\_vars , m\_inds και το ίδιο m\_rel.Αν όλα αυτά ισχύουν τότε επιστρέφει true υποδεικνύοντας ότι το Atom στην είσοδο είναι ίδιο με το παρών Atom.
      12. parse → κάνει το parsing του Atom και αποθηκεύει τα m\_vars , m\_inds και m\_rel.
5. H κλάση Head → είναι η κλάση που αντιπροσωπεύει το Head που μπορεί να περιέχει ένας Rule.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Head τα οποία είναι η τιμή του element μαζί με τα tags.
      2. m\_atoms → Set στο οποίο αποθηκεύονται τα instances των Atoms όταν καλεστεί η μέθοδος parse του Head.
      3. m\_rule → αποθηκεύει ένα αντίγραφο του instance του κάνονα στον οποίο βρίσκετε το παρών Head.
      4. m\_ruleId → είναι το μοναδικό id που δίνει η εφαρμογή στους σε κάθε κάνονα και η μεταβλητή αυτή συγκρατεί το id του κανόνα στον οποίο ανήκει.
   2. Methods →
      1. setRuleId → παίρνει σαν όρισμα έναν αριθμό και τον αναθέτει στην μεταβλητή m\_ruleId.
      2. ruleId → επιστρέφει την μεταβλητή m\_ruleId.
      3. setRule → παίρνει σαν όρισμα ένα instance του κανόνα στον οποίο βρίσκετε και το αντιγράφει στο m\_rule.
      4. numAtoms → επιστρέφει τον αριθμό των Atoms τα οποία βρίσκονται στο Head.
      5. Εάν στην κεφαλή ενός κανόνα υπάρχουν παραπάνω από μια δηλώσεις τότε η εξάρτηση θα είναι μερική ενώ αν έχει έναν κανόνα τότε η εξάρτιση είναι ολική. parse → κάνει το parsing του Head και αποθηκεύει τα m\_atoms με τα Atoms τα οποία βρίσκονται σε αυτό.
6. H κλάση Body → είναι η κλάση που αντιπροσωπεύει το Body που μπορεί να περιέχει ένας Rule.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Body τα οποία είναι η τιμή του element μαζί με τα tags.
      2. m\_atoms → Set στο οποίο αποθηκεύονται τα instances των Atoms όταν καλεστεί η μέθοδος parse του Body.
      3. m\_rule → αποθηκεύει ένα αντίγραφο του instance του κανόνα στον οποίο βρίσκετε το παρών Body.
      4. m\_ruleId → είναι το μοναδικό id που δίνει η εφαρμογή στους σε κάθε κανόνα και η μεταβλητή αυτή συγκρατεί το id του κανόνα στον οποίο ανήκει.
   2. Methods →
      1. setRuleId → παίρνει σαν όρισμα έναν αριθμό και τον αναθέτει στην μεταβλητή m\_ruleId.
      2. ruleId → επιστρέφει την μεταβλητή m\_ruleId.
      3. setRule → παίρνει σαν όρισμα ένα instance του κανόνα στον οποίο βρίσκετε και το αντιγράφει στο m\_rule.
      4. parse → κάνει το parsing του Body και αποθηκεύει τα m\_atoms με τα Atoms τα οποία βρίσκονται σε αυτό.
7. H κλάση Rule → είναι η κλάση που αντιπροσωπεύει τον Rule (κανόνα) , ένας από τους πολλούς που μπορεί να περιέχονται σε στο αρχείο.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Rule τα οποία είναι η τιμή του element μαζί με τα tags.
      2. m\_id → αποθηκεύει το id του κανόνα.
      3. m\_head → αποθηκεύει το instance του Head που υπάρχει στον παρών Rule.
      4. m\_body → αποθηκεύει το instance του Body που υπάρχει στον παρών Rule.
   2. Methods →
      1. setId → παίρνει σαν όρισμα έναν integer και τον αναθέτει στο m\_id.
      2. id → επιστρέφει το m\_id του Rule.
      3. numHeadAtoms → επιστρέφει το πλήθος των Atoms που υπάρχουν στο Head του παρόντος κανόνα.
      4. parse → κάνει το parsing του Rule και αποθηκεύει τα m\_head και m\_body με τα Atoms τα οποία βρίσκονται στο καθένα.
8. H κλάση Rule\_Library → είναι η κλάση που αντιπροσωπεύει την βιβλιοθήκη των Rules που βρέθηκαν στο αρχείο.
   1. Data →
      1. m\_data → αποθηκεύονται τα δεδομένα του Rule τα οποία είναι η τιμή του element μαζί με τα tags.
      2. m\_rules → Set το οποίο αποθηκεύει τα instances των Rules που περιέχονται στο αρχείο.
      3. m\_headAtoms → Set το οποίο αποθηκεύει τα instances των Atoms τα οποία βρίσκονται σε Head τομέα ενός Rule.
      4. m\_bodyΒ.Συνάρτησεις για την δημιούργια της web εφαρμογήςAtoms → Set το οποίο αποθηκεύει τα instances των Atoms τα οποία βρίσκονται σε Body τομέα ενός Rule.
      5. m\_edges → αποθηκεύει τις συνδέσεις των κόμβων των γράφων όπως αυτές δημιουργούνται από την βιβλιοθήκη pydot.
   2. Methods →
      1. collectHeadAtoms → διατρέχει όλα τα m\_rules και από κάθε instance ενός Rule αποθηκεύει τα Atoms του Head τομέα στο m\_headAtoms.
      2. collectBodyAtoms → διατρέχει όλα τα m\_rules και από κάθε instance ενός Rule αποθηκεύει τα Atoms του Bodef upload\_form():dy τομέα στο m\_bodyAtoms.
      3. make\_graph → δημιουργεί την εικόνα στην οποία απεικονίζετε ο γράφος.
      4. make\_label → παίρνει σαν είσοδο ένα Atom και χρησιμοποιώντας τα m\_vars , m\_inds και m\_rel προσπαθεί να δημιουργήσει την ετικέτα που θα χαρακτηρίσει την σύνδεση των δύο κόμβων.
      5. Parse → κάνει parsing τους Rules, τους αποθηκεύει στο m\_rules και καλεί την make\_graph για να δημιουργήσει την εικόνα του γράφου.
9. H κλάση Parser είναι η κλάση που αναλύει το έγγραφο.
   1. Data →
      1. m\_data → περιέχει το κείμενο όλου του εγγράφου σαν ένα ενιαίο string.
      2. m\_file → αντιπροσωπεύει το instance του αρχείου στο οποίο φορτώνετε το έγγραφο xml.
      3. m\_library → είναι ένα instance του Rule\_Library.
   2. Methods →
      1. set\_data → παίρνει σαν είσοδο το αρχείο xml σαν ένα ενιαίο string.

Β.Συνάρτησεις για την δημιουργία της web εφαρμογής.

1. upload\_form → η συνάρτηση η οποία εχει την html σελίδα που δημιουργεί την φόρμα υποβολής του Rule-ML εγγράφου.
2. do\_upload\_file → η συνάρτηση η οποία λαμβάνει το έγγραφο με την μορφή path Rule-ML, και με την βοήθεια του parser που υλοποιήθηκε, δημιουργείτε η εικόνα με τον γράφο των εξαρτήσεων των κανόνων και αλλάζοντας σελίδα εμφανίζει την εικόνα στην νέα σελίδα.
3. do\_upload\_text → η συνάρτηση η οποία λαμβάνει το Rule-ML με την μορφή text, και με την βοήθεια του parser που υλοποιήθηκε, δημιούργητε η εικόνα με τον γράφο των εξαρτήσεων των κανόνων και αλλάζοντας σελίδα εμφανίζει την εικόνα στην νέα σελίδα.

9.Οδηγίες Εγκατάστασης.

1. Κατεβάζουμε την Python 3.2.2 από την ιστοσελίδα της [www.python.org/download/](http://www.python.org/download/).
2. Εγκαθιστούμε την Python 3.2.2 που μόλις κατεβάσαμε.
3. Κατεβάζουμε την GraphViz από την ιστοσελίδα της [www.graphviz.org/Download..php](http://www.graphviz.org/Download..php)
4. Εγκαθιστούμε την GraphViz που μόλις κατεβάσαμε. Για linux δίνουμε εντολή “sudo apt-get install graphviz”
5. Κατεβάζουμε την pydot και την pyparser.
6. Τα εγκαθιστούμε ως εξής:
   1. Ανοίγουμε το command prompt και πλοηγούμαστε στο φάκελο του pyparser.
   2. Εκεί δίνουμε την εντολή {PythonLocation}\python.exe setup.py install.
   3. Ανοίγουμε το command prompt και πλοηγούμαστε στο φάκελο του pydot.
   4. Εκει δίνουμε την εντολή {PythonLocation}\python.exe setup.py install.
7. Κλωνοποιούμε το repository της εφαρμογής που υπάρχει στο https://ironexmaiden@github.com/ironexmaiden/csd\_post\_sw.git με κάποιο εργαλείο του git πχ git-extensions.
8. Ανοίγουμε το command prompt και πλοηγούμαστε στο φάκελο του κλώνου της εφαρμογής.
9. Εκεί δίνουμε την εντολή {PythonLocation}\python.exe webApp.py.
10. Ανοίγουμε κάποιον Web Browser, και πηγαίνουμε στην ιστοσελίδα localhost:8888/ruleml.

10.Επισκόπηση Web Εφαρμογής.

1. Κουμπί επιλογης αρχείου Rule-ML.
2. Κουμπί καταχώρησης του αρχείου Rule-ML.
3. Περιοχή κειμένου Rule-ML.
4. Κουμπί καταχώρησης του κειμένου Rule-ML.

11.Επισκόπηση Window Εφαρμογής.

1. Μενού επιλογής αρχείου Rule-ML.
2. Κουμπί καταχώρησης του αρχείου Rule-ML.
3. Τοποθέσια αρχείου Rule-ML.

12.Βιβλιογραφία.

1.<http://en.wikipedia.org/wiki/RuleML>

2.The Rule Markup Initiative - [http://ruleml.org](http://ruleml.org/)

3.Python Programming Language - [http://python.org](http://python.org/)

4. Graphviz - Graph Visualization Software - [http://www.graphviz.org](http://www.graphviz.org/)

5.The RuleML Family of Web Rule Languages Harold Boley .

Institute for Information Technology – e-Business,

National Research Council of Canada,

Fredericton, NB, E3B 9W4, Canada

harold.boley AT nrc DOT gc DOT ca

6.Modelling Contracts Using RuleML .

Guido Governatori1

Antonino Rotolo2

School of ITEE, The University of Queensland, Brisbane, Australia

Email: guido@itee.uq.edu.au

2 CIRSFID and Law Faculty, University of Bologna, Bologna, Italy

7.Generating Graph Visualizations with pydot and Graphviz - <http://pythonhaven.wordpress.com/tag/pydot>